



# Arm<sup>®</sup> DynamIQ<sup>™</sup> Shared Unit-120AE

Revision r0p1

## Technical Reference Manual

**Non-Confidential**

**Issue 03**

Copyright © 2023–2024 Arm Limited (or its affiliates). 107721\_0001\_03\_en  
All rights reserved.



# Arm® DynamIQ™ Shared Unit-120AE Technical Reference Manual

This document is Non-Confidential.

Copyright © 2023–2024 Arm Limited (or its affiliates). All rights reserved.

This document is protected by copyright and other intellectual property rights.

Arm only permits use of this document if you have reviewed and accepted [Arm's Proprietary Notice](#) found at the end of this document.

This document (107721\_0001\_03\_en) was issued on 2024-12-13. There might be a later issue at <https://developer.arm.com/documentation/107721>

The product revision is r0p1.

See also: [Proprietary Notice](#) | [Product and document information](#) | [Useful resources](#)

## Start Reading

If you prefer, you can skip to [the start of the content](#).

## 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-120AE along with an Arm core or cores.

## 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.

We believe that this document contains no offensive language. To report offensive language in this document, email [terms@arm.com](mailto:terms@arm.com).

## 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>.

# Contents

<b>1. The DynamIQ™ Shared Unit-120AE.....</b>	<b>12</b>
1.1 DynamIQ™ Shared Unit-120AE features.....	16
1.2 DynamIQ™ Shared Unit-120AE configuration options.....	18
1.3 DCLS configurations.....	20
1.3.1 Lock-configuration.....	20
1.3.2 Split-configuration.....	22
1.3.3 Mixed-configuration.....	24
1.3.4 Primary and redundant logic comparison.....	33
1.4 Cluster configurations.....	38
1.4.1 What is a complex?.....	42
1.4.2 L3 memory system variants.....	44
1.5 Supported standards and specifications.....	45
1.6 Test features.....	46
1.7 Design Tasks.....	46
1.8 Core, complex, and processing element numbering.....	47
1.9 Product revisions.....	49
<b>2. Technical overview.....</b>	<b>50</b>
2.1 DynamIQ cluster components.....	50
2.1.1 Integration of the cores in the cluster.....	52
2.2 DynamIQ™ cluster shared logic components.....	52
2.3 DebugBlock components.....	56
2.4 Interfaces.....	57
2.4.1 Page-Based Hardware Attribute.....	61
2.4.2 Sequential hint.....	61
2.4.3 Interface protection.....	62
<b>3. Clocks and resets.....</b>	<b>64</b>
3.1 Clocks.....	64
3.2 Clock domains.....	66
3.3 Resets.....	67
3.4 Resetting with Power Policy Units.....	68

<b>4. Power management.....</b>	<b>70</b>
4.1 Power management in the DSU-120AE.....	70
4.2 DSU-120AE supported power domains.....	71
4.3 Cluster power modes.....	73
4.3.1 On mode (ON).....	74
4.3.2 Off mode (OFF).....	74
4.3.3 Functional retention mode (FUNC_RET).....	74
4.3.4 Cluster full retention mode (FULL_RET).....	75
4.3.5 Memory retention mode (MEM_RET).....	75
4.3.6 Emulated off mode (OFF_EMU).....	76
4.3.7 Emulated memory retention mode (MEM_RET_EMU).....	76
4.3.8 Warm reset mode (WARM_RST).....	76
4.3.9 Debug recovery mode (DBG_RECOV).....	77
4.4 L3 RAM power control.....	78
4.4.1 L3 cache RAM powerdown.....	79
4.4.2 L3 cache slice powerdown.....	83
4.4.3 L3 cache RAM Quick Nap.....	86
4.5 Cluster operating modes.....	87
4.6 Power states for the cluster RAM instances.....	87
4.7 Cluster PPU mode transitions.....	89
4.7.1 Rules governing cluster PPU mode transitions.....	93
4.7.2 PPU mode transition behavior.....	93
4.7.3 DebugBlock power modes.....	95
4.8 Core PPU modes.....	95
4.8.1 Core PPU mode transitions.....	96
4.9 Complex power management.....	99
4.9.1 Complex power modes.....	99
4.9.2 Power mode transition dependencies for a dual-core complex.....	100
4.10 Maximum Power Mitigation Mechanism.....	101
4.11 DSU-120AE voltage domains.....	102
<b>5. Power and reset control with Power Policy Units.....</b>	<b>103</b>
5.1 The Power Policy Unit.....	103
5.2 Power Policy Unit mode control for Lock-configuration and Mixed-configuration.....	106
5.3 Power policy unit operation.....	107
5.3.1 Implicit resets from power modes.....	108

5.3.2 nRESET sequence.....	109
5.3.3 Initial cluster operating mode.....	109
5.4 PPU and CLUSTERAE register protection.....	110
5.5 Utility bus accesses.....	111
5.6 Cluster PPU mode control.....	111
5.6.1 External cluster PPU registers.....	111
5.6.2 Encodings for cluster power and operating modes.....	113
5.7 Core power mode control.....	115
5.7.1 External core PPU registers.....	115
5.7.2 Encodings for core power modes.....	117
5.8 Selecting different modes in Mixed-configuration.....	118
5.9 Programming sequences for the cluster and the core.....	118
5.9.1 Programming sequence to bring the cluster and cores from Off to On mode.....	118
5.9.2 Programming sequence to bring the cluster and cores from On to Off mode.....	119
5.9.3 Programming sequence for an interrupt controller to control transitions between On and Off mode.....	120
5.10 Explicit reset of the cluster and cores.....	121
5.10.1 Powerup (Cold) reset.....	122
5.10.2 Core software initiated Warm reset of an individual core.....	122
5.10.3 Core software initiated Cold or Warm reset of the cluster, excluding the PPUs.....	123
5.10.4 Warm reset of the cluster, excluding the PPUs.....	124
5.10.5 Cold reset of the whole cluster, including the PPUs, retaining cache contents for debug...	127
5.10.6 Reset of the cluster, excluding the PPUs, retaining the cache contents for debug.....	128
5.11 Power mode dependencies between the core and the cluster.....	133
5.12 ECC errors during power transitions.....	134
5.13 Core Full retention mode and static mode restrictions.....	135
5.14 Minimum mode and dynamic mode restrictions.....	135
<b>6. L3 cache.....</b>	<b>136</b>
6.1 L3 cache allocation policy.....	136
6.2 Available number of cache ways.....	137
6.3 Memory System Resource Partitioning and Monitoring control.....	137
6.4 L3 cache partitioning.....	138
6.5 Bandwidth partitioning.....	140
6.6 Cache stashing.....	142
6.7 L3 cache data RAM latency.....	142
6.8 Cache slices and power portions.....	144

6.8.1 Cache slice and requester port selection.....	145
<b>7. CHI requester interface.....</b>	<b>146</b>
7.1 Multiple CHI bus requester port configurations.....	146
7.2 Configure CHI bus requester ports to use address target groups.....	146
7.2.1 Hashing for CHI transaction distribution.....	147
7.2.2 Mapping for address target groups to CHI bus requester ports.....	149
7.2.3 CHI id bit setting.....	150
7.3 CHI transaction routing with multiple requester ports.....	150
7.4 CHI features.....	153
7.5 CHI configurations.....	154
7.6 Attributes of the CHI requester interface.....	155
7.7 CHI channel properties.....	156
7.8 CHI transactions.....	157
7.9 Use of DataSource field.....	161
7.10 Support for memory types.....	161
<b>8. AXI manager interface.....</b>	<b>162</b>
8.1 Multiple AXI bus manager port configurations.....	162
8.2 Configure AXI bus manager ports to use address target groups.....	162
8.2.1 Hashing for AXI transaction distribution.....	163
8.2.2 Mapping for address target groups to AXI bus manager ports.....	165
8.2.3 AXI id bit setting.....	166
8.3 AXI transaction routing with multiple manager ports.....	166
8.4 AXI manager port interface properties.....	167
8.5 AXI configurations.....	169
8.6 AXI 256-bit manager interface attributes.....	169
8.7 AXI transactions.....	170
8.8 Support for memory types.....	171
8.9 Read response.....	171
8.10 Write response.....	172
8.11 Barriers.....	172
8.12 AXI privilege information.....	172
<b>9. ACP subordinate interface.....</b>	<b>173</b>
9.1 ACP features.....	173
9.2 ACP ACE5-LiteDVM protocol subset.....	175

9.3 ACP transactions.....	176
9.4 ACP performance.....	179
9.5 DVM snoop transaction support.....	180
9.5.1 Control the receiving of DVM snoop transactions.....	181
<b>10. AXI or CHI requester peripheral port.....</b>	<b>183</b>
10.1 Supported memory and transaction types.....	183
10.2 Mapping peripheral port address ranges.....	184
10.2.1 Changing peripheral port address range.....	185
10.3 AXI 64-bit peripheral port interface properties.....	186
10.4 AXI 256-bit peripheral port interface properties.....	187
10.5 AXI 64-bit peripheral port transactions.....	188
10.6 AXI 256-bit peripheral port transactions.....	189
10.7 Attributes of the CHI peripheral port.....	190
10.8 CHI peripheral port interface properties.....	192
10.9 CHI peripheral port transactions.....	193
10.10 Read and write capabilities and transaction ID encoding.....	194
10.11 Peripheral port and ACP interface usage.....	196
10.12 AXI privilege information for the AXI-configured peripheral port.....	198
<b>11. RAS extension support.....</b>	<b>199</b>
11.1 Cache protection behavior.....	199
11.2 Error containment.....	202
11.3 Fault detection and reporting.....	202
11.4 Error detection and reporting.....	202
11.4.1 Error reporting and performance monitoring.....	204
11.4.2 Errors not counted.....	204
11.4.3 Double error reporting.....	204
11.5 Error injection.....	205
11.6 ECC errors during power transitions.....	206
11.7 Core RAS registers.....	207
11.8 Cluster RAS registers.....	208
11.8.1 AArch64 RAS registers.....	208
11.8.2 External cluster RAS registers.....	209
<b>12. Utility bus.....</b>	<b>211</b>
12.1 Utility bus accesses.....	211

