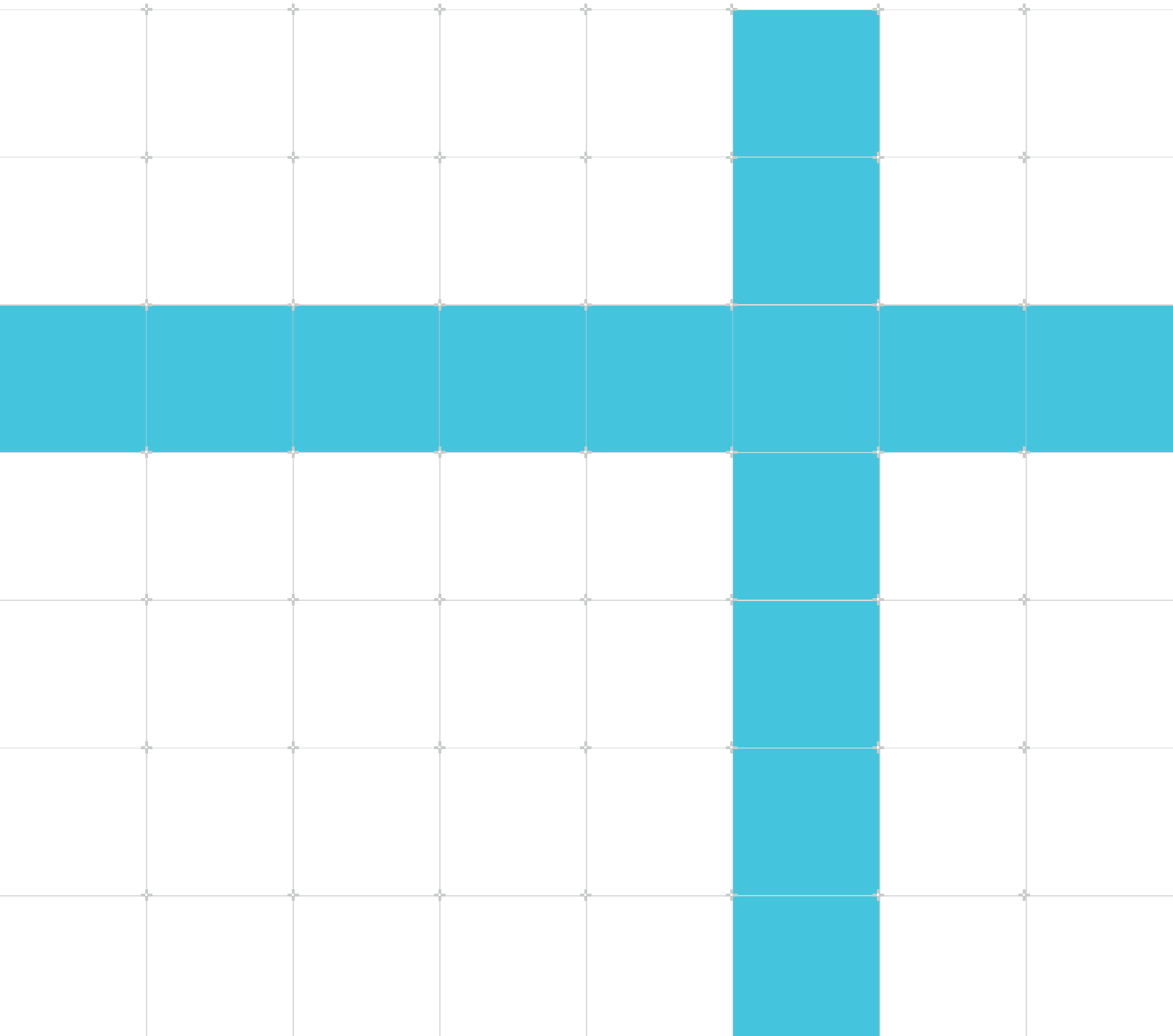
Arm® DynamIQ™ Shared Unit-110

Revision: r4p0

Technical Reference Manual

Non-Confidential

Issue 11
101381_0400_11_en



Arm® DynamIQ™ Shared Unit-110

Technical Reference Manual

Release Information

Document history

Issue	Date	Confidentiality	Change
0000-01	29 August 2019	Confidential	First alpha release for r0p0
0000-02	20 December 2019	Confidential	First beta release for r0p0
0000-03	31 March 2020	Confidential	First limited access release for r0p0
0100-04	29 May 2020	Confidential	First limited access release for r1p0
0200-05	21 August 2020	Confidential	First early access release for r2p0
0201-06	27 October 2020	Confidential	First early access release for r2p1
0300-07	25 February 2021	Confidential	First early access release for r3p0
0300-08	25 May 2021	Confidential	Second early access release for r3p0
0301-09	16 July 2021	Confidential	First early access release for r3p1
0400-10	17 June 2022	Confidential	First release for r4p0
0400-11	28 June 2022	Non-Confidential	Second release for r4p0

Proprietary Notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality Status

Product Status

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

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com.

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

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

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

To report offensive language in this document, email terms@arm.com.

Contents

1. Introduction

1.1 Product revision status

The r*x*p*y* identifier indicates the revision status of the product described in this manual, for example, r1p2, where:

r*x* Identifies the major revision of the product, for example, r1.

p*y* Identifies the minor revision or modification status of the product, for example, p2.

1.2 Intended audience

This manual is for system designers, system integrators, and programmers who are designing or programming a *System on Chip* (SoC) that uses a DynamIQ™ Shared Unit-110 along with an Arm core or cores.

1.3 Conventions

The following subsections describe conventions used in Arm documents.

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: developer.arm.com/glossary.

Typographic conventions

Convention	Use
italic	Citations.
bold	Interface elements, such as menu names. Signal names. Terms in descriptive lists, where appropriate.
`monospace`	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace bold`	Language keywords when used outside example code.

Convention	Use
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`<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	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, **IMPLEMENTATION DEFINED**, **IMPLEMENTATION SPECIFIC**, **UNKNOWN**, and **UNPREDICTABLE**.

Recommendations. Not following these recommendations might lead to system failure or damage.

Requirements for the system. Not following these requirements might result in system failure or damage.

Requirements for the system. Not following these requirements will result in system failure or damage.

An important piece of information that needs your attention.

A useful tip that might make it easier, better or faster to perform a task.

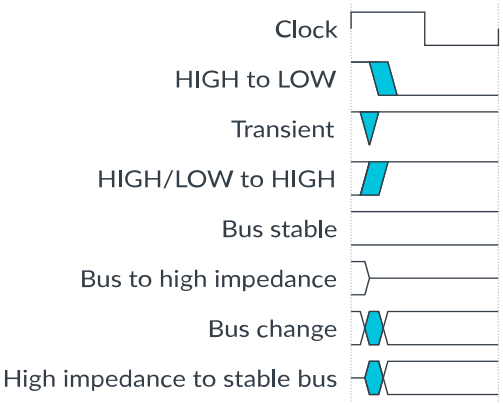
A reminder of something important that relates to the information you are reading.

Timing diagrams

The following figure explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

Figure 1-1: Key to timing diagram conventions



Signals

The signal conventions are:

Signal level

The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:

- HIGH for active-HIGH signals.
- LOW for active-LOW signals.

Lowercase n

At the start or end of a signal name, n denotes an active-LOW signal.

1.4 Additional reading

This document contains information that is specific to this product. See the following documents for other relevant information:

Table 1-2: Arm publications

Document Name	Document ID	Licensee only
AMBA® AXI and ACE Protocol Specification	IHI 0022	No
AMBA® APB Protocol Version 2.0 Specification	IHI 0024	No
AMBA® ATB Protocol Specification	IHI 0032	No
AMBA® 5 CHI Architecture Specification	IHI 0050	No

Document Name	Document ID	Licensee only
AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces	IHI 0068	No
Arm® Power Control System Architecture	DEN 0050	No
Arm® Power Policy Unit Architecture Specification	DEN 0051	No
Arm®Corelink® PCK-600 Power Control Kit Technical Reference Manual	101150	No
Arm® Architecture Reference Manual Armv8, for A-profile architecture	DDI 0487	No
Arm® Architecture Reference Manual Supplement Armv9, for Armv9-A architecture profile	DDI 0608	No
Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4	IHI 0069	No
Arm® CoreSight™ Architecture Specification v3.0	IHI 0029	No
Arm® CoreSight™ DAP-Lite2 Technical Reference Manual	100572	No
Arm® CoreSight™ SoC-600 Technical Reference Manual	100806	No
Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual	101088	No
Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual	101382	Yes
Arm® Architecture Reference Manual Supplement Memory System Resource Partitioning and Monitoring (MPAM), for Armv8-A	DDI 0598	No

Table 1-3: Other publications

Document ID	Document Name
-	-

Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of its documents when used with any other PDF reader.

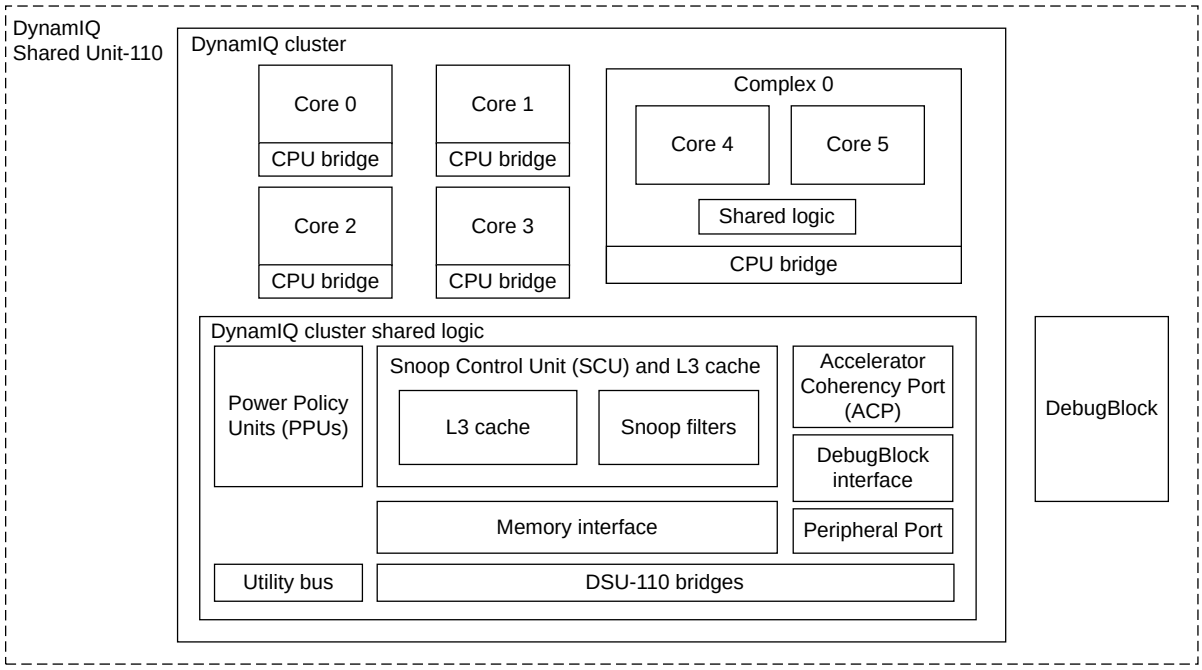
Adobe PDF reader products can be downloaded at http://www.adobe.com

2. The DynamIQ™ Shared Unit-110

The *DynamIQ™ Shared Unit-110* (DSU-110) provides a shared L3 memory system, snoop control and filtering, and other control logic to support a cluster of A-class architecture cores. The cluster is called the DSU-110 DynamIQ™ cluster. Also, all the external interfaces to the *System on Chip* (SoC) are provided through the DSU-110.

The following figure shows an example of a DSU-110-based cluster.

Figure 2-1: DSU-110 DynamIQ™ cluster



Note

In this book, the DSU-110 DynamIQ™ cluster is referred to as a cluster in cases where distinguishing between the DSU-110 DynamIQ™ cluster and DSU-110 is not important to the context.

A DSU-110 DynamIQ™ cluster consists of between one and 12 cores, with up to four different types of cores in the same cluster. Cores can be configured for various performance points during macrocell implementation and run at different frequencies and voltages.

The DSU-110 DynamIQ™ cluster also supports complexes where typically two cores are linked together and share logic. Examples of shared logic include a floating-point unit and an L2 cache. For more information on complexes, see 2.3.1 What is a complex? on page 25.

All cores in the DSU-110 DynamIQ™ cluster, including those in complexes, are coherently connected to an L3 memory system that includes an L3 cache and a *Snoop Control Unit* (SCU). The SCU maintains coherency between caches in the cores and the L3 cache, and includes a snoop filter to optimize coherency maintenance operations. The shared L3 cache simplifies process migration between the cores.

The DSU-110 DynamIQ™ cluster can be implemented with various power domains to target power performance levels. These power domains are managed through the *Power Policy Units* (PPUs). The DSU-110 DynamIQ™ cluster supports many mechanisms to reduce static and dynamic power dissipation. For example, placing the cores and L3 cache into retention and powering down parts of the L3 cache.

All the external interfaces including those to the cores are provided through the DSU-110 to the *System on Chip* (SoC). Main system transactions are supported through the memory interface which can be implemented as a coherent or non-coherent interface. A peripheral port is provided to support low latency access to external system components but also can be used as a non-coherent master interface. The *Accelerator Coherency Port* (ACP) provides coherent access for non-cached masters that need I/O coherency with the cluster. The utility bus is a memory-mapped port that provides a programming interface to the PPUs and some of the other system components.

A dedicated debug component, called the DebugBlock, forms part of the DSU-110 that provides the interface for debug capability. The DebugBlock is instanced as a separate unit for supporting debug over powerdown.

Finally, there are several asynchronous bridges automatically built in across the cluster to resynchronize timing across various clock domain boundaries.


Note

- For information on the behavior and features of your core, including whether your core is supported in a complex, see the *Technical Reference Manual* (TRM) for your core.
- For information on the DSU-110 macrocell implementation, see *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

2.1 DynamIQ™ Shared Unit-110 features

Some features in the *DynamIQ™ Shared Unit-110* (DSU-110) are fixed and some features are optional. You can configure optional features in the RTL during build time configuration, to meet your requirements.

Cache features

The DSU-110 has the following cache features:

- Optional unified 16-way set-associative L3 cache, configurable from 256KB to 16MB
- 64-byte cache lines

- L3 cache slice support, for improved bandwidth and cache RAM layout, up to eight slices supported
- L3 cache powerdown based either on cache slices or cache ways
- Cache partitioning support, compliant with *Memory System Resource Partitioning and Monitoring* (MPAM) architecture
- *Error Correcting Code* (ECC) protection on L3 cache RAM instances
- L3 cache system can be clocked at a rate synchronous to the external system interconnect or at integer multiples.

Coherency and snoop control

The DSU-110 has the following coherency and snoop control features:

- *Snoop Control Unit* (SCU) maintains coherency and consistency in the memory system internal to the cluster, and (optionally) external to the cluster.
- SCU includes a set of snoop filters, automatically sized, one for each cache slice.

Cluster features

The DSU-110 has the following cluster features:

- Support for Arm®v9.0-A architecture cores
- Support for up to four types of core, and a maximum of 12 cores in the cluster
- *Power Policy Units* (PPUs) providing autonomous power management of the L3 cache and the cores
- Support for cores running independently at different frequencies and voltages known as *Dynamic Voltage Frequency Scaling* (DVFS). For cores in a complex, DVFS is only possible for the whole complex not for individual cores.
- The DSU-110 has an internal transport mechanism that is responsible for all communication between components in the design. The topology of the transport is defined by the number of cores and number of L3 cache slices.

Interface features

The DSU-110 has the following interface features:

- Optional AMBA 5 CHI Issue E 256-bit coherent master bus interface, supports up to four *Coherent Hub Interface* (CHI) bus master ports.
- Optional AMBA AXI5 Issue H 256-bit non-coherent master interface, supports up to four *Advanced eXtensible Interface* (AXI) bus master ports.
- Optional 128-bit or 256-bit wide I/O-coherent *Accelerator Coherency Port* (ACP) based on AMBA ACE5-Lite.
- AMBA AXI5 utility bus providing programming interface to PPUs, and other system components.
- Optional peripheral port interface that is implemented as either an AXI 64-bit wide port, AXI 256-bit wide port, or CHI 256-bit wide port.

- Simplified system integration for interfaces such as debug and trace which are already in the correct clock domain at the output of the cluster.

Debug and trace features

The DSU-110 has the following debug and trace features:

- Debug-over-powerdown support
- CoreSight SoC-600 support for *Embedded Trace Extension* (ETE) and *Cross Trigger Interface* (CTI) for each core
- Optional CoreSight *Embedded Logic Analyzer* (ELA)-600 support

The ELA-600 is a separately licensable product.

Related information

2.2 DynamIQ Shared Unit-110 configuration parameters on page 20

2.2 DynamIQ™ Shared Unit-110 configuration parameters

You must configure the *DynamIQ™ Shared Unit-110* (DSU-110) RTL for your implementation requirements prior to hardware synthesis at build time configuration. Configuration for the DSU-110 is carried out together with configuration for the cores in your cluster.

For a complete list of the configuration parameters and guidelines, see *RTL configuration process* in *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual* (DSU-110 CIM).

The DSU-110 implementation options include:

Number of cores

You can configure the cluster to have between one and 12 cores. Each core instantiated within a complex counts towards the total number of cores in the cluster. This is an addition to any stand-alone (non-complexed) cores.

Core type

You can have a cluster that includes up to four different types of cores. See 2.3 Cluster configurations on page 22, for more information on the types of core that are supported.

Direct connect

You can configure the cluster for Direct connect memory system variant. For more information on Direct connect, see 2.3.2 L3 memory system variants on page 26.

L3 cache size

You can configure the L3 cache size to be:

- 0KB
- 256KB
- 512KB
- 1MB
- 1.5MB
- 2MB
- 3MB
- 4MB
- 6MB
- 8MB
- 12MB
- 16MB


Note

Setting the size of 0KB implements the DSU-110 without an L3 cache, see 2.3.2 L3 memory system variants on page 26.

L3 cache slices

You can configure the DSU-110 to have 1, 2, 4, or 8 cache slices. For more information on cache slices, see 7.6 Cache slices and power portions on page 105.

Transport configuration

The topology of the transport mechanism is automatically determined, dependent on the number of cores and L3 cache slices in your cluster. However, you can set transport data path width. For information on the DSU-110 transport, see *RTL configuration process* in *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

Memory interface configuration

You can configure the main memory interface to either use a CHI coherent interface or an AXI non-coherent interface. For either type of memory interface, you can configure up to four master ports.

ACP interface

You can include the *Accelerator Coherency Port* (ACP) interface and specify its size.

Peripheral port

You can include the peripheral port and specify its size. You can also configure it to be a non-coherent bus master.

SCU cache protection

You can configure the L3 cache and snoop filter RAMs with *Error Correcting Code* (ECC) support.

Timing closure

You can configure the L3 cache RAM timing latency and optionally include register slices.

ELA

Include support for integrating the CoreSight *Embedded Logic Analyzer* (ELA)-600 into the DSU-110.


Note

The ELA-600 is a separately licensable product.

Debug system address map

You can configure the DSU-110 to use a new address map on the Debug APB interface using a more efficient encoding than the older version.The ELA-600 is a separately licensable product.

2.3 Cluster configurations

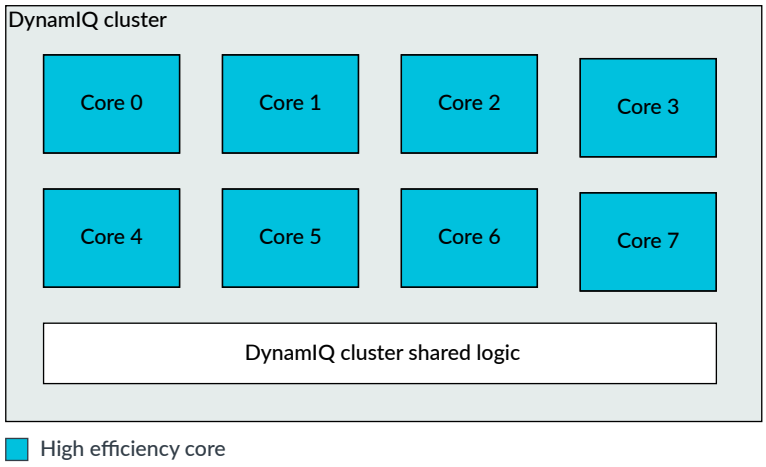
A cluster can be configured with up to four different types of cores in the same cluster. Each core type targeting different power efficiency and performance levels. This arrangement allows for an intermediate core that has an intermediate performance and efficiency level. The cluster also supports complexes.

A cluster can be configured in many arrangements. Examples of cluster arrangements are:

- One or more cores of the same type.
- Various arrangements of two types of cores. For example, one or more cores targeting either a high-performance level or a higher power efficiency level.
- Various arrangements of three or four types of cores. For example, one or more high-peformance cores, power-efficient cores, and intermediate cores.
- One or more complexes and no individual cores. For information on complexes, see 2.3.1 What is a complex? on page 25.
- One or more complexes and individual cores.

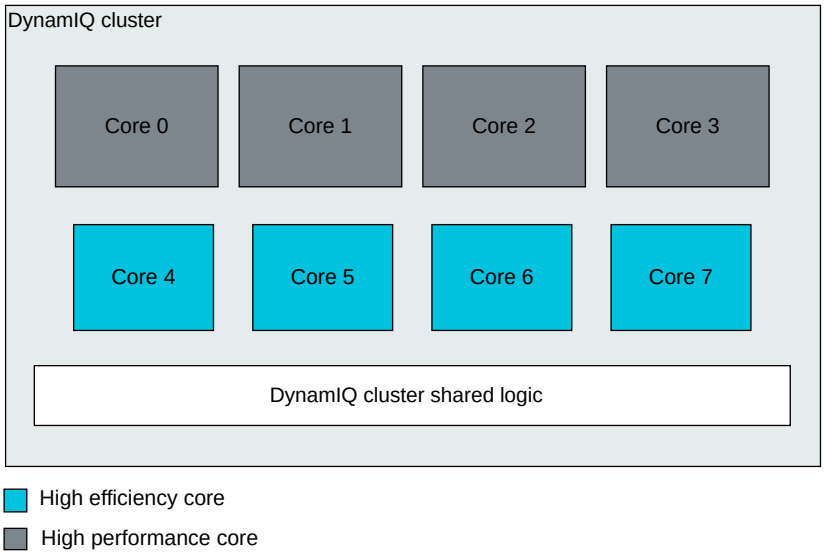
The following figure shows a cluster that is configured with all the same type of core.

Figure 2-2: DynamIQ cluster with one type of core



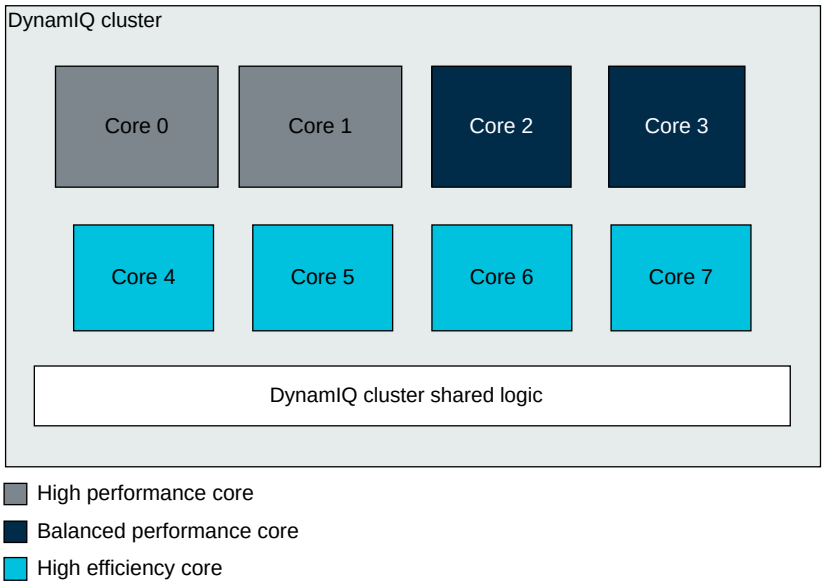
The following figure shows a cluster that is configured with two types of core.

Figure 2-3: DynamIQ cluster with two types of cores



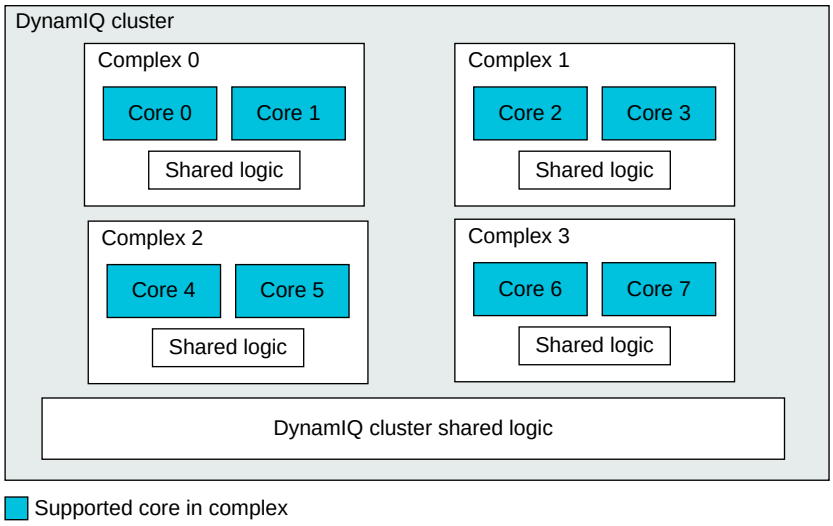
The following figure shows a cluster that is configured with three types of core.

Figure 2-4: DynamIQ cluster with three types of cores



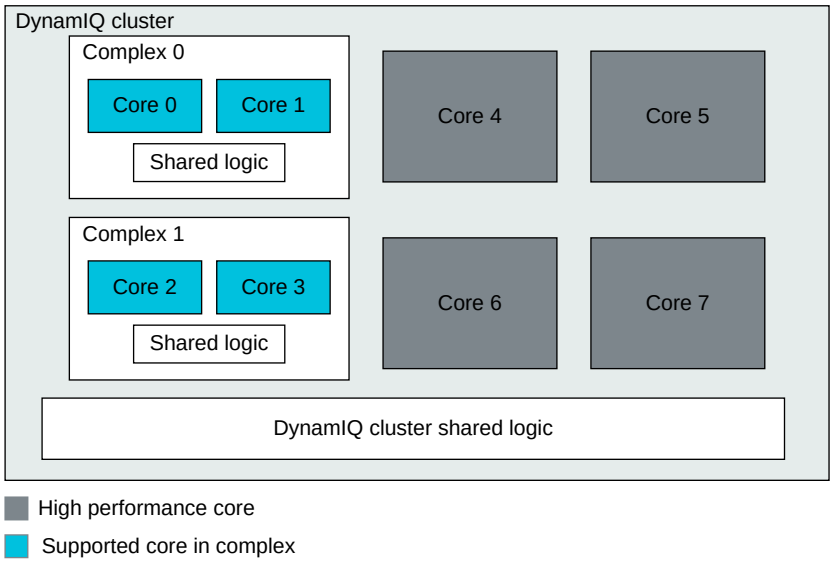
The following figure shows a cluster that is configured with four complexes.

Figure 2-5: DynamIQ cluster with four complexes



The following figure shows a cluster that is configured with two complexes, and four individual cores.

Figure 2-6: DynamIQ cluster with two complexes and four high performance cores



For information about the permissible combinations of cores in the cluster, see your DSU-110 Release Note. For information about the type of core you have licensed, for example the core cache sizes, see the Release Note for your core.

2.3.1 What is a complex?

The DSU-110 DynamIQ™ cluster supports blocks that are called complexes which contain up to two cores of the same type and some shared logic. Sharing some logic between the two cores of a dual core complex can make the dual core complex area efficient. However, this area efficiency is at the cost of reduced performance compared with using two single-core complexes.

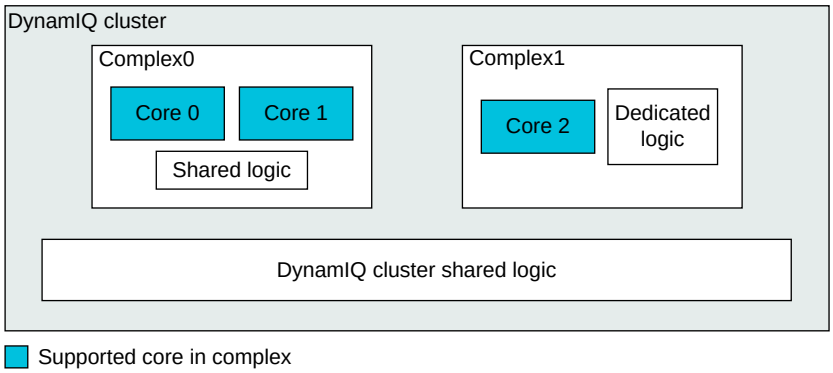
Only certain types of cores which have a merged-core microarchitecture can be used in a complex. To see if your core is supported in a complex and for further details of complexes, see your core *Technical Reference Manual* (TRM).

The maximum number of cores instantiated in the cluster is 12. This number includes:

- Any cores that are not instantiated in a complex. These cores are called standalone cores.
- Any cores instantiated in single core complexes.
- Any cores instantiated in a dual core complex.

The following figure shows a cluster that contains a dual-core complex and a single-core complexes.

Figure 2-7: Cluster with a dual-core complex and a single-core complex



When a core type can be defined as part of a complex, then all instances of that core type (in the cluster) are implemented as complexes. This is either as part of a single-core complex or dual-core complex. Having all instances of a core type formed into complexes within the cluster, ensures consistent clock and power management control.

Within a dual-core complex, logic such as a *Vector Processing Unit* (VPU), L2 *Translation Lookaside Buffer* (TLB), and L2 cache logic is shared between the cores and is collectively known as *shared logic*. In a single-core complex, the same logic resides outside the core but is collectively known as *dedicated logic*.

There is a tradeoff in area and performance between implementing a dual-core complex compared with two single-core complexes or two single cores. A dual-core complex provides better area efficiency but with some reduced performance.

In this document, where reference to a core is made on its own, unless otherwise stated, you can assume this refers to all cores within the cluster. Therefore, this usage applies to both cores within complexes, called complexed cores, and standalone cores.

When describing core functionality, the complexed core is assumed to include the complex shared logic and the unified cache unless otherwise stated. If the functionality being described only applies to either standalone cores or complexed cores, this is stated. In certain situations, to state, where functionality applies to both standalone and complexed cores, is also appropriate.

Related information

2.3 Cluster configurations on page 22
2.7 Core, complex, and processing element numbering on page 30

2.3.2 L3 memory system variants

By default the *DynamIQ™ Shared Unit-110* (DSU-110) is implemented with an L3 cache. Depending on your requirements, you can instead implement the DSU-110 without an L3 cache. Alternatively

you can implement the DSU-110 to support a Direct connect connection to your core if your core supports this.


Note

Not all cores support Direct connect. To check if your core supports Direct connect, see your core *Technical Reference Manual* (TRM).

There are three possible L3 memory system implementations:

L3 cache present

This is the default implementation. It provides the most functionality and is suitable for general-purpose workloads.

L3 cache not present

In this implementation, the L3 cache is not present but snoop filter and *Snoop Control Unit* (SCU) logic are present.

This variant allows multiple cores in the cluster and manages the coherency between them. It supports other implementation options such as *Accelerator Coherency Port* (ACP), peripheral port, and AXI or CHI master ports. Excluding the L3 cache RAMs saves layout area but performance of typical workloads is reduced. Therefore, Arm recommends that this variant is only used in specialized use cases, or when there is a system cache present that can be used by the cores.

Direct connect

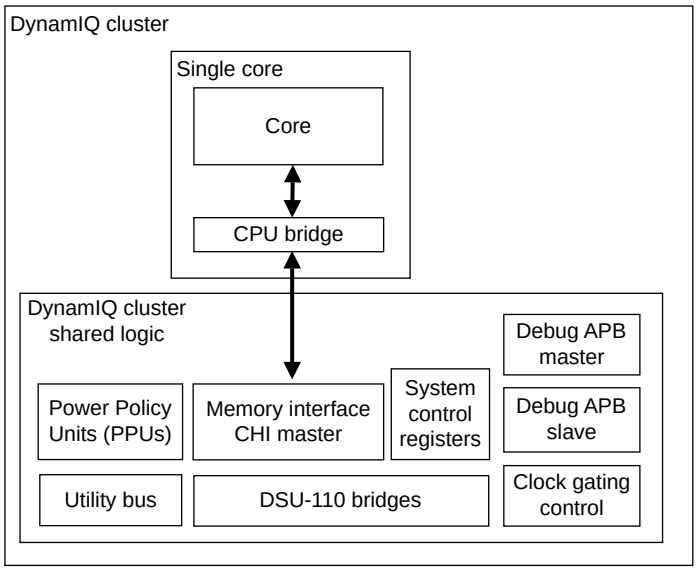
In the Direct connect implementation, the L3 cache, snoop filter, and SCU logic are not present.

This variant is specifically for use with a CHI interconnect. It offers extra area savings and reduced latency when compared to the previous variants. Because there is no L3 cache in the cluster, this variant relies on the system cache in the interconnect for performance. To check if your core supports this variant, see the *DSU-110 dependent features* section in your core TRM.

Because this variant does not include any coherency logic, it is only supported when there is a single core or single-complex in the cluster. When the DSU-110 is configured for Direct connect, optional interfaces such as ACP or the peripheral port are not supported. See *Implementation Parameters* in *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual* for more information. The master port must be a single 256-bit CHI interface.

The following diagram shows the DSU-110 implemented with Direct connect.

Figure 2-8: L3 Direct connect implementation



For more information on how to implement the DSU-110 with one of the L3 memory system variants, see the *Configuration Guidelines* chapter in *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

Related information

2.2 DynamIQ Shared Unit-110 configuration parameters on page 20
2.1 DynamIQ Shared Unit-110 features on page 18

2.4 Supported standards and specifications

The *DynamIQ™ Shared Unit-110* (DSU-110) complies with the Arm®v9.0-A architecture and all previous Armv8-A architectures up to Arm®v8.5-A.

The following table lists the architectures that the DSU-110 is compliant with.

Table 2-1: Standards and specifications support in the DSU-110

Standard of specification	Version	Notes
Arm architecture	Arm®v9.0-A	The DSU-110 supports cores based on the Arm®v9.0-A architecture. These cores also support previous Arm®v8-A architectures up to Arm®v8.5-A, dependent on the core. See the section *Supported standards and specifications* in your core *Technical Reference Manual* (TRM) for details.

Standard of specification	Version	Notes
Reliability, Availability, and Serviceability (RAS)	RAS v8.4	The DSU-110 supports RAS features that conform to the v8.4 RAS architecture.
Advanced Microcontroller Bus Architecture (AMBA)	AMBA 5 CHI Issue E	See 3.4 Interfaces on page 39 for information on what AMBA protocols the interfaces support.
	AMBA AXI5 Issue H	
CoreSight™ architecture	v3.0	-
Debug	Arm®v9.0-A	Arm®v9.0-A architecture that is implemented with Arm®v8.4-A Debug architecture support and Arm®v8.3-A Debug over powerdown support. See the *Arm®v8.3 Debug Architecture* for information on this architecture.
Generic Interrupt Controller (GIC) architecture CPU interface and Stream Protocol interface.	GICv4.1	The DSU-110 uses Affinity level 1 to distinguish between different cores. This level is not supported by some interrupt controllers, such as GIC-500.
Performance Monitoring Unit (PMU)	PMUv3	-

Related information

3.2 DynamIQ cluster shared logic components on page 34
3.4 Interfaces on page 39

2.5 Test features

The *DynamIQ™ Shared Unit-110* (DSU-110) provides test signals that enable the use of *Automatic Test Pattern Generation* (ATPG) to test the *Snoop Control Unit* (SCU) and other logic in the DSU-110. Additionally, internal *Memory Built-In Self Test* (MBIST) interfaces are provided to test the L3 cache and other memory arrays of the DSU-110.

The DSU-110 includes an ATPG test interface that provides signals to control the *Design for Test* (DFT) features of the DSU-110, and the cores in the cluster. For example, there are signals to control the resets on the flip-flops during scan shift. Consideration of how you use these signals can help to prevent problems with DFT implementation.

Arm® also provides an MBIST interface that enables you to test the DSU-110 RAMs at operational frequency. You can add your own MBIST controllers to automatically generate test patterns and perform result comparisons. Optionally, you can use your EDA MBIST interfaces instead of the MBIST interfaces supplied by Arm®.

For a list of external scan control signals and information on their usage, see *DFT integration guidelines* in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*. For information about the test signals related to your core, see your core *Configuration and Integration Manual*.

2.6 Design Tasks

Both the *DynamIQ™ Shared Unit-110* (DSU-110) and the cores in the cluster are delivered as synthesizable *Register Transfer Level* (RTL) descriptions in Verilog HDL. Before you can use the DSU-110 and the cores, you must implement, integrate, and program them.

A different party can perform each of the following tasks. Each task can include implementation and integration choices that affect the behavior and features of the DSU-110 and the cores.

Implementation

The implementer configures and synthesizes the RTL to produce a hard macrocell. This task includes integrating RAMs into the design.

Integration

The integrator connects the macrocell into a System on Chip (SoC). This task includes connecting the macrocell to the memory system and peripherals.

Programming

In the final task, the system programmer develops the software to configure and initialize the DSU-110 and the cores in the cluster and tests the application software.

The operation of the final device depends on the following:

Build configuration

The implementer chooses the options that affect how the RTL source files are pre-processed.

These options usually include or exclude logic that affects one or more of the area, maximum frequency, and features of the resulting macrocell.

Configuration inputs

The integrator configures some features of the DSU-110 and cores in the cluster by tying inputs to specific values.

These configuration settings affect the start-up behavior before any software configuration is made. They can also limit the options available to the software.

Software configuration

The programmer configures the DSU-110 and the cores in the cluster by programming values into registers. The configuration choices affect the behavior of the DSU-110 and the cores.

For implementation options, see the following:

- *RTL configuration process* in the *Configuration and Integration Manual* for your licensed core
- *RTL configuration process* in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*

Related information

14. System control registers on page 168

2.7 Core, complex, and processing element numbering

A cluster contains one or more cores. The cluster can also contain one or more complexes which can be made up of either a single core or two cores. Because certain parts of the design, such as signal names and register bit values, depend on the number of cores and complexes within the cluster, a numbering system has been created.

Throughout this document, the following numbering is used for cores, *Processing Elements* (PEs), and complexes:

Core

The numbering of core instances in the cluster ranges from zero to CN, where CN has the value of the total number of cores minus one. This numbering also includes cores instantiated within a complex. For example, CN = 5 for a cluster comprised of two dual-core complexes and two standalone cores.

For individual core instances, the term y is used, which ranges from zero to CN. For example, when referring to the second instance of a core, y = 1. The term y is called the core instance number.

Complexes

The numbering of complex instances in the cluster ranges from zero to CX, where CX has the value of total number of complexes minus one. For example, CX = 1 for a cluster comprised of two complexes.

For individual complex instances, the term x is used, which ranges from zero to CX. For example, when referring to the second instance of a complex, x = 1. The term x is called the complex instance number.

Processing element

The Arm architecture allows for cores to support multiple PEs.

The *DynamIQ™ Shared Unit-110* (DSU-110) supports cores with multiple PEs. Where a reference to a core is made, the core could be a core with only one PE (single-threaded core) or multiple PEs (multi-threaded core).

PEN is the total number of PEs in the cluster, starting from zero. This numbering also includes cores within complexes. For example, PEN = 5 for a cluster comprised of two dual-core complexes and two standalone cores, with all cores having one PE each.

For reference to individual PEs, the term z is used, which ranges from zero to PEN. For example, when referring to the second PE, z = 1.

In the current DSU-110, each core only has one PE. Therefore, PEN = CN.

For more information on the instance numbering for cores and complexes in the cluster, see *RTL configuration process* in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

2.8 Product revisions

The product revision increments at each release.

The following table indicates the main differences in functionality between product revisions.

Table 2-2: Product revisions

Revision	Notes
r0p0	First release.
r1p0	MTE functionality supported.
r2p0	Support for *Accelerator Coherency Port* (ACP), L3 cache powerdown (by ways and slices), L3 cache retention, *Memory System Resource Partitioning and Monitoring* (MPAM) L3 cache partitioning, and cache debug. Support for new debug system address map.
r2p1	Maintenance updates.
r3p0	Support for a 12 core cluster and ACP atomics. Maintenance updates.
r3p1	Maintenance updates.
r4p0	Support for L2 cache stashing and ACP peripheral port dependency. Added `PPU_RST_STATE` configuration option.

Changes in functionality that have an impact on the documentation also appear in C.1 Revisions on page 853.

3. Technical overview

A *DynamIQ™ Shared Unit-110* (DSU-110) cluster-based system is also known as a DSU-110. The DSU-110 comprises the two top-level modules.

These are:

A module to form a DSU-110 DynamIQ™ cluster

It includes the cores, complexes and the DynamIQ™ cluster shared logic.

A separate module for the DebugBlock

Separating the debug components from the DSU-110 DynamIQ™ cluster enables the debug components to be implemented in a separate power domain, or to be combined with an existing system power domain, allowing debug over power down.

All the main *System on Chip* (SoC) interfaces appear at the top level of the DSU-110. The DSU-110 connects the cores and complexes to an external memory system and the rest of the SoC.

3.1 DynamIQ cluster components

The DSU-110 DynamIQ™ cluster contains all the cores and complexes together with the DynamIQ™ cluster shared logic. All the DynamIQ™ cluster shared logic is automatically connected to the cores and complexes by the configuration script during build-time configuration.

Cores

There can be up to 12 cores in the DSU-110 DynamIQ™ cluster, with up to four different types of core. For information on the behavior and features of each core, see the *Technical Reference Manual* (TRM) of each core.

Complexes

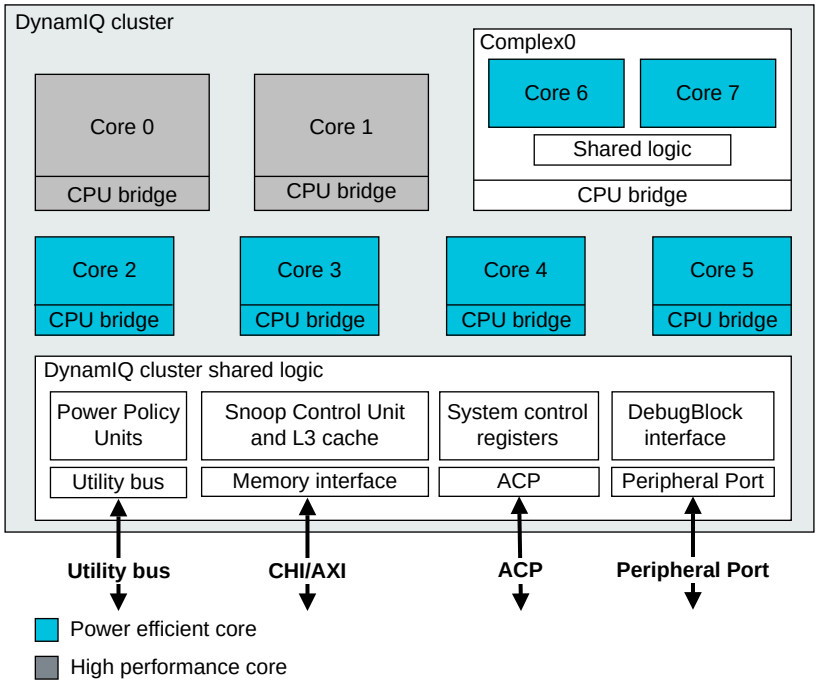
The DSU-110 DynamIQ™ cluster also supports up to either 12 single-core complexes, or six dual-core complexes. Complexes are made up of specialized cores. See 2.3.1 What is a complex? on page 25. See the *DSU-110 dependent features* section in your core TRM to determine if your core is supported in a complex.

DynamIQ™ cluster shared logic

The DynamIQ™ cluster shared logic forms part of the DSU-110 DynamIQ™ cluster. See 3.2 DynamIQ cluster shared logic components on page 34.

The following figure shows the main components that make up the DSU-110 DynamIQ™ cluster within the DSU-110.

Figure 3-1: DSU-110 DynamIQ™ cluster components



Related information

2.3 Cluster configurations on page 22
2.3.1 What is a complex? on page 25
3.2 DynamIQ cluster shared logic components on page 34
3.3 DebugBlock components on page 38

3.1.1 Integration of the cores in the cluster

When you implement a DSU-110 DynamIQ™ cluster, all interfacing between the cores, complexes, and the *DynamIQ™ Shared Unit-110* (DSU-110) is implemented automatically. All the external signal inputs and outputs pass through the DSU-110. The DSU-110 buffers and resynchronizes many of these signals to allow cores and complexes to be clocked at different speeds.

The memory interfacing of each core is internally connected to the DSU-110 L3 memory system. Where necessary, the DSU-110 implements additional buffering to compensate for different clock rates of the core and DSU-110 L3 memory system.

Each core has an external clock interface, which is routed through the DSU-110 to the respective core.

3.2 DynamIQ™ cluster shared logic components

The DynamIQ™ cluster shared logic includes the following components:

Snoop Control Unit

The *Snoop Control Unit* (SCU) maintains coherency between all the data caches in the cluster.

The SCU contains buffers that can handle direct cache-to-cache transfers between cores without having to read or write data to the L3 cache. Cache line migration enables dirty lines to be moved between cores. Also, there is no requirement to write back transferred cache line data to the L3 cache.

The SCU contains a set of snoop filters that track the addresses for locations cached in the core caches. Including the snoop filters means that the SCU does not need to request a look up in the core caches when it receives a coherent memory request. These snoop filters are accessed by coherent requests from the other cores or from the system. If there is a simultaneous hit in the L3 tags and the SCU snoop filters, then the L3 cache normally provides the data in preference to a core. The size of the snoop filter is automatically determined from the configured number of cores and the cache sizes in those cores.

Clock management

Clock gating is supported through Q-Channel requests from an external clock controller to the DSU-110. The Q-Channels allow individual control of the following clock input signals:

- **ATCLK**
- **COREyCLK** where y is the core instance number
- **COMPLEXxCLK** where x is the complex instance number
- **GICCLK**
- **PCLK**
- **PERIPHCLK**
- **PPUCLK**
- **SCLK**

L3 memory interfaces includes the following components:

Main memory master

The main memory master provides an interface between the DynamIQ™ Shared Unit-110 and the external interconnect. For a connection to an external coherent interconnect, the memory interface must be configured to use the AMBA 5 CHI (Issue E) protocol. For connection to a non-coherent external interconnect, the memory interface can either be configured to use the CHI protocol or AXI5 (Issue H) protocol. In either configuration, the interfaces are 256-bit wide, with support up to four interfaces.

Accelerator Coherency Port

The *Accelerator Coherency Port* (ACP) is an optional slave interface. The ACP provides direct memory access to cacheable memory. The SCU maintains cache coherency by checking ACP

accesses for allocation in the core and L3 caches. The ACP implements a subset of the ACE-Lite protocol.

Peripheral port

The peripheral port is an optional master interface and provides accesses to tightly coupled accelerators. The port implements the AXI5 or CHI (Issue E) master interface protocol.

Utility bus

The utility bus is a 64-bit AXI5 slave interface that provides access to the control registers for various system components in the cluster. The control registers are memory-mapped onto the utility bus. The utility bus provides programming access to the following system components:

- PPUs
- Activity monitors in the cores
- L3 cache power-related monitors
- *Maximum Power Mitigation Mechanism* (MPMM) registers in the cores
- *Reliability, Availability, and Serviceability* (RAS) registers

If control registers require access from the cores, then the system must provide a loopback mechanism for accesses from the cluster to the relevant address range to map to the utility bus.

L3 cache

The following table shows the optional L3 cache sizes together with their associativity.

Table 3-1: L3 cache size

Size	Associativity
256KB	16-way
512KB	16-way
1024KB	16-way
1536KB	12-way
2MB	16-way
3MB	12-way
4MB	16-way
6MB	12-way
8MB	16-way
12MB	12-way
16MB	16-way

All caches have 64-byte line cache length. Data and tag RAMs have *Error Correcting Code* (ECC) protection.

Power management and Power Policy Units

The DynamIQ™ cluster shared logic integrates several *Power management and Power Policy Units* (PPUs) to control power modes and resets. The PPUs can be programmed to directly select a

specific power mode or can be programmed to autonomously switch between power modes within a specified range, based on the requirements of the cluster. The PPUs can be programmed from your *System Control Processor* (SCP) using the utility bus to access them.

DSU-110 system control registers

The DynamIQ™ cluster shared logic implements a set of system control registers, which is common to all cores in the cluster. You can access these registers from any core in the cluster. These registers provide:

- Control for power management of the cluster
- L3 cache partitioning control
- CHI *Quality of Service* (QoS) bus control
- Information about the hardware configuration of the DSU-110
- L3 cache hit and miss count information

Some of the system control registers, for example those in the PPU, are memory-mapped to the utility bus and can only be accessed from this bus.

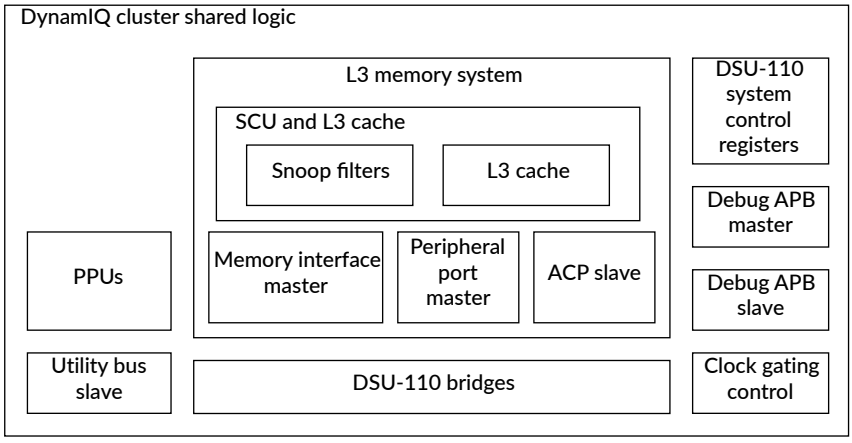
Debug and trace components

Each core includes an *Embedded Trace Extension* (ETE) to allow program tracing while debugging.

Trigger events from the cores are combined and output to the DebugBlock. Trigger events to the cores, and Debug register accesses, are received in the DebugBlock.

The following figure shows the main components of the DynamIQ™ cluster shared logic.

Figure 3-2: DynamIQ™ cluster shared logic components



Related information

4. Clocks and resets on page 44
5. Power management on page 50

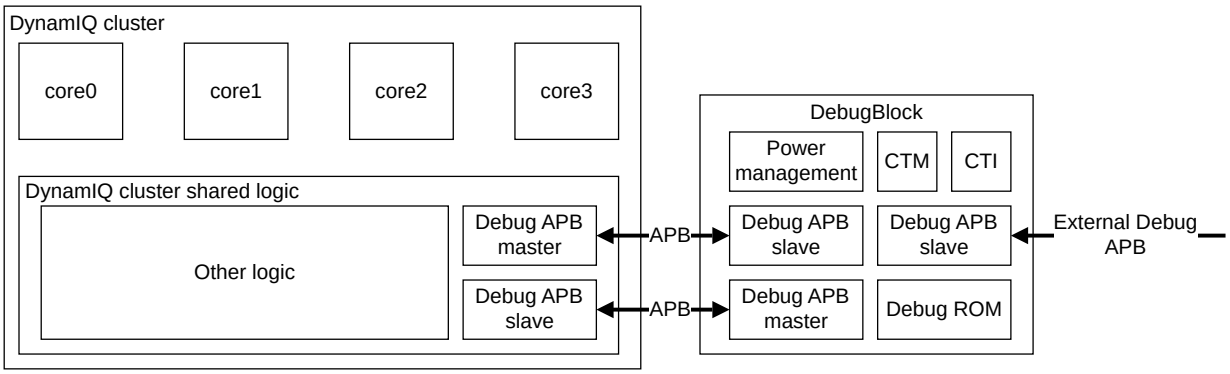
7. L3 cache on page 99
8. CHI master interface on page 107
9. AXI master interface on page 119
11. AXI or CHI master peripheral port on page 136
14. System control registers on page 168
13. Utility bus on page 164
15. Debug on page 170

3.3 DebugBlock components

The DebugBlock is a dedicated debug component for the *DynamIQ™ Shared Unit-110* (DSU-110) but is instanced as a separate unit to support debug over powerdown.

The following figure shows the main components of the DebugBlock.

Figure 3-3: DebugBlock components



Cluster (master) to DebugBlock APB (slave)

Trigger events from the cores are transferred to the DebugBlock as APB writes.

DebugBlock (master) to cluster APB (slave)

Trigger events to the cores are transferred as APB writes to the DSU-110. Register accesses from the system debug APB are transferred to the DSU-110.

System debug APB

The system debug APB slave interface connects to external CoreSight components, such as the *Debug Access Port* (DAP).

CTI and CTM

The DebugBlock implements an *Embedded Cross Trigger* (ECT). A *Cross Trigger Interface* (CTI) is allocated to each *Processing Element* (PE) in the cluster. An additional CTI is allocated to the cluster *Performance Monitoring Unit* (PMU) and the cluster *Embedded Logic Analyzer* (ELA) when present.


Note

The cluster CTI is not present in a Direct connect configuration.

The CTIs are interconnected through the *Cross Trigger Matrix* (CTM). A single external channel interface is implemented to allow cross-triggering to be extended to the *System on Chip* (SoC).

Debug ROM

The ROM table contains a list of components in the system. Debuggers can use the ROM table to determine which CoreSight components are implemented.

Power management and clock gating

The DebugBlock implements two Q-Channel interfaces, one for requests to gate the PCLK clock and a second for requests to control the Debug power domain.

Related information

15. Debug on page 170

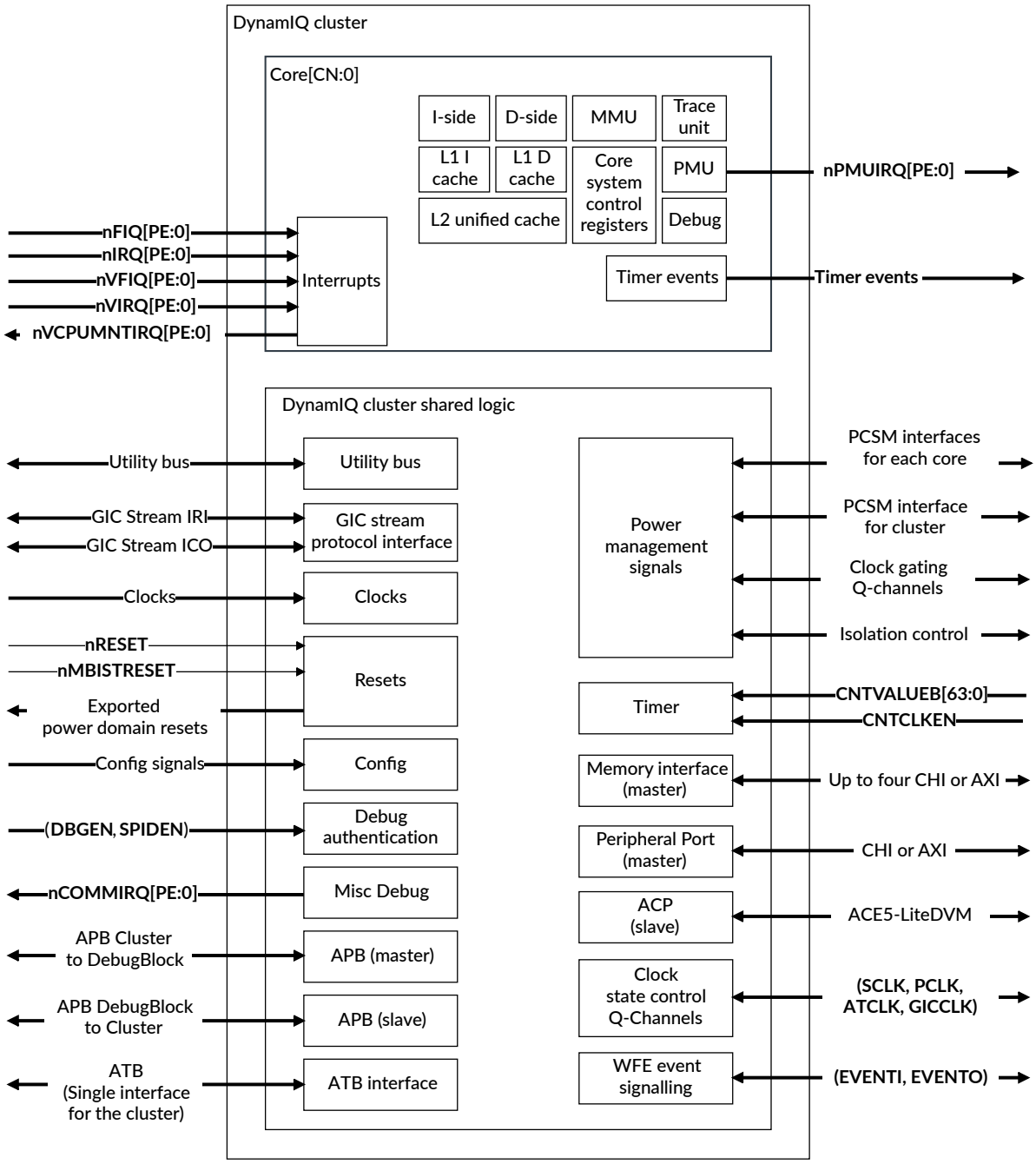
3.4 Interfaces

The *DynamIQ™ Shared Unit-110* (DSU-110) manages all the external interfaces to the *System on Chip* (SoC) including those from the cores and complexes in the cluster.

DSU-110 interfaces

The following figure shows the major external interfaces of the DSU-110 DynamIQ™ cluster.

Figure 3-4: DSU-110 DynamIQ™ cluster interfaces



The following table describes the external interfaces of the DSU-110 DynamIQ™ cluster.

Table 3-2: DSU-110 DynamIQ™ cluster interfaces

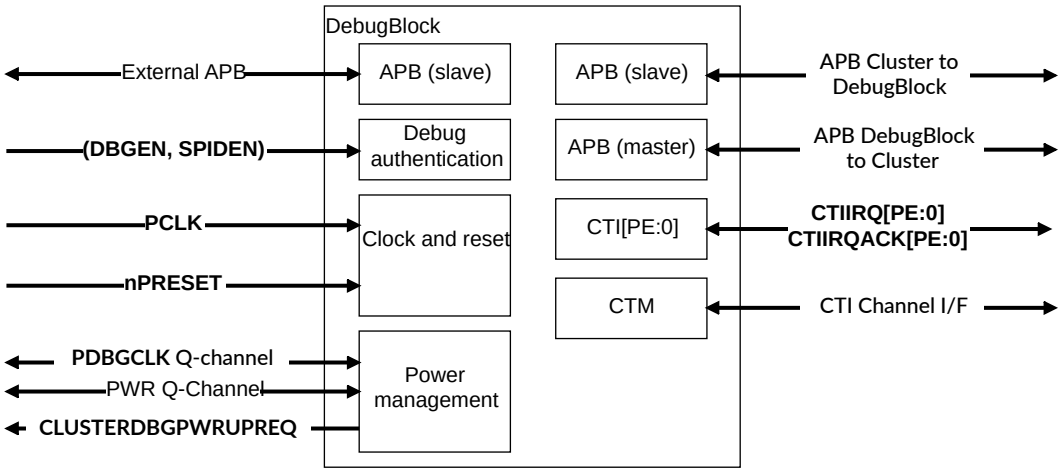
Purpose	Protocol	Notes
Trace	ATB	Master ATB interface. This is a single interface for the whole cluster.

Purpose	Protocol	Notes
Memory	AMBA AXI5 or AMBA 5 CHI (Issue E).	Master interface to main memory. You can configure the DSU-110 with either 1, 2, or 4 CHI master interfaces, or 1, 2, or 4 AXI master interfaces. See 2.2 DynamIQ Shared Unit-110 configuration parameters on page 20.
Accelerator Coherency Port (optional)	AMBA ACE5-Lite	Slave interface allowing an external master to make coherent requests to cacheable memory.
Peripheral port (optional)	AXI or CHI	Low-latency master interface to external memory
Utility bus	AXI	Memory-mapped port for accessing the following: • *Power Policy Units* (PPUs) • Activity monitors in the cores • L3 cache usage monitors and power control registers • *Maximum Power Mitigation Mechanism* (MPMM) registers in the cores • *Reliability, Availability, and Serviceability* (RAS) registers • *Memory System Resource Partitioning and Monitoring* (MPAM) registers
Cluster to DebugBlock	APB	APB interface from the cluster (master) to the DebugBlock (slave)
DebugBlock to cluster	APB	APB interface from the DebugBlock (master) to the cluster (slave)
Power state control	P-Channel	P-Channels for DSU-110 and core power management
Clock state control	Q-Channel	Q-Channels for clock gating control
Wait For Event (WFE) event signaling	-	Signals for WFE wake up events
Generic timer	-	Input for the generic time count value. The count value is distributed to all cores. Each core outputs timer events.
Generic Interrupt Controller (GIC) interfaces	-	Interrupts to individual cores. A single GIC Stream Protocol interface is shared by all cores.
Design for Test (DFT)	-	Interface to allow access for *Automatic Test Pattern Generation* (ATPG) scan-path testing.
Memory Built-In Self Test (MBIST)	Arm MBIST	Internal interface that supports the manufacturing test of the L3 cache and *Snoop Control Unit* (SCU) memories embedded in the DSU-110. Each core has its own internal MBIST interface.

DebugBlock interfaces

The following figure shows the major external interfaces of the DebugBlock.

Figure 3-5: DebugBlock interfaces



The following table describes the major external interfaces of the DebugBlock.

Table 3-3: DebugBlock interfaces

Purpose	Protocol	Notes
External debug	APB	Slave interface to external debug component, for example a *Debug Access Port* (DAP). It allows access to Debug registers and resources.
Cluster to DebugBlock	APB	APB interface from the cluster (master) to the DebugBlock (slave).
DebugBlock to cluster	APB	APB interface from the DebugBlock (master) to the cluster (slave).
Cross-trigger channel interface	CTI	Allows cross-triggering to be extended to external SoC components.
Power management	Q-Channel	Enables communication to an external power controller. To control clock gating and powerdown.

Related information

4. Clocks and resets on page 44
8. CHI master interface on page 107
9. AXI master interface on page 119
10. ACP slave interface on page 126
11. AXI or CHI master peripheral port on page 136
15. Debug on page 170

3.4.1 Page-Based Hardware Attribute

The *Page-Based Hardware Attribute* (PBHA) bits are provided by the cores, and passed on or preserved by the *DynamIQ™ Shared Unit-110* (DSU-110). PBHA is supported on all the master ports, the AXI or CHI peripheral port, and the *Accelerator Coherency Port* (ACP).

The PBHA bits are provided externally as sideband signals on each of the supported PBHA busses, alongside master requests. PHBA affects the following:

RAM sizes

To generate accurate PBHA bits on L3 cache evictions, the bits need to be stored in the L3 cache. This can be configured by setting the `L3_PBHA_STORAGE` configuration parameter. If this parameter is not set, the PBHA bits are only accurate for read transactions. If this is set, the width of all L3 tag RAM instances is increased by four bits.

ACP

The **ARPBHAS** and **AWPBHAS** signals provide the PBHA value for any ACP request.

Cache stash transactions on CHI

Cache stash transactions might be sent on the CHI interface. For these requests, the PBHA bits being used must be sent along with the stash snoop transaction. There are separate signals providing PBHA information for stashing snoops.

Transaction support

Transactions that do not have a physical address associated with them, for example DVM messages, do not provide the PBHA bits. Evict transactions that do not provide any data (for use in de-allocating a snoop filter) do not provide PBHA bits.

Mismatched aliases

If the same physical address is accessed through more than one virtual address mapping, and the PBHA bits are different in the mappings, then the results are **UNPREDICTABLE**. The PBHA value sent on the bus could be for either mapping.


Note

For information on PBHA signals, see *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

Related information

2.2 DynamIQ Shared Unit-110 configuration parameters on page 20

4. Clocks and resets

The *DynamIQ™ Shared Unit-110* (DSU-110) has separate clock signals for each of the standalone cores (those cores not in a complex), and for each complex. There are also clocks for the internal logic, and some of the external interfaces of the DSU-110.

The DSU-110 has a cluster-wide Cold reset, a DebugBlock reset, and a reset to be used with *Memory Built-In Self Test* (MBIST) testing. Implicit resets in the cluster logic and cores can also occur due to power state changes driven from the *Power Policy Units* (PPUs).

4.1 Clocks

The *DynamIQ™ Shared Unit-110* (DSU-110) has a separate clock signal for each standalone core or complex. There are also separate clocks for the internal logic, and some of the external interfaces.

The following table describes the clock signals of the DSU-110.

Table 4-1: DSU-110 clock signals

Signal	Description
COREyCLK	The clocks for each of the cores in the cluster that are not part of a complex. y is the core instance number, for example, **CORE0CLK** is the clock for core 0. These signals clock all core logic, including L1 and L2 caches.
COMPLEXxCLK	The clocks for each complex in the cluster. Each clock is connected to all cores in the respective complex. x is the complex instance number, for example, **COMPLEX0CLK** is the clock for complex 0.
SCLK	This clock is used for the *Snoop Control Unit* (SCU), L3 memory system, and all the external interfaces, including AXI, CHI, and *Accelerator Coherency Port* (ACP). It is also used for cores that are configured to run synchronously with the DSU-110.
PCLK (DebugBlock)	The clock for the DebugBlock **Note:** The DebugBlock and the DSU-110 DynamIQ™ cluster both have **PCLK** inputs. You might choose to connect both of these signals to the same clock. Alternatively, if you are using a different clock to drive the DebugBlock than the DSU-110 DynamIQ™ cluster, ensure that you place an asynchronous bridge between the two clock domains.
PCLK (DSU-110 cluster)	The clock for the debug interface in the DSU-110 DynamIQ™ cluster **Note:** The DebugBlock and the DSU-110 DynamIQ™ cluster both have **PCLK** inputs. You might choose to connect both of these signals to the same clock. Alternatively, if driving the DebugBlock with a different clock to the DSU-110 DynamIQ™ cluster, ensure that you place an asynchronous bridge between the two clock domains.
ATCLK	The clock for the ATB trace bus output from the DSU-110
GICCLK	The clock for the *Generic Interrupt Controller* (GIC) AXI-Stream interface between the DSU-110 and an external GIC
PERIPHCLK	The clock for the peripheral logic inside the DSU-110 such as clock and power management logic and timers
PPUCLK	The clock for the *Power Policy Units* (PPUs). The PPUs reside in their own clock domain, see 4.2 Clock domains on page 46.

For more information on core and complex clock signaling, see *RTL configuration process* in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

All clocks can be driven fully asynchronously to each other. The DSU-110 contains all the necessary synchronizing logic for crossing between clock domains. There are no clock dividers and no latches in the design. The entire design is rising-edge triggered.


Note

- You can configure the cores to run synchronously to the L3 memory system, on a per-core basis at the build time configuration stage. If this option is chosen, the corresponding **COREXCLK** signals and **COMPLEXxCLK** signals (if applicable) are not present and the synchronous cores are run with **SCLK**.
- The DebugBlock can be clocked by a different clock from the cluster **PCLK**. To allow this, you must add asynchronous bridges between the cluster and the DebugBlock.

Some external interfaces, such as the main CHI or AXI master interface, support a clock enable input to allow the external logic to run at a lower-synchronous frequency.

While there is no functional requirement for any of the clocks to have any relationship to any of the others, the DSU-110 is designed with the following expectations to achieve an acceptable performance:

- The **COREXCLK** or **COMPLEXxCLK** can be dynamically scaled to match the performance requirements of that core.
- **SCLK** is recommended to run between the maximum **COREXCLK** or **COMPLEXxCLK** frequency and approximately half of the maximum **COREXCLK** or **COMPLEXxCLK** frequency.
- **SCLK** can run at synchronous 1:1 or 2:1 frequencies with the external interconnect, avoiding the need for an asynchronous bridge between them.
- **ATCLK** can be run at the same frequency as the trace subsystem that it connects to. This would typically be approximately 25% of the maximum **COREXCLK** or **COMPLEXxCLK** frequency.
- **GICCLK** can be run at the same frequency as the interrupt controller that it connects to. This would typically be approximately 25% of the maximum **COREXCLK** or **COMPLEXxCLK** frequency.
- **PCLK** can run at the same frequency as the debug subsystem that it connects to. This would typically be approximately 25% of the maximum **COREXCLK** or **COMPLEXxCLK** frequency.
- The **PERIPHCLK** domain contains the architectural timers, and software performance can be impacted if reads to these registers take too long. Therefore, Arm® recommends that the **PERIPHCLK** frequency is at least 25% of the maximum **COREXCLK** or **COMPLEXxCLK** frequency.
- Arm® recommends that the **PPUCLK** clock frequency is at least 25% of the maximum **COREXCLK** or **COMPLEXxCLK** frequency. When implementing the retention power state controls for retention power and operating modes, retention entry and exit latency is limited by the **PPUCLK** clock frequency.

Related information

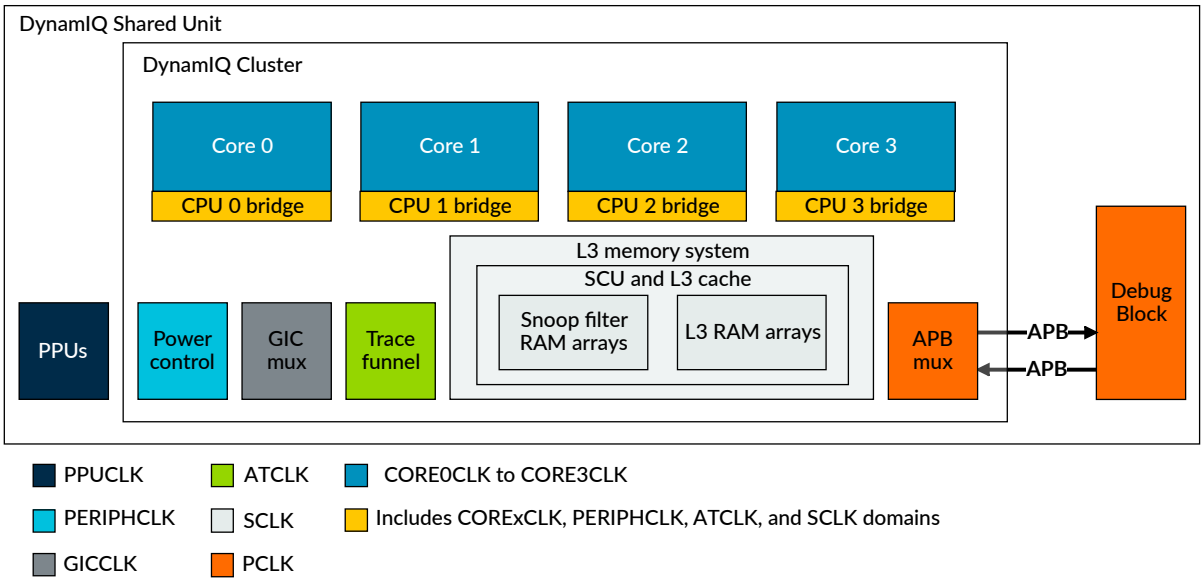
4.2 Clock domains on page 46
2.2 DynamIQ Shared Unit-110 configuration parameters on page 20

4.2 Clock domains

The *DynamIQ™ Shared Unit-110* (DSU-110) has multiple clock domains. Each core or complex can be implemented in a separate clock domain.

The following figure shows the clock domains for an example cluster with four standalone cores.

Figure 4-1: DSU-110 clock domains



The cluster contains several clock domains for functionality that is likely to be connected to different clocks in the system. Within each core, the CPU bridge contains asynchronous bridges for all crossings between the core and cluster clock domains. Each CPU bridge is split, with one half of each bridge in the core clock domain and the other half in the relevant cluster domain. At the cluster level, there is the *Snoop Control Unit* (SCU) bridge which contains crossings between the cluster clock domains as required.

Note

The DebugBlock is shown in a common PCLK domain with the cluster debug logic. However, the DebugBlock can be placed in a different clock domain if asynchronous bridges are inserted on the APB interfaces between the DebugBlock and the cluster.

Related information

2.2 DynamIQ Shared Unit-110 configuration parameters on page 20
4.1 Clocks on page 44

4.3 Resets

The *DynamIQ™ Shared Unit-110* (DSU-110) has three external reset signals, a cluster-wide Cold reset , a cluster-wide MBIST reset, and a reset for the DebugBlock. Other resets, such as Warm reset s to the cores are controlled, inside the cluster, by the *Power Policy Units* (PPUs).

The following table describes the DSU-110 reset signals.

Table 4-2: DSU reset signals

Signal	Description
nRESET	A DSU-110 DynamIQ™ cluster reset. **nRESET** is a single cluster-wide Cold reset . **nRESET** resets the PPU for the cluster and cores, which in turn resets the DynamIQ™ cluster shared logic and cores.
nMBISTRESET	A DSU-110 DynamIQ™ cluster reset. **nMBISTRESET** is a single cluster-wide reset signal that resets all necessary logic in the cores and cluster for *Memory Built-In Self Test* (MBIST) testing.
nPRESET	The DebugBlock reset. A reset for all resettable registers in the DebugBlock.

The **nMBISTRESET** signal is intended for use with an external MBIST controller. This avoids the need for it to control the reset logic in the *System on Chip* (SoC).

All reset inputs can be asserted (HIGH to LOW) and deasserted (LOW to HIGH) asynchronously. Reset synchronization logic inside the DSU-110 ensures that reset deassertion is synchronous for all resettable registers inside those reset domains. The core clock does not need to be present for reset assertion. For reset deassertion, only **PPUCLK** (for the cluster) or **PCLK** (for the DebugBlock **nPRESET**) must be active. However, the reset deassertion in other clock domains is not effective until the relevant clock for that domain is active.

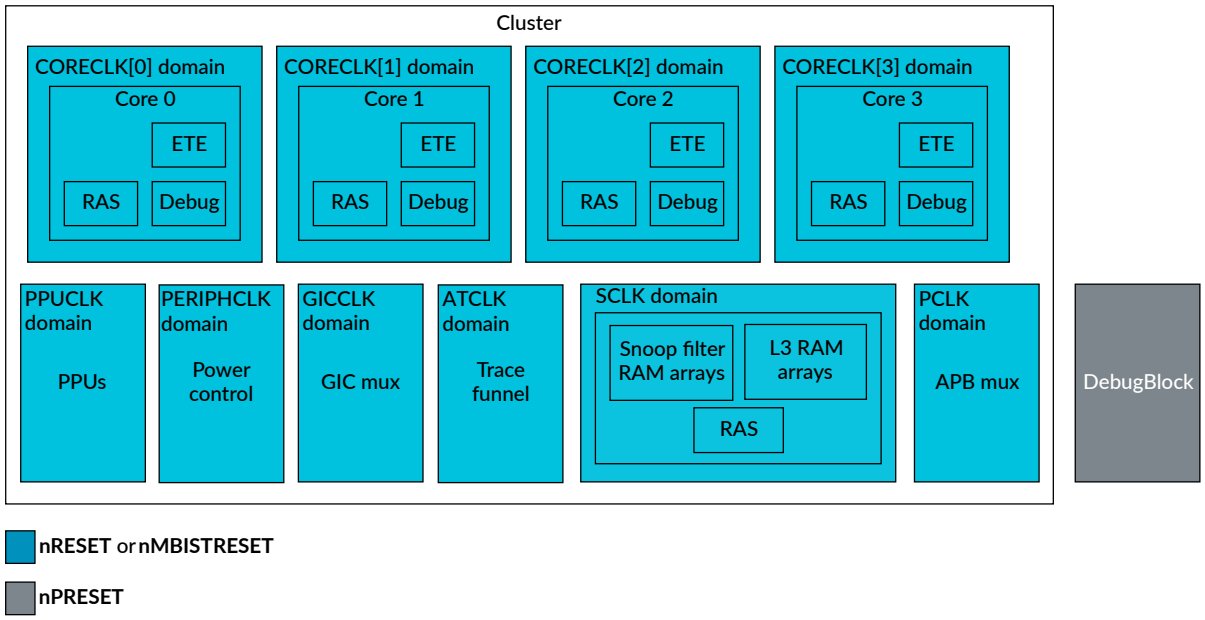
Resetting individual cores and other cluster logic can be performed by programming the appropriate integrated PPU, see 6. Power and reset control with Power Policy Units on page 78. When the cluster internal resets are asserted, the PPUs drive the reset output signals that correspond to the internal reset domain.


Note

You can use the DSU-110 reset output signals, which are driven from the PPUs, to reset any external logic that is in the same power domain as the relevant parts of the cluster. For example, if the DebugBlock is in the same power domain as the cluster, the **nPRESET** input to the DebugBlock can be connected to the **nPRESET** output of the cluster. The **nPRESET** output of the cluster is driven by the cluster PPU. These reset outputs are all generated in the **PPUCLK** domain and therefore must be synchronized before use in the destination component.

The following figure shows the pin-controlled reset domains.

Figure 4-2: DSU-110 pin-controlled reset domains



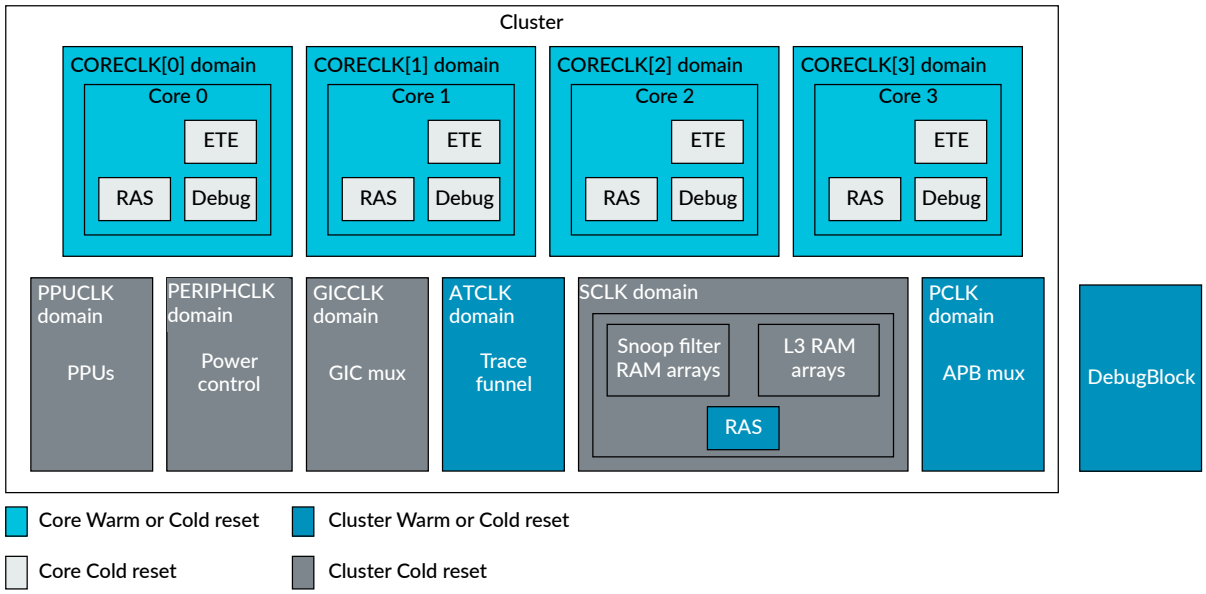
4.4 Resetting with Power Policy Units

The *Power Policy Units* (PPUs) for the cluster and each of the cores are used to control the power management features of the cluster and cores using a software interface. This includes managing various power states and transitions between these states. Certain power mode changes, for example powering up the cluster from a powered down state, include implicit resets to internal logic.

This internal reset is managed by the PPU controlling the transition between the two modes. This internal reset does not require an external signal to be asserted or explicit programming of the PPU. For more information on what internal reset actions result from power mode changes, see 6. Power and reset control with Power Policy Units on page 78.

The following figure shows the reset domains that can be controlled by programming the PPUs.

Figure 4-3: DSU-110 PPU-controlled reset domains



When performing a Cold reset by asserting the **nRESET** signal, the PPUs are reset, and this in turn causes an internal reset to all the cluster and core logic. For more details on this process, see 6.2.2 nRESET sequence on page 82.

5. Power management

This chapter describes the power domains, power modes, and operating modes for the *DynamIQ™ Shared Unit-110* (DSU-110) and for the cores and complexes. It also provides a state transition diagram showing the supported power and operating mode transitions of the cluster, and describes the power-saving features employed by the DSU-110.

This chapter does not describe how the various power and operating modes are managed using the *Power Policy Units* (PPUs). For information on using the PPUs, see 6. Power and reset control with Power Policy Units on page 78.

5.1 Power management in the DSU-110

The *DynamIQ™ Shared Unit-110* (DSU-110) provides various mechanisms to control both dynamic and static power dissipation. These mechanisms are associated with a set of power domains, power modes, and operational modes. Some of these mechanisms are brought under software control using *Power Policy Units* (PPUs).

The power management techniques employed by the DSU-110 and cores in the cluster include:

- Internal core clock gating where different internal parts of the core are clock idle
- Per-core *Dynamic Voltage and Frequency Scaling* (DVFS)
- Powerdown of components of the cluster which can include:
  - Cores
  - All the L3 cache or parts of the L3 cache. See 5.4.1 L3 cache RAM powerdown on page 57 and 5.4.2 L3 cache slice powerdown on page 61.
- Retention which is a low-power mode that retains the register and RAM state. Retention can be applied to the following components of the cluster:
  - Cache RAMs in the cores
  - All of the L3 cache or parts of the L3 cache

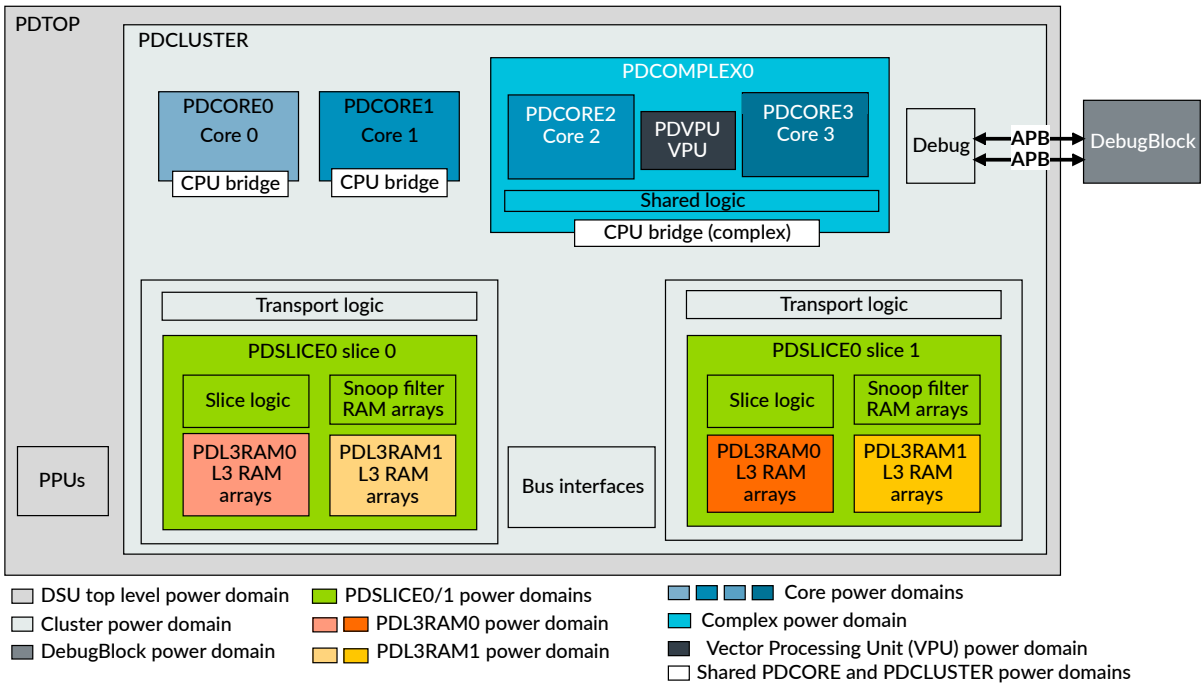
- The DSU-110 power domain architecture, power modes, and operational modes, are based on the Arm Power Control System Architecture, see *Arm® Power Control System Architecture*.
- This chapter does not describe how to use the PPUs for the cluster or the cores, see instead 6. Power and reset control with Power Policy Units on page 78.

5.2 DSU-110 supported power domains

The *DynamIQ™ Shared Unit-110* (DSU-110) supports different power domains. You do not need to implement all power domains. The number and type of domains that are implemented depends on the choices made by the *System on Chip* (SoC) implementer. Each of the L3 cache slices, cores, and complexes can be placed in their own separate power domain. As the number of these components can vary depending on implementation, therefore the total number of power domains can also vary.

The following figure shows all the different types of power domains that are supported in a DSU-110-based cluster.

Figure 5-1: DSU-110 power domains



Note

The logic for each CPU bridge is split across the core and cluster power domains.

The cluster comprises the following power domains:

PDTOP

The top-level power domain (PDTOP) is typically placed in the same power domain as the other system components, for example, external bus infrastructure. The only cluster logic in this domain is the *Power Policy Units* (PPUs). This domain must be relatively always on compared to the other

power domains. This is because the PPUs need to be able to power down the other domains including PDCLUSTER while remaining active. Therefore, the PDTOP power domain must be powered up before any of the other power domains are powered up, and it must only be powered down after the other power domains have been powered down.

The DebugBlock can be in the PDTOP or PDCLUSTER power domains. Alternatively, the DebugBlock can be placed with other debug components in a separate power domain as required.

PDCLUSTER

Separating the cluster power domain (PDCLUSTER) from the power domain where the PPUs reside allows the PPUs and other system logic to stay on when the rest of the cluster is powered off.

PDCORE<CN>

Optionally, you can place each core in its own separate power domain, for example PDCORE0 and PDCORE1 for two cores. Placing the cores in their own power domains allows them to be powered down individually. A core might have further internal power domains. See your core *Technical Reference Manual* (TRM) for details.

For any individually instantiated cores, their respective CPU bridges have logic both in the PDCORE power domain and in the PDCLUSTER power domain.

PDCOMPLEX<CPXN>

If any complexes are included in the cluster, they each reside in their own separate power domain, for example PDCOMPLEX0 and PDCOMPLEX1 for two complexes. Within each PDCOMPLEX power domain, each core is in its own separate power domain, for example PDCORE2 and PDCORE3 as shown in Figure 5-1: DSU-110 power domains on page 51. If the *Vector Processing Unit* (VPU) is included, this resides in its own separate PDVPU power domain within PDCOMPLEX. Both PDCORE and PDVPU are gated power domains that can support retention.

The CPU bridge for a complex has some logic in the PDCLUSTER power domain and remaining logic in the PDCOMPLEX power domain.

For more information on the power domains of a complex, see the *Power management* section of your core TRM.

PDSLICE<SLICEN>

Each L3 cache slice is placed in its own separate power domain (PDSLICE), to allow the logic and RAMs of the cache slice to be powered down when not required. For example, in a cluster with more than one core, where only one core is powered on and lightly loaded most of the L3 cache might not be required.

- For configurations with more than two L3 cache slices, the *Power Policy Unit* (PPU) cannot control the powering up or powering down of each individual cache slice. Instead, either only a single L3 cache slice is powered up or all L3 cache slices are powered up. For example, for a DSU-110 configured with four L3 cache slices, cache slices 1-3 are powered down together while cache slice 0 remains powered up.

- There must always be at least one L3 cache slice powered up, unless powering off the entire cluster. Therefore, for a cluster with only one L3 cache slice configured, the PPU cannot power down this cache slice without powering off the entire cluster.

PDL3RAM0 and PDL3RAM1

Within each L3 cache slice, there are further power domains for the L3 cache RAMs (PDL3RAM0 and PDL3RAM1). These domains enable half or all of the cache ways for those RAMs to be powered off when the cache is empty, saving leakage power. The RAM power domains are expected to use power gates or retention support that is typically built into many RAMs.

5.3 Cluster power modes

The DSU-110 DynamIQ™ cluster and each of the cores and complexes in the cluster have a defined set of power modes and corresponding legal transitions between these modes.

The following table shows the supported power modes for the DSU-110 DynamIQ™ cluster.

Table 5-1: DSU-110 DynamIQ™ cluster power modes

Power mode	Short name	Description
On mode	ON	On mode is the normal mode of operation where all cluster functionality is available.
Off mode	OFF	In Off mode, power is removed from the cluster logic and all the RAMs.
Functional retention mode	FUNC_RET	In Functional retention mode, the L3 cache RAMs and snoop filter RAMs are placed in a retention state. Data is retained in these RAMs. The rest of the DynamIQ™ cluster shared logic remains powered up.
Memory retention mode	MEM_RET	In Memory retention mode, only the L3 cache RAMs are placed in retention. The rest of the DSU-110 DynamIQ™ cluster including the L3 logic, the cores, and the complexes are powered down.
Emulated off mode	OFF_EMU	In Emulated off mode, the cluster behaves logically as if it were in the Off mode, except that the logic remains powered. The Debug state is retained and accessible.
Emulated memory retention mode	MEM_RET_EMU	In Emulated memory retention mode, the cluster behaves logically as if it were in the Memory retention mode, except that the logic remains powered. The Debug state is retained and is accessible.
Warm reset mode	WARM_RST	The Warm reset mode provides a Warm reset to all the DynamIQ™ cluster shared logic apart from the PPUs.
Debug recovery mode	DBG_RECOV	Debug recovery mode is used for applying a Warm reset to the cluster, while preserving memory and *Reliability, Availability, and Serviceability* (RAS) registers for debug purposes. Both L3 cache and RAS state are preserved when transitioning from DBG_RECOV mode to ON mode. Debug recovery mode is typically used in debugging a watchdog timeout.

5.3.1 On mode (ON)

In the On mode, the *DynamIQ™ Shared Unit-110* (DSU-110) is powered up and fully operational.

When a transition to the On mode completes, all caches that are powered up according to the current operating mode are accessible and coherent. Other than the normal architectural steps to enable caches, no additional software configuration is required.

5.3.2 Off mode (OFF)

In the Off mode, all DynamIQ™ cluster shared logic including the snoop filters and L3 cache RAMs is powered down. The PDCLUSTER domain is inoperable and all state is lost.

In the Off mode, power is removed from PDCLUSTER domain (DSU-110 DynamIQ™ cluster), but the PDTOP domain is still powered up including all the *Power Policy Units* (PPUs).

The *DynamIQ™ Shared Unit-110* (DSU-110) can be initialized into this mode on a Cold reset.

5.3.3 Functional retention mode (FUNC_RET)

Functional retention mode allows the L3 cache and snoop filter RAMs to be placed in a retention state if the L3 cache RAMs have not been accessed for a configurable period of time. In this mode, the contents of the L3 cache RAMs are retained, while the rest of the DynamIQ™ cluster shared logic remains powered up and operational.

The time period before the RAMs enter retention can be configured using the IMP_CLUSTERPWRCTLR_EL1 register, see A.1.6 IMP_CLUSTERPWRCTLR_EL1, Cluster Power Control Register on page 226.

The *Power Policy Units* (PPUs) can be programmed to automatically control entry and exit from this mode without software intervention, see 6. Power and reset control with Power Policy Units on page 78.

The length of time before the L3 cache RAMs enter this mode can be configured. Therefore, retention technologies that take multiple cycles to enter or exit retention can be used without significantly degrading performance.

This mode can be entered independently of the current core power modes and is transparent to software. When a core makes an access to the L3 cache, or the system sends a snoop, then the cluster requests to the cluster *Power Policy Unit* (PPU) that it moves from FUNC_RET mode to an ON mode to service the access.

5.3.4 Memory retention mode (MEM_RET)

In Memory retention mode, the L3 cache RAMs are placed in retention while the DynamIQ™ cluster shared logic and the cores are powered down.

It is quicker for the cluster to enter and exit Memory retention mode as compared with going from Off to On mode or On to Off mode. This is because the L3 cache RAMs do not need to be cleaned, and in some circumstances the data reloaded as well.


Note

The *DynamIQ™ Shared Unit-110* (DSU-110) remains coherent when in Memory retention mode. Any snoop arriving is stalled while the DSU-110 automatically requests the cluster *Power Policy Unit* (PPU) to bring the cluster to an On mode to process the snoop. Although it is possible for components of the system to access the L3 cache RAMs while in retention, it comes at considerable time cost as the DSU-110 must be powered up to service the access. Therefore, when using this mode, Arm® strongly recommends that no other external coherent agents are active, for example cores external to the cluster, or other coherent devices.

5.3.5 Emulated off mode (OFF_EMU)

In the Emulated off mode, the cluster behaves logically if it were in the Off mode. However, the DynamIQ™ cluster shared logic remains powered including the L3 cache and snoop filter RAMs.

In this mode, the cluster behaves as if it were powered off for functional logic, but it allows the cluster to maintain debug context and access. On entering this mode, a Warm reset is applied to the cluster, resetting the functional logic but not resetting the debug logic. From the perspective of software running on the core, the cluster appears to be powered off.

5.3.6 Emulated memory retention mode (MEM_RET_EMU)

In Emulated memory retention mode, the cluster behaves logically if it were in the Memory retention mode (MEM_RET) except that the DynamIQ™ cluster shared logic remains powered. This means the L3 cache RAMs are in retention but the snoop filter RAMs and the rest of the *DynamIQ™ Shared Unit-110* (DSU-110) logic remains powered. Therefore, debug accesses to the cluster can be made.

5.3.7 Warm reset mode (WARM_RST)

Warm reset mode applies a Warm reset to the DynamIQ™ cluster shared logic in the cluster.


Note

- If any core is put into Warm reset mode, then the cluster must also put into Warm reset mode and the other cores must go into Warm reset mode or OFF mode.

- To apply a Warm reset to an individual core, you must program the corresponding *Power Policy Unit* (PPU) for the core.
- Warm reset mode is only expected to be used for resets triggered by a system level issue, such as a watchdog timeout.
- Warm reset mode can occur at any time with no guarantee of the state of the cluster. A request to transition to Warm reset mode is accepted immediately. Therefore, its effects on the core, complex, cluster, or the wider system are **UNPREDICTABLE** and a wider reset might be required. For example, if there were outstanding memory transactions at the same time as the reset, then unless the system interconnect is also reset then these transactions might complete after the reset when the cluster is not expecting them and cause a system deadlock.

5.3.8 Debug recovery mode (DBG_RECOV)

The Debug recovery mode can be used to assist debug of external reset events, such as a watchdog timeout. It allows the contents of the L3 cache RAMs, and the *Reliability, Availability, and Serviceability* (RAS) registers that were present before a Warm reset to be observable after the reset, as state information is preserved.

In Debug recovery mode, all DynamIQ™ cluster shared logic including the L3 cache RAMs is powered up.

The DSU-110 invalidates the L3 cache and snoop filter when there is a transition from an Off to an On mode. In Debug recovery mode, cache invalidation is disabled. This allows the contents of the L3 cache that were present before the reset to be observable after the reset. The contents of the L3 cache and snoop filter are preserved and are not altered on the transition back to the On mode.

Debug recovery mode can be entered from any other mode. The cluster *Power Policy Unit* (PPU) controls entry into this mode.

To preserve the RAS state and cache contents, a transition to the Debug recovery mode can be made from any of the current states. When in Debug recovery mode, the cluster and core PPUs apply a cluster-wide Warm reset . The RAS and cache state are preserved when the core transitions to the On mode.

Debug recovery mode is strictly for debug purposes. It must not be used for functional purposes, because correct operation of the cluster is not guaranteed when entering this mode.

Debug recovery mode can occur at any time with no guarantee of the state of the cluster. A request of this type is accepted immediately, therefore its effects on the core, cluster, or the wider system are **UNPREDICTABLE**, and a wider system reset might be required. For example, if there were outstanding memory system transactions at the time of the reset, then unless the system interconnect is also reset, these transactions might complete after the reset when the cluster is not expecting them and cause a system deadlock.

If the system sends a snoop to the cluster during this mode, then depending on the cluster state, two scenario can happen:

- The snoop might get a response and disturb the contents of the caches.
- The snoop might not get a response and cause a system deadlock.

In the following cases, it might not be possible to enter DBG_RECOV without a Cold reset of the cluster:

- When the cluster is in middle of a power transition which cannot complete because of the system hanging.
- When the cluster is in the middle of a clock gating transition on the **SCLK** Q-Channel and the following occur:
  - The Q-Channel does not guarantee the clock availability.
  - The transition cannot complete because of the system hanging.
- The cluster is in Warm reset .


Note

You must choose the correct operating mode corresponding to the L3 cache portions and L3 cache slices that were in use before Debug recovery mode.

After the cores and cluster have entered ON mode from DBG_RECOV, the logic has been reset but the RAM contents are preserved. However, because there could have been outstanding transactions that were partially complete at the time the reset was applied, the contents of the RAMs might be inconsistent. For example, the data RAMs might have been updated by the transaction but the tag RAMs have not. Another example is the snoop filter has been updated but the core caches have not. These inconsistencies can sometimes cause deadlocks or **UNPREDICTABLE** behavior if normal code is executed. Therefore, Arm recommends analyzing or saving the cache contents without executing normal software. For example, putting the cores into Debug state and executing cache debug operations from Debug state.

After the debug has completed, the whole cluster, and potentially other system components such as the system interconnect, must be reset before normal operation can resume. This should be a cluster Cold reset including the PPUs, using the **nRESET** signal, to ensure that no inconsistent state remains in the cluster.

5.4 L3 RAM power control

In addition to retention features, the *DynamIQ™ Shared Unit-110* (DSU-110) can further reduce static leakage power, using two powerdown features. Firstly, optionally power down of all but one of the L3 cache slices. Secondly, within each L3 cache slice, power down a portion of the L3 cache RAM that the cache slice contains.

5.4.1 L3 cache RAM powerdown

The L3 cache RAMs typically contribute to a large proportion of the total leakage power, particularly for large cache sizes. Therefore, it is beneficial to power down the RAMs when only

some of the L3 cache is required, but it also results in reducing cache capacity. Parts of the L3 cache RAM can be independently powered down to reduce RAM leakage power when not in use. L3 cache powerdown is controlled by the cluster *Power Policy Unit* (PPU).

The L3 cache RAM powerdown feature allows the RAMs to be powered down in groups of ways, giving options of 100%, 50%, or 0% of the L3 cache capacity. When a workload is making light use of the L3 cache, then this can be detected and the L3 cache capacity reduced without significant impact on the performance. For example, this can occur when the L3 cache has a relatively small memory footprint that mostly fits within the L2 cache.

Powering down a group of ways involves first cleaning and invalidating the cache lines that are held in those ways. This takes time and consumes dynamic power. Therefore, the decision to power down these ways should balance these costs against the power saved during the time spent in the lower power mode.

Cleaning and invalidating the L3 cache lines is performed by hardware in the background and does not prevent the cores from executing instructions.

L3 cache RAM powerdown can be used, irrespective of the number of cores that are powered on or active.

There are three methods to control the cache portions (SFONLY, ½ RAM, and FULL RAM operating modes) which can be based on cache performance. See the following sections in order of preference:

- 5.4.1.1 Setting automatic L3 cache power portion control on page 58
- 5.4.1.2 Setting CLUSTERPWRCTLR_EL1.PRTNRQ power portion control on page 59
- 5.4.1.3 L3 RAM power control using PPU static transactions on page 60

For information on operating modes, see 5.5 Cluster operating modes on page 62.

5.4.1.1 Setting automatic L3 cache power portion control

The DSU-110 contains hardware to automatically schedule L3 cache power portion requests based on hit and miss counts. The cluster uses the L3 cache hit and miss rates to try to balance the leakage savings from powering down the cache with the energy cost of DRAM accesses.

About this task

To enable automatic L3 cache power portion control:

Procedure

1. The *System Control Processor* (SCP) programs the *Power Policy Units* (PPUs) for dynamic transitions.
2. Software running on the core sets the threshold registers. Alternatively the SCP can program the threshold registers through the utility bus.
   The threshold registers (AArch64 and External versions) are:
   - IMP_CLUSTERL3DNTH0_EL1, CLUSTERL3DNTH0
   - IMP_CLUSTERL3DNTH1_EL1, CLUSTERL3DNTH1
   - IMP_CLUSTERL3UPTH0_EL1, CLUSTERL3UPTH0
   - IMP_CLUSTERL3UPTH1_EL1, CLUSTERL3UPTH1

   See 5.4.1.4 Calculating values for threshold registers on page 60 for information about calculating suitable values for these registers.
3. Software running on the core sets IMP_CLUSTERPWRCTLR_EL1.AUTOPRTN to a nonzero value. Alternatively the SCP can program the CLUSTERPWRCTLR register through the utility bus.

Results

This generates automatic cache power portion requests that are translated to an internal PACTIVE indicator between the cluster and the cluster PPU. The cluster PPU responds accordingly to these requests.

5.4.1.2 Setting CLUSTERPWRCTLR_EL1.PRTNRQ power portion control

Software running on the core can program CLUSTERPWRCTLR_EL1.PRTNRQ to directly control the L3 cache power portion power requests.

About this task

To enable CLUSTERPWRCTLR_EL1.PRTNRQ L3 cache power portion control:

Procedure

1. The *System Control Processor* (SCP) programs the *Power Policy Units* (PPUs) for dynamic operating mode transitions.
2. Software running on the core sets CLUSTERPWRCTLR_EL1.AUTOPRTN = 0.
3. Software running on the core sets the cache power portion requests by programming the CLUSTERPWRCTLR_EL1.PRTNRQ.
   To assist firmware in calculating L3 cache requirements, the cluster L3 cache hit and miss performance counters (IMP_CLUSTERL3HIT_EL1, IMP_CLUSTERL3MISS_EL1) are directly accessible from the cores.

Results

This generates automatic cache power portion requests that are translated to an internal PACTIVE indicator between the cluster and the cluster PPU. The cluster PPU responds accordingly to these requests.

5.4.1.3 L3 RAM power control using PPU static transactions

You can use a *System Control Processor* (SCP) to program the cluster *Power Policy Unit* (PPU) explicitly through the utility bus to control powerup and powerdown of parts of L3 cache RAMs, by setting operating and power modes.

Procedure

1. The SCP programs the PPUs for static transitions.
2. Software (typically running on an SCP) manually programs the cluster PPU.
   The cluster L3 cache hit and miss performance counter registers (CLUSTERL3HIT, CLUSTERL3MISS) are accessible through the utility bus. This is so that the SCP firmware can use its own algorithms, if necessary, to determine its own L3 cache RAM requirements.

5.4.1.4 Calculating values for threshold registers

The DSU-110 has hardware to automatically monitor the cache hit and miss rates and to schedule L3 cache power portion power requests based on these metrics. To use this hardware, Arm recommends suitable values are programmed into the threshold registers.

When an access misses in the cache then it must access DRAM through the system interconnect to fetch the data. The energy cost of this DRAM access is much greater than the energy cost of an L3 access. Therefore, it is more energy-efficient for an access to hit in the L3 cache. However, the L3 RAMs consume leakage power even when the L3 is not accessed. Some workloads do not cache well and therefore have a high L3 miss rate. Other workloads might fit mostly in L1 and L2 caches, therefore make very few L3 accesses. In both cases, the cost of the L3 leakage power might be greater than the cost of any additional DRAM accesses.

The DSU-110 has hardware to automatically monitor the cache hit and miss rates and to schedule L3 cache power portion requests based on these metrics.

The hardware periodically calculates the hit and miss rates, based on the setting in the IMP_CLUSTERPWRCTL_EL1.AUTOPRTN register. The period is configurable and depends on the frequency implemented for the architectural generic timer in the system. Setting a shorter time period allows better responsiveness to changing workloads. However, if it is too short then the cost of frequently resizing the cache might be too high.

At the end of each time period, the value in the IMP_CLUSTERL3HIT_EL1 and IMP_CLUSTERL3MISS_EL1 registers are compared against the values programmed in the threshold registers:

- IMP_CLUSTERL3DNTH0_EL1, CLUSTERL3DNTH0
- IMP_CLUSTERL3DNTH1_EL1, CLUSTERL3DNTH1
- IMP_CLUSTERL3UPTH0_EL1, CLUSTERL3UPTH0
- IMP_CLUSTERL3UPTH1_EL1, CLUSTERL3UPTH1

After the calculations are complete, the IMP_CLUSTERL3HIT_EL1 and IMP_CLUSTERL3MISS_EL1 registers are reset to zero. Depending on the number of L3 ways powered up, and the values in the hit and miss registers, and threshold registers, the following happens:

- If all L3 ways are powered, then when IMP_CLUSTERL3HIT_EL1 is less than IMP_CLUSTERDNTH0_EL1, a request is made to the *Power Policy Units* (PPUs) to power down half of the ways.
- If half of the L3 ways are powered, then:
  - When IMP_CLUSTERL3HIT_EL1 is less than IMP_CLUSTERDNTH1_EL1, a request is made to the PPUs to power down all of the ways.
  - When IMP_CLUSTERL3MISS_EL1 is greater than IMP_CLUSTERUPTH1_EL1, a request is made to the PPUs to power up all of the ways.
- If no L3 ways are powered, then when IMP_CLUSTERL3MISS_EL1 is greater than IMP_CLUSTERUPTH0_EL1, a request is made to the PPUs to power up half of the ways.

Arm strongly recommends that the threshold registers are programmed before enabling the automatic control. The optimum values to program the threshold registers depend on the system characteristics. A recommended set of values is shown below, and these assume there is no system cache therefore every L3 miss requires a DRAM access. These values require the following information:

L is the leakage power (in mW) of all the L3 cache RAMs. This is the L3 tag RAMs and the L3 data RAMs for all ways.

D is the energy (in mJ) required to read 1MB of data from DRAM. While the interconnect will use some energy to transport the request to the DRAM controller, this is typically small compared to the energy used in the DRAM. Therefore, Arm recommends that this value uses just the energy consumed by the DRAM itself. If the DRAM datasheet gives the energy required for a single access, then this value must be multiplied by the number of accesses required to read 1MB of data.

T is the time period (in seconds) that is programmed into the IMP_CLUSTERPWRCTL_EL1.AUTOPRTN register.

```
IMP_CLUSTERDNTH0_EL1 = 12288 * T * L / D
IMP_CLUSTERDNTH1_EL1 = 4096 * T * L / D
IMP_CLUSTERUPTH0_EL1 = 4096 * T * L / D
IMP_CLUSTERUPTH1_EL1 = 4096 * T * L / D
```

5.4.2 L3 cache slice powerdown

In addition to powering down the L3 cache RAMs, you can gain further leakage savings by powering down some of the L3 cache slice control logic as well. Control of powering up or powering down L3 cache slices is performed by the cluster *Power Policy Unit* (PPU).

The L3 cache is split into between one and eight cache slices, depending on configuration. Each cache slice contains a part of the L3 tags, the L3 data, and the snoop filter, split by address. If more than one cache slice is configured, then all but the last slice can be powered off, leaving the last slice handling all addresses.

If N cache slices are implemented, then when in this mode the cache only has 1/N of its total capacity. The slices also contain the snoop filter, therefore the snoop filter also has 1/N of its total capacity. Because of this, if more than approximately 1/N of the cores are powered on (assuming the L1 and L2 cache capacity is evenly distributed between cores), then the snoop filter might limit the usable size of L1 cache and L2 caches. The design is still fully functional, but performance might be limited. Therefore, Arm recommends using L3 cache slice powerdown, in a cluster that has multiple cores in it, but where only a single core is in use.

The process of powering up and powering down the L3 cache slices involves cleaning and invalidating a majority of the cache lines that are held in the L3 cache, and also most of the snoop filter contents. This in turn requires cleaning and invalidating the corresponding cache lines in the cores L1 and L2 caches so that they are consistent with the snoop filter (back-invalidation). This takes time and consumes dynamic power. Therefore, the decision to powerup and powerdown these cache slices should balance these costs against the power saved during the time spent in the lower power mode. Most of this process can be done in the background, and does not prevent the cores from executing during the operation. However, it will reduce the performance of the cores during this time. There will be a short period (of the order of a few thousand cycles, depending on cache and snoop filter sizes) during which any accesses to the L3 cache by the cores are stalled.

The L3 cache slice powerdown can be combined with the L3 cache RAM powerdown, so that only the logic and snoop filter of one cache slice is active, with no L3 cache capacity. This gives the largest leakage saving while still allowing one core to be active.

5.5 Cluster operating modes

An operating mode is a component-specific configuration of the power modes. For the *DynamIQ™ Shared Unit-110* (DSU-110), the operating modes differ in the number of slices that are active, and in the amount of L3 cache RAM that is active. The cluster *Power Policy Unit* (PPU) provides programming access to control the operating modes and the power modes. The DSU-110 supports up to six operating modes.

The cluster PPU can control how many L3 cache slices are active (powered up). The following table shows the operating modes for the L3 cache slices.

Table 5-2: Operating modes for L3 cache slices

Operating mode	Short name	Description
One slice	ONE SLICE	One slice is active (powered up). This slice resides in its own power domain.
All slices	ALL SLICE	All slices are active (powered up).

The cluster PPU can also control how much of L3 cache RAMs are active (powered up) in cache slices that are active. The following table shows the operating modes for the L3 cache RAMs.

Table 5-3: Operating modes for L3 cache RAMs

Operating mode	Short name	Description
Snoop filter only	SFONLY	The L3 cache data and tag RAMs in each cache slice are powered down.
Half L3 cache	½ RAM	One half of the L3 cache data and tag RAMs in each active slice are powered up.

Operating mode	Short name	Description
Full L3 cache	FULL RAM	All of the L3 cache data and tag RAMs in each active slice are powered up.

Note

- In Direct connect configurations, there are no operating modes.
- In the No L3 cache Present configuration, there are only L3 cache slice operating modes.

5.6 Power states for the cluster RAM instances

The cluster power mode controls the power states requested for the L3 data and L3 tag RAM instances.

The following two tables show the power state dependencies between the cluster *Power Policy Unit* (PPU) and those signaled on the *Power Control State Machine* (PCSM) output. They also show the internal power states for the L3 data cache and L3 tag RAM instances and the cluster and slice power domains.

Note

In the following two tables, N is the number of L3 cache slices.

The following table shows the power state dependencies for:

- L3 cache slice 0
- L3 cache slices 1 to N, when the ALL SLICE operating mode is selected.

Table 5-4: DSU-110 RAM power states for cache slice 0 and slices 1 to N when in ALL SLICE mode

Cluster power mode	Cluster PCSM PSTATE power mode	Operating mode	Power portion 1 L3 data and tag RAMs	Power portion 0 L3 data and tag RAMs. Also victim RAMs	Snoop filter and LTDB RAMs	PDSLICE power domain logic	PDCLUSTER power domain logic
ON, WARM_RST, DBG_RECOV, MEM_RET_EMU, OFF_EMU	ON	FULL RAM	on	on	on	on	on
		½ RAM[1]	off	on	on	on	on
		SFONLY[1]	off	off	on	on	on
FUNC_RET	FUNC_RET	FULL RAM	retention	retention	retention	on	on
		½ RAM	off	retention	retention	on	on
		SFONLY	off	off	retention	on	on

[1] Only applicable to ON and MEM_RET_EMU power modes.

Cluster power mode	Cluster PCSM PSTATE power mode	Operating mode	Power portion 1 L3 data and tag RAMs	Power portion 0 L3 data and tag RAMs. Also victim RAMs	Snoop filter and LTDB RAMs	PDSLICE power domain logic	PDCLUSTER power domain logic
MEM_RET	MEM_RET	FULL RAM	retention	retention	off	off	off
		½ RAM	off	retention	off	off	off
		SFONLY	off	off	off	off	off
OFF	OFF	Not applicable	off	off	off	off	off

Note

The power portions 0 and 1 each consist of 8 cache ways with 1 cache way for each of the 8 *Memory System Resource Partitioning and Monitoring* (MPAM) cache partitions. Therefore powering down power portion 1 powers down 1 cache way in each MPAM cache partition. For more information, see 7.3 L3 cache partitioning on page 100.

The following table shows the power state dependencies for the cache slices 1 to N for ONE SLICE operating mode:

Table 5-5: DSU-110 RAM power states for cache slices 1 to N for ONE SLICE operating mode

Cluster power mode	Cluster PCSM PSTATE power mode	Operating mode	Power portion 1 L3 data and tag RAMs	Power portion 0 L3 data and tag RAMs. Also victim RAMs	Snoop filter and LTDB RAMs	PDSLICE power domain logic	PDCLUSTER power domain logic
ON, WARM_RST, DBG_RECOV, MEM_RET_EMU, OFF_EMU	ON	FULL RAM	off	off	off	off	on
		½ RAM[1]					
		SFONLY[1]					
FUNC_RET	FUNC_RET	FULL RAM	off	off	off	off	on
		½ RAM					
		SFONLY					
MEM_RET	MEM_RET	FULL RAM	off	off	off	off	off
		½ RAM					
		SFONLY					
OFF	OFF	Not applicable	off	off	off	off	off

5.7 Cluster PPU mode transitions

The *DynamIQ™ Shared Unit-110* (DSU-110) supports transitions between power and operating modes. Each combination of power mode with an L3 cache slice and L3 cache RAM operating

mode forms a *Power Policy Unit* (PPU) mode, for example, ONE SLICE FULL RAM ON. Some power modes do not have associated operating modes, but these can also be referred to as PPU modes.

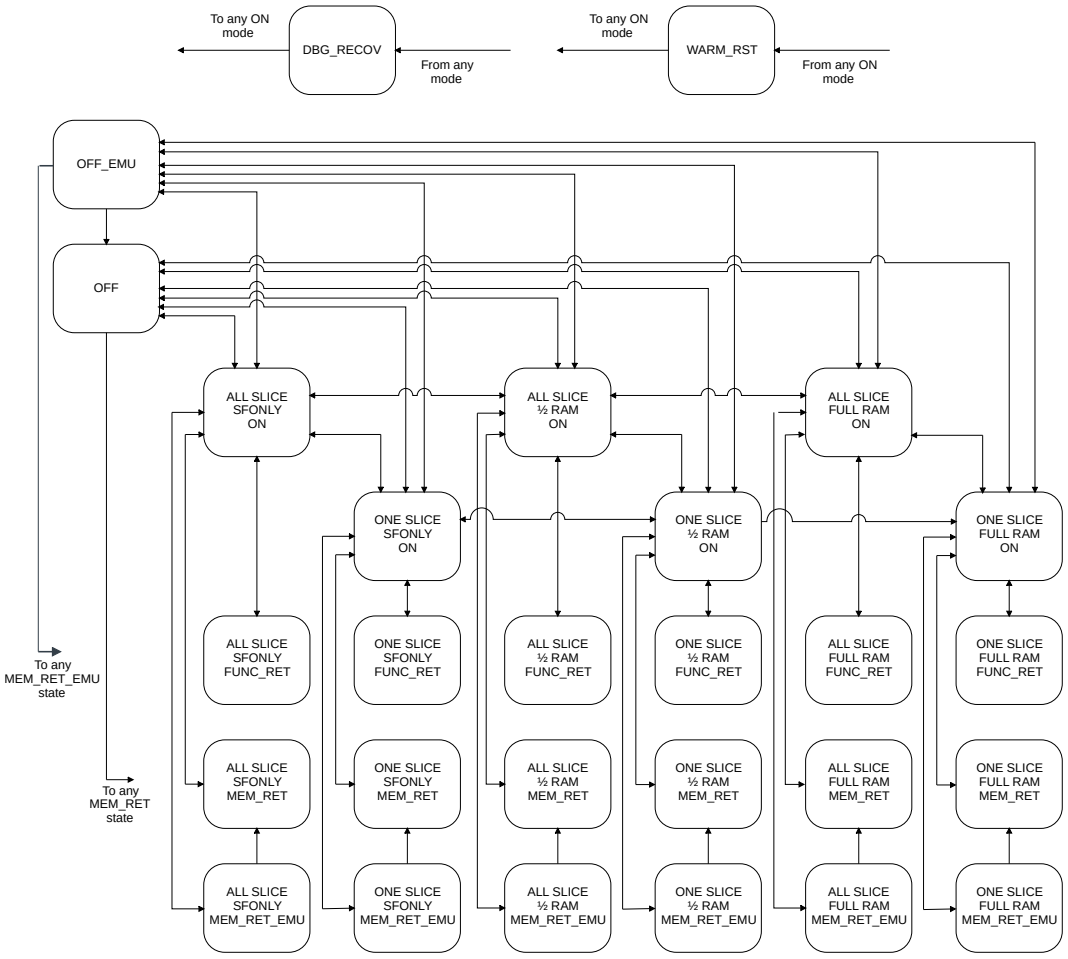
The cluster PPU controls transitions between the cluster PPU modes. Therefore, a *System Control Processor* (SCP) can program the PPU to go to any allowed PPU mode, and the PPU automatically makes the necessary transitions to reach the requested PPU mode.


Note

In all the following transition modes the cluster PPU controls which PPU mode the cluster enters at reset deassertion.

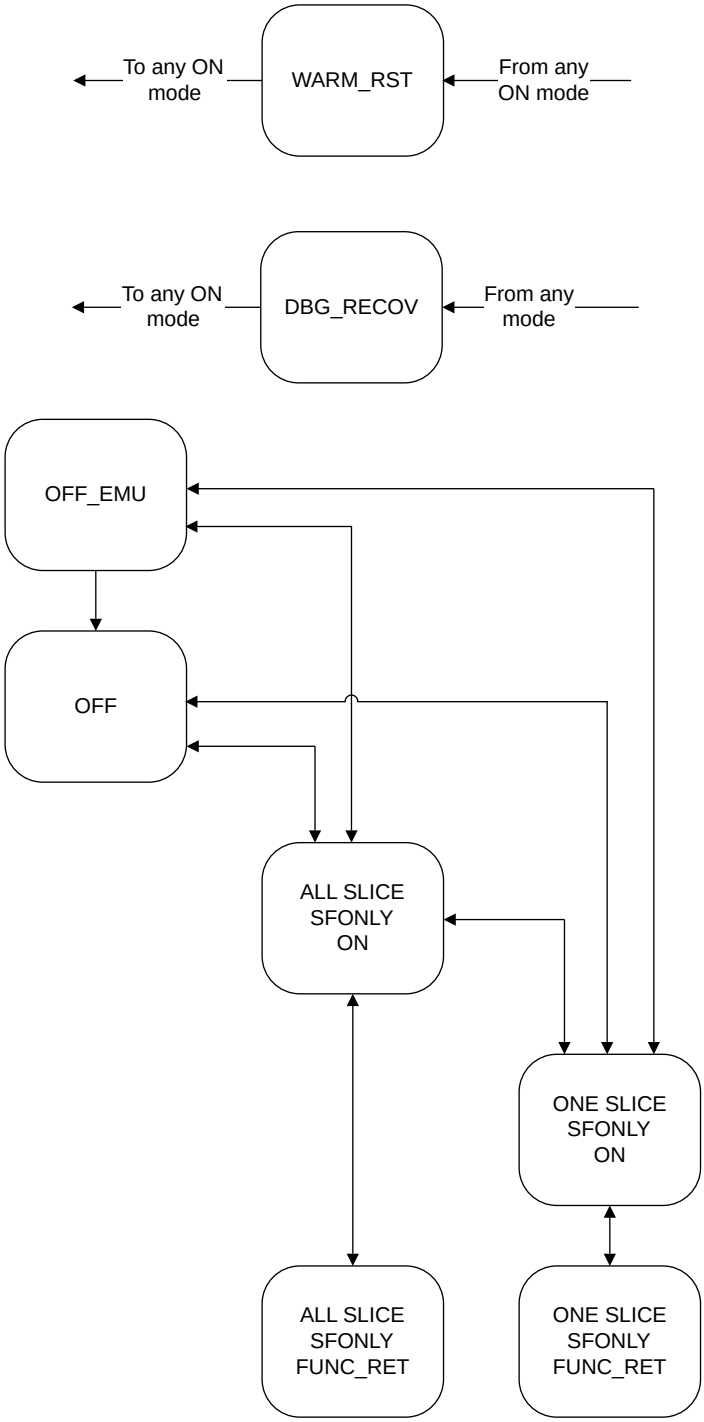
The following figure shows the supported PPU mode transitions for the DSU-110 DynamIQ™ cluster.

Figure 5-2: DSU-110 DynamIQ™ cluster PPU mode transitions



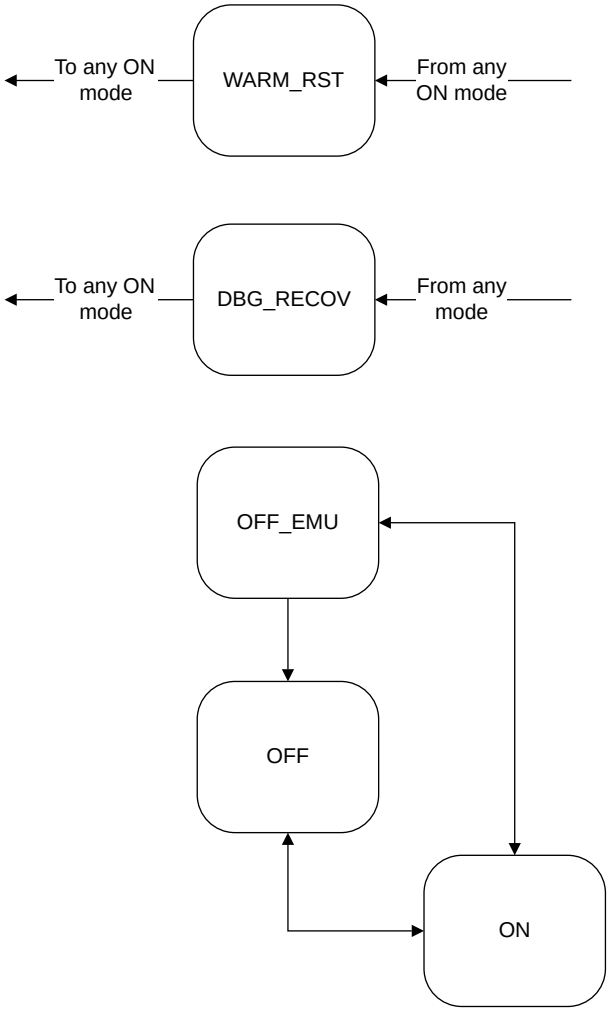
The following figure shows the supported PPU mode transitions for the DSU-110 DynamIQ™ cluster where L3 cache is not implemented.

Figure 5-3: DSU-110 DynamIQ™ cluster PPU mode transitions, no L3 cache



The following figure shows the supported PPU mode transitions for DSU-110 DynamIQ™ cluster when Direct connect is implemented.

Figure 5-4: DSU-110 DynamIQ™ cluster PPU mode transitions, Direct connect



ALL SLICE FULL RAM ON

In this PPU mode, all the DynamIQ™ cluster shared logic, including the L3 cache RAMs and snoop filters, is powered up and fully operational. When a transition to the On mode completes, the L3 cache and the snoop filter are accessible and coherent without requiring any software configuration.

ALL SLICE SFONLY ON and ALL SLICE ½ RAM ON

In these PPU modes, the DynamIQ™ cluster shared logic including snoop filter RAMs are powered up but half or all the L3 cache RAMs remain powered down. If the DSU-110 is implemented with no L3 cache, then only the ALL SLICE SFONLY ON mode is supported.

ONE SLICE SFONLY ON, ONE SLICE ½ RAM ON, and ONE SLICE FULL RAM ON

In these PPU modes, the DynamIQ™ cluster shared logic is powered up and fully operational. This is equivalent to ALL_SLICE operating mode, except that only one cache slice is powered up and active. The other slices are inactive and can be powered down.


Note

If the design is configured with only a single slice, then these modes are identical to the full slice modes.

SFONLY FUNC_RET, ½ RAM FUNC_RET, and FULL RAM FUNC_RET

In these PPU modes, the L3 cache RAMs and snoop filter RAMs are in retention. This means the RAMs are inoperable but their contents are retained. The rest of the DynamIQ™ cluster shared logic is operational. Therefore, if a request from a core or a snoop from the system is required to be serviced while in this mode, it is stalled until the cluster enters one of the On modes.

SFONLY MEM_RET, ½ RAM MEM_RET and FULL RAM MEM_RET

In these PPU modes, the L3 cache RAMs are in retention, but the rest of the DynamIQ™ cluster shared logic is powered down, apart from the PPUs. This is also known as Dormant mode. Because the L3 cache still contains data, if another agent in the system needs to snoop the cluster to access that data then the cluster needs to transition to an On mode before the snoop can proceed. As this transition takes a significant amount of time, Arm® recommends that MEM_RET is only used when other coherent agents are also idle.


Note

SFONLY MEM_RET is equivalent to OFF mode within the cluster but might have an effect on the wider system.

5.7.1 Rules governing cluster PPU mode transitions

For the cluster *Power Policy Unit* (PPU) mode transitions, there is a set of rules that governs the transitions between each PPU mode. There is no requirement for the *System Control Processor* (SCP) to explicitly consider these constraints when programming the cluster PPU.

The following rules govern all transitions between cluster PPU modes:

- When transitioning from OFF to ON, any supported operating mode can be targeted.
- Transitions between operating modes only happen in the ON power mode.
- Active slice changes do not happen at the same time as active RAM changes.
- Switching between SFONLY and FULL ON traverses ½ ON.
- The operating mode is maintained when moving from ON to FUNC_RET or MEM_RET power modes.


Note

For more information, see *Arm® Power Policy Unit Architecture Specification*.

5.7.2 PPU mode transition behavior

Where there is a transition between PPU modes, the DSU-110 cluster logic automatically performs a series of actions before accepting a new PPU mode.

The following table shows the allowed transitions between the cluster PPU modes and the associated actions.


Note

For each of the PPU mode transitions shown in the following table, additional actions (which are technology and implementation dependent) must be performed. These actions are carried out by partner implemented logic as part of the *Power Control State Machine* (PCSM). For more information about the PCSM, see 6.2 Power policy unit operation on page 80.

Table 5-6: Cluster domain PPU mode transition behavior

Start PPU mode	End PPU mode	DSU-110 behavior
OFF	ON	The L3 cache and snoop filter are initialized, and the cluster is brought into coherency with the rest of the system.
MEM_RET	ON	The snoop filter is initialized.
ON	FUNC_RET	If there is any ongoing memory access, the request is denied. Access to the L3 cache RAMs is blocked. Once in FUNC_RET any new transaction to the cache is stalled until there is a return to ON mode.
FUNC_RET	ON	Access to the L3 cache is allowed.
FULL RAM ON	½ RAM ON	Relevant ways in L3 cache are cleaned and invalidated.
½ RAM ON	SFONLY ON	Relevant ways in L3 cache are cleaned and invalidated.
SFONLY ON	½ RAM ON	Relevant ways in L3 cache are initialized.
½ RAM ON	FULL RAM ON	Relevant ways in L3 cache are initialized.
ALL SLICE ON	ONE SLICE ON	Relevant lines in L3 cache are cleaned and invalidated. Relevant snoop filter entries are emptied, causing back-invalidations to the cores if necessary.
ONE SLICE ON	ALL SLICE ON	Relevant lines in L3 cache are cleaned and invalidated. Relevant snoop filter entries are emptied, causing back-invalidations to the cores if necessary.
ON	OFF	If there is any ongoing core or cluster activity, the request is denied. L3 cache allocation disabled, and cleaned and invalidated. The cluster is removed from system coherency.
ON	MEM_RET	If there is any ongoing core or cluster activity, the request is denied.

For information and guidelines on implementing your PCSM, see *System Design* in *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

5.7.3 DebugBlock power modes

The DebugBlock supports only two power modes, ON, and OFF. There is no *Power Policy Unit* (PPU) in the DynamIQ™ Shared Unit-110 for the DebugBlock. Instead, the DebugBlock has a Q-Channel interface for providing power control to the DebugBlock power domain.

When the DebugBlock is in the Off mode, the DebugBlock does not initiate any accesses and all APB accesses to the DebugBlock receive a PSLVERR response.

5.8 Core PPU modes

Each core or complex in the DSU-110 DynamIQ™ cluster has a defined set of *Power Policy Unit* (PPU) modes and corresponding legal transitions between these modes. The PPU mode of each core can be independent of other cores in a cluster.

- As there are no operating modes for the cores in the DSU-110 DynamIQ™ cluster, the core PPU modes are equivalent to core power modes. However, they are called core PPU modes to be consistent with the terminology for programming the PPU.
- Some types of core might not support all the PPU (power) modes. See your core *Technical Reference Manual* (TRM) to see which PPU modes are supported.

The following table shows all the possible PPU modes supported by the cores.

Table 5-7: Core PPU modes

PPU mode	Short name	Description
On	ON	The core is powered up and active.
Functional retention	FUNC_RET	The core is fully powered and operational, but the *Vector Processing Unit* (VPU), if present, is OFF.
Full retention	FULL_RET	The core is in retention state. In this mode, only power that is required to retain register and RAM state is available. The core is non-operational. If the core supports functional retention and functional retention is enabled, then the core must be in Functional retention mode before it enters this mode.
Off	OFF	The core is powered down, either by using internal power switches or externally by the voltage regulator.
Emulated off	OFF_EMU	On mode, with Warm reset asserted. Debug state is retained and accessible.

PPU mode	Short name	Description
Debug recovery	DBG_RECOV	The RAM and logic are powered up. This mode is for applying a Warm reset to the cluster, while preserving memory and RAS registers for debug purposes. Both cache and *Reliability, Availability, and Serviceability* (RAS) state are preserved when transitioning from DBG_RECOV to ON. **Caution:** This mode must not be used during normal system operation.
Warm reset	WARM_RST	This Warm reset mode is used to reset the core. For more information about what is reset, see your core *Technical Reference Manual* (TRM).

5.8.1 Core PPU mode transitions

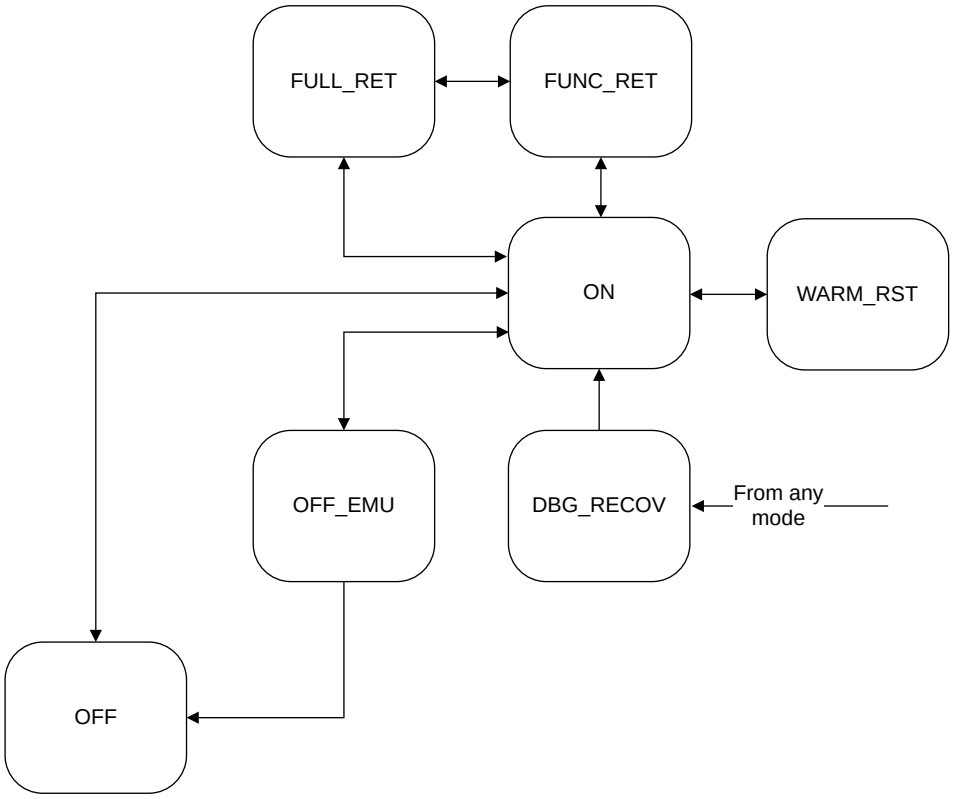
Each core supports a set of *Power Policy Unit* (PPU) mode transitions. These transitions are controlled by their respective core PPU. Therefore, a *System Control Processor* (SCP) can program a core PPU to go to any allowed PPU mode, and the PPU automatically makes the necessary transitions to reach the requested PPU mode.


Note

The core PPU controls which PPU mode the core enters at reset deassertion.

The following figure shows the permitted core PPU mode transitions.

Figure 5-5: Permitted core PPU mode transitions



On mode (ON)

In the On core PPU mode, the core is on and fully operational.

The core can be initialized into the On mode. When a transition to the On mode completes, all caches are accessible and coherent. Other than the normal architectural steps to enable caches, no additional software configuration is required.

Off mode (OFF)

In the Off core PPU mode, all core logic and RAMs are powered down. The domain is inoperable and all core state is lost.

The core L1 and L2 caches are disabled, cleaned and invalidated and the core is removed from coherency automatically on transition to an Off mode.

Any attempted debug access when the core domain is off returns an error response on the internal debug interface, indicating that the core is not available.

Functional retention (FUNC_RET) mode

In the Functional retention core PPU mode, a portion of the core, typically the *Single Instruction Multiple Data* (SIMD) and floating-point logic, is powered down while the remainder of the core is fully powered and operational.

If an instruction needs the logic that is powered down to complete execution, then the instruction is stalled until the core has transitioned to the On mode.

Full retention mode (FULL_RET)

In the Full retention core PPU mode, all core logic, and core cache RAMs are placed in retention. The core is non-operational but retains its state.

Full retention mode is typically used when the core is in *Wait for Interrupt* (WFI) or *Wait for Event* (WFE) state for an extended time. If a snoop, L1 or L2 cache maintenance operation or debug access occurs, then the core transitions to the On mode to process the access. Then it can transition back to Full retention mode without the core leaving corresponding WFI or WFE state.

Debug recovery mode (DBG_RECOV)

Debug recovery core PPU mode can be used to assist debug of external reset events such as watchdog timeout. It allows contents of the core L1 data and L2 caches that were present before the reset to be observable after reset. The contents of the caches are retained and are not invalidated on the transition back to the On mode.


Note

- You must only use Debug recovery mode for debug purposes. You must not use it for functional purposes as correct operation of the caches are not guaranteed when entering this mode.
- Debug recovery mode can occur at any time with no guarantee of the state of the core, therefore its effects on the core, cluster, or the wider system are **UNPREDICTABLE** and a wider system reset might be required. In particular, if there were outstanding memory system transactions at the time of the reset, then these might complete after the reset when the core is not expecting them and therefore might cause a system deadlock.

Emulated off mode (OFF_EMU)

In Emulated off mode core PPU mode, all core domain logic, and core RAMs are kept physically powered up. However, the functional logic is reset to emulate a powerdown scenario while keeping core Debug state and allowing debug access.

All Debug registers must retain their state and are accessible from the external debug interface. All other functional interfaces behave as if the core was in the Off mode.

5.9 Complex power management

Each core in a complex has its own *Power Policy Unit* (PPU), but there is no PPU for the shared logic or dedicated logic of a complex.

For a dual-core complex, the state of the shared logic is automatically managed based on the combined requirements of both of the cores. For example, if one core is powered down (Off mode), the shared logic remains in the On mode while the other core is also in the On mode. When the second core is also powered down, the shared logic powers down (Off mode).

5.9.1 Complex power modes

For a complex containing two cores, a *Power Policy Unit* (PPU) mode change to either of the cores requires some arbitration between the cores in the complex and the shared logic. This is carried out automatically by the complex bridge, without involvement of the core PPU.

For cores outside a complex with a CPU bridge, the power mode being requested by the PPU can be directly applied. When the CPU bridge interfaces with a complex, there might be multiple cores and some shared logic such as L2 cache and a *Vector Processing Unit* (VPU). The complex bridge handles system requests for power mode transitions by translating requests into the correct power mode transitions for a particular complex configuration.

The following table shows an example of all possible combinations of input requests and corresponding power transitions for a dual-core complex with a shared L2 cache and VPU.

Table 5-8: PPU mode and power domain states for a dual-core complex

Requested PPU mode		PCSM channel		
Core0	Core1	Core0	Core1	Shared logic
On	On	ON	ON	ON
On	Functional retention	ON	ON	ON
On	Full retention	ON	FULL_RET	ON
On	Debug recovery	ON	ON	ON
On	Emulated off	ON	ON	ON
On	Off	ON	OFF	ON
Functional retention	On	ON	ON	ON
Functional retention	Functional retention	ON	ON	FUNC_RET
Functional retention	Full retention	ON	FULL_RET	FUNC_RET
Functional retention	Debug recovery	ON	ON	ON
Functional retention	Emulated off	ON	ON	ON
Functional retention	Off	ON	OFF	FUNC_RET
Full retention	On	FULL_RET	ON	ON
Full retention	Functional retention	FULL_RET	ON	FUNC_RET
Full retention	Full retention	FULL_RET	FULL_RET	FULL_RET
Full retention	Debug recovery	FULL_RET	ON	ON
Full retention	Emulated off	FULL_RET	ON	ON
Full retention	Off	FULL_RET	OFF	FULL_RET
Debug recovery	On	ON	ON	ON
Debug recovery	Functional retention	ON	ON	ON
Debug recovery	Full retention	ON	FULL_RET	ON
Debug recovery	Debug recovery	ON	ON	ON
Debug recovery	Emulated off	ON	ON	ON
Debug recovery	Off	ON	OFF	ON

Requested PPU mode		PCSM channel		
Core0	Core1	Core0	Core1	Shared logic
Emulated off	On	ON	ON	ON
Emulated off	Functional retention	ON	ON	ON
Emulated off	Full retention	ON	FULL_RET	ON
Emulated off	Debug recovery	ON	ON	ON
Emulated off	Emulated off	ON	ON	ON
Emulated off	Off	ON	OFF	ON
Off	On	OFF	ON	ON
Off	Functional retention	OFF	ON	FUNC_RET
Off	Full retention	OFF	FULL_RET	FULL_RET
Off	Debug recovery	OFF	ON	ON
Off	Emulated off	OFF	ON	ON
Off	Off	OFF	OFF	OFF

Deviating from the legal power modes can lead to **UNPREDICTABLE** results. You must comply with the dynamic power management and powerup and powerdown sequences, see your core Technical Reference Manual.

5.9.2 Power mode transition dependencies for a dual-core complex

When there are two cores in the same complex, the power modes of the two cores must be consistent with the power mode of the shared logic. The *Power Policy Units* (PPUs) have logic to ensure that these requirements are maintained automatically.

In some cases when both cores request a power transition at the same time, the PPU logic delays the transition of the second core until the first core has completed its transition.

There are some cases where a power transition on one core might require a power transition on the other core to take place before the first core can progress.

The following table describes the power mode transitioning dependencies between the cores in a dual-core complex.

Table 5-9: Complex core power mode dependencies

Core A power mode	Core B power mode	Core A dependency	Power mode dependency	PPU request
ON	FULL_RET or FUNC_RET	Core A carries out one of the following: • Makes a request from ON mode to OFF mode. • Makes a request from ON mode to OFF_EMU mode. • Requests a reset using the RMR.RR register bit field.	Core B must be in the ON power mode before core A can transition.	Core B automatically indicates that it must transition from FULL_RET or FUNC_RET mode to ON mode. The core PPU must request this transition for core B before core A transition can proceed.

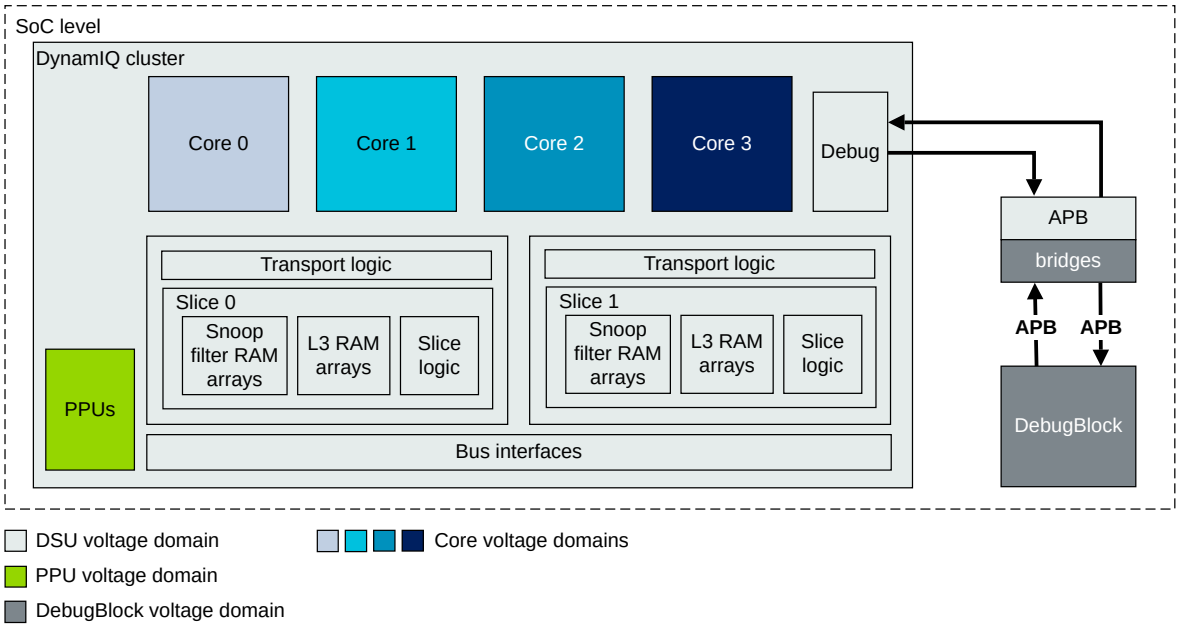
The **DEVPACTIVE*** inputs to the PPUs indicate that the core B must transition to ON mode, and so if the PPUs are in dynamic mode then this is handled automatically. If the PPUs are in static mode then the component programming the PPUs must ensure that this transition can happen. However, Arm recommends that FUNC_RET and FULL_RET modes are not used when the PPUs are in static mode, see 6.10 Core Full retention mode and static mode restrictions on page 97.

5.10 DSU-110 voltage domains

The *DynamIQ™ Shared Unit-110* (DSU-110) supports each core in the DSU-110 DynamIQ™ cluster being implemented in a separate voltage domain. There is also a separate voltage domain for the DSU-110 DynamIQ™ cluster itself.

The following figure shows the voltage domains in the cluster.

Figure 5-6: DSU-110 voltage domains



Having each core in a separate voltage domain allows *Dynamic Voltage Frequency Scaling* (DVFS) to be applied to each core.


Note

Implementing each core in a separate voltage domain is optional. Some implementations might choose to reduce cost by combining groups of cores into the same voltage domain.

The boundary of the core voltage domain is within the core hierarchy itself. For the core asynchronous bridges, part of the bridge is in the core voltage domain and part is in the cluster voltage domain.

The DSU-110 DynamIQ™ cluster is typically placed in the same voltage domain as the *System on Chip* (SoC) interconnect and other system components but can be placed separately if necessary. Similarly, the DebugBlock can be placed in a separate domain if necessary, provided the implementer places appropriate bridges on the APB interfaces between the DebugBlock and the cluster.

6. Power and reset control with Power Policy Units

This chapter describes how to control the power mode and reset behavior for the DSU-110 DynamIQ™ cluster, cores, and complexes using the *Power Policy Units* (PPUs).

6.1 The Power Policy Unit

Power mode control for the *DynamIQ™ Shared Unit-110* (DSU-110) is provided by the *Power Policy Units* (PPUs) that are integrated into the cluster. These PPUs control all the PPU modes for all components in the cluster.

A PPU is a standard component for abstracting software-controlled power domain policy to low-level hardware control signaling. There is one PPU for controlling the DSU-110 DynamIQ™ cluster power domain (PDCLUSTER). Also, each core has its own individual PPU for controlling its respective core power domain (for example, a PPU for PDCORE0 and a PPU for PDCORE1). This includes any cores included as part of a complex.

A component in the system such as a *System Control Processor* (SCP) can program the PPUs through the utility bus to set the required power policy. The PPUs control the low-level details of powering up, powering down, and resetting domains as necessary to implement the requested policy. The hardware performs any actions to reach the requested power mode, such as gating clocks, cleaning and invalidating caches, or disabling coherency.

- Although the cluster and each core in the cluster has their own PPU, the shared logic of a complex does not have a dedicated PPU. Instead, power management of the complex is controlled as a combination of the PPUs for the cores it contains. See Table 5-8: PPU mode and power domain states for a dual-core complex on page 74.
- The cluster and all the core PPUs are provided as part of the DSU-110.
- The implementation process automatically creates the PPU for the cluster and each core PPU, and connects these into the DSU-110 DynamIQ™ cluster. Each PPU has a set of memory-mapped control registers which is accessed using the utility bus.

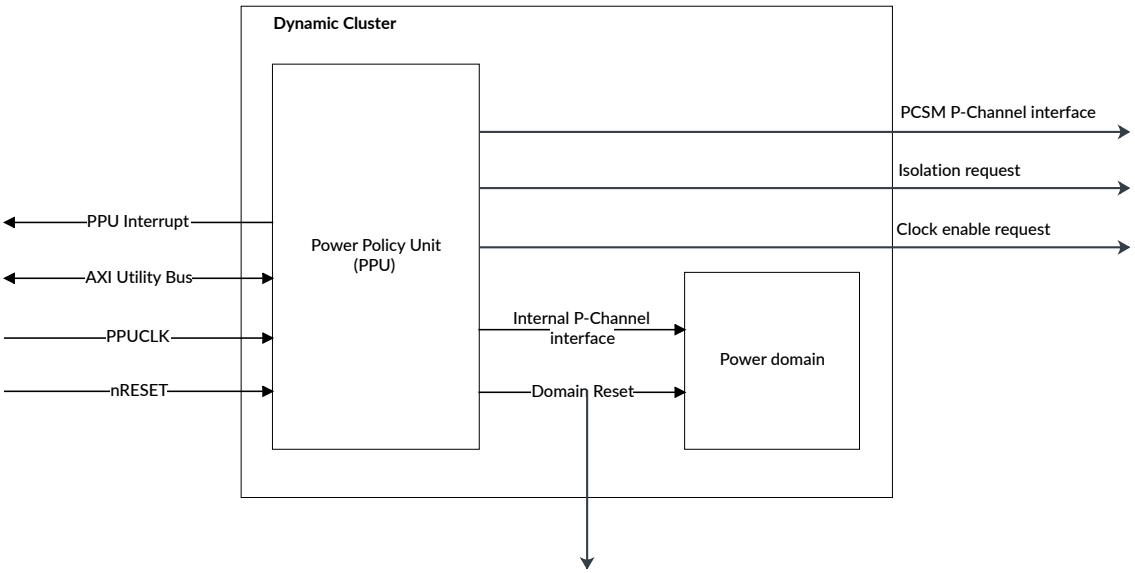
The PPUs:

- Abstract away the underlying mechanics of power state machine control of the DSU-110. This allows the external power manager to focus on the power modes it wants to achieve without being concerned about low-level control.
- Can provide autonomous control of power modes depending on the requirements of the cluster, for example the number of hits into the L3 cache.

PPUs can provide autonomous control of power modes with a range of modes.

The following figure shows the DSU-110 PPU interfaces. All interfaces are external to the DSU-110 apart from the Device Control interface, which has signals that both connect to the internal logic of cluster, and signals that are exported outside of the cluster.

Figure 6-1: DSU-110 PPU interfaces



All PPUs have the following main interfaces:

Software interface

The programming interface for the PPU registers is accessed through the external utility bus. These registers are programmed with the high-level policy and configuration.

Device control interface

The Device control interface is the internal interface that connects to each of the cluster and core power domains.

Note

Some of the device control interface signals are exported outside of the DSU-110 to allow control of other components that might be in the same power domain.

The interface provides low-level device control and ensures device quiescence. The interface comprises:

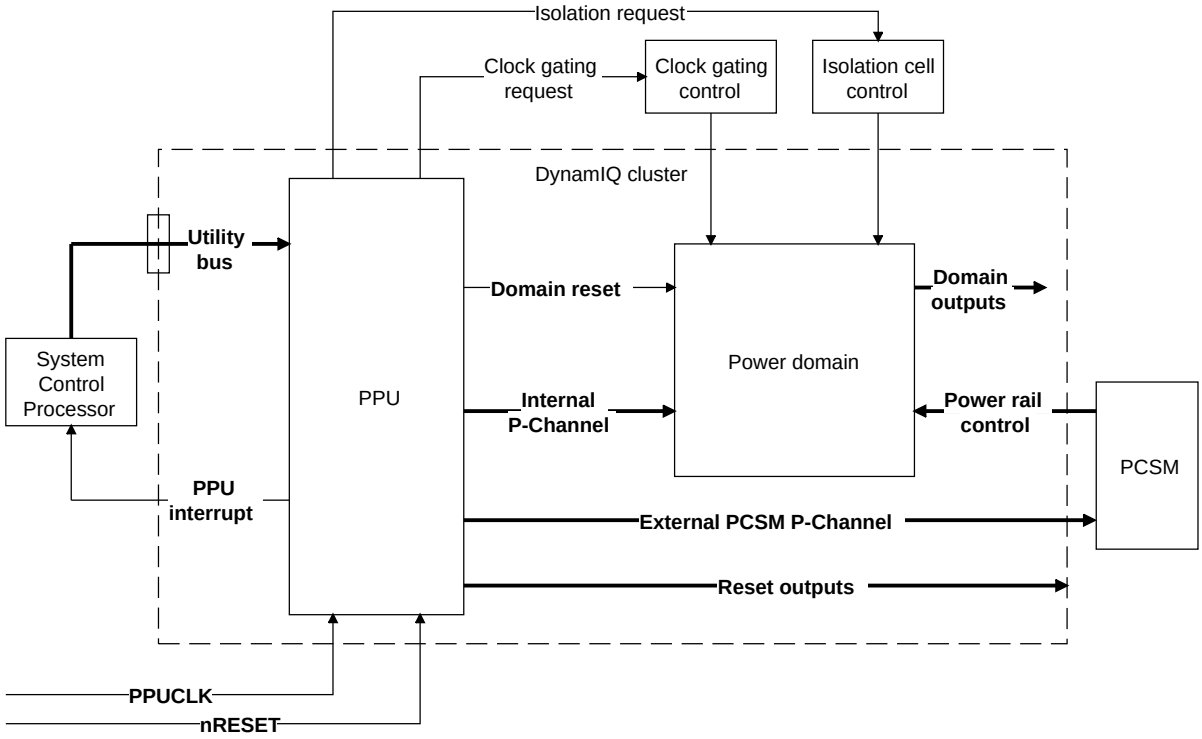
- The device interface, which consists of a P-Channel interface, see *AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces*.
- The device control interface, which includes clock enables, resets, and isolation control.

PCSM interface

The Power Control State Machine (PCSM) interface is an external interface for controlling low-level technology-specific power switch and retention controls. You must connect a PCSM to each interface as part of the DSU-110 implementation. There are separate PSCM interfaces for each core instantiated in the cluster, and a separate PCSM for the DSU-110 DynamIQ™ cluster itself.

The following figure shows a high-level illustration of how the PPU and PCSM controls connect to each other, and to a power-gated domain. The dotted lines indicate the implementation-dependent components and signal connections.

Figure 6-2: DSU-110 PPU connections to a power-gated domain



All the PPUs contained within the DSU-110 are pre-built and are based on configurations of the CoreLink PCK-600 Power Policy Units and comply with the PPU architecture specification version 1.1, see *Arm® Power Policy Unit Architecture Specification*.

6.2 Power policy unit operation

The *Power Policy Unit* (PPU) supports all the DSU-110 DynamIQ™ cluster power modes (ON, OFF, MEM_RET, FUNC_RET, OFF_EMU, MEM_RET_EMU, WARM_RST, DBG_RECOV), and operating

modes. It has extensive support to reflect the various combinations of logic and memory power states into which a domain can be set.

Your software can program a PPU to set a PPU mode in one of two ways:

Static policy

The PPU is programmed to request a specific PPU mode for the power domain. The hardware request to the core or cluster is only made once the core or DSU-110 indicates it is ready to enter this PPU mode.

When a PPU is using static policy to manage power state transitions, this is called static power state manangement.

Dynamic policy

Sets a minimum mode, so the PPU can autonomously change the PPU mode at or above this mode depending on hardware inputs. The upper limit for the range of power modes is ON. The upper limit for the range of operating modes is All slices mode and all RAM instances are active.

When a PPU is using dynamic policy to manage power state transitions, this is called dynamic power state management.


Note

For general use, Arm® recommends using dynamic policy as this gives the most automation and quickest response times to requested power mode changes. However, there are situations where more explicit control is required, such as debugging, and for these situations a static policy might be necessary.

Each PPU contains a state machine representation of its supported PPU mode transitions. For example, the cluster PPU has the PPU mode transitions for the cluster, see 5.7 Cluster PPU mode transitions on page 64. Therefore, a PPU can be programmed to target any supported PPU mode and the route taken follows the permissible route, passing through any intermediate PPU modes.

Each of the PPUs has an interrupt output signal that indicates events such as the completion of power mode transitions and the completion of operating mode transitions. For the cluster, this signal is **CLUSTERPPUIRQ** and for the cores these signals are **COREPPUIRQ[<CN>]**, where CN is the core instance number.

For the *DynamIQ™ Shared Unit-110* (DSU-110), a PPU is programmed by the *System Control Processor* (SCP) through the DSU-110 Utility bus. The SCP programs the PPU mode or range of PPU modes that it wants the DSU-110 DynamIQ™ cluster to enter based on the current system requirements.

The requested PPU mode (power mode and operating mode) is programmed using registers within the PPU. The role of the PPU is to handle the logical operation of a power domain therefore ensuring that the power domain can enter a new power mode safely.

The PPU and the power domain communicate through the device P-Channel. This device P-Channel is internal to the DSU-110. The communication is both from the power domain to the

PPU and from the PPU to the power domain. For example, communication between the power domain and the PPU could include:

- The power domain indicating to the PPU when the domain needs to enter a higher power mode to complete a function. For example, bringing the L3 cache from a memory retention state to an On state to respond to a cache access.
- The power domain indicating to the PPU when the domain could enter a lower power mode.

The **PACTIVE** signal of the P-Channel is used to communicate this information to the PPU.

The PPU can also drive the communication to the power domain. For example, when a request is made to go to a higher power mode, the PPU requests that the domain enters the new power mode. The domain can then accept or deny this new power mode.

The *Power Control State Machine* (PCSM) is responsible for handling functional power requirements, for example controlling power switches to the domain, isolating power supplies, and retention controls. The PPU communicates to the PCSM through an external PCSM P-Channel interface. The P-Channel handshake between the PPU and the PCSM is there to request the specific power rail status change required. It also ensures that the power change happens at the correct time in the PPU power management sequence.

For more information on PPU operation, see *Arm® Power Policy Unit Architecture Specification*. For information on system design considerations when designing the PCSM, see *System Design* in *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

6.2.1 Implicit resets from power modes

Certain power modes include an implicit internal reset of the powered off logic. This internal reset is managed by the PPU mode and does not require an external signal to be asserted or explicit programming of the *Power Policy Unit* (PPU).

For example, if a power domain is in the Off power mode this includes a Cold reset of the logic that was powered off, where both functional logic and debug logic is reset.

The Emulated off power mode includes a Warm reset of the logic that was emulated as powered off, where the functional logic is reset but the debug logic is not reset.

6.2.2 nRESET sequence

Asserting **nRESET** causes all the cluster and core logic to be Cold reset, using the *Power Policy Units* (PPUs). Each PPU has its own set of reset output and input signals, which are internal to the cluster, and connect to the core and cluster logic. Each PPU is responsible for resetting its associated core or cluster logic during **nRESET**.

The following sequence of events occurs when **nRESET** is asserted:

1. **nRESET** is asserted, placing the PPUs under reset. The PPU internal reset outputs are LOW, so the cluster and cores are also reset.
2. **nRESET** is deasserted:

- The PPUs are now active and can start logical operation.
- The cluster and cores are held in reset by the PPUs:
  - If a power domain is required to be in MEM_RET, the PPU does the *Power Control State Machine* (PCSM) handshake to enter MEM_RET. There is no device P-Channel handshake as the logic is OFF and under reset. See figure *Transitions from OFF to MEM_RET with a P-Channel PPU* in *Arm® Power Policy Unit Architecture Specification*.
  - Otherwise, the power domain is OFF and is held in reset.

3. Software programs the PPUs to enter the desired power mode. Typically this is ON.
4. The system continues.

6.2.3 Initial cluster operating mode

When using dynamic power state management for the cluster and the cluster moves from the OFF power mode to the ON power mode, the cluster *Power Policy Unit* (PPU) is requested to initialize the cluster into the ALL SLICE, FULL RAM operating mode.

If you want to initialize the cluster into a different operating mode:

1. Configure the cluster PPU to use a static operating policy.
2. Program the cluster PPU to request the operating mode required.
3. Either use your *System Control Processor* (SCP) or software running on a core in the cluster to program the Cluster Power Control Register, CLUSTERPWRCTLR. The CLUSTERPWRCTLR register is programmed to configure the cluster to request the preferred operating mode for the cluster.
4. The cluster PPU operating mode control can then be programmed to use dynamic operating mode management.

When dynamic power state management is used to control when the cluster moves from the MEM_RET power mode to the ON power mode, the cluster PPU is requested to initialize the cluster into the operating mode that was used for the MEM_RET power mode. The values of the CLUSTERPWRCTLR register and the associated threshold registers reflect the state of the registers when the MEM_RET power mode was entered. For example, if the cluster was in MEM_RET power mode and ONE SLICE, FULL RAM operating mode, then the cluster PPU requests that the cluster enters ON power mode, ONE SLICE, FULL RAM operating mode. This means that the dynamic operating mode request should request the most appropriate initial operating mode for the cluster, based on the memory retention operating mode settings.

6.3 Utility bus accesses

All the *Power Policy Unit* (PPU) control and data registers are accessed using the memory-mapped Utility bus. The Utility bus is implemented as a 64-bit AMBA AXI5 slave port.

Accesses to PPU registers over the Utility bus must be either 32-bits or 64-bits. Any other sized access gets a SLVERR response from the bus.

There is no access to these registers directly from the cores. Instead, you must provide a memory mapped address for the cores to access the Utility bus through the interconnect. The registers for the cluster PPU and each of the core PPUs are grouped on separate 64KB page boundaries allowing access control to be enforced by a *Memory Management Unit* (MMU).

You can only access PPUs by Secure accesses on the Utility bus. Accesses to these registers with the Non-secure bit set are treated as **RAZ/WI**.

6.4 Cluster PPU mode control

The *Power Policy Units* (PPUs), that are integrated into the cluster, control all the PPU modes for all components in the cluster. There is one PPU for the DSU-110 DynamIQ™ cluster which is responsible for controlling the PPU modes of the cluster.

A component such as a *System Control Processor* (SCP) can program the cluster PPU through the utility bus to set the required power policy. The cluster PPU controls the low-level details of powering up, powering down, and resetting domains as necessary to implement the requested policy. The hardware performs any actions to reach the requested power mode, such as gating clocks, cleaning and invalidating caches, or disabling coherency.

6.4.1 External cluster PPU registers

The *Power Policy Unit* (PPU) registers for the DSU-110 DynamIQ™ cluster are only accessible from memory-mapped accesses on the utility bus. You can only access these registers from the Secure address space.

The summary table provides an overview of all the cluster PPU registers in the DSU-110. Individual register descriptions provide detailed information. These register descriptions are configuration of the PPU architecture, see *Arm® Power Policy Unit Architecture Specification* for more details.

- The values for the cluster PPU registers are based on a typical multi-core cluster configuration, but these values might vary for different cluster configurations.
- The cluster PPU registers are treated as **RAZ/WI** if a Non-secure access is made to them. Any address that is not documented is also treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- The base address for the cluster PPU registers is `0x030000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 6-1: Cluster PPU register summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	–	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	–	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	–	32-bit	Power Status Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x010	PPU_DISR	—	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	—	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	—	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	—	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	—	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	—	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	—	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	—	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	—	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	—	32-bit	Additional Interrupt Status Register	Yes
0x040	PPU_IESR	—	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	—	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	—	32-bit	Functional Retention RAM Configuration Register	Yes
0x054	PPU_FULRR	—	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	—	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	—	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	—	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	—	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	—	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	—	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	—	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	—	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	—	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	—	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	—	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	—	32-bit	PPU Peripheral Identification Register 0	Yes
0xFE4	PPU_PIDR1	—	32-bit	PPU Peripheral Identification Register 1	Yes
0xFE8	PPU_PIDR2	—	32-bit	PPU Peripheral Identification Register 2	Yes
0xFEC	PPU_PIDR3	—	32-bit	PPU Peripheral Identification Register 3	Yes
0xFF0	PPU_CIDR0	—	32-bit	PPU Component Identification Register 0	Yes
0xFF4	PPU_CIDR1	—	32-bit	PPU Component Identification Register 1	Yes
0xFF8	PPU_CIDR2	—	32-bit	PPU Component Identification Register 2	Yes
0xFFC	PPU_CIDR3	—	32-bit	PPU Component Identification Register 3	Yes

6.4.2 Encodings for cluster power and operating modes

The *Power Policy Unit* (PPU) registers, for example PPU_PWPR, use power mode and operating mode encodings to set various conditions. For example, register bitfields PPU_PWPR.PWR_POLICY and PPU_PWPR.OP_POLICY require these values.

The following table shows the power mode encodings for the DSU-110 DynamIQ™ cluster. The **CLUSTERPCSMSTATE[15:8]** signals are unused in the DSU-110 DynamIQ™ cluster, and must be tied LOW.


Note

In the following table:

- PCSMPSTATE[3:0] refers to **CLUSTERPCSMPSTATE[3:0]**
- PPUHWSTAT[15:0] refers to **CLUSTERPPUHWSTAT[15:0]**

Table 6-2: Power mode enumeration for the DynamIQ cluster

Power mode	PPU_PWPR.PWR_POLICY	PCSMPSTATE[3:0]	PPUHWSTAT[15:0]
OFF	0x0	0x0	0x0001
OFF_EMU	0x1	0x8	0x0002
MEM_RET	0x2	0x2	0x0004
MEM_RET_EMU	0x3	0x8	0x0008
FUNC_RET	0x7	0x7	0x0080
ON	0x8	0x8	0x0100
WARM_RST	0x9	0x8	0x0200
DBG_RECOV	0xA	0x8	0x0400

The following table shows the DSU-110 DynamIQ™ cluster operating mode encodings for PPU_PWPR.OP_POLICY bit field.

Table 6-3: Operating mode encodings for PPU_PWPR.OP_POLICY bit field

Active slices	Active RAMs		
	Snoop Filter Only (SFONLY)	½ L3 cache (½ RAM)	Full L3 cache (FULL RAM)
One (ONE SLICE)	0x0	0x1	0x3
All (ALL SLICE)	0x4	0x5	0x7

The following table shows the DSU-110 DynamIQ™ cluster operating mode encodings for **CLUSTERPCSMPSTATE[7:4]** and **CLUSTERPPUHWSTAT[23:16]**.


Note

In the following table:

- PCSMPSTATE[7:4] refers to **CLUSTERPCSMPSTATE[7:4]**
- PPUHWSTAT[23:16] refers to **CLUSTERPPUHWSTAT[23:16]**

Table 6-4: Operating mode enumeration for the DSU-110 cluster

Operating mode	Short name	PPU_PWPR.OP_POLICY	PCSMPSTATE[7:4]	PPUHWSTAT[23:16]
One slice, snoop filter only	ONE SLICE, SFONLY	0x0	0x0	0x01
One slice, ½ L3 cache	ONE SLICE, ½ RAM	0x1	0x1	0x02
Not used	-	0x2	-	-
One slice, full L3 cache	ONE SLICE, FULL RAM	0x3	0x3	0x08
All slices, snoop filter only	ALL SLICE, SFONLY	0x4	0x4	0x10
All slices, ½ L3 cache	ALL SLICE, ½ RAM	0x5	0x5	0x20
Not used	-	0x6	-	-
All slices, full L3 cache	ALL SLICE, FULL RAM	0x7	0x7	0x80


Note

In Direct connect configurations, where there is no *Snoop Control Unit* (SCU), none of the operating modes that are listed in tables Table 6-3: Operating mode encodings for PPU_PWPR.OP_POLICY bit field on page 86 and Table 6-4: Operating mode enumeration for the DSU-110 cluster on page 87 are supported. For this configuration, the operating mode must be programmed to 0x0.

The following table shows for each operating mode which L3 memory system variants are supported.

Table 6-5: Supported operating modes for different L3 memory system variants

Operating mode	PPU_PWPR.OP_POLICY	Default configuration (with L3 cache and SCU)	Direct connect (No L3 cache and no SCU)	No L3 cache present (SCU present)
One slice, snoop filter only	0x0	Supported	Not supported	Supported
One slice, ½ L3 cache	0x1			Not supported
One slice, full L3 cache	0x3			Not supported
All slices, snoop filter only	0x4			Supported
All slices, ½ L3 cache	0x5			Not supported
All slices, full L3 cache	0x7			Not supported

For information on these registers, see *Arm® Power Policy Unit Architecture Specification*.

6.5 Core power mode control

There are separate *Power Policy Units* (PPUs) for each of the cores in the DSU-110 DynamIQ™ cluster.

A component such as a *System Control Processor* (SCP) can program each of the core PPUs using AXI transactions to the utility bus to set the appropriate power policy. The core PPU controls the low-level details of powering up, powering down, and resetting domains as necessary to implement the requested policy. The hardware performs any actions to reach the requested power mode, such as gating clocks, flushing caches, or disabling coherency. The power mode of each core can be

changed independently of other cores in the cluster. There is no restriction on the order that cores are powered on or off, with respect to the other cores.

6.5.1 External core PPU registers

Each core *Power Policy Unit* (PPU) in the DSU-110 DynamIQ™ cluster has an individual set of *Power Policy Unit* (PPU) registers. Each set of registers is identical, and are memory-mapped onto the utility bus at different base addresses. You can only access the PPU registers for each core from the Secure address space.

The summary table provides an overview of all the core PPU registers in the DSU-110. Individual register descriptions provide detailed information. These register descriptions are a configuration of the PPU architecture, see *Arm® Power Policy Unit Architecture Specification* for more details.


Note

- The values for the core PPU registers are based on a typical multi-core cluster configuration, but these values might vary for different cluster configurations.
- The core PPU registers are treated as **RAZ/WI** if a Non-secure access is made to them. Any address that is not documented is also treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- The base address for the core PPU registers is `0x<n>80000`, where n is the core instance number. For example, for core 0 the PPU base address is `0x080000` and for core 1 the PPU base address is `0x180000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 6-6: Core PPU register summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	–	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	–	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	–	32-bit	Power Status Register	Yes
0x010	PPU_DISR	–	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	–	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	–	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	–	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	–	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	–	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	–	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	–	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	–	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	–	32-bit	Additional Interrupt Status Register	Yes
0x040	PPU_IESR	–	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	–	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	–	32-bit	Functional Retention RAM Configuration Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x054	PPU_FULRR	–	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	–	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	–	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	–	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	–	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	–	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	–	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	–	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	–	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	–	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	–	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	–	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	–	32-bit	PPU Peripheral Identification Register 0	Yes
0xFE4	PPU_PIDR1	–	32-bit	PPU Peripheral Identification Register 1	Yes
0xFE8	PPU_PIDR2	–	32-bit	PPU Peripheral Identification Register 2	Yes
0xFEC	PPU_PIDR3	–	32-bit	PPU Peripheral Identification Register 3	Yes
0xFF0	PPU_CIDR0	–	32-bit	PPU Component Identification Register 0	Yes
0xFF4	PPU_CIDR1	–	32-bit	PPU Component Identification Register 1	Yes
0xFF8	PPU_CIDR2	–	32-bit	PPU Component Identification Register 2	Yes
0xFFC	PPU_CIDR3	–	32-bit	PPU Component Identification Register 3	Yes

6.5.2 Encodings for core power modes

The core *Power Policy Unit* (PPU) register bit field PPU_PWPR.PWR_POLICY encodes the supported power modes for the cores.

The following table shows the encodings for the core power modes.

In the following table:

- PCSMPSTATE[3:0] refers to **CORE<CN>PCSMPSTATE[3:0]**, where CN is the core instance number
- PPUHWSTAT[15:0] refers to **CORE<CN>PPUHWSTAT[15:0]**, where CN is the core instance number

Table 6-7: Power mode enumeration for the cores in the DSU-110 DynamIQ™ cluster

Power mode	PPU_PWPR.PWR_POLICY	PCSMPSTATE[3:0]	PPUHWSTAT[15:0]
OFF	0x0	0x0	0x0001
OFF_EMU	0x1	0x8	0x0002
FULL_RET	0x5	0x5	0x0020
FUNC_RET	0x7	0x7	0x0080

Power mode	PPU_PWPR.PWR_POLICY	PCSMPSTATE[3:0]	PPUHWSTAT[15:0]
ON	0x8	0x8	0x0100
WARM_RST	0x9	0x8	0x0200
DBG_RECOV	0xA	0x8	0x0400

The **CORE<CN>PCSMSTATE[15:4]** value is 0x000.

For information on the PPU registers, see 6.4.1 External cluster PPU registers on page 84 and 6.5.1 External core PPU registers on page 88.

Related information

2.7 Core, complex, and processing element numbering on page 30

6.6 Programming sequences for the cluster and the core

Example *Power Policy Unit* (PPU) programming sequences are provided for both the cluster and the cores. One of these sequences uses the static mode policy to demonstrate programming using this policy. However, because static power management can require considerable activity from the System Control Processor, Arm strongly recommends that you use dynamic power management for normal operation of the cluster.

6.6.1 Programming sequence to bring the cluster and cores from Off to On mode

Use the following steps, to program the *Power Policy Unit* (PPU) for the DSU-110 DynamIQ™ cluster and each of the cores to request a change of PPU mode from Off mode to On mode.

About this task

This task is using the PPU static policy to request a single mode transition. You can use it as a simple example for initial powerup or debug. However, for normal use cases, see 6.6.3 Programming sequence for an interrupt controller to control transitions between On and Off mode on page 92.

- Steps 2 and 4 are only required if you need to know when the power transition has completed. Otherwise they can be omitted.
- This example programs the cluster power mode before the core power mode. It is possible to program the core power mode before the cluster power mode. However, the power mode transition of the core will not happen, and the cores will not reach the On power mode, until after the cluster has reached the On power mode.
- In this task, <y> is the core instance number.

Procedure

1. Write to the cluster register PPU_PWPR, address `0x030000`, value `0x00070008`.
   This sets the static power mode policy to ON and the static operating mode policy to ALL SLICE FULL RAM.
2. Poll the cluster PPU_PWSR register, address `0x030008`, until the value read matches the value written to the PPU_PWPR register.
3. Write to the core PPU_PWPR register, for core <y>, address `0x<y>80000`, value `0x00000008`. This sets the static power mode policy to ON.
4. Poll the core PPU_PWSR register for core <y>, address `0x<y>80008`, until the value read matches the value written to the PPU_PWPR register.

6.6.2 Programming sequence to bring the cluster and cores from On to Off mode

Use the following steps, to program the *Power Policy Unit* (PPU) for the DSU-110 DynamIQ™ cluster and each of the cores to request a change of PPU mode from On to Off.

About this task

This task is using the PPU static policy to request a single mode transition.


Note

- In this task, <y> is the core instance number.
- Steps 3 and 5 are only required if you must know when the power transition has completed. Otherwise they can be omitted.
- This example programs the cores before the cluster. It is possible to change the order and program the cluster before the cores. However the power mode transition will not take effect until all the cores have reached the Off mode.
- This example programs the PPUs after the core has executed the `WFI` instruction. It is possible to change the order and program the PPUs before the software executes `WFI` instruction. However, the power mode transition will not start until the core has executed the `WFI` instruction.

Procedure

1. Ensure your software running on the core sets the IMP_CPUPWRCTLR_EL1.CORE_PWRDN_EN bit to 1, then executes a `WFI` instruction.
   If the component programming the PPU needs to know when the software has completed this step, it can read the PPU_DISR.PWR_DEVACTIVE_STATUS field, or set the interrupt to occur when this action takes place. This field reads zero when the core is ready to powerdown.
2. Write to the core PPU_PWPR for core <y>, address `0x<y>80000`, value `0x00000000`.
   This sets the static power mode policy to OFF.
3. Poll the core PPU_PWSR register for core <y>, address `0x<y>80008`, until the value read matches the value written to the PPU_PWPR register.
4. Write to the cluster PPU_PWPR register, address `0x030000`, value `0x00000000`.
   This sets the static power mode policy to OFF.

5. Poll the cluster PPU_PWSR register, address `0x030008`, until the value read matches the value written to the PPU_PWPR register.

6.6.3 Programming sequence for an interrupt controller to control transitions between On and Off mode

Use the following steps to program the *Power Policy Units* (PPUs) for the DSU-110 DynamIQ™ cluster and each of the cores to power up the cluster and cores when the signal **COREWAKEREQUEST[<y>]** is asserted, and to power down automatically when software has finished running on the cores.

About this task

This task is using the PPU dynamic policy to request automatic transitions.


Note

In this task, <y> is the core instance number.

Procedure

1. Write to the cluster PPU_PWPR register, address `0x030000`, value `0x01000100`. This sets the dynamic power mode policy and the dynamic operating mode policy, with a minimum power mode of Off.
2. Write to the core PPU_PWPR for core <y>, address `0x<y>80000`, value `0x00000100`. This sets the dynamic power mode policy, with a minimum power mode of Off.
3. To power up core <y> or power down core <y>, see steps in the following table.

Choice	Step
To power up core <y>	Assert the **COREWAKEREQUEST[<y>]** signal. Software on the core sets the IMP_CPUPWRCTLR_EL1.CORE_PWRDN_EN bit to 1, then executes a `WFI` instruction. After all cores are powered down, the cluster powers down automatically, unless the IMP_CLUSTERPWRDN_EL1.PWRDN=1 or IMP_CLUSTERPWRDN_EL1.MEMRET=1.
To power down core <y>	


Note

- The signal **COREWAKEREQUEST[<y>]** is level sensitive.
- The upper limit for the range of power modes is On. The upper limit for the range of cluster operating modes is All slices mode and all RAM instances are active.

6.7 Explicit reset of cluster and cores and debug recovery

There are several reset scenarios for part, or all, of the DSU-110 DynamIQ™ cluster and the cores.

You must ensure that the following sequences are followed exactly.


Note

- The *Power Policy Units* (PPUs) and associated logic prevents unsupported transactions from occurring.
- The WARM_RST and DBG_RECOV power modes do not have an associated operating mode. Therefore before entering these power modes, the current cluster operating mode must be saved. This ensures that the same operating mode can be restored when leaving the power states.

The scenarios for resetting all or part of the DSU-110 DynamIQ™ cluster are:

Powerup (Cold) reset

This reset must be done the first time that the cluster is powered up. It resets parts of the *DynamIQ™ Shared Unit-110* (DSU-110) including the PPUs.

1. Assert the **nRESET** signal for a minimum of three **PPUCLK** cycles.
2. Deassert the **nRESET** signal.
3. Program the PPU for the cluster to On mode, see 6.6.1 Programming sequence to bring the cluster and cores from Off to On mode on page 90.
4. Program the PPU for each required core to On mode, see 6.6.1 Programming sequence to bring the cluster and cores from Off to On mode on page 90.

Core software initiated Warm reset of an individual core

1. Use software running on the core to program the RMR.RR register bit.
2. Execute a `WFI` instruction.

Core software initiated Cold or Warm reset of the cluster (excluding the PPUs)

For the Cold reset case, power is also removed from the cluster during this sequence.

1. Use software running on each core to set the MP_CPUPWRCTLR_EL1.CORE_PWRDN_EN bit.
2. Use software running on each core to execute a `WFI` instruction.
3. Program the PPU for each core to Off mode (Cold reset) or Emulated off mode (Warm reset).
4. Program the PPU for the cluster to Off mode or Emulated off mode.


Note

The cluster Off mode can only be entered if the cores are in Off mode.

Using WARM_RST mode to reset the cluster (excluding the PPUs).

This procedure can be used to recover from a watchdog timeout or similar situations.

1. Ensure that the cluster is in On mode and the cores are either in On mode, Off mode, or Emulated off mode. Read the PPU_PWSR for the cluster to determine the current cluster operating mode.
2. For any of the cores that are in the On mode, write to the core PPU_PWPR for core <y>, address `0x<y>80000`, value `0x00000009`. This sets the core to the WARM_RST power mode.
3. Write to the cluster PPU_PWPR, address `0x030000`, value `0x00000009`. This sets the cluster to the WARM_RST power mode.
4. Write to the cluster PPU_PWPR, address `0x030000`, value `0x000<p>0008`, where <p> is the operating mode value read in step 1. This sets the cluster to the ON power mode.
5. For each core that is in WARM_RST, write to the core PPU_PWPR register, for core <y>, address `0x<y>80000`, value `0x00000008`. This puts each core back to the ON power mode.

Reset of the cluster (excluding the PPUs), retaining cache contents for debug

This can be either a Warm reset or Cold reset , depending on the setting of the PPU_PTCR.DBG_RECOV_PORST_EN bit.


Note

The value of PPU_PTCR.DBG_RECOV_PORST_EN must be consistent across all PPUs (the cluster and all the cores) otherwise the results are **UNPREDICTABLE**.

---

1. Read the PPU_PWSR for the cluster and each core, to determine which cores are powered up and what is the current cluster operating mode.
2. For any cores that are already OFF, you must ensure they are in a static OFF, or in a LOCKED OFF state to ensure they do not power up during this process.
3. Check the cluster operating mode and ensure it is in a static configuration so that the operating mode does not change after this step.
4. Write to the core PPU_PWPR for core <y>, address `0x<y>80000`, value `0x0000000A`, for any cores that are either:
   - Not in the OFF mode.
   - Not in OFF_EMU mode if PPU_PTCR.DBG_RECOV_PORST_EN = 0.

   This sets the core to the DBG_RECOV power mode.
5. Write to the cluster PPU_PWPR, address `0x030000`, value `0x0000000A`. This sets the cluster to the DBG_RECOV power mode.
6. Write to the cluster PPU_PWPR, address `0x030000`, value `0x000<p>0008`, where <p> is the operating mode value read in step 1. This sets the cluster to the ON power mode.
7. For each core that is in DBG_RECOV, write to the core PPU_PWPR register, for core <y>, address `0x<y>80000`, value `0x00000008`. This puts each core back to the ON power mode.

6.8 Power mode dependencies between the core and the cluster

There are some dependencies between the *Power Policy Unit* (PPU) modes of core and the PPU modes of DSU-110 DynamIQ™ cluster to ensure that the correct operation is maintained.

The following table describes dependencies on the requested core PPU modes.

Table 6-8: PPU mode dependencies for core

Current core PPU mode	Requested core PPU mode	Cluster dependency	Effect on core
OFF or OFF_EMU	ON	The core can only transition to ON once the cluster is in ON, or FUNC_RET.	The core request stalls until the cluster has reached the appropriate state.
WARM_RST	ON	The cluster must have previously transitioned from ON to WARM_RST, and then from WARM_RST back to ON, before the core request can be accepted.	The core request stalls until the cluster has transitioned from WARM_RST to ON.
DBG_RECOV	ON	The cluster must have previously transitioned from ON to DBG_RECOV, and then from DBG_RECOV back to ON, before the core request can be accepted.	The core request stalls until the cluster has transitioned from DBG_RECOV to ON.

The following table describes dependencies on the requested DSU-110 DynamIQ™ cluster PPU modes.

Table 6-9: PPU mode dependencies for cluster

Current cluster PPU mode	Requested cluster PPU mode	Dependency	Effect on cluster
ON	MEM_RET or OFF	Not all cores are OFF.	Cluster PPU mode request is denied.
ON	OFF/OFF_EMU/ MEM_RET/ MEM_RET_EMU	If the *Accelerator Coherency Port* (ACP) interface is present and **SYSCOREQS** is asserted.	Cluster PPU mode request is denied.
ON	MEM_RET_EMU or OFF_EMU	Not all cores in OFF or OFF_EMU.	Cluster PPU mode request is denied.
ON	OFF	If a core has requested to leave Off mode while an L3 cache data clean and invalidate is in progress.	L3 cache clean and invalidate process is abandoned and cluster PPU mode OFF request is denied
WARM_RST	ON	Cores not in OFF, OFF_EMU, WARM_RST, or DBG_RECOV.	Cluster PPU mode request is denied.
DBG_RECOV	ON	Cores not in OFF, OFF_EMU, WARM_RST, or DBG_RECOV.	Cluster PPU mode request is denied.

6.9 ECC errors during power transitions

It is possible to get *Error Correcting Code* (ECC) errors in the L3 cache RAMs during a power transition. These ECC errors could happen during the software sequence shortly before the hardware sequence starts. Alternatively these ECC errors could happen during the hardware sequence, when the cache is cleaned and invalidated as part of the On mode to Off mode transition.

As part of the powerdown sequence, software on the core disables or reroutes interrupts away from the core before it executes the WFI instruction.

Therefore the core software cannot be interrupted to manage any RAS fault or error, which is either:

- Detected before the core powerdown procedure the WFI instruction and is not cleared or
- Detected after the core powerdown executes the WFI instruction.

Any RAS fault or error interrupt output from an active cluster prevents the cluster from powering down. This results in scenarios such as:

- The cluster is left powered up but the core software will be inactive.
- All requests from the cluster PPU to power down the cluster will be denied until the error interrupt is cleared in the cluster RAS registers.

As part of the cluster powerdown, the RAS fault and error interrupts status must be managed to avoid this situation.

Managing RAS fault and error interrupts during the cluster powerdown

To manage RAS fault and error interrupts during the cluster powerdown, the following two options can be executed:

Disabling the cluster RAS fault and error generation

1. Disable the generation of the cluster RAS fault and error interrupts by using the ERXCTLR_EL1 registers.
2. Clear any current RAS fault or error interrupts before the core powerdown executes the WFI instruction.

Re-route the cluster RAS fault or error interrupts

Re-route the cluster RAS fault or error interrupts to a separate system error management device as part of the powerdown procedure. This device, for example a *System Control Processor* (SCP), is responsible for clearing the cluster RAS interrupts using the utility bus if a fault or error is signaled.


Note

This approach is only possible if the system is designed to allow the RAS fault or error interrupt outputs to be re-routed to another component.

If all the cluster RAS fault and error interrupt outputs are disabled before the core powerdown, but the error detection is still enabled, then the following occurs:

- Any correctable errors are corrected.
- Any deferrable errors are deferred as part of the automatic cache clean and invalidation procedures.
- The *Error records* for these correctable and deferrable errors are lost once the cluster is powered down.
- If there is an uncorrectable error when the cluster is powering down, then this error is not signalled to the system. Therefore, this uncorrectable error might corrupt the system behaviour.

In some systems it can be preferable to disable the RAS fault and error interrupt subset generation. The following process shows an example of how this can be done:

- Disable the interrupts for correctable and deferrable errors.
  - ERXCTLR_EL1.CFI = 0
  - ERXCTLR_EL1.FI = 0
- Enable the error interrupt for uncorrectable errors.
  - ERXCTLR_EL1.UI = 1
- Cluster error interrupt outputs must be routed to the system error manager before executing the WFI instruction in the core powerdown procedure.

Using this approach, the cluster error interrupt outputs must be re-routed to the system error manager before executing the WFI instruction in the core powerdown procedure.

If an uncorrectable error occurs during the powerdown, the cluster will be powered on, but the core software will be inactive. Then the system error manager is responsible for clearing the cluster RAS interrupt register so that the powerdown procedure can proceed. To use this approach, the system must be designed to allow the cluster RAS error interrupts to be re-routed to the system error manager. The system error manager is able to identify where the uncorrectable error occurred in the cluster and clear the RAS error interrupt by accessing the cluster RAS registers using the utility bus.

6.10 Core Full retention mode and static mode restrictions

The use of Full retention (FULL_RET) mode for a core is not recommended when the *Power Policy Unit* (PPU) is programmed in static mode.

This is because when a utility bus transaction is made to a core that is in FULL_RET, the core must transition to ON to service the utility bus transaction. However in static mode the transition requires programming of the PPU using the utility bus, which is already in use. To avoid this dependency causing a deadlock, if the PPU is in static mode any utility bus access to a core in FULL_RET receives a SLVERR response.


Note

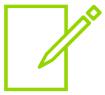
For both core and cluster power modes, Arm® recommends not using FULL_RET or FUNC_RET mode with static mode. This is because of the responsiveness of the system to wake from full retention or functional retention. It is expected that for most use cases that dynamic mode is used.

7. L3 cache

All the cores and complexes in the DSU-110 DynamIQ™ cluster share the L3 cache. The L3 cache is not included in a Direct connect cluster configuration.

The shared L3 cache of the DSU-110 (applies to non-Direct connect cores) provides the following functionality:

- A dynamically optimized cache allocation policy, which is typically exclusive. This cache allocation policy means that in normal use, a line is either in the cache of one or more cores (or complexes) or in the L3 cache, but not in both caches. Only Cacheable, shareable memory locations are allocated in the L3 cache. Non-shareable memory locations are not allocated in the L3 cache.
- Groups of cache ways can be partitioned and assigned to processes[2] by the *Memory System Resource Partitioning and Monitoring* (MPAM) architecture extension. Cache partitioning ensures that each process does not dominate the use of the cache to disadvantage other processes.
- Support for stashing requests from the ACP and CHI interfaces. These stashing requests can also target any of the L2 caches of cores or complexes within the cluster.
- *Error Correcting Code* (ECC) protection is provided on the cache data and tag RAMs.
- The cache can be implemented with up to eight cache slices, depending on the specified L3 cache size. Cache slices can increase the bandwidth of the L3 cache and improve the physical floorplan. Each cache slice consists of data, tag, victim, and snoop filter RAMs and associated logic.


Note

- On powerdown, the DSU-110 automatically performs cache cleaning, eliminating the need for software-controlled cache cleaning.
- For Direct Connect cores, CHI stashing into the core is normally supported.

7.1 L3 cache allocation policy

The DSU-110 L3 cache only allows Cacheable, shareable memory locations. Non-shareable memory locations are not allowed in the L3 cache. In configurations with both the *Accelerator Coherency Port* (ACP) and the 64-bit AXI5 peripheral port, memory in the peripheral port address range is also not allowed in the L3 cache.

The DSU-110 L3 cache uses a dynamically optimized cache allocation policy, which is typically exclusive. This cache allocation policy means that in normal use, a line is either in the cache of one or more cores (or complexes) or in the L3 cache, but not in both caches.

Exclusive allocation is used when data is allocated in only one core or complex. Inclusive allocation is sometimes used when data is shared between cores or complexes.

[2] A process is an instance of a computer program.

Consider the following scenario:

An initial request from core 0 allocates data in the L1 or L2 caches but not in the L3 cache.

When data is evicted from core 0, the evicted data is allocated in the L3 cache. The allocation policy of this cache line is still exclusive.

If core 0 refetches the line, it is allocated in the L1 or L2 caches of core 0 and removed from the L3 cache. The allocation policy of this cache line is still exclusive.

If core 1 accesses this line for reading, then it remains allocated in core 0. It is also allocated in both the core 1 and L3 caches. In this case, this line has an inclusive allocation because it is being shared between cores.

Related information

7. L3 cache on page 99

7.2 Available number of cache ways

The available number of cache ways in each cache slice depend on the L3 cache size that you choose to implement.

When selecting a power-of-two L3 cache size of 256KB, 512KB, 1024KB, 2MB, 4MB, 8MB, or 16MB each cache slice has 16 ways.

When selecting a non-power-of-two L3 cache size of 1536KB, 3MB, 6MB, or 12MB each cache slice only has 12 ways.

Related information

7.6 Cache slices and power portions on page 105
2.2 DynamIQ Shared Unit-110 configuration parameters on page 20

7.3 L3 cache partitioning

The L3 cache supports a partitioning scheme that alters the cache allocation and victim selection policy to prevent processes from using the entire L3 cache to the disadvantage of other processes.

Each transaction that is sent from the cores to the *DynamIQ™ Shared Unit-110* (DSU-110) is given a partition ID by the cores. The core software is responsible for determining the ID value for different transactions. The L3 cache partitioning control registers can be programmed to associate a partition ID value with a particular group of cache ways. Consequently, each transaction is only permitted to allocate into the L3 cache in one of the cache ways in the group defined by the partition ID of the transaction.

Cache partitioning is intended for specialized software where there are distinct classes of processes running with different cache accessing patterns. For example, two processes A and B run on separate cores in the same cluster and therefore share the L3 cache. If process A is more data-intensive than process B, then process A can cause all the cache lines that process B allocates to be evicted. Evicting these allocated cache lines can reduce the performance of process B.

The DSU-110 uses the *Memory System Resource Partitioning and Monitoring* (MPAM) architecture extension to partition the L3 cache. MPAM is an architecture extension that is designed to align the division of memory-system performance between software. MPAM therefore provides a wide range of optional features like cache partitioning, bandwidth partitioning, and the monitoring of processes. The DSU-110 only uses MPAM to partition the L3 cache. For more details about this architecture extension, see *Arm® Architecture Reference Manual Supplement Memory System Resource Partitioning and Monitoring (MPAM), for Armv8-A* (DDI 0598).

MPAM requires a system to pass around the MPAM ID, which cores attach to each memory-system transaction. The MPAM ID is referred to as a partition ID, and so the group of cache ways available for cache allocation with a particular partition ID value are referred to as a cache partition. While the structure of this MPAM ID is architectural, the configuration of its components are **IMPLEMENTATION DEFINED**. The DSU-110 uses this MPAM ID structure as follows:

- The MPAM_NS field, 1 bit, which indicates if this transaction is generated by the Secure or Non-secure state. A Non-secure transaction generated by the Secure state would have the MPAM_NS field set to 0 to indicate that it is generated by the Secure state.
- The PARTID, 6 bits, is the software assigned *Partition Identifier* for the current transaction. This supports up to 64 PARTIDs in Non-secure space, and 8 PARTIDs in Secure space. While a single process can use up to two PARTIDs, one for instruction fetches and one for data accesses, a single PARTID can also be used by multiple processes. The MPAM_NS bit indicates if this transaction is a Secure state or Non-secure state PARTID. If this transaction requires a Secure state PARTID then only the lower three bits of the PARTID are used.
- The PMG, 1 bit, identifies the *Performance Monitoring Group*, which is used by MPAM to provide the fine-grained monitoring of partitions, which is a feature that the DSU-110 does not use.

When the `L3_MPAM_STORAGE` parameter is enabled, then the L3 cache stores this MPAM ID information, which is retrieved on evictions.


Note

Storing the MPAM ID value in the L3 cache is only required if there is a downstream cache which supports MPAM, such as a system cache, that also provides MPAM support. If the system only requires valid MPAM ID values for read transactions, then this MPAM ID storage is not required.

If the MPAM IDs are not being stored, then any L3 evictions use the MPAM ID of the transaction that causes the eviction.


Note

If a transaction is mapped to a partition for which the MPAMCFG_CPBM setting has no portions set, then this transaction is not allocated into the L3 cache.

The partitioning of the L3 cache is done by groups of cache ways, and for the DSU-110 each group contains two ways, so a maximum of 8 partitions are supported. When programming the partitioning, the groups of L3 cache way pairs are referred to as portions.


Note

- The portions referred to when programming MPAM partitions are different from the L3 cache power portions. The term power portion is used to identify the L3 cache ways that are powered up and powered down for power saving purposes.
- The cache sizes that are not a power of two (1.5MB, 3MB, 6MB, and 12MB) support fewer portions than other cache sizes, because they have fewer available ways than the other cache sizes.
- If some cache ways are powered down (for more details, see 5.4.1 L3 cache RAM powerdown on page 57) then the number of ways are halved in each L3 cache partition portion. This reduction in cache ways can degrade the performance, when there are insufficient ways available to a process. Therefore, Arm recommends that caution is used when powering down cache ways while using cache partitioning.

One advantage of MPAM being an architectural extension is that it defines a generic mechanism to partition the L3 cache and can therefore be easily interacted with and configured by standard software.

Cache partitioning allows you to split the L3 cache into up to 8 separate partitions. You can overlap the cache portions defined for each partition. For instance, you might assign:

- Portions 0 to 3 (cache ways 0 to 7) to partition 0 (MPAM PARTID 0)
- Portions 0 to 7 (cache ways 0 to 15) to partition 1 (MPAM PARTID 1)

This would mean that while the processes assigned to partition 1 could use all the ways, the processes assigned to partition 0 could only use half of the ways.

The Secure and Non-secure states have separate control registers for programming the cache portions (cache ways) that are assigned to each partition ID. The Secure state partition control register, MPAMCFG_CPBM_s, has an additional non-architectural control bit that allows the Secure state partitioning programming to override the Non-secure state partitioning programming.

The MPAMCFG_CPBM_s register is used to program the cache portions that can be used by each of the different Secure state partition ID values.

When the MPAMCFG_CPBM_s.S_EXCL is set to 1, then any of the cache portions (and therefore the cache ways) used for a Secure partition ID are only permitted to allocate transactions from the Secure state. Therefore, if any of the Non-secure state partition IDs have been programmed to use these cache portions (that are marked as Exclusive for the Secure state), then Non-secure state transactions are not permitted to allocate into these L3 cache portions.

Related information

5.4.1 L3 cache RAM powerdown on page 57
7. L3 cache on page 99

7.4 Cache stashing

Cache stashing allows an external agent to request that a line is brought (or stashed) into a cache in the cluster.

Cache stashing can either be performed over the ACP interface or the CHI master interface. Stash requests can target either the L3 cache or any of the L2 caches of cores within the cluster. However, the available stashing bandwidth is likely to be higher when stashing to the L3 cache.


Note

If cores share a complex, then a stash request targeting the L2 cache is allocated into the shared L2 cache of this complex.

On the CHI interface, stash requests (snoops) into both the L2 and L3 caches are supported. The field, StashLPIDValid, indicates the target of the stash, as follows:

- If the field is clear, then the stash is directed to the L3 cache.
- If this field is set, then the stash is directed to an L2 cache of the core the StashLPID field specifies.

On the ACP interface, accesses are implicit stash requests into the L3 cache, by default. Signal **AWSTASHLPIDENS** indicates that a stash is targeting a L2 cache of a core within the cluster. In this case, signal **AWSTASHLPIDS[4:0]** indicates which core is being targeted.

The cluster always attempts to allocate a stash request, unless it is heavily utilized and does not have any free buffers. In this case, the cluster drops a stash request to avoid a potential system deadlock.

Performance Monitoring Unit (PMU) events (see 17.2 PMU events on page 203), in particular those events from `0x0500` to `0x0524`, indicates to software how successful the stashing has been. In other words, how many stash requests were received and how many of the received requests were dropped.

For cache stashing behaviour for Direct connect cores, see the Technical Reference Manual of that core.

Related information

7. L3 cache on page 99
17.2 PMU events on page 203

7.5 L3 cache data RAM latency

The DSU-110 L3 data RAM interface can be implemented with a configurable latency on the input and output paths.

The following options are available:

- Either a 1-cycle (the default) or 2-cycles write latency on the input path to the L3 data RAMs
- Either a 2-cycles (the default) or 3-cycles read latency on the output path from the L3 data RAMs
- A 2p write latency option on the input path, when the 3-cycles read latency is configured on the output path

---

Note

This 2p write latency also keeps the RAM input signals stable for an extra cycle, allowing an extra cycle of hold timing on the RAM inputs.

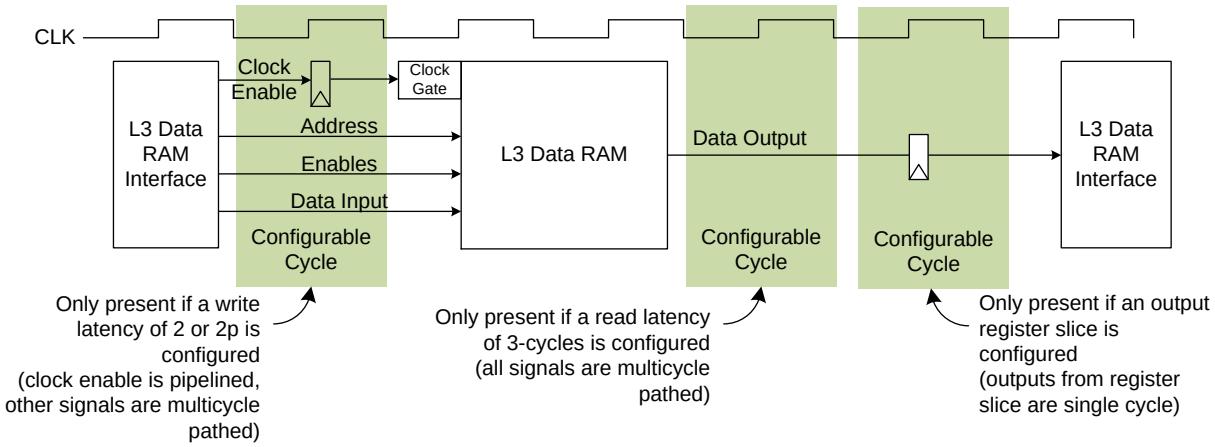
- An optional register slice on the output of the L3 data RAMs.

On the input paths, if a 2 or 2p write latency is requested then the RAM clock enable is pipelined and a multicycle path is applied to all other RAM input signals.

On the output paths, the 2-cycles read latency and 3-cycles read latency applies a multicycle path to all RAM output signals. The output of the optional register slice is single cycle and must never have a multicycle path applied.

The following diagram shows the L3 data RAM timing.

Figure 7-1: L3 cache data RAM latency



An increase in RAM latency increases the L3 hit latency, which reduces performance. For this reason, only use the 3-cycles read latency option if the RAM cannot meet the timing requirement

of the 2-cycles latency. But, if only the wire routing delay from the RAM to the SCU logic cannot meet this timing requirement, then use the register slice instead.


Note

Latency options are only specified for the L3 data RAMs, because the L3 tag RAMs and SCU snoop filter RAMs meet the 1-cycle input and 1-cycle output timing requirement.

The following table describes the impact on L3 data RAM performance with the different latency configuration parameters:

Table 7-1: L3 data RAM performance with different latency configurations

L3_DATA_WR_LATENCY	L3_DATA_RD_LATENCY	L3_DATA_RD_SLICE	L3 data RAM access cycles	L3 lookup bandwidth
1-cycle	2-cycles	No	2	Access every 2-clock cycles
1-cycle	3-cycles	No	3	Access every 3-clock cycles
1-cycle	2-cycles	Yes	3	Access every 2-clock cycles
1-cycle	3-cycles	Yes	4	Access every 3-clock cycles
2-cycles	2-cycles	No	3	Access every 2-clock cycles
2-cycles (including 2p)	3-cycles	No	4	Access every 3-clock cycles
2-cycles	2-cycles	Yes	4	Access every 2-clock cycles
2-cycles (including 2p)	3-cycles	Yes	5	Access every 3-clock cycles

Related information

2.2 DynamIQ Shared Unit-110 configuration parameters on page 20
7. L3 cache on page 99

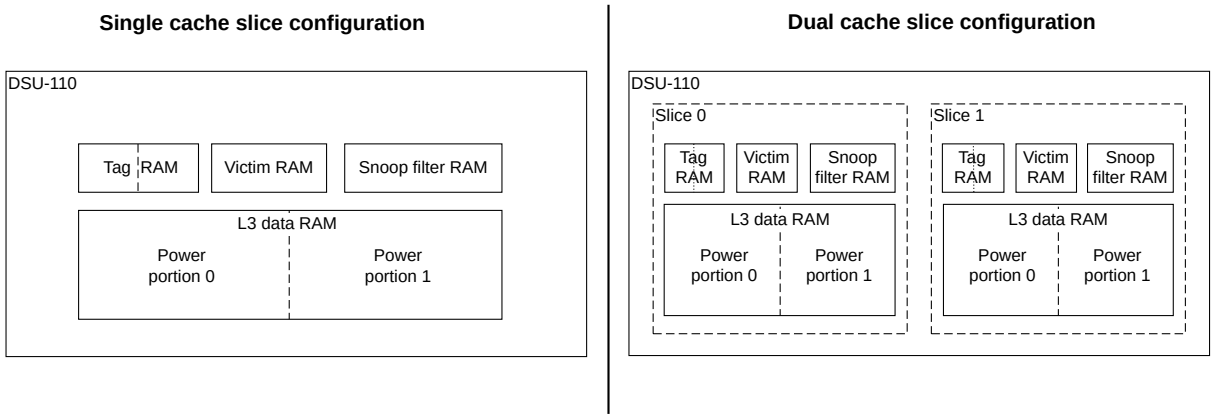
7.6 Cache slices and power portions

The L3 cache of the *DynamIQ™ Shared Unit-110* (DSU-110) can be divided into up to eight identical slices, each containing between 256KB and 2MB of the cache. A cache slice consists of the data, tag, victim, and snoop filter RAMs and associated logic. A power portion is a further subdivision of RAM in a cache slice.

For each cache slice, both the data RAM and tag RAM is subdivided into two power portions.

The following figure shows the differences between a single and a dual cache slice configuration.

Figure 7-2: Comparison between a single and dual L3 cache slice configuration



Splitting the L3 cache into slices provides the following advantages:

- Improving the physical floorplan when implementing the macrocell, by ensuring that the RAMs are located close to the logic that is controlling them.
- Increasing the bandwidth because the slices can be accessed in parallel.

Related information

7.2 Available number of cache ways on page 100
7.6.1 Cache slice and master port selection on page 106
5.4.2 L3 cache slice powerdown on page 61

7.6.1 Cache slice and master port selection

For an implementation with more than one cache slice, requests are sent to a particular slice depending on the address and the memory attributes.

The mapping from address to slice is not configurable, but the mapping from address to master port is configurable and can be independent from the slice mapping.

8. CHI master interface

You can use the *Coherent Hub Interface* (CHI) interface for either a coherent or non-coherent connection to your memory system. You can configure the *DynamIQ™ Shared Unit-110* (DSU-110) to have either one, two, or four master interface ports that use the AMBA 5 CHI (Issue E) protocol.


Note

All L3 memory system variants for the DSU-110 support the CHI Issue E protocol.

8.1 Multiple master configurations

You can configure the *DynamIQ™ Shared Unit-110* (DSU-110) to have one, two, or four CHI master interface ports, at build time configuration, to give a range of bandwidth options. Transactions from the cores are routed to one of the CHI interface ports based on the transaction type, memory type, and transaction address.

The following table summarizes how transactions are routed to based on the transaction type.

Table 8-1: CHI transaction routing

Transaction type	Routed to
Cacheable transactions	Master interface port number that is based on a hash of the transaction address.
Normal Non-cacheable transactions	Master interface port number that is based on a hash of the transaction address.
Device non-reorderable transactions	Master interface port 0 by default. You can configure the routing for Device non-reorderable transactions to select the master interface port number that is based on a hash of the transaction address.
Device reorderable transactions	Master interface port number that is based on a hash of the transaction address.
External snoop transactions	Snoop address is compared with the address hash for the master interface port that the snoop is sent or received on.
Distributed Virtual Memory (DVM) transactions	Always master interface port 0


Note

By default, transactions described in the table Table 8-1: CHI transaction routing on page 107 are directed to one of the master interface ports unless they match one of the peripheral port address ranges. However, if the **DEFAULTMP** signal is asserted at reset, then the mapping is inverted. Therefore all transactions, including DVM operations, go to the peripheral port except those that match the configured address ranges. These transactions that match the configured address ranges are sent to the main master interface ports instead.

Cacheable and Non-cacheable transactions

For Cacheable transactions and Normal Non-cacheable transactions, routing from the cores are based on the address of the transaction. A configurable hash of the transaction address selects which master interface port is used. See 8.1.1 Hashing for transaction distribution on page 109.

Device non-reorderable transactions

By default, Device non-reorderable transactions are always routed to master interface port 0 regardless of the address. Routing Device non-reorderable transactions to master interface port 0 ensures that the ordering requirements can be efficiently met, however this routing arrangement can be overridden. Therefore, all transactions to a given address use the same interface.

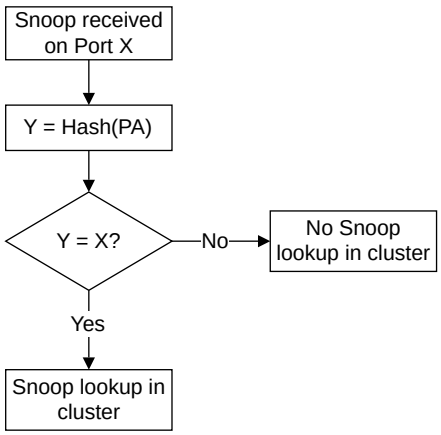
You can change the default routing, of sending Device non-reorderable transactions to master port 0, using the **DEVNRINTERLEAVE[1:0]** input signal as follows:

0b00	All Device non-reorderable transactions are sent to master interface port 0.
0b01	Device non-reorderable transactions are sent to any master interface port that is based on the same address interleaving as for non-Device transactions.
0b10	Reserved
0b11	Device non-reorderable transactions are sent to any master interface port based on the same address interleaving as for non-Device transactions. There is no downstream convergence of traffic, therefore the ReadReceipt or *Data Buffer ID* (DBID) is enough to guarantee global ordering.

External snoop transactions

The following figure shows how a snoop from the external memory system on one of the interface ports is handled. In this figure, PA is the physical address of the snoop.

Figure 8-1: External snoop handling on CHI master port



If there is no match, the response to the snoop is a cache miss and there is no lookup in the cluster. Therefore, when the external memory system sends snoops, it must either:

- Send the snoop to all the master ports. All but one of the snoops are guaranteed to miss, the remaining snoop might hit or miss depending on the state of the cache line in the cluster.
- Send the snoop only to the master interface port that is relevant for the address of the snoop. This behavior is normal operation for an external memory system that contains a snoop filter. The snoop filter indicates that the line is present in one of the masters.

The second method is more efficient, and therefore if multiple master interfaces are implemented, Arm® recommends that the external memory system includes a snoop filter. The snoop filter must be able to either:

- Track the exact master interface port.
- Calculate the correct master interface port based on the snoop transaction address.

DVM message transactions

DVM messages do not have a *Physical Address* (PA) that is associated with them, and therefore cannot be routed to the appropriate master interface port. To avoid processing DVM messages multiple times, the DSU-110 only processes them when sent to master port 0. Any DVM messages sent to master ports one to three are responded to but have no effect inside the cluster. Therefore, for best performance, Arm® recommends that your system is configured to only send DVM messages to master interface port 0.

Outgoing DVM messages are always sent on master interface port 0, unless the **DEFAULTMP** signal is asserted in which case all outgoing DVM messages are sent on the peripheral port.

8.1.1 Hashing for transaction distribution

When more than one CHI master interface port is implemented, the hashing to decide which transaction goes to which CHI master interface port is based on the *Physical Address* (PA) of the transaction.

The hash function masks the physical transaction address with a configurable mask, and then XORs all the resultant bits together. You can set configurable mask using the signal **MASTERINTERLEAVE** before the cluster leaves reset. In the following example:

- MASTERINTERLEAVE is the configurable mask value set by the **MASTERINTERLEAVE** input signal.
- ADDRESS is the PA of the transaction.
- PORT is the master interface port chosen.

```
MASKED_ADDR = ADDRESS[39:6] & MASTERINTERLEAVE[39:6]
For example, for two ports, PORT[0] = MASKED_ADDR[39] ^ MASKED_ADDR[38] ^ to
 MASKED_ADDR[7] ^ MASKED_ADDR[6]
For example, for four ports, PORT[1:0] = MASKED_ADDR[39:38] ^ MASKED_ADDR[37:36] ^
 to MASKED_ADDR[9:8] ^ MASKED_ADDR[7:6]
```

8.2 CHI features

AMBA defines a set of interface properties for the *Coherent Hub Interface* (CHI) interconnect. You must ensure that your system interconnect, where applicable, supports these properties.

The following table shows which of these properties the *DynamIQ™ Shared Unit-110* (DSU-110) supports, or requires the interconnect and system to support.

Table 8-2: CHI interconnect properties for the DSU-110

CHI property	Supported by the DSU-110	Interconnect support required
Atomic_Transactions	Yes if **BROADCASTATOMIC** is HIGH.	Yes if **BROADCASTATOMIC** is HIGH.
Cache_Stash_Transactions	Yes	Yes
Direct_Memory_Transfer	Yes	OPTIONAL. The DSU-110 supports this feature if required by your interconnect.
Direct_Cache_Transfer	Yes	OPTIONAL. The DSU-110 supports this feature if required by your interconnect.
Data_Poison	For non-Direct connect configurations yes, if cache protection is enabled. Cache protection is enabled by setting the configuration parameter `SCU_CACHE_PROTECTION`. For Direct connect configuration, as there is no L3 cache, the Data_Poison property is enabled, depending on if your core has cache protection enabled. See the *theodul.yaml configuration parameters* section in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual* for more information.	Yes if cache protection is enabled.
Data_Check	No	No
CCF_Wrap_Order	No. The DSU-110 sends data packets in any order.	No
Barrier_Transactions	No	No. The DSU-110 does not use these transaction types.
Data return from SC state	Yes	Not applicable
I/O de-allocation transactions (ROMI and ROCI)	No	No. The DSU-110 does not use these transaction types.
ReadNotSharedDirty transactions	Yes	Yes
CleanSharedPersist transactions	Yes if **BROADCASTPERSIST** is HIGH.	Yes if **BROADCASTPERSIST** is HIGH.

The following table shows the values for the CHI master interface values for the DSU-110.

Table 8-3: CHI master interface values for the DSU-110

CHI property	Value	Comment
Req_Addr_Width	52	If the cluster only contains cores that have a *Physical Address* (PA) width which is 48 bits or smaller, then this value is 48. If the cluster only contains cores that have a PA width which are 44 bits or smaller, then this value is 44.
NodeID_Width	11	-
Data_Width	256 bits	-

For more information on these features, see the *AMBA® 5 CHI Architecture Specification*.

8.3 CHI configurations

You can change the coherency configurations to suit your system configuration using the **BROADCASTCACHEMAINT** and **BROADCASTOUTER** input signals.

The following table shows the permitted combinations of these signals and the supported configurations in the *DynamIQ™ Shared Unit-110* (DSU-110), with a CHI bus.

Table 8-4: Supported CHI configurations

Signal	Feature			
	CHI non-coherent		CHI coherent	
	With no cache or invisible system cache	With visible system cache	With invisible system cache	With visible system cache
BROADCASTCACHEMAINT	0	1	0	1
BROADCASTOUTER	0	0	1	1

- A visible system cache requires cache maintenance transactions to ensure that a write is visible to all observers.
- An invisible system cache is one that does not require cache maintenance transactions to ensure that a write is visible to all observers. This is true even if those observers use different memory attributes.

The following table shows the key features in each of the supported CHI configurations.

Table 8-5: Supported features in the CHI configurations

Features	Configuration		
	CHI non-coherent		CHI coherent
	With no cache or invisible system cache	With visible system cache	
Cache maintenance requests on TXREQ channel	No	Yes	Yes
Snoops on RXSNP channel	No	No	Yes
Coherent requests on TXREQ channel	No	No	Yes

The input signals **BROADCASTTLBIINNER** and **BROADCASTTLBIOUTER** control the broadcasting of *TLB Invalidate* (TLBI) DVM messages to the external interconnect. The following table shows how the broadcast of the TLBI messages is controlled for the Inner and Outer Shareable domains depending on the configuration of **BROADCASTTLBIINNER** and **BROADCASTTLBIOUTER**.

Table 8-6: Control of Inner and Outer Shareable TLBI messages to the external interconnect

BROADCASTTLBIINNER	BROADCASTTLBIOUTER	Description
LOW	LOW	No TLBI transactions are broadcast outside the cluster.
LOW	HIGH	Outer Shareable TLBI transactions, TLBI {OS}, generate TLBI transactions that are broadcast from the cluster. No other TLBI instructions generate TLB transactions that are broadcast from the cluster.
HIGH	LOW	Invalid configuration
HIGH	HIGH	Inner Shareable TLBI instructions, TLBI {IS}, and Outer Shareable TLBI instructions, TLBI {OS}, generate TLBI transactions that are broadcast from the cluster.

8.4 Attributes of the CHI master interface

The read and write issuing capabilities of the CHI master interface depend on the configuration of the *DynamIQ™ Shared Unit-110* (DSU-110) at build time configuration, such as the number of L3 cache slices configured. For certain configurations, a maximum number of reads and writes can be up to 128 per master port.

The following table lists the read and write transaction capabilities of the CHI master interface.

Table 8-7: Attributes of the CHI master memory interface

Attribute	Value	Comment
Write issuing capability	Configuration dependent	This can range up to a maximum of 128 per master port, depending on configuration.
Read issuing capability	Configuration dependent	This can range up to a maximum of 128 per master port, depending on configuration.
Exclusive hardware access thread capability	Number of hardware threads	Each hardware thread can have one exclusive access sequence in progress.
Transaction ID width	12 bits	There is no fixed mapping between CHI transaction IDs and cores. Transaction IDs can be used for either reads or writes. **Note:** The source of the transaction is encoded in the LPID field, see Table 8-9: CHI LPID[4:0] bitfields on page 115.
Transaction ID capability	Configuration dependent	The transaction ID capability depends on the number of L3 cache slices configured, see the note following this table. There is never any ID reuse in CHI implementations, regardless of the memory type.

Attribute	Value	Comment
NodeID widths	11 bits	-
TXREQFLIT.RSVDC	0 bits	-
TXDATFLIT.RSVDC	0 bits	-
TXDATFLIT.DataCheck	0 bits	-

- For the write issuing and read issuing capabilities, the total issuing capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. For multiple-master configurations the percentage of total outstanding transactions each master can support is as follows:
  - If there is only one master port configured, then it can support the total number of outstanding transactions.
  - If there are two master ports configured, then each port can support up to 50% of the total outstanding transactions.
  - If there are four master ports configured, then each port can support up to 25% of the total outstanding transactions.

  Note

  The peripheral port can support up to 128 transactions, or the total number of outstanding transaction for the cluster if this is less.
- The issuing capability described in this table is the maximum for the whole cluster. If you want to achieve the maximum performance available, then you can use these values to size interconnect capabilities. However, this maximum issuing capability might not be reached by a single core on its own. It might need multiple cores generating heavy memory traffic simultaneously to reach the maximum value. The capabilities vary by core type, for example high-performance cores typically generate more transactions than balanced-performance cores. It can also vary by memory type, with typically a significantly lower limit for Device or Non-cacheable transactions than for Cacheable transactions.

8.5 CHI channel properties

The CHI master interface supports snoops from your external memory system. The *DynamIQ™ Shared Unit-110* (DSU-110) supports all snoop request types listed in the CHI Issue E protocol.

The following table describes the snoop capabilities and other CHI properties of the DSU-110.

Table 8-8: CHI channel properties

Property	Value	Comment
Snoop acceptance capability	Configuration dependent	The total snoop acceptance capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. Each master port can accept up to this overall limit, however it has more limited tracking of the SrcID field of the snoops. Therefore, if there are snoops outstanding from 15 different other components in the system, then any snoop from a 16th or further component will not be accepted. The number of snoops from each component is only limited by the total cluster acceptance capability.
DVM acceptance capability	Four per master port	The SCU can accept and process a maximum of four DVM transactions per master port from the system. Each of these four transactions can be a two part DVM message. The interconnect must be configured to never send more than four DVM messages to a CHI master interface port, otherwise the system might deadlock.
Snoop latency	Hit and miss latencies depend on configuration	Snoop latencies depend on how many master interfaces are configured, and if the snoops miss in the cluster, hit in the L3 cache, or hit in L1 or L2 caches of the cores. Snoops that hit in the L1 or L2 caches of a core have a higher latency. This latency depends on the type of core, and whether the hit is in the L1 or L2 cache. Typically the rate sustained is at least half that for the L3 cache bandwidth. Latencies can be higher if hazards occur or if there are not enough buffers to absorb requests.
	Miss	Dependent on build-time configuration
	DVM	Dependent on build-time configuration
Snoop filter	Supported	The cluster supports an external snoop filter in an interconnect. It indicates when clean lines are evicted from the cluster by sending Evict transactions on the CHI write channel. However there are some cases that can prevent an Evict transaction from being sent. Therefore you must ensure that you build any external snoop filter to handle a capacity overflow. When exceeding capacity, the snoop filter should send a back-invalidation to the cluster. Examples of case where evicts are not produced include: • Linefills that take External aborts. • Store exclusives that fail. • Mis-matched aliases.
Supported transactions	-	The DSU-110 supports all transaction types produced by the CHI protocol.

8.6 CHI transactions

CHI transactions are sent to a specific node in the interconnect depending on type of access, the address of the access, and settings in the system address map.

Addresses that map to an HN-F node can be marked as Cacheable memory in the translation tables, and can take part in the cache coherency protocol. Addresses that map to an HN-I or MN must be marked as device or Non-cacheable memory.

CHI TXREQ transactions include the *Logical processor ID* (LPID) field. This field uniquely identifies the logical core that generated the request transaction. The following table shows CHI LPID[4:0] bitfields:


Note

For a *Translation Lookaside Buffer* (TLB) translation table walk from a complex, the LPID information is only accurate to the granularity of the complex. Therefore, the LPID might indicate any *Processing Element* (PE) within the complex. You can determine a TLB translation table walk by the signal **TXREQSRCATTRMx[1:0]**=`0b10`.

Table 8-9: CHI LPID[4:0] bitfields

LPID[4:0] bit field	Description
[4]	Reserved
[3:0]	**0x0-0xB** Core instance number **0xC-0xD** Reserved **0xE** ACP request **0xF** Cache copy back

The following table shows the CHI read and write transaction types supported by the CHI-configured master port on the *DynamIQ™ Shared Unit-110* (DSU-110).


Caution

The following table is not applicable to configurations that use the Direct connect configuration option. In a cluster configured with Direct connect, the transaction types that can be generated depend on the core type that has been used. The core might generate additional transaction types to those listed here so this table must not be used when considering a Direct connect configuration.

Table 8-10: CHI read and write transactions supported by CHI-configured master port, not applicable to Direct connect configurations

Transaction	Operation	Produced by DSU-110
AtomicCompare	Atomic instruction that is not allocating inside the cluster	Yes
AtomicLoad	Atomic instruction that is not allocating inside the cluster	Yes
AtomicStore	Atomic instruction that is not allocating inside the cluster	Yes
AtomicSwap	Atomic instruction that is not allocating inside the cluster	Yes
CleanInvalid	Cache maintenance instructions	Yes
CleanShared	Cache maintenance instructions	Yes
CleanSharedPersist	Not used. CleanSharedPersistSep is used instead.	No

Transaction	Operation	Produced by DSU-110
CleanSharedPersistSep	Cache maintenance instructions. The *Data Cache Clean to the Point of Persistence* (`DC CVAP`) cache maintenance instruction generates this transaction when the **BROADCASTPERSIST** input signal is HIGH.	Yes
CleanUnique	Not used	No
Evict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
MakeInvalid	Not used	No
MakeReadUnique	Store instructions when the line is already cached in a Shared state inside the cluster. This includes store exclusive instructions, which set Excl HIGH.	Yes
MakeUnique	Store instructions of a full cache line of data that miss in the caches.	Yes
PCrdReturn	Not used	No
PrefetchTgt	Hardware prefetch hint to the memory controller	Yes
ReadClean	Reading *Memory Tagging Extension* (MTE) tags for a Cacheable shareable line that is already cached in the cluster without tags.	Yes
ReadNoSnp	Non-cacheable loads or instruction fetches, or cache linefills of Non-shareable cache lines into L1 or L2 caches.	Yes
ReadNoSnpSep	Not used	No
ReadNotSharedDirty	Cache data linefills started by a load instruction, or cache linefills started by an instruction fetch	Yes
ReadOnce	Cacheable shareable instruction fetches that are not allocating into a coherent cache	Yes
ReadOnceCleanInvalid	Not used	No
ReadOnceMakeInvalid	Not used	No
ReadPreferUnique	Speculative store to Cacheable shareable memory or, if Excl is HIGH, a load exclusive instruction.	Yes
ReadShared	Not used	No
ReadUnique	Cache data linefills started by a store instruction	Yes
ReqLCrdReturn	Link credit return	Yes
StashOnceSepShared	Cache prefetch when the L3 cache is not present or powered down. Configured by CLUSTERECTLR_EL1.	Not generated
StashOnceSepUnique	Cache prefetch when the L3 cache is not present or powered down. Configured by CLUSTERECTLR_EL1.	Not generated
StashOnceShared	Not used	No
StashOnceUnique	Not used	No
WriteBackFull	Evictions of dirty cacheable shareable lines from the cluster	Yes
WriteBackFullCMO	Cache maintenance instruction evicting a dirty shareable cache line	Yes
WriteBackPtl	Not used	No
WriteCleanFull	Evictions of dirty lines from the L3 cache, when the line is still present in an L1 or L2 cache.	Yes
WriteCleanFullCMO	Cache maintenance instruction cleaning a dirty shareable cache line	Yes
WriteEvictFull	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
WriteEvictOrEvict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1 register.	Yes
WriteNoSnpFull	Non-cacheable store instructions. Evictions of Non-shareable cache lines	Yes
WriteNoSnpFullCMO	Cache maintenance instruction evicting a dirty Non-shareable cache line	Yes
WriteNoSnpPtl	Non-cacheable store instructions	Yes

Transaction	Operation	Produced by DSU-110
WriteNoSnpPtlCMO	Not used	No
WriteNoSnpZero	Write of zeroes to Non-cacheable or Non-shareable memory using the `DC ZVA` instruction.	Yes
WriteUniqueFull	Cacheable writes of a full cache line not allocating into L1, L2, or L3 caches, for example streaming writes	Yes
WriteUniqueFullCMO	Not used	No
WriteUniqueFullStash	Not used	No
WriteUniquePtl	Generated as a result of *Accelerator Coherency Port* (ACP) WriteUniquePtl transactions when not allocating to the L3 cache	Yes
WriteUniquePtlCMO	Not used	No
WriteUniquePtlStash	Not used	No
WriteUniqueZero	Write of zeroes to a Shareable cache line using the `DC ZVA` instruction	Yes

The following table shows the transactions generated by external memory accesses in an implementation configured with a CHI master interface.

Table 8-11: CHI transaction usage

Attributes		CHI transaction				
Memory type	**Shareability**	**SnpAttr**	**Load**	**Store**	**Load exclusive**	**Store exclusive**
Device	Outer Shareable	Non-snoopable	ReadNoSnp	WriteNoSnp	ReadNoSnp and **Excl** set to HIGH.	WriteNoSnp and **Excl** set to HIGH.
Normal, Inner Non-cacheable, Outer Non-cacheable	Non-shareable	Non-snoopable	ReadNoSnp	WriteNoSnp	ReadNoSnp and **Excl** set to HIGH.	WriteNoSnp and **Excl** set to HIGH.
	Inner Shareable					
	Outer Shareable					
Normal, Inner Non-cacheable, Outer Write-Back or Write-Through, or Normal, Inner Write-Through, Outer Write-Back, Write-Through or Non-cacheable, or Normal Inner Write-Back Outer Non-cacheable or Write-Through	Non-shareable	Non-snoopable	ReadNoSnp	WriteNoSnp	ReadNoSnp and **Excl** set to HIGH.	WriteNoSnp and **Excl** set to HIGH.
	Inner Shareable					
	Outer Shareable					
Normal, Inner Write-Back, Outer Write-Back	Non-shareable	Non-snoopable	ReadNoSnp	WriteNoSnp when the line is evicted or if not allocating into the cache.	ReadNoSnp	WriteNoSnp when the line is evicted.

Attributes		CHI transaction				
Memory type	Shareability	SnpAttr	Load	Store	Load exclusive	Store exclusive
	Inner Shareable	Snoopable	ReadNotSharedDirty or ReadClean	ReadUnique, MakeReadUnique, or MakeUnique if allocating into the cache, then a WriteBackFull when the line is evicted. WriteUniqueFull if not allocating into the cache.	ReadNotSharedDirty, ReadClean, or ReadPreferUnique with **Excl** set to HIGH.	MakeReadUnique with **Excl** set to HIGH if required, then a WriteBackFull when the line is evicted.
	Outer Shareable	Snoopable				

8.7 Use of DataSource field

Some CHI responses from the interconnect include a DataSource field indicating where the data was supplied from. When making use of the DataSource field, Arm® recommends providing this information as accurately as possible.

You can use the recommended encodings in the table *Suggested DataSource value encodings* provided in the *AMBA® 5 CHI Architecture Specification*.

The value of this field is used to calculate some *Performance Monitoring Unit* (PMU) events, and can also be used by some cores to tune the performance of their data prefetchers.

8.8 Support for memory types

The cores in the DSU-110 DynamIQ™ cluster simplify the coherency logic by downgrading some memory types.

Normal memory that is marked as both Inner Write-Back Cacheable and Outer Write-Back Cacheable is cached in the data caches that belong to the cores and the L3 cache.

All other Normal memory types are treated as Non-cacheable and are sent on the master interface as Normal Non-cacheable.

9. AXI master interface

You can configure the DSU-110 to have an AMBA® AXI5 master interface to your memory system, at a build-time confuguration. This provides a non-coherent connection to your memory system. You can configure the DSU-110 to have either one, two, or four AXI master interface ports.

9.1 Multiple master configurations

You can configure the *DynamIQ™ Shared Unit-110* (DSU-110) to have one, two, or four AXI master interface ports, at build time configuration, to give a range of bandwidth options. Transactions from the cores are routed to one of the AXI interface ports based on the transaction type, memory type, and transaction address.

The following table summarizes how transactions are routed to based on the transaction type.

Table 9-1: AXI transaction routing

Transaction type	Routed to
Cacheable transactions	Interface port number based on a hash of the transaction address
Normal Non-cacheable transactions	Interface port number based on a hash of the transaction address
Device non-reorderable transactions	Port 0 by default

Cacheable and non-cacheable transactions

For Cacheable transactions and Normal Non-cacheable transactions, routing from the cores are based on the address of the transaction. A configurable hash of the transaction address selects which master port is used. See 8.1.1 Hashing for transaction distribution on page 109.

Device non-reorderable transactions

By default, device non-reorderable transactions are always routed to port 0 regardless of address. Routing device non-reorderable transactions to port 0 ensures that the ordering requirements can be efficiently met, however this routing arrangement can be overridden. Therefore, all transactions to a given address use the same interface.

9.1.1 Hashing for transaction distribution

When more than one AXI master port is implemented, the hashing to decide which transaction goes to which AXI master port is based on the *Physical Address* (PA) of the transaction.

The hash function masks the physical transaction address with a configurable mask, and then exclusive-or all the resultant bits together. You can set configurable mask using the signal **MASTERINTERLEAVE** before the cluster leaves reset. In the following example:

- MASTERINTERLEAVE is the configurable mask value set by the **MASTERINTERLEAVE** input signal.

- ADDRESS is the PA of the transaction.
- PORT is the master port chosen.

```
MASKED_ADDR = ADDRESS[39:6] & MASTERINTERLEAVE[39:6]
```

For two ports:

```
PORT[0] = MASKED_ADDR[39] ^ MASKED_ADDR[38] ^ to MASKED_ADDR[7] ^ MASKED_ADDR[6]
```

For four ports:

```
PORT[1:0] = MASKED_ADDR[39:38] ^ MASKED_ADDR[37:36] ^ to MASKED_ADDR[9:8] ^
 MASKED_ADDR[7:6]
```

9.2 AXI master port interface properties

AMBA defines a set of interface properties for the AXI interconnect. The AXI master port of the *DynamIQ™ Shared Unit-110* (DSU-110) only supports some of these interface properties.

The following table shows which AXI interface properties the AXI master port supports, and if interconnect or system support is required. You must ensure that your system interconnect, where applicable, supports these properties.

Table 9-2: AXI interconnect properties for the DSU-110

AXI property	Supported by the DSU-110	Interconnect or system support required
Continuous_Cache_Line_Read_Data	Not applicable	No
Multi_Copy_Atomicity	Yes	Yes
Ordered_Write_Observation	No	No
WriteEvict_Transaction	No	No
DVM_v8	No	No
Atomic_Transactions	Yes	Optional, to send atomics to the interconnect set the **BROADCASTATOMIC** signal HIGH.
DVM_v8.1	No	No
Cache_Stash_Transactions	No	No
DeAllocation_Transactions	No	No
DVM_v8.4	No	No
DVM_Message_Support	No	No
Regular_Transaction_Only	Yes	No
Exclusive_Accesses	Yes	Yes
Shareable_Transactions	No	No
Max_Transaction_Bytes	64	-
Persistent_CMO	No	No

AXI property	Supported by the DSU-110	Interconnect or system support required
Poison	No	No
Check_Type	No	No
QoS_Accept	No	No
Trace_Signals	No	No
Loopback_Signals	No	No
Wakeup_Signals	Yes	Yes
Untranslated_Transactions	No	No
NSAccess_Identifiers	No	No
Coherency_Connection_Signals	No	No
Barrier_Transactions	No	No
MPAM_Support	MPAM_6_1 supported by DSU-110	Optional
Unique_ID_Support	Yes	No
Read_Interleaving_Disabled	No	No
Partial_Read_Data	No	No
Read_Data_Reordering	No	No
WriteCMO_Transactions	No	No
MTE support	Yes	Optional

9.3 AXI configurations

The AXI master interface of the *DynamIQ™ Shared Unit-110* (DSU-110) by default supports the AXI5 protocol but you can configure it to support the AXI4 protocol.

To make the AXI master interface compliant with AXI4, tie the signals **BROADCASTMTE** and **BROADCASTATOMIC** LOW.

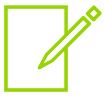
9.4 AXI 256-bit master interface attributes

The read and write issuing capabilities of the AXI master interface depend on the configuration of the *DynamIQ™ Shared Unit-110* (DSU-110) at build time configuration such as the number of L3 cache slices configured. For certain configurations, a maximum number of reads and writes can be up to approximately 128.

The following table shows the AXI master interface attributes.

Table 9-3: AXI 256-bit master interface attributes

Attribute	Value	Comments
Write issuing capability	Configuration dependent	This value can range up to a maximum of 128, depending on configuration. A maximum of 56 non-reorderable Device write transactions can be issued.
Read issuing capability	Configuration dependent	This value can range up to a maximum of 128, depending on configuration. A maximum of 56 outstanding non-reorderable Device read transactions can be issued.
Write ID capability	Configuration dependent	Only Device memory types with nGnRnE or nGnRE can have more than one outstanding transaction with the same AXI ID. All other memory types use a unique AXI ID for every outstanding transaction.
Read ID capability	Configuration dependent	Only Device memory types with nGnRnE or nGnRE can have more than one outstanding transaction with the same AXI ID. All other memory types use a unique AXI ID for every outstanding transaction.
AWID width	10 bits	-
ARID width	10 bits	-


Note

- For the read issuing and write issuing capabilities, the total issuing capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. If there is only one master port configured, then it can support the total number of outstanding transactions. For multiple-master configurations, the percentage of total outstanding transactions each master can support is as follows:
  - If there is only one master port configured, then it can support the total number of outstanding transactions.
  - If there are two master ports configured, then each port can support up to 50% of the total outstanding transactions.
  - If there are four master ports configured, then each port can support up to 25% of the total outstanding transactions.
- The peripheral port can support up to 128 transactions, or the total number of cluster outstanding transaction if this is less.
- The issuing capability described in this table is the maximum for the whole cluster. If you want to achieve the maximum performance available, then you can use these values to size interconnect capabilities. However, this maximum issuing capability might not be reached by a single core on its own. It might need multiple cores generating heavy memory traffic simultaneously to reach the maximum value. The capabilities vary by core type, for example high-performance cores typically generate more transactions than balanced-performance cores. It can also vary by memory type, with typically a significantly lower limit for Device or Non-cacheable transactions than for Cacheable transactions.

For more information about the AXI signals described in this manual, see the *AMBA® AXI and ACE Protocol Specification*.

9.5 AXI transactions

The AXI master interface of the *DynamIQ™ Shared Unit-110* (DSU-110) only generates three types of AXI transactions which are, ReadNoSnoop, WriteNoSnoop, and read and write atomic transactions.

The following table describes the supported AXI transactions, and typical operations that cause these transactions to be generated.

Table 9-4: AXI transactions

Transaction	Operation
ReadNoSnoop	Non-cacheable loads or instruction fetches. Linefills of cache lines into L1, L2, or L3 caches.
WriteNoSnoop	Non-cacheable store instructions. Evictions of cache lines from L1, L2, and L3 caches.
AtomicLoad	-
AtomicStore	-
AtomicSwap	-
AtomicCompare	-

The cache linefill fetch length is always 64 bytes. The DSU-110 does not generate any FIXED bursts and a burst does not cross a cache line boundary.

The DSU-110 generates only a subset of all possible AXI transactions on the master interface.

For Normal Non-cacheable or Device transactions:

- INCR N (N:2) 256-bit read transfers
- INCR N (N:2) 256-bit write transfers
- WRAP N (N:2) 256-bit read transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, 128-bit, and 256-bit read transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, 128-bit, and 256-bit write transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, 128-bit, and 256-bit exclusive read transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, and 128-bit, and 256-bit exclusive write transfers

The following atomic transactions are supported:

- AtomicCompare
- AtomicLoad
- AtomicStore
- AtomicSwap

Atomic transactions are only generated by the cluster if **BROADCASTATOMICMP** signal is HIGH.

The following points apply to AXI transactions:

- WRAP bursts are only 256-bit in size.

- INCR burst, more than one transfer, are only 256-bit in size.
- No transaction is marked as FIXED.
- Write transfers with none, some, or all byte strobes LOW can occur.

9.6 Support for memory types

The cores in the DSU-110 DynamIQ™ cluster simplify the coherency logic by downgrading some memory types.

Normal memory that is marked as both Inner Write-Back Cacheable and Outer Write-Back Cacheable is cached in the core data caches and the L3 cache.

All other Normal memory types are treated as Non-cacheable and are sent on the master interface as Normal Non-cacheable.

9.7 Read response

The AXI master can delay accepting a read data channel transfer by holding **RREADY** LOW for an indeterminate number of cycles.

RREADY can be deasserted LOW between read data channel transfers that form part of the same transaction.

9.8 Write response

The AXI master requires that the slave does not return a write response until it has received the write address.


Note

For interoperability reasons, Arm recommends that system components fully comply with the AXI specification and do not rely on the DSU-110 behavior described here.

9.9 Barriers

The *DynamIQ™ Shared Unit-110* (DSU-110) does not support sending barrier transactions to the interconnect. Barriers are always terminated within the cluster.

You must ensure that your interconnect and any peripherals that are connected to it, do not return a write response for a transaction until that transaction is considered complete by a later barrier.

This means that the write must be observable to all other masters in the system. Arm expects most peripherals to meet this requirement.

9.10 AXI privilege information

AXI provides information about the privilege level of accesses on the **ARPROTM[0]** and **AWPROTM[0]** signals. This information is not available from cores within the cluster. Therefore these signals are always driven HIGH indicating that the access could be a privileged access.

10. ACP slave interface

The *Accelerator Coherency Port* (ACP) is an optional slave interface that provides coherent transaction support between the *DynamIQ™ Shared Unit-110* (DSU-110) and external accelerators such as a *Direct Memory Access* (DMA) engine. It is configured during build time configuration and can be implemented as either a 128-bit or 256-bit port.

The ACP slave interface allows an external master to access memory through the main memory interface of the DSU-110. Accesses are optimized for cache line length.

To maintain cache coherency, accesses are checked in the L3 cache and in the data caches in each core.

By default, ACP write-accesses to cacheable memory are implicit stash requests to the L3 cache. Alternatively, explicit stash requests (WriteUniqueFullStash, WriteUniquePtlStash, StashOnceShared, or StashOnceUnique) can target the L2 cache of a selected core or the L3 cache.


Note

You can configure the DSU-110 to have an ACP port, when the L3 cache is not present. This configuration is only recommended if the ACP is used for cache stashing to L2 caches in the cores.

10.1 ACP features

The *Accelerator Coherency Port* (ACP) interface conforms to a subset of the AMBA ACE5-LiteDVM protocol specification and includes support for atomic transactions and cache stashing. Memory tagging is also supported but only to a basic level as defined by the AMBA specification. This allows reading and writing the tags but does not support tag matching on writes.


Note

See *AMBA® AXI and ACE Protocol Specification* for a description of the AMBA ACE5-LiteDVM protocol.

The following table shows the ACP interface properties that are supported by the *DynamIQ™ Shared Unit-110* (DSU-110).

Table 10-1: ACP interface properties for the DSU-110

ACP property	Supported by the DSU-110
Port_Type	Accelerator
Continuous_Cache_Line_Read_Data	Yes
Multi_Copy_Atomicity	Yes. System support is required.
Ordered_Write_Observation	No

ACP property	Supported by the DSU-110
WriteEvict_Transaction	No
DVM_v8	Yes
Atomic_Transactions	Yes
DVM_v8.1	Yes
DVM_v8.4	Yes
Cache_Stash_Transactions	Yes
Prefetch_Transactions	No
DeAllocation_Transactions	No
Persistent_CMO	No
Write_Plus_CMO	No
Poison	No
Data_Check	No
QoS_Accept	No
Trace_Signals	No
Loopback_Signals	No
Low_Power_Signals	Yes
Untranslated_Transactions	No
NSAccess_Identifiers	No
WriteZero_Transaction	No
Regular_Transactions_Only	Only regular transactions are supported.
Exclusive_Accesses	No
Shareable_Transactions	Yes
Max_Transaction_Bytes	64
DVM_Message_Support	Receiver

Related information

10.2 ACP ACE5-LiteDVM protocol subset on page 127
10.3 ACP transactions on page 128

10.2 ACP ACE5-LiteDVM protocol subset

The *Accelerator Coherency Port* (ACP) interface conforms to a subset of the AMBA ACE5-LiteDVM protocol specification that includes support for Cacheable, Non-cacheable, and Device memory accesses.

The ACP interface supports the following features as defined in the AMBA ACE5-LiteDVM protocol specification:

- Normal Read-Allocate and Write-Allocate cacheable memory is supported.
- Normal Non-cacheable and Device memory accesses are supported.

- Atomics are supported.
- All requests can be Secure or Non-secure.
- All requests can specify Inner Shareable, Outer Shareable, and Non-shareable using the **AWDOMAINS[1:0]** and **ARDOMAINS[1:0]** signals. Inner Shareable is treated identically to Outer Shareable. Transactions to Cacheable Non-shareable memory are not cached in the L3 cache.
- *Distributed Virtual Messages* (DVM) messages are supported for connecting to an upstream *System Memory Management Unit* (SMMU).
- Cache stashing is supported, allowing the stash to target either the L3 cache, or a specific L2 cache belonging to a core.
- All ACE5-LiteDVM signals, apart from **ARQOS** and **AWQOS**, are included in the ACP interface.

The ACP interface does not support the following features as defined in the AMBA ACE5-LiteDVM protocol specification:

- Barriers are not supported. The **BRESPS[1:0]** response for any write transaction indicates global observability for the transaction.
- Exclusive accesses are not supported. Therefore, **ARLOCK** and **AWLOCK** signals are not present.


Note

For information on how to connect the ACP interface to your system, see *Functional Integration* in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual*.

Related information

10.1 ACP features on page 126
10.3 ACP transactions on page 128
10.4 ACP performance on page 132

10.3 ACP transactions

The *Accelerator Coherency Port* (ACP) interface conforms to a subset of the AMBA ACE5-LiteDVM protocol specification. The ACP interface includes support for Cacheable, Non-cacheable, Device, and Atomic memory accesses.

The following table lists the subset of the ACE-LiteDVM transaction types that are supported in the ACP interface.

Table 10-2: ACP supported transaction types

Transaction group	Transaction type
Read	ReadOnce
	ReadNoSnoop

Transaction group	Transaction type
Write	WriteUniquePtl
	WriteUniqueFull
	WriteUniquePtlStash
	WriteNoSnoop
Dataless	StashOnceUnique
	StashOnceShared
Atomic	AtomicStore
	AtomicLoad
	AtomicSwap
	AtomicCompare


Note

The transaction types WriteUniqueFull and WriteUniquePtl in the AMBA ACE5-LiteDVM specification are known in the AMBA 4 ACELite specification as WriteLineUnique and WriteUnique, respectively.

The following table shows the attributes for read transactions types for 128-bit (16-byte) data width mode.

Table 10-3: Attributes for read transaction types for 128-bit data width mode

Read request type	ARSIZE	ARLEN	ARBURST	Address alignment	
				INCR	WRAP
64-byte	`0x4` (16-bytes)	`0x3` (4-beats)	INCR or WRAP	64-byte boundary (**ARADDR[5:0]** = `0b000000`)	16-byte boundary (**ARADDR[3:0]** = `0b0000`)
32-byte	`0x4` (16-bytes)	`0x1` (2-beats)	INCR or WRAP	32-byte boundary (**ARADDR[4:0]** = `0b00000`)	16-byte boundary (**ARADDR[3:0]** = `0b0000`)
16-byte	`0x4` (16-bytes)	`0x0` (1-beat)	INCR or WRAP	16-byte boundary (**ARADDR[3:0]** = `0b0000`)	16-byte boundary (**ARADDR[3:0]** = `0b0000`)
8-byte	`0x3` (8-bytes)	`0x0` (1-beat)	INCR or WRAP	8-byte boundary (**ARADDR[2:0]** = `0b000`)	8-byte boundary (**ARADDR[2:0]** = `0b000`)
4-byte	`0x2` (4-bytes)	`0x0` (1-beat)	INCR or WRAP	4-byte boundary (**ARADDR[1:0]** = `0b00`)	4-byte boundary (**ARADDR[1:0]** = `0b00`)
2-byte	`0x1` (2-bytes)	`0x0` (1-beat)	INCR or WRAP	2-byte boundary (**ARADDR[0]** = `0b0`)	2-byte boundary (**ARADDR[0]** = `0b0`)
1-byte	`0x0` (1-byte)	`0x0` (1-beat)	INCR or WRAP	Any	Any

The following table shows the attributes for read transactions types for 256-bit (32-byte) data width mode.