12.1.1 Core access to system component registers.....	212
12.1.2 Cluster and core PPU register access.....	213
12.2 Base addresses for system-accessible components.....	213
<b>13. System control registers.....</b>	<b>215</b>
13.1 AArch64 generic-system-control registers.....	215
<b>14. Debug.....</b>	<b>217</b>
14.1 Cache debug.....	219
14.2 Supported debug methods.....	226
14.3 Terminology.....	227
14.4 Cluster and core Debug power control.....	227
14.5 DebugBlock overview.....	228
14.6 DebugBlock subcomponents.....	230
14.7 Embedded Cross Trigger overview.....	232
14.7.1 CTI triggers.....	233
14.8 External CTI registers.....	235
14.9 Trace output from cores and DynamIQ cluster.....	237
14.10 CoreSight component identification.....	238
<b>15. ROM tables.....</b>	<b>239</b>
15.1 Debug system address map.....	240
15.2 DebugBlock ROM table.....	243
15.3 Cluster ROM table.....	244
15.4 ROM table power request registers for cluster and cores.....	246
15.5 External cluster ROM registers.....	246
15.6 External debug ROM registers.....	248
<b>16. Performance Monitors Extension support.....</b>	<b>250</b>
16.1 PMU features.....	250
16.2 PMU events.....	251
16.3 PMU interrupt.....	254
16.4 External cluster PMU registers.....	255
<b>17. Activity Monitors Extension support.....</b>	<b>258</b>
17.1 Activity monitors access.....	258
17.2 Activity monitors counters.....	258
17.3 External cluster AMU registers.....	259



17.4 Activity monitors events.....	260
------------------------------------	-----

## **A. AArch64 registers.....262**

A.1 AArch64 generic system control registers summary.....	262
A.1.1 IMP_CLUSTERCFR_EL1, Cluster Configuration Register.....	263
A.1.2 IMP_CLUSTERIDR_EL1, Cluster Main Revision Register.....	269
A.1.3 IMP_CLUSTERREVIDR_EL1, Cluster ECO ID Register.....	271
A.1.4 IMP_CLUSTERACTLR_EL1, Cluster Auxiliary Control Register.....	273
A.1.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register.....	275
A.1.6 IMP_CLUSTERPWRCTLR_EL1, Cluster Power Control Register.....	279
A.1.7 IMP_CLUSTERPWRDN_EL1, Cluster Power Down Register.....	285
A.1.8 IMP_CLUSTERPWRSTAT_EL1, Cluster Power Status Register.....	287
A.1.9 IMP_CLUSTERL3DNTH0_EL1, Cluster L3 Downsize Threshold0 Register.....	290
A.1.10 IMP_CLUSTERL3DNTH1_EL1, Cluster L3 Downsize Threshold1 Register.....	292
A.1.11 IMP_CLUSTERL3UPTH0_EL1, Cluster L3 Upsize Threshold0 Register.....	294
A.1.12 IMP_CLUSTERL3UPTH1_EL1, Cluster L3 Upsize Threshold1 Register.....	296
A.1.13 IMP_CLUSTERBUSQOS_EL1, Cluster Bus QoS Control Register.....	298
A.1.14 IMP_CLUSTERL3HIT_EL1, Cluster L3 Hit Counter Register.....	300
A.1.15 IMP_CLUSTERL3MISS_EL1, Cluster L3 Miss Counter Register.....	301
A.1.16 IMP_CLUSTERPPSTART_EL1, Cluster Peripheral Port Start Address Register.....	303
A.1.17 IMP_CLUSTERPPEND_EL1, Cluster Peripheral Port End Address Register.....	305
A.1.18 IMP_CLUSTERCFR2_EL1, Cluster Configuration Register 2.....	307
A.1.19 IMP_CLUSTERL3UPTH2_EL1, Cluster L3 Upsize Threshold2 Register.....	309
A.1.20 IMP_CLUSTERCDBG_EL3, Cluster Cache Debug Register.....	311
A.1.21 IMP_CLUSTERPMMDCR_EL3, Monitor Debug Configuration Register (EL3).....	314
A.2 AArch64 performance monitors registers summary.....	316
A.2.1 IMP_CLUSTERPMCR_EL1, Performance Monitors Control Register.....	317
A.2.2 IMP_CLUSTERPMCNTENSET_EL1, Performance Monitors Count Enable Set Register.....	319
A.2.3 IMP_CLUSTERPMCNTENCLR_EL1, Performance Monitors Count Enable Clear Register....	322
A.2.4 IMP_CLUSTERPMOVSET_EL1, Performance Monitors Overflow Flag Status Set Register.....	325
A.2.5 IMP_CLUSTERPMOVCLR_EL1, Performance Monitors Overflow Flag Status Clear Register.....	328
A.2.6 IMP_CLUSTERPMSELR_EL1, Performance Monitors Event Counter Selection Register.....	331
A.2.7 IMP_CLUSTERPMINTENSET_EL1, Performance Monitors Interrupt Enable Set Register....	334
A.2.8 IMP_CLUSTERPMINTENCLR_EL1, Performance Monitors Interrupt Enable Clear Register.....	337

A.2.9 IMP_CLUSTERPMCCNTR_EL1, Performance Monitors Cycle Count Register.....	340
A.2.10 IMP_CLUSTERPMXEVTYPER_EL1, Performance Monitors Selected Event Type Register.....	342
A.2.11 IMP_CLUSTERPMXVCNTR_EL1, Performance Monitors Selected Event Count Register.....	344
A.2.12 IMP_CLUSTERPMCEID0_EL1, Performance Monitors Common Event Identification Register 0.....	346
A.2.13 IMP_CLUSTERPMCEID1_EL1, Performance Monitors Common Event Identification Register 1.....	353
A.3 AArch64 RAS registers summary.....	360
A.3.1 ERXFR_EL1, Selected Error Record Feature Register.....	361
A.3.2 ERXCTLR_EL1, Selected Error Record Control Register.....	364
A.3.3 ERXSTATUS_EL1, Selected Error Record Primary Status Register.....	368
A.3.4 ERXPFGF_EL1, Selected Pseudo-fault Generation Feature Register.....	375
A.3.5 ERXPFGCTL_EL1, Selected Pseudo-fault Generation Control Register.....	378
A.3.6 ERXPFGCDN_EL1, Selected Pseudo-fault Generation Countdown Register.....	383
A.3.7 ERXMISCO_EL1, Selected Error Record Miscellaneous Register 0.....	385
A.3.8 ERXMISC1_EL1, Selected Error Record Miscellaneous Register 1.....	389
A.3.9 ERXMISC2_EL1, Selected Error Record Miscellaneous Register 2.....	391
A.3.10 ERXMISC3_EL1, Selected Error Record Miscellaneous Register 3.....	393
<b>B. External registers.....</b>	<b>396</b>
B.1 Registers accessed over the utility bus.....	396
B.1.1 External cluster system control registers summary.....	396
B.1.2 External MPAM registers summary.....	426
B.1.3 External cluster RAS registers summary.....	452
B.1.4 External cluster PPU registers summary.....	503
B.1.5 External cluster AMU registers summary.....	564
B.1.6 External cluster AE registers summary.....	600
B.1.7 External core PPU registers summary.....	623
B.2 Registers accessed over the Debug APB bus.....	678
B.2.1 External cluster and core CTI registers summary.....	678
B.2.2 External cluster ROM registers summary.....	771
B.2.3 External debug ROM registers summary.....	873
B.2.4 External cluster PMU registers summary.....	915
<b>Proprietary Notice.....</b>	<b>1006</b>

**Product and document information..... 1008**

Product status..... 1008

Revision history..... 1008

Conventions..... 1009

**Useful resources..... 1012**

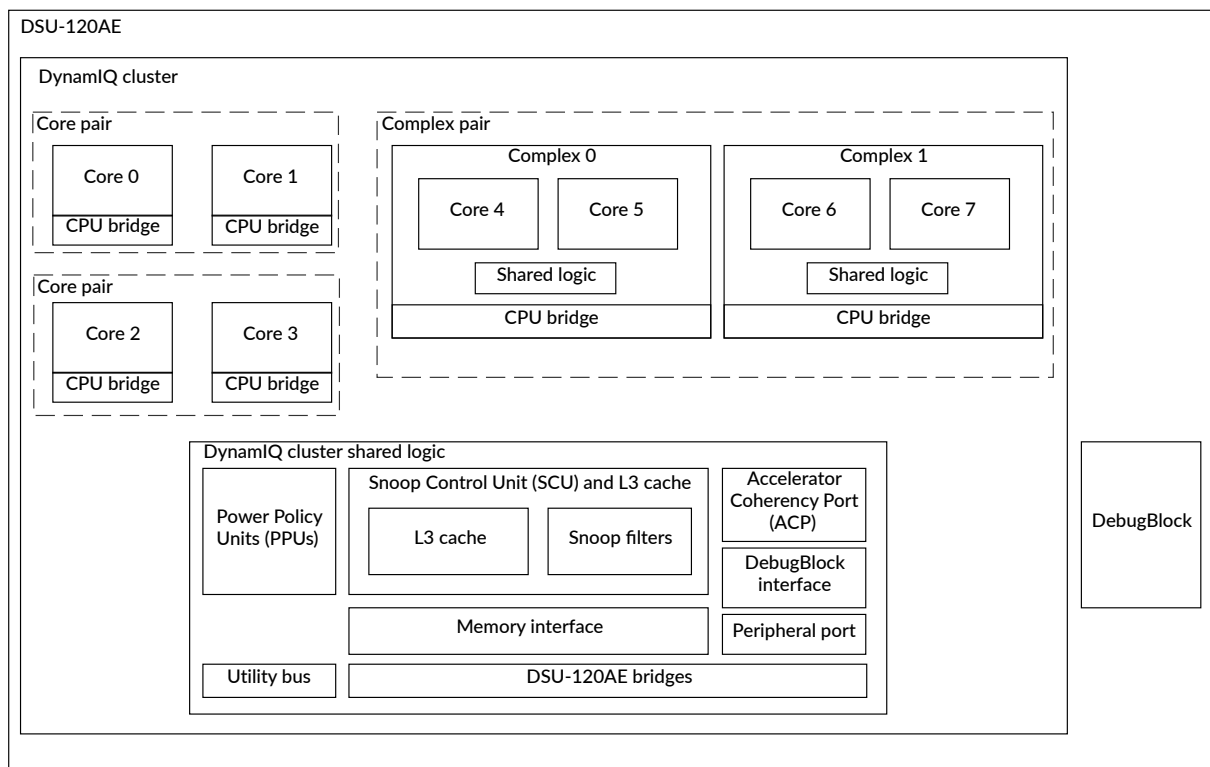
# 1. The DynamIQ™ Shared Unit-120AE

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) provides a shared L3 memory system, snoop control and filtering, and other control logic to support a cluster of A-class architecture cores. The DSU-120AE also adds in bus protection and *Dual-Core Lock-Step* (DCLS) features to support additional fault protection.

## DSU-120AE cluster overview

The following figure shows an example of a DSU-120AE-based cluster in one particular build time configuration. This configuration excludes the DCLS features, so that functionality that is common to all configurations can easily be described.

**Figure 1-1: DSU-120AE DynamIQ™ cluster**



- The figure above is one possible representation of the cluster and it can vary between the different configurations. For more information about the configurations, see the [1.3 DCLS configurations](#) on page 20.
- In this book, the DSU-120AE DynamIQ™ cluster is referred to as a cluster in cases where distinguishing between the DSU-120AE DynamIQ™ cluster and the DSU-120AE is not important to the context.

All cores and complexes in the DSU-120AE DynamIQ™ cluster 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-120AE 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-120AE DynamIQ™ cluster supports many mechanisms to reduce static and dynamic power dissipation. For example, placing the core 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-120AE 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 manager interface. The *Accelerator Coherency Port* (ACP) provides coherent access for non-cached managers 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-120AE that provides the interface for debug capability. The DebugBlock is instantiated 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.

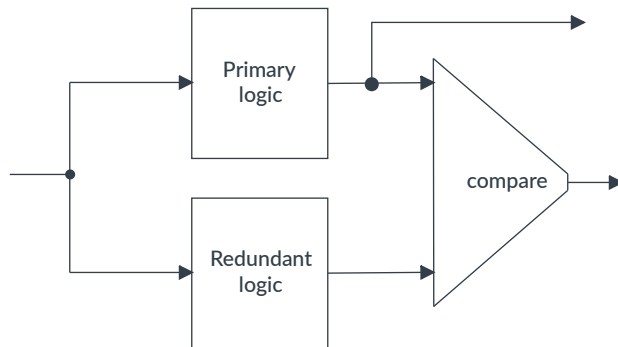


- For information on the behavior and features of your core, including whether your core is supported in a complex, see the *Technical Reference Manual* (TRM) for your core.
  - For information on the DSU-120AE macrocell implementation, see *DSU-120AE Configuration and Integration Manual*.
- 

## DCLS feature overview

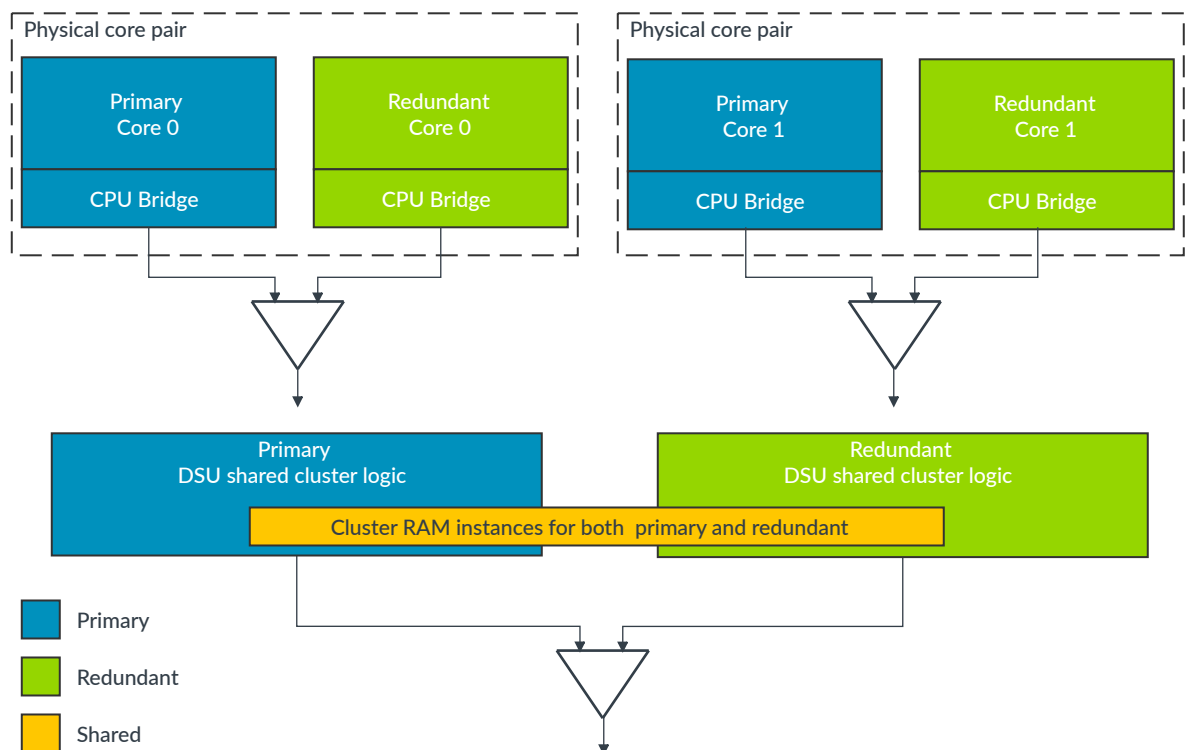
In DCLS implementations, the logic is duplicated to form the primary logic and redundant logic as the following figure shows. The same inputs are applied to both the primary and redundant logic and the outputs are compared. The redundant logic block therefore acts as a check to the operation of the primary logic block. This process of tracking the logic is known as Lock-Step, and the duplication of logic allows for error detection.

**Figure 1-2: DCLS principle**



In a DSU-120AE-based cluster, the same principle is applied to the cores in the cluster, and the DSU-120AE logic itself. For example, the following figure shows a basic top-level view of the duplicated logic in a DSU-120AE based cluster including four *Automotive Enhanced* (AE) cores. Each primary and redundant AE core forms a core pair. Both cores that make up a core pair must be the same type of core with the same configuration. The DSU-120AE logic is also duplicated, apart from the shared memory block which includes the L3 RAMs and snoop filter RAMs. Comparators are also included to compare the outputs of each core pair and the DSU shared cluster logic as shown.