Table 10-4: Attributes for read transaction types for 256-bit data width mode

Read request type	ARSIZE	ARLEN	ARBURST	Address alignment	
				INCR	WRAP
64-byte	0x5 (32-bytes)	0x1 (2-beats)	INCR or WRAP	64-byte boundary (**ARADDR[5:0]** = 0b000000)	32-byte boundary (**ARADDR[4:0]** = 0b00000)
32-byte	0x5 (32-bytes)	0x0 (1-beats)	INCR or WRAP	32-byte boundary (**ARADDR[4:0]** = 0b00000)	32-byte boundary (**ARADDR[4:0]** = 0b00000)
16-byte	0x4 (16-bytes)	0x0 (1-beat)	INCR or WRAP	16-byte boundary (**ARADDR[3:0]** = 0b0000)	16-byte boundary (**ARADDR[3:0]** = 0b0000)
8-byte	0x3 (8-bytes)	0x0 (1-beat)	INCR or WRAP	8-byte boundary (**ARADDR[2:0]** = 0b000)	8-byte boundary (**ARADDR[2:0]** = 0b000)
4-byte	0x2 (4-bytes)	0x0 (1-beat)	INCR or WRAP	4-byte boundary (**ARADDR[1:0]** = 0b00)	4-byte boundary (**ARADDR[1:0]** = 0b00)
2-byte	0x1 (2-bytes)	0x0 (1-beat)	INCR or WRAP	2-byte boundary (**ARADDR[0]** = 0b0)	2-byte boundary (**ARADDR[0]** = 0b0)
1-byte	0x0 (1-byte)	0x0 (1-beat)	INCR or WRAP	Any	Any

The following table shows the attributes for write transactions types for 128-bit (16-byte) data width mode.

Table 10-5: Attributes for write transaction types for 128-bit data width mode

Write request type	AWSIZE	AWLEN	AWBURST	Address alignment		Comment
				INCR	WRAP	
64-byte	0x4 (16-bytes)	0x3 (4-beats)	INCR or WRAP	64-byte boundary (**AWADDR[5:0]** = 0b000000)	16-byte boundary (**ARADDR[3:0]** = 0b0000)	If **AWSNOOP** is WriteUniquePtl, then any combination of bytes is valid. If **AWSNOOP** is WriteUniqueFull, then all bytes must be valid.
32-byte	0x4 (16-bytes)	0x1 (2-beats)	INCR or WRAP	32-byte boundary (**ARADDR[4:0]** = 0b00000)	16-byte boundary (**AWADDR[3:0]** = 0b0000)	Any combination of bytes is valid.
16-byte	0x4 (16-bytes)	0x0 (1-beat)	INCR or WRAP	16-byte boundary (**AWADDR[3:0]** = 0b0000)	16-byte boundary (**AWADDR[3:0]** = 0b0000)	Any combination of bytes is valid. This includes no bytes, which mimics a PLDW instruction (read-unique preload).
8-byte	0x3 (8-bytes)	0x0 (1-beat)	INCR or WRAP	8-byte boundary (**AWADDR[2:0]** = 0b000)	8-byte boundary (**AWADDR[2:0]** = 0b000)	Any combination of bytes is valid.
4-byte	0x2 (4-bytes)	0x0 (1-beat)	INCR or WRAP	4-byte boundary (**AWADDR[1:0]** = 0b00)	4-byte boundary (**AWADDR[1:0]** = 0b00)	Any combination of bytes is valid.
2-byte	0x1 (2-bytes)	0x0 (1-beat)	INCR or WRAP	2-byte boundary (**AWADDR[0]** = 0b0)	2-byte boundary (**AWADDR[0]** = 0b0)	Any combination of bytes is valid.
1-byte	0x0 (1-byte)	0x0 (1-beat)	INCR or WRAP	Any	Any	Any combination of bytes is valid.

The following table shows the attributes for write transactions types for 256-bit (32-byte) data width mode.

Table 10-6: Attributes for write transaction types for 256-bit data width mode

Write request type	AWSIZE	AWLEN	AWBURST	Address alignment		Comment
				INCR	WRAP	
64-byte	0x5 (32-bytes)	0x1 (2-beats)	INCR or WRAP	64-byte boundary (**AWADDR[5:0]** = 0b000000)	32-byte boundary (**AWADDR[4:0]** = 0b00000)	If **AWSNOOP** is WriteUniquePtl, then any combination of bytes is valid. If **AWSNOOP** is WriteUniqueFull, then all bytes must be valid.
32-byte	0x4 (16-bytes)	0x1 (2-beats)	INCR or WRAP	32-byte boundary (**AWADDR[4:0]** = 0b00000)	32-byte boundary (**AWADDR[4:0]** = 0b00000)	Any combination of bytes is valid.
16-byte	0x4 (16-bytes)	0x0 (1-beat)	INCR or WRAP	16-byte boundary (**AWADDR[3:0]** = 0b0000)	16-byte boundary (**AWADDR[3:0]** = 0b0000)	Any combination of bytes is valid. This includes no bytes, which mimics a PLDW instruction (read-unique preload).
8-byte	0x3 (8-bytes)	0x0 (1-beat)	INCR or WRAP	8-byte boundary (**AWADDR[2:0]** = 0b000)	8-byte boundary (**AWADDR[2:0]** = 0b000)	Any combination of bytes is valid.
4-byte	0x2 (4-bytes)	0x0 (1-beat)	INCR or WRAP	4-byte boundary (**AWADDR[1:0]** = 0b00)	4-byte boundary (**AWADDR[1:0]** = 0b00)	Any combination of bytes is valid.
2-byte	0x1 (2-bytes)	0x0 (1-beat)	INCR or WRAP	2-byte boundary (**AWADDR[0]** = 0b0)	2-byte boundary (**AWADDR[0]** = 0b0)	Any combination of bytes is valid.
1-byte	0x0 (1-byte)	0x0 (1-beat)	INCR or WRAP	Any	Any	Any combination of bytes is valid.

Stash requests can target the L2 cache of a selected core by asserting signal **AWSTASHLPIDENS** and indicating the selected core instance number on **AWSTASHLPIDS[3:0]**. See 2.7 Core, complex, and processing element numbering on page 30 for a description of the core instance number.

All ACE5-LiteDVM signals are present on the ACP interface, except **AxLOCK** and **AxQOS**. For the unconnected signal **AxLOCK**, the ACP interface does not support exclusives and therefore the functionality matches the **AxLOCK** signal being tied LOW.


Note

The DSU-110 generates an SLVERRn in response to any of the following conditions:

- **AxDOMAIN** is 0b11 (System domain access) when **AxCACHE** is 0bxx11 (Write-Back cacheable).
- For 128-bit wide data mode, **AxLEN** is a value other than 0b00000011, 0b00000001, or 0b00000000.
- For 256-bit wide data mode, **AxLEN** is a value other than 0b00000001 or 0b00000000.
- For 128-bit wide data mode, an SLVERR is produced if either:
  - **AxLEN** is 00000001 and **AxADDR[4:0]** is a value other than 0b00000.
  - **AxLEN** is 00000011 and **AxADDR[5:0]** is a value other than 0b000000.
- For 256-bit wide data mode, **AxADDR[5:0]** is a value other than 0b000000 when AxLEN is not 0b00000000.

- **AWSNOOP** is any transaction other than WriteNoSnoop, WriteUniquePtl, WriteUniqueFull, WriteUniquePtlStash, WriteUniqueFullStash, StashOnceShared, StashOnceUnique, AtomicLoad, AtomicStore, AtomicSwap, or AtomicCompare.
- **ARSNOOP** is any transaction other than ReadNoSnoop, ReadOnce, or DVMComplete.
- **AxBURST** is a value other than `0b01` or `0b10`. Only incremental or wrap bursts are supported.

Values of **AxCACHE** that are not fully supported are mapped to the nearest supported memory type that has the same or stronger requirements.

10.4 ACP performance

For optimum performance, use the following guidelines for *Accelerator Coherency Port* (ACP) transactions.

AXI ID guidelines

The ACP master must avoid sending more than one outstanding transaction on the same AXI ID to prevent the second transaction stalling the interface until the first has completed. If the master requires explicit ordering between two transactions, Arm recommends that it waits for the response to the first transaction before sending the second transaction.

Write transactions

Writes to memory that use either WriteUniqueFull or WriteUniqueFullStash transactions have higher performance than other types of write transactions.

WriteUniquePtl or WriteUniquePtlStash transactions always incur a read-modify write sequence.

Write transactions use the Write-Allocate bit of the memory type (**AWCACHE[3]**) to decide whether to allocate to the L3 cache, as follows:

- If the stash request does not target a core (**AWSTASHLPIDENS** is LOW) and **AWCACHES[3]** is HIGH, then the cache line is allocated to the L3 cache.
- When the stash request does not target a core (**AWSTASHLPIDENS** is LOW), then the WriteUniqueFullStash transaction performs the same operation as WriteUniqueFull.
- If the stash request does not target a core (**AWSTASHLPIDENS** is LOW) and **AWCACHES[3]** is LOW, then the cache line is not allocated to the L3 cache. Instead, the cache line is written out on the master port instead.
- Stash requests that target a core (**AWSTASHLPIDENS** is HIGH) always attempt to allocate to the core L2 cache. If a stash request targets a core then allocation is determined by the value in **AWCACHES[3]**:
  - If **AWCACHES[3]** is LOW, the cache line is written out to the master port before being fetched back into the L2 cache of the core.

- If **AWCACHES[3]** is HIGH, the cache line is immediately allocated to the L2 cache. Arm recommends that **AWCACHES[3]** is HIGH because it is more efficient.

Heavy ACP traffic

Some data buffering is shared between the ACP interface and the cores. Therefore, heavy traffic on the ACP interface might reduce the performance of the cores.

ACP acceptance capabilities

The following table describes the ACP acceptance capabilities.

Table 10-7: ACP acceptance capabilities

Attribute	Value	Description
Write acceptance capability	256	The ACP can accept up to 256 write transactions depending on configuration. The total is limited to the `NUM_LTDBS` parameter multiplied by the `NUM_L3_SLICES` parameter.
Read acceptance capability	256	The ACP can accept up to 256 read transactions depending on configuration. The total is limited to the `NUM_LTDBS` parameter multiplied by the `NUM_L3_SLICES` parameter.
Combined acceptance capability	512	The ACP can accept up to 512 transactions depending on configuration. The total is limited to the `NUM_LTDBS` parameter multiplied by the `NUM_L3_SLICES` parameter.

Related information

10.1 ACP features on page 126
10.2 ACP ACE5-LiteDVM protocol subset on page 127
10.3 ACP transactions on page 128

10.5 DVM snoop transaction support

The *Accelerator Coherency Port* (ACP) of the *DynamIQ™ Shared Unit-110* (DSU-110) supports *Distributed Virtual Messages* (DVM) snoop transactions. DVM snoop transactions are only sent from the ACP port to the ACP master.

DVM snoop transactions can be used with a *System Memory Management Unit* (SMMU) to manage external coherent memory table walks and memory table updates.

The ACP supports the following DVM transaction features:

- Issue of both DVM Operations and DVM Sync transaction on the AC channel.
- Receiving of DVM Complete on the AR channel.

Only the following DVMOp transaction types are issued by the ACP slave port:

- TLB Invalidate
- Synchronization

The maximum number of outstanding DVMOp transactions that can be processed are:

- 1 DVMOp Sync transaction
- 4 DVMOp non-Sync transactions

To control the broadcast of DVM snoop transactions on the ACP port, see 10.5.1 Control the receiving of DVM snoop transactions on page 134.


Note

- If your ACP master does not support DVMs, then tie the **SYSCOREQS** signal LOW.
- When **SYSCOREQS** signal is HIGH, it prevents powering down of the cluster. Ensure you deassert **SYSCOREQS** when the device connected to ACP is inactive and ready to be powered down.

10.5.1 Control the receiving of DVM snoop transactions

You must use the signals **SYSCOREQS** and **SYSCOACKS** to control the broadcasting of *Distributed Virtual Messages* (DVM) snoop transactions from the *Accelerator Coherency Port* (ACP) to your ACP master.

About this task

How to control the *DynamIQ™ Shared Unit-110* (DSU-110) to start or stop broadcasting DVM snoops from the ACP port, using a four-phase handshake, with the signals **SYSCOREQS** and **SYSCOACKS**.


Note

The use of **SYSCOREQS (SYSCOREQ)** and **SYSCOACKS (SYSCOACK)** is described in *AMBA® AXI and ACE Protocol Specification*.

Procedure

1. Instruct your ACP master to assert the **SYSCOREQS** signal (HIGH) when it is ready to receive DVM snoop transactions.
2. Wait for the signal **SYSCOACKS** to go HIGH. This signal indicates that DSU-110 has acknowledged the request.
   When both the **SYSCOREQS** and **SYSCOACKS** signals are HIGH, the DSU-110 is enabled to start broadcasting DVM snoop transactions on the ACP port.
3. When you want to stop receiving DVM snoop transactions, instruct your ACP master to deassert **SYSCOREQS** signal (LOW).
4. Wait for the signal **SYSCOACKS** to go LOW.
   When the signal **SYSCOACKS** has gone LOW, the DSU-110 stops the broadcasting of the DVM snoop transactions.


Note

You must deassert **SYSCOREQS** before you power off the cluster. Any request to power off the cluster will be denied if **SYSCOACKS** remains HIGH.

11. AXI or CHI master peripheral port

You can use the peripheral port to program registers for peripherals using Device accesses, for example, to configure tightly coupled accelerators. You can also use the peripheral port as an alternative master port to support accesses to the rest of the system whilst the main master ports connect to main memory.

Using the peripheral port as alternative master port can help to optimize the latency to DRAM memory in some system designs.

The peripheral port can be configured at build-time configuration to either:

- A 64-bit AXI5 non-coherent master interface
- A 256-bit AXI5 non-coherent master interface
- A 256-bit CHI Issue E master interface. This is a non-coherent interface by default, but you can make it coherent setting **BROADCASTOUTERMP** signal at reset.

You can optionally include the peripheral port at build-time configuration. You can also configure the peripheral port to use the AXI or CHI protocol at build-time configuration. See *RTL configuration process* in the *Arm® DynamIQ™ Shared Unit-110 Configuration and Integration Manual* for more details on configuring the peripheral port.

11.1 Supported memory and transaction types

The peripheral port supports all transactions that a core can generate including, atomic transactions, cacheable and Non-cacheable accesses, and load and store exclusives.

The peripheral port supports the following transactions:

- Normal Read-Allocate and Write-Allocate cacheable accesses
- Normal Non-cacheable accesses
- Accesses to Device memory types (Device-GRE, nGRE, nGnRE, nGnRnE)
- Atomic transactions
- Load and store exclusive transactions


Note

- Cacheable memory transactions are only coherent outside the *DynamIQ™ Shared Unit-110* (DSU-110) if the peripheral port is configured to use a CHI and the signal **BROADCASTOUTERMP** is tied HIGH.
- For Atomic transactions, the signal **BROADCASTATOMICMP** must be used to indicate if the interconnect supports atomic transactions for the peripheral port.

11.2 Mapping peripheral port address ranges

The peripheral port supports up to four address ranges for access which you can configure using input bus signals at reset. The first address range can also be configured by programming the System registers IMP_CLUSTERPPSTART_EL1 and IMP_CLUSTERPPEND_EL1.

You can define the start address and end address of the first address range using the following bus signals:

ASTART0MP[PA-1:20]

This bus defines the start address for the first address range, which is inclusive. PA is the largest physical address size of any connected core.

AEND0MP[PA-1:20]

This bus defines the end address for the first address range, which is exclusive. PA is the largest physical address size of any connected core.

Therefore, the address range is defined as **ASTART0MP[PA-1:20]** <= peripheral port address range < **AEND0MP[PA-1:20]**.

The first address range is configurable to granularity of 1MB. The values for the **ASTART0MP[PA-1:20]** and **AEND0MP[PA-1:20]** signals are only captured at reset. The first address range is also captured into registers IMP_CLUSTERPPSTART_EL1 and IMP_CLUSTERPPEND_EL1 and can be changed at runtime using software.

You can define the start address and end address of the remaining address ranges using the following bus signals:

ASTART<n>MP[PA-1:30]

This bus defines the start address for the corresponding address range n, where n = 1, 2, or 3 and PA is the largest physical address size of any connected core. The start address is inclusive.

AEND<n>MP[PA-1:30]

This bus defines the end address for the corresponding address range n, where n = 1, 2, or 3, and PA is the largest physical address size of any connected core. The end address is exclusive.

Therefore, the address range is defined as **ASTART<n>MP[PA-1:20]** <= peripheral port address range < **AEND<n>MP[PA-1:20]**.


Note

- If an address range is not used, you must set the start address to the end address. For example, tie **ASTART1MP** and **AEND1MP** LOW.
- If **DEFAULTMP** signal is LOW, and both **ASTART0MP** and **AEND0MP** are tied LOW then traffic is sent to the main master port instead of the peripheral port.

These remaining address ranges have a granularity of 1GB. The values for the **ASTART<n>MP[PA-1:30]** and **AEND<n>MP[PA-1:30]** signals are only captured at reset. These address ranges are not captured into any registers unlike the first address range.


Note

If you are making address range changes, and there are outstanding transactions to either new or the old address ranges, then it is not guaranteed if the transactions go to the peripheral port or the main master port. See 11.2.1 Changing peripheral port address range on page 138 for more details.

By default, all incoming transaction addresses go to the main master port or ports, unless both of the following occur in which case they are routed to the peripheral port:

- The transaction is either a core transaction or *Accelerator Coherency Port* (ACP) transaction.
- The core or ACP transaction matches one of the peripheral port address ranges.

Distributed Virtual Memory (DVM) operations go to master port 0 if supported. See 9.1 Multiple master configurations on page 119 for more details.

However, if the signal **DEFAULTMP** is asserted at reset, then this mapping is inverted and therefore:

- All incoming transaction addresses go to the peripheral port except those that match configured address ranges which are sent to the main master ports.
- DVM operations are sent to the peripheral port if supported.


Note

To avoid system deadlocks, the peripheral port and main master ports must be able to complete their accesses independently of each other. However, when a 64-bit AXI peripheral port is configured, it is permissible for a peripheral port access to depend on an *Accelerator Coherency Port* (ACP) access completing.

11.2.1 Changing peripheral port address range

The *DynamIQ™ Shared Unit-110* (DSU-110) supports changing the peripheral port address range to match your system requirements.

Before you begin

- Ensure both the old and new address ranges are Non-cacheable, for example, this could be done by:
  - Making the memory type a Non-cacheable type or Device type.
  - Making the memory translation invalid.
  - Disabling the L1 and L2 caches in all cores in the *DynamIQ™ Shared Unit-110* cluster.
- If the old address range is marked as Cacheable, then clean and invalidate all the addresses in that address range. This ensures any cached data is written back on the same interface that it originally came from. This must include a *Data Synchronization Barrier* (DSB) at the end to ensure the clean and invalidate has completed.

Caution

Failure to perform this invalidation could cause system deadlocks if data remains in the L3 cache for the old address range.

Procedure

1. Reprogram the IMP_CLUSTERPPSTART_EL1 and IMP_CLUSTERPPEND_EL1 registers as appropriate.
2. Execute a `DSB` and `ISB` instructions to ensure the register updates have completed.
3. You can now map the memory as required or enable the L1 and L2 caches in the cores.

Caution

During steps 1 and 2, transactions to the address range being changed might go to either the old port or the new port. Therefore, transactions to this address range might not occur in the expected order.

11.3 AXI 64-bit peripheral port interface properties

AMBA defines a set of interface properties for the AXI interconnect. The AXI 64-bit configured peripheral port of the *DynamIQ™ Shared Unit-110* (DSU-110) only supports some of these interface properties.

The following table shows which AXI interface properties the AXI 64-bit configured peripheral port supports, and if interconnect or system support is required. You must ensure that your system interconnect, where applicable, supports these properties.

Table 11-1: AXI 64-bit peripheral port interface properties

AXI property	Supported by the DSU-110	Interconnect or system support required
Continuous_Cache_Line_Read_Data	Yes	No
Multi_Copy_Atomicity	Yes	Yes
Ordered_Write_Observation	No	No
WriteEvict_Transaction	No	No
DVM_v8	No	No
Atomic_Transactions	Yes	Optional, to send atomics to the interconnect set the **BROADCASTATOMICMP** signal HIGH.
DVM_v8.1	No	No
Cache_Stash_Transactions	No	No
DeAllocation_Transactions	No	No
Persist_CMO	No	No
Poison	No	No
Check_Type	No	No
QoS_Accept	No	No

AXI property	Supported by the DSU-110	Interconnect or system support required
Trace_Signals	No	No
Loopback_Signals	No	No
Wakeup_Signals	Yes	Yes
Untranslated_Transactions	No	No
NSAccess_Identifiers	No	No
Coherency_Connection_Signals	No	No
Barrier_Transactions	No	No
MPAM_Support	Yes, the DSU-110 cluster supports MPAM_6_1	Optional
Unique_ID_Support	Yes	No
Read_Interleaving_Disabled	No	No
Partial_Read_Data	No	No
Read_Data_Reordering	No	No
WriteCMO_Transactions	No	No
MTE_Support	No	Optional

11.4 AXI 256-bit peripheral port interface properties

AMBA defines a set of interface properties for the AXI interconnect. The AXI 256-bit configured peripheral port of the *DynamIQ™ Shared Unit-110* (DSU-110) only supports some of these interface properties.

The following table shows which AXI interface properties the AXI 256-bit configured peripheral port supports, and if interconnect or system support is required. You must ensure that your system interconnect, where applicable, supports these properties.

Table 11-2: AXI 256-bit peripheral port interface properties

AXI property	Supported by the DSU-110	Interconnect or system support required
Continuous_Cache_Line_Read_Data	Yes	No
Multi_Copy_Atomicity	Yes	Yes
Ordered_Write_Observation	No	No
WriteEvict_Transaction	No	No
DVM_v8	No	No
Atomic_Transactions	Yes	Optional, to send atomics to the interconnect set the **BROADCASTATOMICMP** signal HIGH.
DVM_v8.1	No	No
Cache_Stash_Transactions	No	No
DeAllocation_Transactions	No	No
Persist_CMO	No	No
Poison	No	No
Check_Type	No	No

AXI property	Supported by the DSU-110	Interconnect or system support required
QoS_Accept	No	No
Trace_Signals	No	No
Loopback_Signals	No	No
Wakeup_Signals	Yes	Yes
Untranslated_Transactions	No	No
NSAccess_Identifiers	No	No
Coherency_Connection_Signals	No	No
Barrier_Transactions	No	No
MPAM_Support	Yes, the DSU-110 cluster supports MPAM_6_1	Optional
Unique_ID_Support	Yes	No
Read_Interleaving_Disabled	No	No
Partial_Read_Data	No	No
Read_Data_Reordering	No	No
WriteCMO_Transactions	No	No
MTE_Support	Yes	Optional

11.5 AXI 64-bit peripheral port transactions

The AXI 64-bit configured peripheral port of the *DynamIQ™ Shared Unit-110* (DSU-110) only generates three types of AXI transactions which are, ReadNoSnoop, WriteNoSnoop, and read and write atomic transactions.

The following table describes the supported AXI transactions and typical operations that cause these transactions to be generated, for the AXI 64-bit configured peripheral port:

Table 11-3: AXI transactions

Transaction	Operation
ReadNoSnoop	Non-cacheable loads or instruction fetches. Linefills of cache lines into L1, L2, or L3 caches.
WriteNoSnoop	Non-cacheable store instructions. Evictions of cache lines from L1, L2, and L3 caches.
AtomicLoad	-
AtomicStore	-
AtomicSwap	-
AtomicCompare	-

The cache linefill fetch length is always 64 bytes. The DSU-110 does not generate any FIXED bursts, and a burst does not cross a cache line boundary.

The DSU-110 generates only a subset of all possible AXI transactions on the master interface.

For Normal Non-cacheable or Device accesses, for a 64-bit AXI configured peripheral port, the following burst types are supported:

- INCR N (N:2) 64-bit read transfers
- INCR N (N:2) 64-bit write transfers
- WRAP N (N:2) 64-bit read transfers
- INCR 1 8-bit, 16-bit, 32-bit, and 64-bit read transfers
- INCR 1 8-bit, 16-bit, 32-bit, and 64-bit write transfers
- INCR 1 8-bit, 16-bit, 32-bit, and 64-bit exclusive read transfers
- INCR 1 8-bit, 16-bit, 32-bit, and 64-bit exclusive write transfers

For a 64-bit AXI configured peripheral port, the following points apply to AXI transactions:

- WRAP bursts are only 64-bit size.
- INCR burst, more than one transfer, are only 64-bit size.
- No transaction is marked as FIXED.
- Write transfers with none, some, or all byte strobes being LOW can occur.

The peripheral port supports the following atomic transactions:

- AtomicCompare
- AtomicLoad
- AtomicStore
- AtomicSwap

Atomic transactions are only generated by the cluster if **BROADCASTATOMICMP** signal is HIGH.

11.6 AXI 256-bit peripheral port transactions

The AXI 256-bit configured peripheral port of the *DynamIQ™ Shared Unit-110* (DSU-110) only generates three types of AXI transactions which are, ReadNoSnoop, WriteNoSnoop, and read and write atomic transactions.

The following table describes the supported AXI transactions and typical operations that cause these transactions to be generated, for the AXI 256-bit configured peripheral port:

Table 11-4: AXI transactions

Transaction	Operation
ReadNoSnoop	Non-cacheable loads or instruction fetches. Linefills of cache lines into L1, L2, or L3 caches.
WriteNoSnoop	Non-cacheable store instructions. Evictions of cache lines from L1, L2, and L3 caches.
AtomicLoad	-
AtomicStore	-
AtomicSwap	-
AtomicCompare	-

The cache linefill fetch length is always 64 bytes. The DSU-110 does not generate any FIXED bursts and a burst does not cross a cache line boundary.

The DSU-110 generates only a subset of all possible AXI transactions on the master interface.

For Normal Non-cacheable or Device transactions:

- INCR N (N:2) 256-bit read transfers
- INCR N (N:2) 256-bit write transfers
- WRAP N (N:2) 256-bit read transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, 128-bit, and 256-bit read transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, 128-bit, and 256-bit write transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, 128-bit, and 256-bit exclusive read transfers
- INCR 1 8-bit, 16-bit, 32-bit, 64-bit, and 128-bit, and 256-bit exclusive write transfers

The following atomic transactions are supported:

- AtomicCompare
- AtomicLoad
- AtomicStore
- AtomicSwap

Atomic transactions are only generated by the cluster if **BROADCASTATOMICMP** signal is HIGH.

The following points apply to AXI transactions:

- WRAP bursts are only 256-bit in size.
- INCR burst, more than one transfer, are only 256-bit in size.
- No transaction is marked as FIXED.
- Write transfers with none, some, or all byte strobes LOW can occur.

11.7 Attributes of the CHI peripheral port

The read and write issuing capabilities of the CHI configured peripheral port depend on the configuration of the *DynamIQ™ Shared Unit-110* (DSU-110) at build time configuration, such as the number of L3 cache slices configured. For certain configurations, a maximum number of reads and writes can be up to 128 for the CHI configured peripheral port.

The following table lists the read and write transaction capabilities of the CHI configured peripheral port.

Table 11-5: Attributes of the CHI configured peripheral port

Attribute	Value	Comment
Write issuing capability	Configuration dependent	This can range up to a maximum of 128, depending on configuration.
Read issuing capability	Configuration dependent	This can range up to a maximum of 128, depending on configuration.
Exclusive hardware access thread capability	Number of hardware threads	Each hardware thread can have one exclusive access sequence in progress.
Transaction ID width	12 bits	There is no fixed mapping between CHI transaction IDs and cores. Transaction IDs can be used for either reads or writes. **Note:** The source of the transaction is encoded in the LPID field, see Table 8-9: CHI LPID[4:0] bitfields on page 115.
Transaction ID capability	Configuration dependent	The transaction ID capability depends on the number of L3 cache slices configured, see the note following this table. There is never any ID reuse in CHI implementations, regardless of the memory type.
NodeID widths	11 bits	-
TXREQFLIT.RSVDC	0 bits	-
TXDATFLIT.RSVDC	0 bits	-
TXDATFLIT.DataCheck	0 bits	-

- For the write issuing and read issuing capabilities, the total issuing capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter.

  The peripheral port can support up to 128 transactions, or the total number of outstanding transaction for the cluster if this is less.
- The issuing capability described in this table is the maximum for the whole cluster. If you want to achieve the maximum performance available, then you can use these values to size interconnect capabilities. However, this maximum issuing capability might not be reached by a single core on its own. It might need multiple cores generating heavy memory traffic simultaneously to reach the maximum value. The capabilities vary by core type, for example high-performance cores typically generate more transactions than balanced-performance cores. It can also vary by memory type, with typically a significantly lower limit for Device or Non-cacheable transactions than for Cacheable transactions.

11.8 CHI peripheral port interface properties

AMBA defines a set of CHI interface properties that the interconnect can provide. The CHI configured peripheral port of the*DynamIQ™ Shared Unit-110* (DSU-110) cluster only supports some of these interface properties.

The following table shows which of these properties the CHI-configured peripheral port supports, and if interconnect or system support is required. You must ensure that your system interconnect, where applicable, supports these properties.

Table 11-6: CHI peripheral port interface properties

CHI property	Supported by the DSU-110 cluster	Interconnect support required
Atomic_Transactions	The DSU-110 cluster supports this property if BROADCASTATOMIC is HIGH.	Yes
Cache_Stash_Transactions	Yes	Yes
Direct_Memory_Transfer	Yes	Yes
Direct_Cache_Transfer	Yes	Yes
Data_Poison	The DSU-110 cluster supports this property if cache protection is enabled.	Yes
Data_Check	No	No
CCF_Wrap_Order	No	No
Enhanced_Features	The DSU-110 cluster supports data return from SC state. The DSU-110 cluster does not support input/output deallocation transactions, for example *ReadOnceMakeInvalid* (ROMI) and *ReadOnceCleanInvalid* (ROCI). The DSU-110 cluster supports ReadNotSharedDirty transactions and requires interconnect support. If **BROADCASTPERSIST** is HIGH, the DSU-110 cluster supports CleanSharedPersist transactions and requires interconnect support.	Yes, if the cluster supports ReadNotSharedDirty transactions or if the **BROADCASTPERSIST** signal is set to HIGH.

The following table shows the different width values that the CHI-configured peripheral port supports.

Table 11-7: Supported widths for CHI-configured peripheral port

Width	Value
Req_Addr_Width	The maximum width is 52 bits
NodeID_Width	The maximum width is 11 bits
Data_Width	The maximum width is 256 bits

11.9 CHI peripheral port transactions

The CHI configured peripheral port of DynamIQ™ Shared Unit-110 (DSU-110) supports the same CHI transactions as the CHI configured main master interface.

The following table shows the read and write transactions supported by the CHI-configured peripheral port of the DSU-110.

Table 11-8: CHI read and write transactions supported by DSU-110

Transaction	Operation	Produced by DSU-110
AtomicCompare	Atomic instruction that is not allocating inside the cluster	Yes
AtomicLoad	Atomic instruction that is not allocating inside the cluster	Yes
AtomicStore	Atomic instruction that is not allocating inside the cluster	Yes
AtomicSwap	Atomic instruction that is not allocating inside the cluster	Yes
CleanInvalid	Cache maintenance instructions	Yes
CleanShared	Cache maintenance instructions	Yes
CleanSharedPersist	Not used. CleanSharedPersistSep is used instead.	No
CleanSharedPersistSep	Cache maintenance instructions. The Data Cache Clean to the Point of Persistence (DC CVAP) cache maintenance instruction generates this transaction when the **BROADCASTPERSISTMP** input signal is HIGH.	Yes
CleanUnique	Not used	No
DVMOp	*Translation Lookaside Buffer* (TLB) and instruction cache maintenance instructions when enabled by the **BROADCASTTLBIINNER**, **BROADCASTTLBIOUTER**, and **BROADCASTICINVAL** input signals.	Yes
Evict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
MakeInvalid	Not used	No
MakeReadUnique	Store instructions when the line is already cached in a Shared state inside the cluster. This includes store exclusive instructions, which set Excl HIGH.	Yes
MakeUnique	Store instructions of a full cache line of data that miss in the caches	Yes
PCrdReturn	Not used	No
PrefetchTgt	Hardware prefetch hint to the memory controller	Yes
ReadClean	Reading *Memory Tagging Extension* (MTE) tags for a Cacheable shareable line that is already cached in the cluster without tags	Yes
ReadNoSnp	Non-cacheable loads or instruction fetches, or cache linefills of Non-shareable cache lines into L1 or L2 caches	Yes
ReadNoSnpSep	Not used	No
ReadNotSharedDirty	Cache data linefills started by a load instruction, or cache linefills started by an instruction fetch	Yes
ReadOnce	Cacheable shareable instruction fetches that are not allocating into a coherent cache	Yes
ReadOnceCleanInvalid	Not used	No
ReadOnceMakeInvalid	Not used	No
ReadPreferUnique	Speculative store to Cacheable shareable memory or, if Excl is HIGH, a load exclusive instruction	Yes
ReadShared	Not used	No
ReadUnique	Cache data linefills started by a store instruction	Yes

Transaction	Operation	Produced by DSU-110
ReqLCrdReturn	Link credit return	Yes
StashOnceSepShared	Cache prefetch when the L3 cache is not present or powered down and configured by the CLUSTERECTLR_EL1	No
StashOnceSepUnique	Cache prefetch when the L3 cache is not present or powered down and configured by the CLUSTERECTLR_EL1	No
StashOnceShared	Not used	No
StashOnceUnique	Not used	No
WriteBackFull	Evictions of dirty cacheable shareable lines from the cluster	Yes
WriteBackFullCMO	Cache maintenance instruction evicting a dirty shareable cache line	Yes
WriteBackPtl	Not used	No
WriteCleanFull	Evictions of dirty lines from the L3 cache, when the line is still present in an L1 or L2 cache	Yes
WriteCleanFullCMO	Cache maintenance instruction cleaning a dirty shareable cache line	Yes
WriteEvictFull	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
WriteEvictOrEvict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
WriteNoSnpFull	Non-cacheable store instructions. Evictions of Non-shareable cache lines.	Yes
WriteNoSnpFullCMO	Cache maintenance instruction evicting a dirty Non-shareable cache line	Yes
WriteNoSnpPtl	Non-cacheable store instructions	Yes
WriteNoSnpPtlCMO	Not used	No
WriteNoSnpZero	Write of zeroes to Non-cacheable or Non-shareable memory using the DC ZVA instruction	Yes
WriteUniqueFull	Cacheable writes of a full cache line not allocating into L1, L2, or L3 caches, for example streaming writes	Yes
WriteUniqueFullCMO	Not used	No
WriteUniqueFullStash	Not used	No
WriteUniquePtl	Generated as a result of *Accelerator Coherency Port* (ACP) WriteUniquePtl transactions when not allocating to the L3 cache	Yes
WriteUniquePtlCMO	Not used	No
WriteUniquePtlStash	Not used	No
WriteUniqueZero	Write of zeroes to a Shareable cache line using the DC ZVA instruction	Yes

11.10 Read and write capabilities and transaction ID encoding

The issuing capabilities and the AXI transaction ID encoding of the peripheral port depends on if 64-bit mode or 256-bit mode is configured.

64-bit AXI peripheral port read and write capabilities

Both the read and write issuing capabilities depend on the total number of LPIDs in the cluster. The total number of LPIDs in the cluster is the sum of:

- The number of LPIDs for the cores. Each core has one unique LPID. For example, a cluster of four cores would have four LPIDs.
- One LPID for L3 Evicts
- One LPID for ACP

Therefore, the maximum number of LPIDs for cluster of 12 cores is 14 LPIDs.

The following table describes the read and write issuing capabilities of the peripheral port when configured as AXI 64-bit mode.

Table 11-9: AXI issuing capabilities

Attribute	Value	Comments
Write issuing capability	Up to a maximum of 4 times total number of LPIDs	For the cluster of 12 cores, the maximum issuing capability is 56 outstanding reads or writes. **Note:** These value do not mean that each LPID is only allowed four outstanding transactions. Each LPID can use as much of the remaining resource as required. For example, LPID0 can consume more than 4 queue entries.
Read issuing capability	Up to a maximum of 4 times total number of LPIDs	
Write ID capability	Configuration dependent	All transactions from a given LPID use the same AXI ID.
Read ID capability	Configuration dependent	All transactions from a given LPID use the same AXI ID.
AWID width	6 bits	-
ARID width	6 bits	-

The issuing capability described in this table is the maximum for the whole cluster. If you want to achieve the maximum performance available, then you can use these values to size interconnect capabilities. However, this maximum issuing capability might not be reached by a single core on its own. It might need multiple cores generating heavy memory traffic simultaneously to reach the maximum value. The capabilities vary by core type, for example high-performance cores typically generate more transactions than balanced-performance cores. It can also vary by memory type, with typically a significantly lower limit for Device or Non-cacheable transactions than for Cacheable transactions.

The following table lists the encoding for AXI transaction IDs for the Peripheral port when configured as AXI 64-bit mode.

Table 11-10: AXI transaction ID encoding

Attribute	Value	Comments
All IDs	`0b0r_yyyy`	The value comprises the following fields: **r** Can be 0 or 1 **yyyy** LPID number


Note

These ID and transaction details are provided for information only. Arm strongly recommends that all interconnects and peripherals are designed to support any type and number of transactions on any ID to ensure compatibility with future products.

256-bit AXI peripheral port read and write capabilities

See 9.4 AXI 256-bit master interface attributes on page 121 for the read and write issuing capabilities for a peripheral port configured in 256-bit mode.

See the *AMBA® AXI and ACE Protocol Specification* for more information about the AXI signals described in this manual.

11.11 Peripheral port and ACP interface usage

When using a 256-bit CHI or 256-bit AXI configured peripheral port, ensure the peripheral port and main master ports complete their accesses independently of the *Accelerator Coherency Port* (ACP) interface to avoid a system deadlock. Alternatively use the 64-bit AXI configured peripheral port which does not have this restriction.

There are two main use-cases for the peripheral port:

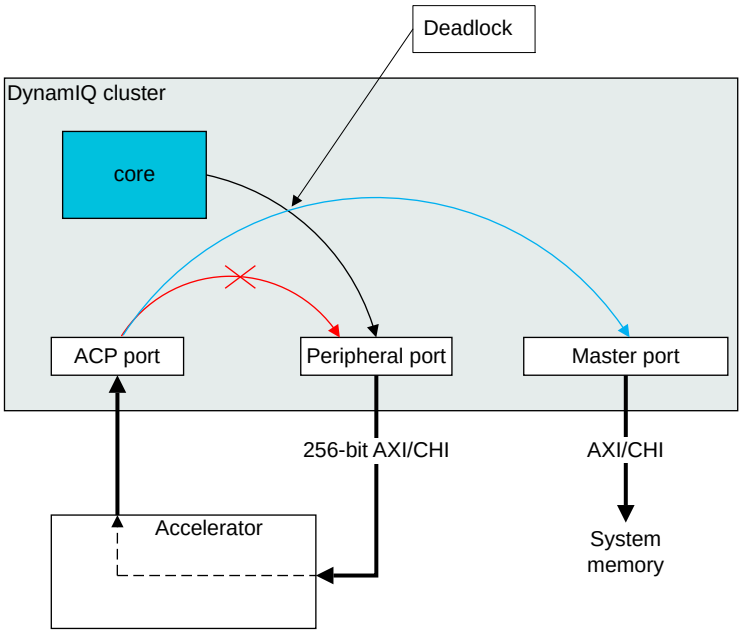
- For connecting to a tightly-coupled accelerator.
- For use as a more general system port.

If the system is not correctly designed, both of these use cases can result in deadlock scenarios.

Deadlock condition when peripheral port is connected to an accelerator

The following figure shows an example of how the deadlock condition can arise when the peripheral port of the *DynamIQ™ Shared Unit-110* (DSU-110) is connected to a tightly-coupled accelerator.

Figure 11-1: Deadlock scenario when peripheral port is connected to an accelerator



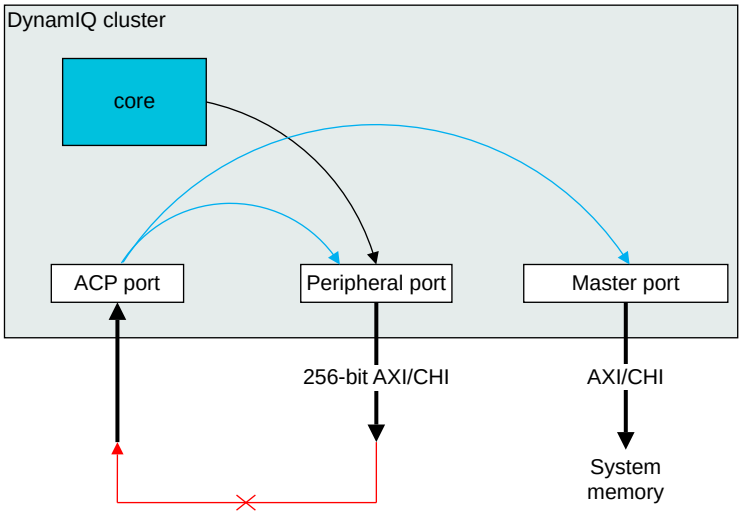
When the peripheral port is connected to a tightly-coupled accelerator, the accelerator might have an internal dependency, this is shown by the dashed line in the preceding diagram. This means, a read or write to the registers of the accelerator through the peripheral port cannot complete until an outstanding transaction that it has started on the *Accelerator Coherency Port* (ACP) completes. Because of this dependency, if an ACP access is routed to the peripheral port (shown by the red arrow in the preceding diagram) then it creates a circular dependency which can result in a system deadlock. Therefore, when the peripheral port is configured as 64-bit mode, any ACP access to the peripheral port address range receives a SLVERR response. Therefore, the ACP cannot access the peripheral port. This illegal condition is shown by the red cross between the ACP port and peripheral port.

If an ACP access is routed to the main master ports, then it travels down the same pipeline as accesses from the core. This is shown where the black and blue arrows cross over each other in the preceding diagram. This could create a circular dependency between the accesses. However, when the peripheral port is configured in 64-bit mode there is additional logic in the cluster that ensures that ACP and core traffic do not depend on each other. Therefore, the deadlock is avoided.

Deadlock condition when peripheral port is used as a system port

The following figure shows an example of how a deadlock condition can arise when the peripheral port is used as a general system port.

Figure 11-2: Deadlock scenario when peripheral port is used as general system port



When the peripheral port is used as a general system port, ACP traffic is allowed to access the peripheral port and the access completes normally. This is shown by the blue line between the ACP port in the preceding diagram. Therefore, when the peripheral port is configured in 256-bit mode, the system must ensure that peripheral port accesses can complete independently without requiring any process on ACP. However, if there is a dependency between the peripheral port and ACP port, shown by the red arrow with a cross in the preceding figure, then the system could deadlock.

11.12 AXI peripheral port privilege information

AXI provides information about the privilege level of accesses on the **ARPROTM[0]** and **AWPROTM[0]** signals. This information is not available from cores within the cluster. Therefore these signals are always driven HIGH indicating that the access could be a privileged access.

12. RAS extension support

The *DynamIQ™ Shared Unit-110* (DSU-110) supports the *Reliability, Availability, Serviceability* (RAS) Extension, including all extensions up to Arm®v9.0-A. You can optionally enable *Error Correcting Code* (ECC) support for the L3 cache RAMs and snoop filter RAMs at build time configuration.

The DSU-110 supports:

- Cache protection with ECC on the L3 cache RAMs and snoop filter RAM
- Poison attribute on bus transfers
- Error Data Record registers
- *Fault Handling Interrupts* (FHIs)
- *Error Recovery Interrupts* (ERIs)
- *Critical Error Interrupts* (CRIs)
- Error injection

Node 0 observed by the cores includes the L3 memory system for the DSU-110. For other nodes observed by the core or complex, see your core *Technical Reference Manual* (TRM).

For more information on the architectural RAS Extension and the definition of a node, see the *Arm® Reliability, Availability, and Serviceability (RAS) Specification Armv8, for the Armv8-A architecture profile*.

12.1 Cache protection behavior

The configuration of the *Reliability, Availability, Serviceability* (RAS) Extension that is implemented in the *DynamIQ™ Shared Unit-110* (DSU-110) includes *Error Correcting Code* (ECC) cache protection. In this case, the DSU-110 protects against errors that result in a RAM bitcell holding the incorrect value.

The RAMs in the DSU-110 support *Single Error Correct Double Error Detect* (SECDED). SECDED allows detection and correction of any 1-bit error, and detection of any 2-bit error in all protected RAMs. When the datum and code bits are all-zero, or all-one, the interpretation is that an error has occurred that the *Error Correcting Code* (ECC) scheme cannot correct. However, it might be corrected by other means, such as refetching cached data.

The following table describes the protection type is applied to each RAM. The DSU-110 can progress and remain functionally correct when there is a single bit error in any RAM.

Table 12-1: RAM cache protection

RAM	Protection	Description
L3 data cache data	ECC, SECDED	9 ECC bits per 132 bits

RAM	Protection	Description
L3 cache tag	ECC, SECDED	The number of ECC bits for each 58-bits depend on the size of entry as follows: • If the tag entry is 58 bits wide, there are 8 ECC bits. • If the tag entry is 57 bits wide or less, there are 7 ECC bits.
L3 cache victim	None	-
Long-Term Data Buffer (LTDB) RAMs	ECC, SECDED	9 ECC bits per 145 bits
Snoop filter RAMs	ECC, SECDED	7 ECC bits per 48 bits

Error correction

If there are multiple single bit errors in different RAMs, or within different protection granules within the same RAM, then the DSU-110 remains functionally correct.

If there is a double bit error in a single RAM within the same protection granule, the DSU-110 detects and either reports or defers the error, as consistent with SECDED behavior. If the error is in a cache line containing dirty data, then that data might be lost.

If there are three or more bit errors within the same protection granule, then depending on the RAM and the position of the errors within the RAM, the DSU-110 might or might not detect the errors. The cache protection feature of the DSU-110 has a minimal performance impact when no errors are present.

When a correctable error is detected in the L3 cache data RAMs, the data is corrected inline before returning to the requestor.

When a correctable error is detected in the L3 cache tag RAMs or the snoop filter RAMs the following correction mechanism is used:

- The value is corrected and written back to the source address (Read-Correct-Write).
- The lookup is replayed.

The DSU-110 has extra hardware that provides limited support for hard error correction. A hard error is a physical error in the RAM that prevents the correct value being written. A single hard error can be corrected and is guaranteed to make progress. However, if there are multiple hard errors then, in some cases, this can cause live-locks as the line could continuously replay.

Uncorrectable errors and Data poisoning

If an error is detected as having 2 bits in error in a RAM protected by ECC, then this error is not correctable. In this case, the behavior depends on the type of RAM, as follows:

Data RAM or Long-Term Data Buffer RAM

When an uncorrectable error is detected in an L3 data RAM or *Long-Term Data Buffer* (LTDB) RAM, the chunk of data with the error is marked as poisoned. This poison status is then transferred with the data and stored:

- In the cache, if the data is allocated back into a cache.
- In the LTDB RAM, if the data is moved there.

The poison status is stored for every 64 bits of data.

If the interconnect supports poisoning, then the poison status is transferred with the data when the line is evicted or snooped from the cluster. No abort is generated when a line is poisoned. The abort is deferred until a load or instruction fetch consumes the poisoned data.

If the interconnect does not support poisoning and a poisoned cache line is evicted or snooped from the cluster, then the DSU-110 generates an interrupt, **nCLUSTERERRIRQ**, to notify software that data has potentially been lost.


Note

Software can indicate if the interconnect supports poisoning or not by setting the interconnect data poisoning support bit in the Extended Control Register of the cluster. For details, see A.1.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register on page 222 or B.1.1.13 CLUSTERECTLR, Cluster Extended Control Register on page 362 depending on how you are accessing the register.

Tag RAM

When an uncorrectable error is detected in an L3 tag RAM, then either the address or coherency state of the line is **UNKNOWN**, so the data cannot be poisoned. In this case, the line is invalidated and the DSU-110 generates an interrupt, **nCLUSTERERRIRQ**, to notify software that data has potentially been lost.

Snoop filter tag RAM

When an uncorrectable error is detected in a snoop filter tag RAM, either the address or coherency state of the line is **UNKNOWN**, so the data cannot be poisoned. In this case, the snoop filter entry is invalidated, but the line remains present in one or more of the cores. The DSU-110 generates an interrupt, **nCLUSTERCRITIRQ**, to notify software that data has potentially been lost.


Note

Arm recommends that a system reset is performed as soon as possible, in response to this interrupt. This is because the core caches and the snoop filter are inconsistent after this error, which can lead to **UNPREDICTABLE** behavior. The effect of the error depends on the type of core, but it could result in further data corruption, or deadlocks, making it impossible to cleanly recover from such an error.

12.2 Error containment

The *DynamIQ™ Shared Unit-110* (DSU-110) supports error containment, which means that an error is detected and not silently propagated.

Error containment also implies support for poisoning if there is a double error on an eviction. This ensures that the error of the associated data is reported when it is consumed.

Support for the *Error Synchronization Barrier* (ESB) instruction in the core also allows further isolation of imprecise exceptions that are reported when poisoned data is consumed.

12.3 Fault detection and reporting

When the *DynamIQ™ Shared Unit-110* (DSU-110) detects a fault, it raises a *Fault Handling Interrupt* (FHI) exception through the fault signals. FHIs are reflected in the Error Data Record Registers that are updated in the node that detects the errors.

Fault handling interrupt

When ERR0CTLR.FI is set, all Deferred errors and Uncorrected errors that the DSU-110 detects generate an FHI through the **nCLUSTERFAULTIRQ** signal.

When ERR0CTLR.CFI or any other CE-counter overflow bits are set, then all detected Corrected errors also cause an FHI to be generated.

Error recovery interrupt

When ERR0CTLR.UI is set, all Uncorrected errors that are detected and not deferred generate an error recovery interrupt through the **nCLUSTERERRIRQ** signal.

Critical error interrupt

When ERR0CTLR.CI is set, all critical errors that the DSU-110 detects generate a critical error interrupt on the **nCLUSTERCRITIRQ** signal.

Clearing reported faults

The signals **nCLUSTERFAULTIRQ**, **nCLUSTERERRIRQ**, and **nCLUSTERCRITIRQ** remain asserted until software clears them by writing to the ERR0STATUS register.

12.4 Error detection and reporting

When the *DynamIQ™ Shared Unit-110* (DSU-110) consumes an error, it raises an *Error Recovery Interrupt* (ERI).

Error detection and reporting registers

The following registers are provided:

- The cluster Error Record Feature Register, CLUSTERRAS_ERR0FR. This is a read-only register that specifies various error record settings.
- The cluster Error Record Control Register, CLUSTERRAS_ERR0CTLR.
- The cluster Error Record Miscellaneous Register 0-3, CLUSTERRAS_ERR0MISC0-3. These registers record details of the error location and counts.
- The cluster Pseudo-fault Generation Feature register, CLUSTERRAS_ERR0PFGF. Read-only register.
- The cluster Error Record Primary Status Register, CLUSTERRAS_ERR0STATUS.

The cluster *Reliability, Availability, and Serviceability* (RAS) registers are accessible either from memory-mapped accesses on the utility bus or from System register accesses from the cores.

Error types

The following describes the different types of errors that can occur in the DSU-110 and their effects:

- Corrected errors.
- Uncorrectable errors in the L3 data RAMs when read by a core can cause a precise or imprecise Data Abort or Prefetch Abort, depending on the implementation of the core.
- Uncorrectable errors in the L3 data RAMs in a line when this line is being evicted from a cache cause the data to be poisoned. The eviction might be because of a natural eviction, a linefill from a higher level of cache, a cache maintenance operation, or a snoop. If the poisoned line is evicted from the cluster for any reason and the interconnect does not support data poisoning, then the **nCLUSTERERRIRQ** signal is asserted.
- Uncorrectable errors in the L3 tag RAMs or *Snoop Control Unit* (SCU) filter RAMs cause the **nCLUSTERERRIRQ** signal to be asserted.


Note

Arm recommends that the **ERRIRQ** signals are connected to the interrupt controller, so that an interrupt or system error is generated when the signals are asserted.

The fault and error interrupt pins can be cleared by writing to the CLUSTERRAS_ERR0STATUS register.

When a dirty cache line with an error on the data RAMs is evicted from the cluster, the write on the master interface still takes place. However, if the error is uncorrectable then:

- If the DSU-110 is configured with an AXI master-port, the uncorrected data is written and the error is reported in the RAS registers.
- If the DSU-110 is configured with a CHI master-port, the uncorrected data is written but the data poison field indicates that there is a data error.

When a snoop hits on a line with an uncorrectable data error, the following happens:

- If the snoop requires the data, then the data is returned.
- If the DSU-110 is configured with a CHI master-port, the snoop response indicates that either the data is poisoned (if supported), or that there is an error.

If a snoop hits on a tag that has an uncorrectable error, then it is treated as a snoop miss. Because the error means that it is not known whether the cache line is valid.

If an *Accelerator Coherency Port* (ACP) access reads a cache line with an uncorrectable error, then it returns an ACP response to indicate a slave error.

Sometimes an error can be counted multiple times. For example, multiple accesses might read the location with the error before the line is evicted.

12.4.1 Error reporting and performance monitoring

All detected memory errors and *Error Correcting Code* (ECC) errors trigger the MEMORY_ERROR event.

The MEMORY_ERROR event is counted by the *Performance Monitoring Unit* (PMU) counters if it is selected and the counter is enabled.

In Secure state, the event is counted only if IMP_CLUSTERPMMDCR_EL3.SPME is asserted.

Related information

17.2 PMU events on page 203

12.4.2 Errors not counted

At most, one error can be counted per clock cycle even if there are multiple Corrected errors, sources, or both errors and sources.

12.4.3 Double error reporting

If the DSU-110 detects an *Error Correcting Code* (ECC) error in the L3 data RAM, the DSU-110 performs a two-stage sequence that typically causes it to report two errors in the Error Record registers, even though there was only one original error.

This occurs because when the DSU-110 detects an error in the L3 data RAM, the DSU-110 reports the error in the Error Record registers and moves the data to the *Long-Term Data Buffer* (LTDB) RAM without correcting it. The LTDB RAM then reads the data and corrects it. When this occurs, the DSU-110 reports a second error in the Error Record registers. Therefore, an ECC error in the L3 Data RAM is reported as if two errors occurred.

An error on a single read of the L3 data RAM results in the following error record contents, assuming the Error Record was initially empty:

- A 1-bit error increments the ERR0MISC0.CECO due to the reporting of a second Correctable Error. The contents of the ERR0STATUS accurately shows that the error came from the L3 Data RAM. For example, ERR0STATUS.SERR=6, ERR0STATUS.V=1 and ERR0STATUS.CE=1.
- A 2-bit error might be Deferred or Uncontainable depending on whether the target of the data supports poison. This is determined during the LTDB RAM read. The L3 data RAM always generates a Deferred Error, if there is a 2-bit error.

  Depending on if the error is Deferred or Uncontainable, the Error Record is updated as follows:

  - For a Deferred error, the contents of the Error Record accurately shows the error that came from the L3 data RAM. For example, ERR0STATUS.V=1, ERR0STATUS.DE=1 and ERR0STATUS.SERR=6. However, the extra error from the LTDB RAM also sets ERR0STATUS.OF=1.

- For an Uncontainable error, the contents of the Error Record shows the LTDB RAM error. However, it does not provide details of the original L3 data RAM error. For example, ERR0STATUS.V=1, ERR0STATUS.UE=1, ERR0STATUS.SERR=2. The extra error also means that ERR0STATUS.OF=1. Also even though L3 data RAM poisoned the data, ERR0STATUS.PN=0.

12.5 Error injection

Error injection is used to test out the error detection reporting and recording structure by deliberately inserting errors into the error reporting logic.

The injected errors are pseudo-errors only. They cause a report of an error to be signaled but the error injection does not corrupt the target location. Therefore, an injected pseudo error does not cause any automatic error correction logic to be activated.

Error injection uses the error injection and reporting registers to insert errors. The *DynamIQ™ Shared Unit-110* (DSU-110) can inject any of the following error types:

- *Corrected Error* (CE)
- *Deferred Error* (DE)
- *Uncontainable Errors* (UC)
  - UC error that is a *Critical (CI) error*

An error can be injected immediately or when a 32-bit counter reaches zero. You can control the value of the counter through the ERRPFGCDN_EL1 register. The value of the counter decrements on a per clock cycle basis.

Pseudo-errors are injected using the CLUSTERRAS-ERR0PFGCTL, Pseudo-fault Generation Control Register.

Pseudo-errors are triggered by either reads to the snoop filter RAM instances or *Long-Term Data Buffer* (LTDB) RAMs depending on the type of error that is programmed.

Errors triggered by reads to the snoop filter RAMs

A UC pseudo-error which is a CI error, can be triggered on a look-up in the snoop filter RAM instances. Arm expects that the execution of typical software triggers the pseudo fault. The pseudo fault can be triggered deliberately by executing a sequence of consecutive load or store transactions to a shareable, cacheable address range where the addresses are not currently cached in the core caches.

Errors triggered by reads to the LTDB RAMs

All three error types (DE, CE, and UC) which are non-critical errors, can be triggered when there is a read of the LTDB RAM instances. Reads of the LTBD RAMs are most likely to be triggered by either:

- Normal, Non-cacheable, store transactions from the core to the cluster.

- Dirty cache-line evictions from the core to the cluster.

Arm expects that the execution of typical software triggers the pseudo fault. The pseudo fault can be deliberately triggered by executing a sequence of consecutive Normal Non-cacheable stores to a Normal Non-cacheable address range


Note

The error injection mechanism only injects pseudo fault reports into the error reporting registers for the purposes of testing error handling and error identification software in real systems. It does not inject actual errors into the hardware.

Related information

B.1.3 External cluster RAS register summary on page 385
A.3 AArch64 RAS register summary on page 308

12.6 ECC errors during power transitions

If an error in a RAS register occurs while the cluster is powering down then the cluster is prevented from powering down.

As part of the powerdown sequence, software on the core disables or reroutes interrupts away from the core before it executes the WFI instruction.

Therefore the core software cannot be interrupted to manage any RAS fault or error, which is either:

- Detected before the core powerdown procedure the WFI instruction and is not cleared or
- Detected after the core powerdown executes the WFI instruction.

Any RAS fault or error interrupt output from an active cluster prevents the cluster from powering down. This results in scenarios such as:

- The cluster is left powered up but the core software will be inactive.
- All requests from the cluster PPU to power down the cluster will be denied until the error interrupt is cleared in the cluster RAS registers.

As part of the cluster powerdown, the RAS fault and error interrupts status must be managed to avoid this situation.

Managing RAS fault and error interrupts during the cluster powerdown

To manage RAS fault and error interrupts during the cluster powerdown, the following two options can be executed:

Disabling the cluster RAS fault and error generation

1. Disable the generation of the cluster RAS fault and error interrupts by using the ERXCTLR_EL1 registers.

2. Clear any current RAS fault or error interrupts before the core powerdown executes the WFI instruction.

Re-route the cluster RAS fault or error interrupts

Re-route the cluster RAS fault or error interrupts to a separate system error management device as part of the powerdown procedure. This device, for example a *System Control Processor* (SCP), is responsible for clearing the cluster RAS interrupts using the utility bus if a fault or error is signaled.

This approach is only possible if the system is designed to allow the RAS fault or error interrupt outputs to be re-routed to another component.

If all the cluster RAS fault and error interrupt outputs are disabled before the core powerdown, but the error detection is still enabled, then the following occurs:

- Any correctable errors are corrected.
- Any deferrable errors are deferred as part of the automatic cache clean and invalidation procedures.
- The *Error records* for these correctable and deferrable errors are lost once the cluster is powered down.
- If there is an uncorrectable error when the cluster is powering down, then this error is not signalled to the system. Therefore, this uncorrectable error might corrupt the system behaviour.

In some systems it can be preferable to disable the RAS fault and error interrupt subset generation. The following process shows an example of how this can be done:

- Disable the interrupts for correctable and deferrable errors.
  - ERXCTLR_EL1.CFI = 0
  - ERXCTLR_EL1.FI = 0
- Enable the error interrupt for uncorrectable errors.
  - ERXCTLR_EL1.UI = 1
- Cluster error interrupt outputs must be routed to the system error manager before executing the WFI instruction in the core powerdown procedure.

Using this approach, the cluster error interrupt outputs must be re-routed to the system error manager before executing the WFI instruction in the core powerdown procedure.

If an uncorrectable error occurs during the powerdown, the cluster will be powered on, but the core software will be inactive. Then the system error manager is responsible for clearing the cluster RAS interrupt register so that the powerdown procedure can proceed. To use this approach, the system must be designed to allow the cluster RAS error interrupts to be re-routed to the system error manager. The system error manager is able to identify where the uncorrectable error occurred in the cluster and clear the RAS error interrupt by accessing the cluster RAS registers using the utility bus.

12.7 Cluster RAS registers

The cluster *Reliability, Availability, and Serviceability* (RAS) registers are treated as a separate node in the memory-mapped view. The cluster RAS registers are accessible either from memory-mapped accesses on the Utility bus or from System register accesses from the cores. You must access the cluster RAS registers from the Secure address space.

Note

The cluster RAS registers are treated as **RAZ/WI** if either:

- The register is marked as Reserved.
- The register is in wrong Security state.

Also, any address that is not documented is treated as **RAZ/WI**.

12.7.1 AArch64 RAS registers

The summary table provides an overview of all *Reliability, Availability, and Serviceability* (RAS) registers with *IMPLEMENTATION DEFINED* values such as AArch64 performance monitors and AArch64 RAS in the *DynamIQ™ Shared Unit-110* (DSU-110). Individual register descriptions provide detailed information.

Note

- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 12-2: RAS registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
ERXFR_EL1	3	0	C5	C4	0	—	64-bit	Selected Error Record Feature Register	No
ERXCTLR_EL1	3	0	C5	C4	1	—	64-bit	Selected Error Record Control Register	No
ERXSTATUS_EL1	3	0	C5	C4	2	—	64-bit	Selected Error Record Primary Status Register	No
ERXPFGF_EL1	3	0	C5	C4	4	—	64-bit	Selected Pseudo-fault Generation Feature Register	No
ERXPFGCTL_EL1	3	0	C5	C4	5	—	64-bit	Selected Pseudo-fault Generation Control Register	No
ERXPFGCDN_EL1	3	0	C5	C4	6	—	64-bit	Selected Pseudo-fault Generation Countdown Register	No
ERXMISC0_EL1	3	0	C5	C5	0	—	64-bit	Selected Error Record Miscellaneous Register 0	No
ERXMISC1_EL1	3	0	C5	C5	1	—	64-bit	Selected Error Record Miscellaneous Register 1	No
ERXMISC2_EL1	3	0	C5	C5	2	—	64-bit	Selected Error Record Miscellaneous Register 2	No
ERXMISC3_EL1	3	0	C5	C5	3	—	64-bit	Selected Error Record Miscellaneous Register 3	No

12.7.2 External cluster RAS registers

The cluster *Reliability, Availability, and Serviceability* (RAS) registers are memory-mapped onto the utility bus. You can only access these registers from the Secure address space.

The summary table provides an overview of all the cluster RAS registers in DSU-110. Individual register descriptions provide detailed information.


Note

- The cluster RAS registers are treated as **RAZ/WI** if a Non-secure access is made to them.
- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- The base address for the cluster RAS registers is `0x020000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 12-3: CLUSTERRAS registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	CLUSTERRAS_ERR0FR	—	64-bit	Error Record Feature Register	No
0x008	CLUSTERRAS_ERR0CTLR	—	64-bit	Error Record Control Register	No
0x010	CLUSTERRAS_ERR0STATUS	—	64-bit	Error Record Primary Status Register	No
0x018	CLUSTERRAS_ERR0ADDR	—	64-bit	Error Record Address Register	No
0x020	CLUSTERRAS_ERR0MISC0	—	64-bit	Error Record Miscellaneous Register 0	No
0x028	CLUSTERRAS_ERR0MISC1	—	64-bit	Error Record Miscellaneous Register 1	No
0x030	CLUSTERRAS_ERR0MISC2	—	64-bit	Error Record Miscellaneous Register 2	No
0x038	CLUSTERRAS_ERR0MISC3	—	64-bit	Error Record Miscellaneous Register 3	No
0x800	CLUSTERRAS_ERR0PFGF	—	64-bit	Pseudo-fault Generation Feature Register	No
0x808	CLUSTERRAS_ERR0PFGCTL	—	64-bit	Pseudo-fault Generation Control Register	No
0x810	CLUSTERRAS_ERR0PFGCDN	—	64-bit	Pseudo-fault Generation Countdown Register	No
0xE00	CLUSTERRAS_ERRGSR	—	64-bit	Error Group Status Register	No
0xE10	CLUSTERRAS_ERRIIDR	—	32-bit	Implementation Identification Register	No
0xFA8	CLUSTERRAS_ERRDEVAFF	—	64-bit	Device Affinity Register	No
0xFBC	CLUSTERRAS_ERRDEVARCH	—	32-bit	Device Architecture Register	No
0xFC8	CLUSTERRAS_ERRDEVID	—	32-bit	Device Configuration Register	No
0xFD0	CLUSTERRAS_ERRPIDR4	—	32-bit	Peripheral Identification Register 4	No
0xFD4	CLUSTERRAS_ERRPIDR5	—	32-bit	Peripheral Identification Register 5	No
0xFD8	CLUSTERRAS_ERRPIDR6	—	32-bit	Peripheral Identification Register 6	No
0xFDC	CLUSTERRAS_ERRPIDR7	—	32-bit	Peripheral Identification Register 7	No
0xFE0	CLUSTERRAS_ERRPIDR0	—	32-bit	Peripheral Identification Register 0	No
0xFE4	CLUSTERRAS_ERRPIDR1	—	32-bit	Peripheral Identification Register 1	No
0xFE8	CLUSTERRAS_ERRPIDR2	—	32-bit	Peripheral Identification Register 2	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFEC	CLUSTERRAS_ERRPIDR3	—	32-bit	Peripheral Identification Register 3	No
0xFF0	CLUSTERRAS_ERRCIDR0	—	32-bit	Component Identification Register 0	No
0xFF4	CLUSTERRAS_ERRCIDR1	—	32-bit	Component Identification Register 1	No
0xFF8	CLUSTERRAS_ERRCIDR2	—	32-bit	Component Identification Register 2	No
0xFFC	CLUSTERRAS_ERRCIDR3	—	32-bit	Component Identification Register 3	No

13. Utility bus

The utility bus provides access to control registers for various system components in the *DynamIQ™ Shared Unit-110* (DSU-110) and the cores within the DSU-110 DynamIQ™ cluster. The utility bus is implemented as a 64-bit AMBA AXI5 slave port, and the control registers are memory-mapped onto the utility bus.

The utility bus provides access to the following system functions:

- *Power Policy Unit* (PPU) registers for the cluster and each of the cores
- Cluster control registers, including the L3 cache power-related monitors
- *Reliability, Availability, and Serviceability* (RAS) registers for the cluster
- *Memory Partitioning and Monitoring* (MPAM) registers for the cluster
- *Activity Monitor Unit* (AMU) registers in the cores
- *Maximum Power Mitigation Mechanism* (MPMM) registers in the cores

Information about the PPU registers for the cores in the cluster is provided in this document. For information on all the other core registers accessible from the utility bus, see your core *Technical Reference Manual* (TRM).

13.1 Utility bus accesses

Transactions on the utility bus comply with a subset of the AXI 5 bus protocol. Access sizes must be either 32-bits or 64-bits. Any other sized access generates a SLVERR response from the utility bus.

You must observe the following requirements when accessing the utility bus:

- Only ReadNoSnoop and WriteNoSnoop transaction types are supported.
- Only 32-bit accesses or 64-bit accesses are supported. Therefore, **ARSIZEU** or **AWSIZEU** must be either `0b010` for 32-bit sized accesses, or `0b011` for 64-bit sized accesses. Any other access size generates a SLVERR response from the utility bus.
- Only single beat bursts are supported. Therefore, **ARLENU** or **AWLENU** must be `0b00000000`. Any other burst length generates a SLVERR response from the utility bus.
- Some of the system components control registers only support Secure accesses on the utility bus, see Table 13-3: Utility bus base addresses for system component registers on page 167. Ensure that you access any system component register with the Secure bit set appropriately. Any register in the wrong Security state is treated as **RAZ/WI**.

Arm® recommends the following, when accessing the utility bus:

- **ARCACHEU** or **AWCACHEU** is either `0b0000` or `0b0001`, although other values are accepted.

- **ARBURSTU** or **AWBURSTU** is `0b01`, although other values are accepted.
- **ARLOCKU** or **AWLOCKU** is tied LOW, as there is no exclusive monitor present.

The following table describes the utility bus acceptance capabilities:

Table 13-1: Utility bus acceptance capabilities

Attribute	Value	Description
Write acceptance capability	1	The utility bus can accept 1 write transaction.
Read acceptance capability	1	The utility bus can accept 1 read transaction.
Combined acceptance capability	2	The utility bus can accept up to 2 transactions.

13.1.1 Core access to system component registers

Some of the system component registers are only available through memory-mapped accesses on the utility bus. For these registers, there is no direct access to the registers from the cores. If you require memory-mapped access from the cores, Arm® recommends allowing your interconnect to provide a loopback address mapping for the cores to access the utility bus through your interconnect.

The following table shows which system components are directly accessible from the cores using System register access instructions, as well as being accessible over the utility bus.

Table 13-2: System component registers accessible from cores

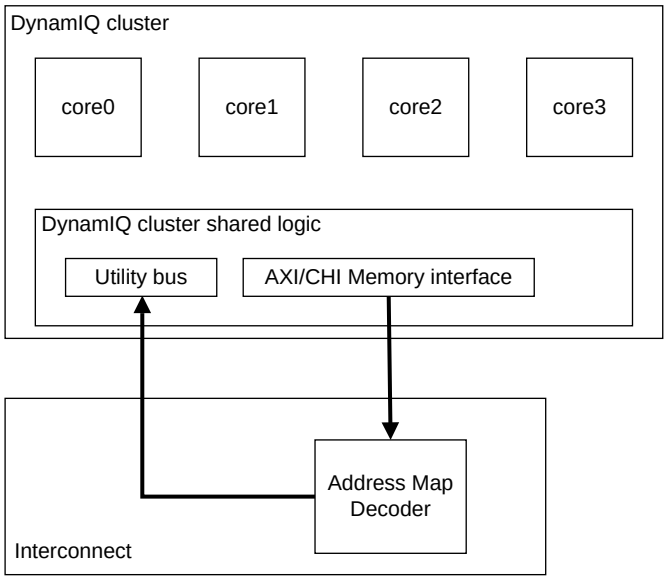
Registers	Directly accessible from cores
Cluster power control	Yes
Cluster *Memory System Resource Partitioning and Monitoring* (MPAM)	No
Cluster *Reliability, Availability, and Serviceability* (RAS)	Yes
Cluster *Power Policy Unit* (PPU)	No
Core PPU	No


Note

For accessibility information on the core registers, other than PPU registers, see your core *Technical Reference Manual* (TRM).

The following diagram shows an example of memory-mapped addressing for the cores to access the utility bus through the interconnect.

Figure 13-1: Memory-mapped access from the cores to the utility bus



13.1.2 Cluster and core PPU register access

The *Power Policy Unit* (PPU) registers for each core and cluster are still accessible when the cluster is powered off.

If a core is not powered on, then any access to a core register (not including the PPU registers) is treated as **RAZ/WI**. Similarly, if the cluster is powered off, then any access to a cluster register (not including the PPU registers) is treated as **RAZ/WI**.

Note

- The PPUs for the cluster and each of the cores are still accessible when the cluster is powered off.
- The PPU registers for a core are still accessible when that core is powered off.

13.2 Base addresses for system components

Each set of System registers is grouped on separate 64KB page boundaries allowing access to be enforced by a *Memory Management Unit* (MMU).

The following table shows the base addresses for each set of system component registers and their Security state.


Note

- The base address for each set of registers for the core Power Policy Units (PPUs) depends on the core instance number <n>, from 0 to CN.
- In the following table, any address space that is not documented is treated as **RAZ/WI**.
- For base addresses of core registers, which are mapped from `0x<n>90000` - `0x<n>F0000`, see your core *Technical Reference Manual* (TRM).
- The base addresses in the following table are the addresses accessed on the Utility bus interface. The system interconnect typically maps these addresses into a particular address range based on the system address map. Therefore, software has to add the base address listed here onto the system address range base to get the absolute physical address of a register.

Table 13-3: Utility bus base addresses for system component registers

Base address, n is core instance number	Registers	Security state	Memory map
`0x000000`	Cluster control	Secure	B.1.1 External cluster register summary on page 344
`0x010000`	Cluster MPAM	Secure and Non-secure	B.1.2 External MPAM register summary on page 367
`0x020000`	Cluster RAS	Secure	B.1.3 External cluster RAS register summary on page 385
`0x030000`	Cluster PPU	Secure	B.1.4 External cluster PPU register summary on page 431
`0x040000` - `0x070000`	Reserved for future cluster registers	-	-
`0x<n>80000`	Core <n> PPU	Secure	B.1.5 External core PPU register summary on page 486
`0x<n>90000` - `0x<n>F0000`	Core <n> registers	See your core TRM	See your core TRM

14. System control registers

The system control registers control and provide status information for the functions that the *DynamIQ™ Shared Unit-110* (DSU-110) implements. They can be accessed from the cores directly or externally through the utility bus.

14.1 AArch64 generic system control registers

The summary table provides an overview of all generic-system-control registers in the *DynamIQ™ Shared Unit-110* (DSU-110). Individual register descriptions provide detailed information.

- Any AArch64 generic system control registers that are not present in Direct connect are treated as **RAZ/WI**.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 14-1: Generic system control registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERCFR_EL1	3	0	C15	C3	0	–	64-bit	Cluster Configuration Register	Yes
IMP_CLUSTERIDR_EL1	3	0	C15	C3	1	–	64-bit	Cluster Main Revision Register	Yes
IMP_CLUSTERREVIDR_EL1	3	0	C15	C3	2	–	64-bit	Cluster ECO ID Register	Yes
IMP_CLUSTERACTLR_EL1	3	0	C15	C3	3	–	64-bit	Cluster Auxiliary Control Register	Yes
IMP_CLUSTERECTLR_EL1	3	0	C15	C3	4	–	64-bit	Cluster Extended Control Register	Yes
IMP_CLUSTERPWRCTLR_EL1	3	0	C15	C3	5	–	64-bit	Cluster Power Control Register	No
IMP_CLUSTERPWRDN_EL1	3	0	C15	C3	6	–	64-bit	Cluster Power Down Register	Yes
IMP_CLUSTERPWRSTAT_EL1	3	0	C15	C3	7	–	64-bit	Cluster Power Status Register	No
IMP_CLUSTERL3DNTH0_EL1	3	0	C15	C4	0	–	64-bit	Cluster L3 Downsize Threshold0 Register	No
IMP_CLUSTERL3DNTH1_EL1	3	0	C15	C4	1	–	64-bit	Cluster L3 Downsize Threshold1 Register	No
IMP_CLUSTERL3UPTH0_EL1	3	0	C15	C4	2	–	64-bit	Cluster L3 Upsize Threshold0 Register	No
IMP_CLUSTERL3UPTH1_EL1	3	0	C15	C4	3	–	64-bit	Cluster L3 Upsize Threshold1 Register	No
IMP_CLUSTERBUSQOS_EL1	3	0	C15	C4	4	–	64-bit	Cluster Bus QoS Control Register	No
IMP_CLUSTERL3HIT_EL1	3	0	C15	C4	5	–	64-bit	Cluster L3 Hit Counter Register	No
IMP_CLUSTERL3MISS_EL1	3	0	C15	C4	6	–	64-bit	Cluster L3 Miss Counter Register	No
IMP_CLUSTERPPSTART_EL1	3	0	C15	C9	0	–	64-bit	Cluster peripheral port Start Address Register	No
IMP_CLUSTERPPEND_EL1	3	0	C15	C9	1	–	64-bit	Cluster peripheral port End Address Register	No

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERCFR2_EL1	3	0	C15	C9	2	—	64-bit	Cluster Configuration Register 2	No
IMP_CLUSTERCDBG_EL3	3	6	C15	C4	7	—	64-bit	Cluster Cache Debug Register	No
IMP_CLUSTERPMMDCR_EL3	3	6	C15	C6	3	—	64-bit	Monitor Debug Configuration Register (EL3)	No

15. Debug

The DSU-110 DynamIQ™ cluster provides a debug system that supports both self-hosted and external debug. It has an external DebugBlock component, and integrates various CoreSight debug related components.

The CoreSight debug related components are split into two groups in the DSU-110. Some components are in the DynamIQ™ cluster itself, while some of the others are in the separate DebugBlock. The DebugBlock is deliberately separate from the cluster, to facilitate the following system design options:

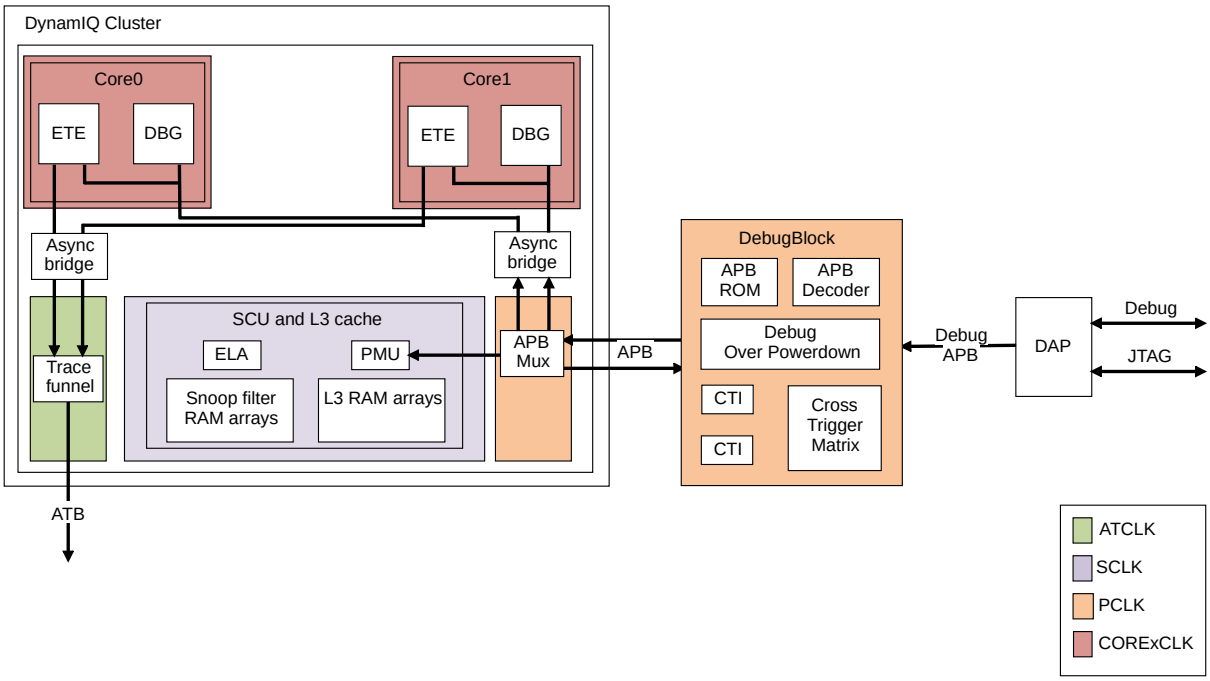
- The DebugBlock is placed in a separate power domain, to ensure that it is possible to maintain the connection to a debugger while the cores and cluster are powered down.
- The DebugBlock is physically placed with the other CoreSight logic in the SoC, rather than close to the cluster.

The connection between the cluster and the DebugBlock consists of a pair of *Advanced Peripheral Bus* (APB) interfaces, one in each direction. All debug traffic, except the authentication interface, takes place over this interface as read or write APB transactions. This debug traffic includes register reads, register writes, and CTI triggers. There are no other wires between these two components to ensure that this traffic can be routed over any standard APB interconnect or APB bridge.

The following figure shows how the DSU-110 implements the following CoreSight debug components:

- Per-core *Embedded Trace Extension* (ETE). Although the ETE is supplied with the core, the DSU-110 integrates this into the CoreSight subsystem.
- Per-core Cross Trigger Interface (CTI). These are contained in the DebugBlock.
- Cross Trigger Matrix (CTM)
- Debug over powerdown support
- APB Decoder
- APB ROM
- APB Mux

Figure 15-1: Cluster debug components



The primary debug APB interface on the DebugBlock, controls all the debug components and forms a standard CoreSight interface that is compatible with the previous generation of cores. The APB decoder decodes the requests on this bus before they are sent to the appropriate component in the DebugBlock or in the cluster. The per-core CTIs are connected to a CTM.

Each core contains a debug component that is accessed by the debug APB bus. The cores support debug over powerdown via modules in the DebugBlock that mirror key core information. These modules allow access to debug over powerdown CoreSight™ registers while the core is powered down.

The ETE in each core outputs trace, which is funneled in the cluster down to a single AMBA 4 ATBv1.1 interface, which is 32 bits wide in small clusters and 64 bits wide in larger clusters.

Cache debug

Cache debug of the DSU-110 cache RAMs is supported, which allows software to read the contents of the L3 cache and snoop filter. This cache debug is under the control of the core, in the same way that L1 or L2 cache debug is controlled. The core sends a read operation to the DSU-110 with the physical location to read, and the DSU-110 returns the RAM contents at that location. The core then exposes this information in a System register.

15.1 Cache debug

Cache debug of the DSU-110 cache RAMs is supported, which allows software to read the contents of the L3 cache and Snoop Filter (SF) RAMs. This cache debug is under control of the core, in the same way that the L1 or L2 cache debug is controlled. Access to the DSU-110 cache debug information is provided through the DSU-110 CLUSTERCDBG register.

The three step process of extracting information from the RAMs is as follows:

1. The core writes to the CLUSTERCDBG register, setting the bitfields for the physical location it wants to retrieve the data from. See the following table for bit field values.
2. The DSU-110 returns the RAM contents in the CLUSTERCDBG register in an encoded form.
3. The core reads this information from the CLUSTERCDBG register.

- The bit field descriptions for the CLUSTERCDBG register depend on if you are writing to the register or reading from the register, and when you are reading from the register what type of access is being made.
- The CLUSTERCDBG register is shared between cores, so to get predictable results software must ensure that only one core accesses the register at a time.
- The cache debug operations only read the cache contents when the cluster is in the ON power mode. If the cluster is in the FUNC_RET power mode, the contents of the CLUSTERCDBG register are **UNKNOWN**. Therefore, Arm recommends before starting any cache debug accesses that software sets the IMP_CLUSTERPWRCTLR_EL1.RETCTL value to zero.

The following table describes the bitfields for the CLUSTERCDBG register when writing to the register.

Table 15-1: CLUSTERCDBG bit descriptions when writing to the register

Bits	Name	Description
[63:32]	RAZ/WI	Reserved
[31:28]	WAY	Way of RAM being accessed. The number of SF ways can be obtained from the IMP_CLUSTERCFR_EL1 register. The number of L3 cache ways can be obtained from the CCSIDR_EL1 register.
[27:24]	RAZ/WI	Reserved

Bits	Name	Description
[23:6]	SLCID_IDX	The L3 cache Set locations in each cache slice are all power-of-2 in size and therefore can be identified using contiguous index locations. The Set index values for slice 0 start from value zero in this field, followed by the index locations for slice 1, and then sequentially up to the total number of cache slices. The total index width varies depending on the size of the RAM being accessed. The cache slice identification number, slice ID, forms the upper used bits of the cache location encoding in this field. For details on tag index widths, see Table 15-3: Tag index width for L3 RAM accesses on page 174. For details on slice ID widths see Table 15-2: Slice ID width on page 173. As the SF RAM sizes are, typically, different from the L3 RAM sizes, the precise encodings of this field will be different when accessing SF RAM locations compared with accessing L3 cache tag and data RAM locations. For details on the SF index widths, see Table 15-4: SF index width for SF RAM accesses on page 174.
[5:3]	CHUNK	Select of 64-bit data chunk to read from 512-bit Data RAM cache line. Only used when accessing Data RAM data. **0b000** Data[63:0] **0b001** Data[127:64] **0b010** Data[191:128] **0b011** Data[255:192] **0b100** Data[319:256] **0b101** Data[383:320] **0b110** Data[447:384] **0b111** Data[511:448]
[2:0]	RAM	RAM to be accessed. All other values are reserved. **0b001** Snoop Filter RAM **0b010** Tag RAM **0b011** Data RAM - accessing cacheline data **0b111** Data RAM - accessing cacheline *Memory Tagging Extension* (MTE) tags

The following table shows how to determine the slice ID width from the number of cache slices configured.

Table 15-2: Slice ID width

Number of cache slices	Slice ID width
1	0

Number of cache slices	Slice ID width
2	1
4	2
8	3

For L3 RAM accesses, the following table shows how to determine the tag index width from the cache size per slice.

Table 15-3: Tag index width for L3 RAM accesses

Cache size per slice	Tag index width
256KB	8
384KB-512KB	9
768KB-1024KB	10
1536KB-2048KB	11

For SF RAM accesses, the following table shows how to determine the snoop filter index widths from the cache size per slice.


Note

In the following table, the snoop filter RAM address width for the 1536KB and 2048KB sizes is only 11-bits wide, but there are two banks of RAMs, so the effective width is 12-bits. For these configurations, the *Least Significant Bit* (LSB) of the SLCID_IDX field selects which bank to access.

Table 15-4: SF index width for SF RAM accesses

SF size per slice	SF index width
128KB, 192KB	9
256KB, 384KB	10
512KB, 768KB, 1024KB	11
1536KB, 2048KB	12

The following table describes how to interpret RAM data read back from the DSU-110 CLUSTERCDBG register, for a snoop filter access.

Table 15-5: CLUSTERCDBG bit descriptions for a snoop filter RAM access

Bits	Width (bits)	Description
[63:MAX_CMPXS+40]	24 - MAX_CMPXS	RAZ
[MAX_CMPXS+39:40]	MAX_CMPXS	One bit per standalone core or per complex. When a bit is 1 it identifies a core or complex where the cache line is allocated. When a bit is 0, it indicates the cache line is not allocated in this core or complex.

Bits	Width (bits)	Description
[39:38]	2	This field has the following values: **0b00** Cache line is invalid **0b01** Cluster received a shared copy of the cache line. The cores know it is shared. **0b10** Cluster received a unique copy of the cache line and has given a unique copy to a core or complex (the core thinks it is unique). **0b11** Cluster received a unique copy of the cache line and has given a shared copy to one or more complexes (the core thinks it is shared).
[37:26]	12	RAZ
[25]	1	NS (Non-secure). This bit has the following values: **0** The cache line is Secure **1** The cache line is Non-secure
[24:0]	25	Physical address tag. The encoding of these bits depend on IMP_CLUSTERCFR_EL1.SFIDX as follows: **0x9** `{PA[39:15}` **0xA** `{PA[39:16},1'b0}` **0xB** `{PA[39:17},2'b00}` **0xC** `{PA[39:18},3'b000}`

The following table describes how to interpret RAM data read back from the DSU-110 CLUSTERCDBG register, for a tag RAM access.

Table 15-6: CLUSTERCDBG bit descriptions for a tag RAM access

Bits	Width (bits)	Description
[63:58]	6	RAZ
[57]	1	*Memory System Resource Partitioning and Monitoring* (MPAM) - PMG bit. If MPAM values are stored in the cache, then this bit saves the MPAM PMG value.
[56:51]	6	MPAM - PartID. If MPAM values are stored in the cache, then these bits save the MPAM PARTID value.
[50]	1	MPAM - NS. If MPAM values are stored in the L3 cache, this bit indicates if it is a Non-secure state PART ID or Secure state PART ID.

Bits	Width (bits)	Description
[49:46]	4	*Page-Based Hardware Attribute* (PBHA). If the PBHA bits are stored in the cache, then these bits report the PBHA values for this cache line.
[45:44]	2	MTE state. If MTE values are stored in the cache, then these bits save the MTE values for this line. The possible values are: **0b00** MTE tag is Invalid **0b01** MTE tag is Clean **0b10** MTE tag is Dirty (the tag value for the cache line has been modified using an instruction such as `STG`.
[43]	1	OA (Outer Allocation). The possible values are: **0** This hints that the system should not allocate the cache line. **1** This hints that the system should allocate the cache line.
[42]	1	PF (PreFetch). The possible values are: **0** This hints that the cache line is not considered to be a prefetch (the cache line has been used). **1** This hints that the cache line is an unused prefetch.
[41]	1	CP (CPU Presence). The possible values are: **0** Indicates that there is no snoop filter entry for this cache line. **1** Indicates a snoop filter entry for this cache line.

Bits	Width (bits)	Description
[40:39]	2	Tag RAM State. The possible values are: **0b00** Cache line entry Invalid **0b01** UniqueDirty. The cluster has a unique, modified (dirty) copy of the cache line. **0b10** SharedClean. The cluster has a shared copy of the cache line that is coherent with the external memory location. **0b11** UniqueClean. The cluster has a unique copy of the cache line.
[38:27]	12	RAZ
[26]	12	NS (Non-secure). The possible values are: **0** The cache line is Secure. **0** The cache line is Non-secure.
[25:0]	26	Physical Address Tag Encoding varies depending on the number of L3 sets divided by the number of L3 cache slices. The number of L3 sets can be found by writing `0x4` to CSSELR_EL1, and then calculating CCSIDR_EL1.NumSets + 1. The number of L3 cache slices can be found from the IMP_CLUSTERCFR_EL1.L3SLC. For (L3 sets/L3 cache slices), the possible values are: **256** {PA[39:14]} **512** {PA[39:15],1'b0} **1024** {PA[39:16],2'b00} **2048** {PA[39:17],3'b000} Where PA is the Physical Address width.

The following table describes how to interpret the data read back from the DSU-110 CLUSTERCDBG register, for a Data RAM data access.

Table 15-7: CLUSTERCDBG bit descriptions for a Data RAM data access

Bits	Width (bits)	Description
[63:0]	64	Cache data from selected cache location and Chunk of data

The following table describes how to interpret the data read back from the DSU-110 CLUSTERCDBG register, for a Data RAM tag value access.

Table 15-8: CLUSTERCDBG bit descriptions for a Data RAM MTE tag value access

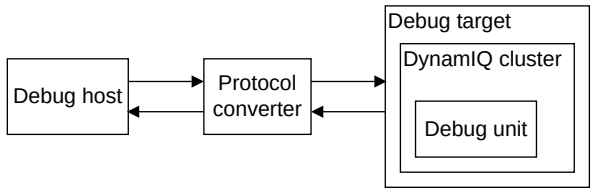
Bits	Width (bits)	Description
[63:16]	-	RAZ
[15:12]	4	MTE tag for selected cache line bits [511:384]
[11:8]	4	MTE tag for selected cache line bits [383:256]
[7:4]	4	MTE tag for selected cache line bits [255:128]
[3:0]	4	MTE tag for selected cache line bits [127:0]

15.2 Supported debug methods

The DSU-110 DynamIQ™ cluster along with its associated complexes and cores is part of a debug system that supports both self-hosted and external debug.

The following figure shows a typical external debug system.

Figure 15-2: External debug system



Debug host

A computer, for example a personal computer, that is running a software debugger such as the Arm Debugger. With the debug host, you can issue high-level commands, such as setting a breakpoint at a certain location or examining the contents of a memory address.

Protocol converter

The debug host sends messages to the debug target using an interface such as Ethernet. However, the debug target typically implements a different interface protocol. A device such as DSTREAM is required to convert between the two protocols.

Debug target

The lowest level of the system implements system support for the protocol converter to access the debug unit. For *DynamIQ™ Shared Unit-110* (DSU-110) based devices, the mechanism used to access the debug unit is based on the CoreSight architecture. The DSU-110 itself is accessed using an APB slave interface. An example of a debug target is a development system with a test chip or a silicon part with a DSU-110.

Debug unit

Helps debugging software that is running on the core:

- DSU-110 and external hardware based around the core.
- Operating systems.
- Application software.

With the debug unit, you can:

- Stop program execution.
- Examine and alter process and coprocessor state.
- Examine and alter memory and the state of the input or output peripherals.
- Restart the PE.

For self-hosted debug, the debug target runs debug monitor software that runs on the core in the cluster. This way, it does not require expensive interface hardware to connect a second host computer.

15.3 Terminology

The DSU-110 DynamIQ™ cluster debug system supports both single and multi-threaded cores.

The Arm architecture allows for cores to be single, or multi-threaded. A *Processing Element* (PE) performs a thread of execution. A single-threaded core has one PE and a multi-threaded core has two or more PEs. Because the debugging system allows individual threads to be debugged, the term PE is used throughout this chapter. Where a reference to a core is made, the core can be a single, or multi-threaded core.

Related information

2.7 Core, complex, and processing element numbering on page 30

15.4 Simplified PE and Debug power domains

The DSU-110 DynamIQ™ cluster debug system implements a simplified programmers' model to reduce the complexity of the PE and Debug power domains.

This simplified programmers' model implements the following changes to the debug system:

- Power control is removed from the external debug and *Embedded Trace Extension* (ETE) registers and replaced by a separate *CoreSight Granular Power Requestor* (GPR) component. The GPR allows a system designer to define a hierarchy of power domains and map components into power domains.
- The PE debug and ETE components return errors for all registers when powered-off. These registers include the EDPRSR (External Debug Processor Status Register) and the Peripheral ID registers, which means that:
  - Register decode is at page-sized granularity.
  - To identify the implementation and variant, the target must be powered-on.
- The "Halting step over powerdown" feature is removed. The debugger can set the Reset Catch to achieve this result.

The No-power Down control in the DBGPRCR_EL1 System register remains unchanged. Software can use this control to request Powerdown emulation.

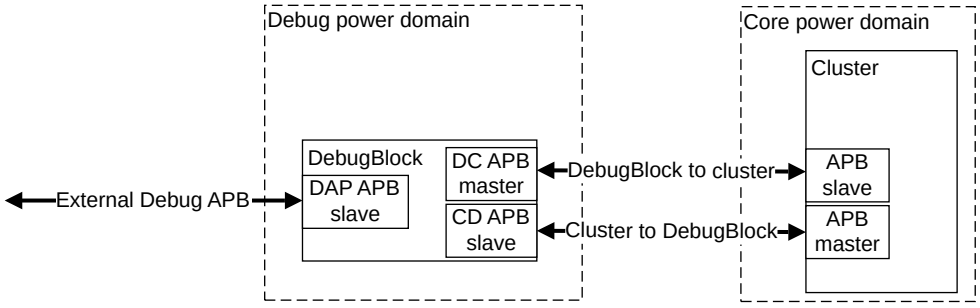
15.5 DebugBlock overview

The DebugBlock combines the functions, registers, and interfaces that are required for debug over powerdown.

The DebugBlock is provided as a separate component to allow implementation in a separate power domain from the cluster. Having a separate debug power domain allows the connection to a debugger be maintained while the cores, complexes, and cluster are powered down. The DynamIQ™ Shared Unit-110 (DSU-110) also allows powering down the DebugBlock when debug is not in process.

The following diagram shows how the DebugBlock is connected to the cluster.

Figure 15-3: Debug APB connections



The DebugBlock has the following APB interfaces:

External Debug APB (DAP APB)

An APB slave interface, allowing communication with an external debugger, for example through a CoreSight *Debug Access Port* (DAP).

All debug register read and write requests from an external debugger are received on this bus.

DebugBlock to cluster (DC APB)

An APB master interface that is connected to the cluster. It sends all debug register read and write requests to the cluster.

CTI output trigger events are sent to the cluster as trigger requests on this bus.

Cluster to DebugBlock (CD APB)

An APB slave interface that is connected to the cluster. It receives CTI input trigger event requests from the cluster.

Debug register reads and writes

The DebugBlock holds all the debug registers that are implemented in the Debug power domain. Registers implemented in the Debug power domain are specified in the *Arm®v9.0-A Architecture Reference Manual Armv9, for Armv9-A architecture profile*.

Accesses through the DAP APB interface to Debug domain registers are handled internally by the DebugBlock.

Accesses through the DAP APB interface to core power domain registers are passed on to the cluster through the DC APB interface.

CTI trigger events

Trigger events are transferred between the DebugBlock and cluster through the CD APB and DC APB interfaces.

Input trigger events

Input trigger events are sent from the cluster to the CTIs through the CD APB as write transactions.

Output trigger events

Output trigger events are sent from the CTIs to the cluster through the DC APB as write transactions.

DebugBlock power states

The DebugBlock supports two power modes: ON and OFF. These power modes are controlled using the power Q-Channel interface. When the DebugBlock is in the OFF power mode, any uncompleted transactions on the external Debug APB interface to complete with an SLVERR.

Related information

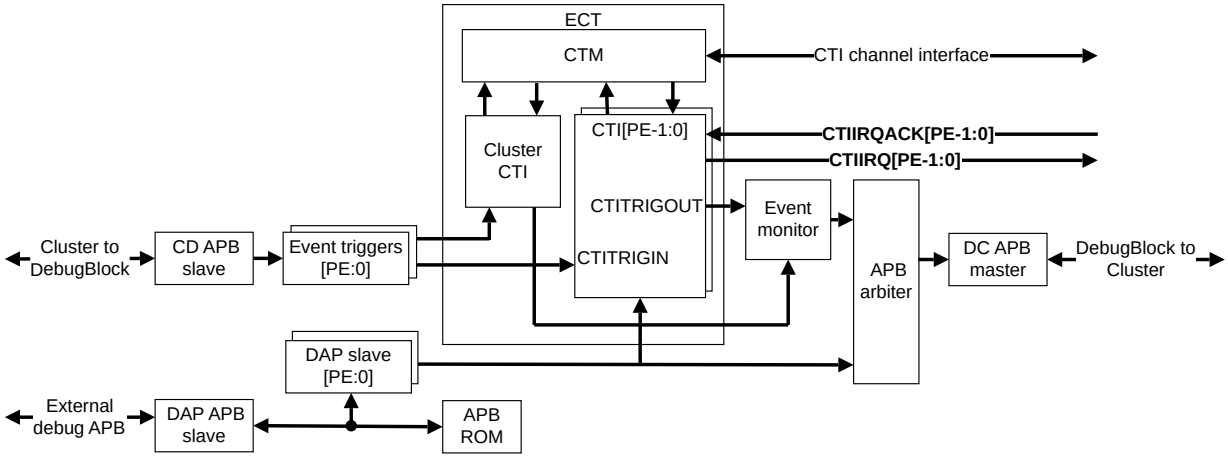
15. Debug on page 170
15.6 DebugBlock subcomponents on page 181
15.7 Embedded Cross Trigger overview on page 183

15.6 DebugBlock subcomponents

The DebugBlock component consists of various subcomponents that facilitate the debugging of the DSU-110 DynamIQ™ cluster while the cores, complexes, and cluster are powered down.

The following figure shows the DebugBlock.

Figure 15-4: DebugBlock block diagram



Note

The CTIs shown in the diagram include both the CTIs attached to each of the *Processing Elements* (PEs) [0:PE-1] and the cluster CTI.

ECT

The DebugBlock implements the *Embedded Cross Trigger* (ECT).

APB ROM

The APB ROM table holds the address decoding for each debug component in the DebugBlock and the cluster. The APB ROM table complies with the *Arm® CoreSight™ Architecture Specification v3.0*. The ROM table is hierarchical, with further ROM tables in the cluster and cores. See 16. ROM tables on page 190 for more information on ROM tables.

Event monitor

The event monitor converts changes in CTI output triggers to APB write transactions.

Event triggers

The event triggers convert APB write transactions to CTI input triggers.

APB arbiter

The DC APB transfers both register accesses and CTI output trigger events. The APB arbiter multiplexes the two sources of transactions.

DAP slave

The DAP slave holds copies of registers in the debug power domain.

Related information

15.7 Embedded Cross Trigger overview on page 183

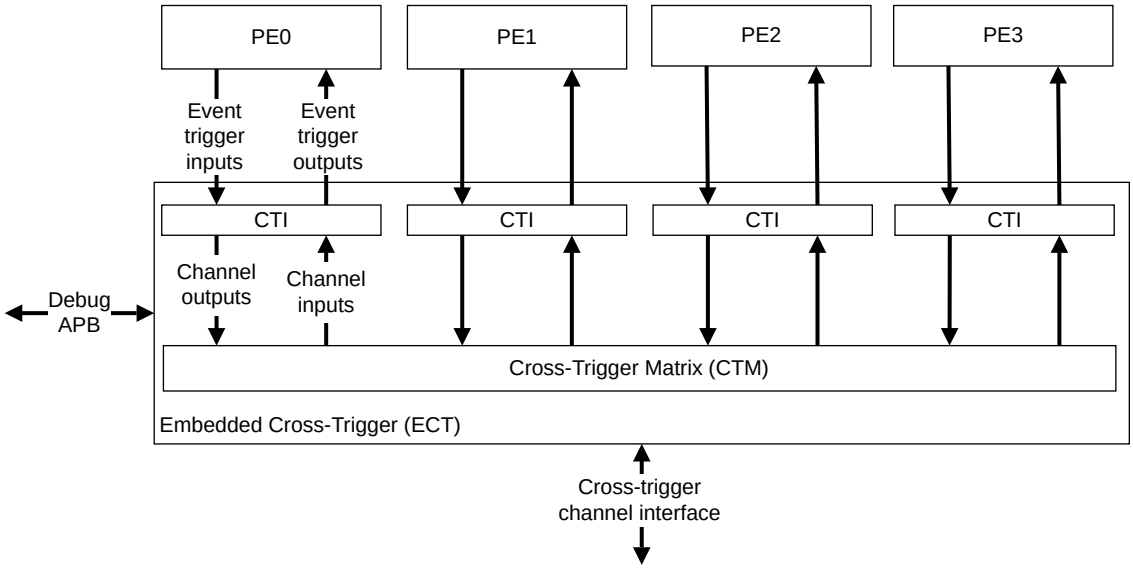
15.7 Embedded Cross Trigger overview

The *Embedded Cross Trigger* (ECT) allows debug events to be sent between *Processing Elements* (PEs).

The ECT provides a *Cross Trigger Interface* (CTI) for each PE in the cluster. There is also a cluster CTI, which is present in all configurations except Direct connect. The CTIs are interconnected through a *Cross Trigger Matrix* (CTM) to send debug and trace events between PEs.

The following diagram shows a conceptual view of the trigger event inputs and outputs between the PEs and the ECT.

Figure 15-5: Embedded Cross Trigger concept



The CTIs selectively send trigger events to the CTM on their respective channel outputs. The CTIs receive trigger events from the CTM on their channel inputs.

Trigger events are transferred between CTIs over the channel interface. The CTM connects the channel interface to the channel inputs and channel outputs of the CTIs.

External interfaces

The external cross-trigger channel interface, from the CTM, allows cross-triggering between SoC external devices.

The Debug APB provides access to the CTI registers to allow an external debugger to configure the trigger event routing, and send events to PEs. For example, an external debugger might use this mechanism to put a PE into Debug state.

CTI registers

Registers in the CTI perform the following functions:

- Control the mapping of the input trigger events to channel outputs.
- Control the mapping of the channel inputs to output trigger events.
- Capture the state of input and output trigger events.
- Set, clear, or pulse output trigger events.

Related information

15.7.1 CTI triggers on page 184
15.6 DebugBlock subcomponents on page 181

15.7.1 CTI triggers

The *Cross Trigger Interfaces* (CTIs) each have input and output trigger events that are mapped onto the debug and trace events in the *Processing Elements* (PEs) and *Embedded Logic Analyzers* (ELAs). All PEs in the cluster have the same mapping.

CTI input triggers from each PE

The following table shows how events are mapped onto the CTI input triggers.

Table 15-9: Allocation of input debug and trace trigger events from the PE to the CTI

Trigger number	Trigger event name	Source	Destination	Type	Description
0	Cross-halt	PE	CTI	Pulse	This trigger event is sent when the PE enters Debug state.
1	Performance monitors overflow	PE	CTI	Pulse	This trigger event is sent when a PMU counter overflows.
2	Profiling sample	PE	CTI	Pulse	This trigger event is sent when a profiling sample is written out.
3	Reserved	-	-	-	Reserved
4-7	ETE trace external output	ETE	CTI	Pulse	This trigger event is sent from the ETE trace in the PE to the CTI.
8-9	ELA output	ELA	CTI	Pulse	This trigger event is sent from the ELA **CTTRIGOUT[1:0]** attached to the PE.

CTI output triggers from each PE

The following table shows how events are mapped onto CTI output triggers.

Table 15-10: Allocation of output debug and trace trigger events from the CTI to the PE

Trigger number	Trigger event name	Source	Destination	Type	Description
0	Debug request	CTI	PE	Level	Request the PE to enter Debug state.

Trigger number	Trigger event name	Source	Destination	Type	Description
1	Restart request	CTI	PE	Pulse	Request the PE to exit Debug state.
2	Generic CTI interrupt	CTI	GIC	Pulse	This trigger event must be sent to the *Generic Interrupt Controller* (GIC) for the PE.
3	Reserved	-	-	-	Reserved
4-7	ETE trace external input	CTI	ETE	Pulse	This trigger event is sent to the *Embedded Trace Extension* (ETE) trace in the PE.
8-9	ELA input	CTI	ELA	Pulse	This trigger event is sent to the ELA **CTTRIGIN[1:0]** attached to the PE.

Allocation of cluster CTI trigger inputs

The following table shows how events are mapped onto the cluster CTI input triggers.

Table 15-11: Allocation of input trigger events from the cluster ELA and PMU to the cluster CTI

Trigger number	Trigger event name	Source	Destination	Type	Description
0	Reserved	-	-	-	Reserved
1	Cluster PMU output	Cluster PMU	Cluster CTI	Pulse	CTI output trigger events that are mapped onto the trigger events in the cluster PMU.
2-7	Reserved	-	-	-	Reserved
8-9	Cluster ELA output	Cluster ELA	Cluster CTI	Pulse	CTI output trigger events that are mapped onto the trigger events in the cluster ELA **CTTRIGOUT[1:0]**.

Allocation of cluster CTI trigger outputs

The following table shows how events are mapped onto the cluster CTI output triggers.

Table 15-12: Allocation of output trigger events from the cluster CTI to the cluster ELA

Trigger number	Trigger event name	Source	Destination	Type	Description
0-1	Reserved	-	-	-	Reserved
2	CTIIRQ	-	-	Pulse	This trigger event must be sent to the *Generic Interrupt Controller* (GIC).
3-7	Reserved	-	-	-	Reserved
8-9	Cluster ELA input	Cluster CTI	Cluster ELA	Pulse	CTI output trigger events that are mapped onto the trigger events in the cluster ELA **CTTRIGIN[1:0]**.

Related information

15.7 Embedded Cross Trigger overview on page 183

15.8 External CTI registers

The summary table provides an overview of all the *Cross Trigger Interface* (CTI) registers that are accessed externally (memory-mapped) over the Debug APB interface. Individual register descriptions provide detailed information.

The summary table provides an overview of all the cluster CTI registers and core CTI registers. For more information about a register, click on the register name in the table.

- Registers that differ in descriptions and values, for cluster and core, are indicated in the Identical CTI core column. These registers are the CTIPIDR0-4 registers, and the CTIDEVAFF0-1 registers.
- The cluster CTI registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect all these registers are present.
- The cluster CTI part number is `0x4EA`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 15-13: CTI registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core CTI
0x000	CTICONTROL	–	32-bit	CTI Control register	Yes	Yes
0x010	CTIINTACK	–	32-bit	CTI Output Trigger Acknowledge register	Yes	Yes
0x014	CTIAPPSET	–	32-bit	CTI Application Trigger Set register	Yes	Yes
0x018	CTIAPPCLEAR	–	32-bit	CTI Application Trigger Clear register	Yes	Yes
0x01C	CTIAPPPULSE	–	32-bit	CTI Application Pulse register	Yes	Yes
0x20	CTIINEN0	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x24	CTIINEN1	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x28	CTIINEN2	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x2C	CTIINEN3	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x30	CTIINEN4	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x34	CTIINEN5	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x38	CTIINEN6	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x3C	CTIINEN7	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core CTI
0x40	CTIINEN8	—	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x44	CTIINEN9	—	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0xA0	CTIOUTEN0	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xA4	CTIOUTEN1	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xA8	CTIOUTEN2	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xAC	CTIOUTEN3	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xB0	CTIOUTEN4	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xB4	CTIOUTEN5	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xB8	CTIOUTEN6	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xBC	CTIOUTEN7	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xC0	CTIOUTEN8	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xC4	CTIOUTEN9	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0x130	CTITRIGINSTATUS	—	32-bit	CTI Trigger In Status register	Yes	Yes
0x134	CTITRIGOUTSTATUS	—	32-bit	CTI Trigger Out Status register	Yes	Yes
0x138	CTICHINSTATUS	—	32-bit	CTI Channel In Status register	Yes	Yes
0x13C	CTICHOUTSTATUS	—	32-bit	CTI Channel Out Status register	Yes	Yes
0x140	CTIGATE	—	32-bit	CTI Channel Gate Enable register	Yes	Yes
0x150	CTIDEVCTL	—	32-bit	CTI Device Control register	Yes	Yes
0xFA0	CTICLAIMSET	—	32-bit	CTI Claim Tag Set register	Yes	Yes
0xFA4	CTICLAIMCLR	—	32-bit	CTI Claim Tag Clear register	Yes	Yes
0xFA8	CTIDEVAFF0	—	32-bit	CTI Device Affinity register 0	Yes	No, see individual register
0xFAC	CTIDEVAFF1	—	32-bit	CTI Device Affinity register 1	Yes	No, see individual register
0xFB8	CTIAUTHSTATUS	—	32-bit	CTI Authentication Status register	Yes	Yes
0xFBC	CTIDEVARCH	—	32-bit	CTI Device Architecture register	Yes	Yes
0xFC0	CTIDEVID2	—	32-bit	CTI Device ID register 2	Yes	Yes
0xFC4	CTIDEVID1	—	32-bit	CTI Device ID register 1	Yes	Yes
0xFC8	CTIDEVID	—	32-bit	CTI Device ID register 0	Yes	Yes
0xFCC	CTIDEVTYPE	—	32-bit	CTI Device Type register	Yes	Yes
0xFD0	CTIPIDR4	—	32-bit	CTI Peripheral Identification Register 4	Yes	No, see individual register

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core CTI
0xFE0	CTIPIDR0	—	32-bit	CTI Peripheral Identification Register 0	Yes	No, see individual register
0xFE4	CTIPIDR1	—	32-bit	CTI Peripheral Identification Register 1	Yes	No, see individual register
0xFE8	CTIPIDR2	—	32-bit	CTI Peripheral Identification Register 2	Yes	No, see individual register
0xFEC	CTIPIDR3	—	32-bit	CTI Peripheral Identification Register 3	Yes	No, see individual register
0xFF0	CTICIDR0	—	32-bit	CTI Component Identification Register 0	Yes	Yes
0xFF4	CTICIDR1	—	32-bit	CTI Component Identification Register 1	Yes	Yes
0xFF8	CTICIDR2	—	32-bit	CTI Component Identification Register 2	Yes	Yes
0xFFC	CTICIDR3	—	32-bit	CTI Component Identification Register 3	Yes	Yes

15.9 Trace output from cores and DynamIQ cluster

Each core in the cluster includes an *Embedded Trace Extension* (ETE) that generates trace. The trace from all the cores is funneled in the cluster down to a single AMBA 4 ATBv1.1 interface, which is 32-bits wide in small clusters and 64-bits wide in larger clusters.


Note

Optionally, the cores and cluster can also include instances of the ELA-600, if this IP has been licensed.

The ELA-600 instances are always configured to generate *Advanced Trace Bus* (ATB) trace. The trace from the *Embedded Logic Analyzer* (ELA) instances is funneled to the same ATB trace interface as the ETE trace.

15.10 CoreSight component identification

The following table lists the CoreSight ID values for the components present within the DSU-110.

For details of the CoreSight ID scheme, see the *Arm® CoreSight™ Architecture Specification v3.0*.

Table 15-14: CoreSight component identification

Component	PID	CID	DevType	DevArch	Revision
DebugBlock ROM table	0x04006BB4E7	0xB105900D	0x00000000	0x47700AF7	r4p0
Cluster ROM	0x04006BB4E8	0xB105900D	0x00000000	0x47711A14	r4p0
Cluster CTI	0x04006BB4E8	0xB105900D	0x14	0x47700AF7	r4p0
Cluster PMU	0x04006BB4E8	0xB105900D	0x00000016	0x47702A16	r4p0

For details on the CoreSight component identification for the cluster ELA, see the *Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual*.

16. ROM tables

The ROM tables hold the locations of debug components, which debuggers can use to determine which components are implemented. The *DynamIQ™ Shared Unit-110* (DSU-110) has three different types of ROM tables. There is a ROM table for DebugBlock components, a ROM table for the cluster components, and a ROM table for each standalone core or complex.

All the ROM tables comply with the *Arm® CoreSight™ Architecture Specification v3.0*. The ROM tables for the DSU-110 contain locations for debug components, locations of some control and identification registers, and entry points for any sub-level ROM tables. For example, the cluster ROM table contains entry points for the ROM tables belonging to each core or complex in the cluster.

The debug components in the DSU-110 include components for each *Processing Element* (PE) in the cluster, for example a *Cross Trigger Inteface* (CTI) for each PE in the cluster.


Note

For a cluster comprised of complexes or cores which have a single PE per core, the PE numbering follows the core instance numbering, see 2.7 Core, complex, and processing element numbering on page 30.

If a component is not included in your implementation, the corresponding ROM table entry indicates that the component is not present.

The following table lists the types of debug components that can be accessed for each ROM table in the DSU-110.

Table 16-1: Types of components listed in the ROM tables for the DSU-110

ROM table	ROM table located in	Components
DebugBlock	DebugBlock	• Cluster CTI • CTI for each PE • Power control and status registers for the cluster • Peripheral and component identification registers • ROM table entry point for the Cluster ROM table
Cluster	DebugBlock	• Cluster *Performance Monitoring Unit* (PMU) • Cluster *Embedded Logic Analyzer* (ELA) • ROM table entry points for each standalone core or complex • Power control and status registers for each standalone core or complex in the cluster • Peripheral and component identification registers
Standalone core	Core	See the *Technical Reference Manual* (TRM) for your core.
Complex	Complex	See the TRM for your core.

16.1 Debug system address map

The debug system address map for the *DynamIQ™ Shared Unit-110* (DSU-110) cluster depends on the specific implementation of your cluster, for example the number of cores configured in the cluster. It also depends on the `DEBUG_NEW_ADDR_MAP` configuration parameter at build-time configuration.

DEBUG_NEW_ADDR_MAP

At build time configuration you can configure the DSU-110 to use one of two encodings for the debug system address map:

- The new debug system address map uses a more efficient encoding and reduces the amount of address space required. Arm recommends this for new designs.
- The older encoding allows backwards compatibility with systems already designed for this address map.

For both the new and older address maps, the following describes when certain entries are present:

Core <n> ROM tables:

Where n is the core instance number. These entries point to the ROM tables for a core or a complex. A complex only contains a single ROM table and so a ROM table will not be present for cores that form the second core of a dual core complex. The single ROM table that is included in a dual core complex contains the addresses for both of the cores in the dual core complex.


Note

The addresses for any component in a core instance, for example *Performance Monitoring Unit* (PMU) and *Embedded Trace Extension* (ETE), are the same irrespective of whether the core instance is a standalone core, a single core complex, or part of a dual core complex. However, the ROM table hierarchy that is used to identify the address values differ depending on the configuration.

Cluster ELA

These entries are only present if the *Embedded Logic Analyzer* (ELA) is included in the cluster.

Core ELAs

These entries are only on present if the ELA is included in the core or complex. When an ELA is included in a dual core complex, there is only one ELA present. The ELA is located after the *Embedded Trace Extension* (ETE) for the first core of the complex.

Components

Components are only present for cores that are included in the cluster.

New debug APB system address map

The following table shows the new debug system address map for the DSU-110 DynamIQ™ cluster.

Table 16-2: New debug APB system address map

Debug component (if present)	Debug APB address offset
DebugBlock ROM Table	0x0
Cluster ROM Table	0xC0000
Cluster PMU	0xD0000
Cluster ELA	0xE0000
Cluster CTI	0xF0000
Complex or core 0 ROM Table	0x100000
Core 0 Debug	0x110000
Core 0 PMU	0x120000
Core 0 ETE	0x130000
Core 0 ELA	0x140000
Core 0 CTI	0x1E0000
Complex or core 1 ROM Table	0x200000
Core 1 Debug	0x210000
Core 1 PMU	0x220000
Core 1 ETE	0x230000
Core 1 ELA	0x240000
Core 1 CTI	0x2E0000
Complex or core 2 ROM Table	0x300000
Core 2 Debug	0x310000
Core 2 PMU	0x320000
Core 2 ETE	0x330000
Core 2 ELA	0x340000
Core 2 CTI	0x3E0000
Complex or core 3 ROM Table	0x400000
Core 3 Debug	0x410000
Core 3 PMU	0x420000
Core 3 ETE	0x430000
Core 3 ELA	0x440000
Core 3 CTI	0x4E0000
Complex or core 4 ROM Table	0x500000
Core 4 Debug	0x510000
Core 4 PMU	0x520000
Core 4 ETE	0x530000
Core 4 ELA	0x540000
Core 4 CTI	0x5E0000
Complex or core 5 ROM Table	0x600000

Debug component (if present)	Debug APB address offset
Core 5 Debug	0x610000
Core 5 PMU	0x620000
Core 5 ETE	0x630000
Core 5 ELA	0x640000
Core 5 CTI	0x6E0000
Complex or core 6 ROM Table	0x700000
Core 6 Debug	0x710000
Core 6 PMU	0x720000
Core 6 ETE	0x730000
Core 6 ELA	0x740000
Core 6 CTI	0x7E0000
Complex or core 7 ROM Table	0x800000
Core 7 Debug	0x810000
Core 7 PMU	0x820000
Core 7 ETE	0x830000
Core 7 ELA	0x840000
Core 7 CTI	0x8E0000
Complex or core 8 ROM Table	0x900000
Core 8 Debug	0x910000
Core 8 PMU	0x920000
Core 8 ETE	0x930000
Core 8 ELA	0x940000
Core 8 CTI	0x9E0000
Complex or core 9 ROM Table	0xA00000
Core 9 Debug	0xA10000
Core 9 PMU	0xA20000
Core 9 ETE	0xA30000
Core 9 ELA	0xA40000
Core 9 CTI	0xAE0000
Complex or core 10 ROM Table	0xB00000
Core 10 Debug	0xB10000
Core 10 PMU	0xB20000
Core 10 ETE	0xB30000
Core 10 ELA	0xB40000
Core 10 CTI	0xBE0000
Complex or core 11 ROM Table	0xC00000
Core 11 Debug	0xC10000
Core 11 PMU	0xC20000
Core 11 ETE	0xC30000
Core 11 ELA	0xC40000

Debug component (if present)	Debug APB address offset
Core 11 CTI	0xCE0000

Older debug APB system address map

The following table shows the older debug APB system address map for the DSU-110 DynamIQ™ cluster.

Table 16-3: Older debug APB system address map

Debug component (if present)	Debug APB address offset
DebugBlock ROM Table	0x0
Core 0 CTI	0x10000
Core 1 CTI	0x20000
Core 2 CTI	0x30000
Core 3 CTI	0x40000
Core 4 CTI	0x50000
Core 5 CTI	0x60000
Core 6 CTI	0x70000
Core 7 CTI	0x80000
Cluster ROM Table	0x200000
Cluster PMU	0x210000
Cluster ELA	0x220000
Cluster CTI	0x230000
Complex or core 0 ROM Table	0x1000000
Core 0 Debug	0x1010000
Core 0 PMU	0x1020000
Core 0 ETE	0x1030000
Core 0 ELA	0x1040000
Complex or core 1 ROM Table	0x1100000
Core 1 Debug	0x1110000
Core 1 PMU	0x1120000
Core 1 ETE	0x1130000
Core 1 ELA	0x1140000
Complex or core 2 ROM Table	0x1200000
Core 2 Debug	0x1210000
Core 2 PMU	0x1220000
Core 2 ETE	0x1230000
Core 2 ELA	0x1240000
Complex or core 3 ROM Table	0x1300000
Core 3 Debug	0x1310000
Core 3 PMU	0x1320000
Core 3 ETE	0x1330000
Core 3 ELA	0x1340000

Debug component (if present)	Debug APB address offset
Complex or core 4 ROM Table	0x1400000
Core 4 Debug	0x1410000
Core 4 PMU	0x1420000
Core 4 ETE	0x1430000
Core 4 ELA	0x1440000
Complex or core 5 ROM Table	0x1500000
Core 5 Debug	0x1510000
Core 5 PMU	0x1520000
Core 5 ETE	0x1530000
Core 5 ELA	0x1540000
Complex or core 6 ROM Table	0x1600000
Core 6 Debug	0x1610000
Core 6 PMU	0x1620000
Core 6 ETE	0x1630000
Core 6 ELA	0x1640000
Complex or core 7 ROM Table	0x1700000
Core 7 Debug	0x1710000
Core 7 PMU	0x1720000
Core 7 ETE	0x1730000
Core 7 ELA	0x1740000

16.2 DebugBlock ROM table

The DebugBlock ROM table contents depend on how you configured your cluster.

The following table lists the entries for the DebugBlock ROM table, together with associated offsets from the physical base address of the ROM table. The DebugBlock ROM table includes:

- All the debug components for DebugBlock including the *Cross Trigger Interfaces* (CTIs) for each *Processing Element* (PE)
- Entry point for the cluster ROM table
- Power control register to allow a cluster powerup request, see 16.4 ROM table power request registers for cluster and cores on page 198.

The ROMENTRY entry values depend on the number and type of cores implemented. The register formats are described in the *Arm® CoreSight™ Architecture Specification v3.0*.


Note

The DebugBlock ROM table part number is 0x4E7.

Table 16-4: DebugBlock ROM table

Offset	Name	Description
0x0000	ROMENTRY0	Cluster ROM table entry point
0x0004	ROMENTRY1	Cluster CTI
0x0008	ROMENTRY2	CTI for PE 0
0x000C	ROMENTRY3	CTI for PE 1
0x0010	ROMENTRY4	CTI for PE 2
0x0014	ROMENTRY5	CTI for PE 3
0x0018	ROMENTRY6	CTI for PE 4
0x001C	ROMENTRY7	CTI for PE 5
0x0020	ROMENTRY8	CTI for PE 6
0x0024	ROMENTRY9	CTI for PE 7
0x0028	-	Reserved
0x0A00	DBGPCR0	Debug Power control register
0x0A04-0x0A7C	-	Reserved
0x0A80	DBGPSR0	Debug Power Status register
0x0A84-0x0AFC	-	Reserved
0x0B00	SYSPCR0	System Power Control Register
0x0B04-0x0B7C	-	Reserved
0x0B80	SYSPSR0	System Power Status Register
0x0B84-0x0BFC	-	Reserved
0x0C00	PRIDR0	Power Reset Identification Register
0x0C04-0x0FB4	-	Reserved
0x0FB8	AUTHSTATUS	Authentication Status Register
0x0FBC	DEVARCH	Device Architecture Register
0x0FC0-0x0FC4	-	Reserved
0x0FC8	DEVID	Device ID Register
0x0FCC	DEVTYPE	Device Type Register
0x0FD0	PIDR4	Peripheral Identification Register 4
0x0FD4-0FDC	-	Reserved
0x0FE0	PIDR0	Peripheral Identification Register 0
0x0FE4	PIDR1	Peripheral Identification Register 1
0x0FE8	PIDR2	Peripheral Identification Register 2
0x0FEC	PIDR3	Peripheral Identification Register 3
0x0FF0	CIDR0	Component Identification Register 0
0x0FF4	CIDR1	Component Identification Register 1
0x0FF8	CIDR2	Component Identification Register 2
0x0FFC	CIDR3	Component Identification Register 3

16.3 Cluster ROM table

The cluster ROM table contents depends on how you configured your cluster.

The following table lists the entries for the cluster ROM table, together with associated offsets from the physical base address of the ROM table. The cluster ROM table includes:

- All the debug components present at the cluster level including the cluster *Performance Monitoring Unit* (PMU) and the cluster *Embedded Logic Analyzer* (ELA).
- Entry points to the ROM tables for each standalone core or complex
- Power control registers for each standalone core or complex to allow core or complex powerup requests, see 16.4 ROM table power request registers for cluster and cores on page 198.

The ROMENTRY entry values depend on the number and type of cores implemented. The register formats are described in the *Arm® CoreSight™ Architecture Specification v3.0*.

Note

- If a complex of two cores is present, then each complex gets a single ROMENTRY that covers all cores in the complex. Therefore, where the table states Core, for example in the entry Core 0 ROM table, this can either be a core, a single-core complex, or a dual-core complex.
- In the following table, n corresponds to the ROMENTRY number for either the core or cluster.
- The cluster ROM table part number is 0x4E8.

Table 16-5: ROM table registers

Offset	Name	Description
0x0000	ROMENTRY0	Cluster PMU
0x0004	ROMENTRY1	Cluster ELA
0x0008	ROMENTRY2	Core 0 ROM table
0x000C	ROMENTRY3	Core 1 ROM table
0x0010	ROMENTRY4	Core 2 ROM table
0x0014	ROMENTRY5	Core 3 ROM table
0x0018	ROMENTRY6	Core 4 ROM table
0x001C	ROMENTRY7	Core 5 ROM table
0x0020	ROMENTRY8	Core 6 ROM table
0x0024	ROMENTRY9	Core 7 ROM table
0x0028-0x09FC	-	Reserved
0x0A00-0x0A1C	DBGPCR<n>	Debug Power Control Register for core <n>
0x0A20-0x0A7C	-	Reserved
0x0A80-0x0A9C	DBGPSR<n>	Debug Power Status Register for core <n>
0x0AA0-0x 0AFC	-	Reserved
0x0B00-0x0B1C	SYSPCR<n>	System Power Control Register for core <n>
0x0B20-0x 0B7C	-	Reserved

Offset	Name	Description
0x0B80-0x 0B9C	SYSPSR<n>	System Power Status Register for core <n>
0x0BA0-0x0BFC	-	Reserved
0x0C00-0x0C1C	PRIDR0	Power Reset Identification Register
0x0C20-0x0FB4	-	Reserved
0x0FB8	AUTHSTATUS	Authentication Status Register
0x0FBC	DEVARCH	Device Architecture Register
0x0FC0-0x0FC4	-	Reserved
0x0FC8	DEVID	Device ID Register
0x0FCC	DEVTYPE	Device Type Register
0x0FD0	PIDR4	Peripheral Identification Register 4
0x0FD4-0x0FDC	-	Reserved
0x0FE0	PIDR0	Peripheral Identification Register 0
0x0FE4	PIDR1	Peripheral Identification Register 1
0x0FE8	PIDR2	Peripheral Identification Register 2
0x0FEC	PIDR3	Peripheral Identification Register 3
0x0FF0	CIDR0	Component Identification Register 0
0x0FF4	CIDR1	Component Identification Register 1
0x0FF8	CIDR2	Component Identification Register 2
0x0FFC	CIDR3	Component Identification Register 3

16.4 ROM table power request registers for cluster and cores

Your debugger can program up the appropriate Debug Power Control Registers to request a powerup for the cluster, cores, or complexes from the corresponding *Power Policy Unit* (PPU).

Your debugger can use the power control register, DBGPCR<n>, located in the cluster ROM table, to make a request to powerup core<n> or complex<n>, where n corresponds to the ROMENTRY number for the core or complex.

Similarly, your debugger can use the power control register, DBGPCR0, located in the DebugBlock ROM table, to make a request to powerup the DSU-110 DynamIQ™ cluster.

Each corresponding core or cluster PPU then reacts to the request that was made as appropriate.

16.5 External cluster ROM registers

The summary table provides an overview of all *cluster ROM* (CLUSTERROM) table registers that are accessed externally (memory-mapped) over the debug APB bus. Individual register descriptions provide detailed information.

The summary table provides an overview of all the cluster ROM table registers. For more information about a register, click on the register name in the table.

- The cluster ROM table registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 16-6: CLUSTERROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	CLUSTERROM_ROMENTRY0	—	32-bit	Cluster ROM table entry 0	Yes
0x004	CLUSTERROM_ROMENTRY1	—	32-bit	Cluster ROM table entry 1	Yes
0x008	CLUSTERROM_ROMENTRY2	—	32-bit	Cluster ROM table entry 2	Yes
0x00C	CLUSTERROM_ROMENTRY3	—	32-bit	Cluster ROM table entry 3	Yes
0x010	CLUSTERROM_ROMENTRY4	—	32-bit	Cluster ROM table entry 4	Yes
0x014	CLUSTERROM_ROMENTRY5	—	32-bit	Cluster ROM table entry 5	Yes
0x018	CLUSTERROM_ROMENTRY6	—	32-bit	Cluster ROM table entry 6	Yes
0x01C	CLUSTERROM_ROMENTRY7	—	32-bit	Cluster ROM table entry 7	Yes
0x020	CLUSTERROM_ROMENTRY8	—	32-bit	Cluster ROM table entry 8	Yes
0x024	CLUSTERROM_ROMENTRY9	—	32-bit	Cluster ROM table entry 9	Yes
0x028	CLUSTERROM_ROMENTRY10	—	32-bit	Cluster ROM table entry 10	Yes
0x02C	CLUSTERROM_ROMENTRY11	—	32-bit	Cluster ROM table entry 11	Yes
0x030	CLUSTERROM_ROMENTRY12	—	32-bit	Cluster ROM table entry 12	Yes
0x034	CLUSTERROM_ROMENTRY13	—	32-bit	Cluster ROM table entry 13	Yes
0xA00	CLUSTERROM_DBGPCR0	—	32-bit	Cluster ROM table Debug Power Control Register 0	Yes
0xA04	CLUSTERROM_DBGPCR1	—	32-bit	Cluster ROM table Debug Power Control Register 1	Yes
0xA08	CLUSTERROM_DBGPCR2	—	32-bit	Cluster ROM table Debug Power Control Register 2	Yes
0xA0C	CLUSTERROM_DBGPCR3	—	32-bit	Cluster ROM table Debug Power Control Register 3	Yes
0xA10	CLUSTERROM_DBGPCR4	—	32-bit	Cluster ROM table Debug Power Control Register 4	Yes
0xA14	CLUSTERROM_DBGPCR5	—	32-bit	Cluster ROM table Debug Power Control Register 5	Yes
0xA18	CLUSTERROM_DBGPCR6	—	32-bit	Cluster ROM table Debug Power Control Register 6	Yes
0xA1C	CLUSTERROM_DBGPCR7	—	32-bit	Cluster ROM table Debug Power Control Register 7	Yes
0xA20	CLUSTERROM_DBGPCR8	—	32-bit	Cluster ROM table Debug Power Control Register 8	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xA24	CLUSTERROM_DBGPCR9	—	32-bit	Cluster ROM table Debug Power Control Register 9	Yes
0xA28	CLUSTERROM_DBGPCR10	—	32-bit	Cluster ROM table Debug Power Control Register 10	Yes
0xA2C	CLUSTERROM_DBGPCR11	—	32-bit	Cluster ROM table Debug Power Control Register 11	Yes
0xA80	CLUSTERROM_DBGPSR0	—	32-bit	Cluster ROM table Debug Power Status Register 0	Yes
0xA84	CLUSTERROM_DBGPSR1	—	32-bit	Cluster ROM table Debug Power Status Register 1	Yes
0xA88	CLUSTERROM_DBGPSR2	—	32-bit	Cluster ROM table Debug Power Status Register 2	Yes
0xA8C	CLUSTERROM_DBGPSR3	—	32-bit	Cluster ROM table Debug Power Status Register 3	Yes
0xA90	CLUSTERROM_DBGPSR4	—	32-bit	Cluster ROM table Debug Power Status Register 4	Yes
0xA94	CLUSTERROM_DBGPSR5	—	32-bit	Cluster ROM table Debug Power Status Register 5	Yes
0xA98	CLUSTERROM_DBGPSR6	—	32-bit	Cluster ROM table Debug Power Status Register 6	Yes
0xA9C	CLUSTERROM_DBGPSR7	—	32-bit	Cluster ROM table Debug Power Status Register 7	Yes
0xAA0	CLUSTERROM_DBGPSR8	—	32-bit	Cluster ROM table Debug Power Status Register 8	Yes
0xAA4	CLUSTERROM_DBGPSR9	—	32-bit	Cluster ROM table Debug Power Status Register 9	Yes
0xAA8	CLUSTERROM_DBGPSR10	—	32-bit	Cluster ROM table Debug Power Status Register 10	Yes
0xAAC	CLUSTERROM_DBGPSR11	—	32-bit	Cluster ROM table Debug Power Status Register 11	Yes
0xC00	CLUSTERROM_PRIDR0	—	32-bit	Cluster ROM table Power Request ID Register 0	Yes
0xFB8	CLUSTERROM_AUTHSTATUS	—	32-bit	Cluster ROM table Authentication Status Register	Yes
0xFBC	CLUSTERROM_DEVARCH	—	32-bit	Cluster ROM table Device Architecture Register	Yes
0xFC8	CLUSTERROM_DEVID	—	32-bit	Cluster ROM table Device Configuration Register	Yes
0xFCC	CLUSTERROM_DEVTYPE	—	32-bit	Cluster ROM table Device Type Register	Yes
0xFD0	CLUSTERROM_PIDR4	—	32-bit	Cluster ROM table Peripheral Identification Register 4	Yes
0xFE0	CLUSTERROM_PIDR0	—	32-bit	Cluster ROM table Peripheral Identification Register 0	Yes
0xFE4	CLUSTERROM_PIDR1	—	32-bit	Cluster ROM table Peripheral Identification Register 1	Yes
0xFE8	CLUSTERROM_PIDR2	—	32-bit	Cluster ROM table Peripheral Identification Register 2	Yes
0xFEC	CLUSTERROM_PIDR3	—	32-bit	Cluster ROM table Peripheral Identification Register 3	Yes
0xFF0	CLUSTERROM_CIDR0	—	32-bit	Cluster ROM table Component Identification Register 0	Yes
0xFF4	CLUSTERROM_CIDR1	—	32-bit	Cluster ROM table Component Identification Register 1	Yes
0xFF8	CLUSTERROM_CIDR2	—	32-bit	Cluster ROM table Component Identification Register 2	Yes
0xFFC	CLUSTERROM_CIDR3	—	32-bit	Cluster ROM table Component Identification Register 3	Yes

16.6 External debug ROM registers

The summary table provides an overview of all debug ROM (DBROM) table registers that are accessed externally (memory-mapped) over the debug APB bus. Individual register descriptions provide detailed information.

The summary table provides an overview of all the debug ROM table registers. For more information about a register, click on the register name in the table.


Note

- The DBROM table entry values are based on a cluster, implemented with the DSU-110. Configuration parameters are:
  - DEBUG_NEW_ADDR_MAP set to TRUE
  - DIRECT_CONNECT is set to FALSE
  - NUM_CORES is set to 12.
- The debug ROM table registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 16-7: DBROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	DBROM_ROMENTRY0	—	32-bit	DebugBlock ROM table Entry 0	Yes
0x004	DBROM_ROMENTRY1	—	32-bit	DebugBlock ROM table Entry 1	Yes
0x008	DBROM_ROMENTRY2	—	32-bit	DebugBlock ROM table Entry 2	Yes
0x00C	DBROM_ROMENTRY3	—	32-bit	DebugBlock ROM table Entry 3	Yes
0x010	DBROM_ROMENTRY4	—	32-bit	DebugBlock ROM table Entry 4	Yes
0x014	DBROM_ROMENTRY5	—	32-bit	DebugBlock ROM table Entry 5	Yes
0x018	DBROM_ROMENTRY6	—	32-bit	DebugBlock ROM table Entry 6	Yes
0x01C	DBROM_ROMENTRY7	—	32-bit	DebugBlock ROM table Entry 7	Yes
0x020	DBROM_ROMENTRY8	—	32-bit	DebugBlock ROM table Entry 8	Yes
0x024	DBROM_ROMENTRY9	—	32-bit	DebugBlock ROM table Entry 9	Yes
0x028	DBROM_ROMENTRY10	—	32-bit	DebugBlock ROM table Entry 10	Yes
0x02C	DBROM_ROMENTRY11	—	32-bit	DebugBlock ROM table Entry 11	Yes
0x030	DBROM_ROMENTRY12	—	32-bit	DebugBlock ROM table Entry 12	Yes
0x034	DBROM_ROMENTRY13	—	32-bit	DebugBlock ROM table Entry 13	Yes
0xA00	DBROM_DBGPCR0	—	32-bit	DebugBlock ROM table Debug Power Control Register 0	Yes
0xA80	DBROM_DBGPSR0	—	32-bit	DebugBlock ROM table Debug Power Status Register 0	Yes
0xC00	DBROM_PRIDR0	—	32-bit	DebugBlock ROM table Power Request ID Register 0	Yes
0xFB8	DBROM_AUTHSTATUS	—	32-bit	DebugBlock ROM table Authentication Status Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFBC	DBROM_DEVARCH	—	32-bit	DebugBlock ROM table Device Architecture Register	Yes
0xFC8	DBROM_DEVID	—	32-bit	DebugBlock ROM table Device Configuration Register	Yes
0xFCC	DBROM_DEVTYPE	—	32-bit	DebugBlock ROM table Device Type Register	Yes
0xFD0	DBROM_PIDR4	—	32-bit	DebugBlock ROM table Peripheral Identification Register 4	Yes
0xFE0	DBROM_PIDR0	—	32-bit	DebugBlock ROM table Peripheral Identification Register 0	Yes
0xFE4	DBROM_PIDR1	—	32-bit	DebugBlock ROM table Peripheral Identification Register 1	Yes
0xFE8	DBROM_PIDR2	—	32-bit	DebugBlock ROM table Peripheral Identification Register 2	Yes
0xFEC	DBROM_PIDR3	—	32-bit	DebugBlock ROM table Peripheral Identification Register 3	Yes
0xFF0	DBROM_CIDR0	—	32-bit	DebugBlock ROM table Component Identification Register 0	Yes
0xFF4	DBROM_CIDR1	—	32-bit	DebugBlock ROM table Component Identification Register 1	Yes
0xFF8	DBROM_CIDR2	—	32-bit	DebugBlock ROM table Component Identification Register 2	Yes
0xFFC	DBROM_CIDR3	—	32-bit	DebugBlock ROM table Component Identification Register 3	Yes

17. Performance Monitors Extension support

The *DynamIQ™ Shared Unit-110* (DSU-110) includes performance monitors that enable you to gather various statistics on the operation of the memory of the cluster during runtime. The performance monitors provide useful information about the behavior of the cluster that you can use when debugging or profiling code.

The *Performance Monitoring Unit* (PMU) provides six counters. Each counter can count any of the events available in the cluster. The absolute counts that are recorded might vary because of pipeline effects. This has negligible effect except in cases where the counters are enabled for a very short time.

17.1 PMU features

The *Performance Monitoring Unit* (PMU) includes the following interfaces and counters:

Event interface

Events from all other units from across the design are provided to the PMU.

System registers

You can program the PMU registers using the System registers. Alternatively, you can access the PMU registers through the memory-mapped Debug APB interface.

The cluster PMU is not accessible when the cluster is in Warm reset, such as during the OFF_EMU power mode.

Counters

The PMU has 64-bit counters that increment when they are enabled, based on events, and a 64-bit cycle counter.

PMU register interfaces

The *DynamIQ™ Shared Unit-110* (DSU-110) supports access to the performance monitor registers from the internal System register interface. The *DynamIQ™ Shared Unit-110* (DSU-110) also supports access to the PMU through the memory-mapped Debug APB interface.

17.2 PMU events

The following table shows the events that are generated and the numbers that the *Performance Monitoring Unit* (PMU) uses to reference the events.

Table 17-1: PMU events

PMU event number	Event mnemonic	Event description
0x0011	CPU_CYCLES	Cycle counter.
0x0019	BUS_ACCESS	Bus access counter. Counts every beat of data that is transferred over the data channels between the *Snoop Control Unit* (SCU) and the interconnect. This event counts the sum of BUS_ACCESS_RD and BUS_ACCESS_WR.
0x001A	MEMORY_ERROR	Local memory error counter. Counts for each cycle where there is a Correctable or Uncorrectable memory error (ECC or parity) in the protected RAMs.
0x001D	BUS_CYCLES	ACE or CHI bus cycle counter.
0x0029	L3D_CACHE_ALLOCATE	Level 3 unified cache allocation without refill counter. Counts every full cache line write into the L3 cache which does not cause a linefill.
0x002A	L3D_CACHE_REFILL	Level 3 unified cache refill counter. Counts every cacheable read transaction issued to the interconnect. This event counts the sum of L3D_CACHE_REFILL_RD and L3D_CACHE_REFILL_WR.
0x002B	L3D_CACHE	Level 3 unified cache access counter. Counts every cacheable read or write transaction issued to the SCU. This event counts the sum of L3D_CACHE_RD and L3D_CACHE_WR.
0x002C	L3D_CACHE_WB	Level 3 unified cache write-back counter. Counts every write-back from the L3 cache.
0x0060	BUS_ACCESS_RD	Bus access, read counter. Counts every beat of data transferred over the read data channel between the *DynamIQ™ Shared Unit-110* (DSU-110) and the interconnect. **Note:** If the cluster generates a CHI MakeReadUnique transaction for a shared line upgrade, it is unknown at the time of counting if this results in a data transfer or not. Therefore, the counter assumes the data will not be transferred.
0x0061	BUS_ACCESS_WR	Bus access, write counter. Counts every beat of data transferred over the write data channel between the DSU-110 and the interconnect. **Note:** If the cluster generates a CHI WriteEvictOrEvict transaction for a clean eviction, it is unknown at the time of counting if this results in a data transfer or not. Therefore, the counter assumes the data will be transferred.
0x0062	BUS_ACCESS_SHARED	Bus access, shared counter. Counts every beat of shared data transferred over the data channels between the SCU and the interconnect.
0x0063	BUS_ACCESS_NOT_SHARED	Bus access, not shared counter. Counts every beat of not shared data transferred over the write data channel between the SCU and the interconnect.
0x0064	BUS_ACCESS_NORMAL	Bus access, normal counter. Counts every beat of normal data transferred over the write data channel between the SCU and the interconnect.
0x0065	BUS_ACCESS_PERIPH	Bus access, periph counter. Counts every beat of device data transferred over the write data channel between the SCU and the interconnect.

PMU event number	Event mnemonic	Event description
0x00A0	L3D_CACHE_RD	Level 3 unified cache access, read counter. Counts every cacheable shareable read transaction that is issued to the SCU. Prefetches and stashes are not counted.
0x00A1	L3D_CACHE_WR	Level 3 unified cache access, write counter. Counts every cacheable write transaction issued to the SCU.
0x00A2	L3D_CACHE_REFILL_RD	Level 3 unified cache refill, read counter. Counts every cacheable read transaction issued to the interconnect caused by a cacheable shareable read transaction. Prefetches and stashes are not counted.
0x00A3	L3D_CACHE_REFILL_WR	Level 3 unified cache refill, write counter. Counts every cacheable read transaction issued to the interconnect caused by a write transaction.
0x0119	ACP_ACCESS	ACP access counter. Counts every beat of data transferred over the data channels between the SCU and the accelerated coherency port. This event counts the sum of ACP_ACCESS_RD and ACP_ACCESS_WR.
0x011D	ACP_CYCLES	ACP cycle counter.
0x0160	ACP_ACCESS_RD	ACP access, read counter. Counts every beat of data transferred over the read data channel between the SCU and the peripheral port.
0x0161	ACP_ACCESS_WR	ACP access, write counter. Counts every beat of data transferred over the write data channel between the SCU and the peripheral port.
0x0219	PPT_ACCESS	Peripheral port access counter. Counts every beat of data transferred over the data channels between the SCU and the peripheral port. This event counts the sum of PP_ACCESS_RD and PP_ACCESS_WR.
0x021D	PP_CYCLES	Peripheral port cycle counter.
0x0260	PP_ACCESS_RD	Peripheral port access, read counter. Counts every beat of data transferred over the read data channel between the SCU and the peripheral port.
0x0261	PP_ACCESS_WR	Peripheral port access, write counter. Counts every beat of data transferred over the write data channel between the SCU and the peripheral port.
0x00C0	SCU_SNP_ACCESS	SNP access counter. Counts every external snoop request.
0x00C1	SCU_SNP_EVICT	SNP evictions counter. Counts every invalidating external snoop request that causes an L3 cache eviction.
0x00C2	SCU_SNP_NO_CPU_SNP	SNP, no CPU snoop counter. Counts every external snoop request that completes without needing to snoop a core.
0x0500	SCU_PFTCH_CPU_ACCESS	Prefetch access, CPU counter. Counts every stash transaction originating from a core.
0x0501	SCU_PFTCH_CPU_MISS	Prefetch data miss, CPU counter. Counts every stash transaction originating from a core where data was read in from outside the cluster.
0x0502	SCU_PFTCH_CPU_HIT	Prefetch data hit, CPU counter. Counts every stash transaction originating from a core where the stash hit in the cluster or the stash is not performed due to the L3 cache being off.
0x0510	SCU_STASH_ICN_ACCESS	Stash access, ICN counter. Counts every stash transaction originating from the interconnect.
0x0511	SCU_STASH_ICN_MISS	Stash data miss, ICN counter. Counts every stash transaction originating from the interconnect which utilizes a data pull, or is added to the stash queue and later issues a read.
0x0512	SCU_STASH_ICN_HIT	Stash data hit, ICN counter. Counts every non-invalidating stash transaction originating from the interconnect which hits in the cluster.
0x0515	SCU_STASH_ICN_DROPPED	Stash dropped, ICN counter. Counter for every dropped stash transaction originating from the interconnect for which a data-pull of read are not used due to a lack of resources or the L3 cache being off.

PMU event number	Event mnemonic	Event description
0x0520	SCU_STASH_ACP_ACCESS	Stash access, ACP counter. Counter for every dataless stash-supported transaction originating from an ACP.
0x0521	SCU_STASH_ACP_MISS	Stash data miss, ACP counter. Counter for every dataless transaction originating from ACP where data was read in from outside the cluster.
0x0522	SCU_STASH_ACP_HIT	Stash data hit, ACP counter. Counter for every dataless stash transaction originating from the ACP where the stash hit in the cluster or was not performed due to the L3 cache being off.
0x00D0	SCU_HZD_ADDRESS	Arbitration hazard, address counter. Counts every flush caused by an address hazard.
0x00F3	SCU_BIB_ACCESS	BIB access. Counts for every snoop filter back invalidation.

17.3 PMU interrupt

The DSU-110 asserts the **nCLUSTERPMUIRQ** signal when the PMU generates an interrupt.

You can route this signal to an external interrupt controller for prioritization and masking. This is the only mechanism that signals this interrupt to a core. When the interrupt is generated, a trigger is also sent to the cluster *Cross Trigger Interface* (CTI).

17.4 External cluster PMU registers

The summary table provides an overview of all the *Cluster Performance Monitoring Unit* (CLUSTERPMU) registers that are accessed externally (memory-mapped) over the Debug APB bus. Individual register descriptions provide detailed information.

The summary table provides an overview of all the cluster PMU registers that are accessed externally (memory-mapped) over the debug APB bus. For more information about a register, click on the register name in the table.

Note

- The cluster PMU registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- The part number is 0x4EA.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 17-2: CLUSTERPMU registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0	CLUSTERPMU_PMEVCNTR0	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x8	CLUSTERPMU_PMEVCNTR1	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x10	CLUSTERPMU_PMEVCNTR2	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x18	CLUSTERPMU_PMEVCNTR3	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x20	CLUSTERPMU_PMEVCNTR4	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x28	CLUSTERPMU_PMEVCNTR5	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x0F8	CLUSTERPMU_PMCCNTR	—	64-bit	Cluster Performance Monitors Cycle Counter	No
0x0FC	CLUSTERPMU_PMCCNTR	—	64-bit	Cluster Performance Monitors Cycle Counter	No
0x400	CLUSTERPMU_PMEVTYPER0	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x404	CLUSTERPMU_PMEVTYPER1	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x408	CLUSTERPMU_PMEVTYPER2	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x40C	CLUSTERPMU_PMEVTYPER3	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x410	CLUSTERPMU_PMEVTYPER4	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x414	CLUSTERPMU_PMEVTYPER5	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x47C	CLUSTERPMU_PMCCFILTR	—	32-bit	Cluster Performance Monitors Cycle Counter Filter Register	No
0x610	CLUSTERPMU_PMSSSR	—	32-bit	Cluster Performance Monitors Snapshot Status register	No
0x614	CLUSTERPMU_PMOVSSR	—	32-bit	Cluster Performance Monitors Overflow Status Snapshot register	No
0x618	CLUSTERPMU_PMCCNTSR	—	64-bit	Cluster Performance Monitors Cycle Counter Snapshot Register	No
0x61C	CLUSTERPMU_PMCCNTSR	—	64-bit	Cluster Performance Monitors Cycle Counter Snapshot Register	No
0x620	CLUSTERPMU_PMEVCNTSR0	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x628	CLUSTERPMU_PMEVCNTSR1	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x630	CLUSTERPMU_PMEVCNTSR2	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x638	CLUSTERPMU_PMEVCNTSR3	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x640	CLUSTERPMU_PMEVCNTSR4	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x648	CLUSTERPMU_PMEVCNTSR5	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x6F0	CLUSTERPMU_PMSSCR	—	32-bit	Cluster Performance Monitors Snapshot Capture register	No
0x6F4	CLUSTERPMU_PMSSRR	—	32-bit	Cluster Performance Monitors Snapshot Reset register	No
0xC00	CLUSTERPMU_PMCNTENSET	—	32-bit	Cluster Performance Monitors Count Enable Set register	No
0xC20	CLUSTERPMU_PMCNTENCLR	—	32-bit	Cluster Performance Monitors Count Enable Clear register	No
0xC40	CLUSTERPMU_PMINTENSET	—	32-bit	Cluster Performance Monitors Interrupt Enable Set register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xC60	CLUSTERPMU_PMINTENCLR	—	32-bit	Cluster Performance Monitors Interrupt Enable Clear register	No
0xC80	CLUSTERPMU_PMOVSCLR	—	32-bit	Cluster Performance Monitors Overflow Flag Status Clear register	No
0xCC0	CLUSTERPMU_PMOVSSET	—	32-bit	Cluster Performance Monitors Overflow Flag Status Set register	No
0xE00	CLUSTERPMU_PMCFGR	—	32-bit	Cluster Performance Monitors Configuration Register	No
0xE04	CLUSTERPMU_PMCR	—	32-bit	Cluster Performance Monitors Control Register	No
0xE20	CLUSTERPMU_PMCEID0	—	32-bit	Cluster Performance Monitors Common Event Identification register 0	No
0xE24	CLUSTERPMU_PMCEID1	—	32-bit	Cluster Performance Monitors Common Event Identification register 1	No
0xE28	CLUSTERPMU_PMCEID2	—	32-bit	Cluster Performance Monitors Common Event Identification register 2	No
0xE2C	CLUSTERPMU_PMCEID3	—	32-bit	Cluster Performance Monitors Common Event Identification register 3	No
0xFA0	CLUSTERPMU_PMCLAIMSET	—	32-bit	Cluster Performance Monitors Claim Set register	No
0xFA4	CLUSTERPMU_PMCLAIMCLR	—	32-bit	Cluster Performance Monitors Claim Clear register	No
0xFA8	CLUSTERPMU_PMDEVAFF0	—	32-bit	Cluster Performance Monitors Device Affinity register 0	No
0xFAC	CLUSTERPMU_PMDEVAFF1	—	32-bit	Cluster Performance Monitors Device Affinity register 1	No
0xFB8	CLUSTERPMU_PMAUTHSTATUS	—	32-bit	Cluster Performance Monitors Authentication Status register	No
0xFBC	CLUSTERPMU_PMDEVARCH	—	32-bit	Cluster Performance Monitors Device Architecture register	No
0xFC8	CLUSTERPMU_PMDEVID	—	32-bit	Cluster Performance Monitors Device ID register	No
0xFCC	CLUSTERPMU_PMDEVTYPE	—	32-bit	Cluster Performance Monitors Device Type register	No
0xFD0	CLUSTERPMU_PMPIDR4	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 4	No
0xFE0	CLUSTERPMU_PMPIDR0	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 0	No
0xFE4	CLUSTERPMU_PMPIDR1	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 1	No
0xFE8	CLUSTERPMU_PMPIDR2	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 2	No
0xFEC	CLUSTERPMU_PMPIDR3	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 3	No
0xFF0	CLUSTERPMU_PMCIDR0	—	32-bit	Cluster Performance Monitors Component Identification Register 0	No
0xFF4	CLUSTERPMU_PMCIDR1	—	32-bit	Cluster Performance Monitors Component Identification Register 1	No
0xFF8	CLUSTERPMU_PMCIDR2	—	32-bit	Cluster Performance Monitors Component Identification Register 2	No
0xFFC	CLUSTERPMU_PMCIDR3	—	32-bit	Cluster Performance Monitors Component Identification Register 3	No

Appendix A AArch64 registers

This appendix contains the descriptions for all the AArch64 registers in the *DynamIQ™ Shared Unit-110* (DSU-110).

A.1 AArch64 generic system control register summary

The summary table provides an overview of all generic-system-control registers in the *DynamIQ™ Shared Unit-110* (DSU-110). Individual register descriptions provide detailed information.

- Any AArch64 generic system control registers that are not present in Direct connect are treated as **RAZ/WI**.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table A-1: Generic system control registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERCFR_EL1	3	0	C15	C3	0	–	64-bit	Cluster Configuration Register	Yes
IMP_CLUSTERIDR_EL1	3	0	C15	C3	1	–	64-bit	Cluster Main Revision Register	Yes
IMP_CLUSTERREVIDR_EL1	3	0	C15	C3	2	–	64-bit	Cluster ECO ID Register	Yes
IMP_CLUSTERACTLR_EL1	3	0	C15	C3	3	–	64-bit	Cluster Auxiliary Control Register	Yes
IMP_CLUSTERECTLR_EL1	3	0	C15	C3	4	–	64-bit	Cluster Extended Control Register	Yes
IMP_CLUSTERPWRCTLR_EL1	3	0	C15	C3	5	–	64-bit	Cluster Power Control Register	No
IMP_CLUSTERPWRDN_EL1	3	0	C15	C3	6	–	64-bit	Cluster Power Down Register	Yes
IMP_CLUSTERPWRSTAT_EL1	3	0	C15	C3	7	–	64-bit	Cluster Power Status Register	No
IMP_CLUSTERL3DNTH0_EL1	3	0	C15	C4	0	–	64-bit	Cluster L3 Downsize Threshold0 Register	No
IMP_CLUSTERL3DNTH1_EL1	3	0	C15	C4	1	–	64-bit	Cluster L3 Downsize Threshold1 Register	No
IMP_CLUSTERL3UPTH0_EL1	3	0	C15	C4	2	–	64-bit	Cluster L3 Upsize Threshold0 Register	No
IMP_CLUSTERL3UPTH1_EL1	3	0	C15	C4	3	–	64-bit	Cluster L3 Upsize Threshold1 Register	No
IMP_CLUSTERBUSQOS_EL1	3	0	C15	C4	4	–	64-bit	Cluster Bus QoS Control Register	No
IMP_CLUSTERL3HIT_EL1	3	0	C15	C4	5	–	64-bit	Cluster L3 Hit Counter Register	No
IMP_CLUSTERL3MISS_EL1	3	0	C15	C4	6	–	64-bit	Cluster L3 Miss Counter Register	No
IMP_CLUSTERPPSTART_EL1	3	0	C15	C9	0	–	64-bit	Cluster peripheral port Start Address Register	No
IMP_CLUSTERPPEND_EL1	3	0	C15	C9	1	–	64-bit	Cluster peripheral port End Address Register	No
IMP_CLUSTERCFR2_EL1	3	0	C15	C9	2	–	64-bit	Cluster Configuration Register 2	No

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERCDBG_EL3	3	6	C15	C4	7	—	64-bit	Cluster Cache Debug Register	No
IMP_CLUSTERPMMDCR_EL3	3	6	C15	C6	3	—	64-bit	Monitor Debug Configuration Register (EL3)	No

A.1.1 IMP_CLUSTERCFR_EL1, Cluster Configuration Register

Contains details of the hardware configuration of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

xxxx xxxx x0xx x000 0000 0000 0000 000x xxxx xxx0 xx0x x0xx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-1: AArch64_imp_clustercfr_el1 bit assignments

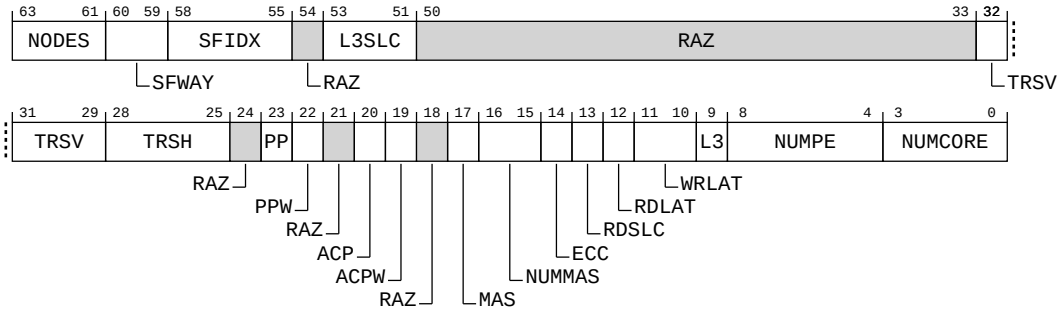


Table A-2: IMP_CLUSTERCFR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:61]	NODES	Number of transport nodes. **0b000** Direct connect. **0b001** One node. **0b010** Two nodes. **0b011** Three nodes. **0b100** Four nodes. **0b101** Eight nodes.	xxx
[60:59]	SFWAY	Number of Snoop Filter ways. **0b00** 4 ways **0b01** 6 ways **0b10** 8 ways **0b11** 12 ways	xx
[58:55]	SFIDX	Log2 of the number of snoop filter indexes.	xxxx
[54]	**RAZ**	Reserved	**RAZ**
[53:51]	L3SLC	Number of L3 cache slices. **0b000** Eight L3 cache slices. **0b001** One L3 cache slice. **0b010** Two L3 cache slices. **0b100** Four L3 cache slices.	xxx
[50:33]	**RAZ**	Reserved	**RAZ**

Bits	Name	Description	Reset
[32:29]	TRSV	Transport register slices, vertical. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx
[28:25]	TRSH	Transport register slices, horizontal. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx
[24]	**RAZ**	Reserved	**RAZ**

Bits	Name	Description	Reset
[23]	PP	Peripheral port presence. **0b0** No peripheral port present **0b1** Peripheral port present	x
[22]	PPW	Peripheral port width. **0b0** 64 bit data width **0b1** 256 bit data width	x
[21]	**RAZ**	Reserved	**RAZ**
[20]	ACP	ACP interface presence. **0b0** No ACP interface present **0b1** ACP interface present	x
[19]	ACPW	ACP interface width. **0b0** 128 bit data width **0b1** 256 bit data width	x
[18]	**RAZ**	Reserved	**RAZ**
[17]	MAS	Master bus interface type. **0b0** AXI interface **0b1** CHI interface	x
[16:15]	NUMMAS	Number of Master interfaces. **0b00** One master **0b01** Two masters **0b10** Three masters **0b11** Four masters	xx
[14]	ECC	SCU-L3 ECC configuration. **0b0** SCU-L3 is configured with no ECC **0b1** SCU-L3 is configured with ECC	x

Bits	Name	Description	Reset
[13]	RDSLC	L3 data RAM read register slice. **0b0** No register slice present **0b1** Register slice present	x
[12]	RDLAT	L3 Data RAM read latency. **0b0** Two cycle output delay from L3 data RAMs **0b1** Three cycle output delay from L3 data RAMs	x
[11:10]	WRLAT	L3 Data RAM write latency. **0b00** One cycle input delay from L3 data RAMs **0b01** Two cycle input delay from L3 data RAMs **0b10** Two cycle input delay plus a one cycle hold	xx
[9]	L3	L3 cache presence. **0b0** No L3 cache present **0b1** L3 cache present	x
[8:4]	NUMPE	Number of PEs present in the cluster. For single threaded cores, this number will be the same as bits [3:0]; for multi-threaded cores it will be larger.	5{x}

Bits	Name	Description	Reset
[3:0]	NUMCORE	Number of cores present in the cluster. **0b0000** One core **0b0001** Two cores **0b0010** Three cores **0b0011** Four cores **0b0100** Five cores **0b0101** Six cores **0b0110** Seven cores **0b0111** Eight cores **0b1000** Nine core **0b1001** Ten cores **0b1010** Eleven cores **0b1011** Twelve cores	xxxx

Access

MRS <Xt>, S3_0_C15_C3_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b000

MSR S3_0_C15_C3_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b000

Accessibility

MRS <Xt>, S3_0_C15_C3_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
```

```
    else
        return IMP_CLUSTERCFR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERCFR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERCFR_EL1;
```

MSR S3_0_C15_C3_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERCFR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERCFR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERCFR_EL1 = X[t];
```

A.1.2 IMP_CLUSTERIDR_EL1, Cluster Main Revision Register

Holds the revision and patch level of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0100 0000


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-2: AArch64_imp_clusteridr_el1 bit assignments

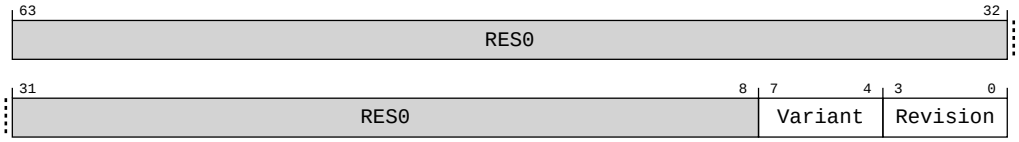


Table A-5: IMP_CLUSTERIDR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:8]	**RES0**	Reserved	**RES0**
[7:4]	Variant	Indicates the variant of the DSU. This is the major revision number x in the rx part of the rxpy description of the product revision status. **0b0000** Cluster major revision 0. **0b0001** Cluster major revision 1. **0b0010** Cluster major revision 2. **0b0011** Cluster major revision 3. **0b0100** Cluster major revision 4.	0b0100
[3:0]	Revision	Indicates the minor revision number of the DSU. This is the minor revision number y in the py part of the rxpy description of the product revision status. **0b0000** Cluster minor revision 0. **0b0001** Cluster minor revision 1. **0b0010** Cluster minor revision 2. **0b0011** Cluster minor revision 3. **0b0100** Cluster minor revision 4.	0b0000

Access

MRS <Xt>, S3_0_C15_C3_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b001

MSR S3_0_C15_C3_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b001

Accessibility

MRS <Xt>, S3_0_C15_C3_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERIDR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERIDR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERIDR_EL1;
```

MSR S3_0_C15_C3_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERIDR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERIDR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERIDR_EL1 = X[t];
```

A.1.3 IMP_CLUSTERREVIDR_EL1, Cluster ECO ID Register

Enables ECO patches to be applied to the cluster level to be identified by software.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-3: AArch64_imp_clusterrevidr_el1 bit assignments

Table A-8: IMP_CLUSTERREVIDR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	ECOID	Contains ECO information. Refer to the errata documentation for any bit allocations. **0b00000000000000000000000000000000000000000000000000000000000000000** Customer ECO ID	0x0000000000000000

Access

MRS <Xt>, S3_0_C15_C3_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b010

MSR S3_0_C15_C3_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b010

Accessibility

MRS <Xt>, S3_0_C15_C3_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERREVIDR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERREVIDR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERREVIDR_EL1;
```

MSR S3_0_C15_C3_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
```

```
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERREVIDR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERREVIDR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERREVIDR_EL1 = X[t];
```

A.1.4 IMP_CLUSTERACTLR_EL1, Cluster Auxiliary Control Register

These register bits are reserved for Arm test purposes only and must not be used except under direction from Arm.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-4: AArch64_imp_clusteractlr_el1 bit assignments

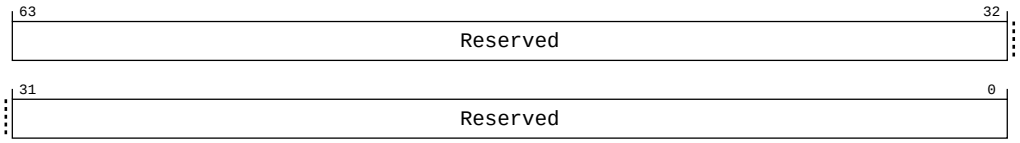


Table A-11: IMP_CLUSTERACTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	Reserved	Reserved for Arm internal use	64{x}

Access

MRS <Xt>, S3_0_C15_C3_3

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b011

MSR S3_0_C15_C3_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b011

Accessibility

MRS <Xt>, S3_0_C15_C3_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERACTLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERACTLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERACTLR_EL1;
```

MSR S3_0_C15_C3_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ACTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ACTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ACTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERACTLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ACTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ACTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERACTLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERACTLR_EL1 = X[t];
```

A.1.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register

This register should be used for dynamically changing implementation specific control bits.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0011 0100 0000 0000 0000 0000 0000 00xx x000 0101 0101 0010

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-5: AArch64_imp_clusterectlr_el1 bit assignments

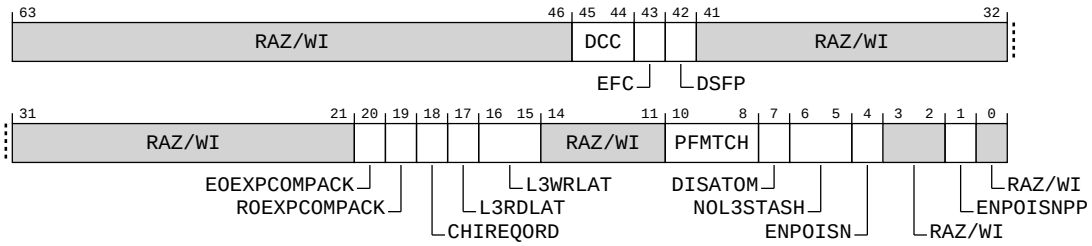


Table A-14: IMP_CLUSTERECTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:46]	**RAZ/WI**	Reserved	**RAZ/WI**

Bits	Name	Description	Reset
[45:44]	DCC	Downstream cache control. Controls whether evictions of clean cachelines send data on the CHI interface. Set this based on whether there is a cache on the path to memory. This bit is **RES0** in direct connect configuration. **0b00** Disables sending data when clean cachelines are evicted. **0b01** Enables sending WriteEvictFull transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **0b10** Enables sending WriteEvictOrEvict transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **0b11** Enables sending WriteEvictOrEvict transactions when Unique Clean or Shared Clean cachelines are evicted. This is the reset value.	0b11
[43]	EFC	Eviction flush control. Controls whether hardware cache flushes and DC CISW instructions send data when evicting clean cachelines on the CHI interface. This bit is **RES0** in direct connect configuration. **0b0** Disables sending data when hardware cache flushes or DC CISW instructions evict a clean cacheline. Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP). This is the reset value. **0b1** Sending of data when hardware cache flushes or DC CISW instructions evict clean cachelines is controlled by Downstream Cache Control (DCC). Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP).	0b0
[42]	DSFP	Downstream snoop filter present. Enables sending Evict transactions on the CHI interface when clean cachelines are evicted without data. Enable this if there is at least one snoop filter in the path to memory. This bit is **RES0** in direct connect configuration. **0b0** Disables sending Evict transactions when clean cachelines are evicted without data. **0b1** Enables sending of Evict transactions when clean cachelines are evicted without data. This is the reset value.	0b1
[41:21]	**RAZ/WI**	Reserved	**RAZ/WI**
[20]	EOEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Endpoint Order transactions to the system **0b0** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=1	0b0
[19]	ROEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Request Order transactions to the system **0b0** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=1	0b0

Bits	Name	Description	Reset
[18]	CHIREQORD	Allow Request Order on CHI ports. Enables the use of Request Order when sending Non-snoopable CHI transactions to the system for Dev-R and Normal NC memory. **0b0** Disables sending Request Order on the CHI interface to the system. Will send No Order instead. **0b1** Enables sending Request Order on the CHI interface to the system for Non-snoopable transactions.	0b0
[17]	L3RDLAT	L3 data RAM read (output) latency. **0b0** The L3 data RAM output latency is 2 cycles. **0b1** The L3 data RAM output latency is 3 cycles.	x
[16:15]	L3WRLAT	L3 data RAM write (input) latency. This bit is **RES0** in direct connect configuration. **0b00** The L3 data RAM input latency is 1 cycle with an additional hold cycle. **0b01** The L3 data RAM input latency is 2 cycles without an additional hold cycle. **0b10** The L3 data RAM input latency is 2 cycles with an additional hold cycle. This is only usable if the L3 data RAM output latency is 3 cycles.	xx
[14:11]	**RAZ/WI**	Reserved	**RAZ/WI**
[10:8]	PFMTCH	Prefetch matching delay. Controls the amount of tie a prefetch waits for a possible match with a later read. Encoded as powers of 2, from 1-128. This bit is **RES0** in direct connect configuration. **0b000** Wait for 1 cycle. **0b001** Wait for 2 cycles. **0b010** Wait for 4 cycles. **0b011** Wait for 8 cycles. **0b100** Wait for 16 cycles. **0b101** Wait for 32 cycles. **0b110** Wait for 64 cycles. **0b111** Wait for 128 cycles.	0b101

Bits	Name	Description	Reset
[7]	DISATOM	Disable cacheable shareable atomics being sent to the interconnect. This bit is **RES0** in direct connect configuration. **0b0** Cacheable shareable atomics will be sent to the interconnect if the BROADCASTATOMIC pin is set. **0b1** Cacheable shareable atomics will be handled inside the cluster.	0b0
[6:5]	NOL3STASH	CPU StashOnce request behaviour when L3 is not present or powered down. This bit is **RES0** in direct connect configuration. **0b00** Stashes are sent out to the interconnect, if supported. **0b01** Normal read request sent to interconnect. **0b10** StashOnce has no effect.	0b10
[4]	ENPOISN	Interconnect data poisoning support for the CHI Master(s). This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES0** in direct connect configuration. **0b0** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **0b1** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	0b1
[3:2]	**RAZ/WI**	Reserved	**RAZ/WI**
[1]	ENPOISNPP	Interconnect data poisoning support for the CHI peripheral port. This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES0** in direct connect configuration. **0b0** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **0b1** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	0b1
[0]	**RAZ/WI**	Reserved	**RAZ/WI**

Access

MRS <Xt>, S3_0_C15_C3_4

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b100

MSR S3_0_C15_C3_4, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b100

Accessibility

MRS <Xt>, S3_0_C15_C3_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERECTLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERECTLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERECTLR_EL1;
```

MSR S3_0_C15_C3_4, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERECTLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERECTLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERECTLR_EL1 = X[t];
```

A.1.6 IMP_CLUSTERPWRCTLR_EL1, Cluster Power Control Register

This register controls power features of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0111 0000

Bit descriptions

Figure A-6: AArch64_imp_clusterpwrctlr_el1 bit assignments

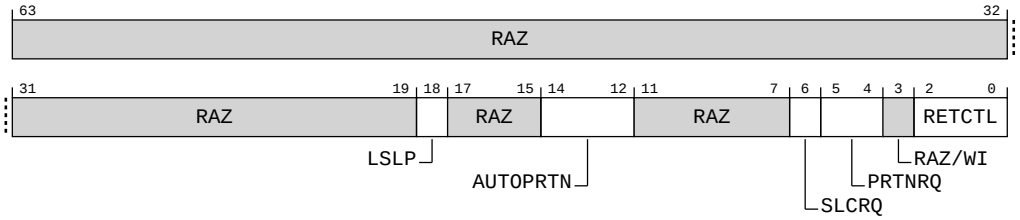


Table A-17: IMP_CLUSTERPWRCTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:19]	**RAZ**	Reserved	**RAZ**
[18]	LSLP	Enable L3 RAM light sleep.	0b0
[17:15]	**RAZ**	Reserved	**RAZ**

Bits	Name	Description	Reset
[14:12]	AUTOPRTN	Enable automatic RAM power down and configure evaluation time period. Note that a shorter time period allows better responsiveness to changing workloads, however if it is too short then the cost of frequent resizing can be too high. **0b000** Disabled **0b001** 8,192 architectural timer ticks, time period of 164us-819us **0b010** 16,384 architectural timer ticks, time period of 328us-1.6ms **0b011** 32,768 architectural timer ticks, time period of 655us-3.3ms **0b100** 65,536 architectural timer ticks, time period of 1.3ms-6.6ms **0b101** 131,072 architectural timer ticks, time period of 2.6ms-13ms **0b110** 262,144 architectural timer ticks, time period of 5.2ms-26ms **0b111** 524,288 architectural timer ticks, time period of 10ms-52ms	0b000
[11:7]	**RAZ**	Reserved	**RAZ**
[6]	SLCRQ	Cache slice power request. These bits are passed to the PPU as an advisory request for which slices to power. **0b0** Request that one L3 cache slice is powered on. **0b1** Request that all L3 cache slices are powered on.	0b1
[5:4]	PRTNRQ	Cache portion power request. These bits are passed to the PPU as an advisory request for which portions to power. Note that these bits are only used when AUTOPRTN bits are 3'b000. **0b00** Request that none of the L3 cache portions in each slice is powered on **0b01** Request that half of the L3 cache portions in each slice are powered on **0b11** Request that both of the L3 cache portions in each slice are powered on	0b11
[3]	**RAZ/WI**	Reserved	**RAZ/WI**

Bits	Name	Description	Reset
[2:0]	RETCTL	L3 Data RAM retention control. **0b000** Disable the retention circuit. **0b001** 2 architectural timer ticks, 40ns-200ns minimum delay before retention **0b010** 8 architectural timer ticks, 160ns-800ns minimum delay before retention **0b011** 32 architectural timer ticks, 640ns-3,200ns minimum delay before retention **0b100** 64 architectural timer ticks, 1280ns-6,400ns minimum delay before retention **0b101** 128 architectural timer ticks, 2,560ns-12,800ns minimum delay before retention **0b110** 256 architectural timer ticks, 5,120ns-25,600ns minimum delay before retention **0b111** 512 architectural timer ticks, 10,240ns-51,200ns minimum delay before retention	0b000

Access

MRS <Xt>, S3_0_C15_C3_5

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b101

MSR S3_0_C15_C3_5, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b101

Accessibility

MRS <Xt>, S3_0_C15_C3_5

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPWRCTLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPWRCTLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPWRCTLR_EL1;
```

MSR S3_0_C15_C3_5, <Xt>

```
if PSTATE.EL == EL0 then
```

```
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRCTLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRCTLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPWRCTLR_EL1 = X[t];
```

A.1.7 IMP_CLUSTERPWRDN_EL1, Cluster Power Down Register

This register controls powerdown requirements of the cluster and is banked per-thread.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-7: AArch64_imp_clusterpwrdn_el1 bit assignments

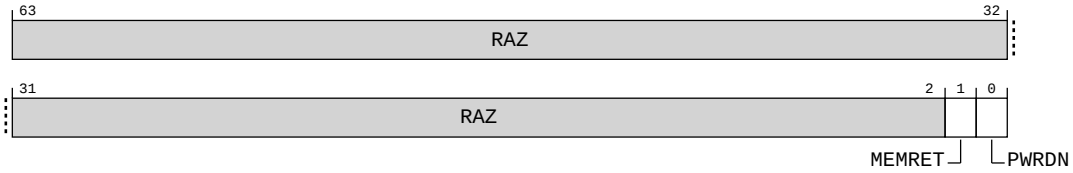


Table A-20: IMP_CLUSTERPWRDN_EL1 bit descriptions

Bits	Name	Description	Reset
[63:2]	RAZ	Reserved	RAZ
[1]	MEMRET	Indicate to the PPU that memory retention is desired when all cores are powered down. This is an advisory status to the PPU and will not cause an explicit request to power off the cluster to be denied.	0b0
[0]	PWRDN	Indicate to the PPU that cluster power is required even when all cores are powered down. This is an advisory status to the PPU and will not cause an explicit request to power off the cluster to be denied.	0b0

Access

MRS <Xt>, S3_0_C15_C3_6

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b110

MSR S3_0_C15_C3_6, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b110

Accessibility

MRS <Xt>, S3_0_C15_C3_6

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPWRDN_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPWRDN_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPWRDN_EL1;
```

MSR S3_0_C15_C3_6, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRDN_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRDN_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPWRDN_EL1 = X[t];
```

A.1.8 IMP_CLUSTERPWRSTAT_EL1, Cluster Power Status Register

This register contains the current status of power features and is read-only.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0xxx 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-8: AArch64_imp_clusterpwrstat_el1 bit assignments

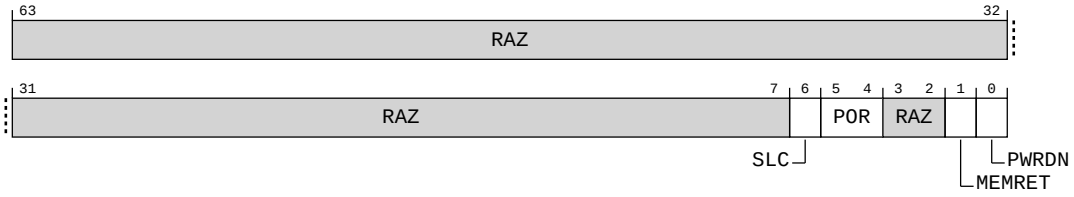


Table A-23: IMP_CLUSTERPWRSTAT_EL1 bit descriptions

Bits	Name	Description	Reset
[63:7]	RAZ	Reserved	RAZ
[6]	SLC	Cache slice power status. This indicates which cache slices are currently powered up and available. It can be used to determine when the state requested in bit [6] of the IMP_CLUSTERPWRCTLR_EL1 has taken effect.	x
[5:4]	POR	Cache portion power status. This indicates which cache portions are currently powered up and available. It can be used to determine when the state requested in bits [5:4] of the IMP_CLUSTERPWRCTLR_EL1 has taken effect.	xx
[3:2]	RAZ	Reserved	RAZ
[1]	MEMRET	Enable memory retention when all cores are powered down. Note this bit is a combined version of all banked per-thread bits from the IMP_CLUSTERPWRDN_EL1 register.	0b0
[0]	PWRDN	Disable cluster power down when all cores are powered down. Note this bit is a combined version of all banked per-thread bits from the IMP_CLUSTERPWRDN_EL1 register.	0b0

Access

MRS <Xt>, S3_0_C15_C3_7

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b111

MSR S3_0_C15_C3_7, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b111

Accessibility

MRS <Xt>, S3_0_C15_C3_7

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPWRSTAT_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPWRSTAT_EL1;
elsif PSTATE.EL == EL3 then
```

```
    return IMP_CLUSTERPWRSTAT_EL1;
```

MSR S3_0_C15_C3_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRSTAT_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRSTAT_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPWRSTAT_EL1 = X[t];
```

A.1.9 IMP_CLUSTERL3DNTH0_EL1, Cluster L3 Downsize Threshold0 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-9: AArch64_imp_clusterl3dnth0_el1 bit assignments

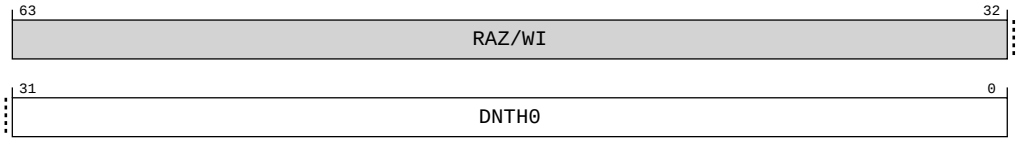


Table A-26: IMP_CLUSTERL3DNTH0_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	DNTH0	If all L3 ways are powered and the cache hit bandwidth falls below this threshold then the cache is downsized to half the ways. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b000

MSR S3_0_C15_C4_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b000

Accessibility

MRS <Xt>, S3_0_C15_C4_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3DNTH0_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3DNTH0_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERL3DNTH0_EL1;
```

MSR S3_0_C15_C4_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3DNTH0_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3DNTH0_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3DNTH0_EL1 = X[t];
```

A.1.10 IMP_CLUSTERL3DNTH1_EL1, Cluster L3 Downsize Threshold1 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-10: AArch64_imp_clusterl3dnth1_el1 bit assignments

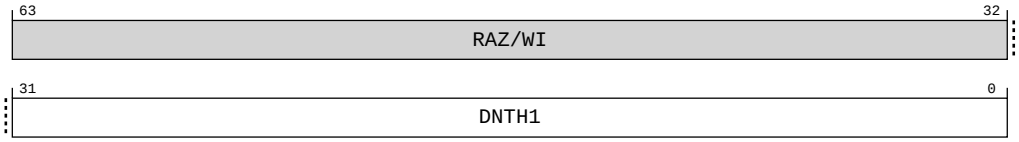


Table A-29: IMP_CLUSTERL3DNTH1_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	DNTH1	If half the L3 cache ways are powered and the L3 cache hit bandwidth falls below this threshold, then the L3 cache is downsized so that none of the ways are powered. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b001

MSR S3_0_C15_C4_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b001

Accessibility

MRS <Xt>, S3_0_C15_C4_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3DNTH1_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3DNTH1_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERL3DNTH1_EL1;
```

MSR S3_0_C15_C4_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
```

```
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3DNTH1_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3DNTH1_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3DNTH1_EL1 = X[t];
```

A.1.11 IMP_CLUSTERL3UPTH0_EL1, Cluster L3 Upsize Threshold0 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-11: AArch64_imp_clusterl3upth0_el1 bit assignments

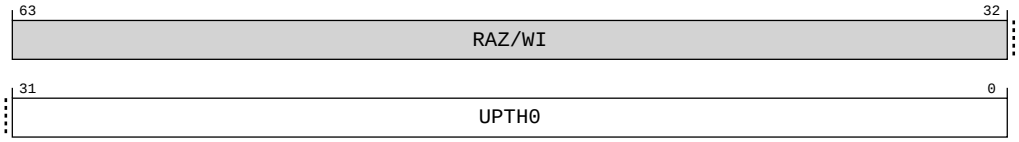


Table A-32: IMP_CLUSTERL3UPTH0_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	UPTH0	If no L3 ways are powered and the cache miss bandwidth rises above this threshold then the cache is upsized to half the ways. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b010

MSR S3_0_C15_C4_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b010

Accessibility

MRS <Xt>, S3_0_C15_C4_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3UPTH0_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3UPTH0_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERL3UPTH0_EL1;
```

MSR S3_0_C15_C4_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
```

```
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3UPTH0_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3UPTH0_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3UPTH0_EL1 = X[t];
```

A.1.12 IMP_CLUSTERL3UPTH1_EL1, Cluster L3 Upsize Threshold1 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-12: AArch64_imp_clusterl3upth1_el1 bit assignments

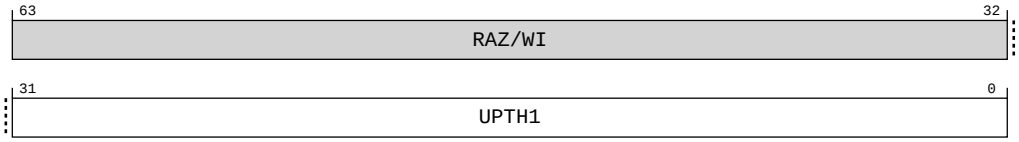


Table A-35: IMP_CLUSTERL3UPTH1_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	UPTH1	If half of the L3 cache ways are powered, and the L3 cache miss bandwidth rises above this threshold then the L3 cache is upsized to all of the ways. The value in this register is compared with the change in the cluster L3 miss counter since the last time period.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_3

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b011

MSR S3_0_C15_C4_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b011

Accessibility

MRS <Xt>, S3_0_C15_C4_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3UPTH1_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3UPTH1_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERL3UPTH1_EL1;
```

MSR S3_0_C15_C4_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
```

```
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3UPTH1_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3UPTH1_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3UPTH1_EL1 = X[t];
```

A.1.13 IMP_CLUSTERBUSQOS_EL1, Cluster Bus QoS Control Register

Determines the value driven on the CHI bus QoS field.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1011 1110

Bit descriptions

Figure A-13: AArch64_imp_clusterbusqos_el1 bit assignments

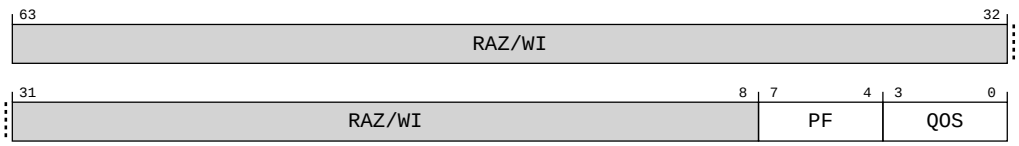


Table A-38: IMP_CLUSTERBUSQOS_EL1 bit descriptions

Bits	Name	Description	Reset
[63:8]	RAZ/WI	Reserved	RAZ/WI
[7:4]	PF	Valid driven on the CHI bus QoS field for prefetches.	0b1011
[3:0]	QOS	Valid driven on the CHI bus QoS field for demand accesses.	0b1110

Access

MRS <Xt>, S3_0_C15_C4_4

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b100

MSR S3_0_C15_C4_4, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b100

Accessibility

MRS <Xt>, S3_0_C15_C4_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERBUSQOS_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERBUSQOS_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERBUSQOS_EL1;
```

MSR S3_0_C15_C4_4, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.QOSEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.QOSEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.QOSEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERBUSQOS_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.QOSEN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.QOSEN == '0' then
```

```
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERBUSQOS_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERBUSQOS_EL1 = X[t];
```

A.1.14 IMP_CLUSTERL3HIT_EL1, Cluster L3 Hit Counter Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-14: AArch64_imp_clusterl3hit_el1 bit assignments

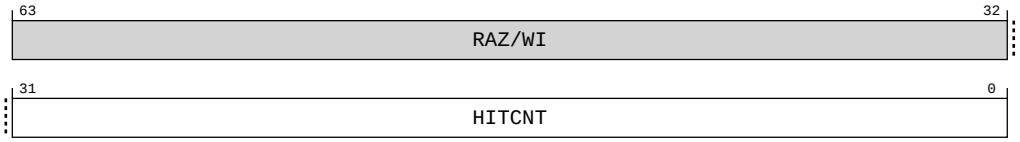


Table A-41: IMP_CLUSTERL3HIT_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	HITCNT	Count of number of L3 hits, for use in portion control calculations.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_5

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b101

MSR S3_0_C15_C4_5, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b101

Accessibility

MRS <Xt>, S3_0_C15_C4_5

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3HIT_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3HIT_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERL3HIT_EL1;
```

MSR S3_0_C15_C4_5, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3HIT_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3HIT_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3HIT_EL1 = X[t];
```

A.1.15 IMP_CLUSTERL3MISS_EL1, Cluster L3 Miss Counter Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-15: AArch64_imp_clusterl3miss_el1 bit assignments

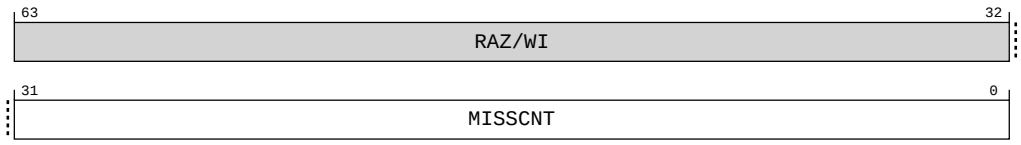


Table A-44: IMP_CLUSTERL3MISS_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	MISSCNT	Count of number of L3 misses, for use in portion control calculations.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_6

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b110

MSR S3_0_C15_C4_6, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b110

Accessibility

MRS <Xt>, S3_0_C15_C4_6

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3MISS_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3MISS_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERL3MISS_EL1;
```

MSR S3_0_C15_C4_6, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3MISS_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERL3MISS_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3MISS_EL1 = X[t];
```

A.1.16 IMP_CLUSTERPPSTART_EL1, Cluster peripheral port Start Address Register

Determines the start address for the peripheral port address range.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-16: AArch64_imp_clusterppstart_el1 bit assignments

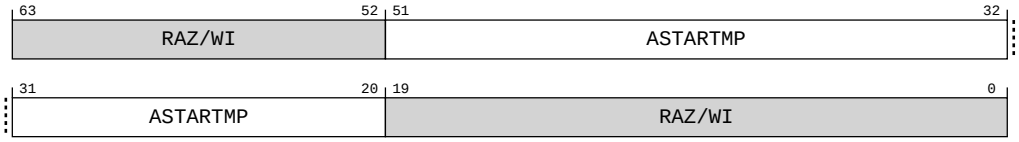


Table A-47: IMP_CLUSTERPPSTART_EL1 bit descriptions

Bits	Name	Description	Reset
[63:52]	RAZ/WI	Reserved	RAZ/WI
[51:20]	ASTARTMP	Start address for peripheral port address range.	32{x}
[19:0]	RAZ/WI	Reserved	RAZ/WI

Access

MRS <Xt>, S3_0_C15_C9_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b000

MSR S3_0_C15_C9_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b000

Accessibility

MRS <Xt>, S3_0_C15_C9_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPPSTART_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPPSTART_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPPSTART_EL1;
```

MSR S3_0_C15_C9_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPSTART_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPSTART_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPPSTART_EL1 = X[t];
```

A.1.17 IMP_CLUSTERPPEND_EL1, Cluster peripheral port End Address Register

Determines the end address for the peripheral port address range.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-17: AArch64_imp_clusterppend_el1 bit assignments

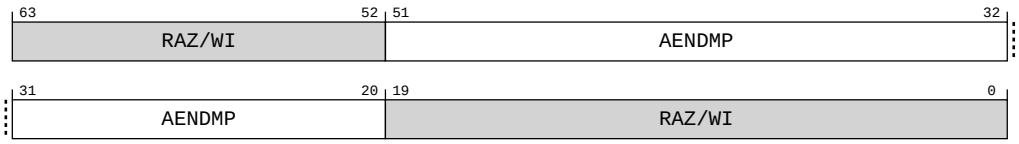


Table A-50: IMP_CLUSTERPPEND_EL1 bit descriptions

Bits	Name	Description	Reset
[63:52]	RAZ/WI	Reserved	RAZ/WI
[51:20]	AENDMP	End address for peripheral port address range. If the end address is the same as the start address then no accesses will be sent to the peripheral port. The end address is non-inclusive. The defined range is from the start address to the end address but excluding the byte at the end address.	32{x}
[19:0]	RAZ/WI	Reserved	RAZ/WI

Access

MRS <Xt>, S3_0_C15_C9_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b001

MSR S3_0_C15_C9_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b001

Accessibility

MRS <Xt>, S3_0_C15_C9_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPPEND_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPPEND_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPPEND_EL1;
```

MSR S3_0_C15_C9_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPEND_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPEND_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPPEND_EL1 = X[t];
```

A.1.18 IMP_CLUSTERCFR2_EL1, Cluster Configuration Register 2

Contains details of the hardware configuration of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-18: AArch64_imp_clustercfr2_el1 bit assignments

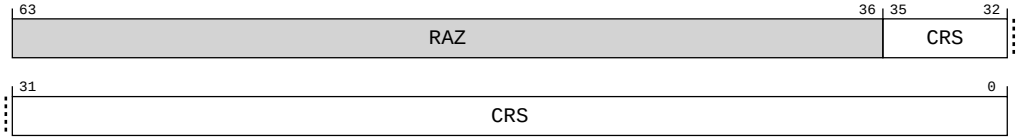


Table A-53: IMP_CLUSTERCFR2_EL1 bit descriptions

Bits	Name	Description	Reset
[63:36]	RAZ	Reserved	RAZ

Bits	Name	Description	Reset
[35:0]	CRS	Core register slices. Each three bits represents a core, with [2:0] for core 0 up to [35:33] for core 11. **0b000000000000000000000000000000000000** No register slices **0b000000000000000000000000000000000001** One register slice **0b000000000000000000000000000000000010** Two register slices **0b000000000000000000000000000000000011** Three register slices **0b000000000000000000000000000000000100** Four register slices **0b000000000000000000000000000000000101** Five register slices **0b000000000000000000000000000000000110** Six register slices **0b000000000000000000000000000000000111** Seven register slices	36{x}

Access

MRS <Xt>, S3_0_C15_C9_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b010

MSR S3_0_C15_C9_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b010

Accessibility

MRS <Xt>, S3_0_C15_C9_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERCFR2_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERCFR2_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERCFR2_EL1;
```

MSR S3_0_C15_C9_2, <Xt>

```
if PSTATE.EL == EL0 then
```

```
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERCFR2_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERCFR2_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERCFR2_EL1 = X[t];
```

A.1.19 IMP_CLUSTERCDBG_EL3, Cluster Cache Debug Register

Can be used to read the contents of the L3 cache RAMs and snoop filter RAMs. The register must be written with the information of which RAM is to be read. Then the same register should be read to read the contents of that RAM.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-19: AArch64_imp_clustercdbg_el3 bit assignments

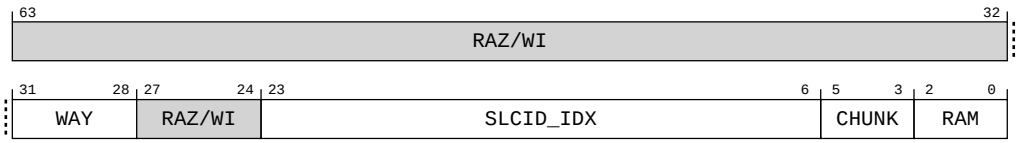


Table A-56: IMP_CLUSTERCDBG_EL3 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RAZ/WI**	Reserved	**RAZ/WI**
[31:28]	WAY	Way of RAM being accessed.	0b0000
[27:24]	**RAZ/WI**	Reserved	**RAZ/WI**

Bits	Name	Description	Reset
[23:6]	SLCID_IDX	The L3 cache Set locations in each cache slice are all power-of-2 in size and therefore can be identified using contiguous index locations. The Set index values for slice 0 start from value zero in this field, followed by the index locations for slice 1, then slice 2, and so on. The total index width varies depending on the size of the RAM being accessed. The cache slice identification number, Slice ID, forms the upper used bits of the cache location encoding in this field. For a Tag RAM or Data RAM access this field will encode as {'0, SLICE_ID_W, TagRAM_IDX_W} For a Snoop Filter RAM access this field will encode as {'0, SLICE_ID_W, SFRAM_IDX_W}.	0b00000000000000000
[5:3]	CHUNK	Select of 64-bit data chunk to read from 512-bit Data RAM cache line. Only used when accessing Data RAM data. **0b000** Data[63:0] **0b001** Data[127:64] **0b010** Data[191:128] **0b011** Data[255:192] **0b100** Data[319:256] **0b101** Data[383:320] **0b110** Data[447:384] **0b111** Data[511:448]	0b000
[2:0]	RAM	RAM to be accessed. All other values are reserved. **0b001** Snoop Filter RAM **0b010** Tag RAM **0b011** Data RAM - accessing cacheline data **0b111** Data RAM - accessing cacheline MTE tags	0b000

Access

MRS <Xt>, S3_6_C15_C4_7

op0	op1	CRn	CRm	op2
0b11	0b110	0b1111	0b0100	0b111

MSR S3_6_C15_C4_7, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b110	0b1111	0b0100	0b111

Accessibility

MRS <Xt>, S3_6_C15_C4_7

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERCDBG_EL3;
```

MSR S3_6_C15_C4_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERCDBG_EL3 = X[t];
```

A.1.20 IMP_CLUSTERPMMDCR_EL3, Monitor Debug Configuration Register (EL3)

Provides EL3 configuration options for self-hosted debug and the Performance Monitors Extension.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-20: AArch64_imp_clusterpmmdcr_el3 bit assignments

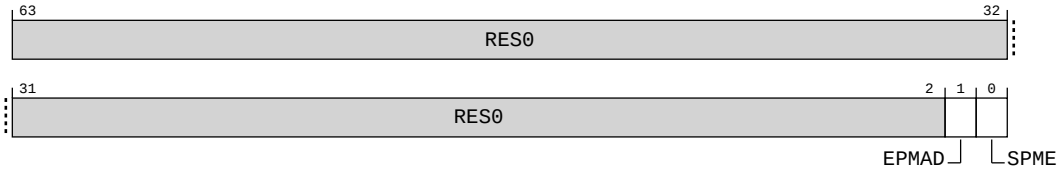


Table A-59: IMP_CLUSTERPMMDCR_EL3 bit descriptions

Bits	Name	Description	Reset
[63:2]	**RES0**	Reserved	**RES0**
[1]	EPMAD	External Performance Monitors Non-secure Access Disable. Controls Non-secure access to Performance Monitor registers by an external debugger. **0b0** Non-secure access to Performance Monitor registers from external debugger is permitted. **0b1** Non-secure access to Performance Monitor registers from external debugger is not permitted.	0b0
[0]	SPME	Secure Performance Monitors enable. This allows event counting in Secure state. **0b0** Event counting prohibited in Secure state. **0b1** Event counting in Secure state not affected by this bit.	0b0

Access

MRS <Xt>, S3_6_C15_C6_3

op0	op1	CRn	CRm	op2
0b11	0b110	0b1111	0b0110	0b011

MSR S3_6_C15_C6_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b110	0b1111	0b0110	0b011

Accessibility

MRS <Xt>, S3_6_C15_C6_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMMDCR_EL3;
```

MSR S3_6_C15_C6_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMMDCR_EL3 = X[t];
```

A.2 AArch64 Performance Monitors register summary

The summary table provides an overview of all Performance Monitors registers in the *DynamIQ™ Shared Unit-110* (DSU-110). Individual register descriptions provide detailed information.

Note

- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table A-62: Performance Monitors registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERPMCR_EL1	3	0	C15	C5	0	—	64-bit	Performance Monitors Control Register	No
IMP_CLUSTERPMCNTENSET_EL1	3	0	C15	C5	1	—	64-bit	Performance Monitors Count Enable Set Register	No

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERPMCNTENCLR_EL1	3	0	C15	C5	2	—	64-bit	Performance Monitors Count Enable Clear Register	No
IMP_CLUSTERPMOVSSET_EL1	3	0	C15	C5	3	—	64-bit	Performance Monitors Overflow Flag Status Set Register	No
IMP_CLUSTERPMOVSCLR_EL1	3	0	C15	C5	4	—	64-bit	Performance Monitors Overflow Flag Status Clear Register	No
IMP_CLUSTERPMSELR_EL1	3	0	C15	C5	5	—	64-bit	Performance Monitors Event Counter Selection Register	No
IMP_CLUSTERPMINTENSET_EL1	3	0	C15	C5	6	—	64-bit	Performance Monitors Interrupt Enable Set Register	No
IMP_CLUSTERPMINTENCLR_EL1	3	0	C15	C5	7	—	64-bit	Performance Monitors Interrupt Enable Clear Register	No
IMP_CLUSTERPMCCNTR_EL1	3	0	C15	C6	0	—	64-bit	Performance Monitors Cycle Count Register	No
IMP_CLUSTERPMXEVTYPER_EL1	3	0	C15	C6	1	—	64-bit	Performance Monitors Selected Event Type Register	No
IMP_CLUSTERPMXEVCNTR_EL1	3	0	C15	C6	2	—	64-bit	Performance Monitors Selected Event Count Register	No
IMP_CLUSTERPMCEID0_EL1	3	0	C15	C6	4	—	64-bit	Performance Monitors Common Event Identification Register 0	No
IMP_CLUSTERPMCEID1_EL1	3	0	C15	C6	5	—	64-bit	Performance Monitors Common Event Identification Register 1	No
IMP_CLUSTERPMCLAIMSET_EL1	3	0	C15	C6	6	—	64-bit	Claim Tag Set Register	No
IMP_CLUSTERPMCLAIMCLR_EL1	3	0	C15	C6	7	—	64-bit	Claim Tag Clear Register	No

A.2.1 IMP_CLUSTERPMCR_EL1, Performance Monitors Control Register

Provides details of the Performance Monitors implementation, including the number of counters implemented, and configures and controls the counters.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-21: AArch64_imp_clusterpmcr_el1 bit assignments

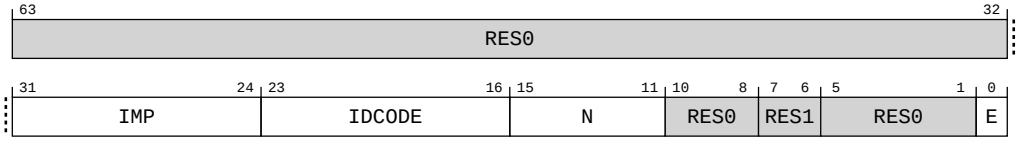


Table A-63: IMP_CLUSTERPMCR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31:24]	IMP	Implementer code. This field is RO with a value of 0 indicating that IMP_CLUSTERPMCR_EL1.IDCODE is **RES0** and software must use the AArch64-MIDR_EL1 to identify the PE. **0b00000000** Software must use the AArch64-MIDR_EL1 to identify the PE.	8{x}
[23:16]	IDCODE	Identification code. This field is RO with a value of 0x00. **0b00000000** Software must use the AArch64-MIDR_EL1 to identify the PE.	8{x}
[15:11]	N	A Read Only field that indicates the number of event counters implemented. **0b00110** Indicates that 6 event counters are implemented. Access to this field is: RO	5{x}
[10:8]	**RES0**	Reserved	**RES0**
[7:6]	**RES1**	Reserved	**RES1**
[5:1]	**RES0**	Reserved	**RES0**
[0]	E	Enable. **0b0** All event counters in the range [0..(PMN-1)] and AArch64-IMP_CLUSTERPMCCNTR_EL1, are disabled. **0b1** All event counters in the range [0..(PMN-1)] and AArch64-IMP_CLUSTERPMCCNTR_EL1, are enabled by AArch64-IMP_CLUSTERPMCNTENSET_EL1.	0b0

Access

MRS <Xt>, S3_0_C15_C5_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b000

MSR S3_0_C15_C5_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b000

Accessibility

MRS <Xt>, S3_0_C15_C5_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCR_EL1;
```

MSR S3_0_C15_C5_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCR_EL1 = X[t];
```

A.2.2 IMP_CLUSTERPMCNTENSET_EL1, Performance Monitors Count Enable Set Register

Enables the Cycle Count Register, AArch64-IMP_CLUSTERPMCCNTR_EL1, and all implemented event counters AArch64-IMP_CLUSTERPMXEVCNTR_EL1.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-22: AArch64_imp_clusterpmcntenset_el1 bit assignments

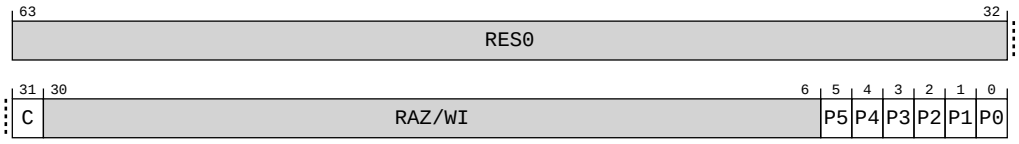


Table A-66: IMP_CLUSTERPMCNTENSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31]	C	AArch64-IMP_CLUSTERPMCCNTR_EL1 enable bit. Enables the cycle counter register. **0b0** When read, means the cycle counter is disabled. When written, has no effect. **0b1** When read, means the cycle counter is enabled. When written, enables the cycle counter.	x

Bits	Name	Description	Reset
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[4]	P4	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[3]	P3	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[2]	P2	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[1]	P1	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[0]	P0	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x

Access

MRS <Xt>, S3_0_C15_C5_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b001

MSR S3_0_C15_C5_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b001

Accessibility

MRS <Xt>, S3_0_C15_C5_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCNTENSET_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCNTENSET_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCNTENSET_EL1;
```

MSR S3_0_C15_C5_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENSET_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENSET_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCNTENSET_EL1 = X[t];
```

A.2.3 IMP_CLUSTERPMCNTENCLR_EL1, Performance Monitors Count Enable Clear Register

Disables the Cycle Count Register, AArch64-IMP_CLUSTERPMCCNTR_EL1, and all implemented event counters AArch64-IMP_CLUSTERPMXEVCNTR_EL1.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-23: AArch64_imp_clusterpmcntenclr_el1 bit assignments

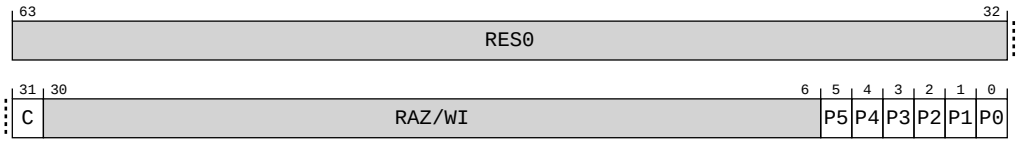


Table A-69: IMP_CLUSTERPMCNTENCLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31]	C	AArch64-IMP_CLUSTERPMCCNTR_EL1 disable bit. Disables the cycle counter register. Possible values are: **0b0** When read, means the cycle counter is disabled. When written, has no effect. **0b1** When read, means the cycle counter is enabled. When written, disables the cycle counter.	x

Bits	Name	Description	Reset
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[4]	P4	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[3]	P3	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[2]	P2	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[1]	P1	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[0]	P0	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x

Access

MRS <Xt>, S3_0_C15_C5_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b010

MSR S3_0_C15_C5_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b010

Accessibility

MRS <Xt>, S3_0_C15_C5_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCNTENCLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCNTENCLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCNTENCLR_EL1;
```

MSR S3_0_C15_C5_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENCLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENCLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCNTENCLR_EL1 = X[t];
```

A.2.4 IMP_CLUSTERPMOVSSET_EL1, Performance Monitors Overflow Flag Status Set Register

Sets the state of the overflow bit for the Cycle Count Register, AArch64-IMP_CLUSTERPMCCNTR_EL1, and each of the implemented event counters AArch64-PMXEVCNTR_EL1.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-24: AArch64_imp_clusterpmovsset_el1 bit assignments

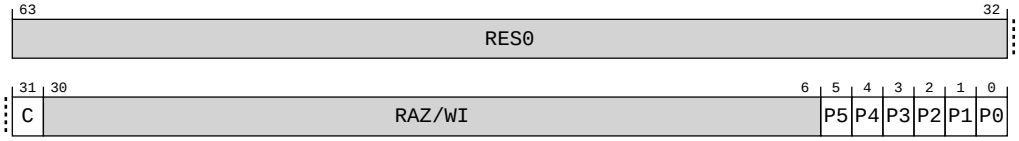


Table A-72: IMP_CLUSTERPMOVSSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[31]	C	Cycle counter overflow set bit. **0b0** When read, means the cycle counter has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means the cycle counter has overflowed since this bit was last cleared. When written, sets the cycle counter overflow bit to 1. AArch64-IMP_CLUSTERPMCR_EL1.LC controls whether an overflow is detected from unsigned overflow of AArch64-IMP_CLUSTERPMCCNTR_EL1[31:0] or unsigned overflow of AArch64-IMP_CLUSTERPMCCNTR_EL1[63:0].	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[4]	P4	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[3]	P3	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[2]	P2	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x

Bits	Name	Description	Reset
[1]	P1	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[0]	P0	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x

Access

MRS <Xt>, S3_0_C15_C5_3

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b011

MSR S3_0_C15_C5_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b011

Accessibility

MRS <Xt>, S3_0_C15_C5_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMOVSSET_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMOVSSET_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMOVSSET_EL1;
```

MSR S3_0_C15_C5_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
```

```
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSSET_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSSET_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMOVSSET_EL1 = X[t];
```

A.2.5 IMP_CLUSTERPMOVSCLR_EL1, Performance Monitors Overflow Flag Status Clear Register

Contains the state of the overflow bit for the Cycle Count Register, AArch64-IMP_CLUSTERPMCCNTR_EL1, and each of the implemented event counters AArch64-PMXEVCNTR_EL1. Writing to this register clears these bits.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000 0000 0000 0000 0000 0000 00xx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-25: AArch64_imp_clusterpmovsclr_el1 bit assignments

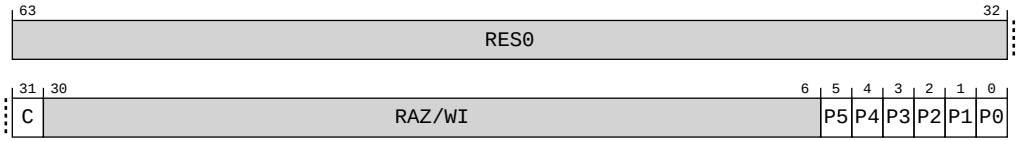


Table A-75: IMP_CLUSTERPMOVSCLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	C	Cycle counter overflow clear bit. **0b0** When read, means the cycle counter has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means the cycle counter has overflowed since this bit was last cleared. When written, clears the cycle counter overflow bit to 0. AArch64-IMP_CLUSTERPMCR_EL1.LC controls whether an overflow is detected from unsigned overflow of AArch64-IMP_CLUSTERPMCCNTR_EL1[31:0] or unsigned overflow of AArch64-IMP_CLUSTERPMCCNTR_EL1[63:0].	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[4]	P4	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x

Bits	Name	Description	Reset
[3]	P3	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[2]	P2	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[1]	P1	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[0]	P0	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x

Access

MRS <Xt>, S3_0_C15_C5_4

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b100

MSR S3_0_C15_C5_4, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b100

Accessibility

MRS <Xt>, S3_0_C15_C5_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMOVSCLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMOVSCLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMOVSCLR_EL1;
```

MSR S3_0_C15_C5_4, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSCLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSCLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMOVSCLR_EL1 = X[t];
```

A.2.6 IMP_CLUSTERPMSELR_EL1, Performance Monitors Event Counter Selection Register

Selects the current event counter AArch64-IMP_CLUSTERPMEVCNTR_EL1 or the cycle counter, CCNT.

IMP_CLUSTERPMSELR_EL1 is used in conjunction with AArch64-IMP_CLUSTERPMXEVTYPER_EL1 to determine the event that increments a selected event counter, and the modes and states in which the selected counter increments.

It is also used in conjunction with AArch64-IMP_CLUSTERPMXEVCNTR_EL1, to determine the value of a selected event counter.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-26: AArch64_imp_clusterpmselr_el1 bit assignments

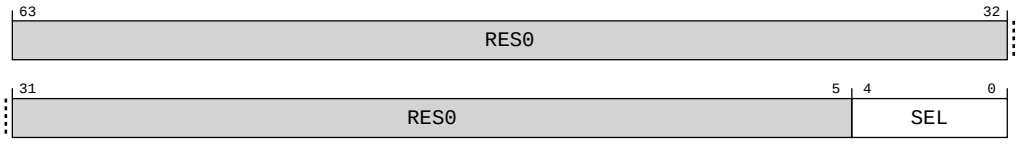


Table A-78: IMP_CLUSTERPMSELR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:5]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[4:0]	SEL	Selects event counter, AArch64-IMP_CLUSTERPMXEVCNTR_EL1, where n is the value held in this field. This value identifies which event counter is accessed when a subsequent access to AArch64-IMP_CLUSTERPMXEVTYPER_EL1 or AArch64-IMP_CLUSTERPMXEVCNTR_EL1 occurs. This field can take any value from 0 (0b00000) to (PMCR.N)-1. When IMP_CLUSTERPMSELR_EL1.SEL is 0b11111, it selects the cycle counter and: • A read or write of AArch64-IMP_CLUSTERPMXEVTYPER_EL1 is treated as **RAZ/WI**. • A read or write of AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is treated as **RAZ/WI**. If this field is set to a value greater than or equal to the number of counters accessible at the current Exception level, but not equal to 31: • Direct reads of this field are treated as **RAZ/WI**. • The results of access to AArch64-IMP_CLUSTERPMXEVTYPER_EL1 or AArch64-IMP_CLUSTERPMXEVCNTR_EL1 are treated as **RAZ/WI**.	5{x}

Access

MRS <Xt>, S3_0_C15_C5_5

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b101

MSR S3_0_C15_C5_5, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b101

Accessibility

MRS <Xt>, S3_0_C15_C5_5

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMSELR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMSELR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMSELR_EL1;
```

MSR S3_0_C15_C5_5, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
```

```
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMSELR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMSELR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMSELR_EL1 = X[t];
```

A.2.7 IMP_CLUSTERPMINTENSET_EL1, Performance Monitors Interrupt Enable Set Register

Enables the generation of interrupt requests on overflows from the Cycle Count Register, AArch64-IMP_CLUSTERPMCCNTR_EL1, and the event counters AArch64-IMP_CLUSTERPMXEVCNTR_EL1.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000 0000 0000 0000 0000 0000 00xx xxxx


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-27: AArch64_imp_clusterpmintenset_el1 bit assignments

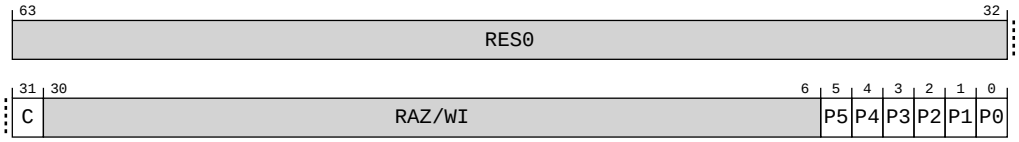


Table A-81: IMP_CLUSTERPMINTENSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	C	AArch64-IMP_CLUSTERPMCCNTR_EL1 overflow interrupt request enable bit. **0b0** When read, means the cycle counter overflow interrupt request is disabled. When written, has no effect. **0b1** When read, means the cycle counter overflow interrupt request is enabled. When written, enables the cycle count overflow interrupt request.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[4]	P4	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[3]	P3	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x

Bits	Name	Description	Reset
[2]	P2	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[1]	P1	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[0]	P0	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x

Access

MRS <Xt>, S3_0_C15_C5_6

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b110

MSR S3_0_C15_C5_6, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b110

Accessibility

MRS <Xt>, S3_0_C15_C5_6

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMINTENSET_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMINTENSET_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMINTENSET_EL1;
```

MSR S3_0_C15_C5_6, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENSET_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENSET_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMINTENSET_EL1 = X[t];
```

A.2.8 IMP_CLUSTERPMINTENCLR_EL1, Performance Monitors Interrupt Enable Clear Register

Disables the generation of interrupt requests on overflows from the Cycle Count Register, AArch64-IMP_CLUSTERPMCCNTR_EL1, and the event counters AArch64-IMP_CLUSTERPMXEVCNTR_EL1.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-28: AArch64_imp_clusterpmintenclr_el1 bit assignments

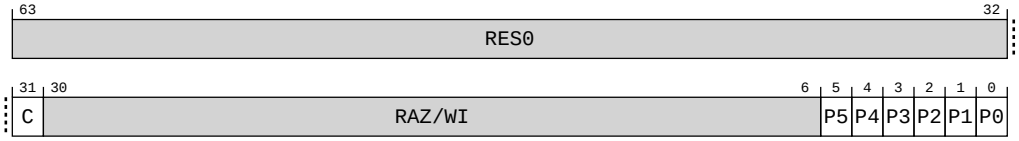


Table A-84: IMP_CLUSTERPMINTENCLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	C	AArch64-IMP_CLUSTERPMCCNTR_EL1 overflow interrupt request disable bit. **0b0** When read, means the cycle counter overflow interrupt request is disabled. When written, has no effect. **0b1** When read, means the cycle counter overflow interrupt request is enabled. When written, disables the cycle count overflow interrupt request.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[4]	P4	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x

Bits	Name	Description	Reset
[3]	P3	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[2]	P2	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[1]	P1	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[0]	P0	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x

Access

MRS <Xt>, S3_0_C15_C5_7

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b111

MSR S3_0_C15_C5_7, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b111

Accessibility

MRS <Xt>, S3_0_C15_C5_7

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
```

```
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMINTENCLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMINTENCLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMINTENCLR_EL1;
```

MSR S3_0_C15_C5_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENCLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENCLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMINTENCLR_EL1 = X[t];
```

A.2.9 IMP_CLUSTERPMCCNTR_EL1, Performance Monitors Cycle Count Register

Holds the value of the processor Cycle Counter, CCNT, that counts processor clock cycles.

Configurations

All counters are subject to any changes in clock frequency, including clock stopping caused by the WFI and WFE instructions.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-29: AArch64_imp_clusterpmccntr_el1 bit assignments

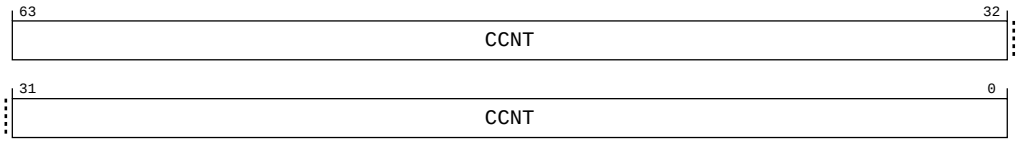


Table A-87: IMP_CLUSTERPMCCNTR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	CCNT	Cycle count. Depending on the values of AArch64-PMCR_EL1.{LC,D}, this field increments in one of the following ways: • Every processor clock cycle. • Every 64th processor clock cycle. Writing 1 to AArch64-PMCR_EL1.C sets this field to 0.	64{x}

Access

MRS <Xt>, S3_0_C15_C6_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b000

MSR S3_0_C15_C6_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b000

Accessibility

MRS <Xt>, S3_0_C15_C6_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCCNTR_EL1;
```

```
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCCNTR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCCNTR_EL1;
```

MSR S3_0_C15_C6_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCCNTR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCCNTR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCCNTR_EL1 = X[t];
```

A.2.10 IMP_CLUSTERPMXEVTYPER_EL1, Performance Monitors Selected Event Type Register

When AArch64-IMP_CLUSTERPMSELR_EL1.SEL selects an event counter, this accesses a AArch64-IMP_CLUSTERPMXEVTYPER_EL1 register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-30: AArch64_imp_clusterpmxevtyper_el1 bit assignments

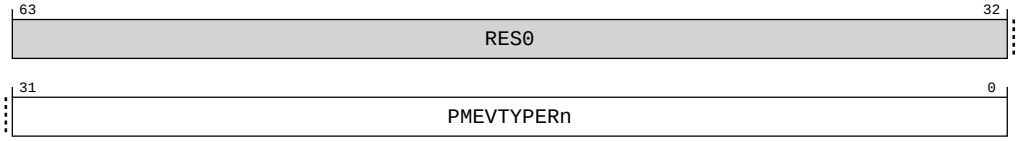


Table A-90: IMP_CLUSTERPMXEVTYPER_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31:0]	PMEVTYPERn	This register accesses AArch64-IMP_CLUSTERPMXEVTYPER_EL1 where n is the value in AArch64-IMP_CLUSTERPMSELR_EL1.SEL.	32{x}

Access

MRS <Xt>, S3_0_C15_C6_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b001

MSR S3_0_C15_C6_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b001

Accessibility

MRS <Xt>, S3_0_C15_C6_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMXEVTYPER_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMXEVTYPER_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMXEVTYPER_EL1;
```

MSR S3_0_C15_C6_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVTYPER_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVTYPER_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMXEVTYPER_EL1 = X[t];
```

A.2.11 IMP_CLUSTERPMXEVCNTR_EL1, Performance Monitors Selected Event Count Register

Reads or writes the value of the selected event counter, AArch64-IMP_CLUSTERPMXEVCNTR_EL1. AArch64-IMP_CLUSTERPMSELR_EL1.SEL determines which event counter is selected.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-31: AArch64_imp_clusterpmxevcntr_el1 bit assignments

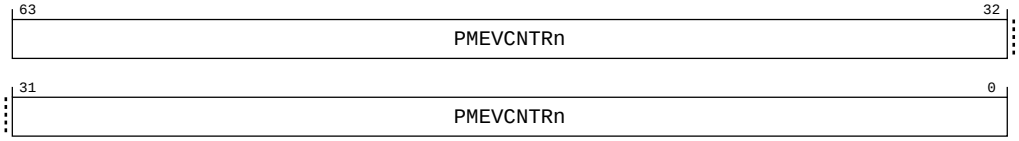


Table A-93: IMP_CLUSTERPMXEVCNTR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTRn	Value of the selected event counter, AArch64-IMP_CLUSTERPMXEVCNTR_EL1, where n is the value stored in AArch64-IMP_CLUSTERPMSELR_EL1.SEL.	64{x}

Access

MRS <Xt>, S3_0_C15_C6_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b010

MSR S3_0_C15_C6_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b010

Accessibility

MRS <Xt>, S3_0_C15_C6_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMXEVCNTR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMXEVCNTR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMXEVCNTR_EL1;
```

MSR S3_0_C15_C6_2, <Xt>

```
if PSTATE.EL == EL0 then
```

```
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVCNTR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVCNTR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMXEVCNTR_EL1 = X[t];
```

A.2.12 IMP_CLUSTERPMCEID0_EL1, Performance Monitors Common Event Identification Register 0

Defines which common architectural events and common microarchitectural events are implemented, or counted, using PMU events in the ranges 0x0000 to 0x001F and 0x4000 to 0x401F.

When the value of a bit in the register is 1 the corresponding common event is implemented and counted.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0001 1110 0000 0000 0010 0110 0000 0010 0000 0000 0000 0000

Bit descriptions

Figure A-32: AArch64_imp_clusterpmceid0_el1 bit assignments

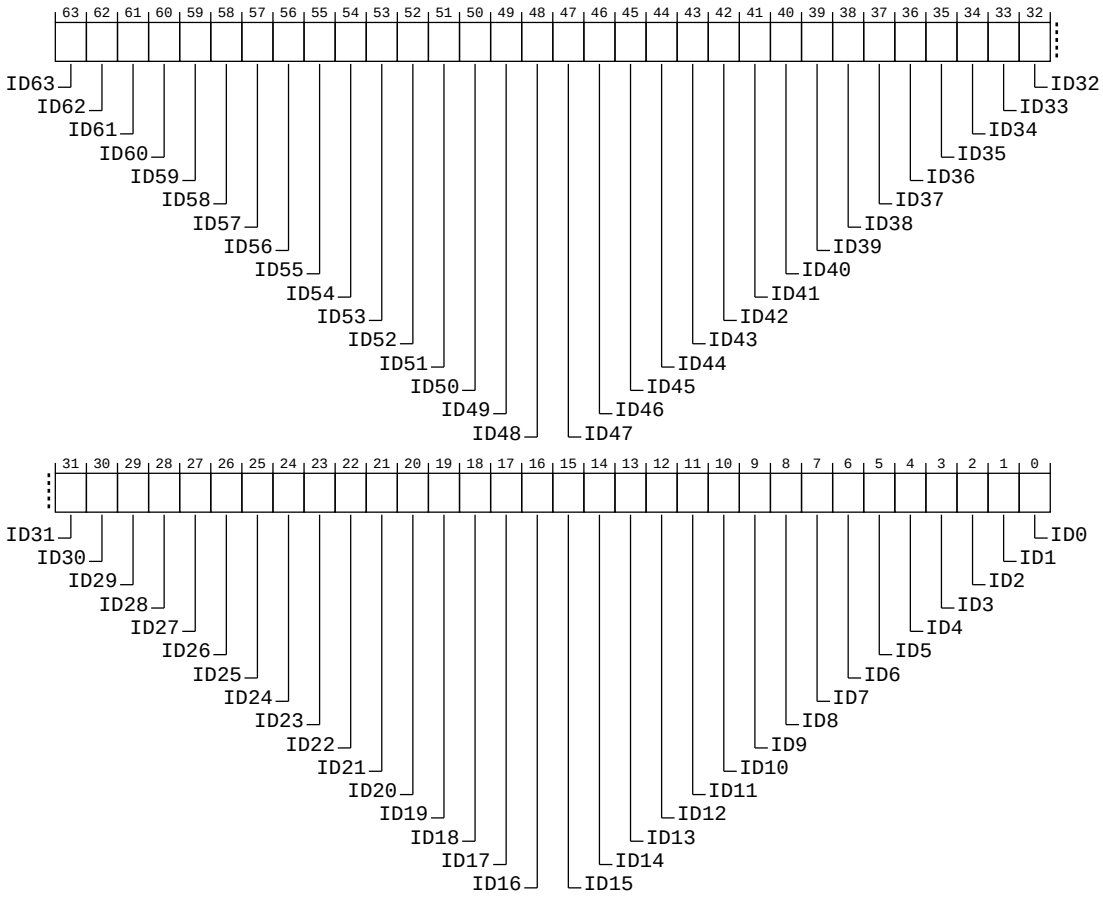


Table A-96: IMP_CLUSTERPMCEID0_EL1 bit descriptions

Bits	Name	Description	Reset
[63]	ID63	Common event `0x003F` implemented. **0b0** Event 0x003F not implemented.	`0b0`
[62]	ID62	Common event `0x003E` implemented. **0b0** Event 0x003E not implemented.	`0b0`
[61]	ID61	Common event `0x003D` implemented. **0b0** Event 0x003D not implemented.	`0b0`
[60]	ID60	Common event `0x003C` implemented. **0b0** Event 0x003C not implemented.	`0b0`

Bits	Name	Description	Reset
[59]	ID59	Common event `0x003B` implemented. **0b0** Event 0x003B not implemented.	`0b0`
[58]	ID58	Common event `0x003A` implemented. **0b0** Event 0x003A not implemented.	`0b0`
[57]	ID57	Common event `0x0039` implemented. **0b0** Event 0x0039 not implemented.	`0b0`
[56]	ID56	Common event `0x0038` implemented. **0b0** Event 0x0038 not implemented.	`0b0`
[55]	ID55	Common event `0x0037` implemented. **0b0** Event 0x0037 not implemented.	`0b0`
[54]	ID54	Common event `0x0036` implemented. **0b0** Event 0x0036 not implemented.	`0b0`
[53]	ID53	Common event `0x0035` implemented. **0b0** Event 0x0035 not implemented.	`0b0`
[52]	ID52	Common event `0x0034` implemented. **0b0** Event 0x0034 not implemented.	`0b0`
[51]	ID51	Common event `0x0033` implemented. **0b0** Event 0x0033 not implemented.	`0b0`
[50]	ID50	Common event `0x0032` implemented. **0b0** Event 0x0032 not implemented.	`0b0`
[49]	ID49	Common event `0x0031` implemented. **0b0** Event 0x0031 not implemented.	`0b0`
[48]	ID48	Common event `0x0030` implemented. **0b0** Event 0x0030 not implemented.	`0b0`
[47]	ID47	Common event `0x002F` implemented. **0b0** Event 0x002F not implemented.	`0b0`
[46]	ID46	Common event `0x002E` implemented. **0b0** Event 0x002E not implemented.	`0b0`

Bits	Name	Description	Reset
[45]	ID45	Common event `0x002D` implemented. **0b0** Event 0x002D not implemented.	`0b0`
[44]	ID44	Common event `0x002C` implemented. **0b1** L3D_CACHE_WB event implemented.	`0b1`
[43]	ID43	Common event `0x002B` implemented. **0b1** L3D_CACHE event implemented.	`0b1`
[42]	ID42	Common event `0x002A` implemented. **0b1** L3D_CACHE_REFILL event implemented.	`0b1`
[41]	ID41	Common event `0x0029` implemented. **0b1** L3D_CACHE_ALLOCATE event implemented.	`0b1`
[40]	ID40	Common event `0x0028` implemented. **0b0** Event 0x0028 not implemented.	`0b0`
[39]	ID39	Common event `0x0027` implemented. **0b0** Event 0x0027 not implemented.	`0b0`
[38]	ID38	Common event `0x0026` implemented. **0b0** Event 0x0026 not implemented.	`0b0`
[37]	ID37	Common event `0x0025` implemented. **0b0** Event 0x0025 not implemented.	`0b0`
[36]	ID36	Common event `0x0024` implemented. **0b0** Event 0x0024 not implemented.	`0b0`
[35]	ID35	Common event `0x0023` implemented. **0b0** Event 0x0023 not implemented.	`0b0`
[34]	ID34	Common event `0x0022` implemented. **0b0** Event 0x0022 not implemented.	`0b0`
[33]	ID33	Common event `0x0021` implemented. **0b0** Event 0x0021 not implemented.	`0b0`
[32]	ID32	Common event `0x0020` implemented. **0b0** Event 0x0020 not implemented.	`0b0`

Bits	Name	Description	Reset
[31]	ID31	Common event `0x001F` implemented. **0b0** Event 0x001F not implemented.	`0b0`
[30]	ID30	Common event `0x001E` implemented. **0b0** CHAIN event implemented.	`0b0`
[29]	ID29	Common event `0x001D` implemented. **0b1** BUS_CYCLES event implemented.	`0b1`
[28]	ID28	Common event `0x001C` implemented. **0b0** Event 0x001C not implemented.	`0b0`
[27]	ID27	Common event `0x001B` implemented. **0b0** Event 0x001B not implemented.	`0b0`
[26]	ID26	Common event `0x001A` implemented. **0b1** MEMORY_ERROR event implemented.	`0b1`
[25]	ID25	Common event `0x0019` implemented. **0b1** BUS_ACCESS event implemented.	`0b1`
[24]	ID24	Common event `0x0018` implemented. **0b0** Event 0x0018 not implemented.	`0b0`
[23]	ID23	Common event `0x0017` implemented. **0b0** Event 0x0017 not implemented.	`0b0`
[22]	ID22	Common event `0x0016` implemented. **0b0** Event 0x0016 not implemented.	`0b0`
[21]	ID21	Common event `0x0015` implemented. **0b0** Event 0x0015 not implemented.	`0b0`
[20]	ID20	Common event `0x0014` implemented. **0b0** Event 0x0014 not implemented.	`0b0`
[19]	ID19	Common event `0x0013` implemented. **0b0** Event 0x0013 not implemented.	`0b0`
[18]	ID18	Common event `0x0012` implemented. **0b0** Event 0x0012 not implemented.	`0b0`

Bits	Name	Description	Reset
[17]	ID17	Common event `0x0011` implemented. **0b1** CYCLES event implemented.	`0b1`
[16]	ID16	Common event `0x0010` implemented. **0b0** Event 0x0010 not implemented.	`0b0`
[15]	ID15	Common event `0x000F` implemented. **0b0** Event 0x000F not implemented.	`0b0`
[14]	ID14	Common event `0x000E` implemented. **0b0** Event 0x000E not implemented.	`0b0`
[13]	ID13	Common event `0x000D` implemented. **0b0** Event 0x000D not implemented.	`0b0`
[12]	ID12	Common event `0x000C` implemented. **0b0** Event 0x000C not implemented.	`0b0`
[11]	ID11	Common event `0x000B` implemented. **0b0** Event 0x000B not implemented.	`0b0`
[10]	ID10	Common event `0x000A` implemented. **0b0** Event 0x000A not implemented.	`0b0`
[9]	ID9	Common event `0x0009` implemented. **0b0** Event 0x0009 not implemented.	`0b0`
[8]	ID8	Common event `0x0008` implemented. **0b0** Event 0x0008 not implemented.	`0b0`
[7]	ID7	Common event `0x0007` implemented. **0b0** Event 0x0007 not implemented.	`0b0`
[6]	ID6	Common event `0x0006` implemented. **0b0** Event 0x0006 not implemented.	`0b0`
[5]	ID5	Common event `0x0005` implemented. **0b0** Event 0x0005 not implemented.	`0b0`
[4]	ID4	Common event `0x0004` implemented. **0b0** Event 0x0004 not implemented.	`0b0`

Bits	Name	Description	Reset
[3]	ID3	Common event `0x0003` implemented. **0b0** Event 0x0003 not implemented.	`0b0`
[2]	ID2	Common event `0x0002` implemented. **0b0** Event 0x0002 not implemented.	`0b0`
[1]	ID1	Common event `0x0001` implemented. **0b0** Event 0x0001 not implemented.	`0b0`
[0]	ID0	Common event `0x0000` implemented. **0b0** Event 0x0000 not implemented.	`0b0`

Access

MRS <Xt>, S3_0_C15_C6_4

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b100

MSR S3_0_C15_C6_4, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b100

Accessibility

MRS <Xt>, S3_0_C15_C6_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCEID0_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCEID0_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCEID0_EL1;
```

MSR S3_0_C15_C6_4, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID0_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID0_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCEID0_EL1 = X[t];
```

A.2.13 IMP_CLUSTERPMCEID1_EL1, Performance Monitors Common Event Identification Register 1

Defines which common architectural events and common microarchitectural events are implemented, or counted, using PMU events in the ranges 0x0020 to 0x003F and 0x4020 to 0x403F.

When the value of a bit in the register is 1 the corresponding common event is implemented and counted.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-33: AArch64_imp_clusterpmceid1_el1 bit assignments

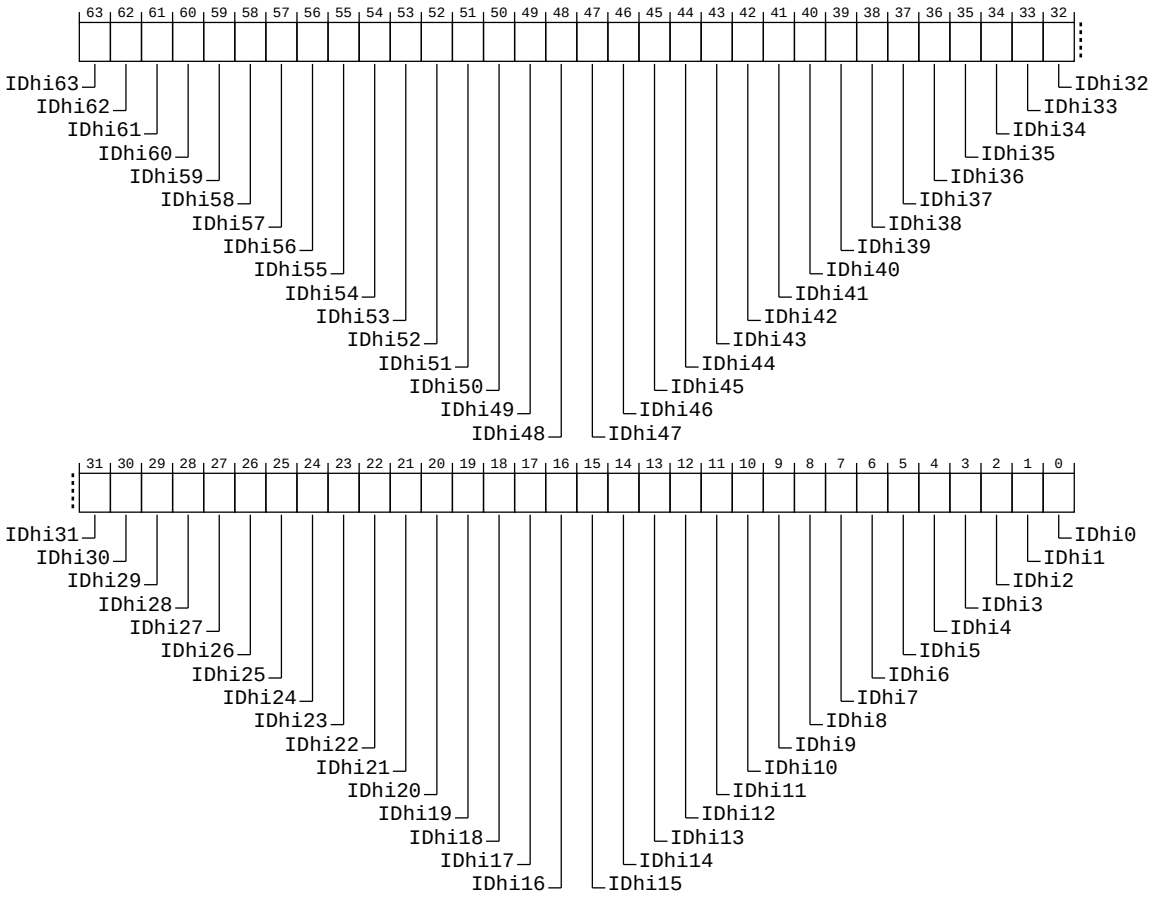


Table A-99: IMP_CLUSTERPMCEID1_EL1 bit descriptions

Bits	Name	Description	Reset
[63]	IDhi63	Common event `0x403F` implemented. **0b0** Event 0x403F not implemented.	`0b0`
[62]	IDhi62	Common event `0x403E` implemented. **0b0** Event 0x403E not implemented.	`0b0`
[61]	IDhi61	Common event `0x403D` implemented. **0b0** Event 0x403D not implemented.	`0b0`
[60]	IDhi60	Common event `0x403C` implemented. **0b0** Event 0x403C not implemented.	`0b0`

Bits	Name	Description	Reset
[59]	IDhi59	Common event `0x403B` implemented. **0b0** Event 0x403B not implemented.	`0b0`
[58]	IDhi58	Common event `0x403A` implemented. **0b0** Event 0x403A not implemented.	`0b0`
[57]	IDhi57	Common event `0x4039` implemented. **0b0** Event 0x4039 not implemented.	`0b0`
[56]	IDhi56	Common event `0x4038` implemented. **0b0** Event 0x4038 not implemented.	`0b0`
[55]	IDhi55	Common event `0x4037` implemented. **0b0** Event 0x4037 not implemented.	`0b0`
[54]	IDhi54	Common event `0x4036` implemented. **0b0** Event 0x4036 not implemented.	`0b0`
[53]	IDhi53	Common event `0x4035` implemented. **0b0** Event 0x4035 not implemented.	`0b0`
[52]	IDhi52	Common event `0x4034` implemented. **0b0** Event 0x4034 not implemented.	`0b0`
[51]	IDhi51	Common event `0x4033` implemented. **0b0** Event 0x4033 not implemented.	`0b0`
[50]	IDhi50	Common event `0x4032` implemented. **0b0** Event 0x4032 not implemented.	`0b0`
[49]	IDhi49	Common event `0x4031` implemented. **0b0** Event 0x4031 not implemented.	`0b0`
[48]	IDhi48	Common event `0x4030` implemented. **0b0** Event 0x4030 not implemented.	`0b0`
[47]	IDhi47	Common event `0x402F` implemented. **0b0** Event 0x402F not implemented.	`0b0`
[46]	IDhi46	Common event `0x402E` implemented. **0b0** Event 0x402E not implemented.	`0b0`

Bits	Name	Description	Reset
[45]	IDhi45	Common event `0x402D` implemented. **0b0** Event 0x402D not implemented.	`0b0`
[44]	IDhi44	Common event `0x402C` implemented. **0b0** Event 0x402C not implemented.	`0b0`
[43]	IDhi43	Common event `0x402B` implemented. **0b0** Event 0x402B not implemented.	`0b0`
[42]	IDhi42	Common event `0x402A` implemented. **0b0** Event 0x402A not implemented.	`0b0`
[41]	IDhi41	Common event `0x4029` implemented. **0b0** Event 0x4029 not implemented.	`0b0`
[40]	IDhi40	Common event `0x4028` implemented. **0b0** Event 0x4028 not implemented.	`0b0`
[39]	IDhi39	Common event `0x4027` implemented. **0b0** Event 0x4027 not implemented.	`0b0`
[38]	IDhi38	Common event `0x4026` implemented. **0b0** Event 0x4026 not implemented.	`0b0`
[37]	IDhi37	Common event `0x4025` implemented. **0b0** Event 0x4025 not implemented.	`0b0`
[36]	IDhi36	Common event `0x4024` implemented. **0b0** Event 0x4024 not implemented.	`0b0`
[35]	IDhi35	Common event `0x4023` implemented. **0b0** Event 0x4023 not implemented.	`0b0`
[34]	IDhi34	Common event `0x4022` implemented. **0b0** Event 0x4022 not implemented.	`0b0`
[33]	IDhi33	Common event `0x4021` implemented. **0b0** Event 0x4021 not implemented.	`0b0`
[32]	IDhi32	Common event `0x4020` implemented. **0b0** Event 0x4020 not implemented.	`0b0`

Bits	Name	Description	Reset
[31]	IDhi31	Common event `0x401F` implemented. **0b0** Event 0x401F not implemented.	`0b0`
[30]	IDhi30	Common event `0x401E` implemented. **0b0** Event 0x401E not implemented.	`0b0`
[29]	IDhi29	Common event `0x401D` implemented. **0b0** Event 0x401D not implemented.	`0b0`
[28]	IDhi28	Common event `0x401C` implemented. **0b0** Event 0x401C not implemented.	`0b0`
[27]	IDhi27	Common event `0x401B` implemented. **0b0** Event 0x401B not implemented.	`0b0`
[26]	IDhi26	Common event `0x401A` implemented. **0b0** Event 0x401A not implemented.	`0b0`
[25]	IDhi25	Common event `0x4019` implemented. **0b0** Event 0x4019 not implemented.	`0b0`
[24]	IDhi24	Common event `0x4018` implemented. **0b0** Event 0x4018 not implemented.	`0b0`
[23]	IDhi23	Common event `0x4017` implemented. **0b0** Event 0x4017 not implemented.	`0b0`
[22]	IDhi22	Common event `0x4016` implemented. **0b0** Event 0x4016 not implemented.	`0b0`
[21]	IDhi21	Common event `0x4015` implemented. **0b0** Event 0x4015 not implemented.	`0b0`
[20]	IDhi20	Common event `0x4014` implemented. **0b0** Event 0x4014 not implemented.	`0b0`
[19]	IDhi19	Common event `0x4013` implemented. **0b0** Event 0x4013 not implemented.	`0b0`
[18]	IDhi18	Common event `0x4012` implemented. **0b0** Event 0x4012 not implemented.	`0b0`

Bits	Name	Description	Reset
[17]	IDhi17	Common event `0x4011` implemented. **0b0** Event 0x4011 not implemented.	`0b0`
[16]	IDhi16	Common event `0x4010` implemented. **0b0** Event 0x4010 not implemented.	`0b0`
[15]	IDhi15	Common event `0x400F` implemented. **0b0** Event 0x400F not implemented.	`0b0`
[14]	IDhi14	Common event `0x400E` implemented. **0b0** Event 0x400E not implemented.	`0b0`
[13]	IDhi13	Common event `0x400D` implemented. **0b0** Event 0x400D not implemented.	`0b0`
[12]	IDhi12	Common event `0x400C` implemented. **0b0** Event 0x400C not implemented.	`0b0`
[11]	IDhi11	Common event `0x400B` implemented. **0b0** Event 0x400B not implemented.	`0b0`
[10]	IDhi10	Common event `0x400A` implemented. **0b0** Event 0x400A not implemented.	`0b0`
[9]	IDhi9	Common event `0x4009` implemented. **0b0** Event 0x4009 not implemented.	`0b0`
[8]	IDhi8	Common event `0x4008` implemented. **0b0** Event 0x4008 not implemented.	`0b0`
[7]	IDhi7	Common event `0x4007` implemented. **0b0** Event 0x4007 not implemented.	`0b0`
[6]	IDhi6	Common event `0x4006` implemented. **0b0** Event 0x4006 not implemented.	`0b0`
[5]	IDhi5	Common event `0x4005` implemented. **0b0** Event 0x4005 not implemented.	`0b0`
[4]	IDhi4	Common event `0x4004` implemented. **0b0** Event 0x4004 not implemented.	`0b0`

Bits	Name	Description	Reset
[3]	IDhi3	Common event `0x4003` implemented. **0b0** Event 0x4003 not implemented.	`0b0`
[2]	IDhi2	Common event `0x4002` implemented. **0b0** Event 0x4002 not implemented.	`0b0`
[1]	IDhi1	Common event `0x4001` implemented. **0b0** Event 0x4001 not implemented.	`0b0`
[0]	IDhi0	Common event `0x4000` implemented. **0b0** Event 0x4000 not implemented.	`0b0`

Access

MRS <Xt>, S3_0_C15_C6_5

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0110`	`0b101`

MSR S3_0_C15_C6_5, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0110`	`0b101`

Accessibility

MRS <Xt>, S3_0_C15_C6_5

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCEID1_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCEID1_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCEID1_EL1;
```