**Figure 1-3: Lock-configuration in the DSU-120AE-based cluster**



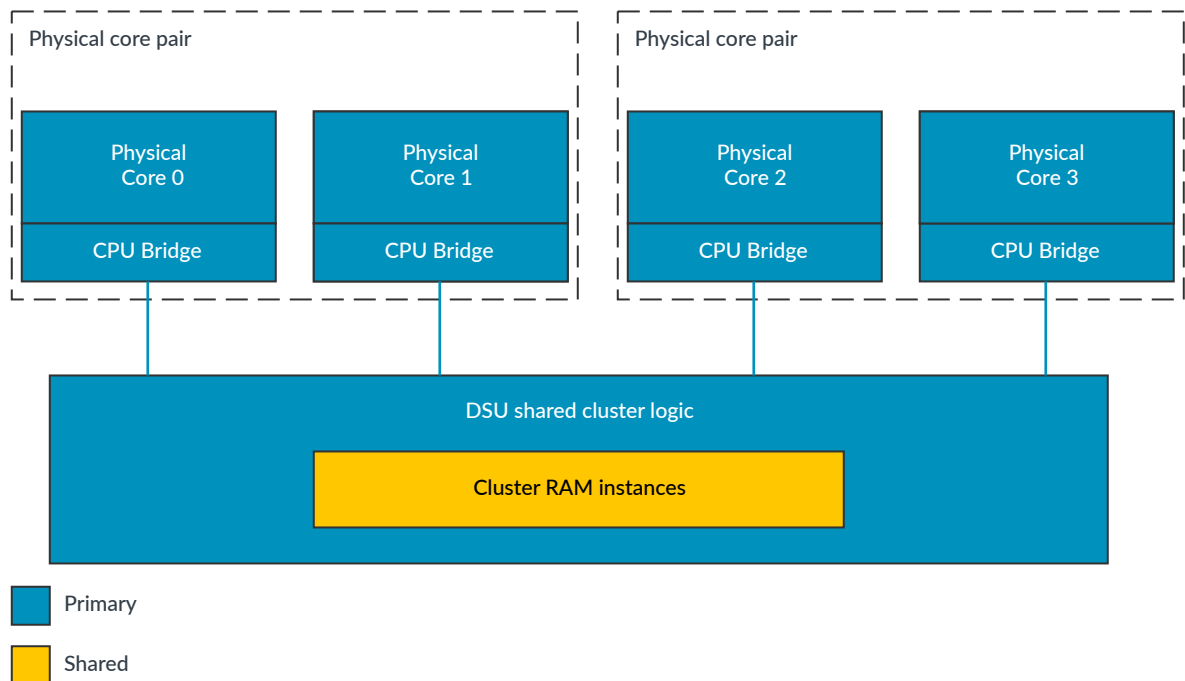


- The preceding figure is a conceptual figure to show how the DCLS principle is applied to a DSU-120AE-based cluster. For more information on the actual comparator arrangement in the DSU-120AE-based cluster, see [1.3.4 Primary and redundant logic comparison](#) on page 33.
- In this book the term core pair is used. Processor cores that are compatible with the DSU-120AE Lock-configuration are always instantiated as core pairs. A core pair consists of two identical instances of the compatible processor core. In Lock-configuration, the two cores act together in the core pair to function architecturally as a single core. In Split-configuration each core behaves independently and architecturally the two cores act as two independent cores.

The preceding arrangement where the DSU-120AE logic and AE cores are duplicated and comparators are instantiated is one possible build-time configuration of the DSU-120AE known as Lock-configuration.

Another build-time configuration is to configure the cluster and cores not to use DCLS. In this configuration, there is no duplication, and no instantiation of comparators, and therefore no DCLS error detection or correction. This configuration is called Split-configuration. The following figure shows the same DSU-120AE-based cluster, as in preceding figure, but configured for Split-configuration.

**Figure 1-4: Split-configuration in the DSU-120AE-based cluster**



As can be seen from comparing the two preceding figures, core pairs consist of evenly numbered core instances. For example, core pair 0 would consist of core 0 and core 1, and core pair 1 is made up of core 2 and core 3.

In a Split-configuration, the DSU-120AE-based cluster can be configured to use between two and 14 cores. However if using a Lock-configuration, the total number of cores configured is the number of primary cores, and ranges between one and 7 cores. The DSU-120AE-based cluster can also be configured to use up to two 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-120AE DynamIQ™ cluster also supports complexes where typically two cores of the same type are linked together and share logic. Examples of shared logic include a floating-point unit and an L2 cache. In addition to supporting core pairs, the DSU-120AE DynamIQ™ cluster also supports complex pairs, where one of the two complexes is a primary complex and the other one a redundant complex. Again both the primary and redundant complexes must be identical to one another. For more information on DCLS organization and operation, see [1.3 DCLS configurations](#) on page 20.

## 1.1 DynamIQ™ Shared Unit-120AE features

Some features in the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) are fixed and some features are optional. You can configure optional features in the RTL during build time configuration, to meet your requirements.

### Fault protection features

The DSU-120AE introduces the following additional fault protection features:

- *Dual-Core Lock-Step* (DCLS) support for fault protection
- Split-configuration, Lock-configuration, and Mixed-configuration support. Mixed-configuration allows for DCLS support to be pin configured and includes Lock-mode, Split-mode, and a Hybrid-mode for flexible fault protection level. DCLS also includes temporal delay support for temporal diversity.
- *Fault Management Unit* (FMU) support for fault reporting structure
- Interface protection for all external interfaces in Lock-configuration and Mixed-configuration, except for the non-safety related debug logic
- Register protection for software fault protection

### Cache features

The DSU-120AE has the following cache features:

- Optional unified 16-way set-associative L3 cache, configurable from 256KB to 32MB
- 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.
- Quick Nap support on the L3 data RAMs

## Coherency and snoop control

The DSU-120AE 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-120AE has the following cluster features:

- Support for Arm®v9.2-A architecture cores
- Support for up to two types of core, and a maximum of 14 cores in the cluster
- The DSU-120AE 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.
- *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.

## Interface features

The DSU-120AE has the following interface features:

- Optional AMBA 5 CHI Issue E 256-bit coherent requester bus interface, supports up to four CHI bus requester ports.
- Optional AMBA AXI5 Issue H 256-bit non-coherent manager bus interface, supports up to four AXI bus manager ports.
- Configurable address target group methodology for CHI and AXI bus manager ports. The address target groups are used to optimize the interconnect connectivity between the bus manager ports and the system.
- Optional 128-bit or 256-bit wide I/O-coherent *Accelerator Coherency Port* (ACP) interfaces based on AMBA ACE5-Lite. Supports up to two ACP interfaces.
- AMBA AXI5 utility bus providing programming interface to PPU, 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 Issue E 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-120AE 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 licensed separately.

---

### Related information

[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

## 1.2 DynamIQ™ Shared Unit-120AE configuration options

You must configure the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) RTL for your implementation requirements prior to hardware synthesis at build time configuration. Configuration for the DSU-120AE 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-120AE Configuration and Integration Manual* (DSU-120AE CIM).

---

The DSU-120AE configuration options include:

#### Number of cores

You can configure the cluster to have between two and 14 cores. Each core within a complex counts towards the total number of cores in the cluster. This is in addition to any cores in the cluster that are not in complexes (stand-alone cores).

#### Core type

You can have a cluster that includes up to two different types of cores. See [1.4 Cluster configurations](#) on page 38, for more information on the types of core that are supported.

#### 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
- 24MB
- 32MB



Setting the size of 0KB implements the DSU-120AE without an L3 cache, see [1.4.2 L3 memory system variants](#) on page 44.

---

### L3 cache slices

You can configure the DSU-120AE to have 1, 2, 4, or 8 cache slices. For more information on cache slices, see [6.8 Cache slices and power portions](#) on page 144.

### 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-120AE transport, see *RTL configuration process* in *Arm® DynamIQ™ Shared Unit-120AE 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 the DSU-120AE to have 1, 2, 3, or 4 bus manager interfaces.

### ACP interface

You can include up to two *Accelerator Coherency Port* (ACP) interfaces and specify their size.

### Peripheral port

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

### 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-120AE.



The ELA-600 is licensed separately.

---

## DCLS mode

Enables *Dual-Core Lock-Step* (DCLS) support. When this parameter is set to Lock-configuration or Mixed-configuration, the DCLS primary and redundant logic is present to support fault protection features.



When this parameter is set to Split-configuration, no DCLS logic is included, and the design behaves similarly to the DynamIQ Shared Unit-120.

---

## DCLS delays

Configures the temporal diversity between the primary and redundant logic. For more information about the `DCLS_DELAYS` values, see the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*. For more information about temporal diversity, see [1.3.3 Mixed-configuration](#) on page 24.

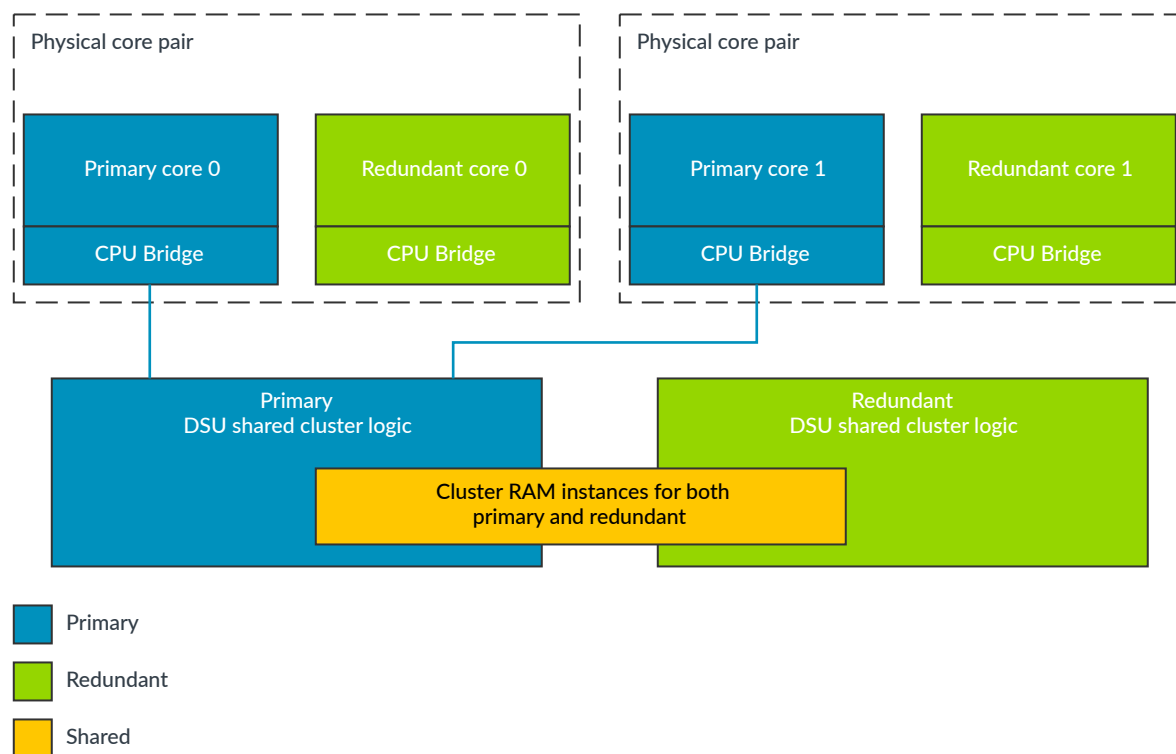
## 1.3 DCLS configurations

At build time configuration, you can configure the cluster to operate in one of the *Dual-Core Lock-Step* (DCLS) configurations where logic is duplicated and compared.

### 1.3.1 Lock-configuration

The arrangement where the DSU-120AE logic and AE cores are duplicated and comparators are instantiated is one possible build-time configuration of the DSU-120AE, known as Lock-configuration. In this configuration both the DSU cluster logic and the core or complex logic is instantiated as primary and redundant pairs of logic. The primary logic provides the functional behavior and the redundant logic is used for comparison purposes. The redundant copy of the logic employs a multi-cycle Temporal Delay (N) along with clock-tree diversity. The following figure shows an example arrangement of the DSU and the cores in Lock-configuration.

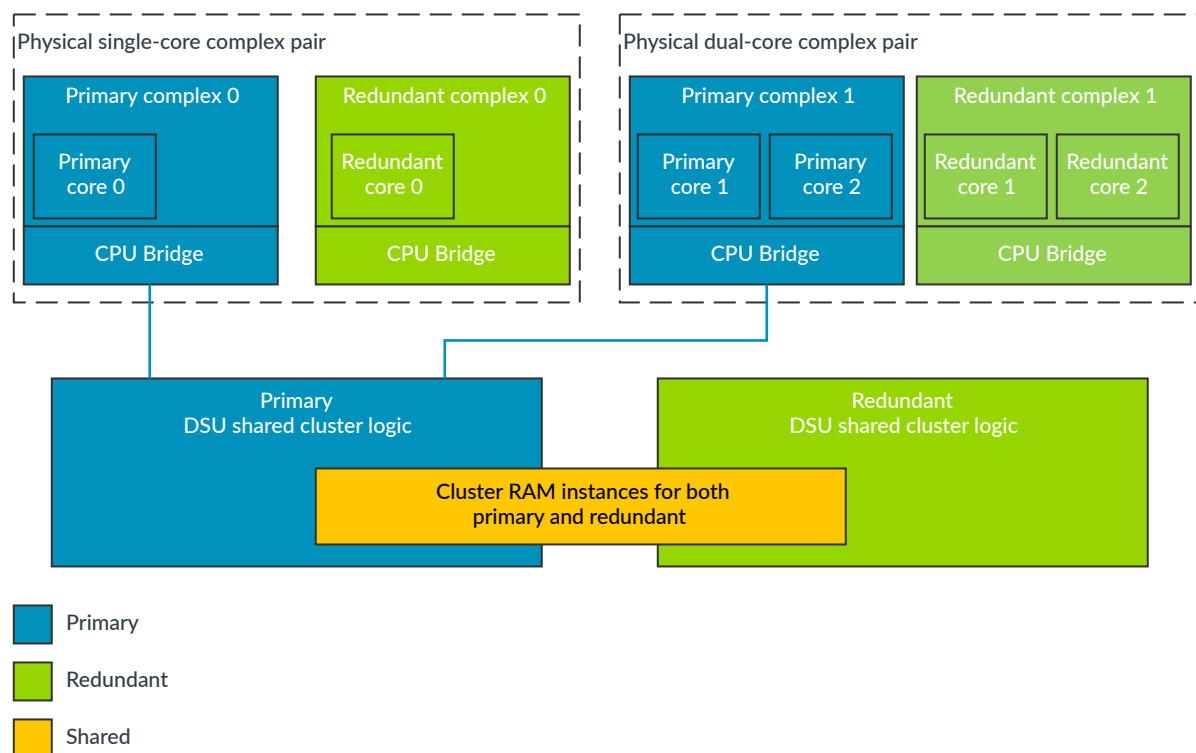
**Figure 1-5: Lock-configuration: core example**



- All cores have to be in core pairs and all complexes have to be in complex pairs in Lock-configuration.
- For comparison and interaction of the primary and redundant logic see [1.3.4 Primary and redundant logic comparison](#) on page 33.

The following figure shows an example arrangement of the DSU and the complexes in Lock-configuration.

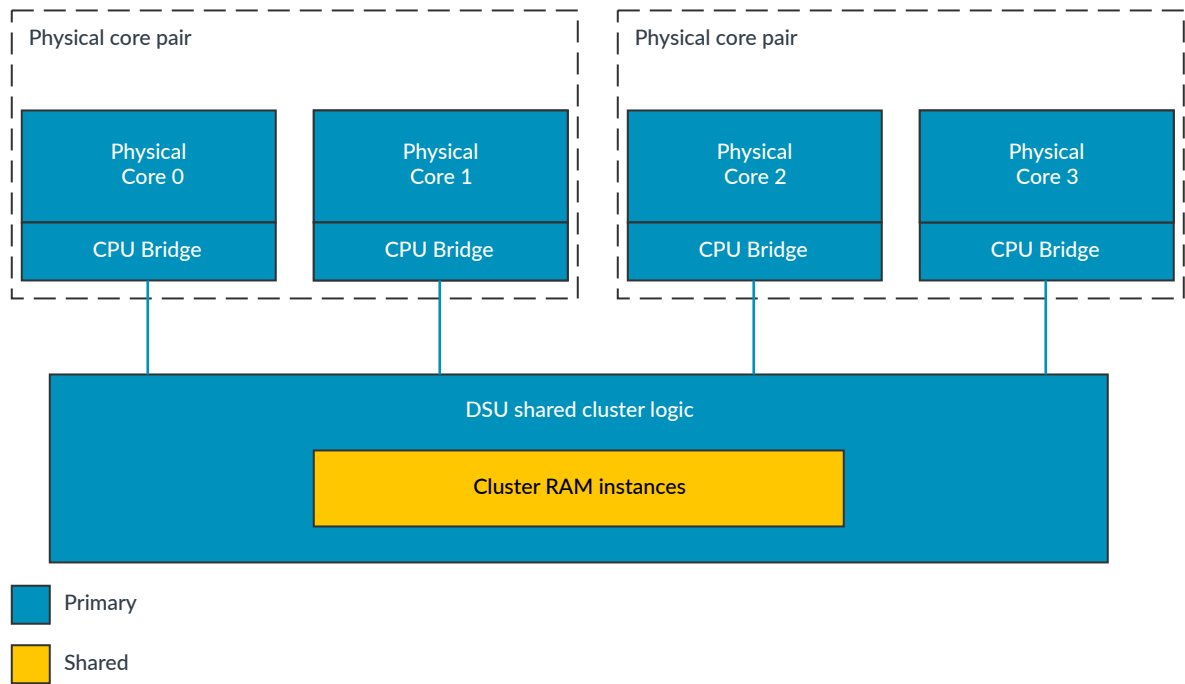
**Figure 1-6: Lock-configuration: complex example**



### 1.3.2 Split-configuration

Another possible build-time configuration is called Split-configuration. In this configuration there is no duplicate logic included for either the cluster or the cores or complexes. This means that there is no duplicate logic error detection capability included. Interface protection is not supported in Split-configuration. The following figure shows an example arrangement of the same DSU-120AE, and cores as in the preceding topic, but configured for Split-configuration.