MSR S3_0_C15_C6_5, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID1_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID1_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCEID1_EL1 = X[t];
```

A.2.14 IMP_CLUSTERPMCLAIMSET_EL1, Claim Tag Set Register

In conjunction with AArch64-PMCCLAIMCLR, provides Claim Tag bits that can be separately set and cleared to indicate whether functionality is in use by a debug agent.

For additional information see the CoreSight Architecture Specification.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-34: AArch64_imp_clusterpmclaimset_el1 bit assignments

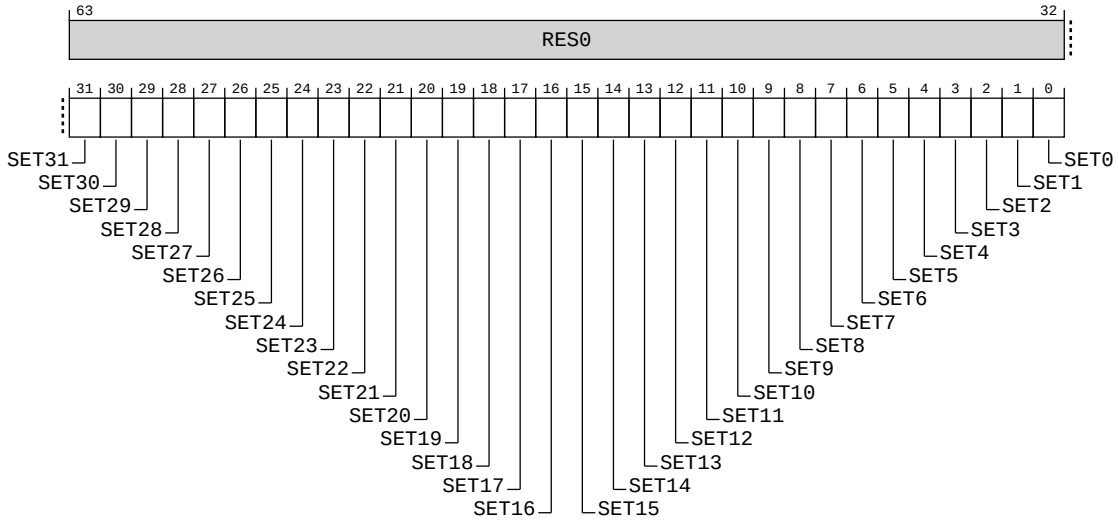


Table A-102: IMP_CLUSTERPMCLAIMSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31:0]	SET<m>, bit[m], where m = 31 to 0	Claim Tag Set. Indicates whether Claim Tag bit m is implemented, and is used to set Claim Tag bit m to `0b1`. **`0b0`** On a read: Claim Tag bit m is not implemented. On a write: Ignored. **`0b1`** On a read: Claim Tag bit m is implemented. On a write: Set Claim Tag bit m to 0b1.	`32{x}`

Access

MRS <Xt>, S3_0_C15_C6_6

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0110`	`0b110`

MSR S3_0_C15_C6_6, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0110`	`0b110`

Accessibility

MRS <Xt>, S3_0_C15_C6_6

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCLAIMSET_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCLAIMSET_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCLAIMSET_EL1;
```

MSR S3_0_C15_C6_6, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCLAIMSET_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCLAIMSET_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCLAIMSET_EL1 = X[t];
```

A.2.15 IMP_CLUSTERPMCLAIMCLR_EL1, Claim Tag Clear Register

In conjunction with AArch64-IMP_CLUSTERPMCLAIMSET, provides Claim Tag bits that can be separately set and cleared to indicate whether functionality is in use by a debug agent.

For additional information see the CoreSight Architecture Specification.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0000 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-35: AArch64_imp_clusterpmclaimclr_el1 bit assignments

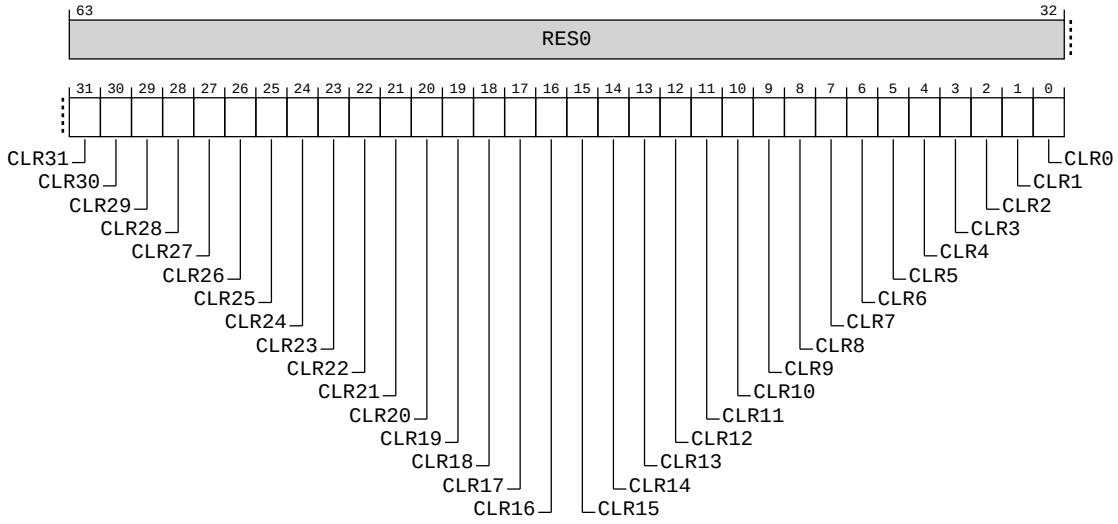


Table A-105: IMP_CLUSTERPMCLAIMCLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[31:0]	CLR<m>, bit[m], where m = 31 to 0	Claim Tag Clear. Indicates the current status of the Claim Tag bit m, and is used to clear Claim Tag bit m to `0b0`. **0b0** On a read: Claim Tag bit m is not set. On a write: Ignored. **0b1** On a read: Claim Tag bit m is set. On a write: Clear Claim tag bit m to 0b0. The number of Claim Tag bits implemented is indicated in AArch64-IMP_CLUSTERPMCLAIMSET_EL1.	`0x00000000`

Access

MRS <Xt>, S3_0_C15_C6_7

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b111

MSR S3_0_C15_C6_7, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b111

Accessibility

MRS <Xt>, S3_0_C15_C6_7

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCLAIMCLR_EL1;
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCLAIMCLR_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCLAIMCLR_EL1;
```

MSR S3_0_C15_C6_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
```

```
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCLAIMCLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCLAIMCLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCLAIMCLR_EL1 = X[t];
```

A.3 AArch64 RAS register summary

The summary table provides an overview of all *Reliability, Availability, and Serviceability* (RAS) registers with *IMPLEMENTATION DEFINED* values such as AArch64 performance monitors and AArch64 RAS in the *DynamIQ™ Shared Unit-110* (DSU-110). Individual register descriptions provide detailed information.

Note

- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table A-108: RAS registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
ERXFR_EL1	3	0	C5	C4	0	—	64-bit	Selected Error Record Feature Register	No
ERXCTLR_EL1	3	0	C5	C4	1	—	64-bit	Selected Error Record Control Register	No
ERXSTATUS_EL1	3	0	C5	C4	2	—	64-bit	Selected Error Record Primary Status Register	No
ERXPFGF_EL1	3	0	C5	C4	4	—	64-bit	Selected Pseudo-fault Generation Feature Register	No
ERXPFGCTL_EL1	3	0	C5	C4	5	—	64-bit	Selected Pseudo-fault Generation Control Register	No
ERXPFGCDN_EL1	3	0	C5	C4	6	—	64-bit	Selected Pseudo-fault Generation Countdown Register	No
ERXMISC0_EL1	3	0	C5	C5	0	—	64-bit	Selected Error Record Miscellaneous Register 0	No
ERXMISC1_EL1	3	0	C5	C5	1	—	64-bit	Selected Error Record Miscellaneous Register 1	No
ERXMISC2_EL1	3	0	C5	C5	2	—	64-bit	Selected Error Record Miscellaneous Register 2	No
ERXMISC3_EL1	3	0	C5	C5	3	—	64-bit	Selected Error Record Miscellaneous Register 3	No

A.3.1 ERXFR_EL1, Selected Error Record Feature Register

Accesses ext-CLUSTERRAS_ERR0FR when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00 1001 0000 1010 1001 1010 0110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-36: AArch64_erxfr_el1 bit assignments

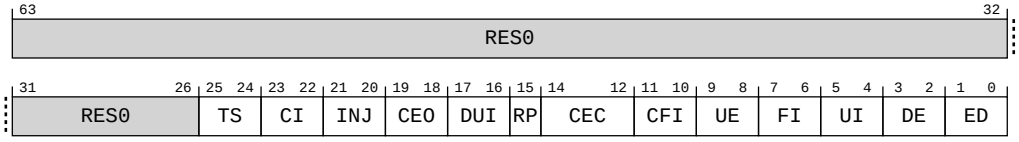


Table A-109: ERXFR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:26]	**RES0**	Reserved	**RES0**
[25:24]	TS	Timestamp Extension. **0b00** The node does not support a timestamp register.	0b00

Bits	Name	Description	Reset
[23:22]	CI	Critical error interrupt. Indicates whether the critical error interrupt and associated controls are implemented. **0b10** Critical error interrupt is supported and it can be enabled using associated controls. All other values are reserved.	0b10
[21:20]	INJ	Fault Injection Extension. Indicates whether the RAS Common Fault Injection Model Extension is implemented. **0b01** The node implements the RAS Common Fault Injection Model Extension. See ext-CLUSTERRAS_ERR0PFGF for more information. All other values are reserved.	0b01
[19:18]	CEO	Corrected Error overwrite. Indicates the behavior when a second Corrected error is detected after a first Corrected error has been recorded by an error record <m> owned by the node. **0b00** Counts Corrected errors. Keeps the previous error syndrome. If the counter overflows then CLUSTERRAS_ERR0STATUS.OF is set to 1. All other values are reserved.	0b00
[17:16]	DUI	Error recovery interrupt for deferred errors. Indicates whether the node implements a control for enabling error recovery interrupts on deferred errors. **0b00** Does not support feature. ext-CLUSTERRAS_ERR0CTLR.DUI is **RES0**. All other values are reserved.	0b00
[15]	RP	Repeat counter. Indicates whether the node implements a repeat Corrected error counter in CLUSTERRAS_ERR0MISC0 for each error record <m> owned by the node that implements a standard Corrected error counter. **0b1** A first (repeat) counter and a second (other) counter are implemented. The repeat counter is the same size as the primary error counter.	0b1
[14:12]	CEC	Corrected Error Counter. Indicates whether the node implements standard Corrected error counter (CE counter) mechanisms in CLUSTERRAS_ERR0MISC0 for each error record <m> owned by the node that can record countable errors. **0b010** Implements an 8-bit Corrected error counter in CLUSTERRAS_ERR0MISC0[39:32]. All other values are reserved.	0b010

Bits	Name	Description	Reset
[11:10]	CFI	Fault handling interrupt for corrected errors. Indicates whether the node implements a control for enabling fault handling interrupts on corrected errors. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.CFI. All other values are reserved.	0b10
[9:8]	UE	In-band uncorrected error reporting. Indicates whether the node implements in-band uncorrected error reporting (External aborts), and, if so, whether the node implements controls for enabling and disabling the reporting. **0b01** Feature always enabled. ext-CLUSTERRAS_ERR0CTLR.UE is **RES0**.	0b01
[7:6]	FI	Fault handling interrupt. Indicates whether the node implements a fault handling interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.FI.	0b10
[5:4]	UI	Error recovery interrupt for uncorrected errors. Indicates whether the node implements an error recovery interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.UI.	0b10
[3:2]	DE	Deferred error enable. **0b01** Deferred errors is always enabled.	0b01
[1:0]	ED	Error reporting and logging. Indicates this is the first record owned by the cluster. The cluster implements controls for enabling and disabling error reporting and logging. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.ED. The value 0b11 is reserved.	0b10

Access

MRS <Xt>, ERXFR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b000

Accessibility

MRS <Xt>, ERXFR_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXFR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXFR_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXFR_EL1;
elsif PSTATE.EL == EL3 then
    return ERXFR_EL1;
```

A.3.2 ERXCTLR_EL1, Selected Error Record Control Register

Accesses ext-CLUSTERRAS_ERR0CTLR when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0 xx0x

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-37: AArch64_erxctlr_el1 bit assignments

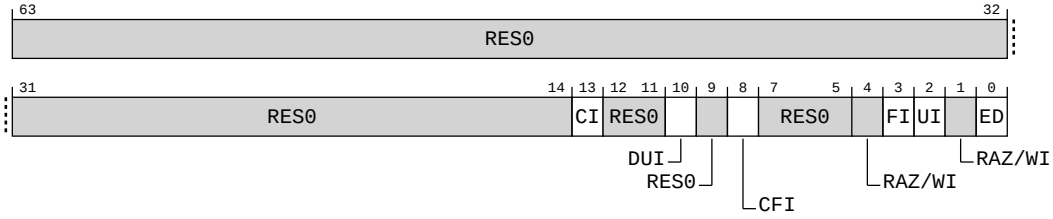


Table A-111: ERXCTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:14]	**RES0**	Reserved	**RES0**
[13]	CI	Critical error interrupt enable. When enabled, the critical error interrupt is generated for a critical error condition. **0b0** Critical error interrupt not generated for critical errors. Critical errors are treated as Uncontained errors. **0b1** Critical error interrupt generated for critical errors.	x
[12:11]	**RES0**	Reserved	**RES0**
[10]	DUI	Error recovery interrupt for deferred errors enable. This control applies to errors arising from both reads and writes. When enabled, an error recovery interrupt is generated for all detected Deferred errors. **0b0** Error recovery interrupt not generated for deferred errors. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error. Access to this field is: RO	x
[9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8]	CFI	Fault handling interrupt for Corrected errors enable. This control applies to errors arising from both reads and writes. When enabled, the fault handling interrupt is generated when a counter overflows and the overflow bit for the counter is set to 1. For more information, see ext-ERR<n>MISC0. **0b0** Fault handling interrupt not generated for Corrected errors. **0b1** Fault handling interrupt generated for Corrected errors. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[7:5]	**RES0**	Reserved	**RES0**
[4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	FI	Fault handling interrupt enable. This control applies to errors arising from both reads and writes. When enabled, the fault handling interrupt is generated for all detected Corrected errors, Deferred errors,and Uncorrected errors. **0b0** Fault handling interrupt disabled. **0b1** Fault handling interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[2]	UI	Uncorrected error recovery interrupt enable. This control applies to errors arising from both reads and writes. When enabled, the error recovery interrupt is generated for all detected Uncorrected errors that are not deferred. **0b0** Error recovery interrupt disabled. **0b1** Error recovery interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[1]	**RAZ/WI**	Reserved	**RAZ/WI**
[0]	ED	Error reporting and logging enable. When disabled, the node behaves as if error detection and correction are disabled, and no errors are recorded or signaled by the node. **0b0** Error reporting disabled. **0b1** Error reporting enabled.	x

Access

MRS <Xt>, ERXCTLR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b001

MSR ERXCTLR_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b001

Accessibility

MRS <Xt>, ERXCTLR_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXCTLR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXCTLR_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXCTLR_EL1;
elsif PSTATE.EL == EL3 then
    return ERXCTLR_EL1;
```

MSR ERXCTLR_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXCTLR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
```

```
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXCTLR_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXCTLR_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXCTLR_EL1 = X[t];
```

A.3.3 ERXSTATUS_EL1, Selected Error Record Primary Status Register

Accesses ext-CLUSTERRAS_ERR0STATUS when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0xxx 0000 0000 0000 0000


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-38: AArch64_erxstatus_el1 bit assignments

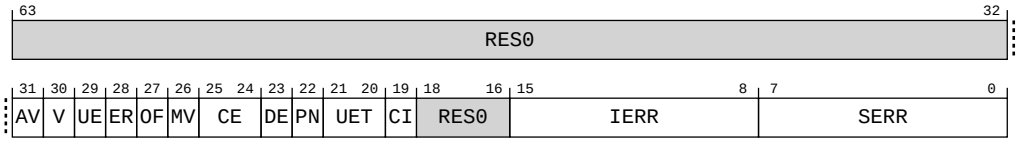


Table A-114: ERXSTATUS_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	AV	Address Valid. **0b0** ext-CLUSTERRAS_ERR0ADDR not valid. This bit is unimplemented and treated as **RAZ/WI**.	0b0
[30]	V	Status Register Valid. **0b0** CLUSTERRAS_ERR0STATUS not valid. **0b1** CLUSTERRAS_ERR0STATUS valid. At least one error has been recorded. This bit is read/write-one-to-clear.	0b0
[29]	UE	Uncorrected error. **0b0** No errors have been detected, or all detected errors have been either corrected or deferred. **0b1** At least one detected error was not corrected and not deferred. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write one to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0
[28]	ER	Error Reported. **0b0** No in-band error (External abort) reported. This bit is unimplemented and treated as **RAZ/WI**.	0b0

Bits	Name	Description	Reset
[27]	OF	Overflow. Indicates that multiple errors have been detected. This bit is set to 1 when one of the following occurs: • A Corrected error is counted and the counter overflows. • CLUSTERRAS_ERR0STATUS.V was previously set to 1 and a type of error other than a Corrected error is recorded. Otherwise, this bit is unchanged when an error is recorded. A direct write that modifies the counter overflow flag indirectly might set this bit to an **UNKNOWN** value. A direct write to this bit that clears this bit to zero might indirectly set the counter overflow flag to an **UNKNOWN** value. **0b0** Since this bit was last cleared to zero, no error syndrome has been discarded and, if a Corrected error counter is implemented, it has not overflowed. **0b1** Since this bit was last cleared to zero, at least one error syndrome has been discarded or, if a Corrected error counter is implemented, it might have overflowed. If this bit is nonzero, then software must write 1 to this bit, to clear this bit to zero, when clearing CLUSTERRAS_ERR0STATUS.V to 0. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0
[26]	MV	Miscellaneous Registers (CLUSTERRAS_ERR0MISC0) Valid. **0b0** CLUSTERRAS_ERR0MISC0 is not valid. **0b1** The contents of CLUSTERRAS_ERR0MISC0 contains additional information for an error recorded by this record. Only CLUSTERRAS_ERR0MISC0 is implemented. CLUSTERRAS_ERR0MISC1,2,3 are treated as **RAZ/WI**. This bit is read/write-one-to-clear.	0b0
[25:24]	CE	Corrected Error. **0b00** No errors were corrected. **0b10** At least one error was corrected. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this field is nonzero, then software must write ones to this field to clear this field to zero. If CLUSTERRAS_ERR0STATUS.V is set to 0, this field is not valid and reads **UNKNOWN**. This field is read/write-one-to-clear. Writing a value other than all-zeros or all-ones sets this field to an **UNKNOWN** value.	0b00

Bits	Name	Description	Reset
[23]	DE	Deferred Error. **0b0** No errors were deferred. **0b1** At least one error was not corrected and deferred. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. If CLUSTERRAS_ERR0STATUS.V is set to 0, this bit is not valid and reads **UNKNOWN**. This bit is read/write-one-to-clear.	0b0
[22]	PN	Poison. **0b0** Uncorrected error or Deferred error recorded because a corrupt value was detected. **0b1** Uncorrected error or Deferred error recorded because a poison value was detected. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if any of the following are true: • CLUSTERRAS_ERR0STATUS.V is set to 0. • CLUSTERRAS_ERR0STATUS.{DE, UE} are both set to 0. This bit is read/write-one-to-clear.	0b0
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. **0b00** Uncorrected error, Uncontainable error (UC). This field is not implemented and is treated as **RAZ/WI**.	0b00
[19]	CI	Critical error. Indicates whether a critical error condition has been recorded. **0b0** No critical error condition recorded. **0b1** Critical error condition recorded. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0
[18:16]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[15:8]	IERR	**IMPLEMENTATION DEFINED** Extended error code. Used with any primary error code SERR value. Additional information is placed in the CLUSTERRAS_ERR0MISC0 register. **0b00000000** If SERR == 0x7, indicates a Tag RAM error. Not used with other SERR values. **0b00000010** If SERR == 0x7, indicates a Snoop Filter RAM error. Not used with other SERR values. This field is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0.	0x00
[7:0]	SERR	Primary error code. Indicates the type of Primary error. **0b00000000** No error. **0b00000001** **IMPLEMENTATION DEFINED** error. **0b00000010** Data value from (non-associative) internal memory. For example, ECC from on-chip SRAM or buffer. **0b00000011** **IMPLEMENTATION DEFINED** pin. For example, nSEI pin. **0b00000100** Assertion failure. For example, consistency failure. **0b00000101** Error detected on internal data path. For example, parity on ALU result. **0b00000110** Data value from associative memory. For example, ECC error on cache data. **0b00000111** Address/control value from associative memory. For example, ECC error on cache tag. **0b00001000** Data value from a TLB. For example, ECC error on TLB data. **0b00001001** Address/control value from a TLB. For example, ECC error on TLB tag.	0x00

Bits	Name	Description	Reset
[7:0] continued	SERR	**0b00001010** Data value from producer. For example, parity error on write data bus. **0b00001011** Address/control value from producer. For example, parity error on address bus. **0b00001100** Data value from (non-associative) external memory. For example, ECC error in SDRAM. **0b00001101** Illegal address (software fault). For example, access to unpopulated memory. **0b00001110** Illegal access (software fault). For example, byte write to word register. **0b00001111** Illegal state (software fault). For example, device not ready. **0b00010000** Internal data register. For example, parity on a SIMD&FP register. For a PE, all general-purpose, stack pointer, SIMD&FP, and SVE registers are data registers. **0b00010001** Internal control register. For example, Parity on a System register. For a PE, all registers other than general-purpose, stack pointer, SIMD&FP, and SVE registers are control registers. **0b00010010** Error response from slave. For example, error response from cache write-back. **0b00010011** External timeout. For example, timeout on interaction with another node.	0x00
[7:0] continued	SERR	**0b00010100** Internal timeout. For example, timeout on interface within the node. **0b00010101** Deferred error from slave not supported at master. For example, poisoned data received from a slave by a master that cannot defer the error further. All other values are reserved. This field is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0.	0x00

Access

MRS <Xt>, ERXSTATUS_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b010

MSR ERXSTATUS_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b010

Accessibility

MRS <Xt>, ERXSTATUS_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXSTATUS_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXSTATUS_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXSTATUS_EL1;
elsif PSTATE.EL == EL3 then
    return ERXSTATUS_EL1;
```

MSR ERXSTATUS_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXSTATUS_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXSTATUS_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXSTATUS_EL1 = X[t];
```

```
elsif PSTATE.EL == EL3 then
    ERXSTATUS_EL1 = X[t];
```

A.3.4 ERXPFGF_EL1, Selected Pseudo-fault Generation Feature Register

Accesses the ext-CLUSTERRAS_ERR0PFGF register when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x11x xxxx xxxx xxxx xxx1 0101 0110 0011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-39: AArch64_erxpfgf_el1 bit assignments

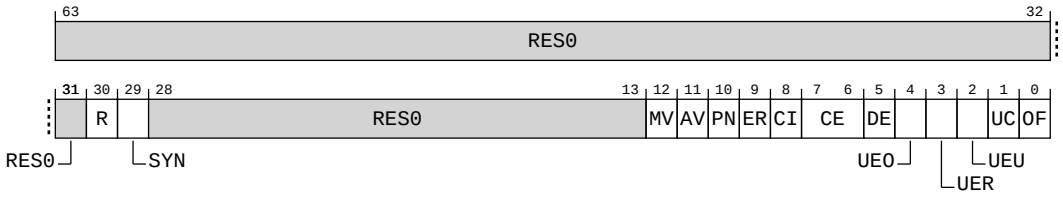


Table A-117: ERXPFGF_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30]	R	Restartable. Support for Error Generation Counter restart mode. **0b1** Feature controllable.	0b1

Bits	Name	Description	Reset
[29]	SYN	Syndrome. Fault syndrome injection. **0b1** When an injected error is recorded, the node does not update the ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} fields. ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} are writable when ext-CLUSTERRAS_ERR0STATUS.V == 0. **Note:** Software can write intended values into the ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} fields when setting up a fault injection event.	0b1
[28:13]	**RES0**	Reserved	**RES0**
[12]	MV	Miscellaneous syndrome. Additional syndrome injection. Defines whether software can control all or part of the syndrome recorded in the CLUSTERRAS_ERR0MISC0 register when an injected error is recorded. CLUSTERRAS_ERR0MISC1-3 registers are reserved and unused for this purpose. **0b1** When an injected error is recorded, the node does not update all the syndrome fields in CLUSTERRAS_ERR0MISC0. The node records syndrome in CLUSTERRAS_ERR0MISC0 OFO, CECO, OFR, CECR, WAY, INDX, LVL, and IND fields and sets ext-CLUSTERRAS_ERR0STATUS.MV to 1. CLUSTERRAS_ERR0PGFCTL.MV is **RAO**. **Note:** Software can write intended values into the CLUSTERRAS_ERR0MISC0 register when setting up a fault injection event.	0b1
[11]	AV	Address syndrome. Address syndrome injection. Always **RAZ/WI**. **0b0** The node does not support ext-CLUSTERRAS_ERR0ADDR and does not set ext-CLUSTERRAS_ERR0STATUS.AV.	0b0
[10]	PN	Poison flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.PN status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.PN is set to ext-CLUSTERRAS_ERR0PFGCTL.PN.	0b1
[9]	ER	Error Reported flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.ER status flag. **0b0** When an injected error is recorded, the node does not set ext-CLUSTERRAS_ERR0STATUS.ER. This bit reads-as-zero.	0b0
[8]	CI	Critical Error flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.CI status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.CI is set to ext-CLUSTERRAS_ERR0PFGCTL.CI.	0b1

Bits	Name	Description	Reset
[7:6]	CE	Corrected Error generation. Describes the types of Corrected Error that the fault generation feature of the node can generate. **0b01** The fault generation feature of the node allows generation of a non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERR0STATUS.CE == 0b10.	0b01
[5]	DE	Deferred Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b1** The fault generation feature of the node allows generation of this type of error.	0b1
[4]	UEO	Latent or Restartable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b0** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	0b0
[3]	UER	Signaled or Recoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b0** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	0b0
[2]	UEU	Unrecoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b0** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	0b0
[1]	UC	Uncontainable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b1** The fault generation feature of the node allows generation of this type of error.	0b1
[0]	OF	Overflow flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.OF status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.OF is set to ext-CLUSTERRAS_ERR0PFGCTL.OF.	0b1

Access

MRS <Xt>, ERXPFGF_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b100

Accessibility

MRS <Xt>, ERXPFGF_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXPFGF_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXPFGF_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXPFGF_EL1;
elsif PSTATE.EL == EL3 then
    return ERXPFGF_EL1;
```

A.3.5 ERXPFGCTL_EL1, Selected Pseudo-fault Generation Control Register

Accesses the ext-CLUSTERRAS_ERR0PFGCTL register when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-40: AArch64_erxpfgctl_el1 bit assignments

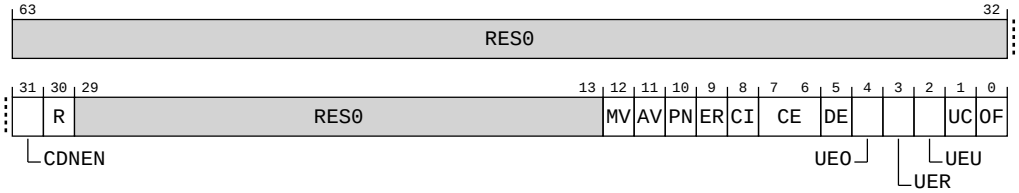


Table A-119: ERXPFGCTL_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	CDNEN	Countdown Enable. Controls transfers from the value that is held in the ext-CLUSTERRAS_ERR0PFGCDN into the Error Generation Counter, and enables this counter. **0b0** The Error Generation Counter is disabled. **0b1** The Error Generation Counter is enabled. On a write of 1 to this bit, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.	0b0
[30]	R	Restart. Controls whether, on reaching zero, the Error Generation Counter restarts from the ext-CLUSTERRAS_ERR0PFGCDN value, or stops. **0b0** On reaching 0, the Error Generation Counter stops. **0b1** On reaching 0, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.	x
[29:13]	**RES0**	Reserved	**RES0**
[12]	MV	Miscellaneous syndrome. The value that is written to ext-CLUSTERRAS_ERR0STATUS.MV when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.MV is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.MV is set to 1 when an injected error is recorded.	x

Bits	Name	Description	Reset
[11]	AV	Address syndrome. The value that is written to ext-CLUSTERRAS_ERR0STATUS.AV when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.AV is set to 0 when an injected error is recorded. This bit is **RES0**. Access to this field is: **RES0**	x
[10]	PN	Poison flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.PN when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.PN is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.PN is set to 1 when an injected error is recorded.	x
[9]	ER	**0b0** ext-CLUSTERRAS_ERR0STATUS.ER is set to 0 when an injected error is recorded. This bit is **RES0**. Access to this field is: **RES0**	x
[8]	CI	Critical Error flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.CI when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.CI is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.CI is set to 1 when an injected error is recorded.	x
[7:6]	CE	Corrected Error generation enable. Controls the type of Corrected Error condition that might be generated. **0b00** No error of this type is generated. **0b01** A non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERR0STATUS.CE == 0b10, might be generated when the Error Generation Counter decrements to zero. The set of permitted values for this field is defined by ext-CLUSTERRAS_ERR0PFGF.CE.	xx
[5]	DE	Deferred Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. **0b1** An error of this type might be generated when the Error Generation Counter decrements to zero.	x
[4]	UEO	Latent or Restartable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x

Bits	Name	Description	Reset
[3]	UER	Signaled or Recoverable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[2]	UEU	Unrecoverable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[1]	UC	Uncontainable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. **0b1** An error of this type might be generated when the Error Generation Counter decrements to zero.	x
[0]	OF	Overflow flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.OF when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.OF is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.OF is set to 1 when an injected error is recorded.	x

Access

MRS <Xt>, ERXPFGCTL_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b101

MSR ERXPFGCTL_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b101

Accessibility

MRS <Xt>, ERXPFGCTL_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
```

```
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXPFGCTL_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXPFGCTL_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXPFGCTL_EL1;
elsif PSTATE.EL == EL3 then
    return ERXPFGCTL_EL1;
```

MSR ERXPFGCTL_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXPFGCTL_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCTL_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCTL_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXPFGCTL_EL1 = X[t];
```

A.3.6 ERXPFGCDN_EL1, Selected Pseudo-fault Generation Countdown Register

Accesses the ext-CLUSTERRAS_ERR0PFGCDN register when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-41: AArch64_erxpfgcdn_el1 bit assignments

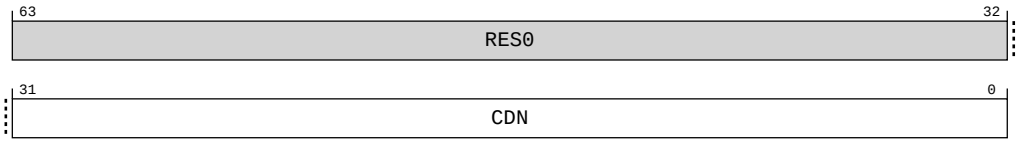


Table A-122: ERXPFGCDN_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[31:0]	CDN	Countdown value. This field is copied to Error Generation Counter when either: • Software writes ext-CLUSTERRAS_ERR0PFGCTL.CDNEN with 1. • The Error Generation Counter decrements to zero and ext-CLUSTERRAS_ERR0PFGCTL.R == 1. While ext-CLUSTERRAS_ERR0PFGCTL.CDNEN == 1 and the Error Generation Counter is nonzero, the counter decrements by 1 for each cycle. When the counter reaches 0, one of the errors enabled in the ext-CLUSTERRAS_ERR0PFGCTL register is generated. **Note:** The current Error Generation Counter value is not visible to software.	32{x}

Access

MRS <Xt>, ERXPFGCDN_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b110

MSR ERXPFGCDN_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b110

Accessibility

MRS <Xt>, ERXPFGCDN_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXPFGCDN_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXPFGCDN_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXPFGCDN_EL1;
```

```
elsif PSTATE.EL == EL3 then
    return ERXPFGCDN_EL1;
```

MSR ERXPFGCDN_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXPFGCDN_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCDN_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCDN_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXPFGCDN_EL1 = X[t];
```

A.3.7 ERXMISC0_EL1, Selected Error Record Miscellaneous Register 0

Accesses ext-CLUSTERRAS_ERR0MISC0 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Miscellaneous error syndrome register. The Miscellaneous error syndrome register contains:

- 2 architecturally-defined Corrected error counters with sticky overflow bits,
- Information to identify the FRU in which the error was detected, including Index, Way, Level, Instruction vs. Data fields.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure A-42: AArch64_erxmisc0_el1 bit assignments

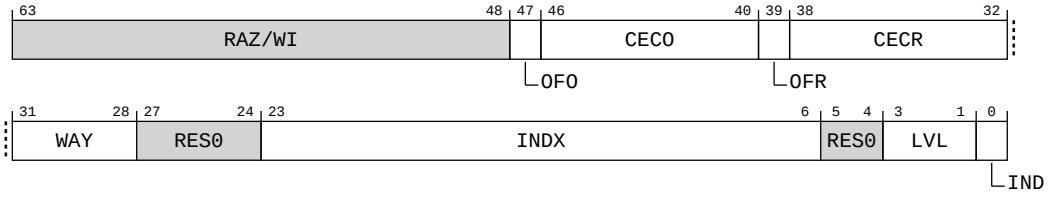


Table A-125: ERXMISC0_EL1 bit descriptions

Bits	Name	Description	Reset
[63:48]	**RAZ/WI**	Reserved	**RAZ/WI**
[47]	OFO	Sticky overflow bit for Other errors. Set to 1 when the Corrected error count Other (CECO) field is incremented and wraps through zero. **0b0** Other counter has not overflowed. **0b1** Other counter has overflowed. A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an **UNKNOWN** value.	x
[46:40]	CECO	Corrected error count for Other errors. The Other error counter increments for all Corrected errors that are not counted by the CECR Repeat error counter due to the syndrome of the new error mismatching against the recorded syndrome of the first Repeat error. Refer to the CECR Repeat error description for fields used to match syndrome. At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.	7{x}

Bits	Name	Description	Reset
[39]	OFR	Sticky overflow bit for Repeat errors. Set to 1 when the Corrected error count Repeat (CECR) field is incremented and wraps through zero. **0b0** Repeat counter has not overflowed. **0b1** Repeat counter has overflowed. A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an **UNKNOWN** value.	x
[38:32]	CECR	Corrected error count for Repeat errors. The Repeat error counter increments for the first Corrected error and records the syndrome for the error in the fields described below. It also increments for each subsequent Corrected error with a syndrome matching the first error's recorded syndrome, otherwise the error causes an increment to the CECO Other counter. The syndrome is recorded in the following fields: • ext-CLUSTERRAS_ERR0STATUS.IERR • ext-CLUSTERRAS_ERR0STATUS.SERR • ext-CLUSTERRAS_ERR0MISC0.INDX • ext-CLUSTERRAS_ERR0MISC0.WAY The syndrome is matched on a new Corrected error if all of the following are true: • ext-CLUSTERRAS_ERR0STATUS.MV bit is set, • ext-CLUSTERRAS_ERR0STATUS.IERR matches the new error, • ext-CLUSTERRAS_ERR0STATUS.SERR matches the new error, • ext-CLUSTERRAS_ERR0MISC0.INDX matches the new error, • ext-CLUSTERRAS_ERR0MISC0.WAY matches the new error. CLUSTERRAS_ERR0STATUS.MV indicates the validity of the INDX and WAY fields of the CLUSTERRAS_ERR0MISC0 register At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.	7{x}
[31:28]	WAY	L3 Cache Way that contained the error.	xxxx
[27:24]	**RES0**	Reserved	**RES0**
[23:6]	INDX	L3 Cache Index that contained the error.	18{x}
[5:4]	**RES0**	Reserved	**RES0**
[3:1]	LVL	L3 Cache Level that contained the error. Always 0x2. **0b010** Level 3 cache.	xxx
[0]	IND	L3 Cache instruction vs. data cache that contained the error. Always data (0x0). **0b0** Data cache error.	x

Access

MRS <Xt>, ERXMISC0_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b000

MSR ERXMISC0_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b000

Accessibility

MRS <Xt>, ERXMISC0_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC0_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC0_EL1;
elsif PSTATE.EL == EL3 then
    return ERXMISC0_EL1;
```

MSR ERXMISC0_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
```

```
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC0_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC0_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXMISC0_EL1 = X[t];
```

A.3.8 ERXMISC1_EL1, Selected Error Record Miscellaneous Register 1

Accesses ext-CLUSTERRAS_ERR0MISC1 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-43: AArch64_erxmisc1_el1 bit assignments

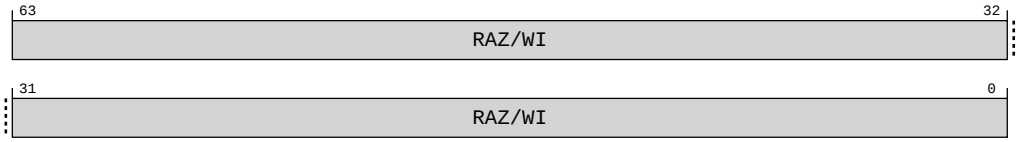


Table A-128: ERXMISC1_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	**RAZ/WI**	Reserved	**RAZ/WI**

Access

MRS <Xt>, ERXMISC1_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b001

MSR ERXMISC1_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b001

Accessibility

MRS <Xt>, ERXMISC1_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC1_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC1_EL1;
elsif PSTATE.EL == EL3 then
    return ERXMISC1_EL1;
```

MSR ERXMISC1_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
```

```
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC1_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC1_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXMISC1_EL1 = X[t];
```

A.3.9 ERXMISC2_EL1, Selected Error Record Miscellaneous Register 2

Accesses ext-CLUSTERRAS_ERR0MISC2 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-44: AArch64_erxmisc2_el1 bit assignments

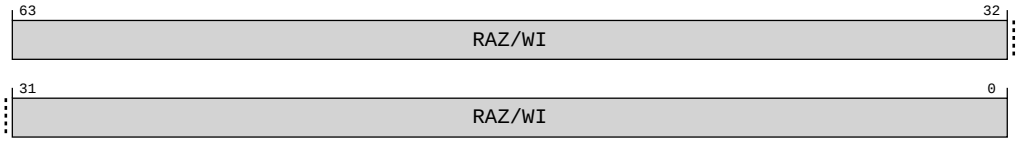


Table A-131: ERXMISC2_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	**RAZ/WI**	Reserved	**RAZ/WI**

Access

MRS <Xt>, ERXMISC2_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b010

MSR ERXMISC2_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b010

Accessibility

MRS <Xt>, ERXMISC2_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC2_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
```

```
    else
        return ERXMISC2_EL1;
elsif PSTATE.EL == EL3 then
    return ERXMISC2_EL1;
```

MSR ERXMISC2_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC2_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC2_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXMISC2_EL1 = X[t];
```

A.3.10 ERXMISC3_EL1, Selected Error Record Miscellaneous Register 3

Accesses ext-CLUSTERRAS_ERR0MISC3 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-45: AArch64_erxmisc3_el1 bit assignments

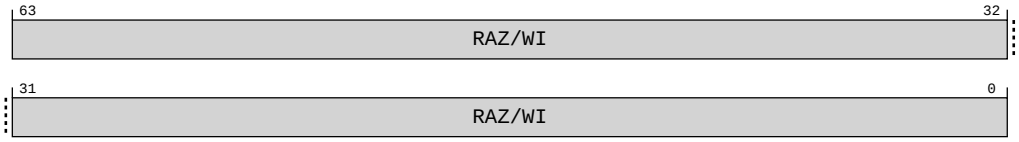


Table A-134: ERXMISC3_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

Access

MRS <Xt>, ERXMISC3_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b011

MSR ERXMISC3_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b011

Accessibility

MRS <Xt>, ERXMISC3_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC3_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
```

```
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXMISC3_EL1;
elsif PSTATE.EL == EL3 then
    return ERXMISC3_EL1;
```