**Figure 1-7: Split-configuration: core example**



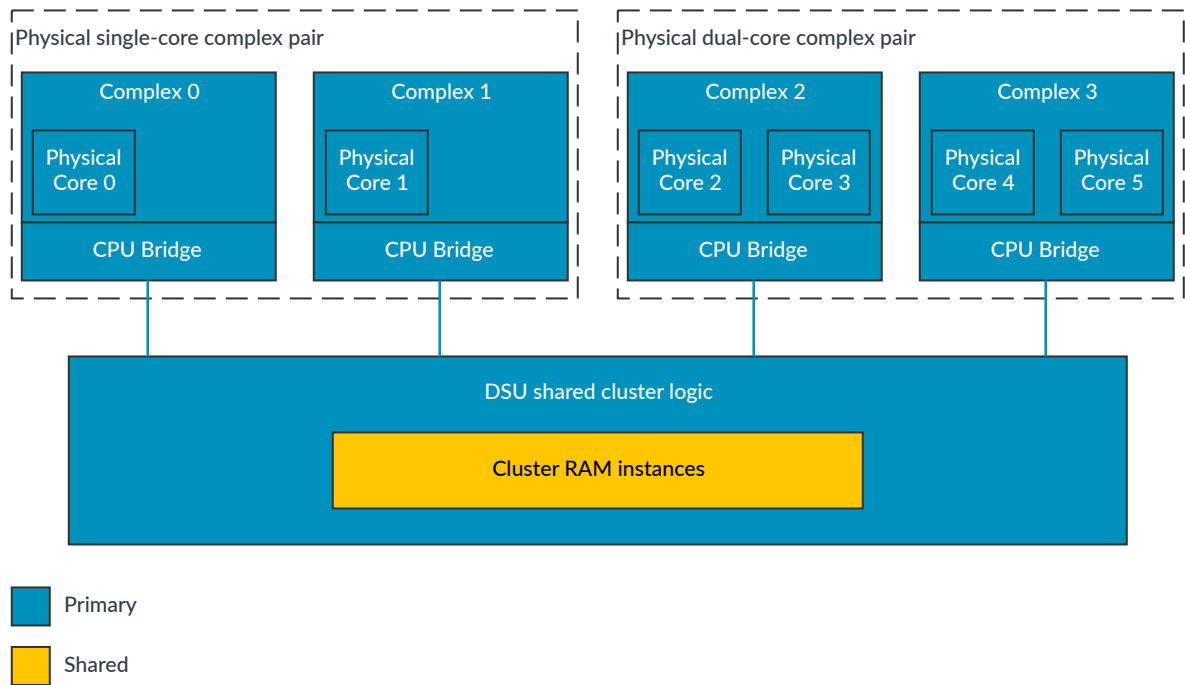
All cores have to be in core pairs and all complexes have to be in complex pairs in Split-configuration.

As can be seen from comparing the Lock-configuration and Split-configuration core example figures, core pairs consist of evenly numbered core instances. For example, core pair 0 consists of core 0 and core 1, and core pair 1 consists of core 2 and core 3.

In a Split-configuration, the DSU-120AE DynamIQ™ cluster can be configured to use between two and 14 cores. However, if using a Lock-configuration, the total number of cores configured is the number of primary cores, and ranges between one and seven cores. The DSU-120AE DynamIQ™ cluster can also be configured to use up to two 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.

In Split-configuration, the complexes are configured the same way as the cores. The following figure shows an example arrangement of the same DSU-120AE, and complexes configured for Split-configuration.

**Figure 1-8: Split-configuration: complex example**



### 1.3.3 Mixed-configuration

A third build-time DCLS configuration is called Mixed-configuration.

In Mixed-configuration, both the DSU cluster logic and the core or complex logic is instantiated as pairs of logic so that cores have to be instantiated in core pairs and complexes in complex pairs. The core pairs of logic can either be used as primary and redundant pairs of logic or each of the cores in the core pair can be used independently. The pair of cluster logic can either be used as primary and redundant pairs of logic or one of the cluster pairs can be unused. The core and cluster logic can be pin-configured at cluster reset to use three different modes.

- In Mixed-configuration Split-mode, each of the cores in the core pairs is used independently and only the primary cluster logic is used.
- In Mixed-configuration Lock-mode, the primary logic for both the cores and cluster provides the functional behavior and the redundant logic is used for comparison purposes.
- In Mixed-configuration Hybrid-mode, each of the cores in the core pairs is used independently but the cluster logic is used as primary and redundant pairs of logic.



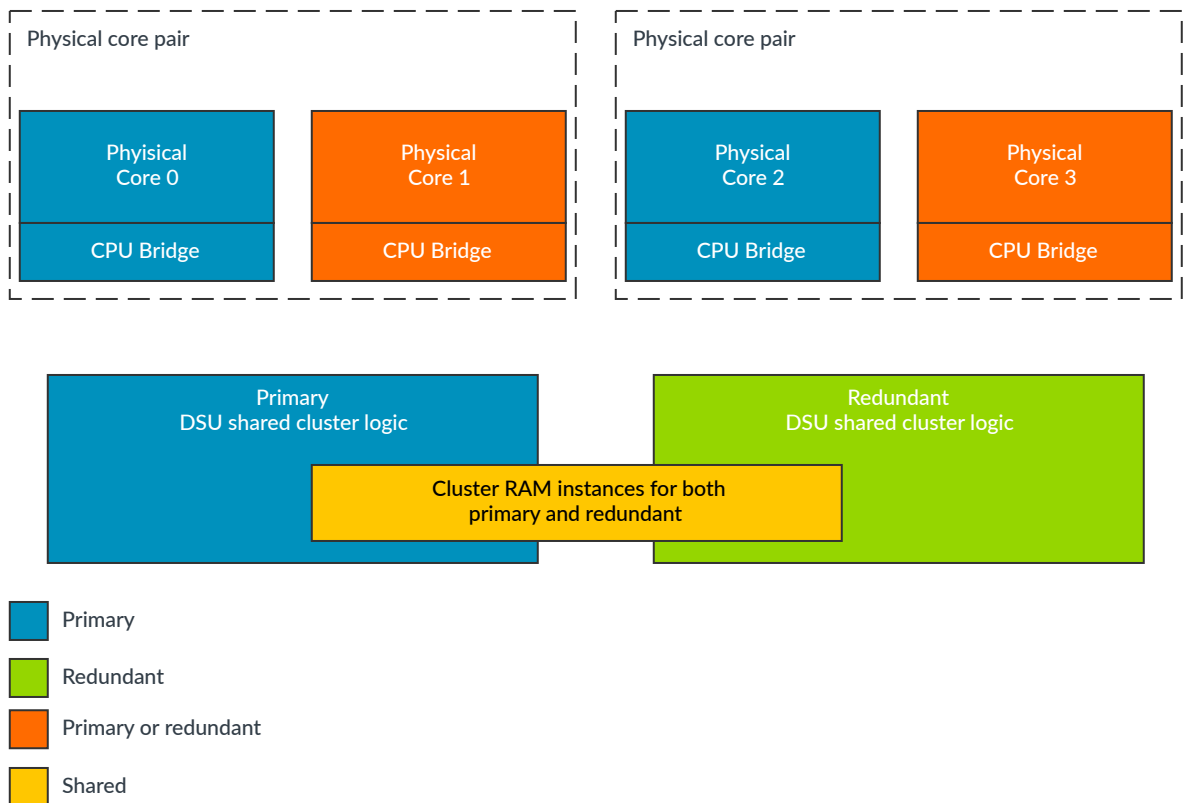
Both the Mixed-configuration Split-mode and Mixed-configuration Lock-mode functions similarly to the Split-configuration and Lock-configuration.



The pin-configurable flexibility of Mixed-configuration Split-mode has a tradeoff with circuit area because the redundant logic is still present but not used in this configuration.

The following figure shows an example arrangement of the logic duplication and core instances, when configured for Mixed-configuration.

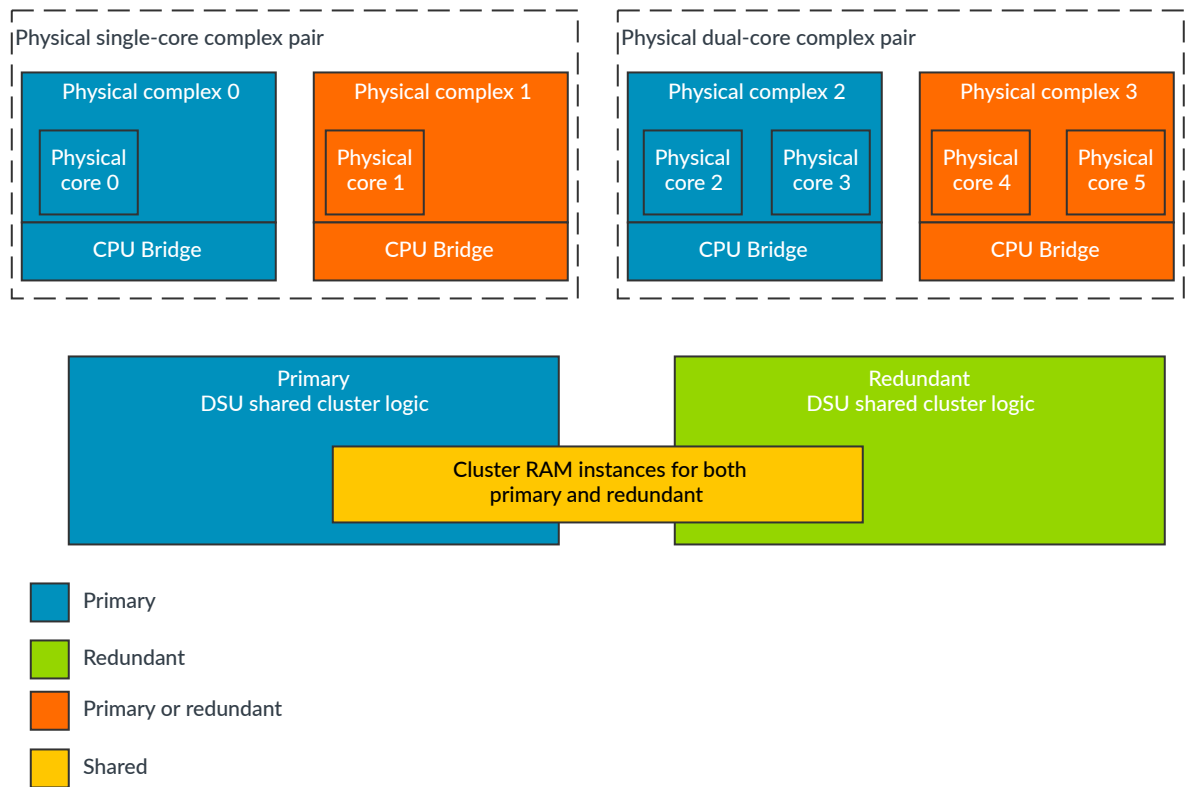
**Figure 1-9: Mixed-configuration: core example**



- All cores have to be in core pairs and all complexes have to be in complex pairs in all modes of Mixed-configuration.
- In Split-mode, all cores operate as primary logic.
- The redundant DSU logic is unused, when the DSU is in Split-mode.
- Mixed-configuration Split-mode will be referred to as Split-mode.
- Mixed-configuration Lock-mode will be referred to as Lock-mode.