MSR ERXMISC3_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC3_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC3_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    ERXMISC3_EL1 = X[t];
```

Appendix B External registers

This appendix contains the descriptions for all the external (memory-mapped) registers in the *DynamIQ™ Shared Unit-110* (DSU-110).

B.1 Registers accessed over the utility bus

This section contains the descriptions for all the external registers in the *DynamIQ™ Shared Unit-110* (DSU-110) accessed over the utility bus.

B.1.1 External cluster register summary

The cluster system control registers are accessible either from memory-mapped accesses on the utility bus or from System register accesses from the cores. You must access the cluster system control registers from the Secure address space.

The summary table provides an overview of all the cluster system control registers that are accessed externally (memory-mapped) from the utility bus of the DSU-110. Individual register descriptions provide detailed information.


Note

- The cluster power control registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
  - The register is not present in Direct connect.
- Any address that is not documented is treated as **RAZ/WI**.
- The base address for the cluster power control registers is `0x000000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-1: Cluster registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0000	CLUSTERIDR	—	64-bit	Cluster Main Revision Register	Yes
0x0008	CLUSTERREVIDR	—	64-bit	Cluster ECO ID Register	Yes
0x0010	CLUSTERPWRCTLR	—	64-bit	Cluster Power Control Register	No
0x0018	CLUSTERL3HIT	—	64-bit	Cluster L3 Hit Counter Register	No
0x0020	CLUSTERL3MISS	—	64-bit	Cluster L3 Miss Counter Register	No
0x0028	CLUSTERL3DNTH0	—	64-bit	Cluster L3 Downsize Threshold0 Register	No
0x0030	CLUSTERL3DNTH1	—	64-bit	Cluster L3 Downsize Threshold1 Register	No
0x0038	CLUSTERL3UPTH0	—	64-bit	Cluster L3 Upsize Threshold0 Register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0040	CLUSTERL3UPTH1	—	64-bit	Cluster L3 Upsize Threshold1 Register	No
0x0048	CLUSTERBUSQOS	—	64-bit	Cluster Bus QoS Control Register	No
0x0050	CLUSTERCFR	—	64-bit	Cluster Configuration Register	Yes
0x0058	CLUSTERACTLR	—	64-bit	Cluster Auxiliary Control Register	Yes
0x0060	CLUSTERECTLR	—	64-bit	Cluster Extended Control Register	Yes
0x0068	CLUSTERCFR2	—	64-bit	Cluster Configuration Register 2	No

B.1.1.1 CLUSTERIDR, Cluster Main Revision Register

Holds the revision and patch level of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0000

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0100 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-1: ext_clusteridr bit assignments

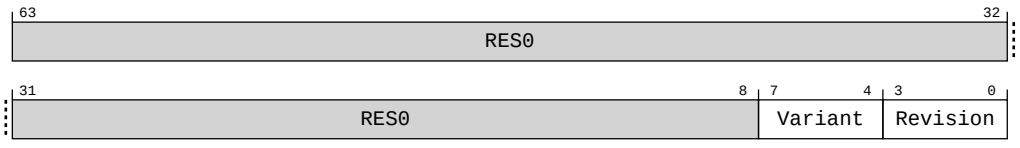


Table B-2: CLUSTERIDR bit descriptions

Bits	Name	Description	Reset
[63:8]	**RES0**	Reserved	**RES0**
[7:4]	Variant	Indicates the variant of the DSU. This is the major revision number x in the rx part of the rxpy description of the product revision status. **0b0000** Cluster major revision number 0. **0b0001** Cluster major revision number 1. **0b0010** Cluster major revision number 2. **0b0011** Cluster major revision number 3. **0b0100** Cluster major revision number 4.	0b0100
[3:0]	Revision	Indicates the minor revision number of the DSU. This is the minor revision number y in the py part of the rxpy description of the product revision status. **0b0000** Cluster minor revision 0. **0b0001** Cluster minor revision 1. **0b0010** Cluster minor revision 2. **0b0011** Cluster minor revision 3. **0b0100** Cluster minor revision 4.	0b0000

B.1.1.2 CLUSTERREVIDR, Cluster ECO ID Register

Enables ECO patches to be applied to the cluster level to be identified by software.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0008

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-2: ext_clusterrevidr bit assignments

Table B-3: CLUSTERREVIDR bit descriptions

Bits	Name	Description	Reset
[63:0]	ECOID	Contains ECO information. Refer to the errata documentation for any bit allocations. **0b00000000000000000000000000000000000000000000000000000000000000000** Customer ECO ID	0x0000000000000000

B.1.1.3 CLUSTERPWRCTLR, Cluster Power Control Register

This register controls power features of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0010

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0xxx 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-3: ext_clusterpwrctlr bit assignments

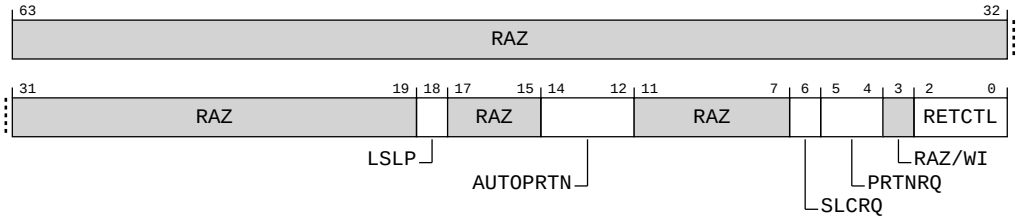


Table B-4: CLUSTERPWRCTLR bit descriptions

Bits	Name	Description	Reset
[63:19]	**RAZ**	Reserved	**RAZ**
[18]	LSLP	Enable L3 RAM light sleep.	0b0
[17:15]	**RAZ**	Reserved	**RAZ**
[14:12]	AUTOPRTN	Enable automatic RAM power down and configure evaluation time period. Note that a shorter time period allows better responsiveness to changing workloads, however if it is too short then the cost of frequent resizing can be too high. **0b000** Disabled **0b001** 8,192 architectural timer ticks, time perious of 164us-819us **0b010** 16,384 architectural timer ticks, time perious of 328us-1.6ms **0b011** 32,768 architectural timer ticks, time perious of 655us-3.3ms **0b100** 65,536 architectural timer ticks, time perious of 1.3ms-6.6ms **0b101** 131,072 architectural timer ticks, time perious of 2.6ms-13ms **0b110** 262,144 architectural timer ticks, time perious of 5.2ms-26ms **0b111** 524,288 architectural timer ticks, time perious of 10ms-52ms	0b000

Bits	Name	Description	Reset
[11:7]	**RAZ**	Reserved	**RAZ**
[6]	SLCRQ	Cache slice power request. These bits are passed to the PPU as an advisory request for which slices to power. **0b0** Request that one L3 cache slice is powered on. **0b1** Request that all L3 cache slices are powered on.	x
[5:4]	PRTNRQ	Cache portion power request. These bits are passed to the PPU as an advisory request for which portions to power. Note that these bits are only used when AUTOPRTN bits are 3'b000. **0b00** Request that none of the L3 cache portions in each slice is powered on **0b01** Request that half of the L3 cache portions in each slice are powered on **0b11** Request that both of the L3 cache portions in each slice are powered on	xx
[3]	**RAZ/WI**	Reserved	**RAZ/WI**
[2:0]	RETCTL	L3 Data RAM retention control. **0b000** Disable the retention circuit. **0b001** 2 architectural timer ticks, 40ns-200ns minimum delay before retention **0b010** 8 architectural timer ticks, 160ns-800ns minimum delay before retention **0b011** 32 architectural timer ticks, 640ns-3,200ns minimum delay before retention **0b100** 64 architectural timer ticks, 1280ns-6,400ns minimum delay before retention **0b101** 128 architectural timer ticks, 2,560ns-12,800ns minimum delay before retention **0b110** 256 architectural timer ticks, 5,120ns-25,600ns minimum delay before retention **0b111** 512 architectural timer ticks, 10,240ns-51,200ns minimum delay before retention	0b000

B.1.1.4 CLUSTERL3HIT, Cluster L3 Hit Counter Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0018

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-4: ext_clusterl3hit bit assignments

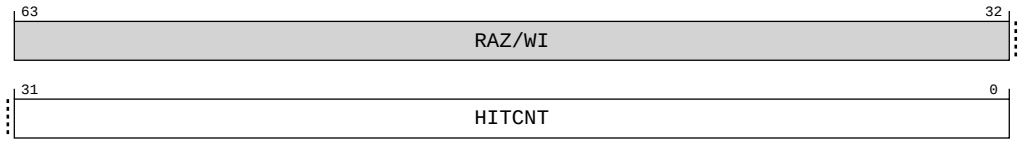


Table B-5: CLUSTERL3HIT bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	HITCNT	Count of number of L3 hits, for use in portion control calculations.	0x00000000

B.1.1.5 CLUSTERL3MISS, Cluster L3 Miss Counter Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0020

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-5: ext_clusterl3miss bit assignments

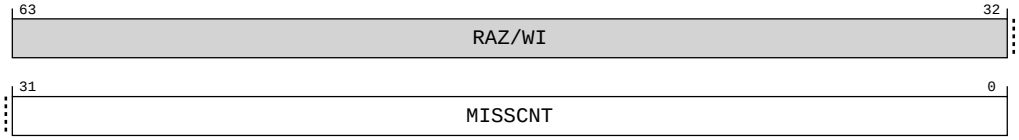


Table B-6: CLUSTERL3MISS bit descriptions

Bits	Name	Description	Reset
[63:32]	**RAZ/WI**	Reserved	**RAZ/WI**
[31:0]	MISSCNT	Count of number of L3 misses, for use in portion control calculations.	0x00000000

B.1.1.6 CLUSTERL3DNTH0, Cluster L3 Downsize Threshold0 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0028

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-6: ext_clusterl3dnth0 bit assignments

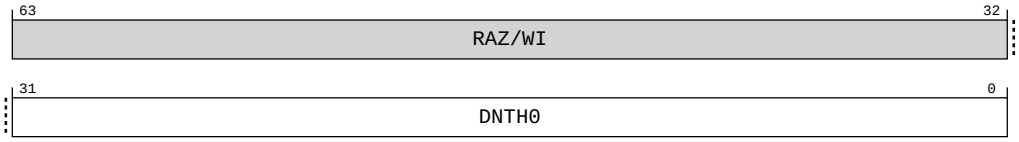


Table B-7: CLUSTERL3DNTH0 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RAZ/WI**	Reserved	**RAZ/WI**
[31:0]	DNTH0	If all L3 ways are powered and the cache hit bandwidth falls below this threshold then the cache is downsized to half the ways. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

B.1.1.7 CLUSTERL3DNTH1, Cluster L3 Downsize Threshold1 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0030

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-7: ext_clusterl3dnth1 bit assignments

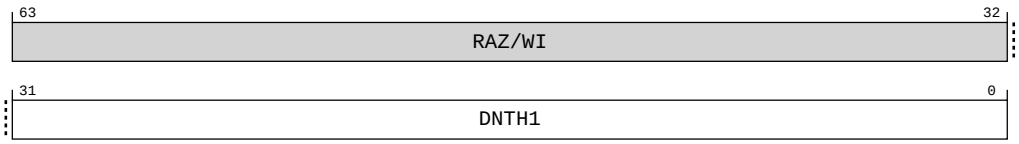


Table B-8: CLUSTERL3DNTH1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RAZ/WI**	Reserved	**RAZ/WI**
[31:0]	DNTH1	If all L3 ways are powered and the cache hit bandwidth falls below this threshold then the cache is downsized to none the ways. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

B.1.1.8 CLUSTERL3UPTH0, Cluster L3 Upsize Threshold0 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0038

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-8: ext_clusterl3upth0 bit assignments

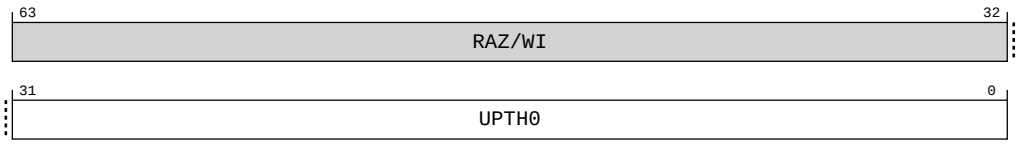


Table B-9: CLUSTERL3UPTH0 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RAZ/WI**	Reserved	**RAZ/WI**
[31:0]	UPTH0	If no L3 ways are powered and the cache miss bandwidth rises above this threshold then the cache is upsized to half the ways. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

B.1.1.9 CLUSTERL3UPTH1, Cluster L3 Upsize Threshold1 Register

This register is intended for use in algorithms for determining when to power up or down cache portions.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0040

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-9: ext_clusterl3upth1 bit assignments

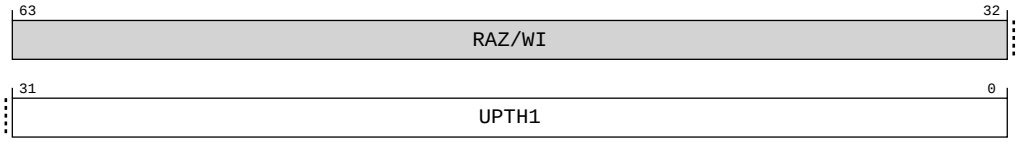


Table B-10: CLUSTERL3UPTH1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	UPTH1	If no L3 ways are powered and the cache miss bandwidth rises above this threshold then the cache is upsized to all of the ways. The value in this register is compared with the change in the cluster L3 hit counter since the last time period.	0x00000000

B.1.1.10 CLUSTERBUSQOS, Cluster Bus QoS Control Register

Determines the value driven on the CHI bus QoS field.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0048

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1011 1110

Bit descriptions

Figure B-10: ext_clusterbusqos bit assignments

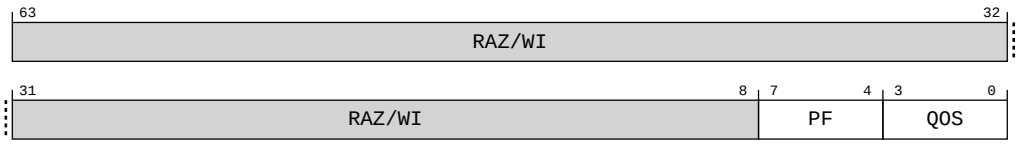


Table B-11: CLUSTERBUSQOS bit descriptions

Bits	Name	Description	Reset
[63:8]	**RAZ/WI**	Reserved	**RAZ/WI**
[7:4]	PF	Valid driven on the CHI bus QoS field for prefetches.	0b1011
[3:0]	QOS	Valid driven on the CHI bus QoS field for demand accesses.	0b1110

B.1.1.11 CLUSTERCFR, Cluster Configuration Register

Contains details of the hardware configuration of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0050

Access type

RO

Reset value

xxxx xxxx x0xx x000 0000 0000 0000 000x xxxx xxx0 xx0x x0xx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-11: ext_clustercfr bit assignments

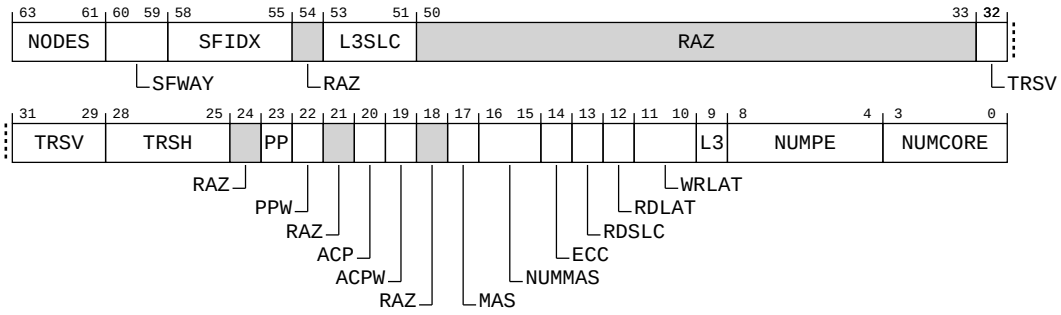


Table B-12: CLUSTERCFR bit descriptions

Bits	Name	Description	Reset
[63:61]	NODES	Number of transport nodes. **0b000** Direct connect. **0b001** One node. **0b010** Two nodes. **0b011** Three nodes. **0b100** Four nodes. **0b101** Eight nodes.	xxx
[60:59]	SFWAY	Number of Snoop Filter ways. **0b00** 4 ways **0b01** 6 ways **0b10** 8 ways **0b11** 12 ways	xx
[58:55]	SFIDX	Log2 of the number of snoop filter indexes.	xxxx
[54]	**RAZ**	Reserved	**RAZ**

Bits	Name	Description	Reset
[53:51]	L3SLC	Number of L3 cache slices. **0b000** Eight L3 cache slices. **0b001** One L3 cache slice. **0b010** Two L3 cache slices. **0b100** Four L3 cache slices.	xxx
[50:33]	**RAZ**	Reserved	**RAZ**
[32:29]	TRSV	Transport register slices, vertical. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx

Bits	Name	Description	Reset
[28:25]	TRSH	Transport register slices, horizontal. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx
[24]	**RAZ**	Reserved	**RAZ**
[23]	PP	Peripheral port presence. **0b0** No peripheral port present **0b1** Peripheral port present	x
[22]	PPW	Peripheral port width. **0b0** 64 bit data width **0b1** 256 bit data width	x
[21]	**RAZ**	Reserved	**RAZ**
[20]	ACP	ACP interface presence. **0b0** No ACP interface present **0b1** ACP interface present	x
[19]	ACPW	ACP interface width. **0b0** 128 bit data width **0b1** 256 bit data width	x
[18]	**RAZ**	Reserved	**RAZ**

Bits	Name	Description	Reset
[17]	MAS	Master bus interface type. **0b0** AXI interface **0b1** CHI interface	x
[16:15]	NUMMAS	Number of Master interfaces. **0b00** One master **0b01** Two masters **0b10** Three masters **0b11** Four masters	xx
[14]	ECC	SCU-L3 ECC configuration. **0b0** SCU-L3 is configured with no ECC **0b1** SCU-L3 is configured with ECC	x
[13]	RDSLC	L3 data RAM read register slice. **0b0** No register slice present **0b1** Register slice present	x
[12]	RDLAT	L3 Data RAM read latency. **0b0** Two cycle output delay from L3 data RAMs **0b1** Three cycle output delay from L3 data RAMs	x
[11:10]	WRLAT	L3 Data RAM write latency. **0b00** One cycle input delay from L3 data RAMs **0b01** Two cycle input delay from L3 data RAMs **0b10** Two cycle input delay plus a one cycle hold	xx
[9]	L3	L3 cache presence. **0b0** No L3 cache present **0b1** L3 cache present	x

Bits	Name	Description	Reset
[8:4]	NUMPE	Number of PEs present in the cluster. For single threaded cores, this number will be the same as bits [3:0]; for multi-threaded cores it will be larger.	5{x}
[3:0]	NUMCORE	Number of cores present in the cluster. **0b0000** One core **0b0001** Two cores **0b0010** Three cores **0b0011** Four cores **0b0100** Five cores **0b0101** Six cores **0b0110** Seven cores **0b0111** Eight cores **0b1000** Nine core **0b1001** Ten cores **0b1010** Eleven cores **0b1011** Twelve cores	xxxx

B.1.1.12 CLUSTERACTLR, Cluster Auxiliary Control Register

These register bits are reserved for Arm test purposes only and must not be used except under direction from Arm.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0058

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-12: ext_clusteractlr bit assignments

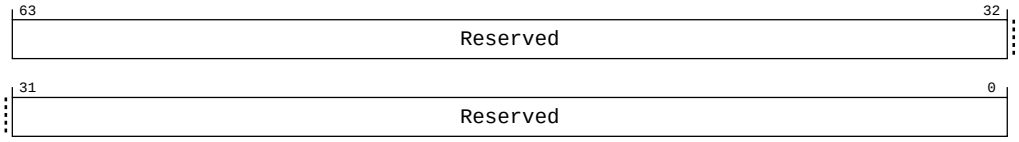


Table B-13: CLUSTERACTLR bit descriptions

Bits	Name	Description	Reset
[63:0]	Reserved	Reserved for Arm internal use	64{x}

B.1.1.13 CLUSTERECTLR, Cluster Extended Control Register

This register should be used for dynamically changing implementation specific control bits.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0060

Access type

RW

Reset value

0000 0000 0000 0000 0011 0100 0000 0000 0000 0000 0000 00xx x000 0101 0101 0010

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-13: ext_clusterectlr bit assignments

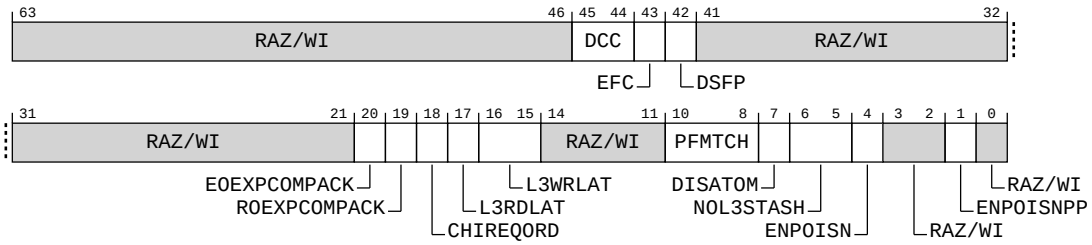


Table B-14: CLUSTERECTLR bit descriptions

Bits	Name	Description	Reset
[63:46]	**RAZ/WI**	Reserved	**RAZ/WI**
[45:44]	DCC	Downstream cache control. Controls whether evictions of clean cachelines send data on the CHI interface. Set this based on whether there is a cache on the path to memory. This bit is **RESO** in direct connect configuration. **0b00** Disables sending data when clean cachelines are evicted. **0b01** Enables sending WriteEvictFull transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **0b10** Enables sending WriteEvictOrEvict transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **0b11** Enables sending WriteEvictOrEvict transactions when Unique Clean or Shared Clean cachelines are evicted. This is the reset value.	0b11

Bits	Name	Description	Reset
[43]	EFC	Eviction flush control. Controls whether hardware cache flushes and DC CISW instructions send data when evicting clean cachelines on the CHI interface. This bit is **RES0** in direct connect configuration. **0b0** Disables sending data when hardware cache flushes or DC CISW instructions evict a clean cacheline. Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP). This is the reset value. **0b1** Sending of data when hardware cache flushes or DC CISW instructions evict clean cachelines is controlled by Downstream Cache Control (DCC). Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP).	0b0
[42]	DSFP	Downstream snoop filter present. Enables sending Evict transactions on the CHI interface when clean cachelines are evicted without data. Enable this if there is at least one snoop filter in the path to memory. This bit is **RES0** in direct connect configuration. **0b0** Disables sending Evict transactions when clean cachelines are evicted without data. **0b1** Enables sending of Evict transactions when clean cachelines are evicted without data. This is the reset value.	0b1
[41:21]	**RAZ/WI**	Reserved	**RAZ/WI**
[20]	EOEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Endpoint Order transactions to the system **0b0** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=1	0b0
[19]	ROEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Request Order transactions to the system **0b0** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=1	0b0
[18]	CHIREQORD	Allow Request Order on CHI ports. Enables the use of Request Order when sending Non-snoopable CHI transactions to the system for Dev-R and Normal NC memory. **0b0** Disables sending Request Order on the CHI interface to the system. Will send No Order instead. **0b1** Enables sending Request Order on the CHI interface to the system for Non-snoopable transactions.	0b0
[17]	L3RDLAT	L3 data RAM read (output) latency. This bit is **RES0** in direct connect configuration. **0b0** The L3 data RAM output latency is 2 cycles. **0b1** The L3 data RAM output latency is 3 cycles.	x

Bits	Name	Description	Reset
[16:15]	L3WRLAT	L3 data RAM write (input) latency. This bit is **RES0** in direct connect configuration. **0b00** The L3 data RAM input latency is 1 cycle with an additional hold cycle. **0b01** The L3 data RAM input latency is 2 cycles without an additional hold cycle. **0b10** The L3 data RAM input latency is 2 cycles with an additional hold cycle. This is only usable if the L3 data RAM output latency is 3 cycles.	xx
[14:11]	**RAZ/WI**	Reserved	**RAZ/WI**
[10:8]	PFMTCH	Prefetch matching delay. Controls the amount of tie a prefetch waits for a possible match with a later read. Encoded as powers of 2, from 1-128. This bit is **RES0** in direct connect configuration. **0b000** Wait for 1 cycle. **0b001** Wait for 2 cycles. **0b010** Wait for 4 cycles. **0b011** Wait for 8 cycles. **0b100** Wait for 16 cycles. **0b101** Wait for 32 cycles. **0b110** Wait for 64 cycles. **0b111** Wait for 128 cycles.	0b101
[7]	DISATOM	Disable cacheable shareable atomics being sent to the interconnect. This bit is **RES0** in direct connect configuration. **0b0** Cacheable shareable atomics will be sent to the interconnect if the BROADCASTATOMIC pin is set. **0b1** Cacheable shareable atomics will be handled inside the cluster.	0b0
[6:5]	NOL3STASH	CPU StashOnce request behaviour when L3 is not present or powered down. This bit is **RES0** in direct connect configuration. **0b00** Stashes are sent out to the interconnect, if supported. **0b01** Normal read request sent to interconnect. **0b10** StashOnce has no effect.	0b10

Bits	Name	Description	Reset
[4]	ENPOISN	Interconnect data poisoning support for the CHI Master(s). This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES0** in direct connect configuration. **0b0** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **0b1** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	0b1
[3:2]	**RAZ/WI**	Reserved	**RAZ/WI**
[1]	ENPOISNPP	Interconnect data poisoning support for the CHI peripheral port. This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES0** in direct connect configuration. **0b0** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **0b1** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	0b1
[0]	**RAZ/WI**	Reserved	**RAZ/WI**

B.1.1.14 CLUSTERCFR2, Cluster Configuration Register 2

Contains details of the hardware configuration of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0068

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-14: ext_clustercfr2 bit assignments

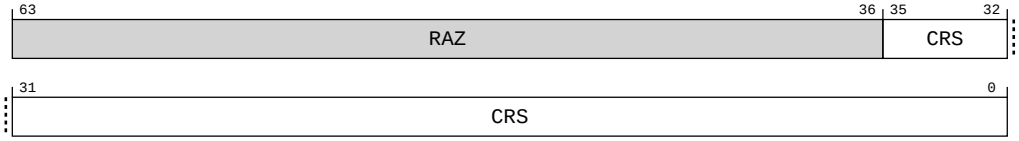


Table B-15: CLUSTERCFR2 bit descriptions

Bits	Name	Description	Reset
[63:36]	RAZ	Reserved	RAZ
[35:0]	CRS	Core register slices. Each three bits represents a core, with [2:0] for core 0 up to [35:33] for core 11. **0b00000000000000000000000000000000000000** No register slices **0b00000000000000000000000000000000000001** One register slice **0b00000000000000000000000000000000000010** Two register slices **0b00000000000000000000000000000000000011** Three register slices **0b00000000000000000000000000000000000100** Four register slices **0b00000000000000000000000000000000000101** Five register slices **0b00000000000000000000000000000000000110** Six register slices **0b00000000000000000000000000000000000111** Seven register slices	`36{x}`

B.1.2 External MPAM register summary

The *Memory System Resource Partitioning and Monitoring* (MPAM) registers are only accessible from memory-mapped accesses on the utility bus. You can access the MPAM registers from both the Secure and Non-secure address space.

The summary table provides an overview of all the MPAM registers that are accessed externally (memory-mapped) from the utility bus of the DSU-110. Individual register descriptions provide detailed information.


Note

- The MPAM Registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- Any address that is not documented is treated as **RAZ/WI**.
- The base address for the MPAM registers is `0x010000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-16: MPAM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0000	MPAMF_IDR	—	32-bit	MPAM Features Identification Register	No
0x0000	MPAMF_IDR	—	32-bit	MPAM Features Identification Register	No
0x0008	MPAMF_SIDR	—	32-bit	MPAM Features Secure Identification Register	No
0x0018	MPAMF_IIDR	—	32-bit	MPAM Implementation Identification Register	No
0x0018	MPAMF_IIDR	—	32-bit	MPAM Implementation Identification Register	No
0x0020	MPAMF_AIDR	—	32-bit	MPAM Architecture Identification Register	No
0x0020	MPAMF_AIDR	—	32-bit	MPAM Architecture Identification Register	No
0x0030	MPAMF_CPOR_IDR	—	32-bit	MPAM Features Cache Portion Partitioning ID register	No
0x0030	MPAMF_CPOR_IDR	—	32-bit	MPAM Features Cache Portion Partitioning ID register	No
0x00F0	MPAMF_ECR	—	32-bit	MPAM Error Control Register	No
0x00F0	MPAMF_ECR	—	32-bit	MPAM Error Control Register	No
0x00F8	MPAMF_ESR	—	32-bit	MPAM Error Status Register	No
0x00F8	MPAMF_ESR	—	32-bit	MPAM Error Status Register	No
0x0100	MPAMCFG_PART_SEL	—	32-bit	MPAM Partition Configuration Selection Register	No
0x0100	MPAMCFG_PART_SEL	—	32-bit	MPAM Partition Configuration Selection Register	No
0x1000	MPAMCFG_CPBM_s	—	32-bit	MPAM Cache Portion Bitmap Partition Configuration Register for Secure PARTIDs	No
0x1000	MPAMCFG_CPBM_ns	—	32-bit	MPAM Cache Portion Bitmap Partition Configuration Register for Non-secure PARTIDs	No

B.1.2.1 MPAMF_IDR, MPAM Features Identification Register

Indicates which memory partitioning and monitoring features are present on this MSC. MPAMF_IDR_s indicates the MPAM features accessed from the Secure MPAM feature page. MPAMF_IDR_ns indicates the MPAM features accessed from the Non-secure MPAM feature page.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0000,0x0000

Access type

RO

Reset value

000x 0010 0000 0001 0000 0000 0011 1111

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-15: ext_mpamf_idr bit assignments

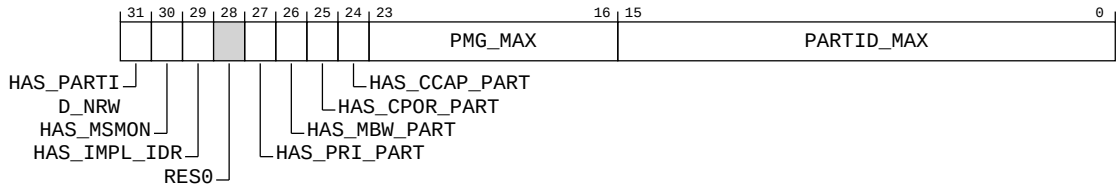


Table B-17: MPAMF_IDR bit descriptions

Bits	Name	Description	Reset
[31]	HAS_PARTID_NRW	Has PARTID narrowing. **0b0** Does not have ext-MPAMF_PARTID_NRW_IDR, ext-MPAMCFG_INTPARTID or intPARTID mapping support.	0b0

Bits	Name	Description	Reset
[30]	HAS_MSMON	Has resource monitors. Indicates whether this MSC has MPAM resource monitors. **0b0** Does not support MPAM resource monitoring by groups or ext-MPAMF_MSMON_IDR.	0b0
[29]	HAS_IMPL_IDR	Has ext-MPAMF_IMPL_IDR. Indicates whether this MSC has the implementation-specific MPAM features register, ext-MPAMF_IMPL_IDR. **0b0** Does not have ext-MPAMF_IMPL_IDR.	0b0
[28]	**RES0**	Reserved	**RES0**
[27]	HAS_PRI_PART	Has priority partitioning. Indicates whether this MSC implements MPAM priority partitioning and ext-MPAMF_PRI_IDR. **0b0** Does not support priority partitioning or have ext-MPAMF_PRI_IDR.	0b0
[26]	HAS_MBW_PART	Has memory bandwidth partitioning. Indicates whether this MSC implements MPAM memory bandwidth partitioning and MPAMF_MBW_IDR. **0b0** Does not support memory bandwidth partitioning or have ext-MPAMF_MBW_IDR register.	0b0
[25]	HAS_CPOR_PART	Has cache portion partitioning. Indicates whether this MSC implements MPAM cache portion partitioning and ext-MPAMF_CPOR_IDR. **0b1** Has ext-MPAMF_CPOR_IDR and ext-MPAMCFG_CPBM registers.	0b1
[24]	HAS_CCAP_PART	Has cache capacity partitioning. Indicates whether this MSC implements MPAM cache capacity partitioning and the MPAMF_CCAP_IDR and MPAMCFG_CMAX registers. **0b0** Does not support cache capacity partitioning or have ext-MPAMF_CCAP_IDR and ext-MPAMCFG_CMAX registers.	0b0
[23:16]	PMG_MAX	Maximum value of Non-secure PMG supported by this component. **0b00000001** Supports 2 Non-secure PMGs.	0x01
[15:0]	PARTID_MAX	Maximum value of Non-secure PARTID supported by this component. **0b0000000000111111** Supports 64 Non-secure PARTIDs.	0x003F

B.1.2.2 MPAMF_SIDR, MPAM Features Secure Identification Register

The MPAMF_SIDR is a 32-bit read-only register that indicates the maximum Secure PARTID and Secure PMG on this MSC.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offset

0x0008

Access type

RO

Reset value

xxxx xxxx 0000 0001 0000 0000 0000 0111

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-16: ext_mpamf_sidr bit assignments

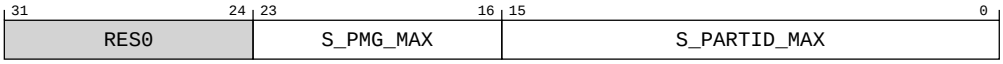


Table B-18: MPAMF_SIDR bit descriptions

Bits	Name	Description	Reset
[31:24]	**RES0**	Reserved	**RES0**
[23:16]	S_PMG_MAX	Maximum value of Secure PMG supported by this component. **0b00000001** Supports 2 Secure PMGs.	0x01
[15:0]	S_PARTID_MAX	Maximum value of Secure PARTID supported by this component. **0b0000000000000111** Supports 8 Secure PARTIDs.	0x0007

B.1.2.3 MPAMF_IIDR, MPAM Implementation Identification Register

Uniquely identifies the MSC implementation by the combination of implementer, product ID, variant and revision.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0018,0x0018

Access type

RO

Reset value

0100 1110 1000 0100 0000 0100 0011 1011

Bit descriptions

Figure B-17: ext_mpamf_iidr bit assignments

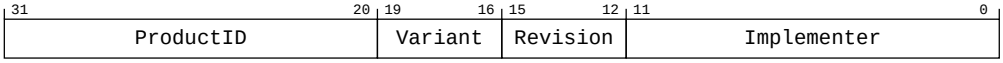


Table B-19: MPAMF_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Value identifying the MPAM Memory System Component. **0b010011101000** DSU-110 Cluster MPAM.	0x4E8
[19:16]	Variant	Value used to distinguish product variants, or major revisions of the product. **0b0000** Product variant 0. **0b0001** Product variant 1. **0b0010** Product variant 2. **0b0011** Product variant 3. **0b0100** Product variant 4.	0b0100

Bits	Name	Description	Reset
[15:12]	Revision	Value used to distinguish minor revisions of the product. **0b0000** Product revision 0. **0b0001** Product revision 1. **0b0010** Product revision 2. **0b0011** Product revision 3. **0b0100** Product revision 4.	0b0000
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the MPAM Memory System Component. For an Arm implementation, bits[11:0] are 0x43B. **0b010000111011** Arm implementation.	0x43B

B.1.2.4 MPAMF_AIDR, MPAM Architecture Identification Register

Identifies the version of the MPAM architecture that this MSC implements.

Note: The following values are defined for bits [7:0]:

- 0x01 == MPAM architecture v0.1
- 0x10 == MPAM architecture v1.0
- 0x11 == MPAM architecture v1.1

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0020,0x0020

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-18: ext_mpamf_aidr bit assignments

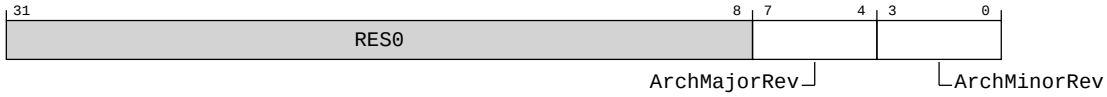


Table B-20: MPAMF_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	ArchMajorRev	Major revision of the MPAM architecture implemented by the MSC. **0b0001** MPAM major version 1.	0b0001
[3:0]	ArchMinorRev	Minor revision of the MPAM architecture implemented by the MSC. **0b0000** MPAM minor version 0.	0b0000

B.1.2.5 MPAMF_CPOR_IDR, MPAM Features Cache Portion Partitioning ID register

Indicates the number of bits in ext-MPAMCFG_CPBM for this MSC. MPAMF_CPOR_IDR_s indicates the number of bits in the Secure instance of ext-MPAMCFG_CPBM. MPAMF_CPOR_IDR_ns indicates the number of bits in the Non-secure instance of ext-MPAMCFG_CPBM.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0030,0x0030

Access type

RO

Reset value

xxxx xxxx xxxx xxxx 0000 0000 0000 1000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-19: ext_mpamf_cpor_idr bit assignments

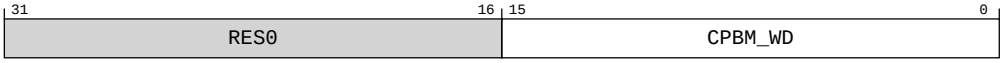


Table B-21: MPAMF_CPOR_IDR bit descriptions

Bits	Name	Description	Reset
[31:16]	**RES0**	Reserved	**RES0**
[15:0]	CPBM_WD	Number of bits in the cache portion partitioning bit map of this device. See ext-MPAMCFG_CPBM. **0b0000000000001000** Supports 8 cache portion partitioning bits.	0x0008

B.1.2.6 MPAMF_ECR, MPAM Error Control Register

MPAMF_ECR is a 32-bit read-write register that controls MPAM error interrupts for this MSC. MPAMF_ECR_s controls Secure MPAM error handling. MPAMF_ECR_ns controls Non-secure MPAM error handling.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x00F0,0x00F0

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-20: ext_mpamf_ecr bit assignments

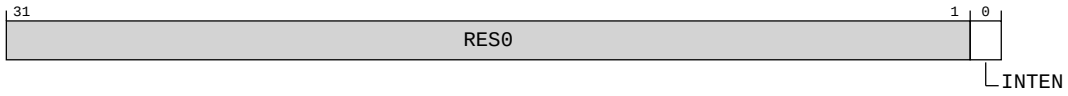


Table B-22: MPAMF_ECR bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	INTEN	Interrupt Enable. **0b0** MPAM error interrupts are not generated. **0b1** MPAM error interrupts are generated.	0b0

B.1.2.7 MPAMF_ESR, MPAM Error Status Register

Indicates MPAM error status for this MSC. MPAMF_ESR_s reports Secure MPAM errors. MPAMF_ESR_ns reports Non-secure MPAM errors.

Software should write this register after reading the status of an error to reset ERRCODE to 0x0000 and OVRWR to 0 so that future errors are not reported with OVRWR set.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x00F8,0x00F8

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-21: ext_mpamf_esr bit assignments

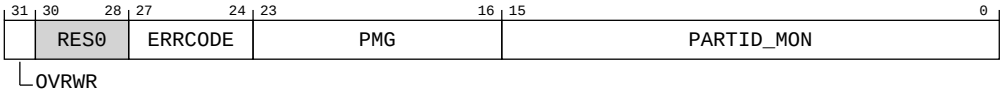


Table B-23: MPAMF_ESR bit descriptions

Bits	Name	Description	Reset
[31]	OVRWR	Overwritten. If 0 and ERRCODE == 0b0000, no errors have occurred. If 0 and ERRCODE is non-zero, a single error has occurred and is recorded in this register. If 1 and ERRCODE is non-zero, multiple errors have occurred and this register records the most recent error. The state where this bit is 1 and ERRCODE is 0 must not be produced by hardware and is only reached when software writes this combination into this register.	x
[30:28]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[27:24]	ERRCODE	Error code. **0b0000** No error. **0b0001** PARTID_SEL_Range. **0b0010** Req_PARTID_Range. **0b0011** MSMONCFG_ID_RANGE. **0b0100** Req_PMG_Range. **0b0101** Monitor_Range. **0b0110** intPARTID_Range. **0b0111** Unexpected_INTERNAL. **0b1000** Reserved. **0b1001** Reserved. **0b1010** Reserved. **0b1011** Reserved. **0b1100** Reserved. **0b1101** Reserved. **0b1110** Reserved. **0b1111** Reserved.	xxxx
[23:16]	PMG	Program monitoring group. Set to the PMG on an error that captures PMG. Otherwise, set to 0x00 on an error that does not capture PMG.	8{x}
[15:0]	PARTID_MON	PARTID or monitor. Set to the PARTID on an error that captures PARTID. Set to the monitor index on an error that captures MON. On an error that captures neither PARTID nor MON, this field is set to 0x0000.	16{x}

B.1.2.8 MPAMCFG_PART_SEL, MPAM Partition Configuration Selection Register

Selects a partition ID to configure. MPAMCFG_PART_SEL_s selects a Secure PARTID to configure. MPAMCFG_PART_SEL_ns selects a Non-secure PARTID to configure.

After setting this register with a PARTID, software (usually a hypervisor) can perform a series of accesses to MPAMCFG registers to configure parameters for MPAM resource controls to use when requests have that PARTID.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0100,0x0100

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-22: ext_mpamcfg_part_sel bit assignments

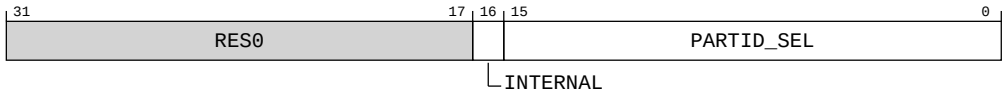


Table B-24: MPAMCFG_PART_SEL bit descriptions

Bits	Name	Description	Reset
[31:17]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[16]	INTERNAL	Internal PARTID. This field is **RAZ/WI**. **0b0** PARTID_SEL is interpreted as a request PARTID and ignored except for use with ext-MPAMCFG_INTPARTID register access.	x
[15:0]	PARTID_SEL	Selects the partition ID to configure. Reads and writes to other MPAMCFG registers are indexed by PARTID_SEL and by the NS bit used to access MPAMCFG_PART_SEL to access the configuration for a single partition.	16{x}

B.1.2.9 MPAMCFG_CPBM_s, MPAM Cache Portion Bitmap Partition Configuration Register for Secure PARTIDs

The MPAMCFG_CPBM register is a read-write register that configures the cache portions that a PARTID is allowed to allocate. After setting ext-MPAMCFG_PART_SEL with a PARTID, software (usually a hypervisor) writes to the MPAMCFG_CPBM register to configure which cache portions the PARTID is allowed to allocate.

MPAMCFG_CPBM_s controls cache portions for the Secure PARTID selected by the Secure instance of ext-MPAMCFG_PART_SEL. MPAMCFG_CPBM_ns controls the cache portions for the Non-secure PARTID selected by the Non-secure instance of ext-MPAMCFG_PART_SEL.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offset

0x1000

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 1111 1111

Bit descriptions

Figure B-23: ext_mpamcfg_cpbm_s bit assignments

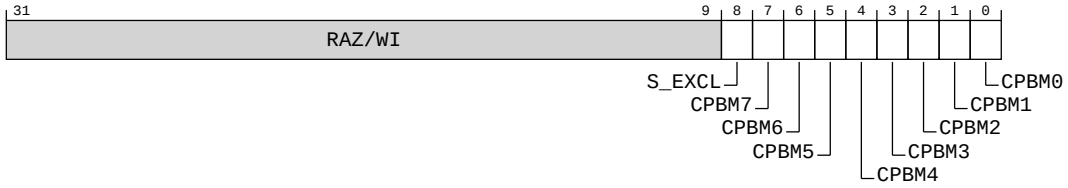


Table B-25: MPAMCFG_CPBM_s bit descriptions

Bits	Name	Description	Reset
[31:9]	**RAZ/WI**	Reserved	**RAZ/WI**
[8]	S_EXCL	Exclusive Secure CPBM enable. If set, all portions enabled in the Secure MPAMCFG_CPBM_s register will prevent corresponding portions enabled in the MPAMCFG_CPBM_ns register from taking effect. **0b0** Each set MPAMCFG_CPBM_s bit has no effect on the corresponding MPAMCFG_CPBM_ns bit. **0b1** Each set MPAMCFG_CPBM_s bit masks the corresponding MPAMCFG_CPBM_ns bit from taking effect.	0b0
[7]	CPBM7	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[6]	CPBM6	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[5]	CPBM5	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

Bits	Name	Description	Reset
[4]	CPBM4	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[3]	CPBM3	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[2]	CPBM2	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[1]	CPBM1	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[0]	CPBM0	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

B.1.2.10 MPAMCFG_CPBM_ns, MPAM Cache Portion Bitmap Partition Configuration Register for Non-secure PARTIDs

The MPAMCFG_CPBM register is a read-write register that configures the cache portions that a PARTID is allowed to allocate. After setting ext-MPAMCFG_PART_SEL with a PARTID, software (usually a hypervisor) writes to the MPAMCFG_CPBM register to configure which cache portions the PARTID is allowed to allocate.

MPAMCFG_CPBM_s controls cache portions for the Secure PARTID selected by the Secure instance of ext-MPAMCFG_PART_SEL. MPAMCFG_CPBM_ns controls the cache portions for the Non-secure PARTID selected by the Non-secure instance of ext-MPAMCFG_PART_SEL.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

MPAM

Register offset

0x1000

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 1111 1111

Bit descriptions

Figure B-24: ext_mpamcfg_cpbm_ns bit assignments

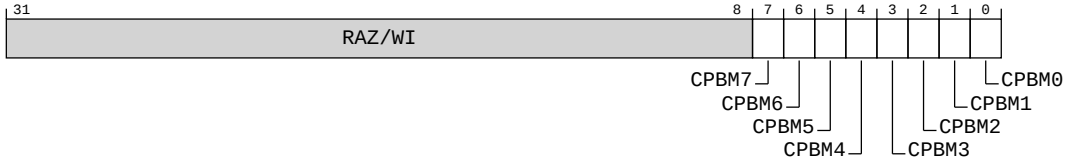


Table B-26: MPAMCFG_CPBM_ns bit descriptions

Bits	Name	Description	Reset
[31:8]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[7]	CPBM7	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[6]	CPBM6	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[5]	CPBM5	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[4]	CPBM4	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[3]	CPBM3	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

Bits	Name	Description	Reset
[2]	CPBM2	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[1]	CPBM1	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[0]	CPBM0	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

B.1.3 External cluster RAS register summary

The cluster *Reliability, Availability, and Serviceability* (RAS) registers are memory-mapped onto the utility bus. You can only access these registers from the Secure address space.

The summary table provides an overview of all the cluster RAS registers in DSU-110. Individual register descriptions provide detailed information.


Note

- The cluster RAS registers are treated as **RAZ/WI** if a Non-secure access is made to them.
- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- The base address for the cluster RAS registers is `0x020000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-27: CLUSTERRAS registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	CLUSTERRAS_ERR0FR	—	64-bit	Error Record Feature Register	No
0x008	CLUSTERRAS_ERR0CTLR	—	64-bit	Error Record Control Register	No
0x010	CLUSTERRAS_ERR0STATUS	—	64-bit	Error Record Primary Status Register	No
0x018	CLUSTERRAS_ERR0ADDR	—	64-bit	Error Record Address Register	No
0x020	CLUSTERRAS_ERR0MISC0	—	64-bit	Error Record Miscellaneous Register 0	No
0x028	CLUSTERRAS_ERR0MISC1	—	64-bit	Error Record Miscellaneous Register 1	No
0x030	CLUSTERRAS_ERR0MISC2	—	64-bit	Error Record Miscellaneous Register 2	No
0x038	CLUSTERRAS_ERR0MISC3	—	64-bit	Error Record Miscellaneous Register 3	No
0x800	CLUSTERRAS_ERR0PFGF	—	64-bit	Pseudo-fault Generation Feature Register	No
0x808	CLUSTERRAS_ERR0PFGCTL	—	64-bit	Pseudo-fault Generation Control Register	No
0x810	CLUSTERRAS_ERR0PFGCDN	—	64-bit	Pseudo-fault Generation Countdown Register	No
0xE00	CLUSTERRAS_ERRGSR	—	64-bit	Error Group Status Register	No
0xE10	CLUSTERRAS_ERRIIDR	—	32-bit	Implementation Identification Register	No
0xFA8	CLUSTERRAS_ERRDEVAFF	—	64-bit	Device Affinity Register	No
0xFBC	CLUSTERRAS_ERRDEVARCH	—	32-bit	Device Architecture Register	No
0xFC8	CLUSTERRAS_ERRDEVID	—	32-bit	Device Configuration Register	No
0xFD0	CLUSTERRAS_ERRPIDR4	—	32-bit	Peripheral Identification Register 4	No
0xFD4	CLUSTERRAS_ERRPIDR5	—	32-bit	Peripheral Identification Register 5	No
0xFD8	CLUSTERRAS_ERRPIDR6	—	32-bit	Peripheral Identification Register 6	No
0xFDC	CLUSTERRAS_ERRPIDR7	—	32-bit	Peripheral Identification Register 7	No
0xFE0	CLUSTERRAS_ERRPIDR0	—	32-bit	Peripheral Identification Register 0	No
0xFE4	CLUSTERRAS_ERRPIDR1	—	32-bit	Peripheral Identification Register 1	No
0xFE8	CLUSTERRAS_ERRPIDR2	—	32-bit	Peripheral Identification Register 2	No
0xFEC	CLUSTERRAS_ERRPIDR3	—	32-bit	Peripheral Identification Register 3	No
0xFF0	CLUSTERRAS_ERRCIDR0	—	32-bit	Component Identification Register 0	No
0xFF4	CLUSTERRAS_ERRCIDR1	—	32-bit	Component Identification Register 1	No
0xFF8	CLUSTERRAS_ERRCIDR2	—	32-bit	Component Identification Register 2	No
0xFFC	CLUSTERRAS_ERRCIDR3	—	32-bit	Component Identification Register 3	No

B.1.3.1 CLUSTERRAS_ERR0FR, Error Record Feature Register

Defines which of the common architecturally-defined features are implemented by the node and, of the implemented features, which are software programmable.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x000

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00 1001 0000 1010 1001 1010 0110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-25: ext_clusterras_err0fr bit assignments

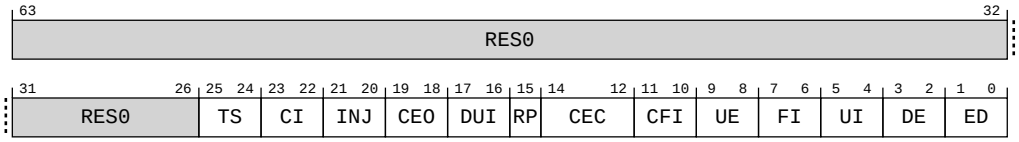


Table B-28: CLUSTERRAS_ERR0FR bit descriptions

Bits	Name	Description	Reset
[63:26]	**RES0**	Reserved	**RES0**
[25:24]	TS	Timestamp Extension. Not implemented and treated as **RAZ/WI**. **0b00** The node does not support a timestamp register. All other values are reserved.	0b00
[23:22]	CI	Critical error interrupt. Indicates whether the critical error interrupt and associated controls are implemented. **0b10** Critical error interrupt is supported and it can be enabled using associated controls. All other values are reserved.	0b10

Bits	Name	Description	Reset
[21:20]	INJ	Fault Injection Extension. Indicates whether the RAS Common Fault Injection Model Extension is implemented. **0b01** The node implements the RAS Common Fault Injection Model Extension. See ext-CLUSTERRAS_ERR0PFGF for more information. All other values are reserved.	0b01
[19:18]	CEO	Corrected Error overwrite. Indicates the behavior when a second Corrected error is detected after a first Corrected error has been recorded by an error record <m> owned by the node. **0b00** Counts Corrected errors. Keeps the previous error syndrome. If the counter overflows then CLUSTERRAS_ERR0STATUS.OF is set to 1. All other values are reserved.	0b00
[17:16]	DUI	Error recovery interrupt for deferred errors. Indicates whether the node implements a control for enabling error recovery interrupts on deferred errors. **0b00** Does not support feature. ext-CLUSTERRAS_ERR0CTLR.DUI is **RES0**. All other values are reserved.	0b00
[15]	RP	Repeat counter. Indicates whether the node implements a repeat Corrected error counter in CLUSTERRAS_ERR0MISC0. **0b1** A first (repeat) counter and a second (other) counter are implemented. The repeat counter is the same size as the primary error counter.	0b1
[14:12]	CEC	Corrected Error Counter. Indicates whether the node implements standard Corrected error counter (CE counter) mechanisms in CLUSTERRAS_ERR0MISC0. **0b010** Implements an 8-bit Corrected error counter in CLUSTERRAS_ERR0MISC0[39:32]. All other values are reserved.	0b010
[11:10]	CFI	Fault handling interrupt for corrected errors. Indicates whether the node implements a control for enabling fault handling interrupts on corrected errors. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.CFI. All other values are reserved.	0b10

Bits	Name	Description	Reset
[9:8]	UE	In-band uncorrected error reporting. Indicates whether the node implements in-band uncorrected error reporting (External aborts), and, if so, whether the node implements controls for enabling and disabling the reporting. **0b01** Feature always enabled. ext-CLUSTERRAS_ERR0CTLR.UE is **RES0**.	0b01
[7:6]	FI	Fault handling interrupt. Indicates whether the node implements a fault handling interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.FI.	0b10
[5:4]	UI	Error recovery interrupt for uncorrected errors. Indicates whether the node implements an error recovery interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.UI.	0b10
[3:2]	DE	Deferred error enable. **0b01** Deferred errors is always enabled.	0b01
[1:0]	ED	Error reporting and logging. Indicates this is the first record owned by the cluster. The cluster implements controls for enabling and disabling error reporting and logging. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.ED. The value 0b11 is reserved.	0b10

B.1.3.2 CLUSTERRAS_ERR0CTLR, Error Record Control Register

The error control register contains enable bits for the node that writes to this record, which:

- Enable error detection and correction.
- Enable an error recovery interrupt.
- Enable a fault handling interrupt.
- Enable error recovery reporting as a read or write error response.
- Enable a critical error interrupt.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x008

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0 xx0x

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-26: ext_clusterras_err0ctlr bit assignments

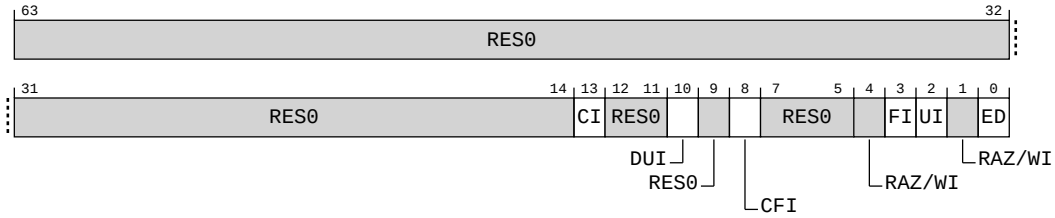


Table B-29: CLUSTERRAS_ERR0CTLR bit descriptions

Bits	Name	Description	Reset
[63:14]	**RES0**	Reserved	**RES0**
[13]	CI	Critical error interrupt enable. When enabled, the critical error interrupt is generated for a critical error condition. **0b0** Critical error interrupt not generated for critical errors. Critical errors are treated as Uncontained errors. **0b1** Critical error interrupt generated for critical errors.	x
[12:11]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[10]	DUI	Error recovery interrupt for deferred errors enable. This control applies to errors arising from both reads and writes. When enabled, an error recovery interrupt is generated for all detected Deferred errors. **0b0** Error recovery interrupt not generated for deferred errors. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error. Access to this field is: RO	x
[9]	**RES0**	Reserved	**RES0**
[8]	CFI	Fault handling interrupt for Corrected errors enable. This control applies to errors arising from both reads and writes. When enabled, the fault handling interrupt is generated when a Corrected error counter overflows and the overflow bit for the counter is set to 1. For more information, see ext-ERR<n>MISC0. **0b0** Fault handling interrupt not generated for Corrected errors. **0b1** Fault handling interrupt generated for Corrected errors. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[7:5]	**RES0**	Reserved	**RES0**
[4]	**RAZ/ WI**	Reserved	**RAZ/ WI**
[3]	FI	Fault handling interrupt enable. This control applies to errors arising from both reads and writes. When enabled, the fault handling interrupt is generated for all detected Corrected errors, Deferred errors, and Uncorrected errors. **0b0** Fault handling interrupt disabled. **0b1** Fault handling interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[2]	UI	Uncorrected error recovery interrupt enable. This control applies to errors arising from both reads and writes. When enabled, the error recovery interrupt is generated for all detected Uncorrected errors that are not deferred. **0b0** Error recovery interrupt disabled. **0b1** Error recovery interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x

Bits	Name	Description	Reset
[1]	**RAZ/WI**	Reserved	**RAZ/WI**
[0]	ED	Error reporting and logging enable. When disabled, the node behaves as if error detection and correction are disabled, and no errors are recorded or signaled by the node. **0b0** Error reporting disabled. **0b1** Error reporting enabled.	x

B.1.3.3 CLUSTERRAS_ERR0STATUS, Error Record Primary Status Register

Contains status information for the error record, including:

- Whether any error has been detected (valid).
- Whether any detected error was not corrected, and returned to a master.
- Whether any detected error was not corrected and deferred.
- Whether an error record has been discarded because additional errors have been detected before the first error was handled by software (overflow).
- Whether any error has been reported.
- Whether the other error record registers contain valid information.
- Whether the error was recorded because poison data was detected or because a corrupt value was detected by an error detection code.
- A Primary error code.
- An **IMPLEMENTATION DEFINED** Extended error code.

Within this register:

- The {AV, V, MV} bits are valid bits that define whether the error record registers are valid.
- The {UE, OF, CE, DE, UET} bits encode the type of error or errors recorded.
- The {CI, ER, PN, IERR, SERR} fields are syndrome fields.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x010

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0xxx 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-27: ext_clusterras_err0status bit assignments

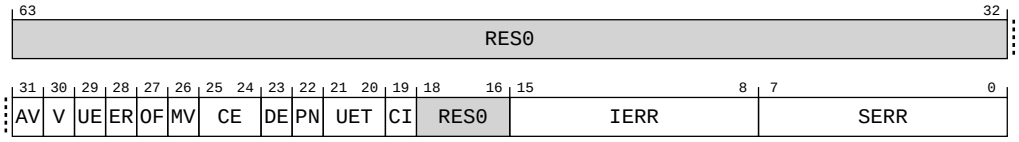


Table B-30: CLUSTERRAS_ERR0STATUS bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	AV	Address Valid. **0b0** ext-CLUSTERRAS_ERR0ADDR not valid. This bit is unimplemented and treated as **RAZ/WI**.	0b0
[30]	V	Status Register Valid. **0b0** CLUSTERRAS_ERR0STATUS not valid. **0b1** CLUSTERRAS_ERR0STATUS valid. At least one error has been recorded. This bit is read/write-one-to-clear.	0b0

Bits	Name	Description	Reset
[29]	UE	Uncorrected error. **0b0** No errors have been detected, or all detected errors have been either corrected or deferred. **0b1** At least one detected error was not corrected and not deferred. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write one to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0
[28]	ER	Error Reported. **0b0** No in-band error (External abort) reported. This bit is unimplemented and treated as **RAZ/WI**.	0b0
[27]	OF	Overflow. Indicates that multiple errors have been detected. This bit is set to 1 when one of the following occurs: • A Corrected error is counted and the counter overflows. • CLUSTERRAS_ERR0STATUS.V was previously set to 1 and a type of error other than a Corrected error is recorded. Otherwise, this bit is unchanged when an error is recorded. A direct write that modifies the counter overflow flag indirectly might set this bit to an **UNKNOWN** value. A direct write to this bit that clears this bit to zero might indirectly set the counter overflow flag to an **UNKNOWN** value. **0b0** Since this bit was last cleared to zero, no error syndrome has been discarded and, if a Corrected error counter is implemented, it has not overflowed. **0b1** Since this bit was last cleared to zero, at least one error syndrome has been discarded or, if a Corrected error counter is implemented, it might have overflowed. If this bit is nonzero, then software must write 1 to this bit, to clear this bit to zero, when clearing CLUSTERRAS_ERR0STATUS.V to 0. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0

Bits	Name	Description	Reset
[26]	MV	Miscellaneous Registers (CLUSTERRAS_ERR0MISC0) Valid. **0b0** CLUSTERRAS_ERR0MISC0 is not valid. **0b1** The contents of CLUSTERRAS_ERR0MISC0 contains additional information for an error recorded by this record. Only CLUSTERRAS_ERR0MISC0 is implemented. CLUSTERRAS_ERR0MISC1,2,3 are treated as **RAZ/WI**. This bit is read/write-one-to-clear.	0b0
[25:24]	CE	Corrected Error. **0b00** No errors were corrected. **0b10** At least one error was corrected. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this field is nonzero, then software must write ones to this field to clear this field to zero. If CLUSTERRAS_ERR0STATUS.V is set to 0, this field is not valid and reads **UNKNOWN**. This field is read/write-one-to-clear. Writing a value other than all-zeros or all-ones sets this field to an **UNKNOWN** value.	0b00
[23]	DE	Deferred Error. **0b0** No errors were deferred. **0b1** At least one error was not corrected and deferred. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. If CLUSTERRAS_ERR0STATUS.V is set to 0, this bit is not valid and reads **UNKNOWN**. This bit is read/write-one-to-clear.	0b0

Bits	Name	Description	Reset
[22]	PN	Poison. **0b0** Uncorrected error or Deferred error recorded because a corrupt value was detected. **0b1** Uncorrected error or Deferred error recorded because a poison value was detected. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if any of the following are true: • CLUSTERRAS_ERR0STATUS.V is set to 0. • CLUSTERRAS_ERR0STATUS.{DE, UE} are both set to 0. This bit is read/write-one-to-clear.	0b0
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. **0b00** Uncorrected error, Uncontainable error (UC). This field is not implemented and is treated as **RAZ/WI**.	0b00
[19]	CI	Critical error. Indicates whether a critical error condition has been recorded. **0b0** No critical error condition recorded. **0b1** Critical error condition recorded. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0
[18:16]	**RES0**	Reserved	**RES0**
[15:8]	IERR	**IMPLEMENTATION DEFINED** Extended error code. Used with any primary error code SERR value. Additional information is placed in the CLUSTERRAS_ERR0MISC0 register. **0b00000000** If SERR == 0x7, indicates a Tag RAM error. Not used with other SERR values. **0b00000010** If SERR == 0x7, indicates a Snoop Filter RAM error. Not used with other SERR values. This field is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0.	0x00

Bits	Name	Description	Reset
[7:0]	SERR	Primary error code. Indicates the type of Primary error. **0b00000000** No error. **0b00000001** **IMPLEMENTATION DEFINED** error. **0b00000010** Data value from (non-associative) internal memory. For example, ECC from on-chip SRAM or buffer. **0b00000011** **IMPLEMENTATION DEFINED** pin. For example, nSEI pin. **0b00000100** Assertion failure. For example, consistency failure. **0b00000101** Error detected on internal data path. For example, parity on ALU result. **0b00000110** Data value from associative memory. For example, ECC error on cache data. **0b00000111** Address/control value from associative memory. For example, ECC error on cache tag. **0b00001000** Data value from a TLB. For example, ECC error on TLB data. **0b00001001** Address/control value from a TLB. For example, ECC error on TLB tag.	0x00

Bits	Name	Description	Reset
[7:0] continued	SERR	**0b00001010** Data value from producer. For example, parity error on write data bus. **0b00001011** Address/control value from producer. For example, parity error on address bus. **0b00001100** Data value from (non-associative) external memory. For example, ECC error in SDRAM. **0b00001101** Illegal address (software fault). For example, access to unpopulated memory. **0b00001110** Illegal access (software fault). For example, byte write to word register. **0b00001111** Illegal state (software fault). For example, device not ready. **0b00010000** Internal data register. For example, parity on a SIMD&FP register. For a PE, all general-purpose, stack pointer, SIMD&FP, and SVE registers are data registers. **0b00010001** Internal control register. For example, Parity on a System register. For a PE, all registers other than general-purpose, stack pointer, SIMD&FP, and SVE registers are control registers. **0b00010010** Error response from slave. For example, error response from cache write-back. **0b00010011** External timeout. For example, timeout on interaction with another node.	0x00
[7:0] continued	SERR	**0b00010100** Internal timeout. For example, timeout on interface within the node. **0b00010101** Deferred error from slave not supported at master. For example, poisoned data received from a slave by a master that cannot defer the error further. All other values are reserved. This field is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0.	0x00

B.1.3.4 CLUSTERRAS_ERR0ADDR, Error Record Address Register

This register is reserved since the implementation does not provide an address with RAS errors.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x018

Access type

RO

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-28: ext_clusterras_err0addr bit assignments

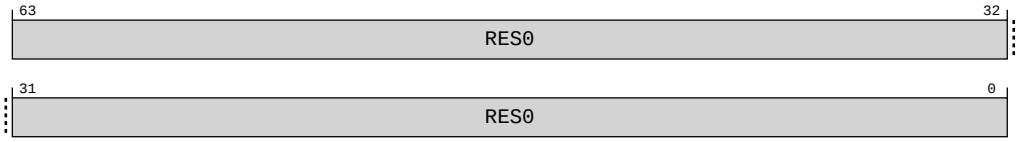


Table B-31: CLUSTERRAS_ERR0ADDR bit descriptions

Bits	Name	Description	Reset
[63:0]	**RES0**	Reserved	**RES0**

B.1.3.5 CLUSTERRAS_ERR0MISC0, Error Record Miscellaneous Register 0

Miscellaneous error syndrome register. The Miscellaneous error syndrome register contains:

- 2 architecturally-defined Corrected error counters with sticky overflow bits,
- Information to identify the FRU in which the error was detected, including Index, Way, Level, Instruction vs. Data fields.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x020

Access type

RW

Reset value

0000 0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-29: ext_clusterras_err0misc0 bit assignments

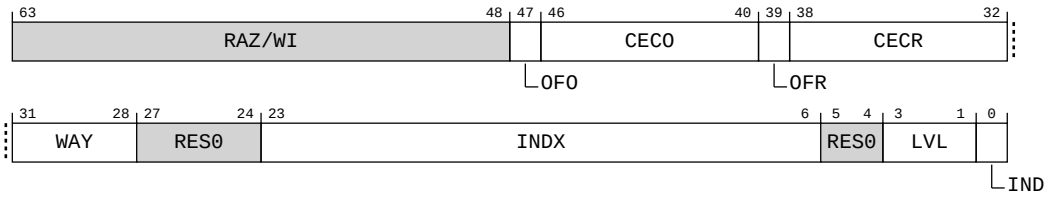


Table B-32: CLUSTERRAS_ERR0MISC0 bit descriptions

Bits	Name	Description	Reset
[63:48]	**RAZ/WI**	Reserved	**RAZ/WI**
[47]	OFO	Sticky overflow bit for Other errors. Set to 1 when the Corrected error count Other (CECO) field is incremented and wraps through zero. **0b0** Other counter has not overflowed. **0b1** Other counter has overflowed. A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an **UNKNOWN** value.	x

Bits	Name	Description	Reset
[46:40]	CECO	Corrected error count for Other errors. The Other error counter increments for all Corrected errors that are not counted by the CECR Repeat error counter due to the syndrome of the new error mismatching against the recorded syndrome of the first Repeat error. Refer to the CECR Repeat error description for fields used to match syndrome. At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.	7{x}
[39]	OFR	Sticky overflow bit for Repeat errors. Set to 1 when the Corrected error count Repeat (CECR) field is incremented and wraps through zero. **0b0** Repeat counter has not overflowed. **0b1** Repeat counter has overflowed. A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an **UNKNOWN** value.	x
[38:32]	CECR	Corrected error count for Repeat errors. The Repeat error counter increments for the first Corrected error and records the syndrome for the error in the fields described below. It also increments for each subsequent Corrected error with a syndrome matching the first error's recorded syndrome, otherwise the error causes an increment to the CECO Other counter. The syndrome is recorded in the following fields: • ext-CLUSTERRAS_ERR0STATUS.IERR • ext-CLUSTERRAS_ERR0STATUS.SERR • ext-CLUSTERRAS_ERR0MISC0.INDX • ext-CLUSTERRAS_ERR0MISC0.WAY The syndrome is matched on a new Corrected error if all of the following are true: • ext-CLUSTERRAS_ERR0STATUS.MV bit is set, • ext-CLUSTERRAS_ERR0STATUS.IERR matches the new error, • ext-CLUSTERRAS_ERR0STATUS.SERR matches the new error, • ext-CLUSTERRAS_ERR0MISC0.INDX matches the new error, • ext-CLUSTERRAS_ERR0MISC0.WAY matches the new error. CLUSTERRAS_ERR0STATUS.MV indicates the validity of the INDX and WAY fields of the CLUSTERRAS_ERR0MISC0 register At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.	7{x}
[31:28]	WAY	L3 Cache Way that contained the error.	xxxx
[27:24]	**RES0**	Reserved	**RES0**
[23:6]	INDX	L3 Cache Index that contained the error.	18{x}
[5:4]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[3:1]	LVL	L3 Cache Level that contained the error. Always `0x2`. **`0b010`** Level 3 cache.	`xxx`
[0]	IND	L3 Cache instruction vs. data cache that contained the error. Always data (`0x0`). **`0b0`** Data cache error.	`x`

B.1.3.6 CLUSTERRAS_ERR0MISC1, Error Record Miscellaneous Register 1

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x028

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-30: ext_clusterras_err0misc1 bit assignments

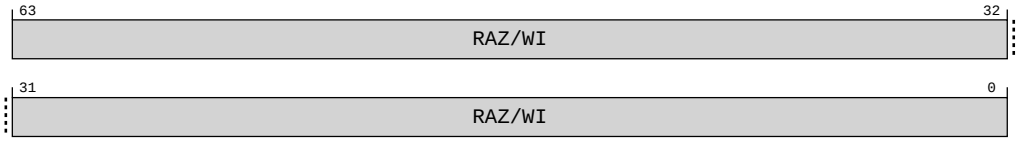


Table B-33: CLUSTERRAS_ERR0MISC1 bit descriptions

Bits	Name	Description	Reset
[63:0]	**RAZ/WI**	Reserved	**RAZ/WI**

B.1.3.7 CLUSTERRAS_ERR0MISC2, Error Record Miscellaneous Register 2

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x030

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-31: ext_clusterras_err0misc2 bit assignments

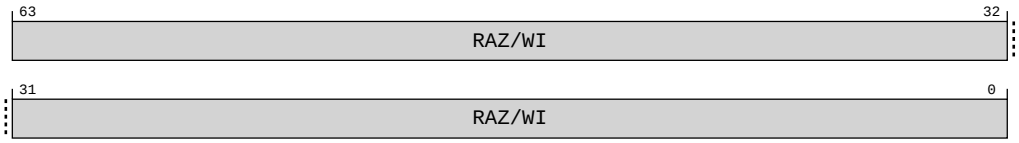


Table B-34: CLUSTERRAS_ERR0MISC2 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

B.1.3.8 CLUSTERRAS_ERR0MISC3, Error Record Miscellaneous Register 3

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x038

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-32: ext_clusterras_err0misc3 bit assignments

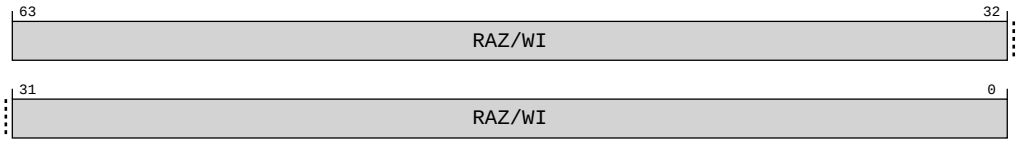


Table B-35: CLUSTERRAS_ERR0MISC3 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

B.1.3.9 CLUSTERRAS_ERR0PFGF, Pseudo-fault Generation Feature Register

Defines which common architecturally-defined fault generation features are implemented.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x800

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx x11x xxxx xxxx xxxx xxx1 0101 0110 0011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-33: ext_clusterras_err0pfgf bit assignments

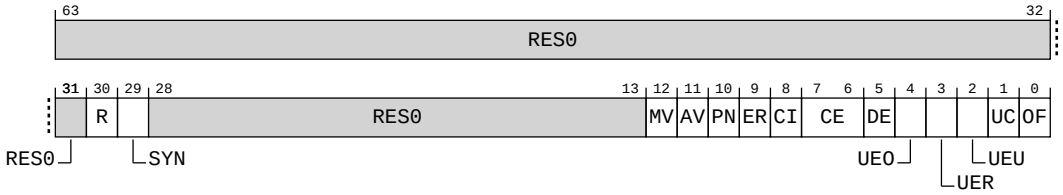


Table B-36: CLUSTERRAS_ERR0PFGF bit descriptions

Bits	Name	Description	Reset
[63:31]	**RES0**	Reserved	**RES0**
[30]	R	Restartable. Support for Error Generation Counter restart mode. **0b1** Feature controllable.	0b1
[29]	SYN	Syndrome. Fault syndrome injection. **0b1** When an injected error is recorded, the node does not update the ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} fields. ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} are writable when ext-CLUSTERRAS_ERR0STATUS.V == 0. **Note:** Software can write intended values into the ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} fields when setting up a fault injection event.	0b1
[28:13]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[12]	MV	Miscellaneous syndrome. Additional syndrome injection. Defines whether software can control all or part of the syndrome recorded in the CLUSTERRAS_ERR0MISC0 register when an injected error is recorded. CLUSTERRAS_ERR0MISC1-3 registers are reserved and unused for this purpose. **0b1** When an injected error is recorded, the node does not update all the syndrome fields in CLUSTERRAS_ERR0MISC0. The node records syndrome in CLUSTERRAS_ERR0MISC0 OFO, CECO, OFR, CECR, WAY, INDX, LVL, and IND fields and sets ext-CLUSTERRAS_ERR0STATUS.MV to 1. CLUSTERRAS_ERR0PGFCTL.MV is **RAO**. **Note:** Software can write intended values into the CLUSTERRAS_ERR0MISC0 register when setting up a fault injection event.	0b1
[11]	AV	Address syndrome. Address syndrome injection. Always **RAZ/WI**. **0b0** The node does not support ext-CLUSTERRAS_ERR0ADDR and does not set ext-CLUSTERRAS_ERR0STATUS.AV.	0b0
[10]	PN	Poison flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.PN status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.PN is set to ext-CLUSTERRAS_ERR0PFGCTL.PN.	0b1
[9]	ER	Error Reported flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.ER status flag. **0b0** When an injected error is recorded, the node does not set ext-CLUSTERRAS_ERR0STATUS.ER. This bit reads-as-zero.	0b0
[8]	CI	Critical Error flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.CI status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.CI is set to ext-CLUSTERRAS_ERR0PFGCTL.CI.	0b1
[7:6]	CE	Corrected Error generation. Describes the types of Corrected Error that the fault generation feature of the node can generate. **0b01** The fault generation feature of the node allows generation of a non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERR0STATUS.CE == 0b10.	0b01
[5]	DE	Deferred Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b1** The fault generation feature of the node allows generation of this type of error.	0b1

Bits	Name	Description	Reset
[4]	UEO	Latent or Restartable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b0** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	0b0
[3]	UER	Signaled or Recoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b0** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	0b0
[2]	UEU	Unrecoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b0** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	0b0
[1]	UC	Uncontainable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **0b1** The fault generation feature of the node allows generation of this type of error.	0b1
[0]	OF	Overflow flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.OF status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.OF is set to ext-CLUSTERRAS_ERR0PFGCTL.OF.	0b1

B.1.3.10 CLUSTERRAS_ERR0PFGCTL, Pseudo-fault Generation Control Register

Enables controlled fault generation.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x808

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-34: ext_clusterras_err0pfgctl bit assignments

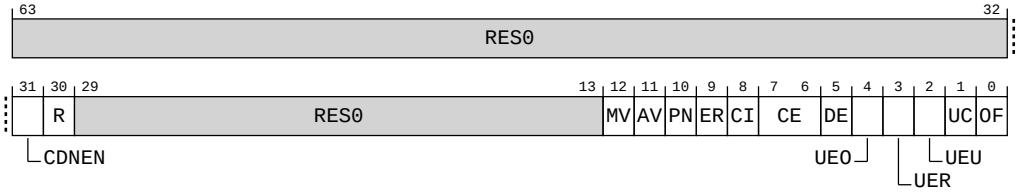


Table B-37: CLUSTERRAS_ERR0PFGCTL bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	CDNEN	Countdown Enable. Controls transfers from the value that is held in the ext-CLUSTERRAS_ERR0PFGCDN into the Error Generation Counter, and enables this counter. **0b0** The Error Generation Counter is disabled. **0b1** The Error Generation Counter is enabled. On a write of 1 to this bit, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.	0b0
[30]	R	Restart. Controls whether, on reaching zero, the Error Generation Counter restarts from the ext-CLUSTERRAS_ERR0PFGCDN value, or stops. **0b0** On reaching 0, the Error Generation Counter stops. **0b1** On reaching 0, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.	x
[29:13]	**RES0**	Reserved	**RES0**
[12]	MV	Miscellaneous syndrome. The value that is written to ext-CLUSTERRAS_ERR0STATUS.MV when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.MV is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.MV is set to 1 when an injected error is recorded.	x

Bits	Name	Description	Reset
[11]	AV	Address syndrome. The value that is written to ext-CLUSTERRAS_ERR0STATUS.AV when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.AV is set to 0 when an injected error is recorded. This bit is **RES0**. Access to this field is: **RES0**	x
[10]	PN	Poison flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.PN when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.PN is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.PN is set to 1 when an injected error is recorded.	x
[9]	ER	Error Reported flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.ER when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.ER is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.ER is set to 1 when an injected error is recorded. This bit is **RES0**. Access to this field is: **RES0**	x
[8]	CI	Critical Error flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.CI when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.CI is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.CI is set to 1 when an injected error is recorded.	x
[7:6]	CE	Corrected Error generation enable. Controls the type of Corrected Error condition that might be generated. **0b00** No error of this type is generated. **0b01** A non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERR0STATUS.CE == 0b10, might be generated when the Error Generation Counter decrements to zero. The set of permitted values for this field is defined by ext-CLUSTERRAS_ERR0PFGF.CE.	xx
[5]	DE	Deferred Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. **0b1** An error of this type might be generated when the Error Generation Counter decrements to zero.	x

Bits	Name	Description	Reset
[4]	UEO	Latent or Restartable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[3]	UER	Signaled or Recoverable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[2]	UEU	Unrecoverable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[1]	UC	Uncontainable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. **0b1** An error of this type might be generated when the Error Generation Counter decrements to zero.	x
[0]	OF	Overflow flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.OF when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.OF is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.OF is set to 1 when an injected error is recorded.	x

B.1.3.11 CLUSTERRAS_ERR0PFGCDN, Pseudo-fault Generation Countdown Register

Generates one of the errors enabled in the corresponding ext-CLUSTERRAS_ERR0PFGCTL register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x810

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-35: ext_clusterras_err0pfgcdn bit assignments

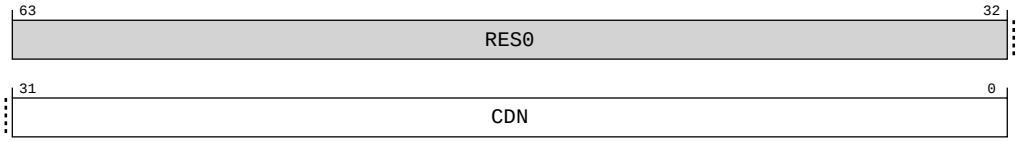


Table B-38: CLUSTERRAS_ERR0PFGCDN bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31:0]	CDN	Countdown value. This field is copied to Error Generation Counter when either: • Software writes ext-CLUSTERRAS_ERR0PFGCTL.CDNEN with 1. • The Error Generation Counter decrements to zero and ext-CLUSTERRAS_ERR0PFGCTL.R == 1. While ext-CLUSTERRAS_ERR0PFGCTL.CDNEN == 1 and the Error Generation Counter is nonzero, the counter decrements by 1 for each cycle. When the counter reaches 0, one of the errors enabled in the ext-CLUSTERRAS_ERR0PFGCTL register is generated. **Note:** The current Error Generation Counter value is not visible to software.	32{x}

B.1.3.12 CLUSTERRAS_ERRGSR, Error Group Status Register

ERRGSR shows the status for the records in the group.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0xE00

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-36: ext_clusterras_errgsr bit assignments

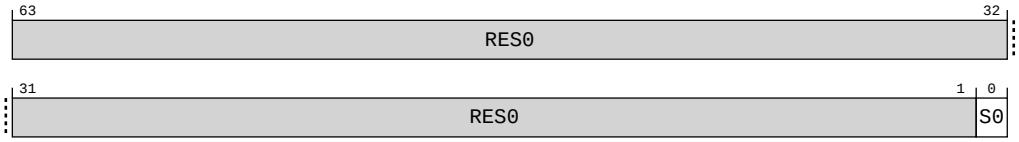


Table B-39: CLUSTERRAS_ERRGSR bit descriptions

Bits	Name	Description	Reset
[63:1]	**RES0**	Reserved	**RES0**
[0]	S0	The status for Error Record 0. A read-only copy of CLUSTERRAS_ERR0STATUS.V. **0b0** No error. **0b1** One or more errors.	0b0

B.1.3.13 CLUSTERRAS_ERRIIDR, Implementation Identification Register

Defines the implementer of the product.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xE10

Access type

RO

Reset value

0100 1110 1000 0100 0001 0100 0011 1011

Bit descriptions

Figure B-37: ext_clusterras_erriidr bit assignments

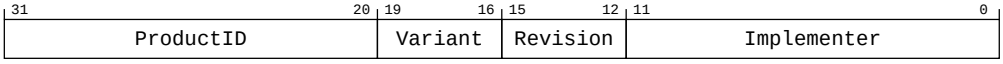


Table B-40: CLUSTERRAS_ERRIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Part number, bits [11:0]. The part number is selected by the designer of the product. **0b010011101000** DSU-110 Cluster RAS. ext-CLUSTERRAS_ERRPIDR0.PART_0 matches bits [7:0] of CLUSTERRAS_ERRIIDR.ProductID and ext-CLUSTERRAS_ERRPIDR1.PART_1 matches bits [11:8] of CLUSTERRAS_ERRIIDR.ProductID.	0x4E8

Bits	Name	Description	Reset
[19:16]	Variant	Product major revision. This field distinguishes product variants or major revisions of the product. **0b0000** Product variant 0. **0b0001** Product variant 1. **0b0010** Product variant 2. **0b0011** Product variant 3. **0b0100** Product variant 4. ext-CLUSTERRAS_ERRPIDR2.REVISION matches CLUSTERRAS_ERRIIDR.Variant.	0b0100
[15:12]	Revision	Product minor revision. This field distinguishes minor revisions of the product. **0b0000** Product revision 0. **0b0001** Product revision 1. **0b0010** Product revision 2. **0b0011** Product revision 3. ext-CLUSTERRAS_ERRPIDR3.REVAND matches CLUSTERRAS_ERRIIDR.Revision.	0b0001
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the RAS component. For an Arm implementation, this field has the value 0x43B. **0b010000111011** JEP106 ID code for Arm Limited. Bits [11:8] contain the JEP106 continuation code of the implementer, and bits [6:0] contain the JEP106 identity code of the implementer. Bit 7 is **RES0**. ext-CLUSTERRAS_ERRPIDR4.DES_2 matches bits [11:8] of CLUSTERRAS_ERRIIDR.Implementer, ext-CLUSTERRAS_ERRPIDR2.DES_1 matches bits [6:4] of CLUSTERRAS_ERRIIDR.Implementer, and ext-CLUSTERRAS_ERRPIDR1.DES_0 matches bits [3:0] of CLUSTERRAS_ERRIIDR.Implementer.	0x43B

B.1.3.14 CLUSTERRAS_ERRDEVAFF, Device Affinity Register

ERRDEVAFF is a copy of part of AArch64-MPIDR_EL1.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0xFA8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 00xx xxx0 xxxx xxxx 1000 0000 1000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-38: ext_clusterras_errdevaff bit assignments

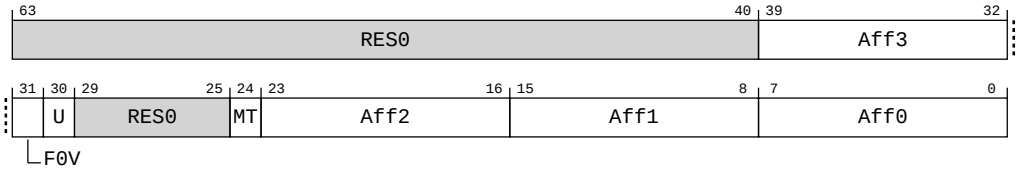


Table B-41: CLUSTERRAS_ERRDEVAFF bit descriptions

Bits	Name	Description	Reset
[63:40]	**RES0**	Reserved	**RES0**
[39:32]	Aff3	Affinity level 3. The AArch64-MPIDR_EL1.Aff3 field, viewed from the highest Exception level of the associated PE or PEs.	8{x}
[31]	F0V	Indicates that the ERRDEVAFF.Aff0 field is valid. **0b0** ERRDEVAFF.Aff0 is not valid, and the PE affinity level is 1, 2 or 3.	0b0

Bits	Name	Description	Reset
[30]	U	Uniprocessor. The AArch64-MPIDR_EL1.U bit viewed from the highest Exception level of the associated PE. **0b0** The PE is part of a multiprocessor system. If ERRDEVAFF.Aff0 is not valid, this bit is not valid and reads as **UNKNOWN**.	0b0
[29:25]	**RES0**	Reserved	**RES0**
[24]	MT	Multithreaded. The AArch64-MPIDR_EL1.MT bit viewed from the highest Exception level of the associated PE. **0b0** Performance of PEs at the lowest affinity level is largely independent. If ERRDEVAFF.Aff0 is not valid, this bit is not valid and reads as **UNKNOWN**.	0b0
[23:16]	Aff2	Affinity level 2. This field is the AArch64-MPIDR_EL1.Aff2 field viewed from the highest Exception level of the associated PE or PEs.	8{x}
[15:8]	Aff1	Affinity level 1. **0b10000000** ERRDEVAFF.Aff2 is valid, and the PE affinity level is 2.	0x80
[7:0]	Aff0	Affinity level 0. **0b10000000** ERRDEVAFF.Aff1 is valid, and the PE affinity level is 1. All other values are reserved.	0x80

B.1.3.15 CLUSTERRAS_ERRDEVARCH, Device Architecture Register

Provides discovery information for the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFBC

Access type

RO

Reset value

0100 0111 0111 0001 0000 1010 0000 0000

Bit descriptions

Figure B-39: ext_clusterras_errdevarch bit assignments

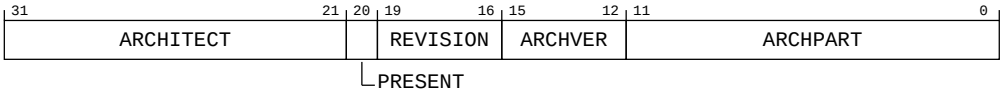


Table B-42: CLUSTERRAS_ERRDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. Defines the architect of the component. Bits [31:28] are the JEP106 continuation code (JEP106 bank ID, minus 1) and bits [27:21] are the JEP106 ID code. **0b01000111011** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	DEVARCH Present. Defines that the DEVARCH register is present. **0b1** Device Architecture information present. This bit is **RAO**.	0b1
[19:16]	REVISION	Revision. Defines the architecture revision of the component. The defined values of this field are: **0b0001** RAS System Architecture v1.1 All other values are reserved.	0b0001
[15:12]	ARCHVER	Architecture Version. Defines the architecture version of the component. The defined values of this field are: **0b0000** RAS System Architecture v1. This field reads as 0b0000. All other values are reserved. ARCHVER and ARCHPART are also defined as a single field, ARCHID, so that ARCHVER is ARCHID[15:12].	0b0000

Bits	Name	Description	Reset
[11:0]	ARCHPART	Architecture Part. Defines the architecture of the component. **0b101000000000** RAS system architecture. This register reads as 0xA00. ARCHVER and ARCHPART are also defined as a single field, ARCHID, so that ARCHPART is ARCHID[11:0].	0xA00

B.1.3.16 CLUSTERRAS_ERRDEVID, Device Configuration Register

Provides discovery information for the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFC8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx 0000 0000 0000 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-40: ext_clusterras_errdevid bit assignments

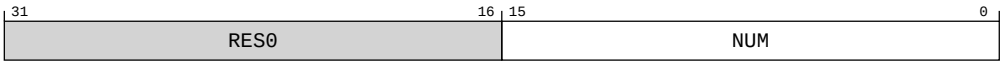


Table B-43: CLUSTERRAS_ERRDEVID bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:0]	NUM	Highest numbered index of the error records in this group, plus one. **0b0000000000000001** One record implemented in this group.	0x0001

B.1.3.17 CLUSTERRAS_ERRPIDR4, Peripheral Identification Register 4

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFD0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-41: ext_clusterras_errpidr4 bit assignments

Table B-44: CLUSTERRAS_ERRPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	SIZE	Size of the component. The distance from the start of the address space used by this component to the end of the component identification registers. **0b0000** The component uses a single 4KB block.	0b0000
[3:0]	DES_2	Designer, JEP106 continuation code. This is the JEDEC-assigned JEP106 bank identifier for the designer of the component, minus 1. The code identifies the designer of the component, which might not be not the same as the implementer of the device containing the component. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B. **Note:** For a component designed by Arm Limited, the JEP106 bank is 5, meaning this field has the value 0x4.	0b0100

B.1.3.18 CLUSTERRAS_ERRPIDR5, Peripheral Identification Register 5

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFD4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-42: ext_clusterras_errpidr5 bit assignments

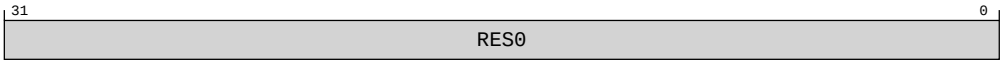


Table B-45: CLUSTERRAS_ERRPIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.3.19 CLUSTERRAS_ERRPIDR6, Peripheral Identification Register 6

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFD8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-43: ext_clusterras_errpidr6 bit assignments

Table B-46: CLUSTERRAS_ERRPIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.3.20 CLUSTERRAS_ERRPIDR7, Peripheral Identification Register 7

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFDC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-44: ext_clusterras_errpidr7 bit assignments

Table B-47: CLUSTERRAS_ERRPIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.3.21 CLUSTERRAS_ERRPIDR0, Peripheral Identification Register 0

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFE0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1110 1000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-45: ext_clusterras_errpidr0 bit assignments

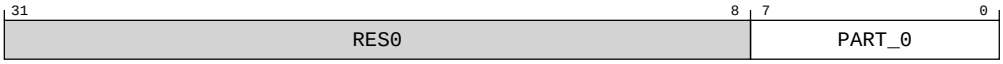


Table B-48: CLUSTERRAS_ERRPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number, bits [7:0]. The part number is a 12-bit part number stored in ext-ERRPIDR1.PART_1 and this field. **0b11101000** DSU-110 Cluster RAS. Bits [7:0] of part number 0x4E8.	0xE8

B.1.3.22 CLUSTERRAS_ERRPIDR1, Peripheral Identification Register 1

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFE4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-46: ext_clusterras_errpidr1 bit assignments

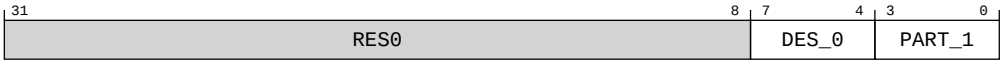


Table B-49: CLUSTERRAS_ERRPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	Designer, JEP106 identification code, bits [3:0]. This field and ext-ERRPIDR2.DES_1 together form the JEDEC-assigned JEP106 identification code for the designer of the component. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B. **Note:** For a component designed by Arm Limited, the JEP106 identification code is `0x3B`.	`0b1011`

Bits	Name	Description	Reset
[3:0]	PART_1	Part number, bits [11:8] The part number is a 12-bit part number stored in ext-ERRPIDR0.PART_1 and this field. **0b0100** DSU-110 Cluster RAS. Bits [11:8] of part number 0x4E8.	0b0100

B.1.3.23 CLUSTERRAS_ERRPIDR2, Peripheral Identification Register 2

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFE8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0110 1011

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-47: ext_clusterras_errpidr2 bit assignments

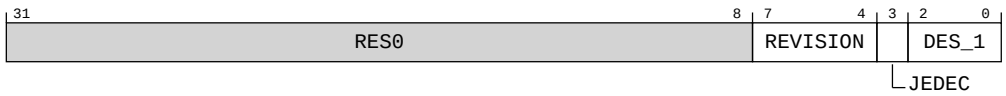


Table B-50: CLUSTERRAS_ERRPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	REVISION	Component major revision. This field and ext-ERRPIDR3.REVAND together form the revision number of the component, with REVISION being the most significant part and REVAND the least significant part. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4. For DSU-110: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r1p0. • Major revision 2 corresponds to r2p0. • Major revision 3 corresponds to r2p1. • Major revision 4 corresponds to r3p0.	0b0110
[3]	JEDEC	JEDEC-assigned JEP106 implementer code is used. This bit is **RAO**. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	Designer, JEP106 identification code, bits [6:4]. ext-ERRPIDR1.DES_0 and this field together form the JEDEC-assigned JEP106 identification code for the designer of the component. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B. **Note:** For a component designed by Arm Limited, the JEP106 identification code is 0x3B.	0b011

B.1.3.24 CLUSTERRAS_ERRPIDR3, Peripheral Identification Register 3

Provides discovery information about the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFEC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-48: ext_clusterras_errpidr3 bit assignments

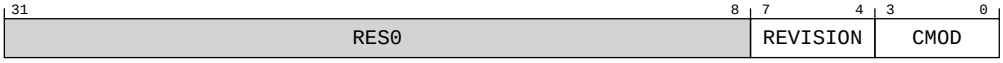


Table B-51: CLUSTERRAS_ERRPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component minor revision. **0b0000** Component minor revision 0. **0b0001** Component minor revision 1. **0b0010** Component minor revision 2. **0b0011** Component minor revision 3. **0b0100** Component minor revision 4.	0b0000
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design. For any two components with the same Unique Component Identifier: • If the value of the CMOD fields of both components equals zero, the components are identical. • If the value of the CMOD field of either of the two components is non-zero, they might not be identical, even though they have the same Unique Component Identifier. • If the CMOD fields of both components have the same non-zero value, it does not necessarily mean that they have the same modifications.	0b0000

B.1.3.25 CLUSTERRAS_ERRCIDR0, Component Identification Register 0

Provides discovery information for the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFF0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-49: ext_clusterras_errcidr0 bit assignments

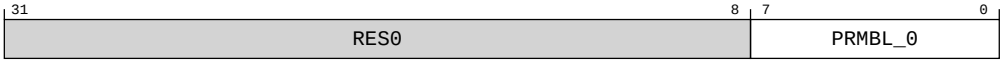


Table B-52: CLUSTERRAS_ERRCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	Component identification preamble, segment 0. This field reads as `0x0D`.	`0x0D`

B.1.3.26 CLUSTERRAS_ERRCIDR1, Component Identification Register 1

Provides discovery information for the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFF4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1111 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-50: ext_clusterras_errcidr1 bit assignments

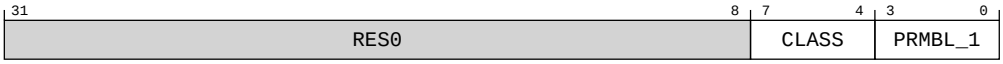


Table B-53: CLUSTERRAS_ERRCIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	Component class. **0b1111** System component with no standardized register layout.	0b1111
[3:0]	PRMBL_1	Component identification preamble, segment 1. This field reads as 0x0.	0b0000

B.1.3.27 CLUSTERRAS_ERRCIDR2, Component Identification Register 2

Provides discovery information for the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFF8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-51: ext_clusterras_errcidr2 bit assignments

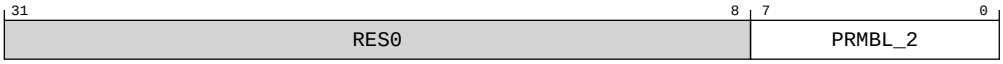


Table B-54: CLUSTERRAS_ERRCIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	Component identification preamble, segment 2. This field reads as 0x05.	0x05

B.1.3.28 CLUSTERRAS_ERRCIDR3, Component Identification Register 3

Provides discovery information for the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFFC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-52: ext_clusterras_errcidr3 bit assignments

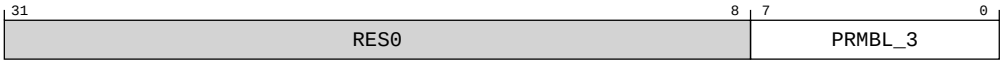


Table B-55: CLUSTERRAS_ERRCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Component identification preamble, segment 3. This field reads as `0xB1`.	`0xB1`

B.1.4 External cluster PPU register summary

The *Power Policy Unit* (PPU) registers for the DSU-110 DynamIQ™ cluster are only accessible from memory-mapped accesses on the utility bus. You can only access these registers from the Secure address space.

The summary table provides an overview of all the cluster PPU registers in the DSU-110. Individual register descriptions provide detailed information. These register descriptions are configuration of the PPU architecture, see *Arm® Power Policy Unit Architecture Specification* for more details.


Note

- The values for the cluster PPU registers are based on a typical multi-core cluster configuration, but these values might vary for different cluster configurations.
- The cluster PPU registers are treated as **RAZ/WI** if a Non-secure access is made to them. Any address that is not documented is also treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- The base address for the cluster PPU registers is `0x030000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-56: Cluster PPU register summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	–	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	–	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	–	32-bit	Power Status Register	Yes
0x010	PPU_DISR	–	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	–	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	–	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	–	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	–	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	–	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	–	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	–	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	–	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	–	32-bit	Additional Interrupt Status Register	Yes
0x040	PPU_IESR	–	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	–	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	–	32-bit	Functional Retention RAM Configuration Register	Yes
0x054	PPU_FULRR	–	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	–	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	–	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	–	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	–	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	–	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	–	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	–	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	–	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	–	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	–	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	–	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	–	32-bit	PPU Peripheral Identification Register 0	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFE4	PPU_PIDR1	–	32-bit	PPU Peripheral Identification Register 1	Yes
0xFE8	PPU_PIDR2	–	32-bit	PPU Peripheral Identification Register 2	Yes
0xFEC	PPU_PIDR3	–	32-bit	PPU Peripheral Identification Register 3	Yes
0xFF0	PPU_CIDR0	–	32-bit	PPU Component Identification Register 0	Yes
0xFF4	PPU_CIDR1	–	32-bit	PPU Component Identification Register 1	Yes
0xFF8	PPU_CIDR2	–	32-bit	PPU Component Identification Register 2	Yes
0xFFC	PPU_CIDR3	–	32-bit	PPU Component Identification Register 3	Yes

B.1.4.1 PPU_PWPR, Power Policy Register

This register enables software to program both power and operating mode policy. It also contains related settings including the enable for dynamic transitions and the lock enable.

This register does not reflect the current power mode value. The current power mode of the domain is reflected in the Power Status Register (ext-PPU_PWSR).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x000

Access type

RW

Reset value

xxxx xxx0 xxxx 0000 xxx0 xxx0 xxxx 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-53: ext_ppu_pwpr bit assignments

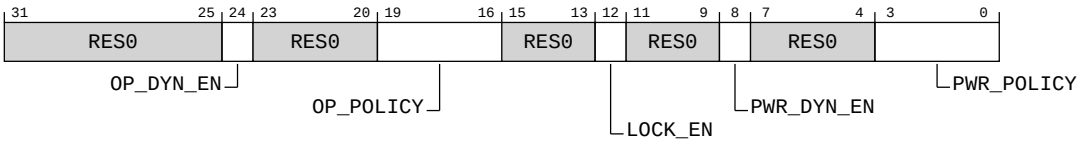


Table B-57: PPU_PWPR bit descriptions

Bits	Name	Description	Reset
[31:25]	**RES0**	Reserved	**RES0**
[24]	OP_DYN_EN	Operating mode dynamic transition enable. **0b0** Dynamic transitions disabled for operating modes. **0b1** Dynamic transitions enabled for operating modes, allowing transitions to be initiated by changes on operating mode DEVACTIVE inputs.	0b0
[23:20]	**RES0**	Reserved	**RES0**
[19:16]	OP_POLICY	Operating mode policy. When static operating mode transitions are enabled, OP_DYN_EN is set to 0b0, then this is the target operating mode for the PPU. When dynamic operating mode transitions are enabled, OP_DYN_EN is set to 0b1, then this is the minimum operating mode for the PPU. All other values are reserved. **0b0000** OPMODE_00: ONE_SLICE_SF_ONLY_ON: One L3 Cache slice is operational, the Cache RAM is powered down. **0b0001** OPMODE_01: ONE_SLICE_HALF_RAM_ON: One L3 Cache slice is operational, half of the Cache RAMs are powered on. **0b0011** OPMODE_03: ONE_SLICE_FULL_RAM_ON: One L3 Cache slice is operational, all of the Cache RAMs are powered on. **0b0100** OPMODE_04: ALL_SLICE_SF_ONLY_ON: All L3 Cache slices are operational, the Cache RAMs in each slice are powered down. **0b0101** OPMODE_05: ALL_SLICE_HALF_RAM_ON: All L3 Cache slices are operational, half of the Cache RAMs are powered on. **0b0111** OPMODE_07: ALL_SLICE_FULL_RAM_ON: All L3 Cache slices are operational, all of the Cache RAMs are powered on.	0b0000
[15:13]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[12]	LOCK_EN	Lock enable bit for OFF, OFF_EMU, MEM_RET and MEM_RET_EMU power modes. **0b0** Lock feature disabled. **0b1** Lock feature enabled.	0b0
[11:9]	**RES0**	Reserved	**RES0**
[8]	PWR_DYN_EN	Power mode dynamic transition enable. **0b0** Dynamic transitions disabled for power modes. **0b1** Dynamic transitions enabled for power modes, allowing transitions to be initiated by changes on power mode DEVACTIVE inputs.	0b0
[7:4]	**RES0**	Reserved	**RES0**
[3:0]	PWR_POLICY	Power mode policy. When static power mode transitions are enabled, PWR_DYN_EN is set to 0b0, this is the target power mode for the PPU. When dynamic power mode transitions are enabled, PWR_DYN_EN is set to 0b1, this is the minimum power mode for the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0010** MEM_RET. Memory Retention. Logic off with RAM retained. **0b0011** MEM_RET_EMU. Emulated Memory Retention. Logic on with RAM on. This mode is used to emulate the functional condition of MEM_RET without removing power. **0b0111** FUNC_RET. Functional Retention. Logic on with L3 Cache and Snoop Filter retained. **0b1000** ON. Logic on with RAM on, cluster is functional. **0b1001** WARM_RST. Warm reset . Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.4.2 PPU_PMER, Power Mode Emulation Enable Register

This register allows software to enable entry into emulated modes.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x004

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-54: ext_ppu_pmer bit assignments

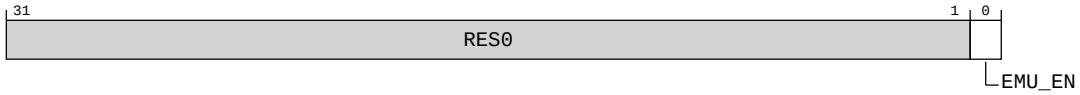


Table B-58: PPU_PMER bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	EMU_EN	Power mode emulation enable. **0b0** Power mode emulation disabled. **0b1** Power mode emulation enabled. Transitions to OFF and MEM_RET instead transition to OFF_EMU and MEM_RET_EMU.	0b0

B.1.4.3 PPU_PWSR, Power Status Register

This read-only register contains status information for the power mode, operating mode, dynamic transitions, and lock feature.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x008

Access type

RO

Reset value

xxxx xxx0 xxxx 0000 xxx0 xxx0 xxxx 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-55: ext_ppu_pwsr bit assignments

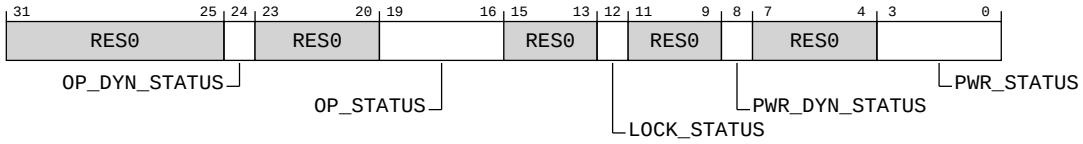


Table B-59: PPU_PWSR bit descriptions

Bits	Name	Description	Reset
[31:25]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[24]	OP_DYN_STATUS	Operating mode dynamic transition status. There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when ext-PPU_PWPR.OP_DYN_EN is programmed. **0b0** Dynamic transitions disabled for operating modes. **0b1** Dynamic transitions enabled for operating modes.	0b0
[23:20]	**RES0**	Reserved	**RES0**
[19:16]	OP_STATUS	Operating mode status. These bits reflect the current operating mode of the PPU. In the OFF, OFF_EMU, DBG_RECOV, and WARM_RST power modes, this field reflects the current programmed OP_POLICY even though the operating mode DEVPSTATE output bits are set to zero. All other values are reserved. **0b0000** OPMODE_00: ONE_SLICE_SF_ONLY_ON: One L3 Cache slice is operational, only the snoop filter RAM instances are active in the slice **0b0001** OPMODE_01: ONE_SLICE_HALF_RAM_ON: One L3 Cache slice is operational, half of the Cache RAMs are powered on. **0b0011** OPMODE_03: ONE_SLICE_FULL_RAM_ON: One L3 Cache slice is operational, all of the Cache RAMs are powered on. **0b0100** OPMODE_04: ALL_SLICE_SF_ONLY_ON: All L3 Cache slices are operational, only the snoop filter RAM instances are active in each slice. **0b0101** OPMODE_05: ALL_SLICE_HALF_RAM_ON: All L3 Cache slices are operational, half of the Cache RAMs are powered on. **0b0111** OPMODE_07: ALL_SLICE_FULL_RAM_ON: All L3 Cache slices are operational, all of the Cache RAMs are powered on.	0b0000
[15:13]	**RES0**	Reserved	**RES0**
[12]	LOCK_STATUS	Lock status. **0b0** The PPU is not locked in the current mode. **0b1** The PPU is locked in the current mode.	0b0
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8]	PWR_DYN_STATUS	Power mode dynamic transition status. There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when ext-PPU_PWPR.DYN_EN is programmed. **0b0** Dynamic transitions disabled for power modes. **0b1** Dynamic transitions enabled for power modes.	0b0
[7:4]	**RES0**	Reserved	**RES0**
[3:0]	PWR_STATUS	Power mode status. These bits reflect the current power mode of the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0010** MEM_RET. Memory Retention. Logic off with RAM retained. **0b0011** MEM_RET_EMU. Emulated Memory Retention. Logic on with RAM on. This mode is used to emulate the functional condition of MEM_RET without removing power. **0b0111** FUNC_RET. Functional Retention. Logic on with L3 Cache and Snoop Filter retained. **0b1000** ON. Logic on with RAM on, cluster is functional. **0b1001** WARM_RST. Warm reset . Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.4.4 PPU_DISR, Device Interface Input Current Status Register

This read-only register contains status reflecting the values of the device interface inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x010

Access type

RO

Reset value

xxxx x000 xxxx xxxx xxxx x000 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-56: ext_ppu_disr bit assignments

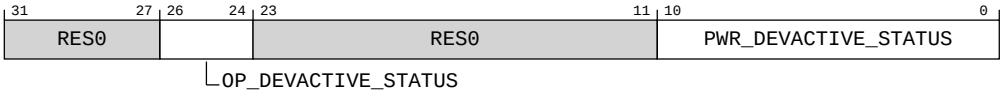


Table B-60: PPU_DISR bit descriptions

Bits	Name	Description	Reset
[31:27]	**RES0**	Reserved	**RES0**
[26:24]	OP_DEVACTIVE_STATUS	Status of the operating mode DEVPACTIVE inputs. All other values are reserved. **0b000** Request for OPMODE_00, ONE_SLICE_SF_ONLY_ON. **0b001** Request for OPMODE_01, ONE_SLICE_HALF_RAM_ON. **0b011** Request for OPMODE_03, ONE_SLICE_FULL_RAM_ON. **0b100** Request for OPMODE_04, ALL_SLICE_SF_ONLY_ON. **0b101** Request for OPMODE_05, ALL_SLICE_HALF_RAM_ON. **0b111** Request for OPMODE_07, ALL_SLICE_FULL_RAM_ON.	0b000
[23:11]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[10:0]	PWR_DEVACTIVE_STATUS	Status of the power mode DEVPACTIVE inputs. **0b00000000000** Request for OFF. **0000000001x** Request for OFF_EMU. **000000001xx** Request for MEM_RET. **00000001xxx** Request for MEM_RET_EMU. **0001xxxxxxx** Request for FUNC_RET. **001xxxxxxxx** Request for ON. **01xxxxxxxxx** Request for WARM_RST. **1xxxxxxxxxx** Request for DBG_RECOV.	0b00000000000

B.1.4.5 PPU_MISR, Miscellaneous Input Current Status Register

This read-only register contains status reflecting the values of miscellaneous inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x014

Access type

RO

Reset value

xxxx xxxx xxxx xxx0 xxxx xxx0 xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-57: ext_ppu_misr bit assignments

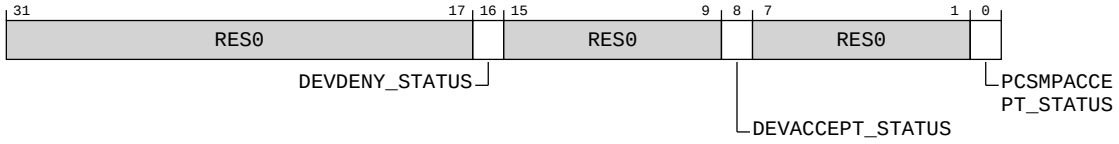


Table B-61: PPU_MISR bit descriptions

Bits	Name	Description	Reset
[31:17]	**RES0**	Reserved	**RES0**
[16]	DEVDENY_STATUS	Status of the device interface DEVPDENY inputs. **0b0** DEVPDENY deasserted. **0b1** DEVPDENY asserted.	0b0
[15:9]	**RES0**	Reserved	**RES0**
[8]	DEVACCEPT_STATUS	Status of the device interface DEVPACCEPT inputs. **0b0** DEVPACCEPT deasserted. **0b1** DEVPACCEPT asserted.	0b0
[7:1]	**RES0**	Reserved	**RES0**
[0]	PCSMPACCEPT_STATUS	Status of the PCSMPACCEPT inputs. **0b0** PCSMPACCEPT deasserted. **0b1** PCSMPACCEPT asserted.	0b0

B.1.4.6 PPU_STSR, Stored Status Register

This register is reserved for P-Channel PPUs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x018

Access type

RO

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-58: ext_ppu_stsr bit assignments

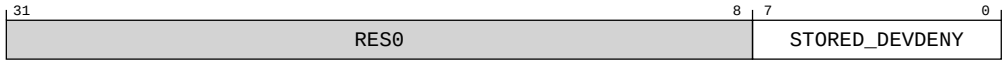


Table B-62: PPU_STSR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	STORED_DEVDENY	Status of the DEVDENY signals from the last device interface Q-Channel transition. This field is reserved. **0b00000000** Reserved for P-Channel PPUs.	8{x}

B.1.4.7 PPU_UNLK, Unlock Register

This register allows software to unlock the PPU from a locked power mode.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x01C

Access type

UNKNOWNW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-59: ext_ppu_unlk bit assignments

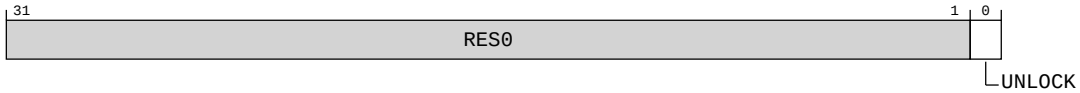


Table B-63: PPU_UNLK bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	UNLOCK	When `0b1` is written to this bit the PPU is unlocked from a locked power mode. A read always returns `0b0`.	`x`

B.1.4.8 PPU_PWCR, Power Configuration Register

This register controls enabling and disabling of hardware control inputs to the PPU.

Note

Before software programs the DEVREQEN bits it must configure the PPU for static transitions and ensure the requested power mode has been reached, this means that no further transitions can occur, otherwise behavior is UNPREDICTABLE.

The PWR_DEVACTIVEEN and OP_DEVACTIVEEN fields in this register control the ability of the DEVACTIVE inputs to initiate power mode transitions, but not the ability to generate input edge interrupt events.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x020

Access type

RW

Reset value

xxxx x111 xxxx x111 1111 111x xxxx xxx1

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-60: ext_ppu_pwcr bit assignments

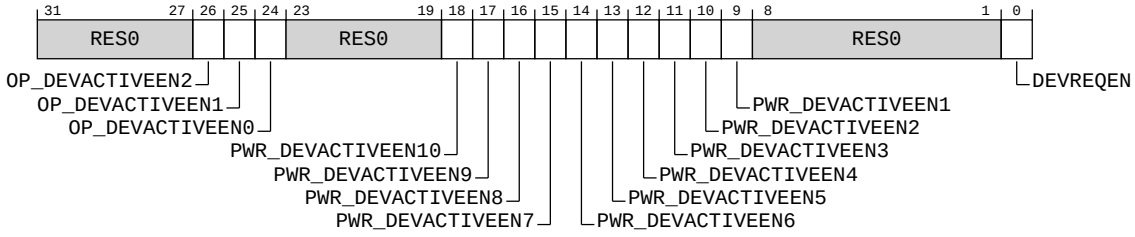


Table B-64: PPU_PWCR bit descriptions

Bits	Name	Description	Reset
[31:27]	**RES0**	Reserved	**RES0**
[26]	OP_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[18] input. **0b0** DEVPACTIVE[18] input (All L3 Cache Slices active) disabled. **0b1** DEVPACTIVE[18] input (All L3 Cache Slices active) enabled.	0b1

Bits	Name	Description	Reset
[25]	OP_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[17] input. **0b0** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) disabled. **0b1** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) enabled.	0b1
[24]	OP_DEVACTIVEEN0	Enables the operating mode DEVPACTIVE[16] input. **0b0** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) disabled. **0b1** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) enabled.	0b1
[23:19]	**RES0**	Reserved	**RES0**
[18]	PWR_DEVACTIVEEN10	Enables the operating mode DEVPACTIVE[10] input. **0b0** DEVPACTIVE[10] input (DBG_RECOV) disabled. **0b1** DEVPACTIVE[10] input (DBG_RECOV) enabled.	0b1
[17]	PWR_DEVACTIVEEN9	Enables the operating mode DEVPACTIVE[9] input. **0b0** DEVPACTIVE[9] input (WARM_RST) disabled. **0b1** DEVPACTIVE[9] input (WARM_RST) enabled.	0b1
[16]	PWR_DEVACTIVEEN8	Enables the operating mode DEVPACTIVE[8] input. **0b0** DEVPACTIVE[8] input (ON) disabled. **0b1** DEVPACTIVE[8] input (ON) enabled.	0b1
[15]	PWR_DEVACTIVEEN7	Enables the operating mode DEVPACTIVE[7] input. **0b0** DEVPACTIVE[7] input (FUNC_RET) disabled. **0b1** DEVPACTIVE[7] input (FUNC_RET) enabled.	0b1
[14]	PWR_DEVACTIVEEN6	Enables the operating mode DEVPACTIVE[6] input. **0b1** DEVPACTIVE[6] input (MEM_OFF) enabled.	0b1
[13]	PWR_DEVACTIVEEN5	Enables the operating mode DEVPACTIVE[5] input. **0b0** DEVPACTIVE[5] input (FULL_RET) disabled. **0b1** DEVPACTIVE[5] input (FULL_RET) enabled.	0b1
[12]	PWR_DEVACTIVEEN4	Enables the operating mode DEVPACTIVE[4] input. **0b1** DEVPACTIVE[4] input (LOGIC_RET) enabled.	0b1

Bits	Name	Description	Reset
[11]	PWR_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[3] input. **0b0** DEVPACTIVE[3] input (MEM_RET_EMU) disabled. **0b1** DEVPACTIVE[3] input (MEM_RET_EMU) enabled.	0b1
[10]	PWR_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[2] input. **0b0** DEVPACTIVE[2] input (MEM_RET) disabled. **0b1** DEVPACTIVE[2] input (MEM_RET) enabled.	0b1
[9]	PWR_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[1] input. **0b0** DEVPACTIVE[1] input (OFF_EMU) disabled. **0b1** DEVPACTIVE[1] input (OFF_EMU) enabled.	0b1
[8:1]	**RES0**	Reserved	**RES0**
[0]	DEVREQEN	Device interface handshake enable. **0b0** Device interface handshake disabled for transitions. **0b1** Device interface handshake enabled for transitions.	0b1

B.1.4.9 PPU_PTCR, Power Mode Transition Register

This register contains settings which affect the behaviour of certain power mode transitions.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x024

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-61: ext_ppu_ptcr bit assignments

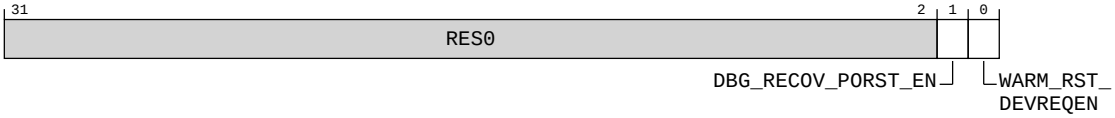


Table B-65: PPU_PTCR bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	DBG_RECOV_PORST_EN	Power-on reset behavior in DBG_RECOV. This bit should not be modified when the PPU is in DBG_RECOV or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **0b0** DEVPORESETn is not asserted when in DBG_RECOV. **0b1** DEVPORESETn is asserted when in DBG_RECOV.	0b0
[0]	WARM_RST_DEVREQEN	Device interface handshake behavior. This bit should not be modified when the PPU is in WARM_RST, or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **0b0** The PPU does not perform a device interface handshake when transitioning between ON and WARM_RST. **0b1** The PPU performs a device interface handshake when transitioning between ON and WARM_RST.	0b0

B.1.4.10 PPU_IMR, Interrupt Mask Register

This register controls the events that assert the interrupt output. Additional event masking controls are in the Additional Interrupt Mask Register (ext-PPU_AIMR), Input Edge Sensitivity Register (ext-PPU_IESR), and the Operating Mode Active Edge Sensitivity Register (ext-PPU_OPSR).

When an interrupt event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x030

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx11 1010

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-62: ext_ppu_imr bit assignments

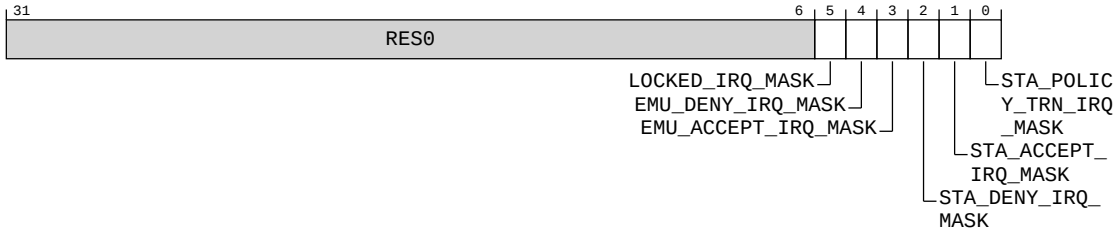


Table B-66: PPU_IMR bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	LOCKED_IRQ_MASK	Locked event mask **0b0** Locked event enabled. **0b1** Locked event masked.	0b1

Bits	Name	Description	Reset
[4]	EMU_DENY_IRQ_MASK	Emulation transition denial event mask **0b0** Emulation transition denial event enabled. **0b1** Emulation transition denial event masked.	0b1
[3]	EMU_ACCEPT_IRQ_MASK	Emulation transition acceptance event mask **0b0** Emulation transition acceptance event enabled. **0b1** Emulation transition acceptance event masked.	0b1
[2]	STA_DENY_IRQ_MASK	Static transition denial event mask **0b0** Static transition denial event enabled. **0b1** Static transition denial event masked.	0b0
[1]	STA_ACCEPT_IRQ_MASK	Static transition acceptance event mask **0b0** Static transition acceptance event enabled. **0b1** Static transition acceptance event masked.	0b1
[0]	STA_POLICY_TRN_IRQ_MASK	Static full policy transition completion event mask **0b0** Static full policy transition completion event enabled. **0b1** Static full policy transition completion event masked.	0b0

B.1.4.11 PPU_AIMR, Additional Interrupt Mask Register

This register controls the events that assert the interrupt output. Additional event masking controls are in the Interrupt Mask Register (ext-PPU_IMR), Input Edge Sensitivity Register (ext-PPU_IESR), and the Operating Mode Active Edge Sensitivity Register (ext-PPU_OPSR).

When an interrupt event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x034

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxx1 1110

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-63: ext_ppu_aimr bit assignments

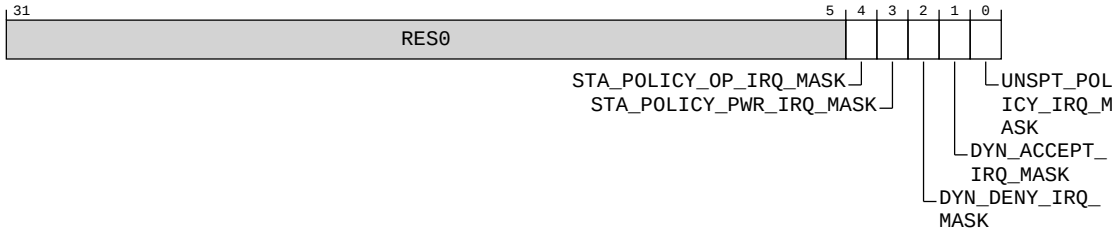


Table B-67: PPU_AIMR bit descriptions

Bits	Name	Description	Reset
[31:5]	**RES0**	Reserved	**RES0**
[4]	STA_POLICY_OP_IRQ_MASK	Static operating policy transition completion event mask **0b0** Static operating policy transition completion event enabled. **0b1** Static operating policy transition completion event masked.	0b1
[3]	STA_POLICY_PWR_IRQ_MASK	Static power policy transition completion event mask **0b0** Static power policy transition completion event enabled. **0b1** Static power policy transition completion event masked.	0b1

Bits	Name	Description	Reset
[2]	DYN_DENY_IRQ_MASK	Dynamic transition denial event mask **0b0** Dynamic transition denial event enabled. **0b1** Dynamic transition denial event masked.	0b1
[1]	DYN_ACCEPT_IRQ_MASK	Dynamic transition acceptance event mask **0b0** Dynamic transition acceptance event enabled. **0b1** Dynamic transition acceptance event masked.	0b1
[0]	UNSPT_POLICY_IRQ_MASK	Unsupported policy event mask **0b0** Unsupported policy event enabled. **0b1** Unsupported policy event masked.	0b0

B.1.4.12 PPU_ISR, Interrupt Status Register

This register contains information about events causing the assertion of the interrupt output. It is also used to clear interrupt events.

A bit set to 0b1 indicates the event asserted the interrupt output. Multiple events can be active at the same time. When an interrupt event is masked an occurrence of that event does not set the status bit.

A write of 0b1 to an event bit clears that event. A write of 0b0 to a bit has no effect. The interrupt output stays HIGH until all status bits in the Interrupt Status Register (PPU_ISR) and the Additional Interrupt Status Register (ext-PPU_AISR) are 0b0.

When the OTHER_IRQ bit is set, this indicates an event from the Additional Interrupt Status Register (PPU_AISR) has caused the interrupt output to be asserted. This bit cannot be cleared by writing to this register. It must be cleared by writing to the active event in the Additional Interrupt Status Register (ext-PPU_AISR).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x038

Access type

RW

Reset value

xxxx x000 xxxx x000 0xxx 000x 0x00 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-64: ext_ppu_isr bit assignments

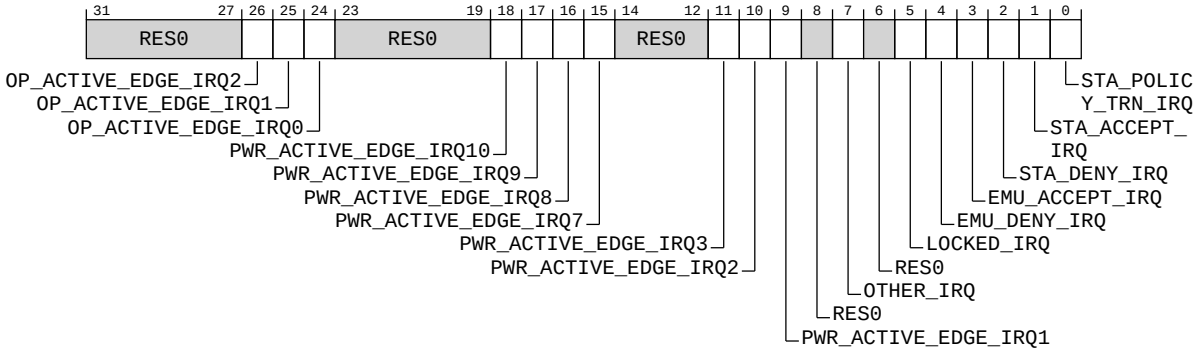


Table B-68: PPU_ISR bit descriptions

Bits	Name	Description	Reset
[31:27]	RES0	Reserved	RES0
[26]	OP_ACTIVE_EDGE_IRQ2	Indicates if operating mode DEVPACTIVE[18] input caused the input edge event. **0b0** DEVPACTIVE[18] input (All L3 Cache Slices active) did not assert the interrupt output. **0b1** DEVPACTIVE[18] input (All L3 Cache Slices active) asserted the interrupt output.	0b0
[25]	OP_ACTIVE_EDGE_IRQ1	Indicates if operating mode DEVPACTIVE[17] input caused the input edge event. **0b0** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) did not assert the interrupt output. **0b1** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[24]	OP_ACTIVE_EDGE_IRQ0	Indicates if operating mode DEVPACTIVE[16] input caused the input edge event. **0b0** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) did not assert the interrupt output. **0b1** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) asserted the interrupt output.	0b0
[23:19]	**RES0**	Reserved	**RES0**
[18]	PWR_ACTIVE_EDGE_IRQ10	Indicates if power mode DEVPACTIVE[10] input caused the input edge event. **0b0** DEVPACTIVE[10] input (DBG_RECOV) did not assert the interrupt output. **0b1** DEVPACTIVE[10] input (DBG_RECOV) asserted the interrupt output.	0b0
[17]	PWR_ACTIVE_EDGE_IRQ9	Indicates if power mode DEVPACTIVE[9] input caused the input edge event. **0b0** DEVPACTIVE[9] input (WARM_RST) did not assert the interrupt output. **0b1** DEVPACTIVE[9] input (WARM_RST) asserted the interrupt output.	0b0
[16]	PWR_ACTIVE_EDGE_IRQ8	Indicates if power mode DEVPACTIVE[8] input caused the input edge event. **0b0** DEVPACTIVE[8] input (ON) did not assert the interrupt output. **0b1** DEVPACTIVE[8] input (ON) asserted the interrupt output.	0b0
[15]	PWR_ACTIVE_EDGE_IRQ7	Indicates if power mode DEVPACTIVE[7] input caused the input edge event. **0b0** DEVPACTIVE[7] input (FUNC_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[7] input (FUNC_RET) asserted the interrupt output.	0b0
[14:12]	**RES0**	Reserved	**RES0**
[11]	PWR_ACTIVE_EDGE_IRQ3	Indicates if power mode DEVPACTIVE[3] input caused the input edge event. **0b0** DEVPACTIVE[3] input (MEM_RET_EMU) did not assert the interrupt output. **0b1** DEVPACTIVE[3] input (MEM_RET_EMU) asserted the interrupt output.	0b0
[10]	PWR_ACTIVE_EDGE_IRQ2	Indicates if power mode DEVPACTIVE[2] input caused the input edge event. **0b0** DEVPACTIVE[2] input (MEM_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[2] input (MEM_RET) asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[9]	PWR_ACTIVE_EDGE_IRQ1	Indicates if power mode DEVPACTIVE[1] input caused the input edge event. **0b0** DEVPACTIVE[1] input (OFF_EMU) did not assert the interrupt output. **0b1** DEVPACTIVE[1] input (OFF_EMU) asserted the interrupt output.	0b0
[8]	**RES0**	Reserved	**RES0**
[7]	OTHER_IRQ	Indicates there is an interrupt event pending in the Additional Interrupt Status Register (ext-PPU_AISR). **0b0** No interrupt pending in ext-PPU_AISR. **0b1** Interrupt pending in ext-PPU_AISR.	0b0
[6]	**RES0**	Reserved	**RES0**
[5]	LOCKED_IRQ	Locked event status. **0b0** No locked event. **0b1** A locked event asserted the interrupt output.	0b0
[4]	EMU_DENY_IRQ	Emulated transition denial event status. **0b0** No emulated transition denial event. **0b1** An emulated transition denial event asserted the interrupt output.	0b0
[3]	EMU_ACCEPT_IRQ	Emulated transition acceptance event status. **0b0** No emulated transition acceptance event. **0b1** An emulated transition acceptance event asserted the interrupt output.	0b0
[2]	STA_DENY_IRQ	Static transition denial event status. **0b0** No static transition denial event. **0b1** An static transition denial event asserted the interrupt output.	0b0
[1]	STA_ACCEPT_IRQ	Static transition acceptance event status. **0b0** No static transition acceptance event. **0b1** An static transition acceptance event asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[0]	STA_POLICY_TRN_IRQ	Static full policy transition completion event status. **0b0** No static full policy transition completion event. **0b1** An static full policy transition completion event asserted the interrupt output.	0b0

B.1.4.13 PPU_AISR, Additional Interrupt Status Register

This register contains information about events causing the assertion of the interrupt output. It is also used to clear interrupt events.

A bit set to 0b1 indicates the event asserted the interrupt output. Multiple events can be active at the same time. When an interrupt event is masked an occurrence of that event does not set the status bit.

A write of 0b1 to an event bit clears that event. A write of 0b0 has no effect. The interrupt output stays HIGH until all status bits in the Interrupt Status Register (ext-PPU_ISR) and the Additional Interrupt Status Register (PPU_AISR) are set to 0b0.

When an interrupt status is set to 0b1 in this register it sets the OTHER_IRQ bit in the Interrupt Status Register (ext-PPU_ISR). Status bits in this register are only cleared by writing to this register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x03C

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxx0 0000


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-65: ext_ppu_aisr bit assignments

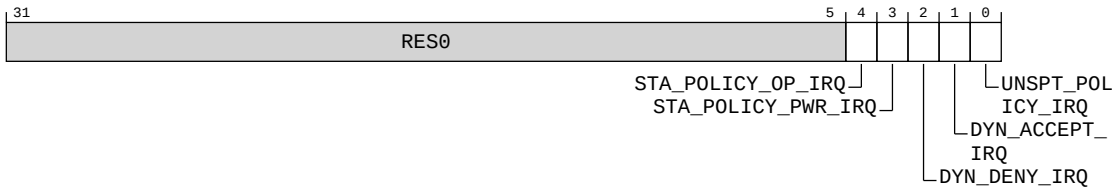


Table B-69: PPU_AISR bit descriptions

Bits	Name	Description	Reset
[31:5]	**RES0**	Reserved	**RES0**
[4]	STA_POLICY_OP_IRQ	Static operating policy transition completion event status **0b0** No static operating policy transition completion event. **0b1** A static operating policy transition completion event asserted the interrupt output.	0b0
[3]	STA_POLICY_PWR_IRQ	Static power policy transition completion event status **0b0** No static power policy transition completion event. **0b1** A static power policy transition completion event asserted the interrupt output.	0b0
[2]	DYN_DENY_IRQ	Dynamic transition denial event status **0b0** No dynamic transition denial event. **0b1** A dynamic transition denial event asserted the interrupt output.	0b0
[1]	DYN_ACCEPT_IRQ	Dynamic transition acceptance event status **0b0** No dynamic transition acceptance event. **0b1** A dynamic transition acceptance event asserted the interrupt output.	0b0
[0]	UNSPT_POLICY_IRQ	Unsupported policy event status **0b0** No unsupported policy event. **0b1** An unsupported policy event asserted the interrupt output.	0b0

B.1.4.14 PPU_IESR, Input Edge Sensitivity Register

This register configures the transitions on the power mode DEVPACTIVE inputs that generate an Input Edge interrupt event.

When an event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x040

Access type

RW

Reset value

xxxx xxxx xx00 0000 00xx xxxx 0000 00xx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-66: ext_ppu_iesr bit assignments

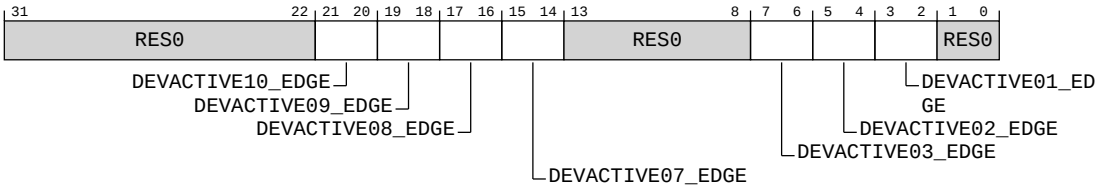


Table B-70: PPU_IESR bit descriptions

Bits	Name	Description	Reset
[31:22]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[21:20]	DEVACTIVE10_EDGE	Configures the transitions on the DEVPACTIVE[10] input (DBG_RECOV) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[19:18]	DEVACTIVE09_EDGE	Configures the transitions on the DEVPACTIVE[9] input (WARM_RST) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[17:16]	DEVACTIVE08_EDGE	Configures the transitions on the DEVPACTIVE[8] input (ON) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[15:14]	DEVACTIVE07_EDGE	Configures the transitions on the DEVPACTIVE[7] input (FUNC_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[13:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:6]	DEVACTIVE03_EDGE	Configures the transitions on the DEVPACTIVE[3] input (MEM_RET_EMU) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[5:4]	DEVACTIVE02_EDGE	Configures the transitions on the DEVPACTIVE[2] input (MEM_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[3:2]	DEVACTIVE01_EDGE	Configures the transitions on the DEVPACTIVE[1] input (OFF_EMU) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[1:0]	**RES0**	Reserved	**RES0**

B.1.4.15 PPU_OPSR, Operating Mode Active Edge Sensitivity Register

This register configures the transitions on the operating mode DEVPACTIVE inputs that generate an Input Edge interrupt event.

When an event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x044

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx00 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-67: ext_ppu_opsr bit assignments

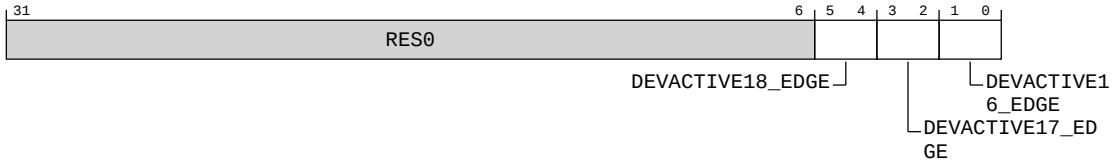


Table B-71: PPU_OPSR bit descriptions

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5:4]	DEVACTIVE18_EDGE	Configures the transitions on the DEVPACTIVE[18] input (All L3 Cache Slices active) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00

Bits	Name	Description	Reset
[3:2]	DEVACTIVE17_EDGE	Configures the transitions on the DEVPACTIVE[17] input (Upper L3 Cache RAMs active) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[1:0]	DEVACTIVE16_EDGE	Configures the transitions on the DEVPACTIVE[16] input (Lower L3 Cache RAMs active) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00

B.1.4.16 PPU_FUNRR, Functional Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x050

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-68: ext_ppu_funrr bit assignments

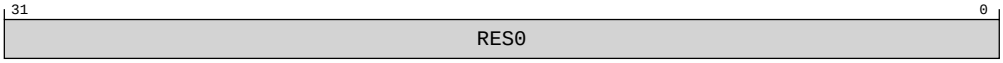


Table B-72: PPU_FUNRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.17 PPU_FULRR, Full Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x054

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-69: ext_ppu_fulrr bit assignments

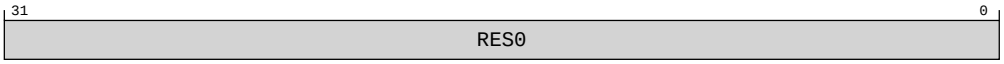


Table B-73: PPU_FULRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.4.18 PPU_MEMRR, Memory Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x058

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-70: ext_ppu_memrr bit assignments

Table B-74: PPU_MEMRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.4.19 PPU_DCDR0, Device Control Delay Configuration Register 0

This register is used to program device control delay parameters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x170

Access type

RW

Reset value

xxxx xxxx 0000 0000 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-71: ext_ppu_dcdr0 bit assignments

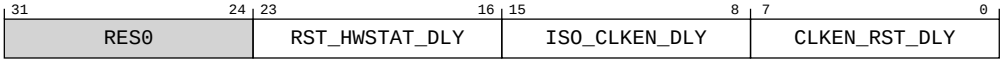


Table B-75: PPU_DCDR0 bit descriptions

Bits	Name	Description	Reset
[31:24]	RES0	Reserved	RES0
[23:16]	RST_HWSTAT_DLY	Delay from reset de-assertion to HWSTAT update.	0x00
[15:8]	ISO_CLKEN_DLY	Delay from isolation enable de-assertion to clock enable assertion.	0x00
[7:0]	CLKEN_RST_DLY	Delay from clock enable assertion to reset de-assertion.	0x00

B.1.4.20 PPU_DCDR1, Device Control Delay Configuration Register 1

This register is used to program device control delay parameters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x174

Access type

RW

Reset value

xxxx xxxx xxxx xxxx 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-72: ext_ppu_dcdr1 bit assignments

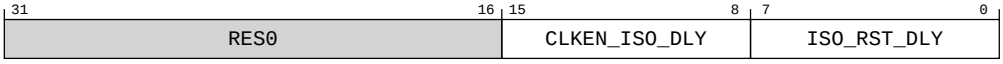


Table B-76: PPU_DCDR1 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:8]	CLKEN_ISO_DLY	Delay from clock enable de-assertion to isolation enable assertion.	0x00
[7:0]	ISO_RST_DLY	Delay from isolation enable assertion to reset assertion.	0x00

B.1.4.21 PPU_IDR0, PPU Identification Register 0

This read-only register contains information on the type and number of channels on the device interface and power and operating modes supported.

Additional information on optional features can be found in the PPU Identification Register 1 (ext-PPU_IDR1).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFB0

Access type

RO

Reset value

xx01 1000 1111 x111 1000 1111 0111 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-73: ext_ppu_idr0 bit assignments

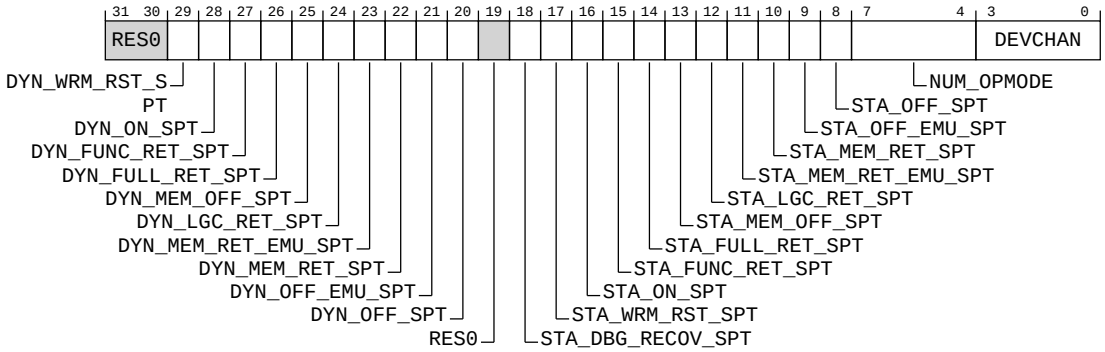


Table B-77: PPU_IDR0 bit descriptions

Bits	Name	Description	Reset
[31:30]	**RES0**	Reserved	**RES0**
[29]	DYN_WRM_RST_SPT	Dynamic WARM_RST support. **0b0** Dynamic WARM_RST not supported.	0b0
[28]	DYN_ON_SPT	Dynamic ON support. **0b1** Dynamic ON supported.	0b1
[27]	DYN_FUNC_RET_SPT	Dynamic DYN_FUNC_RET_SPT support. **0b1** Dynamic DYN_FUNC_RET_SPT supported.	0b1
[26]	DYN_FULL_RET_SPT	Dynamic DYN_FULL_RET_SPT support. **0b0** Dynamic DYN_FULL_RET_SPT not supported.	0b0
[25]	DYN_MEM_OFF_SPT	Dynamic MEM_OFF support. **0b0** Dynamic MEM_OFF not supported.	0b0
[24]	DYN_LGC_RET_SPT	Dynamic LOGIC_RET support. **0b0** Dynamic LOGIC_RET not supported.	0b0
[23]	DYN_MEM_RET_EMU_SPT	Dynamic DYN_MEM_RET_EMU_SPT support. **0b1** Dynamic DYN_MEM_RET_EMU_SPT supported.	0b1
[22]	DYN_MEM_RET_SPT	Dynamic DYN_MEM_RET_SPT support. **0b1** Dynamic DYN_MEM_RET_SPT supported.	0b1
[21]	DYN_OFF_EMU_SPT	Dynamic OFF_EMU support. **0b1** Dynamic OFF_EMU supported.	0b1
[20]	DYN_OFF_SPT	Dynamic OFF support. **0b1** Dynamic OFF supported.	0b1
[19]	**RES0**	Reserved	**RES0**
[18]	STA_DBG_RECOV_SPT	DBG_RECOV support. **0b1** DBG_RECOV supported.	0b1
[17]	STA_WRM_RST_SPT	WARM_RST support. **0b1** WRM_RST supported.	0b1

Bits	Name	Description	Reset
[16]	STA_ON_SPT	ON support. **0b1** ON supported.	0b1
[15]	STA_FUNC_RET_SPT	FUNC_RET support. **0b1** FUNC_RET supported.	0b1
[14]	STA_FULL_RET_SPT	FULL_RET support. **0b0** FULL_RET not supported.	0b0
[13]	STA_MEM_OFF_SPT	MEM_OFF support. **0b0** MEM_OFF not supported.	0b0
[12]	STA_LGC_RET_SPT	LOGIC_RET support. **0b0** LOGIC_RET not supported.	0b0
[11]	STA_MEM_RET_EMU_SPT	MEM_RET_EMU support. **0b1** MEM_RET_EMU supported.	0b1
[10]	STA_MEM_RET_SPT	MEM_RET support. **0b1** MEM_RET supported.	0b1
[9]	STA_OFF_EMU_SPT	OFF_EMU support. **0b1** OFF_EMU supported.	0b1
[8]	STA_OFF_SPT	OFF support. **0b1** OFF supported.	0b1
[7:4]	NUM_OPMODE	No. of operating modes supported, minus 1. **0b0111** 8 operating modes supported.	0b0111
[3:0]	DEVCHAN	No. of Device Interface Channels. **0b0000** 0 (P-channel PPU).	0b0000

B.1.4.22 PPU_IDR1, PPU Identification Register 1

This read-only register contains information on the optional features and configurations that are supported by this PPU.

Additional information on optional features can be found in the PPU Identification Register 0 (ext-PPU_IDR0).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFB4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxx1 x111 x000 x110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-74: ext_ppu_idr1 bit assignments

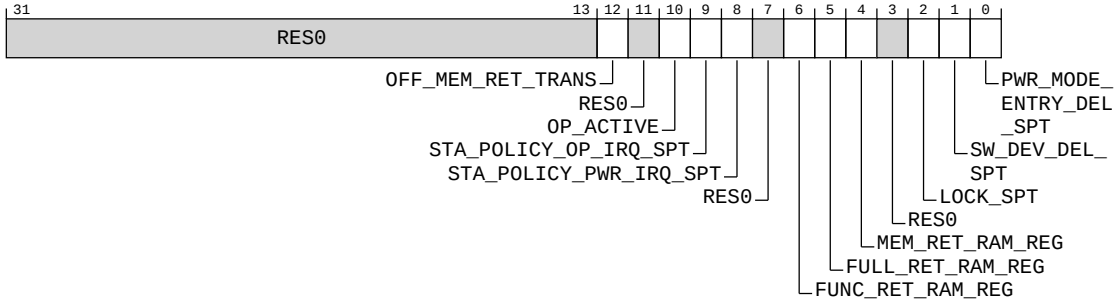


Table B-78: PPU_IDR1 bit descriptions

Bits	Name	Description	Reset
[31:13]	**RES0**	Reserved	**RES0**
[12]	OFF_MEM_RET_TRANS	OFF to MEM_RET direct transition. Indicates if direct transitions from OFF to MEM_RET and from OFF_EMU to MEM_RET_EMU are supported. **0b1** OFF to MEM_RET direct transition supported.	0b1
[11]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[10]	OP_ACTIVE	Operating mode use model for dynamic transitions. **0b1** Independent use model.	0b1