The following figure shows an example arrangement of the logic duplication and complex instances, when configured for Mixed-configuration.

**Figure 1-10: Mixed-configuration: complex example**



For further details on the modes of Mixed-configuration, see the following topics below. For details on programming the different Mixed-configuration modes, see [5.8 Selecting different modes in Mixed-configuration](#) on page 117.

### 1.3.3.1 Split-mode (Mixed-configuration with cores split and DSU logic split)

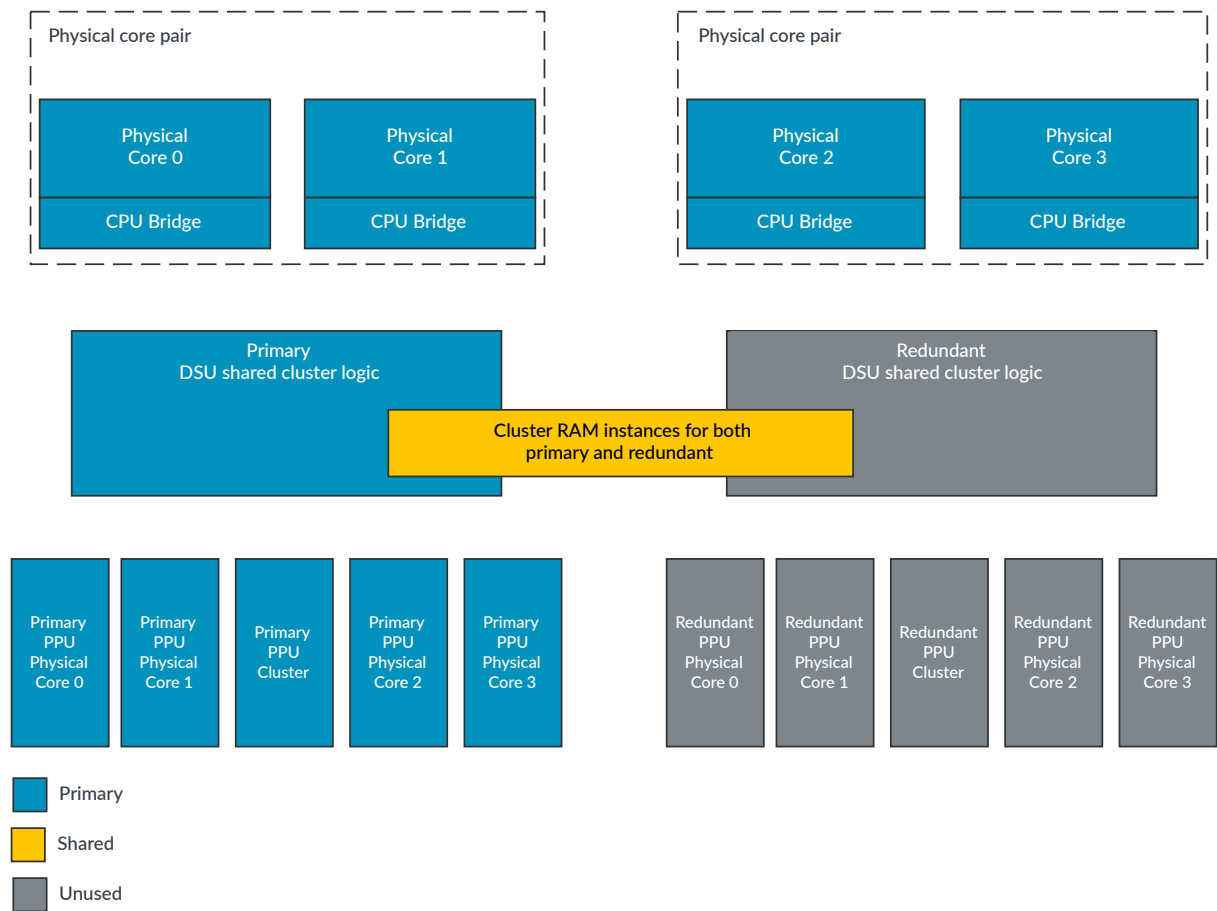
Split-mode has the same functionality as Split-configuration. In Split-configuration, the compute performance is doubled compared with the equivalent Lock-configuration and Lock-mode. Split-mode does not support *Dual-Core Lock-Step* (DCLS).

In Split-mode, each core is logically independent such that the cluster provides maximum compute throughput. The Split-mode sacrifices the increased fault tolerance of the primary and redundant logic for the compute performance, achieved by doubling the number of independent processor cores that are active. The redundant cluster logic is still present, but is not utilized and it is clock gated where feasible. The selection of Split-mode is controlled from the CLUSTERAE\_CLUSTERSLCTLR and CLUSTERAE\_CORESLCTLR registers, where the values of the registers are set using the CLUSTERSLDEFAULT and CORESLDEFAULT input signals. See the [B.1.6 External cluster AE registers summary](#) on page 600. The cores that support DCLS are provided as core pairs, that combine two cores together with two interfaces to the DSU. When operating in

Split-mode, the two cores in the core pair act independently. All cores are instantiated as core pairs, combining two cores together with two interfaces to the DSU.

The following figure shows an example arrangement of the DSU-120AE cluster and the core instances in Split-mode.

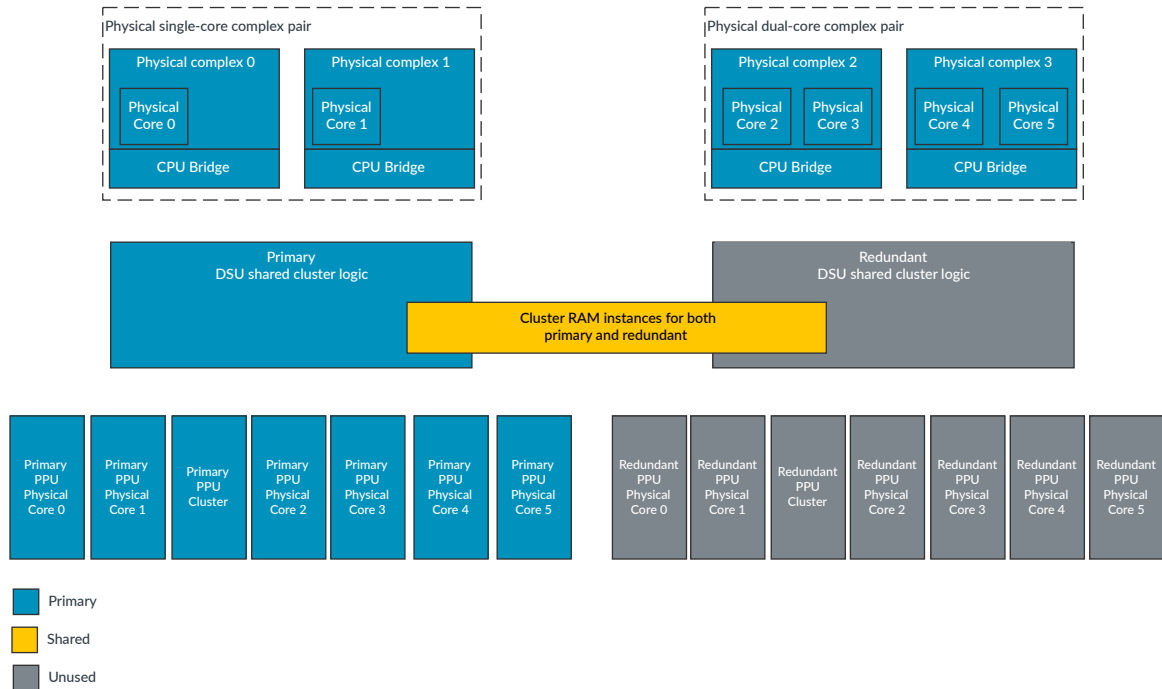
Figure 1-11: Mixed-configuration Split-mode: core example



If DSU-AE is executing in Split-mode, there are two CPU bridges per physical core pair. One is the primary, the other is redundant. In Split-mode they are both operating independently and both their cores are also operating independently.

The following figure shows an example arrangement of the Split-mode cluster overview with the complex instances.