[9]	STA_POLICY_OP_IRQ_SPT	Operating policy transition completion event status. **0b1** Operating policy transition completion events supported.	0b1
[8]	STA_POLICY_PWR_IRQ_SPT	Power policy transition completion event status. **0b1** Power policy transition completion events supported.	0b1
[7]	**RES0**	Reserved	**RES0**
[6]	FUNC_RET_RAM_REG	Indicates if the ext-PPU_FUNRR register is present or reserved. **0b0** ext-PPU_FUNRR is reserved.	0b0
[5]	FULL_RET_RAM_REG	Indicates if the ext-PPU_FULRR register is present or reserved. **0b0** ext-PPU_FULRR is reserved.	0b0
[4]	MEM_RET_RAM_REG	Indicates if the ext-PPU_MEMRR register is present or reserved. **0b0** ext-PPU_MEMRR is reserved.	0b0
[3]	**RES0**	Reserved	**RES0**
[2]	LOCK_SPT	Indicates if the lock and the lock interrupt event are supported. **0b1** Lock and the lock interrupt event are supported.	0b1
[1]	SW_DEV_DEL_SPT	Software device delay control configuration support. **0b1** Software device delay control configuration supported.	0b1
[0]	PWR_MODE_ENTRY_DEL_SPT	Power mode entry delay support. **0b0** Power mode entry delay not supported.	0b0

B.1.4.23 PPU_IIDR, Implementation Identification Register

This register provides information about the implementer and implementation of the PPU.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFC8

Access type

RO

Reset value

0000 1011 0110 0001 0001 0100 0011 1011

Bit descriptions

Figure B-75: ext_ppu_iidr bit assignments

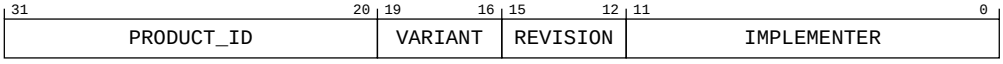


Table B-79: PPU_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	PRODUCT_ID	Value identifying the PPU part. **0b000010110110** Power Policy Unit.	0x0B6
[19:16]	VARIANT	Value used to distinguish PPU variants, or major revisions of the PPU. **0b0000** PPU variant 0. **0b0001** PPU variant 1. **0b0010** PPU variant 2. **0b0011** PPU variant 3. **0b0100** PPU variant 4.	0b0001

Bits	Name	Description	Reset
[15:12]	REVISION	Value used to distinguish minor revisions of the PPU. **0b0000** PPU revision 0. **0b0001** PPU revision 1. **0b0010** PPU revision 2. **0b0011** PPU revision 3. **0b0100** PPU revision 4.	0b0001
[11:0]	IMPLEMENTER	Implementer identification. **0b010000111011** Arm Limited.	0x43B

B.1.4.24 PPU_AIDR, Architecture Identification Register

This register identifies the PPU architecture revision.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFCC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-76: ext_ppu_aidr bit assignments

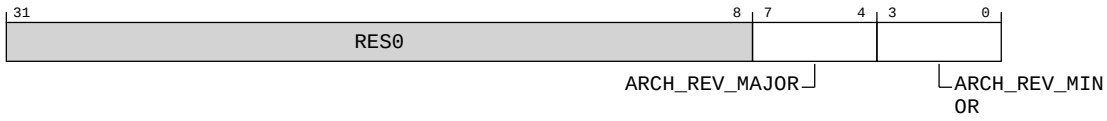


Table B-80: PPU_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	ARCH_REV_MAJOR	PPU architecture major revision. **0b0001** PPU architecture major revision 1.	0b0001
[3:0]	ARCH_REV_MINOR	PPU architecture minor revision. **0b0001** PPU architecture minor revision 1.	0b0001

B.1.4.25 PPU_PIDR4, PPU Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFD0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-77: ext_ppu_pidr4 bit assignments

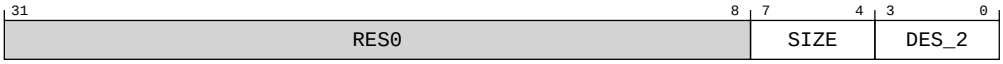


Table B-81: PPU_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	xxxx
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	xxxx

B.1.4.26 PPU_PIDR5, PPU Peripheral Identification Register 5

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFD4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-78: ext_ppu_pidr5 bit assignments

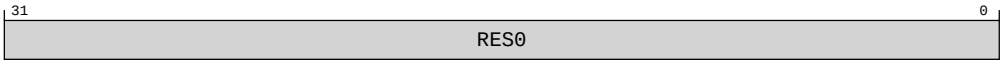


Table B-82: PPU_PIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.4.27 PPU_PIDR6, PPU Peripheral Identification Register 6

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFD8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-79: ext_ppu_pidr6 bit assignments

Table B-83: PPU_PIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.28 PPU_PIDR7, PPU Peripheral Identification Register 7

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFDC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-80: ext_ppu_pidr7 bit assignments

Table B-84: PPU_PIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.29 PPU_PIDR0, PPU Peripheral Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFE0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-81: ext_ppu_pidr0 bit assignments

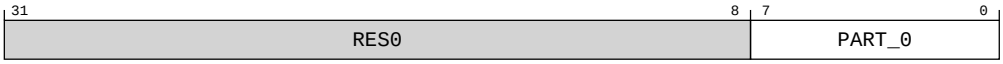


Table B-85: PPU_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **0b10110110** DSU-110 Power Policy Unit. Bits [7:0] of part number 0x0B6.	0xB6

B.1.4.30 PPU_PIDR1, PPU Peripheral Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFE4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-82: ext_ppu_pidr1 bit assignments

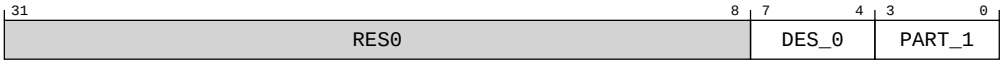


Table B-86: PPU_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0000** DSU-110 Power Policy Unit. Bits [11:8] of part number 0x0B6.	0b0000

B.1.4.31 PPU_PIDR2, PPU Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFE8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 1011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-83: ext_ppu_pidr2 bit assignments

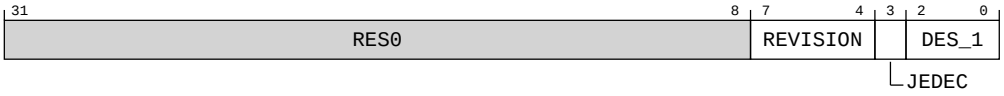


Table B-87: PPU_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4.	0b0001
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

B.1.4.32 PPU_PIDR3, PPU Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFEC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-84: ext_ppu_pidr3 bit assignments

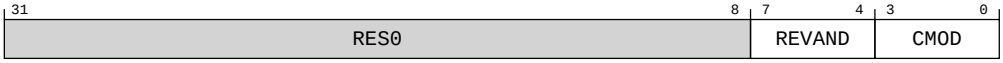


Table B-88: PPU_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0. **0b0001** Component minor revision 1. **0b0010** Component minor revision 2. **0b0011** Component minor revision 3. **0b0100** Component minor revision 4.	0b0001
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

B.1.4.33 PPU_CIDR0, PPU Component Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFF0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-85: ext_ppu_cidr0 bit assignments

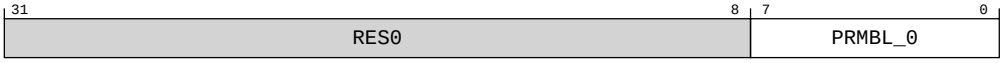


Table B-89: PPU_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

B.1.4.34 PPU_CIDR1, PPU Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFF4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1111 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-86: ext_ppu_cidr1 bit assignments

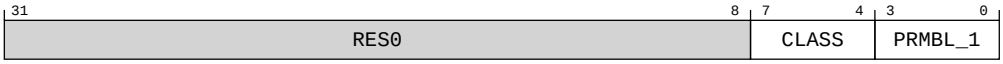


Table B-90: PPU_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	CoreSight component class. **0b1111** CoreLink component.	0b1111
[3:0]	PRMBL_1	CoreSight component identification preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.1.4.35 PPU_CIDR2, PPU Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFF8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-87: ext_ppu_cidr2 bit assignments

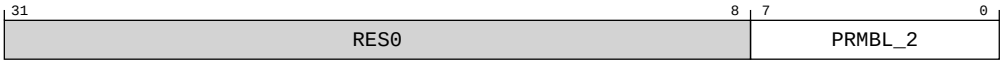


Table B-91: PPU_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	CoreSight component identification preamble. **0b00000101** CoreSight component identification preamble.	0x05

B.1.4.36 PPU_CIDR3, PPU Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFFC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-88: ext_ppu_cidr3 bit assignments

Table B-92: PPU_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **0b10110001** CoreSight component identification preamble.	0xB1

B.1.5 External core PPU register summary

Each core *Power Policy Unit* (PPU) in the DSU-110 DynamIQ™ cluster has an individual set of *Power Policy Unit* (PPU) registers. Each set of registers is identical, and are memory-mapped onto the utility bus at different base addresses. You can only access the PPU registers for each core from the Secure address space.

The summary table provides an overview of all the core PPU registers in the DSU-110. Individual register descriptions provide detailed information. These register descriptions are a configuration of the PPU architecture, see *Arm® Power Policy Unit Architecture Specification* for more details.

Note

- The values for the core PPU registers are based on a typical multi-core cluster configuration, but these values might vary for different cluster configurations.
- The core PPU registers are treated as **RAZ/WI** if a Non-secure access is made to them. Any address that is not documented is also treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- The base address for the core PPU registers is 0x<n>80000, where n is the core instance number. For example, for core 0 the PPU base address is 0x080000 and for core 1 the PPU base address is 0x180000.

- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-93: Core PPU register summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	–	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	–	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	–	32-bit	Power Status Register	Yes
0x010	PPU_DISR	–	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	–	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	–	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	–	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	–	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	–	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	–	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	–	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	–	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	–	32-bit	Additional Interrupt Status Register	Yes
0x040	PPU_IESR	–	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	–	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	–	32-bit	Functional Retention RAM Configuration Register	Yes
0x054	PPU_FULRR	–	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	–	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	–	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	–	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	–	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	–	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	–	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	–	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	–	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	–	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	–	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	–	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	–	32-bit	PPU Peripheral Identification Register 0	Yes
0xFE4	PPU_PIDR1	–	32-bit	PPU Peripheral Identification Register 1	Yes
0xFE8	PPU_PIDR2	–	32-bit	PPU Peripheral Identification Register 2	Yes
0xFEC	PPU_PIDR3	–	32-bit	PPU Peripheral Identification Register 3	Yes
0xFF0	PPU_CIDR0	–	32-bit	PPU Component Identification Register 0	Yes
0xFF4	PPU_CIDR1	–	32-bit	PPU Component Identification Register 1	Yes
0xFF8	PPU_CIDR2	–	32-bit	PPU Component Identification Register 2	Yes
0xFFC	PPU_CIDR3	–	32-bit	PPU Component Identification Register 3	Yes

B.1.5.1 PPU_PWPR, Power Policy Register

This register enables software to program both power and operating mode policy. It also contains related settings including the enable for dynamic transitions and the lock enable.

This register does not reflect the current power mode value. The current power mode of the domain is reflected in the Power Status Register (PPU_PWSR).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x000

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxx0 xxx0 xxxx 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-89: ext_ppu_pwpr bit assignments

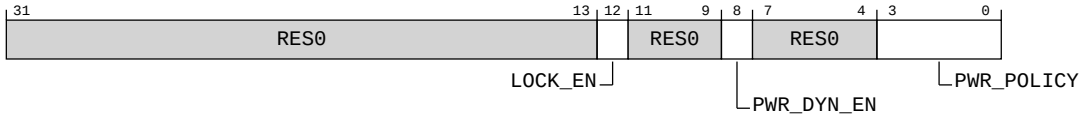


Table B-94: PPU_PWPR bit descriptions

Bits	Name	Description	Reset
[31:13]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[12]	LOCK_EN	Lock enable bit for OFF and OFF_EMU power modes **0b0** Lock feature disabled. **0b1** Lock feature enabled.	0b0
[11:9]	**RES0**	Reserved	**RES0**
[8]	PWR_DYN_EN	Power mode dynamic transition enable. **0b0** Dynamic transitions disabled for power modes. **0b1** Dynamic transitions enabled for power modes, allowing transitions to be initiated by changes on power mode DEVACTIVE inputs.	0b0
[7:4]	**RES0**	Reserved	**RES0**
[3:0]	PWR_POLICY	Power mode policy. When static power mode transitions are enabled, PWR_DYN_EN is set to 0b0, this is the target power mode for the PPU. When dynamic power mode transitions are enabled, PWR_DYN_EN is set to 0b1, this is the minimum power mode for the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0101** FULL_RET. Full Retention. Logic and RAM in retention. **0b0111** FUNC_RET. Functional Retention. Floating-point/Vector logic retained, rest of the core logic and RAM on, core is functional. **0b1000** ON. Logic on with RAM on, core is functional. **0b1001** WARM_RST. Warm reset . Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.5.2 PPU_PMER, Power Mode Emulation Enable Register

This register allows software to enable entry into emulated modes.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x004

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-90: ext_ppu_pmer bit assignments

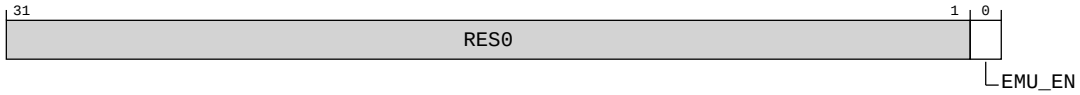


Table B-95: PPU_PMER bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	EMU_EN	Power mode emulation enable. **0b0** Power mode emulation disabled. **0b1** Power mode emulation enabled. Transitions to OFF and MEM_RET instead transition to OFF_EMU and MEM_RET_EMU.	0b0

B.1.5.3 PPU_PWSR, Power Status Register

This read-only register contains status information for the power mode, operating mode, dynamic transitions, and lock feature.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x008

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxx0 xxx0 xxxx 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-91: ext_ppu_pwsr bit assignments

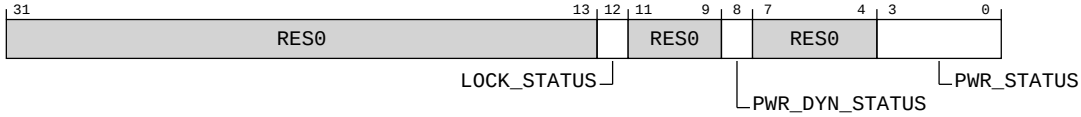


Table B-96: PPU_PWSR bit descriptions

Bits	Name	Description	Reset
[31:13]	**RES0**	Reserved	**RES0**
[12]	LOCK_STATUS	Lock status. **0b0** The PPU is not locked in the current mode. **0b1** The PPU is locked in the current mode.	0b0
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8]	PWR_DYN_STATUS	Power mode dynamic transition status. There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when PPU_PWPR.DYN_EN is programmed. **0b0** Dynamic transitions disabled for power modes. **0b1** Dynamic transitions enabled for power modes.	0b0
[7:4]	**RES0**	Reserved	**RES0**
[3:0]	PWR_STATUS	Power mode status. These bits reflect the current power mode of the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0101** FULL_RET. Full Retention. Logic and RAM in retention. **0b0111** FUNC_RET. Functional Retention. Floating-point/Vector logic retained, rest of the core logic and RAM on, core is functional. **0b1000** ON. Logic on with RAM on, core is functional. **0b1001** WARM_RST. Warm reset . Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.5.4 PPU_DISR, Device Interface Input Current Status Register

This read-only register contains status reflecting the values of the device interface inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x010

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx x000 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-92: ext_ppu_disr bit assignments

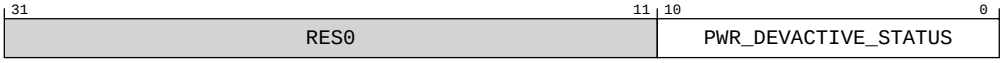


Table B-97: PPU_DISR bit descriptions

Bits	Name	Description	Reset
[31:11]	**RES0**	Reserved	**RES0**
[10:0]	PWR_DEVACTIVE_STATUS	Status of the power mode DEVPACTIVE inputs. **0b00000000000** Minimum mode OFF. **0000000001x** Minimum mode OFF_EMU. **000001xxxxx** Minimum mode FULL_RET. **0001xxxxxxx** Minimum mode FUNC_RET. **001xxxxxxxx** Minimum mode ON. **01xxxxxxxxx** Minimum mode WARM_RST. **1xxxxxxxxxx** Minimum mode DBG_RECOV.	0b00000000000

B.1.5.5 PPU_MISR, Miscellaneous Input Current Status Register

This read-only register contains status reflecting the values of miscellaneous inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x014

Access type

RO

Reset value

xxxx xxxx xxxx xxx0 xxxx xxx0 xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-93: ext_ppu_misr bit assignments

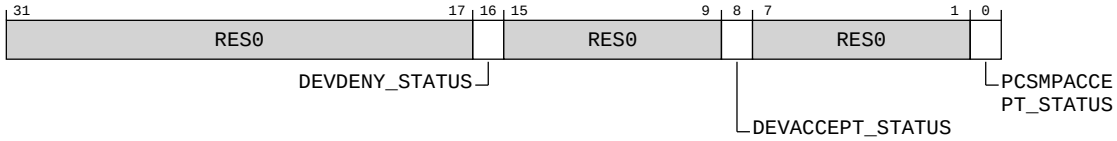


Table B-98: PPU_MISR bit descriptions

Bits	Name	Description	Reset
[31:17]	**RES0**	Reserved	**RES0**
[16]	DEVDENY_STATUS	Status of the device interface DEVPDENY inputs. **0b0** DEVPDENY deasserted. **0b1** DEVPDENY asserted.	0b0
[15:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8]	DEVACCEPT_STATUS	Status of the device interface DEVPACCEPT inputs. **0b0** DEVPACCEPT deasserted. **0b1** DEVPACCEPT asserted.	0b0
[7:1]	**RES0**	Reserved	**RES0**
[0]	PCSMPACCEPT_STATUS	Status of the PCSMPACCEPT inputs. **0b0** PCSMPACCEPT deasserted. **0b1** PCSMPACCEPT asserted.	0b0

B.1.5.6 PPU_STSR, Stored Status Register

This register is reserved for P-Channel PPUs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x018

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-94: ext_ppu_stsr bit assignments

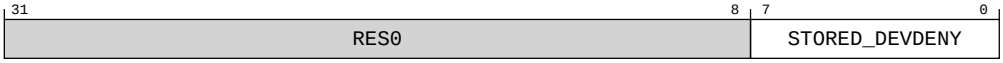


Table B-99: PPU_STSR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	STORED_DEVDENY	Status of the DEVDENY signals from the last device interface Q-Channel transition. This field is reserved. **0b00000000** Reserved for P-Channel PPUs.	8{x}

B.1.5.7 PPU_UNLK, Unlock Register

This register allows software to unlock the PPU from a locked power mode.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x01C

Access type

UNKNOWNW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-95: ext_ppu_unlk bit assignments

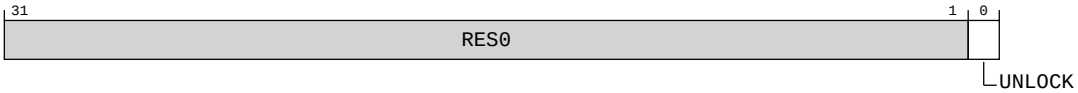


Table B-100: PPU_UNLK bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	UNLOCK	When `0b1` is written to this bit the PPU is unlocked from a locked power mode. A read always returns `0b0`.	`x`

B.1.5.8 PPU_PWCR, Power Configuration Register

This register controls enabling and disabling of hardware control inputs to the PPU.

Before software programs the DEVREQEN bits it must configure the PPU for static transitions and ensure the requested power mode has been reached, this means that no further transitions can occur, otherwise behavior is UNPREDICTABLE.

The PWR_DEVACTIVEEN and OP_DEVACTIVEEN fields in this register control the ability of the DEVACTIVE inputs to initiate power mode transitions, but not the ability to generate input edge interrupt events.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x020

Access type

RW

Reset value

xxxx xxxx xxxx x111 1111 111x xxxx xxx1

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-96: ext_ppu_pwcr bit assignments

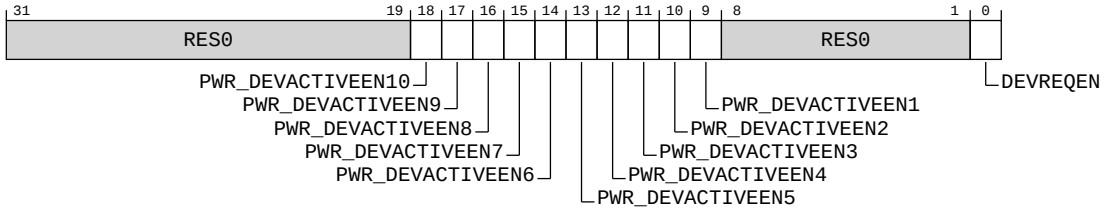


Table B-101: PPU_PWCR bit descriptions

Bits	Name	Description	Reset
[31:19]	**RES0**	Reserved	**RES0**
[18]	PWR_DEVACTIVEEN10	Enables the operating mode DEVPACTIVE[10] input. **0b0** DEVPACTIVE[10] input (DBG_RECOV) disabled. **0b1** DEVPACTIVE[10] input (DBG_RECOV) enabled.	0b1
[17]	PWR_DEVACTIVEEN9	Enables the operating mode DEVPACTIVE[9] input. **0b0** DEVPACTIVE[9] input (WARM_RST) disabled. **0b1** DEVPACTIVE[9] input (WARM_RST) enabled.	0b1
[16]	PWR_DEVACTIVEEN8	Enables the operating mode DEVPACTIVE[8] input. **0b0** DEVPACTIVE[8] input (ON) disabled. **0b1** DEVPACTIVE[8] input (ON) enabled.	0b1
[15]	PWR_DEVACTIVEEN7	Enables the operating mode DEVPACTIVE[7] input. **0b0** DEVPACTIVE[7] input (FUNC_RET) disabled. **0b1** DEVPACTIVE[7] input (FUNC_RET) enabled.	0b1
[14]	PWR_DEVACTIVEEN6	Enables the operating mode DEVPACTIVE[6] input. **0b1** DEVPACTIVE[6] input (MEM_OFF) enabled.	0b1

Bits	Name	Description	Reset
[13]	PWR_DEVACTIVEEN5	Enables the operating mode DEVPACTIVE[5] input. **0b0** DEVPACTIVE[5] input (FULL_RET) disabled. **0b1** DEVPACTIVE[5] input (FULL_RET) enabled.	0b1
[12]	PWR_DEVACTIVEEN4	Enables the operating mode DEVPACTIVE[4] input. **0b1** DEVPACTIVE[4] input (LOGIC_RET) enabled.	0b1
[11]	PWR_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[3] input. **0b1** DEVPACTIVE[3] input (MEM_RET_EMU) enabled.	0b1
[10]	PWR_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[2] input. **0b1** DEVPACTIVE[2] input (MEM_RET) enabled.	0b1
[9]	PWR_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[1] input. **0b0** DEVPACTIVE[1] input (OFF_EMU) disabled. **0b1** DEVPACTIVE[1] input (OFF_EMU) enabled.	0b1
[8:1]	**RES0**	Reserved	**RES0**
[0]	DEVREQEN	Device interface handshake enable. **0b0** Device interface handshake disabled for transitions. **0b1** Device interface handshake enabled for transitions.	0b1

B.1.5.9 PPU_PTCR, Power Mode Transition Register

This register contains settings which affect the behaviour of certain power mode transitions.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x024

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-97: ext_ppu_ptcr bit assignments

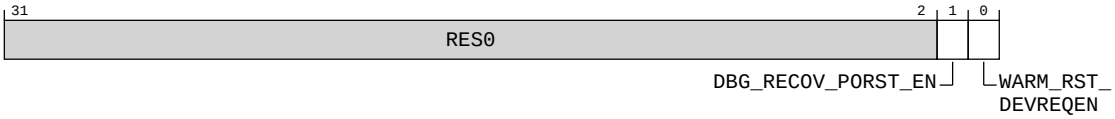


Table B-102: PPU_PTCR bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	DBG_RECOV_PORST_EN	Power-on reset behavior in DBG_RECOV. This bit should not be modified when the PPU is in DBG_RECOV or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **0b0** DEVPORESETn is not asserted when in DBG_RECOV. **0b1** DEVPORESETn is asserted when in DBG_RECOV.	0b0
[0]	WARM_RST_DEVREQEN	Device interface handshake behavior. This bit should not be modified when the PPU is in WARM_RST, or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **0b0** The PPU does not perform a device interface handshake when transitioning between ON and WARM_RST. **0b1** The PPU performs a device interface handshake when transitioning between ON and WARM_RST.	0b0

B.1.5.10 PPU_IMR, Interrupt Mask Register

This register controls the events that assert the interrupt output. Additional event masking controls are in the Additional Interrupt Mask Register (ext-PPU_AIMR), Input Edge Sensitivity Register (ext-PPU_IESR), and the Operating Mode Active Edge Sensitivity Register (ext-PPU_OPSR).

When an interrupt event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x030

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx11 1010

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-98: ext_ppu_imr bit assignments

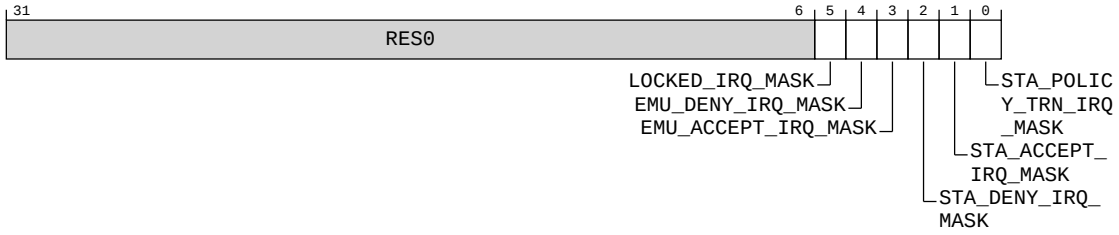


Table B-103: PPU_IMR bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	LOCKED_IRQ_MASK	Locked event mask **0b0** Locked event enabled. **0b1** Locked event masked.	0b1
[4]	EMU_DENY_IRQ_MASK	Emulation transition denial event mask **0b0** Emulation transition denial event enabled. **0b1** Emulation transition denial event masked.	0b1
[3]	EMU_ACCEPT_IRQ_MASK	Emulation transition acceptance event mask **0b0** Emulation transition acceptance event enabled. **0b1** Emulation transition acceptance event masked.	0b1
[2]	STA_DENY_IRQ_MASK	Static transition denial event mask **0b0** Static transition denial event enabled. **0b1** Static transition denial event masked.	0b0
[1]	STA_ACCEPT_IRQ_MASK	Static transition acceptance event mask **0b0** Static transition acceptance event enabled. **0b1** Static transition acceptance event masked.	0b1
[0]	STA_POLICY_TRN_IRQ_MASK	Static full policy transition completion event mask **0b0** Static full policy transition completion event enabled. **0b1** Static full policy transition completion event masked.	0b0

B.1.5.11 PPU_AIMR, Additional Interrupt Mask Register

This register controls the events that assert the interrupt output. Additional event masking controls are in the Interrupt Mask Register (PPU_IMR), Input Edge Sensitivity Register (PPU_IESR), and the Operating Mode Active Edge Sensitivity Register (PPU_OPSR).

When an interrupt event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x034

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx x110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-99: ext_ppu_aimr bit assignments

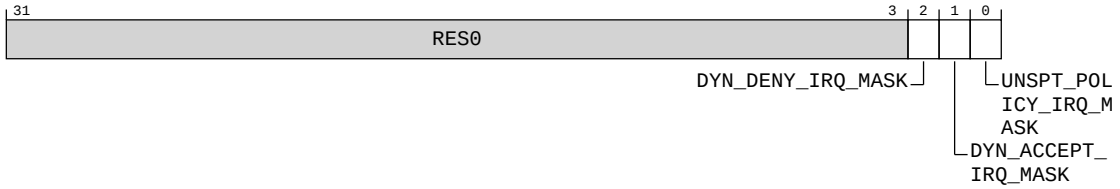


Table B-104: PPU_AIMR bit descriptions

Bits	Name	Description	Reset
[31:3]	**RES0**	Reserved	**RES0**
[2]	DYN_DENY_IRQ_MASK	Dynamic transition denial event mask **0b0** Dynamic transition denial event enabled. **0b1** Dynamic transition denial event masked.	0b1

Bits	Name	Description	Reset
[1]	DYN_ACCEPT_IRQ_MASK	Dynamic transition acceptance event mask **0b0** Dynamic transition acceptance event enabled. **0b1** Dynamic transition acceptance event masked.	0b1
[0]	UNSPT_POLICY_IRQ_MASK	Unsupported policy event mask **0b0** Unsupported policy event enabled. **0b1** Unsupported policy event masked.	0b0

B.1.5.12 PPU_ISR, Interrupt Status Register

This register contains information about events causing the assertion of the interrupt output. It is also used to clear interrupt events.

A bit set to 0b1 indicates the event asserted the interrupt output. Multiple events can be active at the same time. When an interrupt event is masked an occurrence of that event does not set the status bit.

A write of 0b1 to an event bit clears that event. A write of 0b0 to a bit has no effect. The interrupt output stays HIGH until all status bits in the Interrupt Status Register (PPU_ISR) and the Additional Interrupt Status Register (PPU_AISR) are 0b0.

When the OTHER_IRQ bit is set, this indicates an event from the Additional Interrupt Status Register (PPU_AISR) has caused the interrupt output to be asserted. This bit cannot be cleared by writing to this register. It must be cleared by writing to the active event in the Additional Interrupt Status Register (PPU_AISR).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x038

Access type

RW

Reset value

xxxx xxxx xxxx x000 0x0x xx0x 0x00 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-100: ext_ppu_isr bit assignments

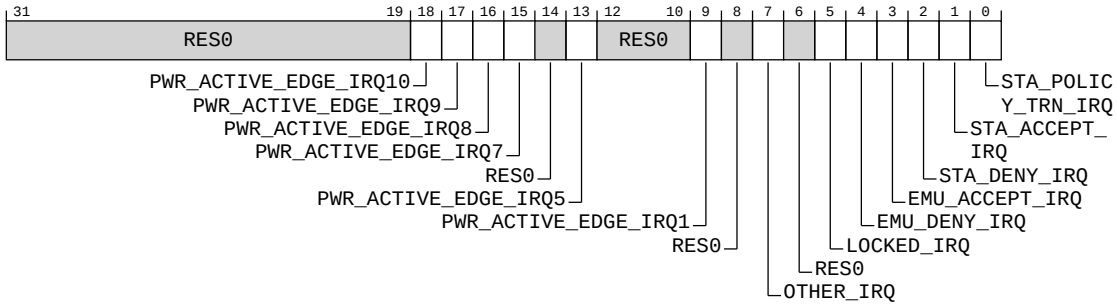


Table B-105: PPU_ISR bit descriptions

Bits	Name	Description	Reset
[31:19]	**RES0**	Reserved	**RES0**
[18]	PWR_ACTIVE_EDGE_IRQ10	Indicates if power mode DEVPACTIVE[10] input caused the input edge event. **0b0** DEVPACTIVE[10] input (DBG_RECOV) did not assert the interrupt output. **0b1** DEVPACTIVE[10] input (DBG_RECOV) asserted the interrupt output.	0b0
[17]	PWR_ACTIVE_EDGE_IRQ9	Indicates if power mode DEVPACTIVE[9] input caused the input edge event. **0b0** DEVPACTIVE[9] input (WARM_RST) did not assert the interrupt output. **0b1** DEVPACTIVE[9] input (WARM_RST) asserted the interrupt output.	0b0
[16]	PWR_ACTIVE_EDGE_IRQ8	Indicates if power mode DEVPACTIVE[8] input caused the input edge event. **0b0** DEVPACTIVE[8] input (ON) did not assert the interrupt output. **0b1** DEVPACTIVE[8] input (ON) asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[15]	PWR_ACTIVE_EDGE_IRQ7	Indicates if power mode DEVPACTIVE[7] input caused the input edge event. **0b0** DEVPACTIVE[7] input (FUNC_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[7] input (FUNC_RET) asserted the interrupt output.	0b0
[14]	**RES0**	Reserved	**RES0**
[13]	PWR_ACTIVE_EDGE_IRQ5	Indicates if power mode DEVPACTIVE[5] input caused the input edge event. **0b0** DEVPACTIVE[5] input (FULL_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[5] input (FULL_RET) asserted the interrupt output.	0b0
[12:10]	**RES0**	Reserved	**RES0**
[9]	PWR_ACTIVE_EDGE_IRQ1	Indicates if power mode DEVPACTIVE[1] input caused the input edge event. **0b0** DEVPACTIVE[1] input (OFF_EMU) did not assert the interrupt output. **0b1** DEVPACTIVE[1] input (OFF_EMU) asserted the interrupt output.	0b0
[8]	**RES0**	Reserved	**RES0**
[7]	OTHER_IRQ	Indicates there is an interrupt event pending in the Additional Interrupt Status Register (PPU_AISR). **0b0** No interrupt pending in PPU_AISR. **0b1** Interrupt pending in PPU_AISR.	0b0
[6]	**RES0**	Reserved	**RES0**
[5]	LOCKED_IRQ	Locked event status. **0b0** No locked event. **0b1** A locked event asserted the interrupt output.	0b0
[4]	EMU_DENY_IRQ	Emulated transition denial event status. **0b0** No emulated transition denial event. **0b1** An emulated transition denial event asserted the interrupt output.	0b0
[3]	EMU_ACCEPT_IRQ	Emulated transition acceptance event status. **0b0** No emulated transition acceptance event. **0b1** An emulated transition acceptance event asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[2]	STA_DENY_IRQ	Static transition denial event status. **0b0** No static transition denial event. **0b1** A static transition denial event asserted the interrupt output.	0b0
[1]	STA_ACCEPT_IRQ	Static transition acceptance event status. **0b0** No static transition acceptance event. **0b1** A static transition acceptance event asserted the interrupt output.	0b0
[0]	STA_POLICY_TRN_IRQ	Static full policy transition completion event status. **0b0** No static full policy transition completion event. **0b1** A static full policy transition completion event asserted the interrupt output.	0b0

B.1.5.13 PPU_AISR, Additional Interrupt Status Register

This register contains information about events causing the assertion of the interrupt output. It is also used to clear interrupt events.

A bit set to 0b1 indicates the event asserted the interrupt output. Multiple events can be active at the same time. When an interrupt event is masked by the corresponding bit in PPU_AIMR, an occurrence of that event does not set the status bit.

A write of 0b1 to a set event bit clears that event. A write of 0b0 has no effect. The interrupt output stays HIGH until all status bits in the Interrupt Status Register (PPU_ISR) and the Additional Interrupt Status Register (PPU_AISR) are set to 0b0.

When an interrupt status is set to 0b1 in this register it sets the OTHER_IRQ bit in the Interrupt Status Register (PPU_ISR). Status bits in this register (PPU_AISR) are only cleared by writing to this register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x03C

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx x000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-101: ext_ppu_aisr bit assignments

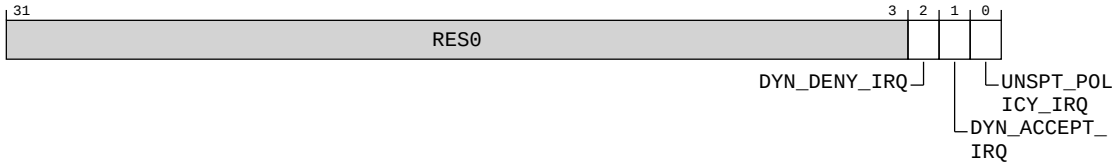


Table B-106: PPU_AISR bit descriptions

Bits	Name	Description	Reset
[31:3]	**RES0**	Reserved	**RES0**
[2]	DYN_DENY_IRQ	Dynamic transition denial event status **0b0** No dynamic transition denial event. **0b1** A dynamic transition denial event asserted the interrupt output.	0b0
[1]	DYN_ACCEPT_IRQ	Dynamic transition acceptance event status **0b0** No dynamic transition acceptance event. **0b1** A dynamic transition acceptance event asserted the interrupt output.	0b0
[0]	UNSPT_POLICY_IRQ	Unsupported policy event status **0b0** No unsupported policy event. **0b1** An unsupported policy event asserted the interrupt output.	0b0

B.1.5.14 PPU_IESR, Input Edge Sensitivity Register

This register configures the transitions on the power mode DEVPACTIVE inputs that generate an Input Edge interrupt event.

When an event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x040

Access type

RW

Reset value

xxxx xxxx xx00 0000 00xx 00xx xxxx 00xx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-102: ext_ppu_iesr bit assignments

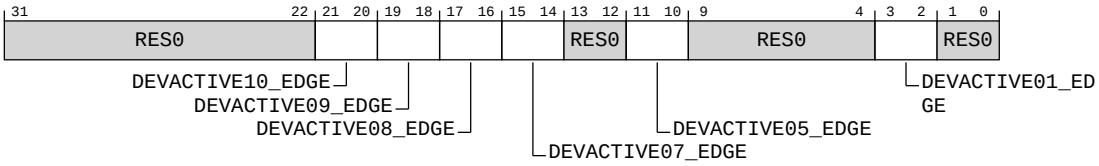


Table B-107: PPU_IESR bit descriptions

Bits	Name	Description	Reset
[31:22]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[21:20]	DEVACTIVE10_EDGE	Configures the transitions on the DEVPACTIVE[10] input (DBG_RECOV) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[19:18]	DEVACTIVE09_EDGE	Configures the transitions on the DEVPACTIVE[9] input (WARM_RST) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[17:16]	DEVACTIVE08_EDGE	Configures the transitions on the DEVPACTIVE[8] input (ON) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[15:14]	DEVACTIVE07_EDGE	Configures the transitions on the DEVPACTIVE[7] input (FUNC_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[13:12]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[11:10]	DEVACTIVE05_EDGE	Configures the transitions on the DEVPACTIVE[5] input (FULL_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[9:4]	**RES0**	Reserved	**RES0**
[3:2]	DEVACTIVE01_EDGE	Configures the transitions on the DEVPACTIVE[1] input (OFF_EMU) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[1:0]	**RES0**	Reserved	**RES0**

B.1.5.15 PPU_OPSR, Operating Mode Active Edge Sensitivity Register

This register configures the transitions on the operating mode DEVPACTIVE inputs that generate an Input Edge interrupt event.

When an event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x044

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-103: ext_ppu_opsr bit assignments

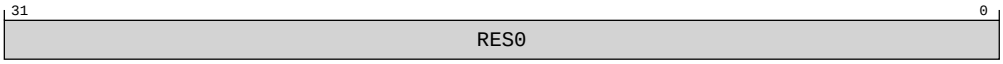


Table B-108: PPU_OPSR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.5.16 PPU_FUNRR, Functional Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x050

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-104: ext_ppu_funrr bit assignments

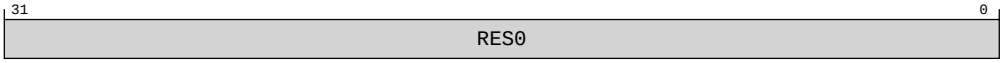


Table B-109: PPU_FUNRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.5.17 PPU_FULRR, Full Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x054

Access type

RW

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-105: ext_ppu_fulrr bit assignments

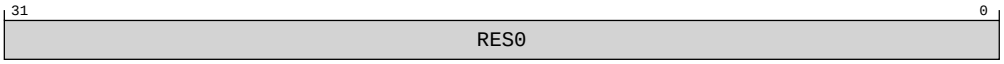


Table B-110: PPU_FULRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.5.18 PPU_MEMRR, Memory Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x058

Access type

RW

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-106: ext_ppu_memrr bit assignments

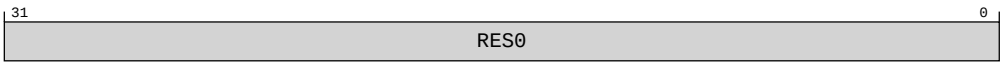


Table B-111: PPU_MEMRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.5.19 PPU_DCDR0, Device Control Delay Configuration Register 0

This register is used to program device control delay parameters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x170

Access type

RW

Reset value

xxxx xxxx 0000 0000 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-107: ext_ppu_dcdr0 bit assignments

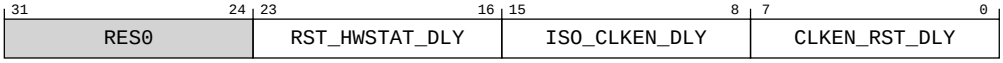


Table B-112: PPU_DCDR0 bit descriptions

Bits	Name	Description	Reset
[31:24]	**RES0**	Reserved	**RES0**
[23:16]	RST_HWSTAT_DLY	Delay from reset de-assertion to HWSTAT update.	0x00
[15:8]	ISO_CLKEN_DLY	Delay from isolation enable de-assertion to clock enable assertion.	0x00
[7:0]	CLKEN_RST_DLY	Delay from clock enable assertion to reset de-assertion.	0x00

B.1.5.20 PPU_DCDR1, Device Control Delay Configuration Register 1

This register is used to program device control delay parameters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0x174

Access type

RW

Reset value

xxxx xxxx xxxx xxxx 0000 0000 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-108: ext_ppu_dcdr1 bit assignments

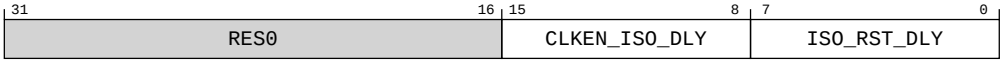


Table B-113: PPU_DCDR1 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:8]	CLKEN_ISO_DLY	Delay from clock enable de-assertion to isolation enable assertion.	0x00
[7:0]	ISO_RST_DLY	Delay from isolation enable assertion to reset assertion.	0x00

B.1.5.21 PPU_IDR0, PPU Identification Register 0

This read-only register contains information on the type and number of channels on the device interface and power and operating modes supported.

Additional information on optional features can be found in the PPU Identification Register 1 (PPU_IDR1).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFB0

Access type

RO

Reset value

xx01 1100 0011 x111 1100 0011 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-109: ext_ppu_idr0 bit assignments

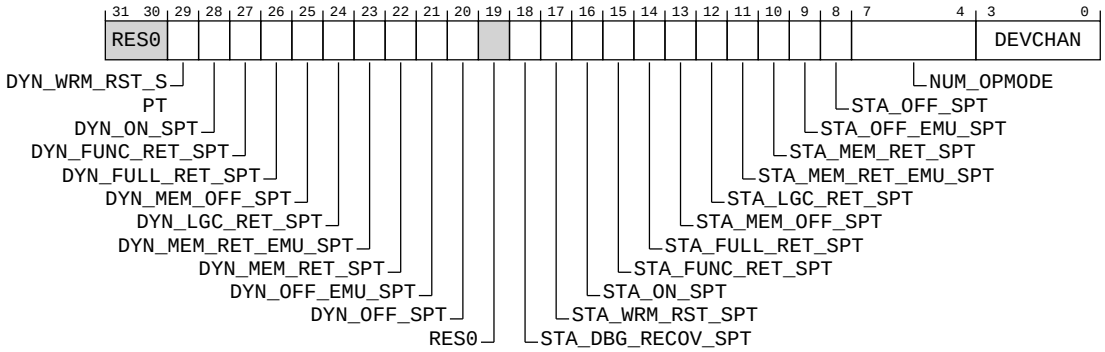


Table B-114: PPU_IDR0 bit descriptions

Bits	Name	Description	Reset
[31:30]	**RES0**	Reserved	**RES0**
[29]	DYN_WRM_RST_SPT	Dynamic WARM_RST support. **0b0** Dynamic WARM_RST not supported.	0b0
[28]	DYN_ON_SPT	Dynamic ON support. **0b1** Dynamic ON supported.	0b1
[27]	DYN_FUNC_RET_SPT	Dynamic DYN_FUNC_RET_SPT support. **0b1** Dynamic DYN_FUNC_RET_SPT supported.	0b1
[26]	DYN_FULL_RET_SPT	Dynamic DYN_FULL_RET_SPT support. **0b1** Dynamic DYN_FULL_RET_SPT supported.	0b1
[25]	DYN_MEM_OFF_SPT	Dynamic MEM_OFF support. **0b0** Dynamic MEM_OFF not supported.	0b0
[24]	DYN_LGC_RET_SPT	Dynamic LOGIC_RET support. **0b0** Dynamic LOGIC_RET not supported.	0b0
[23]	DYN_MEM_RET_EMU_SPT	Dynamic DYN_MEM_RET_EMU_SPT support. **0b0** Dynamic DYN_MEM_RET_EMU_SPT not supported.	0b0
[22]	DYN_MEM_RET_SPT	Dynamic DYN_MEM_RET_SPT support. **0b0** Dynamic DYN_MEM_RET_SPT not supported.	0b0
[21]	DYN_OFF_EMU_SPT	Dynamic OFF_EMU support. **0b1** Dynamic OFF_EMU supported.	0b1
[20]	DYN_OFF_SPT	Dynamic OFF support. **0b1** Dynamic OFF supported.	0b1
[19]	**RES0**	Reserved	**RES0**
[18]	STA_DBG_RECOV_SPT	DBG_RECOV support. **0b1** DBG_RECOV supported.	0b1
[17]	STA_WRM_RST_SPT	WARM_RST support. **0b1** WRM_RST supported.	0b1

Bits	Name	Description	Reset
[16]	STA_ON_SPT	ON support. **0b1** ON supported.	0b1
[15]	STA_FUNC_RET_SPT	FUNC_RET support. **0b1** FUNC_RET supported.	0b1
[14]	STA_FULL_RET_SPT	FULL_RET support. **0b1** FULL_RET supported.	0b1
[13]	STA_MEM_OFF_SPT	MEM_OFF support. **0b0** MEM_OFF not supported.	0b0
[12]	STA_LGC_RET_SPT	LOGIC_RET support. **0b0** LOGIC_RET not supported.	0b0
[11]	STA_MEM_RET_EMU_SPT	MEM_RET_EMU support. **0b0** MEM_RET_EMU not supported.	0b0
[10]	STA_MEM_RET_SPT	MEM_RET support. **0b0** MEM_RET not supported.	0b0
[9]	STA_OFF_EMU_SPT	OFF_EMU support. **0b1** OFF_EMU supported.	0b1
[8]	STA_OFF_SPT	OFF support. **0b1** OFF supported.	0b1
[7:4]	NUM_OPMODE	No. of operating modes supported, minus 1. **0b0000** 1 operating mode supported.	0b0000
[3:0]	DEVCHAN	No. of Device Interface Channels. **0b0000** 0 (P-channel PPU).	0b0000

B.1.5.22 PPU_IDR1, PPU Identification Register 1

This read-only register contains information on the optional features and configurations that are supported by this PPU.

Additional information on optional features can be found in the PPU Identification Register 0 (PPU_IDR0).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFB4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxx0 x000 x000 x110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-110: ext_ppu_idr1 bit assignments

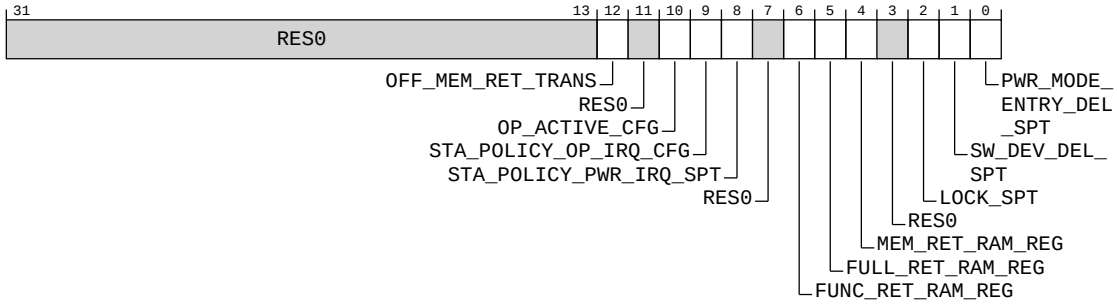


Table B-115: PPU_IDR1 bit descriptions

Bits	Name	Description	Reset
[31:13]	**RES0**	Reserved	**RES0**
[12]	OFF_MEM_RET_TRANS	OFF to MEM_RET direct transition. Indicates if direct transitions from OFF to MEM_RET and from OFF_EMU to MEM_RET_EMU are supported. **0b0** OFF to MEM_RET direct transition not supported.	0b0
[11]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[10]	OP_ACTIVE_CFG	Operating mode use model for dynamic transitions. **0b0** Ladder use model.	0b0
[9]	STA_POLICY_OP_IRQ_CFG	Operating policy transition completion event status. **0b0** Operating policy transition completion events not supported.	0b0
[8]	STA_POLICY_PWR_IRQ_SPT	Power policy transition completion event status. **0b0** Power policy transition completion events not supported.	0b0
[7]	**RES0**	Reserved	**RES0**
[6]	FUNC_RET_RAM_REG	Indicates if the PPU_FUNRR register is present or reserved. **0b0** PPU_FUNRR is reserved.	0b0
[5]	FULL_RET_RAM_REG	Indicates if the PPU_FULRR register is present or reserved. **0b0** PPU_FULRR is reserved.	0b0
[4]	MEM_RET_RAM_REG	Indicates if the PPU_MEMRR register is present or reserved. **0b0** PPU_MEMRR is reserved.	0b0
[3]	**RES0**	Reserved	**RES0**
[2]	LOCK_SPT	Indicates if the lock and the lock interrupt event are supported. **0b1** Lock and the lock interrupt event are supported.	0b1
[1]	SW_DEV_DEL_SPT	Software device delay control configuration support. **0b1** Software device delay control configuration supported.	0b1
[0]	PWR_MODE_ENTRY_DEL_SPT	Power mode entry delay support. **0b0** Power mode entry delay not supported.	0b0

B.1.5.23 PPU_IIDR, Implementation Identification Register

This register provides information about the implementer and implementation of the PPU.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFC8

Access type

RO

Reset value

0000 1011 0110 0001 0001 0100 0011 1011

Bit descriptions

Figure B-111: ext_ppu_iidr bit assignments

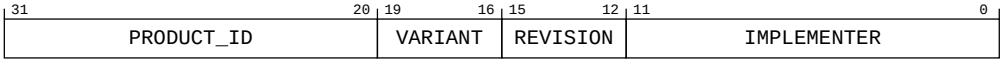


Table B-116: PPU_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	PRODUCT_ID	Value identifying the PPU part. **0b000010110110** Power Policy Unit.	0x0B6
[19:16]	VARIANT	Value used to distinguish PPU variants, or major revisions of the PPU. **0b0000** PPU variant 0. **0b0001** PPU variant 1. **0b0010** PPU variant 2. **0b0011** PPU variant 3. **0b0100** PPU variant 4.	0b0001

Bits	Name	Description	Reset
[15:12]	REVISION	Value used to distinguish minor revisions of the PPU. **0b0000** PPU revision 0. **0b0001** PPU revision 1. **0b0010** PPU revision 2. **0b0011** PPU revision 3. **0b0100** PPU revision 4.	0b0001
[11:0]	IMPLEMENTER	Implementer identification. **0b010000111011** Arm Limited.	0x43B

B.1.5.24 PPU_AIDR, Architecture Identification Register

This register identifies the PPU architecture revision.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFCC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0001

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-112: ext_ppu_aidr bit assignments

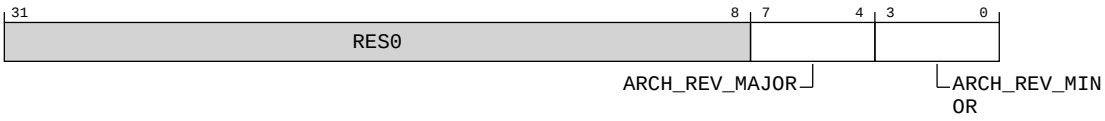


Table B-117: PPU_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	ARCH_REV_MAJOR	PPU architecture major revision. **0b0001** PPU architecture major revision 1.	0b0001
[3:0]	ARCH_REV_MINOR	PPU architecture minor revision. **0b0001** PPU architecture minor revision 1.	0b0001

B.1.5.25 PPU_PIDR4, PPU Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFD0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-113: ext_ppu_pidr4 bit assignments

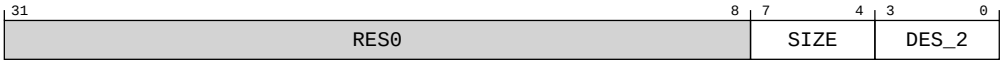


Table B-118: PPU_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	xxxx
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	xxxx

B.1.5.26 PPU_PIDR5, PPU Peripheral Identification Register 5

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFD4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-114: ext_ppu_pidr5 bit assignments

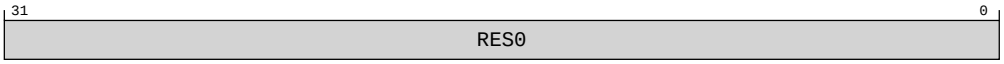


Table B-119: PPU_PIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.5.27 PPU_PIDR6, PPU Peripheral Identification Register 6

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFD8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-115: ext_ppu_pidr6 bit assignments

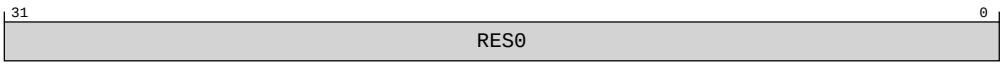


Table B-120: PPU_PIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.5.28 PPU_PIDR7, PPU Peripheral Identification Register 7

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFDC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-116: ext_ppu_pidr7 bit assignments

Table B-121: PPU_PIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.5.29 PPU_PIDR0, PPU Peripheral Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFE0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-117: ext_ppu_pidr0 bit assignments

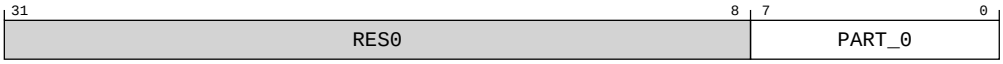


Table B-122: PPU_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **0b10110110** Core Power Policy Unit. Bits [7:0] of part number 0x0B6.	0xB6

B.1.5.30 PPU_PIDR1, PPU Peripheral Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFE4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-118: ext_ppu_pidr1 bit assignments

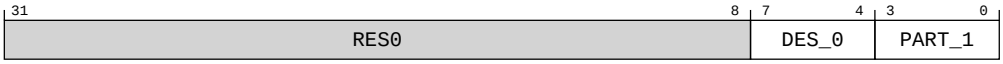


Table B-123: PPU_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0000** Core Power Policy Unit. Bits [11:8] of part number 0x0B6.	0b0000

B.1.5.31 PPU_PIDR2, PPU Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFE8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 1011

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-119: ext_ppu_pidr2 bit assignments

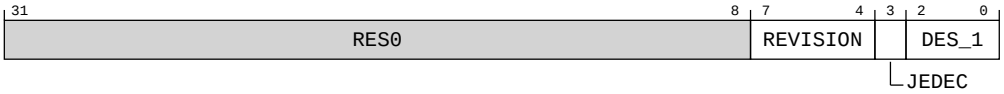


Table B-124: PPU_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4.	0b0001
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

B.1.5.32 PPU_PIDR3, PPU Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFEC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-120: ext_ppu_pidr3 bit assignments

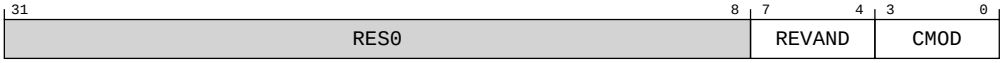


Table B-125: PPU_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0. **0b0001** Component minor revision 1. **0b0010** Component minor revision 2. **0b0011** Component minor revision 3. **0b0100** Component minor revision 4.	0b0001
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

B.1.5.33 PPU_CIDR0, PPU Component Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFF0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-121: ext_ppu_cidr0 bit assignments

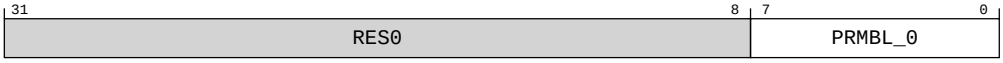


Table B-126: PPU_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

B.1.5.34 PPU_CIDR1, PPU Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFF4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1111 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-122: ext_ppu_cidr1 bit assignments

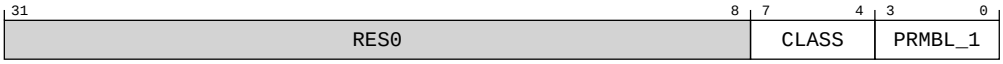


Table B-127: PPU_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	CoreSight component class. **0b1111** CoreLink component.	0b1111
[3:0]	PRMBL_1	CoreSight component identification preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.1.5.35 PPU_CIDR2, PPU Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFF8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-123: ext_ppu_cidr2 bit assignments

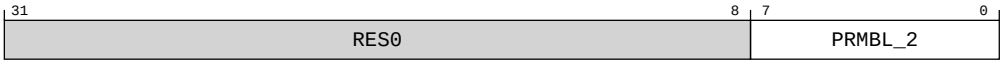


Table B-128: PPU_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	CoreSight component identification preamble. **0b00000101** CoreSight component identification preamble.	0x05

B.1.5.36 PPU_CIDR3, PPU Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

Register offset

0xFFC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-124: ext_ppu_cidr3 bit assignments

Table B-129: PPU_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **0b10110001** CoreSight component identification preamble.	0xB1

B.2 Registers accessed over the Debug APB bus

This section contains the descriptions for all the external registers in the *DynamIQ™ Shared Unit-110* (DSU-110) accessed over the Debug APB bus.

B.2.1 External CTI register summary

The summary table provides an overview of all the *Cross Trigger Interface* (CTI) registers that are accessed externally (memory-mapped) over the Debug APB interface. Individual register descriptions provide detailed information.

The summary table provides an overview of all the cluster CTI registers and core CTI registers. For more information about a register, click on the register name in the table.

Note

- Registers that differ in descriptions and values, for cluster and core, are indicated in the Identical CTI core column. These registers are the CTIPIDR0-4 registers, and the CTIDEVAFF0-1 registers.
- The cluster CTI registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.

- If the DSU-110 is configured for Direct connect all these registers are present.
- The cluster CTI part number is `0x4EA`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-130: CTI registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core CTI
0x000	CTICONTROL	–	32-bit	CTI Control register	Yes	Yes
0x010	CTIINTACK	–	32-bit	CTI Output Trigger Acknowledge register	Yes	Yes
0x014	CTIAPPSET	–	32-bit	CTI Application Trigger Set register	Yes	Yes
0x018	CTIAPPCLEAR	–	32-bit	CTI Application Trigger Clear register	Yes	Yes
0x01C	CTIAPPPULSE	–	32-bit	CTI Application Pulse register	Yes	Yes
0x20	CTIINEN0	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x24	CTIINEN1	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x28	CTIINEN2	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x2C	CTIINEN3	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x30	CTIINEN4	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x34	CTIINEN5	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x38	CTIINEN6	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x3C	CTIINEN7	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x40	CTIINEN8	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0x44	CTIINEN9	–	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes	Yes
0xA0	CTIOUTEN0	–	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xA4	CTIOUTEN1	–	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xA8	CTIOUTEN2	–	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xAC	CTIOUTEN3	–	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xB0	CTIOUTEN4	–	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xB4	CTIOUTEN5	–	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core CTI
0xB8	CTIOUTEN6	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xBC	CTIOUTEN7	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xC0	CTIOUTEN8	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0xC4	CTIOUTEN9	—	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes	Yes
0x130	CTITRIGINSTATUS	—	32-bit	CTI Trigger In Status register	Yes	Yes
0x134	CTITRIGOUTSTATUS	—	32-bit	CTI Trigger Out Status register	Yes	Yes
0x138	CTICHINSTATUS	—	32-bit	CTI Channel In Status register	Yes	Yes
0x13C	CTICHOUTSTATUS	—	32-bit	CTI Channel Out Status register	Yes	Yes
0x140	CTIGATE	—	32-bit	CTI Channel Gate Enable register	Yes	Yes
0x150	CTIDEVCTL	—	32-bit	CTI Device Control register	Yes	Yes
0xFA0	CTICLAIMSET	—	32-bit	CTI Claim Tag Set register	Yes	Yes
0xFA4	CTICLAIMCLR	—	32-bit	CTI Claim Tag Clear register	Yes	Yes
0xFA8	CTIDEVAFF0	—	32-bit	CTI Device Affinity register 0	Yes	No, see individual register
0xFAC	CTIDEVAFF1	—	32-bit	CTI Device Affinity register 1	Yes	No, see individual register
0xFB8	CTIAUTHSTATUS	—	32-bit	CTI Authentication Status register	Yes	Yes
0xFBC	CTIDEVARCH	—	32-bit	CTI Device Architecture register	Yes	Yes
0xFC0	CTIDEVID2	—	32-bit	CTI Device ID register 2	Yes	Yes
0xFC4	CTIDEVID1	—	32-bit	CTI Device ID register 1	Yes	Yes
0xFC8	CTIDEVID	—	32-bit	CTI Device ID register 0	Yes	Yes
0xFCC	CTIDEVTYPE	—	32-bit	CTI Device Type register	Yes	Yes
0xFD0	CTIPIDR4	—	32-bit	CTI Peripheral Identification Register 4	Yes	No, see individual register
0xFE0	CTIPIDR0	—	32-bit	CTI Peripheral Identification Register 0	Yes	No, see individual register
0xFE4	CTIPIDR1	—	32-bit	CTI Peripheral Identification Register 1	Yes	No, see individual register
0xFE8	CTIPIDR2	—	32-bit	CTI Peripheral Identification Register 2	Yes	No, see individual register
0xFEC	CTIPIDR3	—	32-bit	CTI Peripheral Identification Register 3	Yes	No, see individual register
0xFF0	CTICIDR0	—	32-bit	CTI Component Identification Register 0	Yes	Yes
0xFF4	CTICIDR1	—	32-bit	CTI Component Identification Register 1	Yes	Yes
0xFF8	CTICIDR2	—	32-bit	CTI Component Identification Register 2	Yes	Yes
0xFFC	CTICIDR3	—	32-bit	CTI Component Identification Register 3	Yes	Yes

B.2.1.1 CTICONTROL, CTI Control register

Controls whether the CTI is enabled.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x000

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-125: ext_cticontrol bit assignments

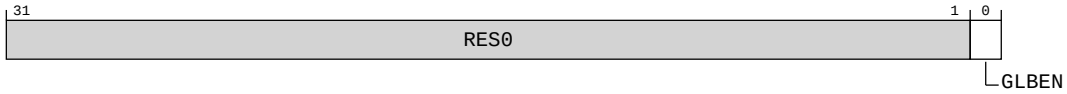


Table B-131: CTICONTROL bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[0]	GLBEN	Enables or disables the CTI mapping functions. Possible values of this field are: **0b0** CTI mapping functions and application trigger disabled. **0b1** CTI mapping functions and application trigger enabled. When GLBEN is 0, the input channel to output trigger, input trigger to output channel, and application trigger functions are disabled and do not signal new events on either output triggers or output channels. If a previously asserted output trigger has not been acknowledged, it remains asserted after the mapping functions are disabled. All output triggers are disabled by CTI reset.	0b0

Accessibility

Component	Offset	Instance
CTI	0x000	CTICONTROL

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.2 CTIINTACK, CTI Output Trigger Acknowledge register

Can be used to deactivate the output triggers.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x010

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 00xx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-126: ext_ctiintack bit assignments

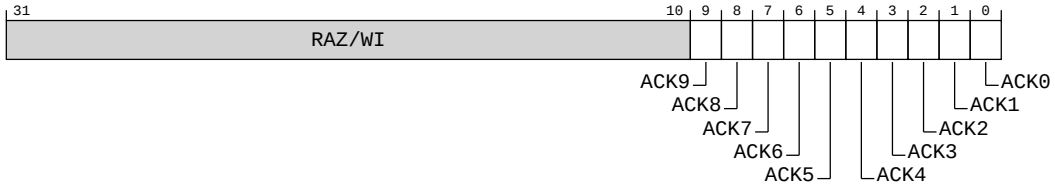


Table B-133: CTIINTACK bit descriptions

Bits	Name	Description	Reset
[31:10]	**RAZ/WI**	Reserved	**RAZ/WI**
[9]	ACK9	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x

Bits	Name	Description	Reset
[8]	ACK8	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x
[7]	ACK7	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x

Bits	Name	Description	Reset
[6]	ACK6	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x
[5]	ACK5	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x

Bits	Name	Description	Reset
[4]	ACK4	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x
[3]	ACK3	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x

Bits	Name	Description	Reset
[2]	ACK2	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x
[1]	ACK1	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x

Bits	Name	Description	Reset
[0]	ACK0	Acknowledge for output trigger <n>. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **0b0** No effect **0b1** Deactivate the trigger.	x

Accessibility

Component	Offset	Instance
CTI	0x010	CTIINTACK

This interface is accessible as follows:

When SoftwareLockStatus()

WI

When !SoftwareLockStatus()

WO

B.2.1.3 CTIAPPSET, CTI Application Trigger Set register

Sets bits of the Application Trigger register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x014

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-127: ext_ctiappset bit assignments

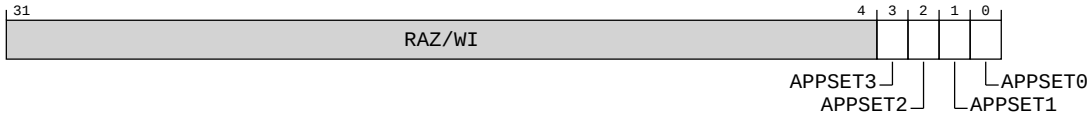


Table B-135: CTIAPPSET bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	APPSET3	Application trigger3 enable. Possible values of this bit are: **0b0** Reading this means the application trigger is inactive. Writing this has no effect. **0b1** Reading this means the application trigger is active. Writing this sets the corresponding bit in CTIAPPTRIG to 1 and generates a channel event.	0b0
[2]	APPSET2	Application trigger2 enable. Possible values of this bit are: **0b0** Reading this means the application trigger is inactive. Writing this has no effect. **0b1** Reading this means the application trigger is active. Writing this sets the corresponding bit in CTIAPPTRIG to 1 and generates a channel event.	0b0
[1]	APPSET1	Application trigger1 enable. Possible values of this bit are: **0b0** Reading this means the application trigger is inactive. Writing this has no effect. **0b1** Reading this means the application trigger is active. Writing this sets the corresponding bit in CTIAPPTRIG to 1 and generates a channel event.	0b0

Bits	Name	Description	Reset
[0]	APPSET0	Application trigger0 enable. Possible values of this bit are: **0b0** Reading this means the application trigger is inactive. Writing this has no effect. **0b1** Reading this means the application trigger is active. Writing this sets the corresponding bit in CTIAPPTRIG to 1 and generates a channel event.	0b0

Accessibility

Component	Offset	Instance
CTI	0x014	CTIAPPSET

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.4 CTIAPPCLEAR, CTI Application Trigger Clear register

Clears bits of the Application Trigger register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x018

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-128: ext_ctiappclear bit assignments

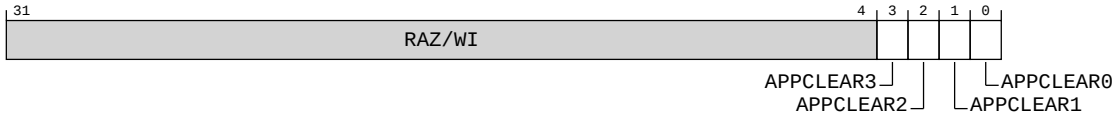


Table B-137: CTIAPPCLEAR bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	APPCLEAR3	Application trigger <x> disable. Writing to this bit has the following effect: **0b0** No effect. **0b1** Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.	x
[2]	APPCLEAR2	Application trigger <x> disable. Writing to this bit has the following effect: **0b0** No effect. **0b1** Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.	x
[1]	APPCLEAR1	Application trigger <x> disable. Writing to this bit has the following effect: **0b0** No effect. **0b1** Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.	x
[0]	APPCLEAR0	Application trigger <x> disable. Writing to this bit has the following effect: **0b0** No effect. **0b1** Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.	x

Accessibility

Component	Offset	Instance
CTI	0x018	CTIAPPCLEAR

This interface is accessible as follows:

When SoftwareLockStatus()

WI

When !SoftwareLockStatus()

WO

B.2.1.5 CTIAPPPULSE, CTI Application Pulse register

Causes event pulses to be generated on ECT channels.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x01C

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-129: ext_ctiapppulse bit assignments

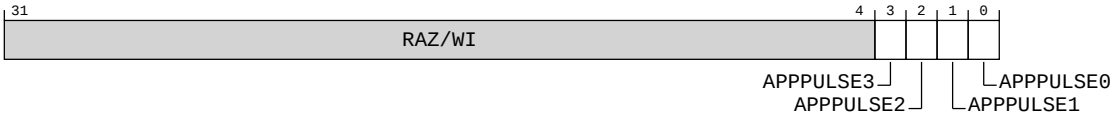


Table B-139: CTIAPPPULSE bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	APPPULSE3	Generate event pulse on ECT channel <x>. Writing to this bit has the following effect: **0b0** No effect. **0b1** Channel <x> event pulse generated.	x
[2]	APPPULSE2	Generate event pulse on ECT channel <x>. Writing to this bit has the following effect: **0b0** No effect. **0b1** Channel <x> event pulse generated.	x
[1]	APPPULSE1	Generate event pulse on ECT channel <x>. Writing to this bit has the following effect: **0b0** No effect. **0b1** Channel <x> event pulse generated.	x
[0]	APPPULSE0	Generate event pulse on ECT channel <x>. Writing to this bit has the following effect: **0b0** No effect. **0b1** Channel <x> event pulse generated.	x

Accessibility

Component	Offset	Instance
CTI	0x01C	CTIAPPPULSE

This interface is accessible as follows:

When SoftwareLockStatus()

WI

When !SoftwareLockStatus()

WO

B.2.1.6 CTIINEN0, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x20

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-130: ext_ctiinen0 bit assignments

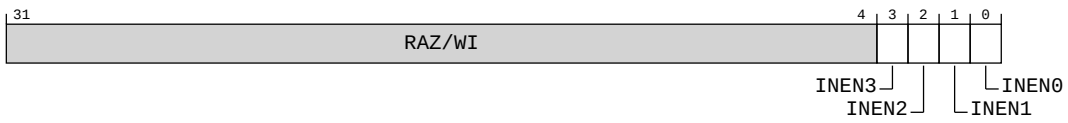


Table B-141: CTIINEN0 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x20	CTIINEN0

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.7 CTIINEN1, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x24

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-131: ext_ctiinen1 bit assignments

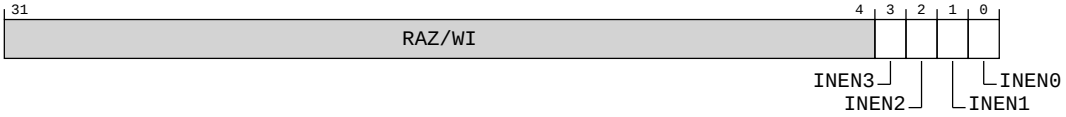


Table B-143: CTIINEN1 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x24	CTIINEN1

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.8 CTIINEN2, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x28

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-132: ext_ctiinen2 bit assignments

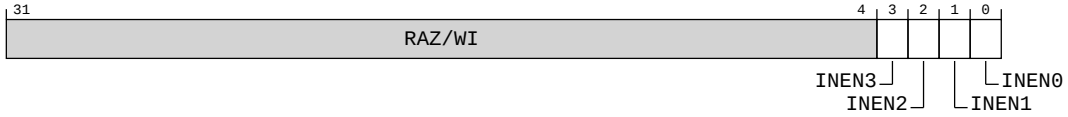


Table B-145: CTIINEN2 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x28	CTIINEN2

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.9 CTIINEN3, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x2C

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-133: ext_ctiinen3 bit assignments

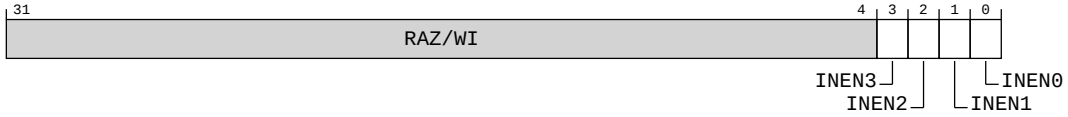


Table B-147: CTIINEN3 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x2C	CTIINEN3

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.10 CTIINEN4, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x30

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-134: ext_ctiinen4 bit assignments

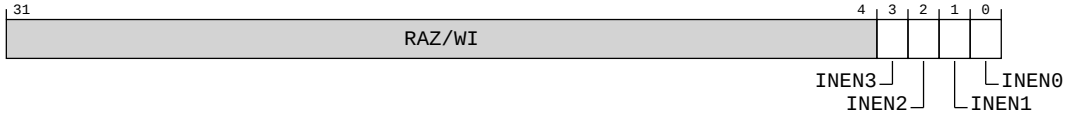


Table B-149: CTIINEN4 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x30	CTIINEN4

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.11 CTIINEN5, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x34

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-135: ext_ctiinen5 bit assignments

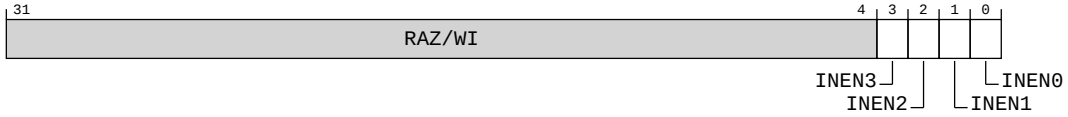


Table B-151: CTIINEN5 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x34	CTIINEN5

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.12 CTIINEN6, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x38

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-136: ext_ctiinen6 bit assignments

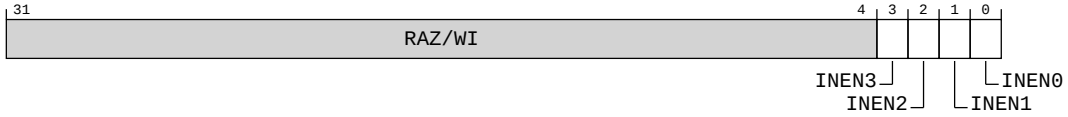


Table B-153: CTIINEN6 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x38	CTIINEN6

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.13 CTIINEN7, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x3C

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-137: ext_ctiinen7 bit assignments

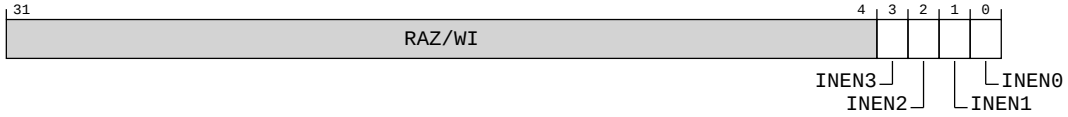


Table B-155: CTIINEN7 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x3C	CTIINEN7

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.14 CTIINEN8, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x40

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-138: ext_ctiinen8 bit assignments

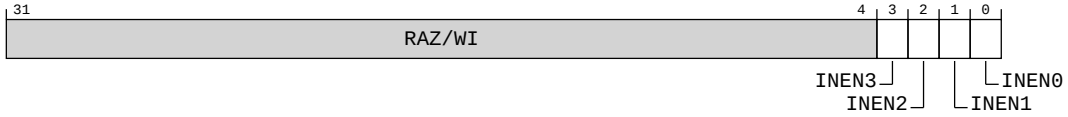


Table B-157: CTIINEN8 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x40	CTIINEN8

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.15 CTIINEN9, CTI Input Trigger to Output Channel Enable registers

Enables the signaling of an event on output channels when input trigger event n is received by the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x44

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-139: ext_ctiinen9 bit assignments

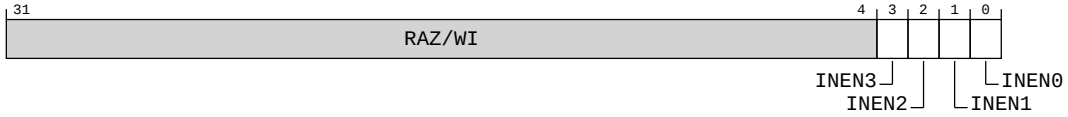


Table B-159: CTIINEN9 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	INEN3	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[1]	INEN1	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Possible values of this bit are: **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance
CTI	0x44	CTIINEN9

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.16 CTIOUTEN0, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xA0

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-140: ext_ctiouten0 bit assignments

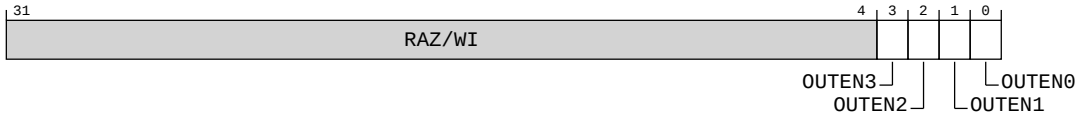


Table B-161: CTIOUTEN0 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xA0	CTIOUTEN0

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.17 CTIOUTEN1, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xA4

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-141: ext_ctiouten1 bit assignments

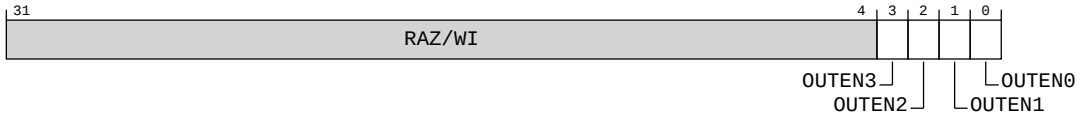


Table B-163: CTIOUTEN1 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xA4	CTIOUTEN1

This interface is accessible as follows:

When SoftwareLockStatus()
RO

When !SoftwareLockStatus()
RW

B.2.1.18 CTIOUTEN2, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width
32

Component
CTI

Register offset
0xA8

Access type
See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-142: ext_ctiouten2 bit assignments

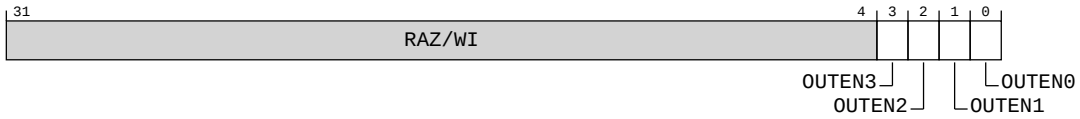


Table B-165: CTIOUTEN2 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xA8	CTIOUTEN2

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.19 CTIOUTEN3, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xAC

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-143: ext_ctiouten3 bit assignments

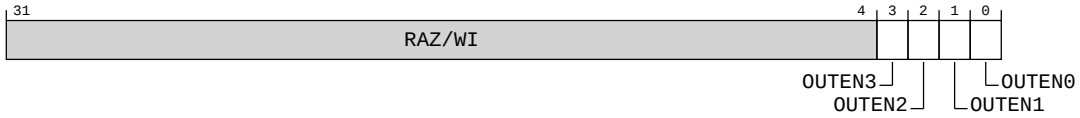


Table B-167: CTIOUTEN3 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xAC	CTIOUTEN3

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.20 CTIOUTEN4, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xB0

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-144: ext_ctiouten4 bit assignments

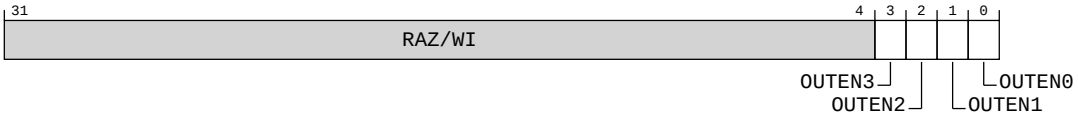


Table B-169: CTIOUTEN4 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xB0	CTIOUTEN4

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.21 CTIOUTEN5, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xB4

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-145: ext_ctiouten5 bit assignments

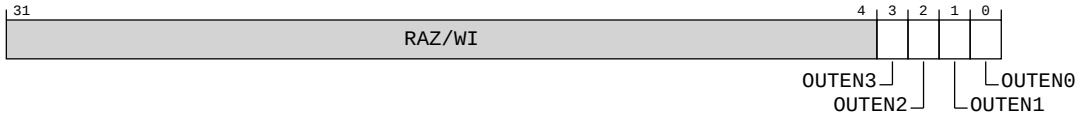


Table B-171: CTIOUTEN5 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xB4	CTIOUTEN5

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.22 CTIOUTEN6, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xB8

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-146: ext_ctiouten6 bit assignments

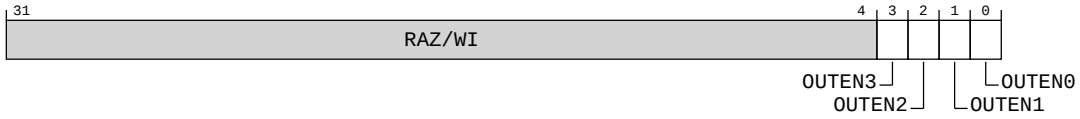


Table B-173: CTIOUTEN6 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xB8	CTIOUTEN6

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.23 CTIOUTEN7, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xBC

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-147: ext_ctiouten7 bit assignments

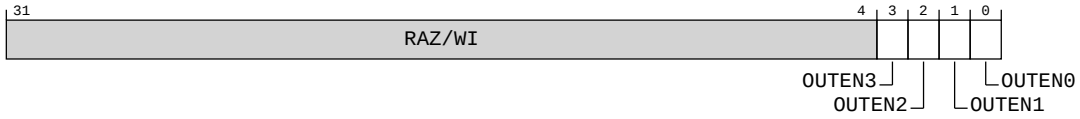


Table B-175: CTIOUTEN7 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xBC	CTIOUTEN7

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.24 CTIOUTEN8, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xC0

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-148: ext_ctiouten8 bit assignments

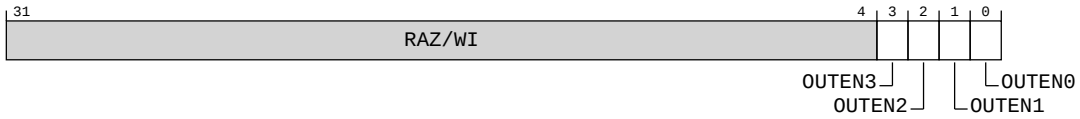


Table B-177: CTIOUTEN8 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xC0	CTIOUTEN8

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.25 CTIOUTEN9, CTI Input Channel to Output Trigger Enable registers

Defines which input channels generate output trigger n.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xC4

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-149: ext_ctiouten9 bit assignments

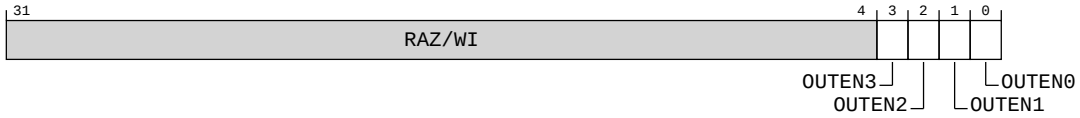


Table B-179: CTIOUTEN9 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance
CTI	0xC4	CTIOUTEN9

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.26 CTITRIGINSTATUS, CTI Trigger In Status register

Provides the status of the trigger inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x130

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-150: ext_ctitriginstatus bit assignments

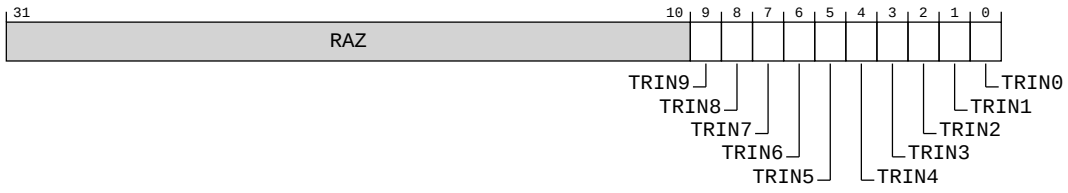


Table B-181: CTITRIGINSTATUS bit descriptions

Bits	Name	Description	Reset
[31:10]	**RAZ**	Reserved	**RAZ**
[9]	TRIN9	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[8]	TRIN8	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[7]	TRIN7	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[6]	TRIN6	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[5]	TRIN5	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[4]	TRIN4	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0

Bits	Name	Description	Reset
[3]	TRIN3	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[2]	TRIN2	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[1]	TRIN1	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0
[0]	TRIN0	Trigger input <n> status. Possible values of this bit are: **0b0** Input trigger n is inactive. **0b1** Input trigger n is active.	0b0

B.2.1.27 CTITRIGOUTSTATUS, CTI Trigger Out Status register

Provides the raw status of the trigger outputs after processing by trigger interface logic.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x134

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-151: ext_ctitrigoutstatus bit assignments

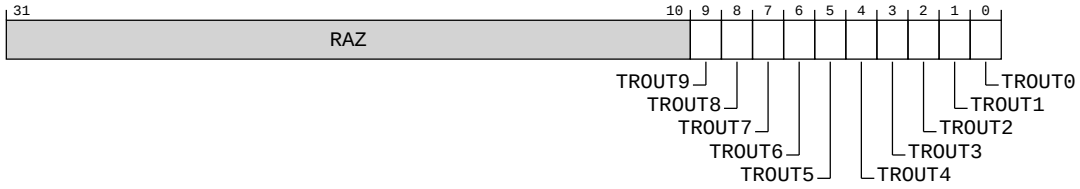


Table B-182: CTITRIGOUTSTATUS bit descriptions

Bits	Name	Description	Reset
[31:10]	**RAZ**	Reserved	**RAZ**
[9]	TROUT9	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[8]	TROUT8	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[7]	TROUT7	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0

Bits	Name	Description	Reset
[6]	TROUT6	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[5]	TROUT5	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[4]	TROUT4	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[3]	TROUT3	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[2]	TROUT2	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0
[1]	TROUT1	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0

Bits	Name	Description	Reset
[0]	TROUT0	Trigger output <n> status. **0b0** Output trigger n is inactive. **0b1** Output trigger n is active. Otherwise when n < N TROUT<n> is **RAZ**.	0b0

B.2.1.28 CTICHINSTATUS, CTI Channel In Status register

Provides the raw status of the ECT channel inputs to the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x138

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-152: ext_ctichinstatus bit assignments

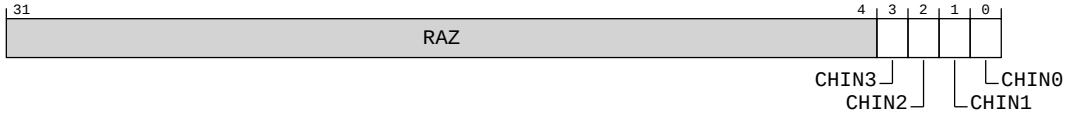


Table B-183: CTICHINSTATUS bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ**	Reserved	**RAZ**

Bits	Name	Description	Reset
[3]	CHIN3	Input channel <n> status. Possible values of this bit are: **0b0** Input channel <n> is inactive. **0b1** Input channel <n> is active.	0b0
[2]	CHIN2	Input channel <n> status. Possible values of this bit are: **0b0** Input channel <n> is inactive. **0b1** Input channel <n> is active.	0b0
[1]	CHIN1	Input channel <n> status. Possible values of this bit are: **0b0** Input channel <n> is inactive. **0b1** Input channel <n> is active.	0b0
[0]	CHIN0	Input channel <n> status. Possible values of this bit are: **0b0** Input channel <n> is inactive. **0b1** Input channel <n> is active.	0b0

B.2.1.29 CTICHOUTSTATUS, CTI Channel Out Status register

Provides the status of the ECT channel outputs from the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x13C

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-153: ext_ctichoutstatus bit assignments

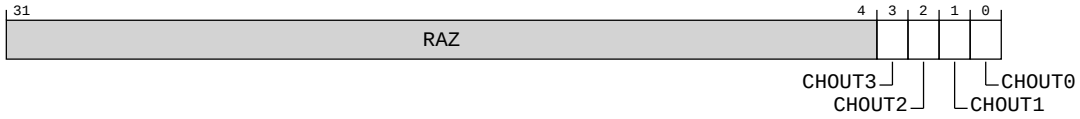


Table B-184: CTICHOUTSTATUS bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ**	Reserved	**RAZ**
[3]	CHOUT3	Output channel <n> status. Possible values of this bit are: **0b0** Output channel <n> is inactive. **0b1** Output channel <n> is active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	0b0
[2]	CHOUT2	Output channel <n> status. Possible values of this bit are: **0b0** Output channel <n> is inactive. **0b1** Output channel <n> is active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	0b0
[1]	CHOUT1	Output channel <n> status. Possible values of this bit are: **0b0** Output channel <n> is inactive. **0b1** Output channel <n> is active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	0b0

Bits	Name	Description	Reset
[0]	CHOUT0	Output channel <n> status. Possible values of this bit are: **0b0** Output channel <n> is inactive. **0b1** Output channel <n> is active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	0b0

B.2.1.30 CTIGATE, CTI Channel Gate Enable register

Determines whether events on channels propagate through the CTM to other ECT components, or from the CTM into the CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x140

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 1111

Bit descriptions

Figure B-154: ext_ctigate bit assignments

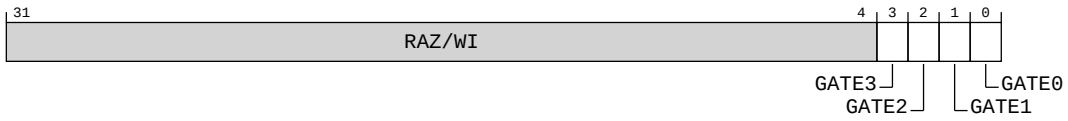


Table B-185: CTIGATE bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	GATE3	Channel <x> gate enable. Possible values of this bit are: **0b0** Disable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. **0b1** Enable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. If GATE[x] is set to 0, no new events will be propagated to the ECT and any existing output channel events will be terminated.	0b1
[2]	GATE2	Channel <x> gate enable. Possible values of this bit are: **0b0** Disable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. **0b1** Enable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. If GATE[x] is set to 0, no new events will be propagated to the ECT and any existing output channel events will be terminated.	0b1
[1]	GATE1	Channel <x> gate enable. Possible values of this bit are: **0b0** Disable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. **0b1** Enable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. If GATE[x] is set to 0, no new events will be propagated to the ECT and any existing output channel events will be terminated.	0b1
[0]	GATE0	Channel <x> gate enable. Possible values of this bit are: **0b0** Disable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. **0b1** Enable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation. If GATE[x] is set to 0, no new events will be propagated to the ECT and any existing output channel events will be terminated.	0b1

Accessibility

Component	Offset	Instance
CTI	0x140	CTIGATE

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.31 CTIDEVCTL, CTI Device Control register

Provides target-specific device controls

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x150

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-155: ext_ctidevctl bit assignments

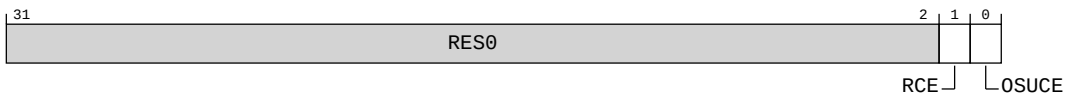


Table B-187: CTIDEVCTL bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	RCE	Reset Catch Enable. **0b0** Reset Catch debug event disabled. **0b1** Reset Catch debug event enabled.	0b0
[0]	OSUCE	OS Unlock Catch Enable **0b0** OS Unlock Catch debug event disabled. **0b1** OS Unlock Catch debug event enabled.	0b0

Accessibility

Component	Offset	Instance
CTI	0x150	CTIDEVCTL

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.32 CTICLAIMSET, CTI Claim Tag Set register

Used by software to set CLAIM bits to 1.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFA0

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-156: ext_cticlaimset bit assignments

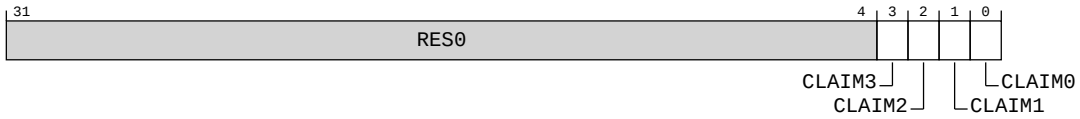


Table B-189: CTICLAIMSET bit descriptions

Bits	Name	Description	Reset
[31:4]	RES0	Reserved	RES0
[3]	CLAIM3	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x
[2]	CLAIM2	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x
[1]	CLAIM1	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x
[0]	CLAIM0	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x

Accessibility

Component	Offset	Instance
CTI	0xFA0	CTICLAIMSET

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.33 CTICLAIMCLR, CTI Claim Tag Clear register

Used by software to read the values of the CLAIM bits, and to clear these bits to 0.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFA4

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-157: ext_cticlaimclr bit assignments

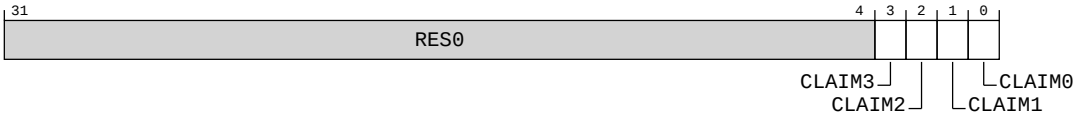


Table B-191: CTICLAIMCLR bit descriptions

Bits	Name	Description	Reset
[31:4]	RES0	Reserved	RES0
[3]	CLAIM3	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x
[2]	CLAIM2	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x
[1]	CLAIM1	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x
[0]	CLAIM0	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x

Accessibility

Component	Offset	Instance
CTI	0xFA4	CTICLAIMCLR

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.34 CTIDEVAFF0, CTI Device Affinity register 0

Copy of the low half of the PE AArch64-MPIDR_EL1 register that allows a debugger to determine which PE in a multiprocessor system the CTI component relates to.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFA8

Access type

RO

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-158: ext_ctidevaff0 bit assignments

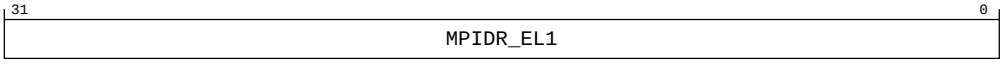


Table B-193: CTIDEVAFF0 bit descriptions

Bits	Name	Description	Reset
[31:0]	MPIDR_EL1	This field is a read-only copy of the low half of any of the cores' AArch64-MPIDR_EL1, as seen from the highest implemented Exception level, but with bits [15:8] set to `0x80`.	**Cluster** `32{x}` **Core** For core values, see your core documentation.

B.2.1.35 CTIDEVAFF1, CTI Device Affinity register 1

Copy of the high half of the PE AArch64-MPIDR_EL1 register that allows a debugger to determine which PE in a multiprocessor system the CTI component relates to.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFAC

Access type

RO

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-159: ext_ctidevaff1 bit assignments

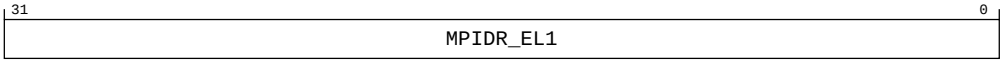


Table B-194: CTIDEVAFF1 bit descriptions

Bits	Name	Description	Reset
[31:0]	MPIDR_EL1	This field is a read-only copy of the high half of any of the cores' AArch64-MPIDR_EL1, as seen from the highest implemented Exception level.	**Cluster** 32{x} **Core** For core values see your core documentation.

B.2.1.36 CTIAUTHSTATUS, CTI Authentication Status register

Provides information about the state of the authentication interface for CTI. For more information on Debug authentication, see the *Arm® CoreSight™ Architecture Specification v3.0*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFB8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 11xx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-160: ext_ctiauthstatus bit assignments

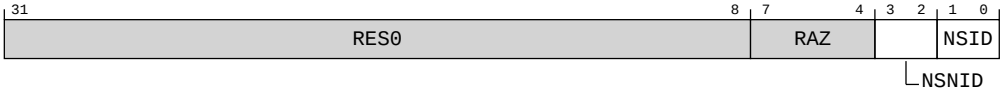


Table B-195: CTIAUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	**RAZ**	Reserved	**RAZ**
[3:2]	NSNID	Holds the same value as ext-DBGAUTHSTATUS_EL1.SNID. **0b11** Supported and enabled. DBGEN == TRUE.	0b11

Bits	Name	Description	Reset
[1:0]	NSID	Holds the same value as ext-DBGAUTHSTATUS_EL1.SID. **0b10** Secure invasive debug disabled. DBGEN == FALSE. **0b11** Secure invasive debug enabled. DBGEN == TRUE.	xx

B.2.1.37 CTIDEVARCH, CTI Device Architecture register

Identifies the programmers' model architecture of the CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFBC

Access type

RO

Reset value

0100 0111 0111 0001 0001 1010 0001 0100

Bit descriptions

Figure B-161: ext_ctidevarch bit assignments

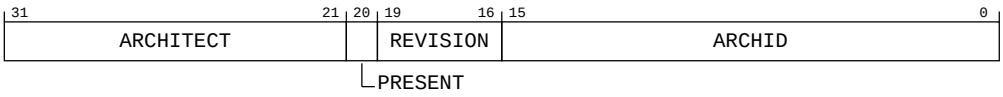


Table B-196: CTIDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **0b01000111011** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. **0b1** DEVARCH information present.	0b1

Bits	Name	Description	Reset
[19:16]	REVISION	Revision. Defines the architecture revision of the component. **0b0001** First revision, and also adds support for ext-CTIDEVCTL.	0b0001
[15:0]	ARCHID	Architecture ID. **0b0001101000010100** Cross Trigger Interface (CTI) architecture CTIv2.	0x1A14

B.2.1.38 CTIDEVID2, CTI Device ID register 2

Reserved for future information about the CTI component to the debugger.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFC0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-162: ext_ctidevid2 bit assignments

Table B-197: CTIDEVID2 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.1.39 CTIDEVID1, CTI Device ID register 1

Reserved for future information about the CTI component to the debugger.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFC4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-163: ext_ctidevid1 bit assignments

Table B-198: CTIDEVID1 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.1.40 CTIDEVID, CTI Device ID register 0

Describes the CTI component to the debugger.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFC8

Access type

RO

Reset value

xxxx xx01 xx00 0100 xx00 1010 xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-164: ext_ctidevid bit assignments

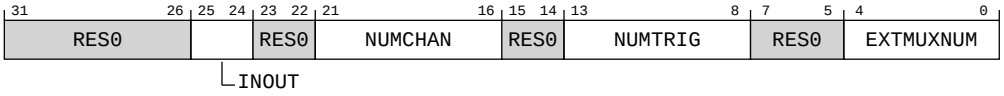


Table B-199: CTIDEVID bit descriptions

Bits	Name	Description	Reset
[31:26]	**RES0**	Reserved	**RES0**
[25:24]	INOUT	Input/output options. Indicates presence of the input gate. **0b00** ext-CTIGATE does not mask propagation of input events from external channels. **0b01** ext-CTIGATE masks propagation of input events from external channels. All other values are reserved.	0b01
[23:22]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[21:16]	NUMCHAN	Number of ECT channels implemented. **0b000100** 4 channels (0..3) implemented.	0b000100
[15:14]	**RES0**	Reserved	**RES0**
[13:8]	NUMTRIG	Number of triggers implemented. **0b001010** 10 triggers (0..9) implemented.	0b001010
[7:5]	**RES0**	Reserved	**RES0**
[4:0]	EXTMUXNUM	Number of multiplexors available on triggers. This value is used in conjunction with External Control register, ext-ASICCTL. **0b00000** No multiplexors implemented. ext-ASICCTL is unused.	5{x}

B.2.1.41 CTIDEVTYPE, CTI Device Type register

Indicates to a debugger that this component is part of a PEs cross-trigger interface.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFCC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0100

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-165: ext_ctidevtype bit assignments

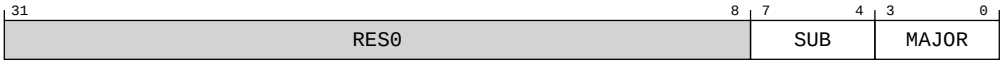


Table B-200: CTIDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SUB	Subtype. **0b0001** Embedded Cross-Trigger.	0b0001
[3:0]	MAJOR	Major type. **0b0100** Debug Control.	0b0100

B.2.1.42 CTIPIDR4, CTI Peripheral Identification Register 4

Provides information to identify a CTI component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFD0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-166: ext_ctipidr4 bit assignments

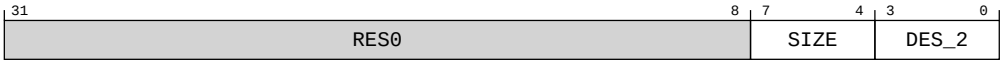


Table B-201: CTIPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SIZE	4KB count. **Cluster: 0b0000** The component uses a single 4KB block. **Core** For core values, see your core documentation.	**Cluster** 0b0000 **Core** For core values, see your core documentation.
[3:0]	DES_2	JEP106 continuation code. **Cluster: 0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B. **Core** For core values, see your core documentation.	**Cluster** 0b0100 **Core** For core values, see your core documentation.

B.2.1.43 CTIPIDR0, CTI Peripheral Identification Register 0

Provides information to identify a CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFE0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1110 1000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-167: ext_ctipidr0 bit assignments

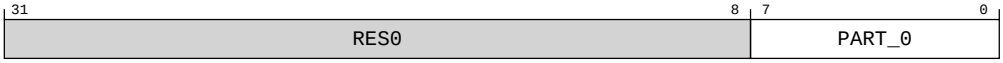


Table B-202: CTIPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **Cluster: 0b11101000** DSU-110 Cross Trigger Interface. Bits [7:0] of part number 0x4E8. **Core** For core values, see your core documentation.	**Cluster** 0xE8 **Core** For core values, see your core documentation.

B.2.1.44 CTIPIDR1, CTI Peripheral Identification Register 1

Provides information to identify a CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFE4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-168: ext_ctipidr1 bit assignments

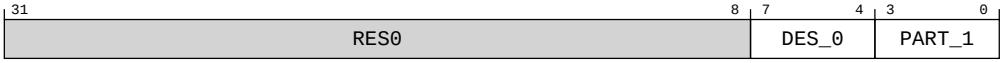


Table B-203: CTIPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **Cluster: `0b1011`** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B. **Core** For core values, see your core documentation.	**Cluster** `0b1011` **Core** For core values, see your core documentation.
[3:0]	PART_1	Part number bits [11:8]. **Cluster: `0b0100`** DSU-110 Cross Trigger Interface. Bits [11:8] of part number 0x4E8. **Core** For core values, see your core documentation.	**Cluster** `0b0100` **Core** For core values, see your core documentation.

B.2.1.45 CTIPIDR2, CTI Peripheral Identification Register 2

Provides information to identify a CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFE8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0110 1011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-169: ext_ctipidr2 bit assignments

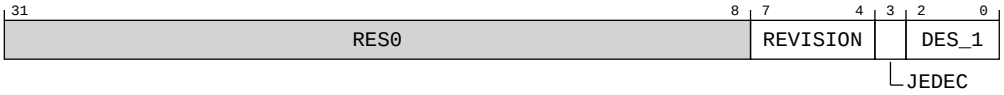


Table B-204: CTIPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	REVISION	Component major revision. **Cluster:0b0000** Component major revision 0. **Cluster: 0b0001** Component major revision 1. **Cluster: 0b0010** Component major revision 2. **Cluster: 0b0011** Component major revision 3. **Cluster: 0b0100** Component major revision 4. **Cluster: 0b0101** Component major revision 5. **Cluster: 0b0110** Component major revision 6. **Core** For core values, see your core documentation. For DSU-110: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r1p0. • Major revision 2 corresponds to r2p0. • Major revision 3 corresponds to r2p1. • Major revision 4 corresponds to r3p0. • Major revision 5 corresponds to r3p1. • Major revision 6 corresponds to r4p0.	**Cluster** 0b0110 **Core** For core values, see your core documentation.
[3]	JEDEC	JEDEC assignee. **Cluster: 0b1** JEDEC-assignee values is used. **Core** For core values, see your core documentation.	**Cluster** 0b1 **Core** For core values, see your core documentation.
[2:0]	DES_1	JEP106 identification code bits [6:4]. **Cluster: 0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B. **Core** For core values, see your core documentation.	**Cluster** 0b011 **Core** For core values, see your core documentation.

B.2.1.46 CTIPIDR3, CTI Peripheral Identification Register 3

Provides information to identify a CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFEC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-170: ext_ctipidr3 bit assignments

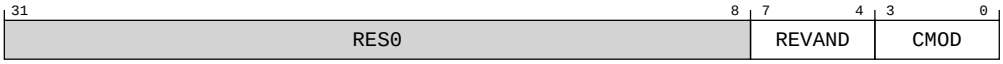


Table B-205: CTIPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	REVAND	Component minor revision. **Cluster: 0b0000** Component minor revision 0. **Cluster: 0b0001** Component minor revision 1. **Cluster: 0b0010** Component minor revision 2. **Cluster: 0b0011** Component minor revision 3. **Cluster: 0b0100** Component minor revision 4. **Core** For core values see your core documentation.	**Cluster** 0b0000 **Core** For core values, see your core documentation.
[3:0]	CMOD	Customer Modified. **Cluster: 0b0000** The component is not modified from the original design. **Core** For core values see your core documentation.	**Cluster** 0b0000 **Core** For core values, see your core documentation.

B.2.1.47 CTICIDR0, CTI Component Identification Register 0

Provides information to identify a CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFF0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-171: ext_cticidr0 bit assignments

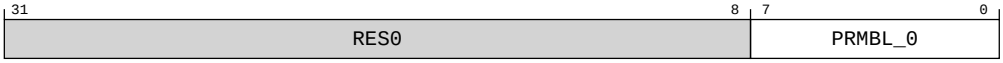


Table B-206: CTICIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	Preamble. **0b00001101** CoreSight component identification preamble.	0x0D

B.2.1.48 CTICIDR1, CTI Component Identification Register 1

Provides information to identify a CTI component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFF4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-172: ext_cticidr1 bit assignments

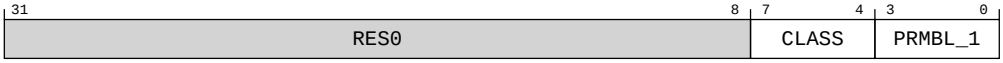


Table B-207: CTICIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	Component class. **0b1001** CoreSight debug component.	0b1001
[3:0]	PRMBL_1	Preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.2.1.49 CTICIDR2, CTI Component Identification Register 2

Provides information to identify a CTI component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFF8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-173: ext_cticidr2 bit assignments

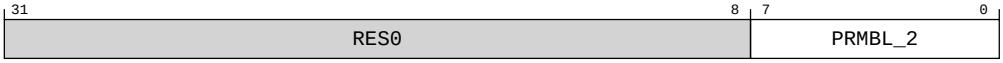


Table B-208: CTICIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	Preamble. **0b00000101** CoreSight component identification preamble.	0x05

B.2.1.50 CTICIDR3, CTI Component Identification Register 3

Provides information to identify a CTI component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFFC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-174: ext_cticidr3 bit assignments

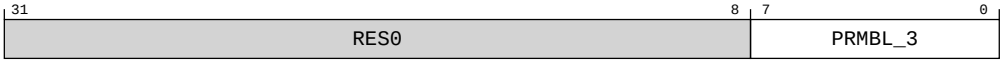


Table B-209: CTICIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Preamble. **0b10110001** CoreSight component identification preamble.	0xB1

B.2.2 External cluster ROM register summary

The summary table provides an overview of all *cluster ROM* (CLUSTERROM) table registers that are accessed externally (memory-mapped) over the debug APB bus. Individual register descriptions provide detailed information.

The summary table provides an overview of all the cluster ROM table registers. For more information about a register, click on the register name in the table.

Note

- The cluster ROM table registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-210: CLUSTERROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	CLUSTERROM_ROMENTRY0	—	32-bit	Cluster ROM table entry 0	Yes
0x004	CLUSTERROM_ROMENTRY1	—	32-bit	Cluster ROM table entry 1	Yes
0x008	CLUSTERROM_ROMENTRY2	—	32-bit	Cluster ROM table entry 2	Yes
0x00C	CLUSTERROM_ROMENTRY3	—	32-bit	Cluster ROM table entry 3	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x010	CLUSTERROM_ROMENTRY4	—	32-bit	Cluster ROM table entry 4	Yes
0x014	CLUSTERROM_ROMENTRY5	—	32-bit	Cluster ROM table entry 5	Yes
0x018	CLUSTERROM_ROMENTRY6	—	32-bit	Cluster ROM table entry 6	Yes
0x01C	CLUSTERROM_ROMENTRY7	—	32-bit	Cluster ROM table entry 7	Yes
0x020	CLUSTERROM_ROMENTRY8	—	32-bit	Cluster ROM table entry 8	Yes
0x024	CLUSTERROM_ROMENTRY9	—	32-bit	Cluster ROM table entry 9	Yes
0x028	CLUSTERROM_ROMENTRY10	—	32-bit	Cluster ROM table entry 10	Yes
0x02C	CLUSTERROM_ROMENTRY11	—	32-bit	Cluster ROM table entry 11	Yes
0x030	CLUSTERROM_ROMENTRY12	—	32-bit	Cluster ROM table entry 12	Yes
0x034	CLUSTERROM_ROMENTRY13	—	32-bit	Cluster ROM table entry 13	Yes
0xA00	CLUSTERROM_DBGPCR0	—	32-bit	Cluster ROM table Debug Power Control Register 0	Yes
0xA04	CLUSTERROM_DBGPCR1	—	32-bit	Cluster ROM table Debug Power Control Register 1	Yes
0xA08	CLUSTERROM_DBGPCR2	—	32-bit	Cluster ROM table Debug Power Control Register 2	Yes
0xA0C	CLUSTERROM_DBGPCR3	—	32-bit	Cluster ROM table Debug Power Control Register 3	Yes
0xA10	CLUSTERROM_DBGPCR4	—	32-bit	Cluster ROM table Debug Power Control Register 4	Yes
0xA14	CLUSTERROM_DBGPCR5	—	32-bit	Cluster ROM table Debug Power Control Register 5	Yes
0xA18	CLUSTERROM_DBGPCR6	—	32-bit	Cluster ROM table Debug Power Control Register 6	Yes
0xA1C	CLUSTERROM_DBGPCR7	—	32-bit	Cluster ROM table Debug Power Control Register 7	Yes
0xA20	CLUSTERROM_DBGPCR8	—	32-bit	Cluster ROM table Debug Power Control Register 8	Yes
0xA24	CLUSTERROM_DBGPCR9	—	32-bit	Cluster ROM table Debug Power Control Register 9	Yes
0xA28	CLUSTERROM_DBGPCR10	—	32-bit	Cluster ROM table Debug Power Control Register 10	Yes
0xA2C	CLUSTERROM_DBGPCR11	—	32-bit	Cluster ROM table Debug Power Control Register 11	Yes
0xA80	CLUSTERROM_DBGPSR0	—	32-bit	Cluster ROM table Debug Power Status Register 0	Yes
0xA84	CLUSTERROM_DBGPSR1	—	32-bit	Cluster ROM table Debug Power Status Register 1	Yes
0xA88	CLUSTERROM_DBGPSR2	—	32-bit	Cluster ROM table Debug Power Status Register 2	Yes
0xA8C	CLUSTERROM_DBGPSR3	—	32-bit	Cluster ROM table Debug Power Status Register 3	Yes
0xA90	CLUSTERROM_DBGPSR4	—	32-bit	Cluster ROM table Debug Power Status Register 4	Yes
0xA94	CLUSTERROM_DBGPSR5	—	32-bit	Cluster ROM table Debug Power Status Register 5	Yes
0xA98	CLUSTERROM_DBGPSR6	—	32-bit	Cluster ROM table Debug Power Status Register 6	Yes
0xA9C	CLUSTERROM_DBGPSR7	—	32-bit	Cluster ROM table Debug Power Status Register 7	Yes
0xAA0	CLUSTERROM_DBGPSR8	—	32-bit	Cluster ROM table Debug Power Status Register 8	Yes
0xAA4	CLUSTERROM_DBGPSR9	—	32-bit	Cluster ROM table Debug Power Status Register 9	Yes
0xAA8	CLUSTERROM_DBGPSR10	—	32-bit	Cluster ROM table Debug Power Status Register 10	Yes
0xAAC	CLUSTERROM_DBGPSR11	—	32-bit	Cluster ROM table Debug Power Status Register 11	Yes
0xC00	CLUSTERROM_PRIDR0	—	32-bit	Cluster ROM table Power Request ID Register 0	Yes
0xFB8	CLUSTERROM_AUTHSTATUS	—	32-bit	Cluster ROM table Authentication Status Register	Yes
0xFBC	CLUSTERROM_DEVARCH	—	32-bit	Cluster ROM table Device Architecture Register	Yes
0xFC8	CLUSTERROM_DEVID	—	32-bit	Cluster ROM table Device Configuration Register	Yes
0xFCC	CLUSTERROM_DEVTYPE	—	32-bit	Cluster ROM table Device Type Register	Yes
0xFD0	CLUSTERROM_PIDR4	—	32-bit	Cluster ROM table Peripheral Identification Register 4	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFE0	CLUSTERROM_PIDR0	—	32-bit	Cluster ROM table Peripheral Identification Register 0	Yes
0xFE4	CLUSTERROM_PIDR1	—	32-bit	Cluster ROM table Peripheral Identification Register 1	Yes
0xFE8	CLUSTERROM_PIDR2	—	32-bit	Cluster ROM table Peripheral Identification Register 2	Yes
0xFEC	CLUSTERROM_PIDR3	—	32-bit	Cluster ROM table Peripheral Identification Register 3	Yes
0xFF0	CLUSTERROM_CIDR0	—	32-bit	Cluster ROM table Component Identification Register 0	Yes
0xFF4	CLUSTERROM_CIDR1	—	32-bit	Cluster ROM table Component Identification Register 1	Yes
0xFF8	CLUSTERROM_CIDR2	—	32-bit	Cluster ROM table Component Identification Register 2	Yes
0xFFC	CLUSTERROM_CIDR3	—	32-bit	Cluster ROM table Component Identification Register 3	Yes

B.2.2.1 CLUSTERROM_ROMENTRY0, Cluster ROM table entry 0

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x000

Access type

RO

Reset value

0000 0000 0000 0001 0000 xxxx xxxx x011

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-175: ext_clusterrom_romentry0 bit assignments

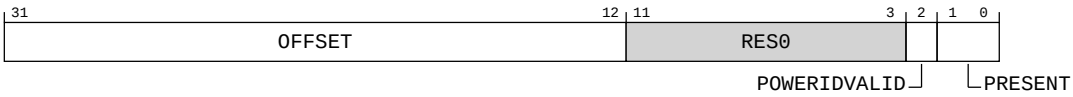


Table B-211: CLUSTERROM_ROMENTRY0 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000000010000** Cluster PMU table at address 0xD_0000 in the Cluster Debug APB address map.	0x00010
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM entry is present.	0b11

B.2.2.2 CLUSTERROM_ROMENTRY1, Cluster ROM table entry 1

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x004

Access type

RO

Reset value

0000 0000 0000 0010 0000 xxxx xxxx x0xx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-176: ext_clusterrom_romentry1 bit assignments

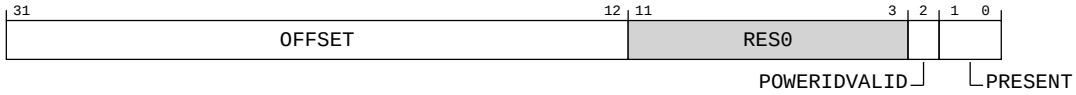


Table B-212: CLUSTERROM_ROMENTRY1 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000000100000** Cluster ELA at address 0xE_0000 in the Cluster Debug APB address map.	0x00020
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b10** ELA is not present so the ROM entry is not present. **0b11** ELA is present so the ROM entry is present.	xx

B.2.2.3 CLUSTERROM_ROMENTRY2, Cluster ROM table entry 2

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x008

Access type

RO

Reset value

0000 0000 0000 0100 0000 xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-177: ext_clusterrom_romentry2 bit assignments

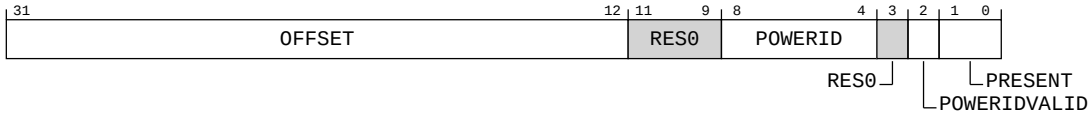


Table B-213: CLUSTERROM_ROMENTRY2 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000000001000000** Core 0 ROM table at address 0x10_0000 in the Cluster Debug APB address map.	0x00040
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

B.2.2.4 CLUSTERROM_ROMENTRY3, Cluster ROM table entry 3

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x00C

Access type

RO

Reset value

When NUM_CORES >= 2

0000 0000 0001 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 2

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 2

Figure B-178: ext_clusterrom_romentry3 bit assignments

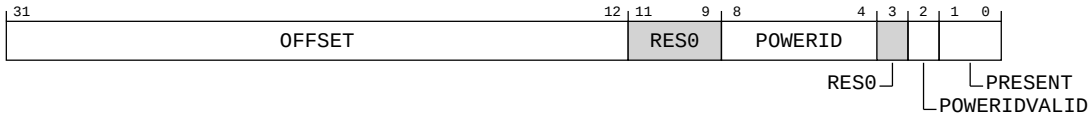


Table B-214: CLUSTERROM_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000000101000000** Core 1 ROM table at address 0x20_0000 in the Cluster Debug APB address map.	0x00140
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 2

Figure B-179: ext_clusterrom_romentry3 bit assignments

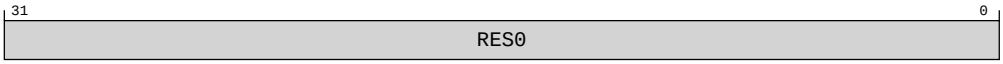


Table B-215: CLUSTERROM_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.5 CLUSTERROM_ROMENTRY4, Cluster ROM table entry 4

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x010

Access type

RO

Reset value

When NUM_CORES >= 3

0000 0000 0010 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 3

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 3

Figure B-180: ext_clusterrom_romentry4 bit assignments

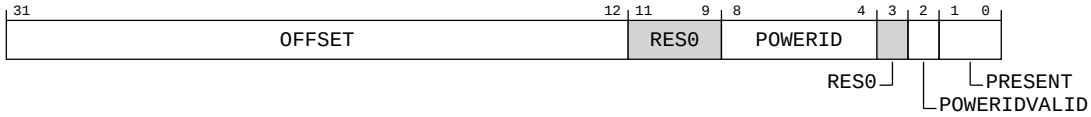


Table B-216: CLUSTERROM_ROMENTRY4 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000001001000000** Core 2 ROM table at address 0x30_0000 in the Cluster Debug APB address map.	0x00240
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 3

Figure B-181: ext_clusterrom_romentry4 bit assignments

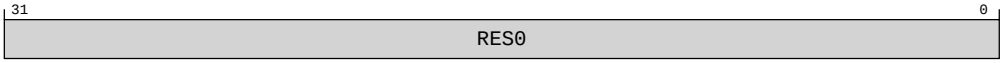


Table B-217: CLUSTERROM_ROMENTRY4 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.6 CLUSTERROM_ROMENTRY5, Cluster ROM table entry 5

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x014

Access type

RO

Reset value

When NUM_CORES >= 4

0000 0000 0011 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 4

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 4

Figure B-182: ext_clusterrom_romentry5 bit assignments

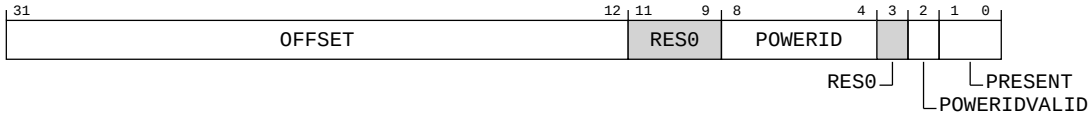


Table B-218: CLUSTERROM_ROMENTRY5 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000001101000000** Core 3 ROM table at address 0x40_0000 in the Cluster Debug APB address map.	0x00340
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 4

Figure B-183: ext_clusterrom_romentry5 bit assignments

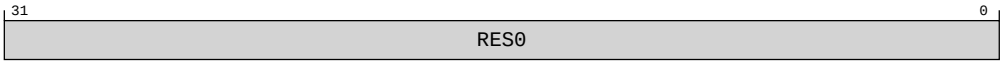


Table B-219: CLUSTERROM_ROMENTRY5 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.7 CLUSTERROM_ROMENTRY6, Cluster ROM table entry 6

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x018

Access type

RO

Reset value

When NUM_CORES >= 5

0000 0000 0100 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 5

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 5

Figure B-184: ext_clusterrom_romentry6 bit assignments

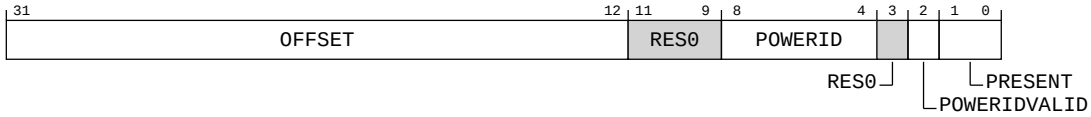


Table B-220: CLUSTERROM_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000010001000000** Core 4 ROM table at address 0x50_0000 in the Cluster Debug APB address map.	0x00440
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 5

Figure B-185: ext_clusterrom_romentry6 bit assignments

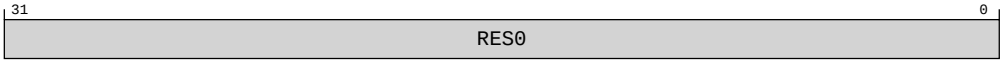


Table B-221: CLUSTERROM_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.8 CLUSTERROM_ROMENTRY7, Cluster ROM table entry 7

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x01C

Access type

RO

Reset value

When NUM_CORES >= 6

0000 0000 0101 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 6

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 6

Figure B-186: ext_clusterrom_romentry7 bit assignments

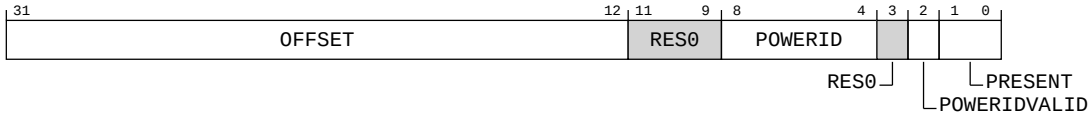


Table B-222: CLUSTERROM_ROMENTRY7 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000010101000000** Core 5 ROM table at address 0x60_0000 in the Cluster Debug APB address map.	0x00540
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 6

Figure B-187: ext_clusterrom_romentry7 bit assignments

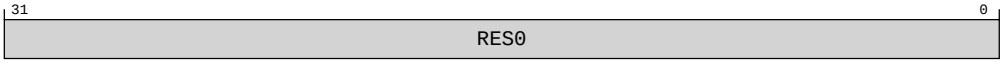


Table B-223: CLUSTERROM_ROMENTRY7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.9 CLUSTERROM_ROMENTRY8, Cluster ROM table entry 8

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x020

Access type

RO

Reset value

When NUM_CORES >= 7

0000 0000 0110 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 7

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 7

Figure B-188: ext_clusterrom_romentry8 bit assignments

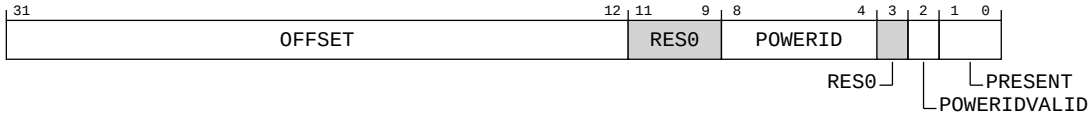


Table B-224: CLUSTERROM_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000011001000000** Core 6 ROM table at address 0x70_0000 in the Cluster Debug APB address map.	0x00640
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 7

Figure B-189: ext_clusterrom_romentry8 bit assignments

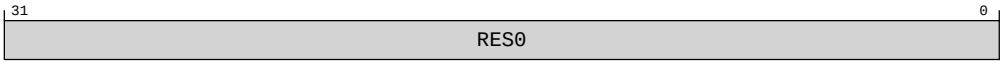


Table B-225: CLUSTERROM_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.10 CLUSTERROM_ROMENTRY9, Cluster ROM table entry 9

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x024

Access type

RO

Reset value

When NUM_CORES >= 8

0000 0000 0111 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 8

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 8

Figure B-190: ext_clusterrom_romentry9 bit assignments

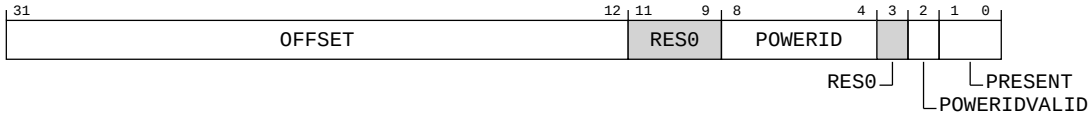


Table B-226: CLUSTERROM_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000011101000000** Core 7 ROM table at address 0x80_0000 in the Cluster Debug APB address map.	0x00740
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 8

Figure B-191: ext_clusterrom_romentry9 bit assignments

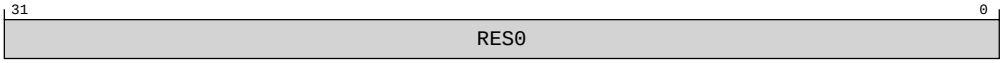


Table B-227: CLUSTERROM_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.11 CLUSTERROM_ROMENTRY10, Cluster ROM table entry 10

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x028

Access type

RO

Reset value

When NUM_CORES >= 9

0000 0000 1000 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 9

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 9

Figure B-192: ext_clusterrom_romentry10 bit assignments

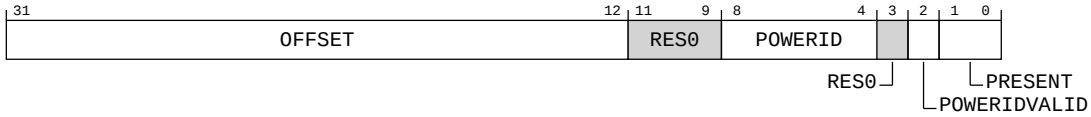


Table B-228: CLUSTERROM_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000100001000000** Core 8 ROM table at address 0x90_0000 in the Cluster Debug APB address map.	0x00840
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 9

Figure B-193: ext_clusterrom_romentry10 bit assignments

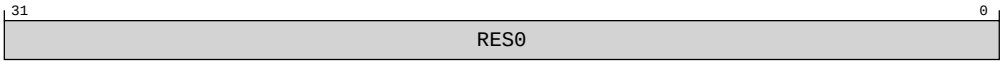


Table B-229: CLUSTERROM_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.12 CLUSTERROM_ROMENTRY11, Cluster ROM table entry 11

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x02C

Access type

RO

Reset value

When NUM_CORES >= 10

0000 0000 1001 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 10

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 10

Figure B-194: ext_clusterrom_romentry11 bit assignments

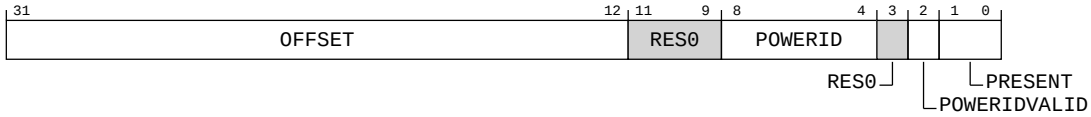


Table B-230: CLUSTERROM_ROMENTRY11 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000100101000000** Core 9 ROM table at address 0xA0_0000 in the Cluster Debug APB address map.	0x00940
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 10

Figure B-195: ext_clusterrom_romentry11 bit assignments

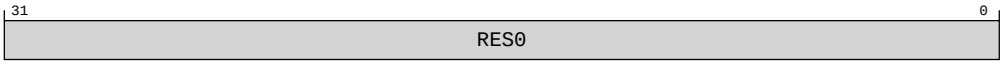


Table B-231: CLUSTERROM_ROMENTRY11 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.13 CLUSTERROM_ROMENTRY12, Cluster ROM table entry 12

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x030

Access type

RO

Reset value

When NUM_CORES >= 11

0000 0000 1010 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 11

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 11

Figure B-196: ext_clusterrom_romentry12 bit assignments

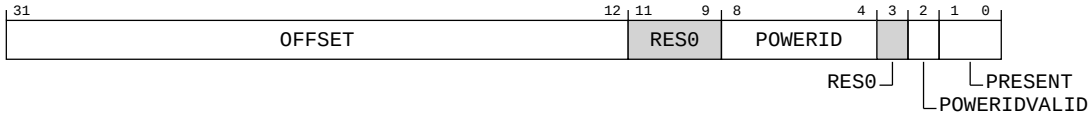


Table B-232: CLUSTERROM_ROMENTRY12 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000101001000000** Core 10 ROM table at address 0xB0_0000 in the Cluster Debug APB address map.	0x00A40
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 11

Figure B-197: ext_clusterrom_romentry12 bit assignments

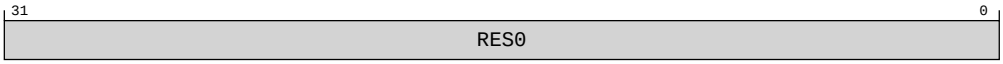


Table B-233: CLUSTERROM_ROMENTRY12 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.14 CLUSTERROM_ROMENTRY13, Cluster ROM table entry 13

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x034

Access type

RO

Reset value

When NUM_CORES >= 12

0000 0000 1011 0100 0000 xxxx xxxx xxxx

When NUM_CORES < 12

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 12

Figure B-198: ext_clusterrom_romentry13 bit assignments

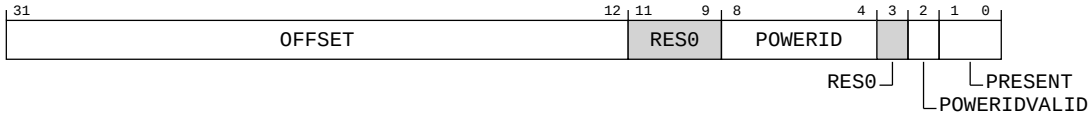


Table B-234: CLUSTERROM_ROMENTRY13 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000101101000000** Core 11 ROM table at address 0xC0_0000 in the Cluster Debug APB address map.	0x00B40
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **0b00000** PDCOMPLEX0 power domain. **0b00001** PDCOMPLEX1 power domain. **0b00010** PDCOMPLEX2 power domain. **0b00011** PDCOMPLEX3 power domain. **0b00100** PDCOMPLEX4 power domain. **0b00101** PDCOMPLEX5 power domain. **0b00110** PDCOMPLEX6 power domain. **0b00111** PDCOMPLEX7 power domain. **0b01000** PDCOMPLEX8 power domain. **0b01001** PDCOMPLEX9 power domain. **0b01010** PDCOMPLEX10 power domain. **0b01011** PDCOMPLEX11 power domain.	`5{x}`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **0b0** A power domain ID is not provided. **0b1** The POWERID field provides a power domain ID.	`x`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	xx

When NUM_CORES < 12

Figure B-199: ext_clusterrom_romentry13 bit assignments

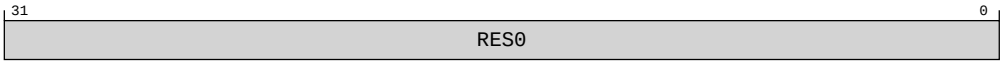


Table B-235: CLUSTERROM_ROMENTRY13 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.15 CLUSTERROM_DBGPCR0, Cluster ROM table Debug Power Control Register 0

Controls power requests for PDCOMPLEX0.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA00

Access type

RO

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-200: ext_clusterrom_dbgpcr0 bit assignments

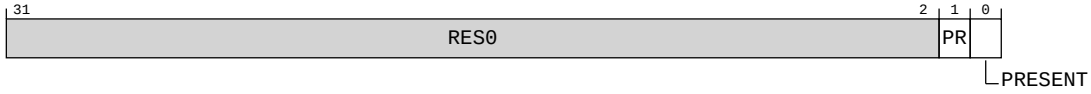


Table B-236: CLUSTERROM_DBGPCR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX0. **0b1** Power is requested for PDCOMPLEX0.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX0 is implemented.	x

B.2.2.16 CLUSTERROM_DBGPCR1, Cluster ROM table Debug Power Control Register 1

Controls power requests for PDCOMPLEX1.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA04

Access type

RO

Reset value

When NUM_CORES >= 2

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

When NUM_CORES < 2

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 2

Figure B-201: ext_clusterrom_dbgpcr1 bit assignments

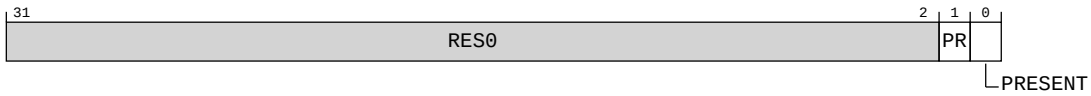


Table B-237: CLUSTERROM_DBGPCR1 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX1. **0b1** Power is requested for PDCOMPLEX1.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX1 is implemented.	x

When NUM_CORES < 2

Figure B-202: ext_clusterrom_dbgpcr1 bit assignments

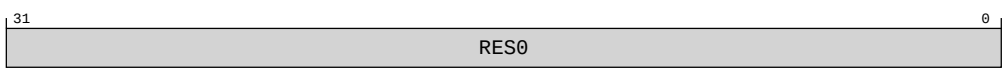


Table B-238: CLUSTERROM_DBGPCR1 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.17 CLUSTERROM_DBGPCR2, Cluster ROM table Debug Power Control Register 2

Controls power requests for PDCOMPLEX2.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA08

Access type

RO

Reset value

When NUM_CORES >= 3

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 3

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 3

Figure B-203: ext_clusterrom_dbgpcr2 bit assignments

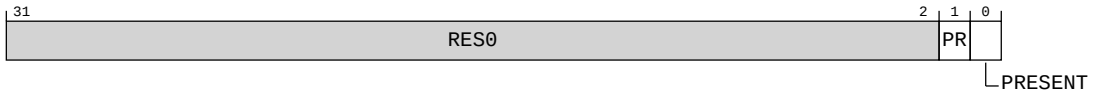


Table B-239: CLUSTERROM_DBGPCR2 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX2. **0b1** Power is requested for PDCOMPLEX2.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX2 is implemented.	x

When NUM_CORES < 3

Figure B-204: ext_clusterrom_dbgpcr2 bit assignments

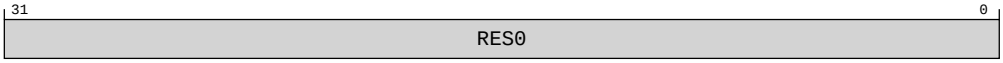


Table B-240: CLUSTERROM_DBGPCR2 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.18 CLUSTERROM_DBGPCR3, Cluster ROM table Debug Power Control Register 3

Controls power requests for PDCOMPLEX3.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA0C

Access type

RO

Reset value

When NUM_CORES >= 4

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

When NUM_CORES < 4

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 4

Figure B-205: ext_clusterrom_dbgpcr3 bit assignments

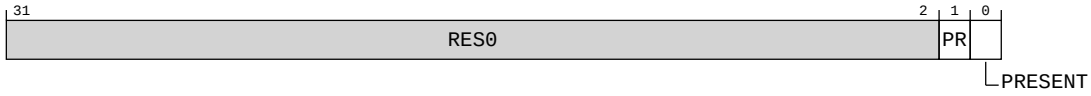


Table B-241: CLUSTERROM_DBGPCR3 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX3. **0b1** Power is requested for PDCOMPLEX3.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX3 is implemented.	x

When NUM_CORES < 4

Figure B-206: ext_clusterrom_dbgpcr3 bit assignments

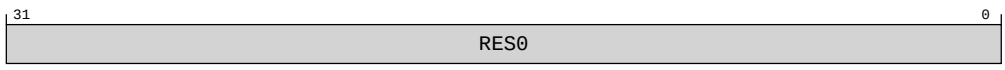


Table B-242: CLUSTERROM_DBGPCR3 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.19 CLUSTERROM_DBGPCR4, Cluster ROM table Debug Power Control Register 4

Controls power requests for PDCOMPLEX4.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA10

Access type

RO

Reset value

When NUM_CORES >= 5

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 5

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 5

Figure B-207: ext_clusterrom_dbgpcr4 bit assignments

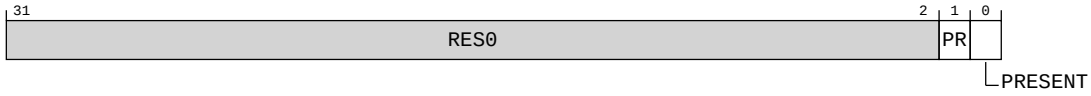


Table B-243: CLUSTERROM_DBGPCR4 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX4. **0b1** Power is requested for PDCOMPLEX4.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX4 is implemented.	x

When NUM_CORES < 5

Figure B-208: ext_clusterrom_dbgpcr4 bit assignments

Table B-244: CLUSTERROM_DBGPCR4 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.20 CLUSTERROM_DBGPCR5, Cluster ROM table Debug Power Control Register 5

Controls power requests for PDCOMPLEX5.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA14

Access type

RO

Reset value

When NUM_CORES >= 6

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

When NUM_CORES < 6

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 6

Figure B-209: ext_clusterrom_dbgpcr5 bit assignments

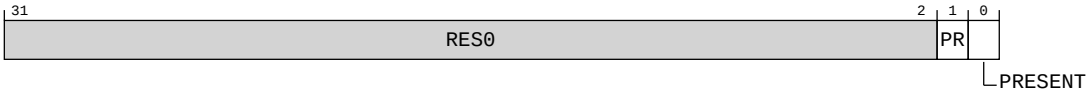


Table B-245: CLUSTERROM_DBGPCR5 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX5. **0b1** Power is requested for PDCOMPLEX5.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX5 is implemented.	x

When NUM_CORES < 6

Figure B-210: ext_clusterrom_dbgpcr5 bit assignments

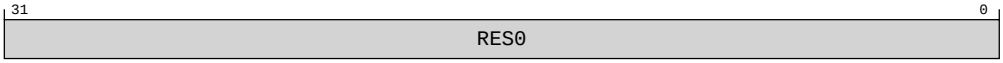


Table B-246: CLUSTERROM_DBGPCR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.21 CLUSTERROM_DBGPCR6, Cluster ROM table Debug Power Control Register 6

Controls power requests for PDCOMPLEX6.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA18

Access type

RO

Reset value

When NUM_CORES >= 7

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 7

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 7

Figure B-211: ext_clusterrom_dbgpcr6 bit assignments

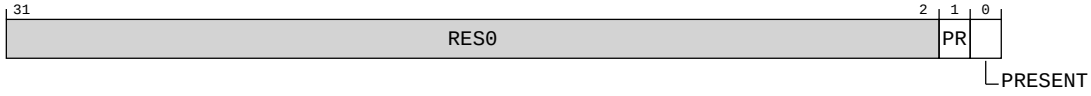


Table B-247: CLUSTERROM_DBGPCR6 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX6. **0b1** Power is requested for PDCOMPLEX6.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX6 is implemented.	x

When NUM_CORES < 7

Figure B-212: ext_clusterrom_dbgpcr6 bit assignments

Table B-248: CLUSTERROM_DBGPCR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.22 CLUSTERROM_DBGPCR7, Cluster ROM table Debug Power Control Register 7

Controls power requests for PDCOMPLEX7.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA1C

Access type

RO

Reset value

When NUM_CORES >= 8

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

When NUM_CORES < 8

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 8

Figure B-213: ext_clusterrom_dbgpcr7 bit assignments

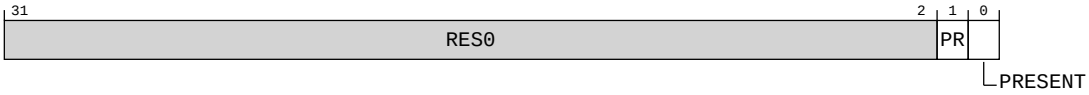


Table B-249: CLUSTERROM_DBGPCR7 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX7. **0b1** Power is requested for PDCOMPLEX7.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX7 is implemented.	x

When NUM_CORES < 8

Figure B-214: ext_clusterrom_dbgpcr7 bit assignments

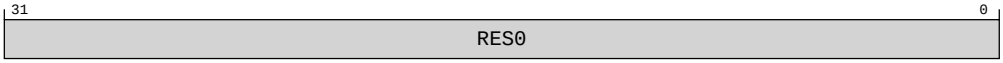


Table B-250: CLUSTERROM_DBGPCR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.23 CLUSTERROM_DBGPCR8, Cluster ROM table Debug Power Control Register 8

Controls power requests for PDCOMPLEX8.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA20

Access type

RO

Reset value

When NUM_CORES >= 9

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 9

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 9

Figure B-215: ext_clusterrom_dbgpcr8 bit assignments

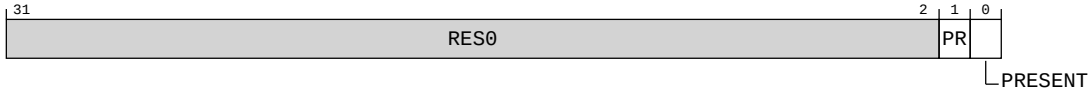


Table B-251: CLUSTERROM_DBGPCR8 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX8. **0b1** Power is requested for PDCOMPLEX8.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX8 is implemented.	x

When NUM_CORES < 9

Figure B-216: ext_clusterrom_dbgpcr8 bit assignments

Table B-252: CLUSTERROM_DBGPCR8 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.24 CLUSTERROM_DBGPCR9, Cluster ROM table Debug Power Control Register 9

Controls power requests for PDCOMPLEX9.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA24

Access type

RO

Reset value

When NUM_CORES >= 10

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 10

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 10

Figure B-217: ext_clusterrom_dbgpcr9 bit assignments

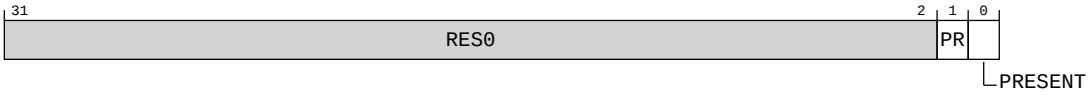


Table B-253: CLUSTERROM_DBGPCR9 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX9. **0b1** Power is requested for PDCOMPLEX9.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX9 is implemented.	x

When NUM_CORES < 10

Figure B-218: ext_clusterrom_dbgpcr9 bit assignments

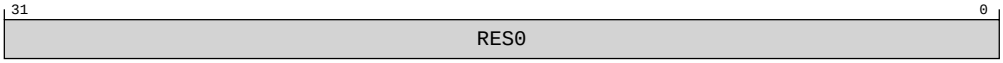


Table B-254: CLUSTERROM_DBGPCR9 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.25 CLUSTERROM_DBGPCR10, Cluster ROM table Debug Power Control Register 10

Controls power requests for PDCOMPLEX10.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA28

Access type

RO

Reset value

When NUM_CORES >= 11

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 11

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 11

Figure B-219: ext_clusterrom_dbgpcr10 bit assignments

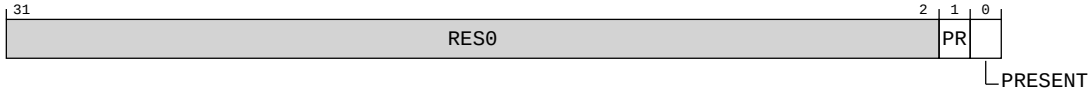


Table B-255: CLUSTERROM_DBGPCR10 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX10. **0b1** Power is requested for PDCOMPLEX10.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX10 is implemented.	x

When NUM_CORES < 11

Figure B-220: ext_clusterrom_dbgpcr10 bit assignments

Table B-256: CLUSTERROM_DBGPCR10 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.26 CLUSTERROM_DBGPCR11, Cluster ROM table Debug Power Control Register 11

Controls power requests for PDCOMPLEX11.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA2C

Access type

RO

Reset value

When NUM_CORES >= 12

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

When NUM_CORES < 12

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 12

Figure B-221: ext_clusterrom_dbgpcr11 bit assignments

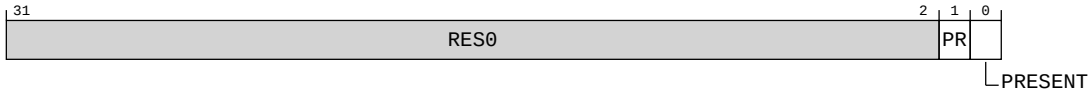


Table B-257: CLUSTERROM_DBGPCR11 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCOMPLEX11. **0b1** Power is requested for PDCOMPLEX11.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCOMPLEX11 is implemented.	x

When NUM_CORES < 12

Figure B-222: ext_clusterrom_dbgpcr11 bit assignments

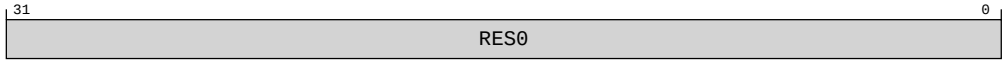


Table B-258: CLUSTERROM_DBGPCR11 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.27 CLUSTERROM_DBGPSR0, Cluster ROM table Debug Power Status Register 0

Indicates the power status for PDCOMPLEX0.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA80

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-223: ext_clusterrom_dbgpsr0 bit assignments

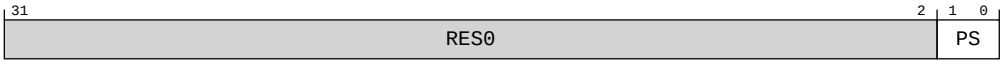


Table B-259: CLUSTERROM_DBGPSR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** PDCOMPLEX0 debug power domain might not be powered. **0b01** PDCOMPLEX0 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX0 debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

B.2.2.28 CLUSTERROM_DBGPSR1, Cluster ROM table Debug Power Status Register 1

Indicates the power status for PDCOMPLEX1.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA84

Access type

RO

Reset value

When NUM_CORES >= 2

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 2

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 2

Figure B-224: ext_clusterrom_dbgpsr1 bit assignments

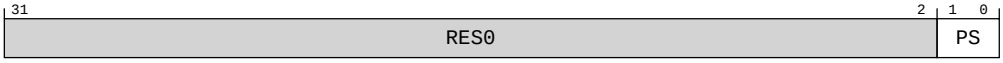


Table B-260: CLUSTERROM_DBGPSR1 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** PDCOMPLEX1 debug power domain might not be powered. **0b01** PDCOMPLEX1 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX1 debug power domain is powered and must remain powered until DBGPCR1.PR is set to 0.	0b00

When NUM_CORES < 2

Figure B-225: ext_clusterrom_dbgpsr1 bit assignments

Table B-261: CLUSTERROM_DBGPSR1 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.29 CLUSTERROM_DBGPSR2, Cluster ROM table Debug Power Status Register 2

Indicates the power status for PDCOMPLEX2.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA88

Access type

RO

Reset value

When NUM_CORES >= 3

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 3

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 3

Figure B-226: ext_clusterrom_dbgpsr2 bit assignments

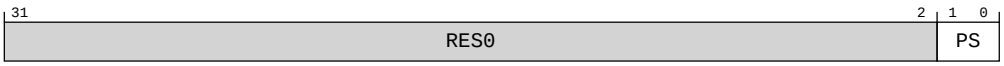


Table B-262: CLUSTERROM_DBGPSR2 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** PDCOMPLEX2 debug power domain might not be powered. **0b01** PDCOMPLEX2 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX2 debug power domain is powered and must remain powered until DBGPCR2.PR is set to 0.	0b00

When NUM_CORES < 3

Figure B-227: ext_clusterrom_dbgpsr2 bit assignments

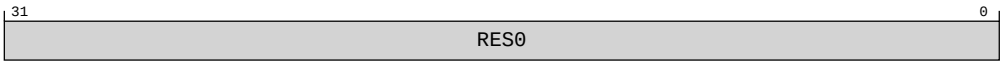


Table B-263: CLUSTERROM_DBGPSR2 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.30 CLUSTERROM_DBGPSR3, Cluster ROM table Debug Power Status Register 3

Indicates the power status for PDCOMPLEX3.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA8C

Access type

RO

Reset value

When NUM_CORES >= 4

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 4

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 4

Figure B-228: ext_clusterrom_dbgpsr3 bit assignments

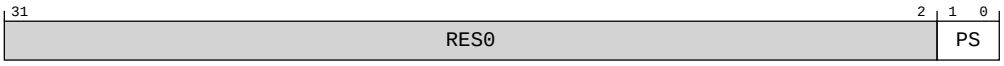


Table B-264: CLUSTERROM_DBGPSR3 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** PDCOMPLEX3 debug power domain might not be powered. **0b01** PDCOMPLEX3 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX3 debug power domain is powered and must remain powered until DBGPCR3.PR is set to 0.	0b00

When NUM_CORES < 4

Figure B-229: ext_clusterrom_dbgpsr3 bit assignments

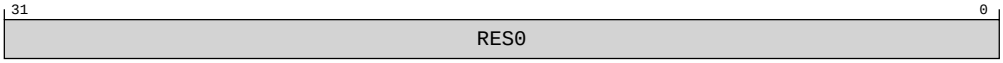


Table B-265: CLUSTERROM_DBGPSR3 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.31 CLUSTERROM_DBGPSR4, Cluster ROM table Debug Power Status Register 4

Indicates the power status for PDCOMPLEX4.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA90

Access type

RO

Reset value

When NUM_CORES >= 5

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 5

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 5

Figure B-230: ext_clusterrom_dbgpsr4 bit assignments

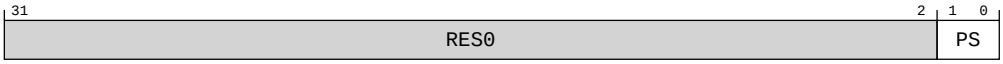


Table B-266: CLUSTERROM_DBGPSR4 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status. **0b00** PDCOMPLEX4 debug power domain might not be powered. **0b01** PDCOMPLEX4 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX4 debug power domain is powered and must remain powered until DBGPCR4.PR is set to 0.	0b00

When NUM_CORES < 5

Figure B-231: ext_clusterrom_dbgpsr4 bit assignments

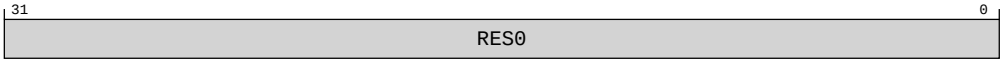


Table B-267: CLUSTERROM_DBGPSR4 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.32 CLUSTERROM_DBGPSR5, Cluster ROM table Debug Power Status Register 5

Indicates the power status for PDCOMPLEX5.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA94

Access type

RO

Reset value

When NUM_CORES >= 6

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 6

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 6

Figure B-232: ext_clusterrom_dbgpsr5 bit assignments

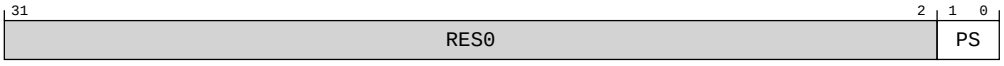


Table B-268: CLUSTERROM_DBGPSR5 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** PDCOMPLEX5 debug power domain might not be powered. **0b01** PDCOMPLEX5 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX5 debug power domain is powered and must remain powered until DBGPCR5.PR is set to 0.	0b00

When NUM_CORES < 6

Figure B-233: ext_clusterrom_dbgpsr5 bit assignments

Table B-269: CLUSTERROM_DBGPSR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.33 CLUSTERROM_DBGPSR6, Cluster ROM table Debug Power Status Register 6

Indicates the power status for PDCOMPLEX6.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA98

Access type

RO

Reset value

When NUM_CORES >= 7

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 7

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 7

Figure B-234: ext_clusterrom_dbgpsr6 bit assignments

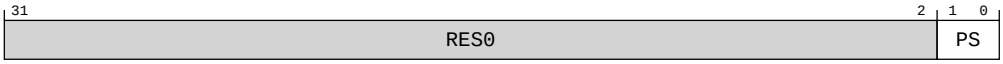


Table B-270: CLUSTERROM_DBGPSR6 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status. **0b00** PDCOMPLEX6 debug power domain might not be powered. **0b01** PDCOMPLEX6 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX6 debug power domain is powered and must remain powered until DBGPCR6.PR is set to 0.	0b00

When NUM_CORES < 7

Figure B-235: ext_clusterrom_dbgpsr6 bit assignments

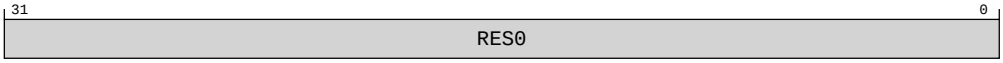


Table B-271: CLUSTERROM_DBGPSR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.2.2.34 CLUSTERROM_DBGPSR7, Cluster ROM table Debug Power Status Register 7

Indicates the power status for PDCOMPLEX7.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA9C

Access type

RO

Reset value

When NUM_CORES >= 8

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 8

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 8

Figure B-236: ext_clusterrom_dbgpsr7 bit assignments

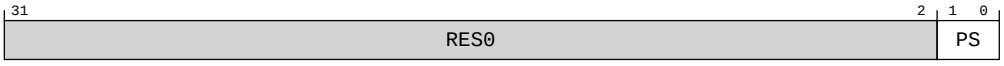


Table B-272: CLUSTERROM_DBGPSR7 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** PDCOMPLEX7 debug power domain might not be powered. **0b01** PDCOMPLEX7 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX7 debug power domain is powered and must remain powered until DBGPCR7.PR is set to 0.	0b00

When NUM_CORES < 8

Figure B-237: ext_clusterrom_dbgpsr7 bit assignments

Table B-273: CLUSTERROM_DBGPSR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.35 CLUSTERROM_DBGPSR8, Cluster ROM table Debug Power Status Register 8

Indicates the power status for PDCOMPLEX8.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAA0

Access type

RO

Reset value

When NUM_CORES >= 9

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 9

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 9

Figure B-238: ext_clusterrom_dbgpsr8 bit assignments

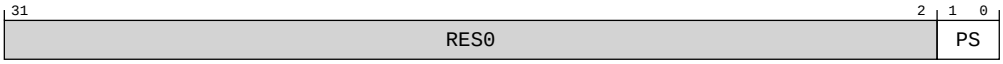


Table B-274: CLUSTERROM_DBGPSR8 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** PDCOMPLEX8 debug power domain might not be powered. **0b01** PDCOMPLEX8 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX8 debug power domain is powered and must remain powered until DBGPCR8.PR is set to 0.	0b00

When NUM_CORES < 9

Figure B-239: ext_clusterrom_dbgpsr8 bit assignments

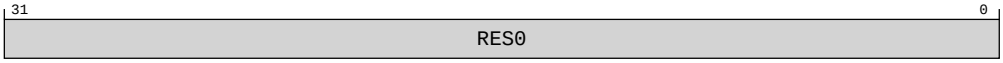


Table B-275: CLUSTERROM_DBGPSR8 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.36 CLUSTERROM_DBGPSR9, Cluster ROM table Debug Power Status Register 9

Indicates the power status for PDCOMPLEX9.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAA4

Access type

RO

Reset value

When NUM_CORES >= 10

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 10

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 10

Figure B-240: ext_clusterrom_dbgpsr9 bit assignments

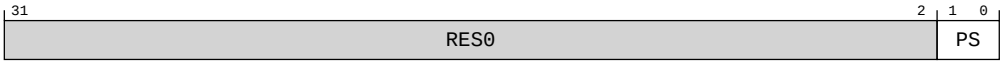


Table B-276: CLUSTERROM_DBGPSR9 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** PDCOMPLEX9 debug power domain might not be powered. **0b01** PDCOMPLEX9 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX9 debug power domain is powered and must remain powered until DBGPCR9.PR is set to 0.	0b00

When NUM_CORES < 10

Figure B-241: ext_clusterrom_dbgpsr9 bit assignments

Table B-277: CLUSTERROM_DBGPSR9 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.37 CLUSTERROM_DBGPSR10, Cluster ROM table Debug Power Status Register 10

Indicates the power status for PDCOMPLEX10.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAA8

Access type

RO

Reset value

When NUM_CORES >= 11

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 11

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 11

Figure B-242: ext_clusterrom_dbgpsr10 bit assignments

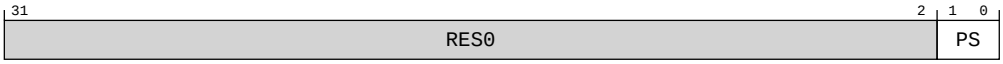


Table B-278: CLUSTERROM_DBGPSR10 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** PDCOMPLEX10 debug power domain might not be powered. **0b01** PDCOMPLEX10 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX10 debug power domain is powered and must remain powered until DBGPCR10.PR is set to 0.	0b00

When NUM_CORES < 11

Figure B-243: ext_clusterrom_dbgpsr10 bit assignments

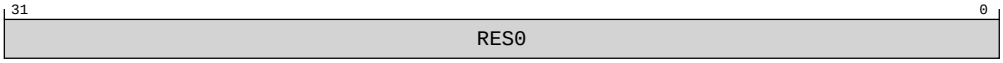


Table B-279: CLUSTERROM_DBGPSR10 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.38 CLUSTERROM_DBGPSR11, Cluster ROM table Debug Power Status Register 11

Indicates the power status for PDCOMPLEX11.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAAC

Access type

RO

Reset value

When NUM_CORES >= 12

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

When NUM_CORES < 12

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 12

Figure B-244: ext_clusterrom_dbgpsr11 bit assignments

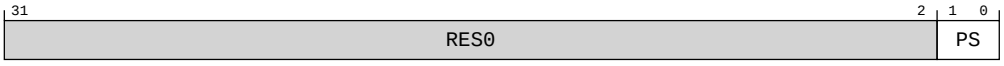


Table B-280: CLUSTERROM_DBGPSR11 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** PDCOMPLEX11 debug power domain might not be powered. **0b01** PDCOMPLEX11 debug power domain is powered. **0b10** Reserved. **0b11** PDCOMPLEX11 debug power domain is powered and must remain powered until DBGPCR11.PR is set to 0.	0b00

When NUM_CORES < 12

Figure B-245: ext_clusterrom_dbgpsr11 bit assignments

Table B-281: CLUSTERROM_DBGPSR11 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.2.39 CLUSTERROM_PRIDR0, Cluster ROM table Power Request ID Register 0

Indicates the features of the power request functionality.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xC00

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx00 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-246: ext_clusterrom_pridr0 bit assignments

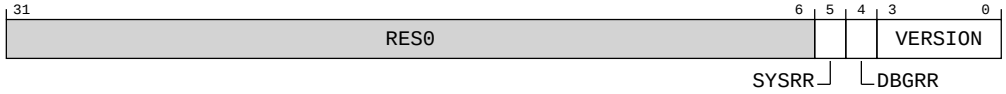


Table B-282: CLUSTERROM_PRIDR0 bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	SYSRR	System reset request functionality present. **0b0** The system reset request functionality is not implemented.	0b0
[4]	DBGRR	Debug reset request functionality present. **0b0** The debug reset request functionality is not implemented.	0b0
[3:0]	VERSION	Version of the power request functionality. **0b0001** The power request functionality version 0, and the per-core controls for power requests (e.g. ext-CLUSTERROM_DBGPCR0 and ext-CLUSTERROM_DBGPSR0), are implemented.	0b0001

B.2.2.40 CLUSTERROM_AUTHSTATUS, Cluster ROM table Authentication Status Register

Provides information about the state of the authentication interface for debug.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFB8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-247: ext_clusterrom_authstatus bit assignments

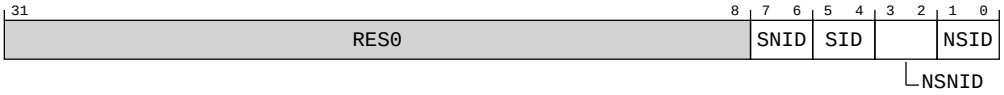


Table B-283: CLUSTERROM_AUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:6]	SNID	Secure Non-invasive Debug. **0b00** Debug level is not supported. ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled(). This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug. **0b00** Debug level is not supported.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug. **0b11** Supported and enabled.	0b11
[1:0]	NSID	Non-secure Invasive Debug. **0b00** Debug level is not supported.	0b00

B.2.2.41 CLUSTERROM_DEVARCH, Cluster ROM table Device Architecture Register

Identifies the architect and architecture of a CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFBC

Access type

RO

Reset value

0100 0111 0111 0000 0000 1010 1111 0111

Bit descriptions

Figure B-248: ext_clusterrom_devarch bit assignments

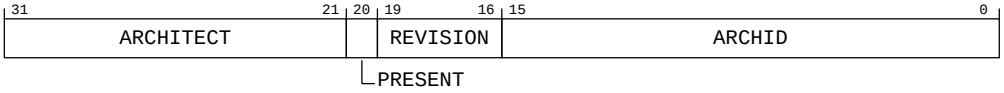


Table B-284: CLUSTERROM_DEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **0b01000111011** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. **0b1** DEVARCH information present.	0b1
[19:16]	REVISION	Revision. **0b0000** Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. **0b0000101011110111** ROM Table v0. The debug tool must inspect ext-CLUSTERROM_DEVTYPE and ext-CLUSTERROM_DEVID to determine further information about the ROM Table.	0x0AF7

B.2.2.42 CLUSTERROM_DEVID, Cluster ROM table Device Configuration Register

Indicates the capabilities of the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFC8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx10 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-249: ext_clusterrom_devid bit assignments

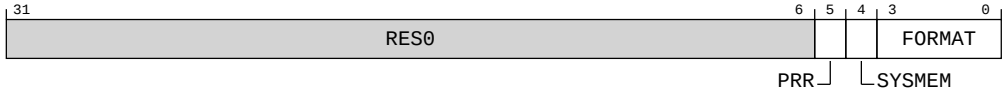


Table B-285: CLUSTERROM_DEVID bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	PRR	Power Request functionality included. **0b1** Power Request functionality included. ext-CLUSTERROM_PRIDR0 is implemented.	0b1
[4]	SYSMEM	System memory present. **0b0** System memory is not present on the bus.	0b0

Bits	Name	Description	Reset
[3:0]	FORMAT	ROM format. **0b0000** 32-bit format 0.	0b0000

B.2.2.43 CLUSTERROM_DEVTYPE, Cluster ROM table Device Type Register

A debugger can use DEVTYPE to obtain information about a component that has an unrecognized part number.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFCC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-250: ext_clusterrom_devtype bit assignments

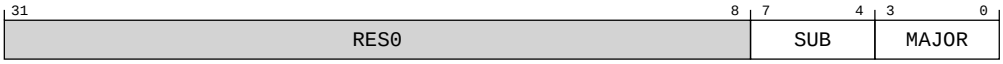


Table B-286: CLUSTERROM_DEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	SUB	Sub number **0b0000** Other, undefined.	0b0000
[3:0]	MAJOR	Major number **0b0000** Miscellaneous.	0b0000

B.2.2.44 CLUSTERROM_PIDR4, Cluster ROM table Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFD0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-251: ext_clusterrom_pidr4 bit assignments

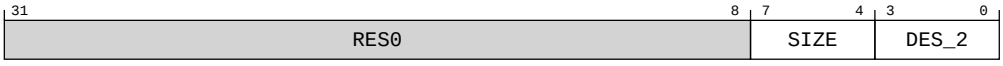


Table B-287: CLUSTERROM_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

B.2.2.45 CLUSTERROM_PIDR0, Cluster ROM table Peripheral Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFE0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1110 1000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-252: ext_clusterrom_pidr0 bit assignments

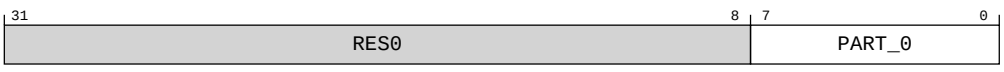


Table B-288: CLUSTERROM_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **0b11101000** DSU-110 Cluster ROM table. Bits [7:0] of part number 0x4E8.	0xE8

B.2.2.46 CLUSTERROM_PIDR1, Cluster ROM table Peripheral Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFE4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-253: ext_clusterrom_pidr1 bit assignments

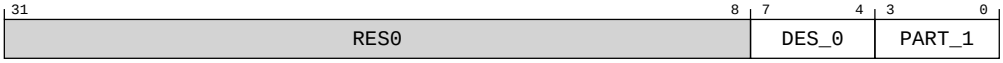


Table B-289: CLUSTERROM_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0100** DSU-110 Cluster ROM table. Bits [11:8] of part number 0x4E8.	0b0100

B.2.2.47 CLUSTERROM_PIDR2, Cluster ROM table Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFE8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0110 1011

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-254: ext_clusterrom_pidr2 bit assignments

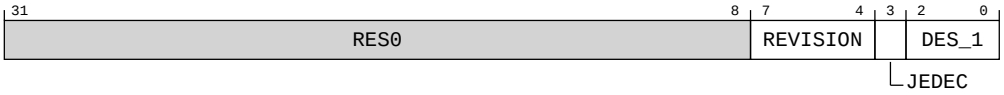


Table B-290: CLUSTERROM_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4. **0b0101** Component major revision 5. **0b0110** Component major revision 6. For DSU-110: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r1p0. • Major revision 2 corresponds to r2p0. • Major revision 3 corresponds to r2p1. • Major revision 4 corresponds to r3p0. • Major revision 5 corresponds to r3p1. • Major revision 6 corresponds to r4p0.	0b0110
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

B.2.2.48 CLUSTERROM_PIDR3, Cluster ROM table Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFEC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-255: ext_clusterrom_pidr3 bit assignments

Table B-291: CLUSTERROM_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0. **0b0001** Component minor revision 1. **0b0010** Component minor revision 2. **0b0011** Component minor revision 3. **0b0100** Component minor revision 4.	0b0000
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

B.2.2.49 CLUSTERROM_CIDR0, Cluster ROM table Component Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFF0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-256: ext_clusterrom_cidr0 bit assignments

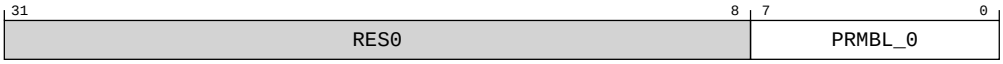


Table B-292: CLUSTERROM_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

B.2.2.50 CLUSTERROM_CIDR1, Cluster ROM table Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFF4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-257: ext_clusterrom_cidr1 bit assignments

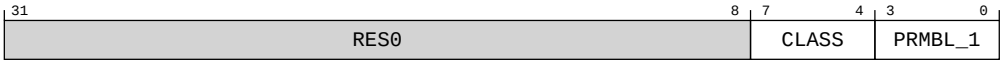


Table B-293: CLUSTERROM_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	CoreSight component class. **0b1001** CoreSight component.	0b1001
[3:0]	PRMBL_1	CoreSight component identification preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.2.2.51 CLUSTERROM_CIDR2, Cluster ROM table Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFF8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-258: ext_clusterrom_cidr2 bit assignments

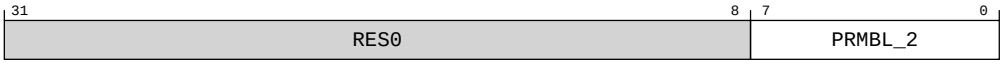


Table B-294: CLUSTERROM_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	CoreSight component identification preamble. **0b00000101** CoreSight component identification preamble.	0x05

B.2.2.52 CLUSTERROM_CIDR3, Cluster ROM table Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFFC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-259: ext_clusterrom_cidr3 bit assignments

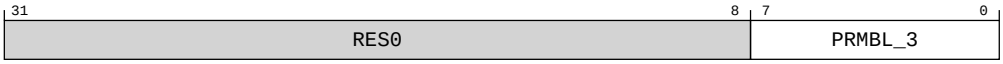


Table B-295: CLUSTERROM_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **0b10110001** CoreSight component identification preamble.	0xB1

B.2.3 External debug ROM register summary

The summary table provides an overview of all debug ROM (DBROM) table registers that are accessed externally (memory-mapped) over the debug APB bus. Individual register descriptions provide detailed information.

The summary table provides an overview of all the debug ROM table registers. For more information about a register, click on the register name in the table.

Note

- The DBROM table entry values are based on a cluster, implemented with the DSU-110. Configuration parameters are:
  - DEBUG_NEW_ADDR_MAP set to TRUE
  - DIRECT_CONNECT is set to FALSE
  - NUM_CORES is set to 12.
- The debug ROM table registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-296: DBROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	DBROM_ROMENTRY0	–	32-bit	DebugBlock ROM table Entry 0	Yes
0x004	DBROM_ROMENTRY1	–	32-bit	DebugBlock ROM table Entry 1	Yes
0x008	DBROM_ROMENTRY2	–	32-bit	DebugBlock ROM table Entry 2	Yes
0x00C	DBROM_ROMENTRY3	–	32-bit	DebugBlock ROM table Entry 3	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x010	DBROM_ROMENTRY4	—	32-bit	DebugBlock ROM table Entry 4	Yes
0x014	DBROM_ROMENTRY5	—	32-bit	DebugBlock ROM table Entry 5	Yes
0x018	DBROM_ROMENTRY6	—	32-bit	DebugBlock ROM table Entry 6	Yes
0x01C	DBROM_ROMENTRY7	—	32-bit	DebugBlock ROM table Entry 7	Yes
0x020	DBROM_ROMENTRY8	—	32-bit	DebugBlock ROM table Entry 8	Yes
0x024	DBROM_ROMENTRY9	—	32-bit	DebugBlock ROM table Entry 9	Yes
0x028	DBROM_ROMENTRY10	—	32-bit	DebugBlock ROM table Entry 10	Yes
0x02C	DBROM_ROMENTRY11	—	32-bit	DebugBlock ROM table Entry 11	Yes
0x030	DBROM_ROMENTRY12	—	32-bit	DebugBlock ROM table Entry 12	Yes
0x034	DBROM_ROMENTRY13	—	32-bit	DebugBlock ROM table Entry 13	Yes
0xA00	DBROM_DBGPCR0	—	32-bit	DebugBlock ROM table Debug Power Control Register 0	Yes
0xA80	DBROM_DBGPSR0	—	32-bit	DebugBlock ROM table Debug Power Status Register 0	Yes
0xC00	DBROM_PRIDR0	—	32-bit	DebugBlock ROM table Power Request ID Register 0	Yes
0xFB8	DBROM_AUTHSTATUS	—	32-bit	DebugBlock ROM table Authentication Status Register	Yes
0xFBC	DBROM_DEVARCH	—	32-bit	DebugBlock ROM table Device Architecture Register	Yes
0xFC8	DBROM_DEVID	—	32-bit	DebugBlock ROM table Device Configuration Register	Yes
0xFCC	DBROM_DEVTYPE	—	32-bit	DebugBlock ROM table Device Type Register	Yes
0xFD0	DBROM_PIDR4	—	32-bit	DebugBlock ROM table Peripheral Identification Register 4	Yes
0xFE0	DBROM_PIDR0	—	32-bit	DebugBlock ROM table Peripheral Identification Register 0	Yes
0xFE4	DBROM_PIDR1	—	32-bit	DebugBlock ROM table Peripheral Identification Register 1	Yes
0xFE8	DBROM_PIDR2	—	32-bit	DebugBlock ROM table Peripheral Identification Register 2	Yes
0xFEC	DBROM_PIDR3	—	32-bit	DebugBlock ROM table Peripheral Identification Register 3	Yes
0xFF0	DBROM_CIDR0	—	32-bit	DebugBlock ROM table Component Identification Register 0	Yes
0xFF4	DBROM_CIDR1	—	32-bit	DebugBlock ROM table Component Identification Register 1	Yes
0xFF8	DBROM_CIDR2	—	32-bit	DebugBlock ROM table Component Identification Register 2	Yes
0xFFC	DBROM_CIDR3	—	32-bit	DebugBlock ROM table Component Identification Register 3	Yes

B.2.3.1 DBROM_ROMENTRY0, DebugBlock ROM table Entry 0

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x000

Access type

RO

Reset value

0000 0000 0000 1100 0000 xxx0 0000 x111

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-260: ext_dbrom_romentry0 bit assignments

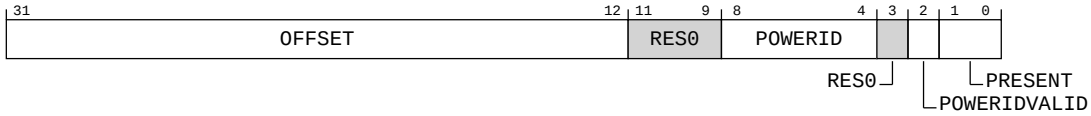


Table B-297: DBROM_ROMENTRY0 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000011000000** Cluster ROM table at address 0xC_0000.	0x000C0
[11:9]	**RES0**	Reserved	**RES0**
[8:4]	POWERID	The power domain ID of the component. **0b00000** PDCLUSTER power domain.	0b00000
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b1** The POWERID field provides a power domain ID.	0b1
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

B.2.3.2 DBROM_ROMENTRY1, DebugBlock ROM table Entry 1

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x004

Access type

RO

Reset value

0000 0000 0000 1111 0000 xxxx xxxx x011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-261: ext_dbrom_romentry1 bit assignments

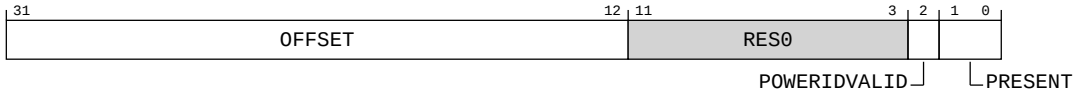


Table B-298: DBROM_ROMENTRY1 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000011110000** Cluster CTI at address 0xF_0000.	0x000F0
[11:3]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

B.2.3.3 DBROM_ROMENTRY2, DebugBlock ROM table Entry 2

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x008

Access type

RO

Reset value

0000 0000 0001 1110 0000 xxxx xxxx x011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-262: ext_dbrom_romentry2 bit assignments

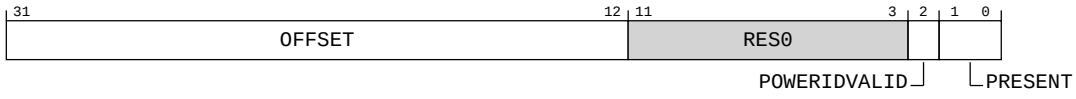


Table B-299: DBROM_ROMENTRY2 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000111100000** Core 0 CTI at address 0x1E_0000.	0x001E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

B.2.3.4 DBROM_ROMENTRY3, DebugBlock ROM table Entry 3

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x00C

Access type

RO

Reset value

When NUM_CORES >= 2

0000 0000 0010 1110 0000 xxxx xxxx x011

When NUM_CORES < 2

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 2

Figure B-263: ext_dbrom_romentry3 bit assignments

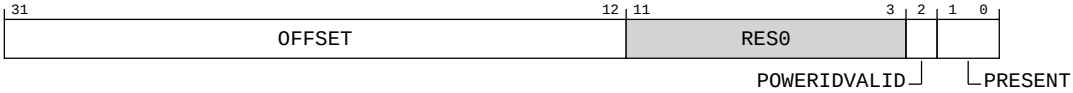


Table B-300: DBROM_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000001011100000** Core 1 CTI at address 0x2E_0000.	0x002E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 2

Figure B-264: ext_dbrom_romentry3 bit assignments

Table B-301: DBROM_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.5 DBROM_ROMENTRY4, DebugBlock ROM table Entry 4

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x010

Access type

RO

Reset value

When NUM_CORES >= 3

0000 0000 0011 1110 0000 xxxx xxxx x011

When NUM_CORES < 3

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 3

Figure B-265: ext_dbrom_romentry4 bit assignments

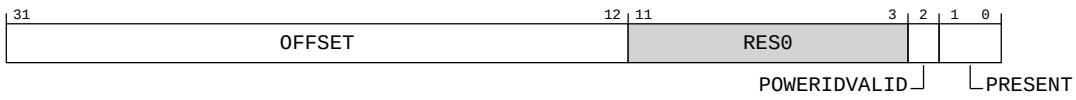


Table B-302: DBROM_ROMENTRY4 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000001111100000** Core 2 CTI at address 0x3E_0000.	0x003E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 3

Figure B-266: ext_dbrom_romentry4 bit assignments

Table B-303: DBROM_ROMENTRY4 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.6 DBROM_ROMENTRY5, DebugBlock ROM table Entry 5

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x014

Access type

RO

Reset value

When NUM_CORES >= 4

0000 0000 0100 1110 0000 xxxx xxxx x011

When NUM_CORES < 4

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 4

Figure B-267: ext_dbrom_romentry5 bit assignments

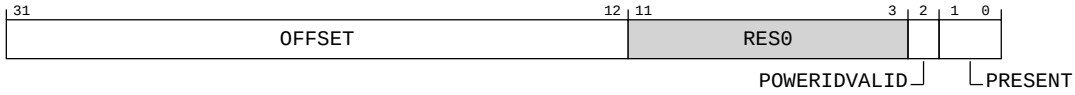


Table B-304: DBROM_ROMENTRY5 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000010011100000** Core 3 CTI at address 0x4E_0000.	0x004E0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 4

Figure B-268: ext_dbrom_romentry5 bit assignments

Table B-305: DBROM_ROMENTRY5 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.7 DBROM_ROMENTRY6, DebugBlock ROM table Entry 6

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x018

Access type

RO

Reset value

When NUM_CORES >= 5

0000 0000 0101 1110 0000 xxxx xxxx x011

When NUM_CORES < 5

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 5

Figure B-269: ext_dbrom_romentry6 bit assignments

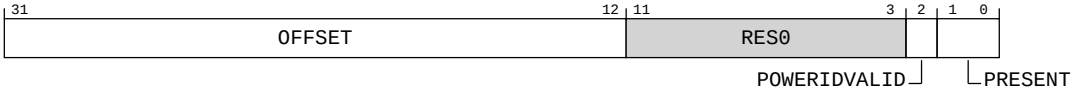


Table B-306: DBROM_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000010111100000** Core 4 CTI at address 0x5E_0000.	0x005E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 5

Figure B-270: ext_dbrom_romentry6 bit assignments

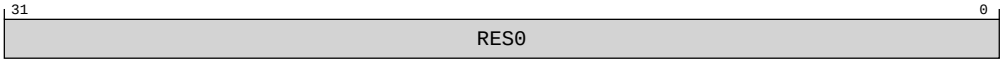


Table B-307: DBROM_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.8 DBROM_ROMENTRY7, DebugBlock ROM table Entry 7

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x01C

Access type

RO

Reset value

When NUM_CORES >= 6

0000 0000 0110 1110 0000 xxxx xxxx x011

When NUM_CORES < 6

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 6

Figure B-271: ext_dbrom_romentry7 bit assignments

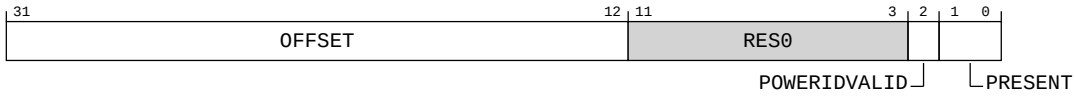


Table B-308: DBROM_ROMENTRY7 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000011011100000** Core 5 CTI at address 0x6E_0000.	0x006E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 6

Figure B-272: ext_dbrom_romentry7 bit assignments

Table B-309: DBROM_ROMENTRY7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.9 DBROM_ROMENTRY8, DebugBlock ROM table Entry 8

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x020

Access type

RO

Reset value

When NUM_CORES >= 7

0000 0000 0111 1110 0000 xxxx xxxx x011

When NUM_CORES < 7

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 7

Figure B-273: ext_dbrom_romentry8 bit assignments

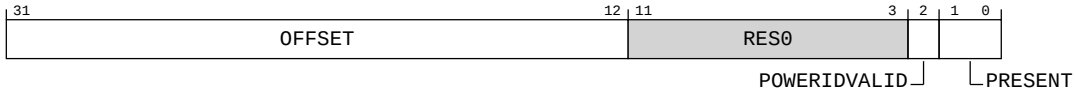


Table B-310: DBROM_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000011111100000** Core 6 CTI at address 0x7E_0000.	0x007E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 7

Figure B-274: ext_dbrom_romentry8 bit assignments

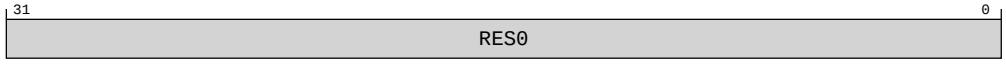


Table B-311: DBROM_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.10 DBROM_ROMENTRY9, DebugBlock ROM table Entry 9

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x024

Access type

RO

Reset value

When NUM_CORES >= 8

0000 0000 1000 1110 0000 xxxx xxxx x011

When NUM_CORES < 8

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 8

Figure B-275: ext_dbrom_romentry9 bit assignments

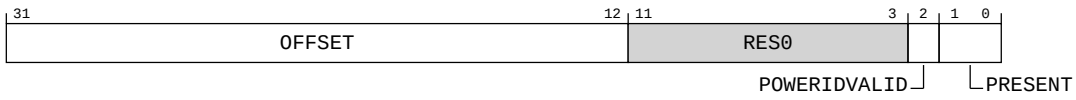


Table B-312: DBROM_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000100011100000** Core 7 CTI at address 0x8E_0000.	0x008E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 8

Figure B-276: ext_dbrom_romentry9 bit assignments

Table B-313: DBROM_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.11 DBROM_ROMENTRY10, DebugBlock ROM table Entry 10

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x028

Access type

RO

Reset value

When NUM_CORES >= 9

0000 0000 1001 1110 0000 xxxx xxxx x011

When NUM_CORES < 9

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 9

Figure B-277: ext_dbrom_romentry10 bit assignments

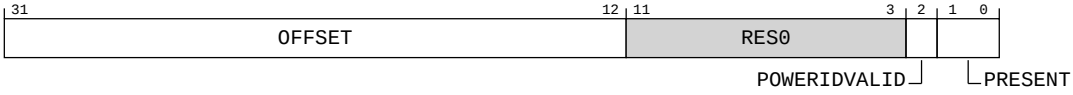


Table B-314: DBROM_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000100111100000** Core 8 CTI at address 0x9E_0000.	0x009E0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 9

Figure B-278: ext_dbrom_romentry10 bit assignments

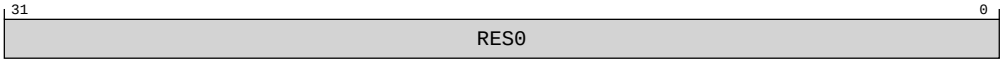


Table B-315: DBROM_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.12 DBROM_ROMENTRY11, DebugBlock ROM table Entry 11

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x02C

Access type

RO

Reset value

When NUM_CORES >= 10

0000 0000 1010 1110 0000 xxxx xxxx x011

When NUM_CORES < 10

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 10

Figure B-279: ext_dbrom_romentry11 bit assignments

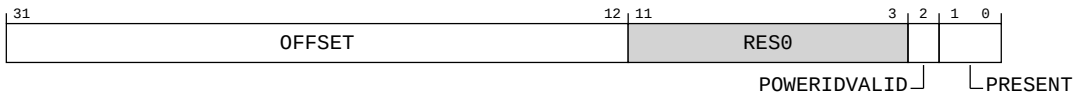


Table B-316: DBROM_ROMENTRY11 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000101011100000** Core 9 CTI at address 0xAE_0000.	0x00AE0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 10

Figure B-280: ext_dbrom_romentry11 bit assignments

Table B-317: DBROM_ROMENTRY11 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.13 DBROM_ROMENTRY12, DebugBlock ROM table Entry 12

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x030

Access type

RO

Reset value

When NUM_CORES >= 11

0000 0000 1011 1110 0000 xxxx xxxx x011

When NUM_CORES < 11

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 11

Figure B-281: ext_dbrom_romentry12 bit assignments

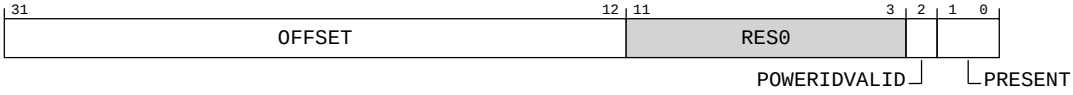


Table B-318: DBROM_ROMENTRY12 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000101111100000** Core 10 CTI at address 0xBE_0000.	0x00BE0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 11

Figure B-282: ext_dbrom_romentry12 bit assignments

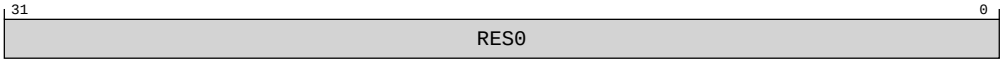


Table B-319: DBROM_ROMENTRY12 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.14 DBROM_ROMENTRY13, DebugBlock ROM table Entry 13

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x034

Access type

RO

Reset value

When NUM_CORES >= 12

0000 0000 1100 1110 0000 xxxx xxxx x011

When NUM_CORES < 12

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

When NUM_CORES >= 12

Figure B-283: ext_dbrom_romentry13 bit assignments

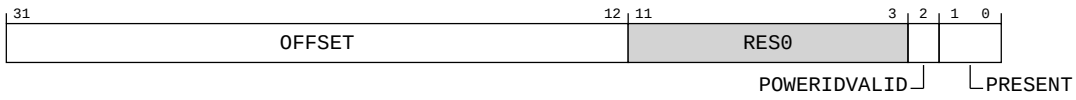


Table B-320: DBROM_ROMENTRY13 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000110011100000** Core 11 CTI at address 0xCE_0000.	0x00CE0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

When NUM_CORES < 12

Figure B-284: ext_dbrom_romentry13 bit assignments

Table B-321: DBROM_ROMENTRY13 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.2.3.15 DBROM_DBGPCR0, DebugBlock ROM table Debug Power Control Register 0

Controls power requests for PDCLUSTER.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xA00

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-285: ext_dbrom_dbgpcr0 bit assignments

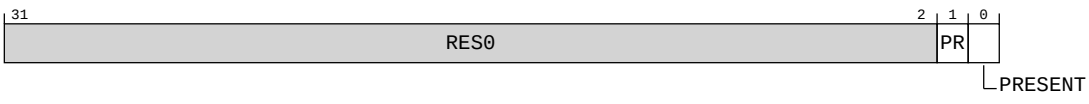


Table B-322: DBROM_DBGPCR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **0b0** Power is not requested for PDCLUSTER. **0b1** Power is requested for PDCLUSTER.	x
[0]	PRESENT	Power request implemented. **0b1** Power request for PDCLUSTER is implemented.	x

B.2.3.16 DBROM_DBGPSR0, DebugBlock ROM table Debug Power Status Register 0

Indicates the power status for PDCLUSTER.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xA80

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-286: ext_dbrom_dbgpsr0 bit assignments

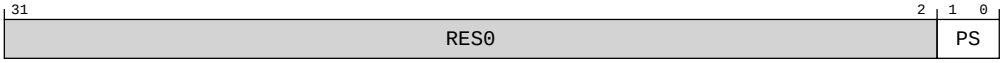


Table B-323: DBROM_DBGPSR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** PDCLUSTER might not be powered. **0b01** PDCLUSTER is powered.	0b00

B.2.3.17 DBROM_PRIDR0, DebugBlock ROM table Power Request ID Register 0

Indicates the features of the power request functionality.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xC00

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx00 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-287: ext_dbrom_pridr0 bit assignments

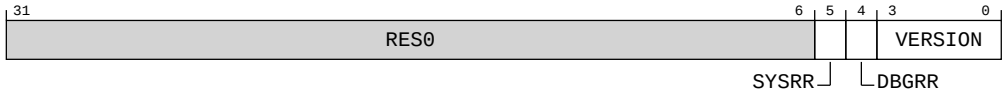


Table B-324: DBROM_PRIDR0 bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	SYSRR	System reset request functionality present. **0b0** The system reset request functionality is not implemented.	0b0
[4]	DBGRR	Debug reset request functionality present. **0b0** The debug reset request functionality is not implemented.	0b0
[3:0]	VERSION	Version of the power request functionality. **0b0001** The power request functionality version 0, and the ext-DBROM_DBGPCR0, ext-DBROM_DBGPSR0, which provide controls for power requests, are implemented.	0b0001

B.2.3.18 DBROM_AUTHSTATUS, DebugBlock ROM table Authentication Status Register

Provides information about the state of the authentication interface for debug.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFB8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-288: ext_dbrom_authstatus bit assignments

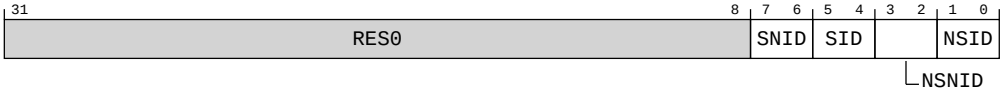


Table B-325: DBROM_AUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:6]	SNID	Secure Non-invasive Debug. **0b00** Debug level is not supported. ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled(). This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug. **0b00** Debug level is not supported.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug. **0b11** Supported and enabled.	0b11
[1:0]	NSID	Non-secure Invasive Debug. **0b00** Debug level is not supported.	0b00

B.2.3.19 DBROM_DEVARCH, DebugBlock ROM table Device Architecture Register

Identifies the architect and architecture of a CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFBC

Access type

RO

Reset value

0100 0111 0111 0000 0000 1010 1111 0111

Bit descriptions

Figure B-289: ext_dbrom_devarch bit assignments

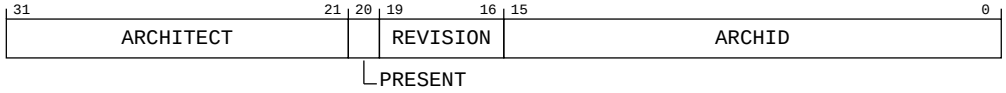


Table B-326: DBROM_DEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **0b01000111011** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. **0b1** DEVARCH information present.	0b1
[19:16]	REVISION	Revision. **0b0000** Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. **0b0000101011110111** ROM Table v0. The debug tool must inspect ext-DBROM_DEVTYPE and ext-DBROM_DEVID to determine further information about the ROM Table.	0x0AF7

B.2.3.20 DBROM_DEVID, DebugBlock ROM table Device Configuration Register

Indicates the capabilities of the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFC8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xx10 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-290: ext_dbrom_devid bit assignments

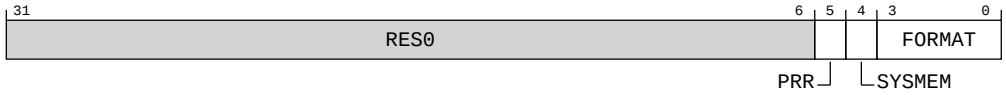


Table B-327: DBROM_DEVID bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	PRR	Power Request functionality included. **0b1** Power Request functionality included. ext-DBROM_PRIDR0 is implemented.	0b1
[4]	SYSMEM	System memory present. **0b0** System memory is not present on the bus.	0b0
[3:0]	FORMAT	ROM format. **0b0000** 32-bit format 0.	0b0000

B.2.3.21 DBROM_DEVTYPE, DebugBlock ROM table Device Type Register

A debugger can use DEVTYPE to obtain information about a component that has an unrecognized part number.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFCC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-291: ext_dbrom_devtype bit assignments

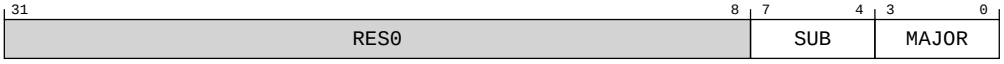


Table B-328: DBROM_DEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SUB	Sub number **0b0000** Other, undefined.	0b0000
[3:0]	MAJOR	Major number **0b0000** Miscellaneous.	0b0000

B.2.3.22 DBROM_PIDR4, DebugBlock ROM table Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFD0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-292: ext_dbrom_pidr4 bit assignments

Table B-329: DBROM_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

B.2.3.23 DBROM_PIDR0, DebugBlock ROM table Peripheral Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFE0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1110 0111

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-293: ext_dbrom_pidr0 bit assignments

Table B-330: DBROM_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **0b11100111** DSU-110 DebugBlock ROM table. Bits [7:0] of part number 0x4E7.	0xE7

B.2.3.24 DBROM_PIDR1, DebugBlock ROM table Peripheral Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFE4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-294: ext_dbrom_pidr1 bit assignments

Table B-331: DBROM_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0100** DSU-110 DebugBlock ROM table. Bits [11:8] of part number 0x4E7.	0b0100

B.2.3.25 DBROM_PIDR2, DebugBlock ROM table Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFE8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0110 1011

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-295: ext_dbrom_pidr2 bit assignments

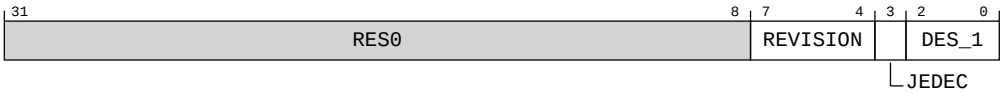


Table B-332: DBROM_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4. **0b0101** Component major revision 5. **0b0110** Component major revision 6. For DSU-110: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r1p0. • Major revision 2 corresponds to r2p0. • Major revision 3 corresponds to r2p1. • Major revision 4 corresponds to r3p0. • Major revision 5 corresponds to r3p1. • Major revision 6 corresponds to r4p0.	0b0110
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

B.2.3.26 DBROM_PIDR3, DebugBlock ROM table Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFEC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-296: ext_dbrom_pidr3 bit assignments

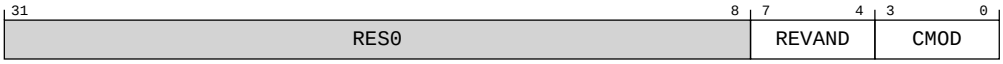


Table B-333: DBROM_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0. **0b0001** Component minor revision 1. **0b0010** Component minor revision 2. **0b0011** Component minor revision 3. **0b0100** Component minor revision 4.	0b0000
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

B.2.3.27 DBROM_CIDR0, DebugBlock ROM table Component Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFF0

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-297: ext_dbrom_cidr0 bit assignments

Table B-334: DBROM_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

B.2.3.28 DBROM_CIDR1, DebugBlock ROM table Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFF4

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-298: ext_dbrom_cidr1 bit assignments

Table B-335: DBROM_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	CoreSight component class. **0b1001** CoreSight component.	0b1001
[3:0]	PRMBL_1	CoreSight component identification preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.2.3.29 DBROM_CIDR2, DebugBlock ROM table Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFF8

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-299: ext_dbrom_cidr2 bit assignments

Table B-336: DBROM_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	CoreSight component identification preamble. **0b00000101** CoreSight component identification preamble.	0x05

B.2.3.30 DBROM_CIDR3, DebugBlock ROM table Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFFC

Access type

RO

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-300: ext_dbrom_cidr3 bit assignments

Table B-337: DBROM_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **0b10110001** CoreSight component identification preamble.	0xB1

B.2.4 External cluster PMU register summary

The summary table provides an overview of all the *Cluster Performance Monitoring Unit* (CLUSTERPMU) registers that are accessed externally (memory-mapped) over the Debug APB bus. Individual register descriptions provide detailed information.

The summary table provides an overview of all the cluster PMU registers that are accessed externally (memory-mapped) over the debug APB bus. For more information about a register, click on the register name in the table.

- The cluster PMU registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the DSU-110 is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- The part number is `0x4EA`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-338: CLUSTERPMU registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0	CLUSTERPMU_PMEVCNTR0	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x8	CLUSTERPMU_PMEVCNTR1	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x10	CLUSTERPMU_PMEVCNTR2	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x18	CLUSTERPMU_PMEVCNTR3	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x20	CLUSTERPMU_PMEVCNTR4	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x28	CLUSTERPMU_PMEVCNTR5	—	64-bit	Cluster Performance Monitors Event Count Registers	No
0x0F8	CLUSTERPMU_PMCCNTR	—	64-bit	Cluster Performance Monitors Cycle Counter	No
0x0FC	CLUSTERPMU_PMCCNTR	—	64-bit	Cluster Performance Monitors Cycle Counter	No
0x400	CLUSTERPMU_PMEVTYPER0	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x404	CLUSTERPMU_PMEVTYPER1	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x408	CLUSTERPMU_PMEVTYPER2	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x40C	CLUSTERPMU_PMEVTYPER3	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x410	CLUSTERPMU_PMEVTYPER4	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x414	CLUSTERPMU_PMEVTYPER5	—	32-bit	Cluster Performance Monitors Event Type Registers	No
0x47C	CLUSTERPMU_PMCCFILTR	—	32-bit	Cluster Performance Monitors Cycle Counter Filter Register	No
0x610	CLUSTERPMU_PMSSSR	—	32-bit	Cluster Performance Monitors Snapshot Status register	No
0x614	CLUSTERPMU_PMOVSSR	—	32-bit	Cluster Performance Monitors Overflow Status Snapshot register	No
0x618	CLUSTERPMU_PMCCNTSR	—	64-bit	Cluster Performance Monitors Cycle Counter Snapshot Register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x61C	CLUSTERPMU_PMCCNTSR	—	64-bit	Cluster Performance Monitors Cycle Counter Snapshot Register	No
0x620	CLUSTERPMU_PMEVCNTSR0	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x628	CLUSTERPMU_PMEVCNTSR1	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x630	CLUSTERPMU_PMEVCNTSR2	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x638	CLUSTERPMU_PMEVCNTSR3	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x640	CLUSTERPMU_PMEVCNTSR4	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x648	CLUSTERPMU_PMEVCNTSR5	—	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x6F0	CLUSTERPMU_PMSSCR	—	32-bit	Cluster Performance Monitors Snapshot Capture register	No
0x6F4	CLUSTERPMU_PMSSRR	—	32-bit	Cluster Performance Monitors Snapshot Reset register	No
0xC00	CLUSTERPMU_PMCNTENSET	—	32-bit	Cluster Performance Monitors Count Enable Set register	No
0xC20	CLUSTERPMU_PMCNTENCLR	—	32-bit	Cluster Performance Monitors Count Enable Clear register	No
0xC40	CLUSTERPMU_PMINTENSET	—	32-bit	Cluster Performance Monitors Interrupt Enable Set register	No
0xC60	CLUSTERPMU_PMINTENCLR	—	32-bit	Cluster Performance Monitors Interrupt Enable Clear register	No
0xC80	CLUSTERPMU_PMOVSCLR	—	32-bit	Cluster Performance Monitors Overflow Flag Status Clear register	No
0xCC0	CLUSTERPMU_PMOVSSET	—	32-bit	Cluster Performance Monitors Overflow Flag Status Set register	No
0xE00	CLUSTERPMU_PMCFGR	—	32-bit	Cluster Performance Monitors Configuration Register	No
0xE04	CLUSTERPMU_PMCR	—	32-bit	Cluster Performance Monitors Control Register	No
0xE20	CLUSTERPMU_PMCEID0	—	32-bit	Cluster Performance Monitors Common Event Identification register 0	No
0xE24	CLUSTERPMU_PMCEID1	—	32-bit	Cluster Performance Monitors Common Event Identification register 1	No
0xE28	CLUSTERPMU_PMCEID2	—	32-bit	Cluster Performance Monitors Common Event Identification register 2	No
0xE2C	CLUSTERPMU_PMCEID3	—	32-bit	Cluster Performance Monitors Common Event Identification register 3	No
0xFA0	CLUSTERPMU_PMCLAIMSET	—	32-bit	Cluster Performance Monitors Claim Set register	No
0xFA4	CLUSTERPMU_PMCLAIMCLR	—	32-bit	Cluster Performance Monitors Claim Clear register	No
0xFA8	CLUSTERPMU_PMDEVAFF0	—	32-bit	Cluster Performance Monitors Device Affinity register 0	No
0xFAC	CLUSTERPMU_PMDEVAFF1	—	32-bit	Cluster Performance Monitors Device Affinity register 1	No
0xFB8	CLUSTERPMU_PMAUTHSTATUS	—	32-bit	Cluster Performance Monitors Authentication Status register	No
0xFBC	CLUSTERPMU_PMDEVARCH	—	32-bit	Cluster Performance Monitors Device Architecture register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFC8	CLUSTERPMU_PMDEVID	—	32-bit	Cluster Performance Monitors Device ID register	No
0xFCC	CLUSTERPMU_PMDEVTYPE	—	32-bit	Cluster Performance Monitors Device Type register	No
0xFD0	CLUSTERPMU_PMPIDR4	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 4	No
0xFE0	CLUSTERPMU_PMPIDR0	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 0	No
0xFE4	CLUSTERPMU_PMPIDR1	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 1	No
0xFE8	CLUSTERPMU_PMPIDR2	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 2	No
0xFEC	CLUSTERPMU_PMPIDR3	—	32-bit	Cluster Performance Monitors Peripheral Identification Register 3	No
0xFF0	CLUSTERPMU_PMCIDR0	—	32-bit	Cluster Performance Monitors Component Identification Register 0	No
0xFF4	CLUSTERPMU_PMCIDR1	—	32-bit	Cluster Performance Monitors Component Identification Register 1	No
0xFF8	CLUSTERPMU_PMCIDR2	—	32-bit	Cluster Performance Monitors Component Identification Register 2	No
0xFFC	CLUSTERPMU_PMCIDR3	—	32-bit	Cluster Performance Monitors Component Identification Register 3	No

B.2.4.1 CLUSTERPMU_PMEVCNTR0, Cluster Performance Monitors Event Count Registers

Holds event counter 0, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x0

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-301: ext_clusterpmu_pmevcntr0 bit assignments

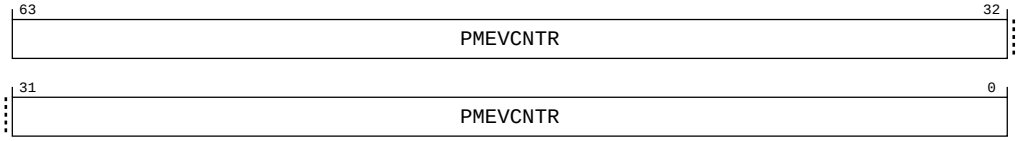


Table B-339: CLUSTERPMU_PMEVCNTR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 0.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x0

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.2 CLUSTERPMU_PMEVCNTR1, Cluster Performance Monitors Event Count Registers

Holds event counter 1, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x8

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-302: ext_clusterpmu_pmevcntr1 bit assignments

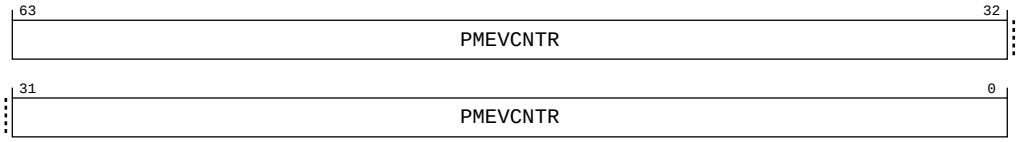


Table B-341: CLUSTERPMU_PMEVCNTR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 1.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x8

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.3 CLUSTERPMU_PMEVCNTR2, Cluster Performance Monitors Event Count Registers

Holds event counter 2, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x10

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-303: ext_clusterpmu_pmevcntr2 bit assignments

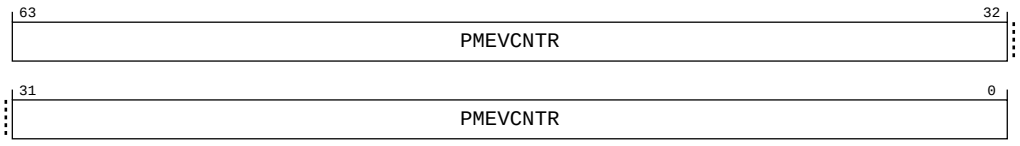


Table B-343: CLUSTERPMU_PMEVCNTR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 2.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x10

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.4 CLUSTERPMU_PMEVCNTR3, Cluster Performance Monitors Event Count Registers

Holds event counter 3, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x18

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-304: ext_clusterpmu_pmevcntr3 bit assignments

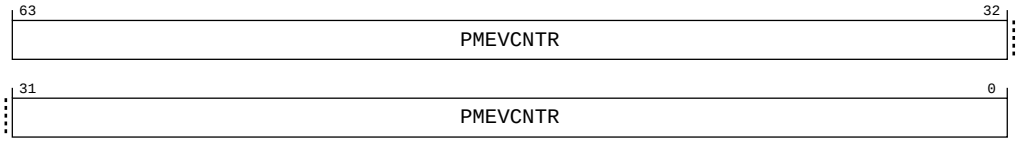


Table B-345: CLUSTERPMU_PMEVCNTR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 3.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x18

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.5 CLUSTERPMU_PMEVCNTR4, Cluster Performance Monitors Event Count Registers

Holds event counter 4, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x20

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-305: ext_clusterpmu_pmevcntr4 bit assignments

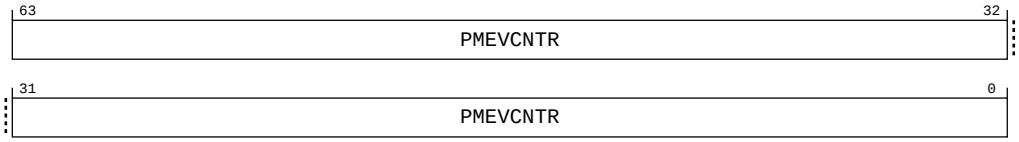


Table B-347: CLUSTERPMU_PMEVCNTR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 4.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x20

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.6 CLUSTERPMU_PMEVCNTR5, Cluster Performance Monitors Event Count Registers

Holds event counter 5, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x28

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-306: ext_clusterpmu_pmevcntr5 bit assignments

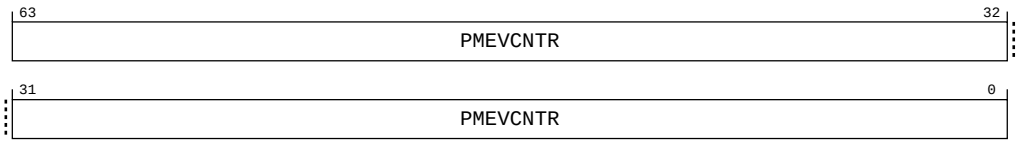


Table B-349: CLUSTERPMU_PMEVCNTR5 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 5.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x28

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.7 CLUSTERPMU_PMCCNTR, Cluster Performance Monitors Cycle Counter

Holds the value of the Cluster Cycle Counter, CCNT, that counts processor clock cycles.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offsets (2)

0x0F8,0x0FC

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-307: ext_clusterpmu_pmccntr bit assignments

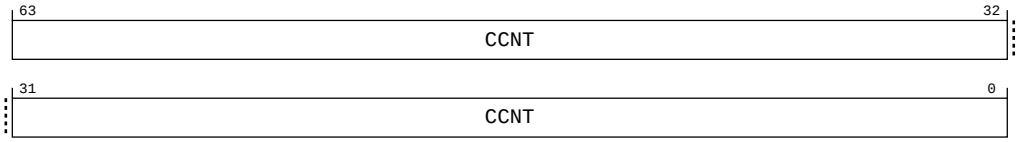


Table B-351: CLUSTERPMU_PMCCNTR bit descriptions

Bits	Name	Description	Reset
[63:0]	CCNT	Cycle count. Depending on the values of ext-CLUSTERPMU_PMCR.{LC,D}, the cycle count increments in one of the following ways: • Every processor clock cycle. • Every 64th processor clock cycle. Writing 1 to ext-CLUSTERPMU_PMCR.C sets this field to 0.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x0F8

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

Component	Offset
CLUSTERPMU	0x0FC

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.8 CLUSTERPMU_PMEVTYPER0, Cluster Performance Monitors Event Type Registers

Configures event counter n, where n is 0 to 30.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x400

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-308: ext_clusterpmu_pmevtyper0 bit assignments

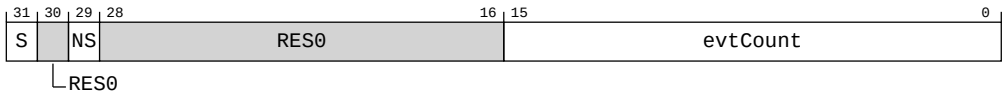


Table B-354: CLUSTERPMU_PMEVTYPER0 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	16{x}

Accessibility

Component	Offset
CLUSTERPMU	0x400

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.9 CLUSTERPMU_PMEVTYPER1, Cluster Performance Monitors Event Type Registers

Configures event counter n, where n is 0 to 30.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x404

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-309: ext_clusterpmu_pmevtyper1 bit assignments

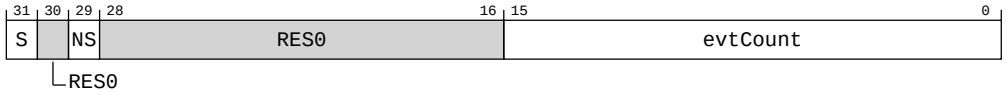


Table B-356: CLUSTERPMU_PMEVTYPER1 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	16{x}

Accessibility

Component	Offset
CLUSTERPMU	0x404

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.10 CLUSTERPMU_PMEVTYPER2, Cluster Performance Monitors Event Type Registers

Configures event counter n, where n is 0 to 30.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x408

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-310: ext_clusterpmu_pmevtyper2 bit assignments

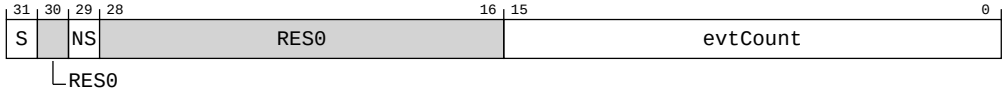


Table B-358: CLUSTERPMU_PMEVTYPER2 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	16{x}

Accessibility

Component	Offset
CLUSTERPMU	0x408

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.11 CLUSTERPMU_PMEVTYPER3, Cluster Performance Monitors Event Type Registers

Configures event counter n, where n is 0 to 30.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x40C

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-311: ext_clusterpmu_pmevtyper3 bit assignments

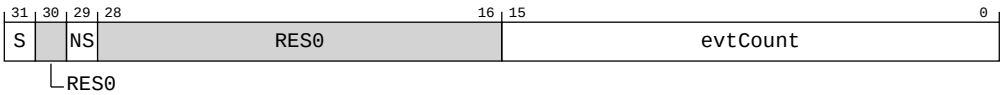


Table B-360: CLUSTERPMU_PMEVTYPER3 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	16{x}

Accessibility

Component	Offset
CLUSTERPMU	0x40C

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.12 CLUSTERPMU_PMEVTYPER4, Cluster Performance Monitors Event Type Registers

Configures event counter n, where n is 0 to 30.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x410

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-312: ext_clusterpmu_pmevtyper4 bit assignments

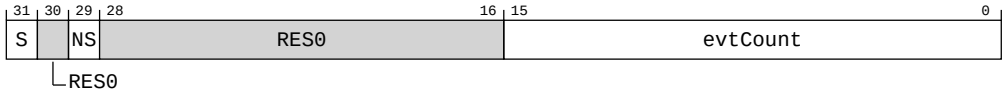


Table B-362: CLUSTERPMU_PMEVTYPER4 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	`16{x}`

Accessibility

Component	Offset
CLUSTERPMU	0x410

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.13 CLUSTERPMU_PMEVTYPER5, Cluster Performance Monitors Event Type Registers

Configures event counter n, where n is 0 to 30.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x414

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-313: ext_clusterpmu_pmevtyper5 bit assignments

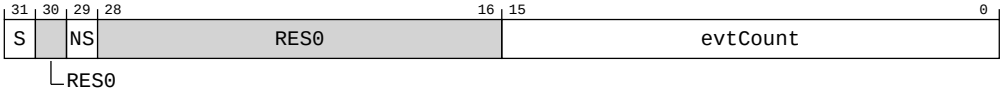


Table B-364: CLUSTERPMU_PMEVTYPER5 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	16{x}

Accessibility

Component	Offset
CLUSTERPMU	0x414

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.14 CLUSTERPMU_PMCCFILTR, Cluster Performance Monitors Cycle Counter Filter Register

Determines the modes in which the Cluster Cycle Counter, ext-CLUSTERPMU_PMCCNTR, increments. Unlike the per-core PMUs, the Cluster Cycle Counter always increments.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x47C

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-314: ext_clusterpmu_pmccfiltr bit assignments

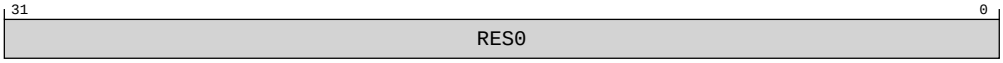


Table B-366: CLUSTERPMU_PMCCFILTR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

Accessibility

Component	Offset
CLUSTERPMU	0x47C

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.15 CLUSTERPMU_PMSSSR, Cluster Performance Monitors Snapshot Status register

Holds status information about the captured counters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x610

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx1

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-315: ext_clusterpmu_pmsssr bit assignments

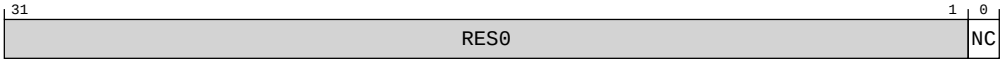


Table B-368: CLUSTERPMU_PMSSSR bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	NC	No capture. Indicates whether the PMU counters have been captured. **0b0** PMU counters captured. **0b1** PMU counters not captured.	0b1

Accessibility

Component	Offset
CLUSTERPMU	0x610

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.16 CLUSTERPMU_PMOVSSR, Cluster Performance Monitors Overflow Status Snapshot register

Captured copy of ext-CLUSTERPMU_PMOVSR. Once captured, the value in ext-CLUSTERPMU_PMOVSSR is unaffected by writes to ext-CLUSTERPMU_PMOVSSET and ext-CLUSTERPMU_PMOVSCLR.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x614

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-316: ext_clusterpmu_pmovssr bit assignments

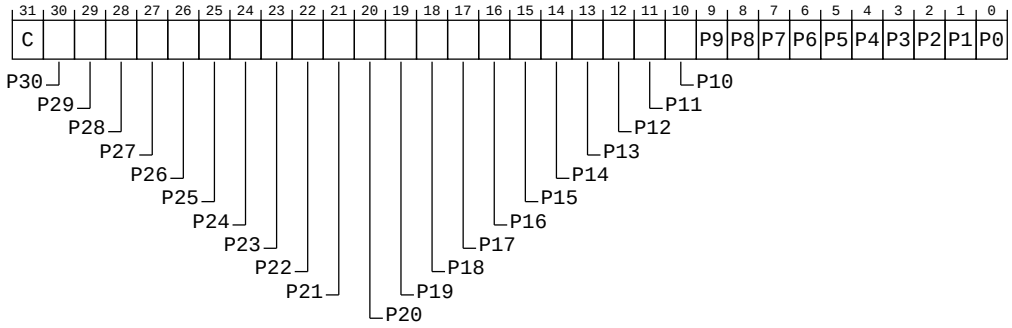


Table B-370: CLUSTERPMU_PMOVSSR bit descriptions

Bits	Name	Description	Reset
[31]	C	Cycle counter overflow bit. **0b0** When read, means the cycle counter has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means the cycle counter has overflowed since this bit was last cleared. When written, sets the cycle counter overflow bit to 1.	x

Bits	Name	Description	Reset
[30:0]	P<n>, bit[n], where n = 30 to 0	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	`31{x}`

Accessibility

Component	Offset
CLUSTERPMU	0x614

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.17 CLUSTERPMU_PMCCNTSR, Cluster Performance Monitors Cycle Counter Snapshot Register

Holds the captured copy of ext-CLUSTERPMU_PMCCNTR. Once captured, the value in ext-CLUSTERPMU_PMCCNTSR is unaffected by writes to ext-CLUSTERPMU_PMCCNTR and ext-CLUSTERPMU_PMCR.C.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offsets (2)

0x618,0x61C

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-317: ext_clusterpmu_pmccntsr bit assignments

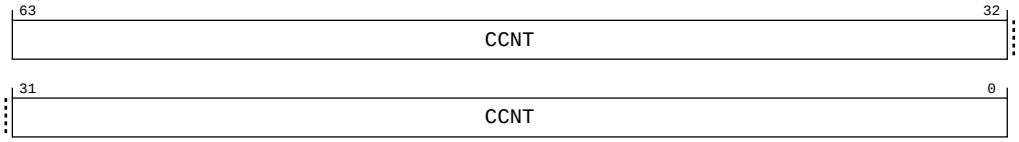


Table B-372: CLUSTERPMU_PMCCNTSR bit descriptions

Bits	Name	Description	Reset
[63:0]	CCNT	Captured cycle count.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x618

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

Component	Offset
CLUSTERPMU	0x61C

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.18 CLUSTERPMU_PMEVCNTSR0, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 0, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x620

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx


Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-318: ext_clusterpmu_pmevcntsr0 bit assignments

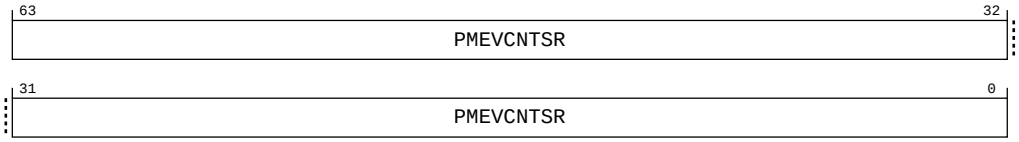


Table B-375: CLUSTERPMU_PMEVCNTSR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 0.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x620

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.19 CLUSTERPMU_PMEVCNTSR1, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 1, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x628

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-319: ext_clusterpmu_pmevcntsr1 bit assignments

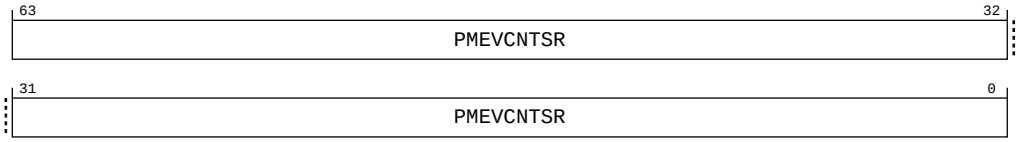


Table B-377: CLUSTERPMU_PMEVCNTSR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 1.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x628

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.20 CLUSTERPMU_PMEVCNTSR2, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 2, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x630

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-320: ext_clusterpmu_pmevcntsr2 bit assignments

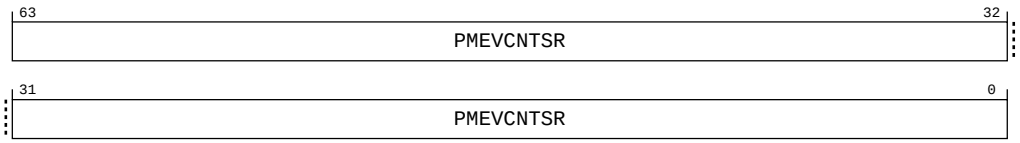


Table B-379: CLUSTERPMU_PMEVCNTSR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 2.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x630

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.21 CLUSTERPMU_PMEVCNTSR3, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 3, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x638

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-321: ext_clusterpmu_pmevcntsr3 bit assignments

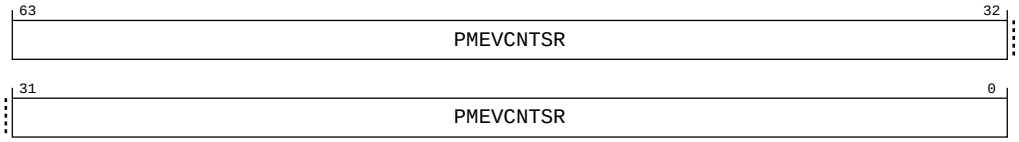


Table B-381: CLUSTERPMU_PMEVCNTSR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 3.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x638

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.22 CLUSTERPMU_PMEVCNTSR4, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 4, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x640

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-322: ext_clusterpmu_pmevcntsr4 bit assignments

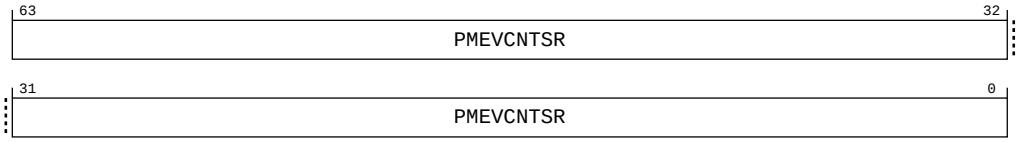


Table B-383: CLUSTERPMU_PMEVCNTSR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 4.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x640

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.23 CLUSTERPMU_PMEVCNTSR5, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 5, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERPMU

Register offset

0x648

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-323: ext_clusterpmu_pmevcntsr5 bit assignments

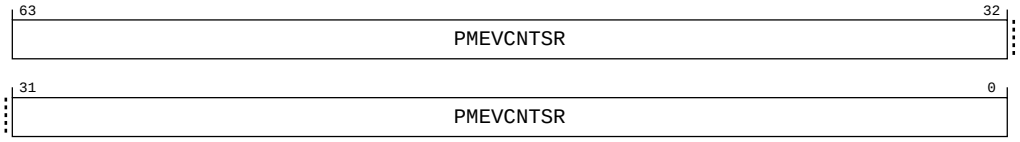


Table B-385: CLUSTERPMU_PMEVCNTSR5 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 5.	64{x}

Accessibility

Component	Offset
CLUSTERPMU	0x648

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.24 CLUSTERPMU_PMSSCR, Cluster Performance Monitors Snapshot Capture register

Provides a mechanism for software to initiate a sample.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x6F0

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-324: ext_clusterpmu_pmsscr bit assignments

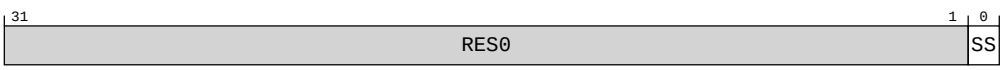


Table B-387: CLUSTERPMU_PMSSCR bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	SS	Capture now. The possible values for writing to this bit are: **0b0** Ignored. **0b1** Initiate a capture immediately.	0b0

Accessibility

Component	Offset
CLUSTERPMU	0x6F0

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

WI

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

WO

Otherwise

ERROR

B.2.4.25 CLUSTERPMU_PMSSRR, Cluster Performance Monitors Snapshot Reset register

Configure PMU Snapshot to reset counters after each sample taken.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0x6F4

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-325: ext_clusterpmu_pmssrr bit assignments

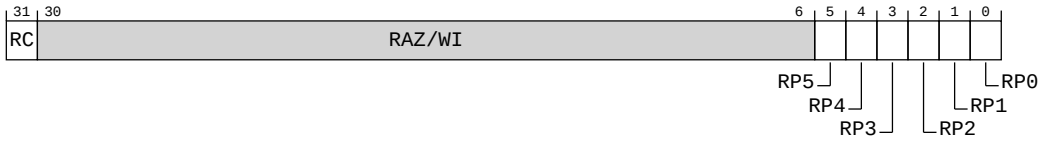


Table B-389: CLUSTERPMU_PMSSRR bit descriptions

Bits	Name	Description	Reset
[31]	RC	Reset cycle counter. Indicates whether ext-CLUSTERPMU_PMCCNTR and ext-CLUSTERPMU_PMOVSR are to be reset after a capture. **0b0** Do not reset cycle counter and ext-CLUSTERPMU_PMOVSR[31] on capture. **0b1** Reset cycle counter and ext-CLUSTERPMU_PMOVSR[31] on capture.	0b0
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	RP5	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[4]	RP4	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x

Bits	Name	Description	Reset
[3]	RP3	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[2]	RP2	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[1]	RP1	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[0]	RP0	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x

Accessibility

Component	Offset
CLUSTERPMU	0x6F4

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.26 CLUSTERPMU_PMCNTENSET, Cluster Performance Monitors Count Enable Set register

Enables the Cycle Count Register, ext-CLUSTERPMU_PMCCNTR, and any implemented event counters AArch64-IMP_CLUSTERPMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC00

Access type

See bit descriptions

Reset value

x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-326: ext_clusterpmu_pmcntenset bit assignments

Table B-391: CLUSTERPMU_PMCNTENSET bit descriptions

Bits	Name	Description	Reset
[31]	C	ext-CLUSTERPMU_PMCCNTR enable bit. Enables the cycle counter register. **0b0** When read, means the cycle counter is disabled. When written, has no effect. **0b1** When read, means the cycle counter is enabled. When written, enables the cycle counter.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**

Bits	Name	Description	Reset
[5]	P5	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[4]	P4	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[3]	P3	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[2]	P2	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[1]	P1	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x

Bits	Name	Description	Reset
[0]	P0	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x

Accessibility

Component	Offset
CLUSTERPMU	0xC00

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.27 CLUSTERPMU_PMCNTENCLR, Cluster Performance Monitors Count Enable Clear register

Disables the Cycle Count Register, ext-CLUSTERPMU_PMCCNTR, and any implemented event counters AArch64-IMP_CLUSTERPMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC20

Access type

See bit descriptions

Reset value

x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-327: ext_clusterpmu_pmcntenclr bit assignments

Table B-393: CLUSTERPMU_PMCNTENCLR bit descriptions

Bits	Name	Description	Reset
[31]	C	ext-CLUSTERPMU_PMCCNTR disable bit. Disables the cycle counter register. **0b0** When read, means the cycle counter is disabled. When written, has no effect. **0b1** When read, means the cycle counter is enabled. When written, disables the cycle counter.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[4]	P4	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	x

Bits	Name	Description	Reset
[3]	P3	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[2]	P2	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[1]	P1	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[0]	P0	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	x

Accessibility

Component	Offset
CLUSTERPMU	0xC20

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.28 CLUSTERPMU_PMINTENSET, Cluster Performance Monitors Interrupt Enable Set register

Enables the generation of interrupt requests on overflows from the Cycle Count Register, ext-CLUSTERPMU_PMCCNTR, and the event counters ext-CLUSTERPMU_PMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC40

Access type

See bit descriptions

Reset value

x000 0000 0000 0000 0000 0000 00xx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-328: ext_clusterpmu_pmintenset bit assignments

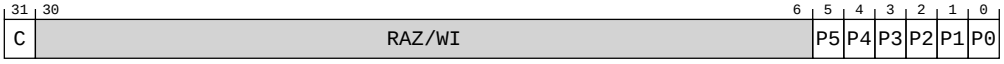


Table B-395: CLUSTERPMU_PMINTENSET bit descriptions

Bits	Name	Description	Reset
[31]	C	ext-CLUSTERPMU_PMCCNTR overflow interrupt request enable bit. **0b0** When read, means the cycle counter overflow interrupt request is disabled. When written, has no effect. **0b1** When read, means the cycle counter overflow interrupt request is enabled. When written, enables the cycle count overflow interrupt request.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[4]	P4	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[3]	P3	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[2]	P2	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x

Bits	Name	Description	Reset
[1]	P1	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[0]	P0	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x

Accessibility

Component	Offset
CLUSTERPMU	0xC40

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.29 CLUSTERPMU_PMINTENCLR, Cluster Performance Monitors Interrupt Enable Clear register

Disables the generation of interrupt requests on overflows from the Cycle Count Register, ext-CLUSTERPMU_PMCCNTR, and the event counters ext-CLUSTERPMU_PMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC60

Access type

See bit descriptions

Reset value

x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-329: ext_clusterpmu_pmintenclr bit assignments

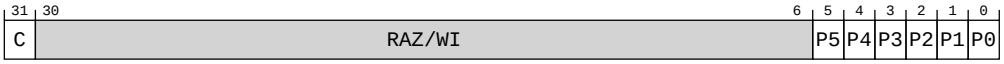


Table B-397: CLUSTERPMU_PMINTENCLR bit descriptions

Bits	Name	Description	Reset
[31]	C	ext-CLUSTERPMU_PMCCNTR overflow interrupt request disable bit. **0b0** When read, means the cycle counter overflow interrupt request is disabled. When written, has no effect. **0b1** When read, means the cycle counter overflow interrupt request is enabled. When written, disables the cycle count overflow interrupt request.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x

Bits	Name	Description	Reset
[4]	P4	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[3]	P3	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[2]	P2	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[1]	P1	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[0]	P0	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x

Accessibility

Component	Offset
CLUSTERPMU	0xC60

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.30 CLUSTERPMU_PMOVSCLR, Cluster Performance Monitors Overflow Flag Status Clear register

Contains the state of the overflow bit for the Cycle Count Register, ext-CLUSTERPMU_PMCCNTR, and each of the implemented event counters AArch64-PMEVCNTR<n>. Writing to this register clears these bits.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC80

Access type

See bit descriptions

Reset value

x000 0000 0000 0000 0000 0000 00xx xxxx

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-330: ext_clusterpmu_pmovsclr bit assignments

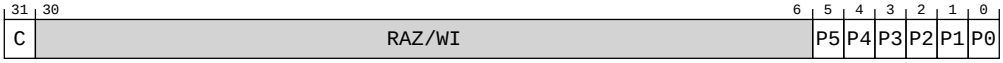


Table B-399: CLUSTERPMU_PMOVSCLR bit descriptions

Bits	Name	Description	Reset
[31]	C	Cycle counter overflow clear bit. **0b0** When read, means the cycle counter has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means the cycle counter has overflowed since this bit was last cleared. When written, clears the cycle counter overflow bit to 0.	x
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	x
[4]	P4	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	x
[3]	P3	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	x

Bits	Name	Description	Reset
[2]	P2	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	x
[1]	P1	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	x
[0]	P0	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	x

Accessibility

Component	Offset
CLUSTERPMU	0xC80

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.31 CLUSTERPMU_PMOVSSET, Cluster Performance Monitors Overflow Flag Status Set register

Sets the state of the overflow bit for the Cycle Count Register, ext-CLUSTERPMU_PMCCNTR, and each of the implemented event counters AArch64-PMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xCC0

Access type

See bit descriptions

Reset value

x000 0000 0000 0000 0000 0000 00xx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-331: ext_clusterpmu_pmovsset bit assignments

Table B-401: CLUSTERPMU_PMOVSSET bit descriptions

Bits	Name	Description	Reset
[31]	C	Cycle counter overflow set bit. **0b0** When read, means the cycle counter has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means the cycle counter has overflowed since this bit was last cleared. When written, sets the cycle counter overflow bit to 1.	x

Bits	Name	Description	Reset
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[4]	P4	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[3]	P3	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[2]	P2	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[1]	P1	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x

Bits	Name	Description	Reset
[0]	P0	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x

Accessibility

Component	Offset
CLUSTERPMU	0xCC0

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.32 CLUSTERPMU_PMCFGR, Cluster Performance Monitors Configuration Register

Contains PMU-specific configuration data.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xE00

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx 0xx0 0111 1111 0000 0110

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-332: ext_clusterpmu_pmcfgr bit assignments

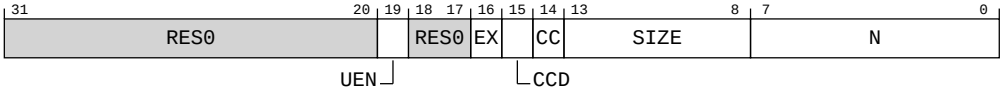


Table B-403: CLUSTERPMU_PMCFGR bit descriptions

Bits	Name	Description	Reset
[31:20]	**RES0**	Reserved	**RES0**
[19]	UEN	User-mode Enable Register supported. **0b0** User-mode Enable Register not supported.	0b0
[18:17]	**RES0**	Reserved	**RES0**
[16]	EX	Export supported. **0b0** ext-CLUSTERPMU_PMCR.X is **RES0**.	0b0
[15]	CCD	Cycle counter has prescale. **0b0** ext-CLUSTERPMU_PMCR.D is **RES0**.	0b0
[14]	CC	Dedicated cycle counter. **0b1** Dedicated cycle counter ext-CLUSTERPMU_PMCCNTR is supported.	0b1
[13:8]	SIZE	Size of counters, minus one. This field defines the size of the largest counter implemented by the Performance Monitors Unit. This field is used by software to determine the spacing of the counters in the memory-map. **0b111111** The largest counter is 64-bits. Counters are at doubleword-aligned addresses.	0b111111
[7:0]	N	Number of counters implemented in addition to the cycle counter, ext-CLUSTERPMU_PMCCNTR. **0b00000110** ext-CLUSTERPMU_PMCCNTR plus six event counters implemented.	0x06

Accessibility

Component	Offset
CLUSTERPMU	0xE00

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess()

RO

Otherwise

ERROR

B.2.4.33 CLUSTERPMU_PMCR, Cluster Performance Monitors Control Register

Provides details of the Performance Monitors implementation, including the number of counters implemented, and configures and controls the counters.

Configurations

This register is only partially mapped to the internal AArch64-IMP_CLUSTERPMCR System register. An external agent must use other means to discover the information held in AArch64-IMP_CLUSTERPMCR[31:11], such as accessing ext-CLUSTERPMU_PMCFGR and the ID registers.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xE04

Access type

inconsistent

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxx0 x000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-333: ext_clusterpmu_pmcr bit assignments

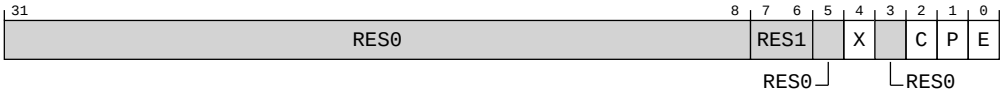


Table B-405: CLUSTERPMU_PMCR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:6]	**RES1**	Reserved	**RES1**
[5]	**RES0**	Reserved	**RES0**
[4]	X	This field enables the exporting of events over an event bus to another device. **0b0** Cluster PMU events are not exported externally.	0b0
[3]	**RES0**	Reserved	**RES0**
[2]	C	Cycle counter reset. This bit is WO. **0b0** No action. **0b1** Reset ext-CLUSTERPMU_PMCCNTR to zero. This bit is always **RAZ**. **Note:** Resetting ext-CLUSTERPMU_PMCCNTR does not change the cycle counter overflow bit.	0b0
[1]	P	Event counter reset. This bit is WO. **0b0** No action. **0b1** Reset all event counters, not including ext-CLUSTERPMU_PMCCNTR, to zero. This bit is always **RAZ**. **Note:** Resetting the event counters does not change the event counter overflow bits.	0b0
[0]	E	Enable. **0b0** All event counters, including ext-CLUSTERPMU_PMCCNTR, are disabled. **0b1** All event counters can be enabled by ext-CLUSTERPMU_PMCNTENSET. This bit is RW.	0b0

Accessibility

Component	Offset
CLUSTERPMU	0xE04

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.34 CLUSTERPMU_PMCEID0, Cluster Performance Monitors Common Event Identification register 0

Defines which common architectural events and common microarchitectural events are implemented, or counted, using PMU events in the range 0x0000 to 0x001F

When the value of a bit in the register is 1 the corresponding common event is implemented and counted.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xE20

Access type

See bit descriptions

Reset value

0010 0110 0000 0010 0000 0000 0000 0000

Bit descriptions

Figure B-334: ext_clusterpmu_pmceid0 bit assignments

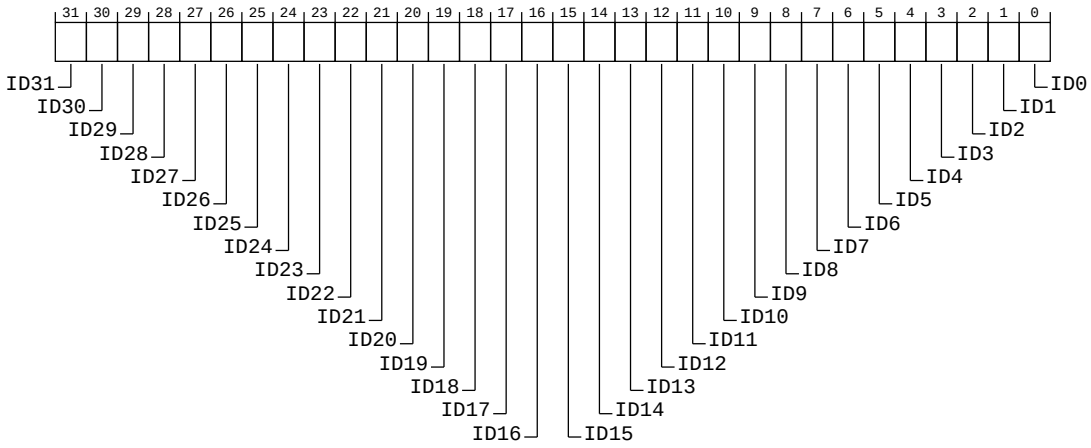


Table B-407: CLUSTERPMU_PMCEID0 bit descriptions

Bits	Name	Description	Reset
[31]	ID31	Common event `0x001F` implemented. **0b0** Event 0x001F not implemented.	`0b0`
[30]	ID30	Common event `0x001E` implemented. **0b0** CHAIN event not implemented.	`0b0`
[29]	ID29	Common event `0x001D` implemented. **0b1** BUS_CYCLES event implemented.	`0b1`
[28]	ID28	Common event `0x001C` implemented. **0b0** Event 0x001C not implemented.	`0b0`
[27]	ID27	Common event `0x001B` implemented. **0b0** Event 0x001B not implemented.	`0b0`
[26]	ID26	Common event `0x001A` implemented. **0b1** MEMORY_ERROR event implemented.	`0b1`
[25]	ID25	Common event `0x0019` implemented. **0b1** BUS_ACCESS event implemented.	`0b1`
[24]	ID24	Common event `0x0018` implemented. **0b0** Event 0x0018 not implemented.	`0b0`

Bits	Name	Description	Reset
[23]	ID23	Common event `0x0017` implemented. **0b0** Event 0x0017 not implemented.	`0b0`
[22]	ID22	Common event `0x0016` implemented. **0b0** Event 0x0016 not implemented.	`0b0`
[21]	ID21	Common event `0x0015` implemented. **0b0** Event 0x0015 not implemented.	`0b0`
[20]	ID20	Common event `0x0014` implemented. **0b0** Event 0x0014 not implemented.	`0b0`
[19]	ID19	Common event `0x0013` implemented. **0b0** Event 0x0013 not implemented.	`0b0`
[18]	ID18	Common event `0x0012` implemented. **0b0** Event 0x0012 not implemented.	`0b0`
[17]	ID17	Common event `0x0011` implemented. **0b1** CYCLES event implemented.	`0b1`
[16]	ID16	Common event `0x0010` implemented. **0b0** Event 0x0010 not implemented.	`0b0`
[15]	ID15	Common event `0x000F` implemented. **0b0** Event 0x000F not implemented.	`0b0`
[14]	ID14	Common event `0x000E` implemented. **0b0** Event 0x000E not implemented.	`0b0`
[13]	ID13	Common event `0x000D` implemented. **0b0** Event 0x000D not implemented.	`0b0`
[12]	ID12	Common event `0x000C` implemented. **0b0** Event 0x000C not implemented.	`0b0`
[11]	ID11	Common event `0x000B` implemented. **0b0** Event 0x000B not implemented.	`0b0`
[10]	ID10	Common event `0x000A` implemented. **0b0** Event 0x000A not implemented.	`0b0`

Bits	Name	Description	Reset
[9]	ID9	Common event `0x0009` implemented. **0b0** Event 0x0009 not implemented.	`0b0`
[8]	ID8	Common event `0x0008` implemented. **0b0** Event 0x0008 not implemented.	`0b0`
[7]	ID7	Common event `0x0007` implemented. **0b0** Event 0x0007 not implemented.	`0b0`
[6]	ID6	Common event `0x0006` implemented. **0b0** Event 0x0006 not implemented.	`0b0`
[5]	ID5	Common event `0x0005` implemented. **0b0** Event 0x0005 not implemented.	`0b0`
[4]	ID4	Common event `0x0004` implemented. **0b0** Event 0x0004 not implemented.	`0b0`
[3]	ID3	Common event `0x0003` implemented. **0b0** Event 0x0003 not implemented.	`0b0`
[2]	ID2	Common event `0x0002` implemented. **0b0** Event 0x0002 not implemented.	`0b0`
[1]	ID1	Common event `0x0001` implemented. **0b0** Event 0x0001 not implemented.	`0b0`
[0]	ID0	Common event `0x0000` implemented. **0b0** Event 0x0000 not implemented.	`0b0`

Accessibility

Component	Offset
CLUSTERPMU	0xE20

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess()

RO

Otherwise

ERROR

B.2.4.35 CLUSTERPMU_PMCEID1, Cluster Performance Monitors Common Event Identification register 1

Defines which common architectural events and common microarchitectural events are implemented, or counted, using PMU events in the range 0x020 to 0x03F.

When the value of a bit in the register is 1 the corresponding common event is implemented and counted.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xE24

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0001 1110 0000 0000

Bit descriptions

Figure B-335: ext_clusterpmu_pmceid1 bit assignments

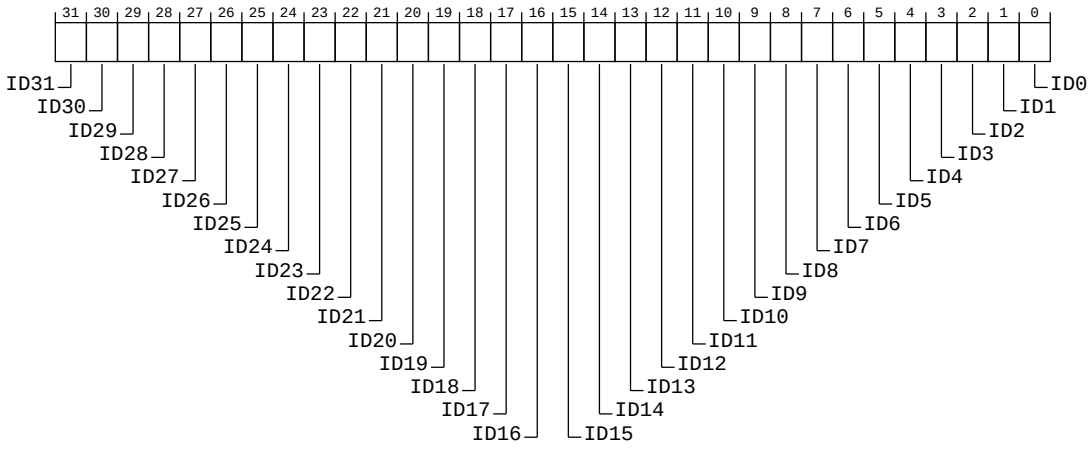


Table B-409: CLUSTERPMU_PMCEID1 bit descriptions

Bits	Name	Description	Reset
[31]	ID31	Common event `0x003F` implemented. **0b0** Event 0x003F not implemented.	`0b0`
[30]	ID30	Common event `0x003E` implemented. **0b0** Event 0x003E not implemented.	`0b0`
[29]	ID29	Common event `0x003D` implemented. **0b0** Event 0x003D not implemented.	`0b0`
[28]	ID28	Common event `0x003C` implemented. **0b0** Event 0x003C not implemented.	`0b0`
[27]	ID27	Common event `0x003B` implemented. **0b0** Event 0x003B not implemented.	`0b0`
[26]	ID26	Common event `0x003A` implemented. **0b0** Event 0x003A not implemented.	`0b0`
[25]	ID25	Common event `0x0039` implemented. **0b0** Event 0x0039 not implemented.	`0b0`
[24]	ID24	Common event `0x0038` implemented. **0b0** Event 0x0038 not implemented.	`0b0`
[23]	ID23	Common event `0x0037` implemented. **0b0** Event 0x0037 not implemented.	`0b0`
[22]	ID22	Common event `0x0036` implemented. **0b0** Event 0x0036 not implemented.	`0b0`
[21]	ID21	Common event `0x0035` implemented. **0b0** Event 0x0035 not implemented.	`0b0`
[20]	ID20	Common event `0x0034` implemented. **0b0** Event 0x0034 not implemented.	`0b0`
[19]	ID19	Common event `0x0033` implemented. **0b0** Event 0x0033 not implemented.	`0b0`

Bits	Name	Description	Reset
[18]	ID18	Common event `0x0032` implemented. **0b0** Event 0x0032 not implemented.	`0b0`
[17]	ID17	Common event `0x0031` implemented. **0b0** Event 0x0031 not implemented.	`0b0`
[16]	ID16	Common event `0x0030` implemented. **0b0** Event 0x0030 not implemented.	`0b0`
[15]	ID15	Common event `0x002F` implemented. **0b0** Event 0x002F not implemented.	`0b0`
[14]	ID14	Common event `0x002E` implemented. **0b0** Event 0x002E not implemented.	`0b0`
[13]	ID13	Common event `0x002D` implemented. **0b0** Event 0x002D not implemented.	`0b0`
[12]	ID12	Common event `0x002C` implemented. **0b1** L3D_CACHE_WB event implemented.	`0b1`
[11]	ID11	Common event `0x002B` implemented. **0b1** L3D_CACHE event implemented.	`0b1`
[10]	ID10	Common event `0x002A` implemented. **0b1** L3D_CACHE_REFILL event implemented.	`0b1`
[9]	ID9	Common event `0x0029` implemented. **0b1** L3D_CACHE_ALLOCATE event implemented.	`0b1`
[8]	ID8	Common event `0x0028` implemented. **0b0** Event 0x0028 not implemented.	`0b0`
[7]	ID7	Common event `0x0027` implemented. **0b0** Event 0x0027 not implemented.	`0b0`
[6]	ID6	Common event `0x0026` implemented. **0b0** Event 0x0026 not implemented.	`0b0`
[5]	ID5	Common event `0x0025` implemented. **0b0** Event 0x0025 not implemented.	`0b0`

Bits	Name	Description	Reset
[4]	ID4	Common event `0x0024` implemented. **0b0** Event 0x0024 not implemented.	`0b0`
[3]	ID3	Common event `0x0023` implemented. **0b0** Event 0x0023 not implemented.	`0b0`
[2]	ID2	Common event `0x0022` implemented. **0b0** Event 0x0022 not implemented.	`0b0`
[1]	ID1	Common event `0x0021` implemented. **0b0** Event 0x0021 not implemented.	`0b0`
[0]	ID0	Common event `0x0020` implemented. **0b0** Event 0x0020 not implemented.	`0b0`

Accessibility

Component	Offset
CLUSTERPMU	0xE24

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess()

RO

Otherwise

ERROR

B.2.4.36 CLUSTERPMU_PMCEID2, Cluster Performance Monitors Common Event Identification register 2

Defines which common architectural events and common microarchitectural events are implemented, or counted, using PMU events in the range 0x4000 to 0x401F.

When the value of a bit in the register is 1 the corresponding common event is implemented and counted.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xE28

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-336: ext_clusterpmu_pmceid2 bit assignments

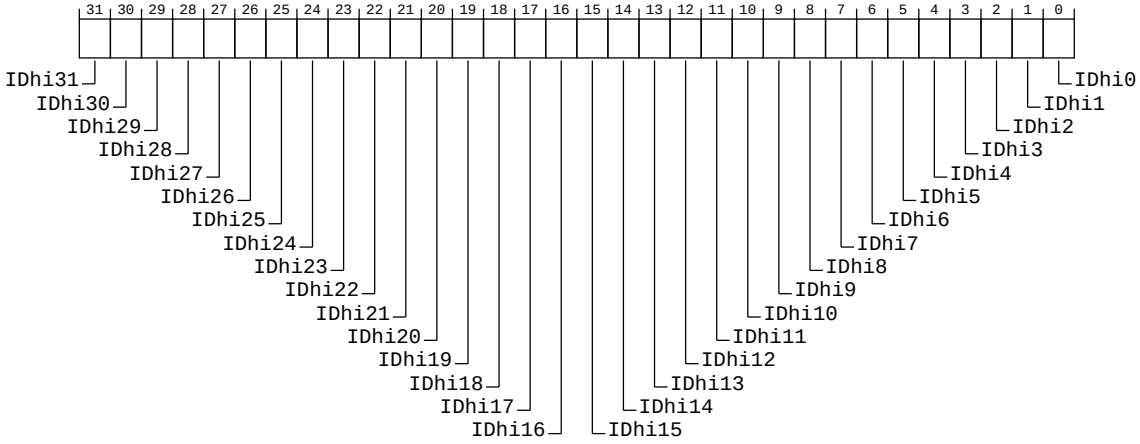


Table B-411: CLUSTERPMU_PMCEID2 bit descriptions

Bits	Name	Description	Reset
[31]	IDhi31	Common event `0x401F` implemented. **0b0** Event 0x401F not implemented.	`0b0`
[30]	IDhi30	Common event `0x401E` implemented. **0b0** Event 0x401E not implemented.	`0b0`
[29]	IDhi29	Common event `0x401D` implemented. **0b0** Event 0x401D not implemented.	`0b0`
[28]	IDhi28	Common event `0x401C` implemented. **0b0** Event 0x401C not implemented.	`0b0`

Bits	Name	Description	Reset
[27]	IDhi27	Common event `0x401B` implemented. **0b0** Event 0x401B not implemented.	`0b0`
[26]	IDhi26	Common event `0x401A` implemented. **0b0** Event 0x401A not implemented.	`0b0`
[25]	IDhi25	Common event `0x4019` implemented. **0b0** Event 0x4019 not implemented.	`0b0`
[24]	IDhi24	Common event `0x4018` implemented. **0b0** Event 0x4018 not implemented.	`0b0`
[23]	IDhi23	Common event `0x4017` implemented. **0b0** Event 0x4017 not implemented.	`0b0`
[22]	IDhi22	Common event `0x4016` implemented. **0b0** Event 0x4016 not implemented.	`0b0`
[21]	IDhi21	Common event `0x4015` implemented. **0b0** Event 0x4015 not implemented.	`0b0`
[20]	IDhi20	Common event `0x4014` implemented. **0b0** Event 0x4014 not implemented.	`0b0`
[19]	IDhi19	Common event `0x4013` implemented. **0b0** Event 0x4013 not implemented.	`0b0`
[18]	IDhi18	Common event `0x4012` implemented. **0b0** Event 0x4012 not implemented.	`0b0`
[17]	IDhi17	Common event `0x4011` implemented. **0b0** Event 0x4011 not implemented.	`0b0`
[16]	IDhi16	Common event `0x4010` implemented. **0b0** Event 0x4010 not implemented.	`0b0`
[15]	IDhi15	Common event `0x400F` implemented. **0b0** Event 0x400F not implemented.	`0b0`
[14]	IDhi14	Common event `0x400E` implemented. **0b0** Event 0x400E not implemented.	`0b0`

Bits	Name	Description	Reset
[13]	IDhi13	Common event `0x400D` implemented. **0b0** Event 0x400D not implemented.	`0b0`
[12]	IDhi12	Common event `0x400C` implemented. **0b0** Event 0x400C not implemented.	`0b0`
[11]	IDhi11	Common event `0x400B` implemented. **0b0** Event 0x400B not implemented.	`0b0`
[10]	IDhi10	Common event `0x400A` implemented. **0b0** Event 0x400A not implemented.	`0b0`
[9]	IDhi9	Common event `0x4009` implemented. **0b0** Event 0x4009 not implemented.	`0b0`
[8]	IDhi8	Common event `0x4008` implemented. **0b0** Event 0x4008 not implemented.	`0b0`
[7]	IDhi7	Common event `0x4007` implemented. **0b0** Event 0x4007 not implemented.	`0b0`
[6]	IDhi6	Common event `0x4006` implemented. **0b0** Event 0x4006 not implemented.	`0b0`
[5]	IDhi5	Common event `0x4005` implemented. **0b0** Event 0x4005 not implemented.	`0b0`
[4]	IDhi4	Common event `0x4004` implemented. **0b0** Event 0x4004 not implemented.	`0b0`
[3]	IDhi3	Common event `0x4003` implemented. **0b0** Event 0x4003 not implemented.	`0b0`
[2]	IDhi2	Common event `0x4002` implemented. **0b0** Event 0x4002 not implemented.	`0b0`
[1]	IDhi1	Common event `0x4001` implemented. **0b0** Event 0x4001 not implemented.	`0b0`
[0]	IDhi0	Common event `0x4000` implemented. **0b0** Event 0x4000 not implemented.	`0b0`

Accessibility

Component	Offset
CLUSTERPMU	0xE28

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess()

RO

Otherwise

ERROR

B.2.4.37 CLUSTERPMU_PMCEID3, Cluster Performance Monitors Common Event Identification register 3

Defines which common architectural events and common microarchitectural events are implemented, or counted, using PMU events in the range 0x4020 to 0x403F.

When the value of a bit in the register is 1 the corresponding common event is implemented and counted.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xE2C

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-337: ext_clusterpmu_pmceid3 bit assignments

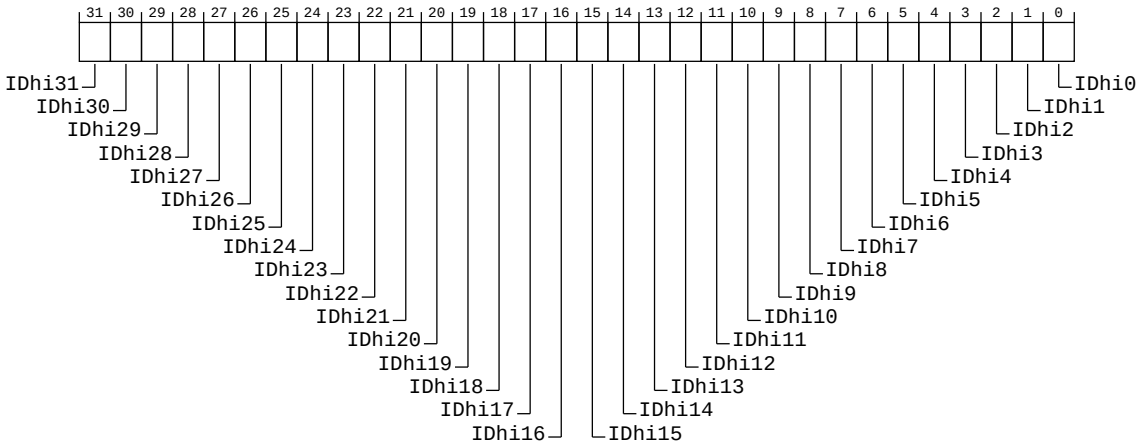


Table B-413: CLUSTERPMU_PMCEID3 bit descriptions

Bits	Name	Description	Reset
[31]	IDhi31	Common event `0x403F` implemented. **0b0** Event 0x403F not implemented.	`0b0`
[30]	IDhi30	Common event `0x403E` implemented. **0b0** Event 0x403E not implemented.	`0b0`
[29]	IDhi29	Common event `0x403D` implemented. **0b0** Event 0x403D not implemented.	`0b0`
[28]	IDhi28	Common event `0x403C` implemented. **0b0** Event 0x403C not implemented.	`0b0`
[27]	IDhi27	Common event `0x403B` implemented. **0b0** Event 0x403B not implemented.	`0b0`
[26]	IDhi26	Common event `0x403A` implemented. **0b0** Event 0x403A not implemented.	`0b0`
[25]	IDhi25	Common event `0x4039` implemented. **0b0** Event 0x4039 not implemented.	`0b0`
[24]	IDhi24	Common event `0x4038` implemented. **0b0** Event 0x4038 not implemented.	`0b0`

Bits	Name	Description	Reset
[23]	IDhi23	Common event `0x4037` implemented. **0b0** Event 0x4037 not implemented.	`0b0`
[22]	IDhi22	Common event `0x4036` implemented. **0b0** Event 0x4036 not implemented.	`0b0`
[21]	IDhi21	Common event `0x4035` implemented. **0b0** Event 0x4035 not implemented.	`0b0`
[20]	IDhi20	Common event `0x4034` implemented. **0b0** Event 0x4034 not implemented.	`0b0`
[19]	IDhi19	Common event `0x4033` implemented. **0b0** Event 0x4033 not implemented.	`0b0`
[18]	IDhi18	Common event `0x4032` implemented. **0b0** Event 0x4032 not implemented.	`0b0`
[17]	IDhi17	Common event `0x4031` implemented. **0b0** Event 0x4031 not implemented.	`0b0`
[16]	IDhi16	Common event `0x4030` implemented. **0b0** Event 0x4030 not implemented.	`0b0`
[15]	IDhi15	Common event `0x402F` implemented. **0b0** Event 0x402F not implemented.	`0b0`
[14]	IDhi14	Common event `0x402E` implemented. **0b0** Event 0x402E not implemented.	`0b0`
[13]	IDhi13	Common event `0x402D` implemented. **0b0** Event 0x402D not implemented.	`0b0`
[12]	IDhi12	Common event `0x402C` implemented. **0b0** Event 0x402C not implemented.	`0b0`
[11]	IDhi11	Common event `0x402B` implemented. **0b0** Event 0x402B not implemented.	`0b0`
[10]	IDhi10	Common event `0x402A` implemented. **0b0** Event 0x402A not implemented.	`0b0`

Bits	Name	Description	Reset
[9]	IDhi9	Common event `0x4029` implemented. **0b0** Event 0x4029 not implemented.	`0b0`
[8]	IDhi8	Common event `0x4028` implemented. **0b0** Event 0x4028 not implemented.	`0b0`
[7]	IDhi7	Common event `0x4027` implemented. **0b0** Event 0x4027 not implemented.	`0b0`
[6]	IDhi6	Common event `0x4026` implemented. **0b0** Event 0x4026 not implemented.	`0b0`
[5]	IDhi5	Common event `0x4025` implemented. **0b0** Event 0x4025 not implemented.	`0b0`
[4]	IDhi4	Common event `0x4024` implemented. **0b0** Event 0x4024 not implemented.	`0b0`
[3]	IDhi3	Common event `0x4023` implemented. **0b0** Event 0x4023 not implemented.	`0b0`
[2]	IDhi2	Common event `0x4022` implemented. **0b0** Event 0x4022 not implemented.	`0b0`
[1]	IDhi1	Common event `0x4021` implemented. **0b0** Event 0x4021 not implemented.	`0b0`
[0]	IDhi0	Common event `0x4020` implemented. **0b0** Event 0x4020 not implemented.	`0b0`

Accessibility

Component	Offset
CLUSTERPMU	0xE2C

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess()

RO

Otherwise

ERROR

B.2.4.38 CLUSTERPMU_PMCLAIMSET, Cluster Performance Monitors Claim Set register

Used by software to set CLAIM bits to 1.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFA0

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-338: ext_clusterpmu_pmclaimset bit assignments

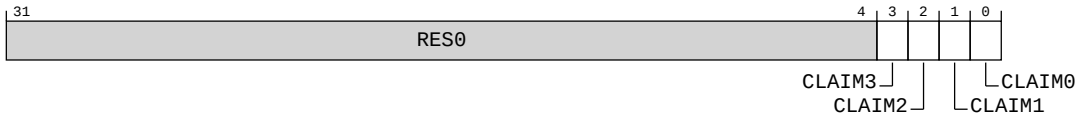


Table B-415: CLUSTERPMU_PMCLAIMSET bit descriptions

Bits	Name	Description	Reset
[31:4]	RES0	Reserved	RES0
[3]	CLAIM3	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x

Bits	Name	Description	Reset
[2]	CLAIM2	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x
[1]	CLAIM1	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x
[0]	CLAIM0	CLAIM tag set bit. **0b0** No action. **0b1** Indirectly set claim bit to 1.	x

Accessibility

Component	Offset
CLUSTERPMU	0xFA0

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.39 CLUSTERPMU_PMCLAIMCLR, Cluster Performance Monitors Claim Clear register

Used by software to read the values of the CLAIM bits, and to clear these bits to 0.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFA4

Access type

See bit descriptions

Reset value

XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-339: ext_clusterpmu_pmclaimclr bit assignments

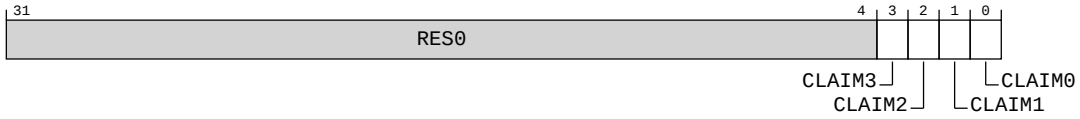


Table B-417: CLUSTERPMU_PMCLAIMCLR bit descriptions

Bits	Name	Description	Reset
[31:4]	RES0	Reserved	RES0
[3]	CLAIM3	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x
[2]	CLAIM2	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x
[1]	CLAIM1	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x

Bits	Name	Description	Reset
[0]	CLAIM0	CLAIM tag clear bit. **0b0** No action. **0b1** Indirectly clear claim bit to 0.	x

Accessibility

Component	Offset
CLUSTERPMU	0xFA4

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RW

Otherwise

ERROR

B.2.4.40 CLUSTERPMU_PMDEVAFF0, Cluster Performance Monitors Device Affinity register 0

Allows a debugger to determine which PE in a multiprocessor system the Performance Monitor component relates to.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFA8

Access type

See bit descriptions

Reset value

x0xx xxx0 xxxx xxxx 1000 0000 1000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-340: ext_clusterpmu_pmdevaff0 bit assignments

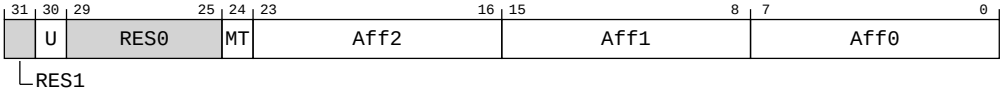


Table B-419: CLUSTERPMU_PMDEVAFF0 bit descriptions

Bits	Name	Description	Reset
[31]	**RES1**	Reserved	**RES1**
[30]	U	Uniprocessor/Multiprocessor system. **0b0** Processor is part of a multiprocessor system.	0b0
[29:25]	**RES0**	Reserved	**RES0**
[24]	MT	Indicates whether the lowest level of affinity consists of logical PEs that are implemented using a multithreading type approach. **0b0** Performance of PEs at the lowest affinity level is largely independent.	0b0
[23:16]	Aff2	Affinity level 2. Value read from the CFGMPIDRAFF2 configuration pins.	8{x}
[15:8]	Aff1	Affinity level 1. **0b10000000** Affinity with all cores in cluster.	0x80
[7:0]	Aff0	Affinity level 0. **0b10000000** Affinity with all core threads in cluster.	0x80

Accessibility

Component	Offset
CLUSTERPMU	0xFA8

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.41 CLUSTERPMU_PMDEVAFF1, Cluster Performance Monitors Device Affinity register 1

Allows a debugger to determine which PE in a multiprocessor system the Performance Monitor component relates to.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFAC

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-341: ext_clusterpmu_pmdevaff1 bit assignments

Table B-421: CLUSTERPMU_PMDEVAFF1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	Aff3	Affinity level 3. Value read from the CFGMPIDRAFF3 configuration pins.	8{x}

Accessibility

Component	Offset
CLUSTERPMU	0xFAC

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.42 CLUSTERPMU_PMAUTHSTATUS, Cluster Performance Monitors Authentication Status register

Provides information about the state of the authentication interface for Performance Monitors.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFB8

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-342: ext_clusterpmu_pmauthstatus bit assignments

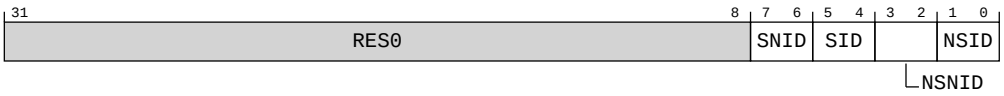


Table B-423: CLUSTERPMU_PMAUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:6]	SNID	Secure Non-invasive Debug. ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled(). This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug. **0b10** Secure invasive debug disabled. ExternalSecureInvasiveDebugEnabled() == FALSE. **0b11** Secure invasive debug enabled. ExternalSecureInvasiveDebugEnabled() == TRUE.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug. **0b00** Debug level is not supported.	0b00
[1:0]	NSID	Non-secure Invasive Debug. **0b00** Debug level is not supported.	0b00

Accessibility

Component	Offset
CLUSTERPMU	0xFB8

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.43 CLUSTERPMU_PMDEVARCH, Cluster Performance Monitors Device Architecture register

Identifies the programmers' model architecture of the Performance Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFBC

Access type

See bit descriptions

Reset value

0100 0111 0111 0000 0010 1010 0001 0110

Bit descriptions

Figure B-343: ext_clusterpmu_pmdevarch bit assignments

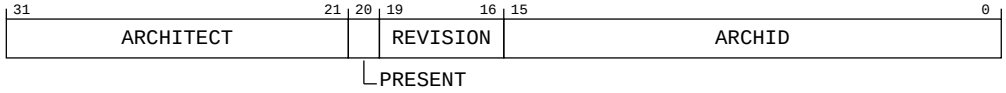


Table B-425: CLUSTERPMU_PMDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **0b01000111011** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. **0b1** DEVARCH information present.	0b1
[19:16]	REVISION	Revision. **0b0000** Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. **0b0010101000010110** Processor Performance Monitor (PMU) architecture PMUv3.	0x2A16

Accessibility

Component	Offset
CLUSTERPMU	0xFBC

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.44 CLUSTERPMU_PMDEVID, Cluster Performance Monitors Device ID register

Provides information about features of the Performance Monitors implementation.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFC8

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-344: ext_clusterpmu_pmdevid bit assignments

Table B-427: CLUSTERPMU_PMDEVID bit descriptions

Bits	Name	Description	Reset
[31:4]	**RES0**	Reserved	**RES0**
[3:0]	PCSample	Indicates the level of PC Sample-based Profiling support using Performance Monitors registers. **0b0000** PC Sample-based Profiling Extension is not implemented for the Cluster PMU.	0b0000

Accessibility

Component	Offset
CLUSTERPMU	0xFC8

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.45 CLUSTERPMU_PMDEVTYPE, Cluster Performance Monitors Device Type register

Indicates to a debugger that this component is part of a processor performance monitor interface.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFCC

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0001 0110

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-345: ext_clusterpmu_pmdevtype bit assignments

Table B-429: CLUSTERPMU_PMDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SUB	Subtype. **0b0001** Associated with a processor.	0b0001
[3:0]	MAJOR	Major type. **0b0110** Performance Monitor.	0b0110

Accessibility

Component	Offset
CLUSTERPMU	0xFCC

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.46 CLUSTERPMU_PMPIDR4, Cluster Performance Monitors Peripheral Identification Register 4

Provides information to identify a Performance Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFD0

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-346: ext_clusterpmu_pmpidr4 bit assignments

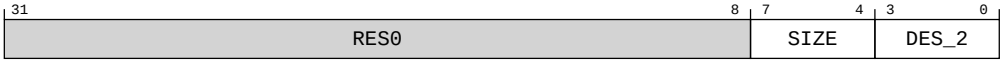


Table B-431: CLUSTERPMU_PMPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

Component	Offset
CLUSTERPMU	0xFD0

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.47 CLUSTERPMU_PMPIDR0, Cluster Performance Monitors Peripheral Identification Register 0

Provides information to identify a Performance Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFE0

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1110 1000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-347: ext_clusterpmu_pmpidr0 bit assignments

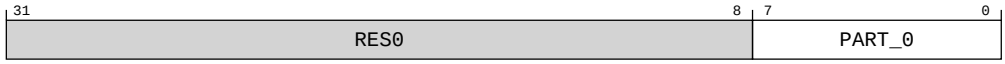


Table B-433: CLUSTERPMU_PMPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **0b11101000** DSU-110 Cluster PMU. Bits [7:0] of part number 0x4E8.	0xE8

Accessibility

Component	Offset
CLUSTERPMU	0xFE0

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.48 CLUSTERPMU_PMPIDR1, Cluster Performance Monitors Peripheral Identification Register 1

Provides information to identify a Performance Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFE4

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-348: ext_clusterpmu_pmpidr1 bit assignments

Table B-435: CLUSTERPMU_PMPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0100** DSU-110 Cluster PMU. Bits [11:8] of part number 0x4E8.	0b0100

Accessibility

Component	Offset
CLUSTERPMU	0xFE4

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.49 CLUSTERPMU_PMPIDR2, Cluster Performance Monitors Peripheral Identification Register 2

Provides information to identify a Performance Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFE8

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0110 1011

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-349: ext_clusterpmu_pmpidr2 bit assignments

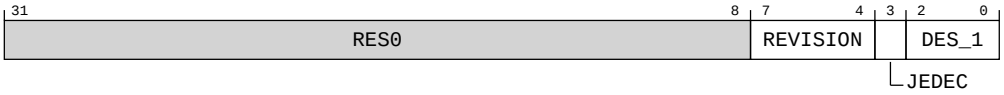


Table B-437: CLUSTERPMU_PMPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4. **0b0101** Component major revision 5. **0b0110** Component major revision 6. For DSU-110: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r1p0. • Major revision 2 corresponds to r2p0. • Major revision 3 corresponds to r2p1. • Major revision 4 corresponds to r3p0. • Major revision 5 corresponds to r3p1. • Major revision 6 corresponds to r4p0.	0b0110
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

Component	Offset
CLUSTERPMU	0xFE8

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.50 CLUSTERPMU_PMPIDR3, Cluster Performance Monitors Peripheral Identification Register 3

Provides information to identify a Performance Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFEC

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-350: ext_clusterpmu_pmpidr3 bit assignments

Table B-439: CLUSTERPMU_PMPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0. **0b0001** Component minor revision 1. **0b0010** Component minor revision 2. **0b0011** Component minor revision 3. **0b0100** Component minor revision 4.	0b0000
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

Accessibility

Component	Offset
CLUSTERPMU	0xFEC

This interface is accessible as follows:

When IsCorePowered()
RO

Otherwise
ERROR

B.2.4.51 CLUSTERPMU_PMCIDR0, Cluster Performance Monitors Component Identification Register 0

Provides information to identify a Performance Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width
32

Component
CLUSTERPMU

Register offset

0xFF0

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-351: ext_clusterpmu_pmcidr0 bit assignments

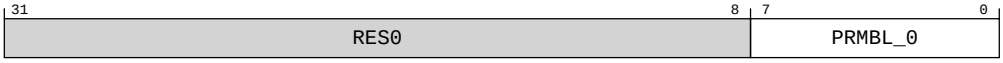


Table B-441: CLUSTERPMU_PMCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	Preamble. **0b00001101** CoreSight component identification preamble.	0x0D

Accessibility

Component	Offset
CLUSTERPMU	0xFF0

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.52 CLUSTERPMU_PMCIDR1, Cluster Performance Monitors Component Identification Register 1

Provides information to identify a Performance Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFF4

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-352: ext_clusterpmu_pmcidr1 bit assignments

Table B-443: CLUSTERPMU_PMCIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	Component class. **0b1001** CoreSight debug component.	0b1001
[3:0]	PRMBL_1	Preamble. **0b0000** CoreSight component identification preamble.	0b0000

Accessibility

Component	Offset
CLUSTERPMU	0xFF4

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.53 CLUSTERPMU_PMCIDR2, Cluster Performance Monitors Component Identification Register 2

Provides information to identify a Performance Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFF8

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-353: ext_clusterpmu_pmcidr2 bit assignments

Table B-445: CLUSTERPMU_PMCIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	Preamble. **0b00000101** CoreSight component identification preamble.	0x05

Accessibility

Component	Offset
CLUSTERPMU	0xFF8

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.54 CLUSTERPMU_PMCIDR3, Cluster Performance Monitors Component Identification Register 3

Provides information to identify a Performance Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xFFC

Access type

See bit descriptions

Reset value

xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001

Note

Where the reset reads xxxx, see individual bits

Bit descriptions

Figure B-354: ext_clusterpmu_pmcidr3 bit assignments

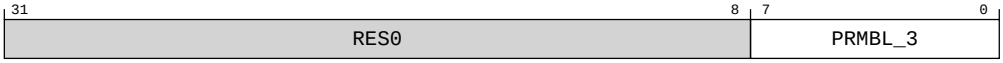


Table B-447: CLUSTERPMU_PMCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Preamble. **0b10110001** CoreSight component identification preamble.	0xB1

Accessibility

Component	Offset
CLUSTERPMU	0xFFC

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

Appendix C Document revisions

This appendix records the changes between released issues of this document.

C.1 Revisions

Changes between released issues of this book are summarized in tables.

Table C-1: Issue 0000-01

Change	Location
First confidential alpha release for r0p0	-

Table C-2: Differences between Issue 0000-01 and Issue 0000-02

Change	Location
First confidential beta release for r0p0	-
Added new chapter	7. L3 cache on page 99
Added new chapter	8. CHI master interface on page 107
Added new chapter	9. AXI master interface on page 119
Added new chapter	10. ACP slave interface on page 126
Added new chapter	11. AXI or CHI master peripheral port on page 136
Added new chapter	12. RAS extension support on page 152
Added new chapter	13. Utility bus on page 164
Added new chapter	15. Debug on page 170
Added new chapter	16. ROM tables on page 190
Added new chapter	17. Performance Monitors Extension support on page 203
Added new registers	A.1 AArch64 generic system control register summary on page 209
Added new registers	A.2 AArch64 Performance Monitors register summary on page 258
Added new registers	A.3 AArch64 RAS register summary on page 308
Added new registers	B.1.1 External cluster register summary on page 344
Added new registers	B.1.2 External MPAM register summary on page 367
Added new registers	B.1.3 External cluster RAS register summary on page 385
Added new registers	B.1.4 External cluster PPU register summary on page 431
Added new registers	B.1.5 External core PPU register summary on page 486
Added new registers	B.2.1 External CTI register summary on page 536
Added new registers	B.2.2 External cluster ROM register summary on page 624
Added new registers	B.2.3 External debug ROM register summary on page 715
Added new registers	B.2.4 External cluster PMU register summary on page 754

Table C-3: Differences between Issue 0000-02 and Issue 0000-03

Change	Location
First confidential limited access release for r0p0	-
Added notes about unsupported features for LAC release	Throughout document, for the chapter to which it applies
Minor additions and clarifications	2. The DynamIQ Shared Unit-110 on page 17
Updated clock signals to include PPUCLK	Throughout document
Updated interface diagram to clarify interfaces	3. Technical overview on page 33
Clarified how PBHA information is sent out	3.4.1 Page-Based Hardware Attribute on page 42
Updated clocks, clock domains, and reset domain diagrams and descriptions to include PPUCLK	4. Clocks and resets on page 44
Clarified the description of the reset signals	4. Clocks and resets on page 44
Updated power domains and voltage domain diagrams to show CPU bridges	5. Power management on page 50
Various clarifications on the cluster power modes	5. Power management on page 50
Added scenarios on powering down L3 RAMs	5.4.1 L3 cache RAM powerdown on page 57
Various minor clarifications to L3 cache slice powerdown	5.4.2 L3 cache slice powerdown on page 61
Various minor clarifications to cluster power states	5.7 Cluster PPU mode transitions on page 64
Various minor clarifications to core power modes	5.8 Core PPU modes on page 70
Clarifications on the operation of the Power Policy Unit	6.2 Power policy unit operation on page 80
Updates and clarifications to the *Power Policy Unit* (PPU) programming sequences	6.6 Programming sequences for the cluster and the core on page 90
Updates and clarifications	6.7 Explicit reset of cluster and cores and debug recovery on page 92
Added new section on usage of Full retention mode with static mode policy	6.10 Core Full retention mode and static mode restrictions on page 97
Various clarifications throughout chapter	6. Power and reset control with Power Policy Units on page 78
Minor edits	7. L3 cache on page 99
Updated CHI features	8.2 CHI features on page 110
Clarifications and updates	8.3 CHI configurations on page 111
Updated CHI read and write issuing capabilities and added descriptions	8.4 Attributes of the CHI master interface on page 112
Updated snoop acceptance capability	8.5 CHI channel properties on page 113
Updated CHI read and write transactions	8.6 CHI transactions on page 114
Updated AXI master port properties	9.2 AXI master port interface properties on page 120
Updated AXI read and write issuing capabilities and added descriptions	9.4 AXI 256-bit master interface attributes on page 121
Various updates and clarifications	11. AXI or CHI master peripheral port on page 136
Moved information about deadlock conditions on peripheral port with ACP over to the <cite CIM>	9. AXI master interface on page 119
Minor edits	12. RAS extension support on page 152
Minor updates and clarifications	13. Utility bus on page 164
Minor updates and clarifications	15. Debug on page 170
Moved all AArch64 register descriptions into a new appendix	A. AArch64 registers on page 209
Moved all External register descriptions into a new appendix	B. External registers on page 344

Change	Location
Updated various register descriptions and register summary tables	A. AArch64 registers on page 209 and B. External registers on page 344

Table C-4: Differences between Issue 0000-03 and Issue 0100-04

Change	Location
First confidential Limited Access release for r1p0	-
Minor clarifications	2. The DynamIQ Shared Unit-110 on page 17
Minor clarifications	4. Clocks and resets on page 44
Updated minimum CHI address width	8.2 CHI features on page 110
Added note to qualify an LPID for TLB translation table walks	8.6 CHI transactions on page 114
New section added on double error reporting	12.4.3 Double error reporting on page 157
Various clarifications and minor updates	12. RAS extension support on page 152
Added acceptance capabilities for utility bus	13.1 Utility bus accesses on page 164
Updated various register descriptions	A. AArch64 registers on page 209 and B. External registers on page 344
Updated core and cluster *Power Policy Unit* (PPU) register descriptions	B. External registers on page 344
Updated *Reliability, Availability, and Serviceability* (RAS) PIDRn register	B. External registers on page 344
Updated *Performance Monitoring Unit* (PMU) EVCNTSRN offset	B. External registers on page 344
Updated cluster PPEND description	B. External registers on page 344
Updated ROM entry OFFSET field description	B. External registers on page 344
Updated MPAM CPBM field width	B. External registers on page 344
Updated RAS CIDR1 CLASS field value	B. External registers on page 344
Updated accessor values to RAS System registers	B. External registers on page 344

Table C-5: Differences between Issue 0100-04 and Issue 0200-05

Change	Location
First confidential Early Access release for r2p0	-
Minor updates and clarifications	Throughout book
Updated list of features not supported for this release	2. The DynamIQ Shared Unit-110 on page 17 and throughout book
Updates and clarifications	4. Clocks and resets on page 44
Updates and clarifications	5. Power management on page 50
Updates and clarifications	6. Power and reset control with Power Policy Units on page 78
Added in CoreSight component identification	15.10 CoreSight component identification on page 188
Added new debug system address map	16.1 Debug system address map on page 190
Updated CLUSTERACTLR description	A.1.4 IMP_CLUSTERACTLR_EL1, Cluster Auxiliary Control Register on page 220 and B.1.1.12 CLUSTERACTLR, Cluster Auxiliary Control Register on page 361
Updated cluster PPU_DISR description	B.1.4.4 PPU_DISR, Device Interface Input Current Status Register on page 439

Change	Location
Updated CLUSTERECTLR register fields	A.1.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register on page 222 and B.1.1.13 CLUSTERECTLR, Cluster Extended Control Register on page 362
Updated MPAM register descriptions	B.1.2 External MPAM register summary on page 367
Removed most of language regarding IMPLEMENTATION DEFINED choices and added in actual behavior where applicable.	A. AArch64 registers on page 209 and B. External registers on page 344
Updated register descriptions to clarify fields that are supported.	A.3 AArch64 RAS register summary on page 308 and B.1.3 External cluster RAS register summary on page 385
Updated reset values for the AArch64 registers	A. AArch64 registers on page 209
Updated reset values for the Ext registers	B. External registers on page 344

Table C-6: Differences between Issue 0200-05 and Issue 0201-06

Change	Location
First confidential Early Access release for r2p1	-
Clarification about L3 cache allocations	7. L3 cache on page 99
Updated the values for the percentage of total outstanding transactions that each master port can support	8.4 Attributes of the CHI master interface on page 112
Updated the read and write issuing capabilities for AXI 256-bit master port	9.4 AXI 256-bit master interface attributes on page 121
Updated the values for the percentage of total outstanding transactions that each master port can support	9.4 AXI 256-bit master interface attributes on page 121
Clarifications and corrections to MEMORY_ERROR, L3D_CACHE_RD, and L3D_CACHE_REFILL_RD PMU events	17.2 PMU events on page 203
Updated AArch64 register ID values to match the r2p1 product revision	A. AArch64 registers on page 209
Updated Ext register ID values to match the r2p1 product revision	B. External registers on page 344

Table C-7: Differences between Issue 0201-06 and Issue 0300-07

Change	Location
First confidential Early Access release for r3p0	-
Updated section for 12 core cluster support	2. The DynamIQ Shared Unit-110 on page 17
Updated section for 12 core cluster support	2.1 DynamIQ Shared Unit-110 features on page 18
Updated section for 12 core cluster support	2.2 DynamIQ Shared Unit-110 configuration parameters on page 20
Updated section for 12 core cluster support	2.3.1 What is a complex? on page 25
Updated description for instance numbering of a complex and *Processing Element* (PE)	2.7 Core, complex, and processing element numbering on page 30
Updated section for 12 core cluster support	3.1 DynamIQ cluster components on page 33
Updated terminology to replace portion by power portion where appropriate	5.4.1.1 Setting automatic L3 cache power portion control on page 58
Updated terminology to replace portion by power portion where appropriate	5.4.1.2 Setting CLUSTERPWRCTLR_EL1.PRTNRQ power portion control on page 59
Updated terminology to replace portion by power portion where appropriate	5.4.1.4 Calculating values for threshold registers on page 60

Change	Location
Updated footnote to table Theodul DSU RAM power states for cache slices 1 to N for ONE slice operation. Updated terminology to replace portion by power portion where appropriate.	5.6 Power states for the cluster RAM instances on page 63
Updated description for functional retention mode	5.8 Core PPU modes on page 70
Updated description for functional retention mode	5.8.1 Core PPU mode transitions on page 71
Major updates and clarifications	7.3 L3 cache partitioning on page 100
Updated terminology to replace portion by power portion where appropriate	7.6 Cache slices and power portions on page 105
Updated section for 12 core cluster support	11.10 Read and write capabilities and transaction ID encoding on page 147
Added new section to describe possible deadlock conditions	11.11 Peripheral port and ACP interface usage on page 149
Major updates and clarifications	12.5 Error injection on page 158
Added new section	15.1 Cache debug on page 171
Updated section for 12 core cluster support	16.1 Debug system address map on page 190
Updated to show what registers are supported for Direct connect	A.1 AArch64 generic system control register summary on page 209
Updates to register to allow more horizontal and vertical register slices	A.1.1 IMP_CLUSTERCFR_EL1, Cluster Configuration Register on page 210
Updated register to indicate r3p0 product revision	A.1.2 IMP_CLUSTERIDR_EL1, Cluster Main Revision Register on page 216
Updated the UPTH1 bit field description	A.1.12 IMP_CLUSTERL3UPTH1_EL1, Cluster L3 Upsize Threshold1 Register on page 240
Updates to register to allow QoS field for prefetches	A.1.13 IMP_CLUSTERBUSQOS_EL1, Cluster Bus QoS Control Register on page 242
New register added to support 12 core cluster	A.1.18 IMP_CLUSTERCFR2_EL1, Cluster Configuration Register 2 on page 251
Updated to show what registers are supported for Direct connect	A.2 AArch64 Performance Monitors register summary on page 258
Changed register summary topic text, to include statement about Direct connect	A.3 AArch64 RAS register summary on page 308
Changed register summary topic text, to include statement about Direct connect	B.1.1 External cluster register summary on page 344
Updated register to indicate r3p0 product revision	B.1.1.1 CLUSTERIDR, Cluster Main Revision Register on page 345
Changed register summary topic text, to include statement about Direct connect	B.1.2 External MPAM register summary on page 367
Updated register to indicate r3p0 product revision	B.1.2.3 MPAMF_IIDR, MPAM Implementation Identification Register on page 371
Changed register summary topic text, to include statement about Direct connect	B.1.3 External cluster RAS register summary on page 385
Added reset value for ED bit field	B.1.3.2 CLUSTERRAS_ERR0CTLR, Error Record Control Register on page 389
Register updated to indicate r3p0 product revision	B.1.3.13 CLUSTERRAS_ERRIIDR, Implementation Identification Register on page 413
Register updated to indicate p0 (in r3p0 product revision)	B.1.3.24 CLUSTERRAS_ERRPIDR3, Peripheral Identification Register 3 on page 426
Changed register summary topic text, to include statement about Direct connect	B.1.4 External cluster PPU register summary on page 431

Change	Location
Updated LOCK_EN bit field description	B.1.4.1 PPU_PWPR, Power Policy Register on page 433
Updated OPSTATUS bit field description	B.1.4.3 PPU_PWSR, Power Status Register on page 436
Changed register summary topic text, to include statement about Direct connect	B.1.5 External core PPU register summary on page 486
Updated LOCK_EN bit field description	B.1.5.1 PPU_PWPR, Power Policy Register on page 488
Changed register summary topic text, to include statement about Direct connect	B.2.1 External CTI register summary on page 536
Register updated to indicate p0 (in r3p0 product revision)	B.2.1.46 CTIPIDR3, CTI Peripheral Identification Register 3 on page 618
Changed register summary topic text, to include statement about Direct connect	B.2.2 External cluster ROM register summary on page 624
Updated the POWERID field	B.2.2.3 CLUSTERROM_ROMENTRY2, Cluster ROM table entry 2 on page 628
Updated the POWERID field	B.2.2.4 CLUSTERROM_ROMENTRY3, Cluster ROM table entry 3 on page 631
Updated the POWERID field	B.2.2.5 CLUSTERROM_ROMENTRY4, Cluster ROM table entry 4 on page 634
Updated the POWERID field	B.2.2.6 CLUSTERROM_ROMENTRY5, Cluster ROM table entry 5 on page 637
Updated the POWERID field	B.2.2.7 CLUSTERROM_ROMENTRY6, Cluster ROM table entry 6 on page 640
Updated the POWERID field	B.2.2.8 CLUSTERROM_ROMENTRY7, Cluster ROM table entry 7 on page 643
Updated the POWERID field	B.2.2.9 CLUSTERROM_ROMENTRY8, Cluster ROM table entry 8 on page 646
Updated the POWERID field	B.2.2.10 CLUSTERROM_ROMENTRY9, Cluster ROM table entry 9 on page 649
Added new register in the cluster ROM table	B.2.2.11 CLUSTERROM_ROMENTRY10, Cluster ROM table entry 10 on page 652
Added new register in the cluster ROM table	B.2.2.12 CLUSTERROM_ROMENTRY11, Cluster ROM table entry 11 on page 655
Added new register in the cluster ROM table	B.2.2.13 CLUSTERROM_ROMENTRY12, Cluster ROM table entry 12 on page 658
Added new register in the cluster ROM table	B.2.2.14 CLUSTERROM_ROMENTRY13, Cluster ROM table entry 13 on page 661
Added new register in the cluster ROM table	B.2.2.23 CLUSTERROM_DBGPCR8, Cluster ROM table Debug Power Control Register 8 on page 675
Added new register in the cluster ROM table	B.2.2.24 CLUSTERROM_DBGPCR9, Cluster ROM table Debug Power Control Register 9 on page 677
Added new register in the cluster ROM table	B.2.2.25 CLUSTERROM_DBGPCR10, Cluster ROM table Debug Power Control Register 10 on page 678
Added new register in the cluster ROM table	B.2.2.26 CLUSTERROM_DBGPCR11, Cluster ROM table Debug Power Control Register 11 on page 680
Added new register in the cluster ROM table	B.2.2.35 CLUSTERROM_DBGPSR8, Cluster ROM table Debug Power Status Register 8 on page 693

Change	Location
Added new register in the cluster ROM table	B.2.2.36 CLUSTERROM_DBGPSR9, Cluster ROM table Debug Power Status Register 9 on page 695
Added new register in the cluster ROM table	B.2.2.37 CLUSTERROM_DBGPSR10, Cluster ROM table Debug Power Status Register 10 on page 696
Added new register in the cluster ROM table	B.2.2.38 CLUSTERROM_DBGPSR11, Cluster ROM table Debug Power Status Register 11 on page 698
Added new register in the cluster ROM table	B.2.2.48 CLUSTERROM_PIDR3, Cluster ROM table Peripheral Identification Register 3 on page 709
Changed register summary topic text, to include statement about Direct connect	B.2.3 External debug ROM register summary on page 715
Added new register in the debug ROM table	B.2.3.11 DBROM_ROMENTRY10, DebugBlock ROM table Entry 10 on page 731
Added new register in the debug ROM table	B.2.3.12 DBROM_ROMENTRY11, DebugBlock ROM table Entry 11 on page 732
Added new register in the debug ROM table	B.2.3.13 DBROM_ROMENTRY12, DebugBlock ROM table Entry 12 on page 734
Added new register in the debug ROM table	B.2.3.14 DBROM_ROMENTRY13, DebugBlock ROM table Entry 13 on page 735
Updated register to indicate p0 (in r3p0 product revision)	B.2.3.26 DBROM_PIDR3, DebugBlock ROM table Peripheral Identification Register 3 on page 749
Changed register summary topic text, to include statement about Direct connect	B.2.4 External cluster PMU register summary on page 754
Updated register to indicate p0 (in r3p0 product revision)	B.2.4.50 CLUSTERPMU_PMPIDR3, Cluster Performance Monitors Peripheral Identification Register 3 on page 846

Table C-8: Differences between Issue 0300-07 and Issue 0300-08

Change	Location
Second confidential Early Access release for r3p0	-
Updated product name along the book accordingly: • Theodul DSU changed to DSU-110 • Theodul DynamIQ Shared Unit changed to DynamIQ Shared Unit-110 • DynamIQ Shared Unit changed to DynamIQ Shared Unit-110 • Theodul DynamIQ cluster changed to DSU-110 DynamIQ cluster • DynamIQ Shared Unit changed to DynamIQ Shared Unit-110 • DSU bridge changed to DSU-110 bridge	Throughout book
AXI 256-bit master interface Write issuing capability and Read issuing capability - maximum non-reorderable devices changed from 40 to 56.	9.4 AXI 256-bit master interface attributes on page 121
Various arrangements of three types of cores (or four added).	2.3 Cluster configurations on page 22
Added new task section as 6.2.3 where describing, how to initialize the cluster into a different operating mode.	6.2.3 Initial cluster operating mode on page 83
The following description added: The upper limit for the range of power modes is ON. The upper limit for the range of cluster operating modes is All slices and all RAM instances are active.	6.2 Power policy unit operation on page 80

Change	Location
The following description added: The upper limit for the range of power modes is ON. The upper limit for the range of cluster operating modes is All slices and all RAM instances are active.	6.6.3 Programming sequence for an interrupt controller to control transitions between On and Off mode on page 92
AWSTASHLPIDS[3:1] changed to **AWSTASHLPIDS[3:0]**. The following statement was deleted: The signal **AWSTASHLPIDS[0]** is reserved for the thread number, but this does not affect the stash request.	10.3 ACP transactions on page 128
New topic was created as 15.9 in the *Debug* section to describe ELA content.	15.9 Trace output from cores and DynamIQ cluster on page 188
Updated the values in the table *CHI LPID[4:0] bitfields* for bit fields **3:0** to align with the 12 core cluster.	8.6 CHI transactions on page 114
Added note at end of section, to describe assumptions about the word core, and complexed core.	2.3.1 What is a complex? on page 25
Rephrased short description for better explanation on complexes.	2.3.1 What is a complex? on page 25
Note states that the *Accelerator Coherency Port* (ACP) interface is not supported in this release, deleted.	4. Clocks and resets on page 44
The DebugBlock can be clocked by a different clock from the cluster, and bridges have to be established between the cluster and the DebugBlock instead of DSU-110.	4.1 Clocks on page 44
The ROM table entry values are specified.	B.2.2 External cluster ROM register summary on page 624
The ROM table entry values are specified.	B.2.3 External debug ROM register summary on page 715
Updated register description to indicate what silicon revision the component revision applies too.	B.1.3.23 CLUSTERRAS_ERRPIDR2, Peripheral Identification Register 2 on page 425
Updated register description to indicate what silicon revision the component revision applies too.	B.2.1.45 CTIPIDR2, CTI Peripheral Identification Register 2 on page 616
Updated register description to indicate what silicon revision the component revision applies too.	B.2.2.47 CLUSTERROM_PIDR2, Cluster ROM table Peripheral Identification Register 2 on page 708
Updated register description to indicate what silicon revision the component revision applies too.	B.2.3.25 DBROM_PIDR2, DebugBlock ROM table Peripheral Identification Register 2 on page 748
Updated register description to indicate what silicon revision the component revision applies too.	B.2.4.49 CLUSTERPMU_PMPIDR2, Cluster Performance Monitors Peripheral Identification Register 2 on page 844

Table C-9: Differences between Issue 0300-08 and Issue 0301-09

Change	Location
First confidential Early Access release for r3p1	-
Clarified the description of transaction ID capability for the CHI master ports.	8.4 Attributes of the CHI master interface on page 112
Clarified the description of transaction ID capability for the CHI peripheral ports.	11.7 Attributes of the CHI peripheral port on page 143
Updated the CoreSight component identification values for revision r3p1.	15.10 CoreSight component identification on page 188
Corrected description of bit[4] of the ERXCTLR_EL1 register.	A.3.2 ERXCTLR_EL1, Selected Error Record Control Register on page 312
Corrected description of bit[4] of the CLUSTERRAS_ERR0CTLR register.	A.3.2 ERXCTLR_EL1, Selected Error Record Control Register on page 312

Table C-10: Differences between Issue 0301-09 and Issue 0400-10

Change	Location
First release for r4p0	-
Added L3 cache stashing content.	7.4 Cache stashing on page 103
Added Direct connect to list of configuration parameters.	2.2 DynamIQ Shared Unit-110 configuration parameters on page 20
Added an open-ended arrow from OFF_EMU state to any MEM_RET_EMU state in figure *DSU-110 DynamIQ ™ cluster PPU mode transitions*.	Figure 5-2: DSU-110 DynamIQ cluster PPU mode transitions on page 65
Fixed arrow between PPI and SCP in figure *DSU-110 PPU connections to a power-gated domain*.	Figure 6-2: DSU-110 PPU connections to a power-gated domain on page 80
Added new topic about power mode transition dependencies for a dual-core complex .	5.9.2 Power mode transition dependencies for a dual-core complex on page 75
Updated **CLUSTERPCSMSTATE[15:8]** bits. Bits are tied to 0.	6.4.2 Encodings for cluster power and operating modes on page 85
Qualified note about not using FULL_RET or FUNC_RET with static mode, to say applies to core and cluster power modes.	6.10 Core Full retention mode and static mode restrictions on page 97
Updated Core 7 CTI value.	16.1 Debug system address map on page 190
Added that the DSU-110 buffers the cores and complexes to be clocked.	3.1.1 Integration of the cores in the cluster on page 34
Added L1 or L2 cache maintenance operation to FULL_RET mode when the core transitions to On mode.	5.8.1 Core PPU mode transitions on page 71
Updated *HIT events by L3 cache being off.	17.2 PMU events on page 203
Removed *MATCH and *KILL events.	17.2 PMU events on page 203
Added note about cache way information.	5.6 Power states for the cluster RAM instances on page 63
Updated note about "*Storing MPAM ID value in the L3 cache is only required if there is a downstream cache which supports MPAM*".	7.3 L3 cache partitioning on page 100
Added note about 64-bit AXI peripheral port dependency.	11.11 Peripheral port and ACP interface usage on page 149
Improved CLUSTERCDBG[50] description about MPAM.	15.1 Cache debug on page 171
Removed DVMOp branch predictor.	11.9 CHI peripheral port transactions on page 145
Updated AArch64 RAS register summary by RAS registers with *IMPLEMENTATION DEFINED*.	A.3 AArch64 RAS register summary on page 308
Added configuration parameters to the External debug ROM register summary.	B.2.3 External debug ROM register summary on page 715
Added explanatory notes about L3 cache applying to non-Direct Connect cores, and for Direct Connect cores CHI stashing is supported.	7. L3 cache on page 99
Added content about managing RAS fault and error interrupts during the cluster powerdown procedure.	6.9 ECC errors during power transitions on page 95
Added content about managing RAS fault and error interrupts during the cluster powerdown procedure.	12.6 ECC errors during power transitions on page 159
Qualified table introduction: "*The following table shows the base addresses for each set of system component registers and what Security state they should be accessed from.*"	13.2 Base addresses for system components on page 166
Added note to BUS_ACCESS_RD `0x0060` about CHI MakeReadUnique transaction.	17.2 PMU events on page 203

Change	Location
Added note to BUS_ACCESS_WR `0x0061` about CHI WriteEvictOrEvict transaction.	17.2 PMU events on page 203
Added access cases when RAZ/WI is returned and CTI part number.	B.2.1 External CTI register summary on page 536
Added access cases when RAZ/WI is returned.	B.2.2 External cluster ROM register summary on page 624
Added access cases when RAZ/WI is returned.	B.2.3 External debug ROM register summary on page 715
Added access cases when RAZ/WI is returned.	B.2.4 External cluster PMU register summary on page 754
Added reference to *Arm® CoreSight™ Architecture Specification v3.0* for Debug authentication details.	B.2.1.36 CTIAUTHSTATUS, CTI Authentication Status register on page 606
Removed registers: IMP_CLUSTERRSVD_9_3_EL1, IMP_CLUSTERRSVD_9_4_EL1, IMP_CLUSTERRSVD_9_5_EL1, IMP_CLUSTERRSVD_9_6_EL1, IMP_CLUSTERRSVD_9_7_EL1 and associated register descriptions.	14.1 AArch64 generic system control registers on page 168
Removed note about 64-bit AXI configured peripheral port to complete its accesses independently of the ACP as is supported for REL.	11.11 Peripheral port and ACP interface usage on page 149
Updated Data_Poison CHI interface description by added Direct connect and non-Direct connect options.	8.2 CHI features on page 110
Updated CoreSight component table for r4p0.	15.10 CoreSight component identification on page 188
Removed notes about cache stashing support for future releases, as is supported for REL.	Throughout book

Table C-11: Differences between Issue 0400-10 and Issue 0400-11

Change	Location
Second release for r4p0	-
Updated Warm reset content.	5.3.7 Warm reset mode (WARM_RST) on page 55
Editorial changes	Throughout book