**Figure 1-12: Mixed-configuration Split-mode: complex example**



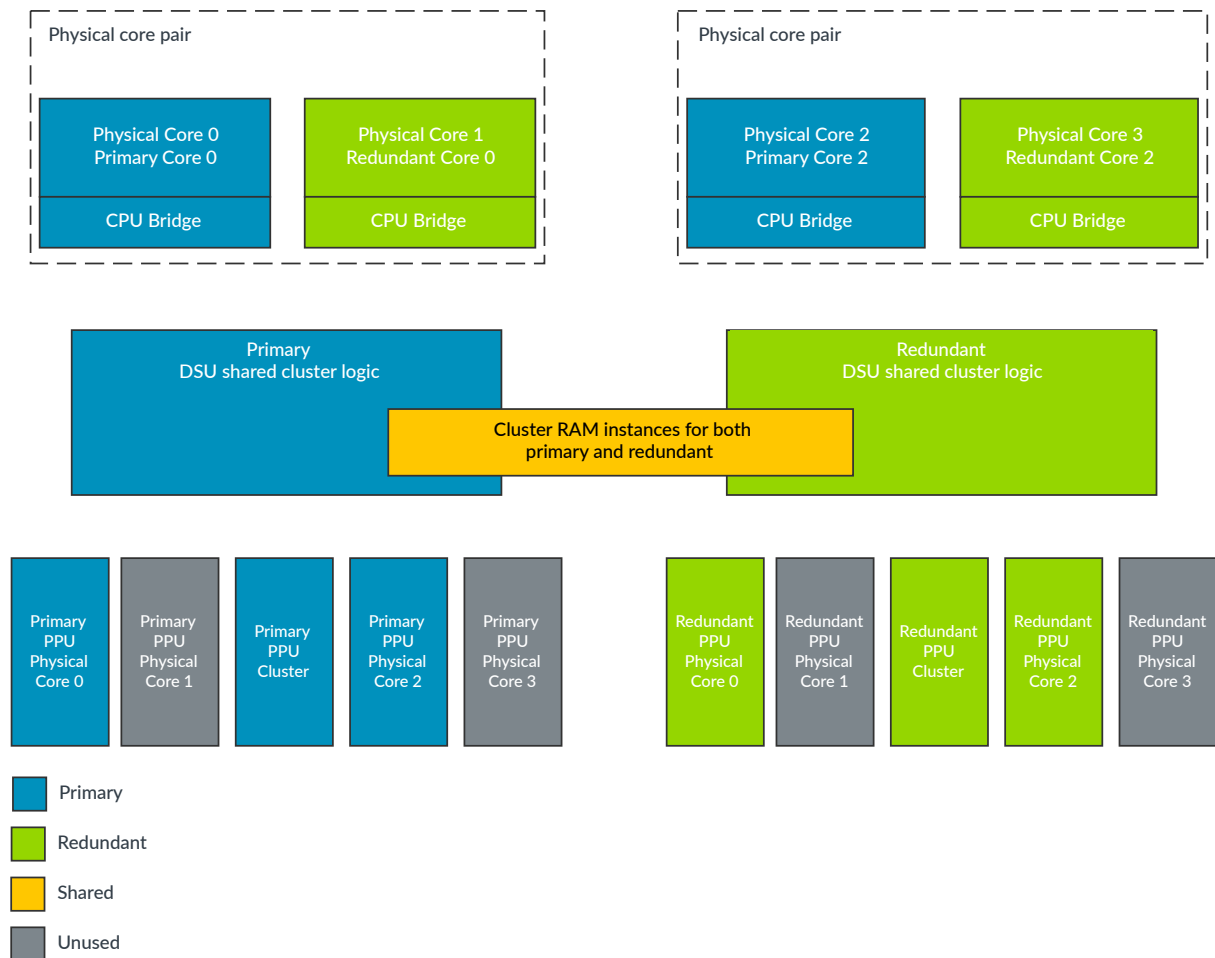
### 1.3.3.2 Lock-mode (Mixed-configuration with cores locked and DSU logic locked)

In Lock-mode, similar to Lock-configuration, the cluster and the cores work in *Dual-Core Lock-Step* (DCLS). This mode introduces primary and redundant logic. The comparators and timeout mechanisms catch the divergences between the primary and redundant logic.

In Lock-mode, the DSU-120AE and the cores are configured to operate in Lock-Step. In addition, each pair of cores is logically viewed as a single core by the DSU. While the total number of cores available for compute performance is halved, the resulting configuration provides a complete *Dual-Core Lock-Step* (DCLS) solution for the whole cluster, significantly improving the fault detection capabilities. The selection of Lock-mode is controlled from the CLUSTERAE\_CLUSTERSLCTLR and CLUSTERAE\_CORESLCTLR registers, where the values of the registers are set using the CLUSTERSLDEFAULT and CORESLDEFAULT input signals. See the [B.1.6 External cluster AE registers summary](#) on page 600. The cores that support DCLS are configured as core pairs, that combine two cores together with two interfaces to the DSU logic. When operating in Lock-mode, the second core becomes the redundant copy of the first (primary) core, and comparators check the outputs of the two cores match. The second core's interface to the DSU is unused and looks to the software as if the second core is powered off. The redundant copy of the logic employs a multi-cycle Temporal Delay (N) along with clock-tree diversity.

The following figure shows an example arrangement of the cluster and core instances in Lock-mode.

**Figure 1-13: Mixed-configuration Lock-mode: core example**

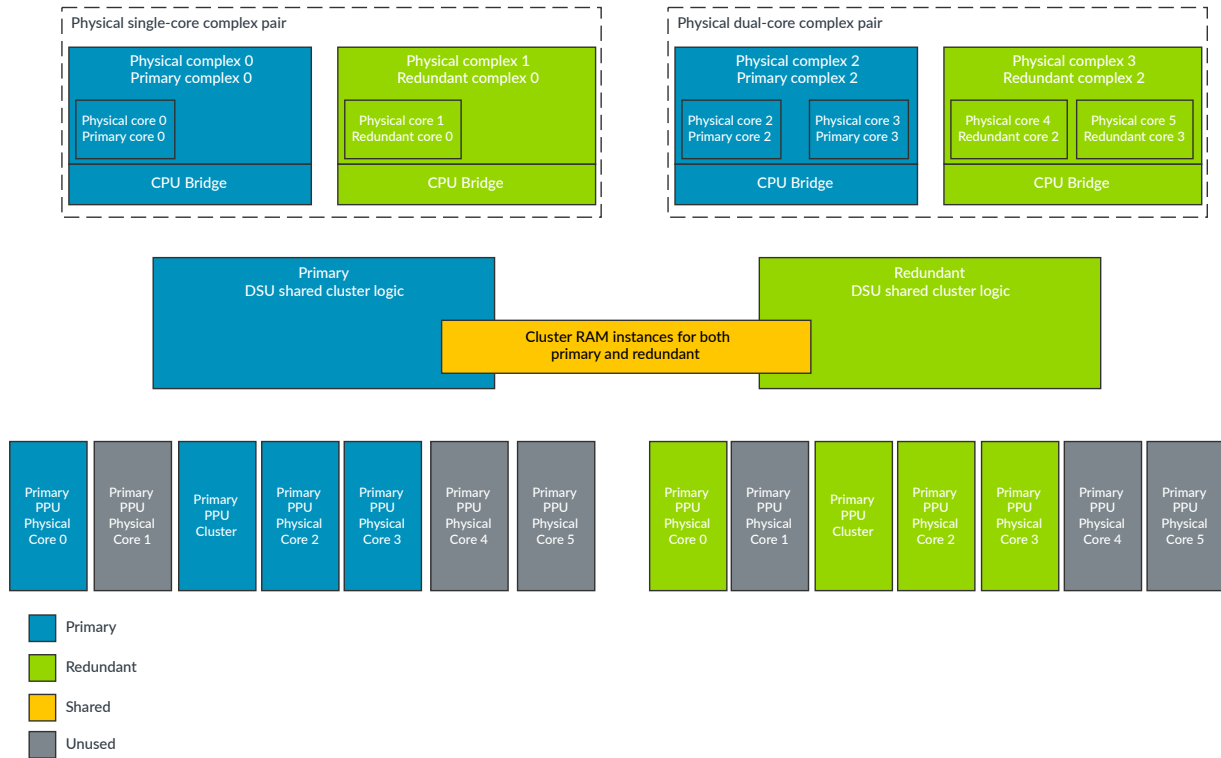


Note

- Logical core and complex numbering is always based on the number of physical cores and complexes. For more information about logical core numbering, see [1.8 Core, complex, and processing element numbering](#) on page 47.
- Any attempts to access core 1 and core 3 will respond as if the core is powered down and the wakeup requests to these cores will not have any effect.
- The unused *Power Policy Units* (PPUs) are still present and can still be accessed by software.

The following figure shows an example arrangement of the Lock-mode cluster overview and complex instances.

**Figure 1-14: Mixed-configuration Lock-mode: complex example**



Note

- Logical core and complex numbering is always based on the number of physical cores and complexes. For more information about logical core numbering, see [1.8 Core, complex, and processing element numbering](#) on page 47.
- Any attempts to access core 1 in complex 1 or core 4 and 5 in complex 3 will respond as if the core is powered down and the wakeup requests to these cores will not have any effect.
- The unused *Power Policy Units* (PPUs) are still present and can still be accessed by software.

### 1.3.3.3 Hybrid-mode (Mixed-configuration with cores split and DSU logic locked)

Hybrid-mode is a Mixed execution mode where the cores are configured to execute independently (similar to Split-mode) while the DSU-120AE is configured to execute in Lock-Step (similar to Lock-mode). All of the duplicate DSU logic is used in Hybrid-mode, including the redundant shared cluster logic and the redundant *Power Policy Unit* (PPU) instances for all of the cores.

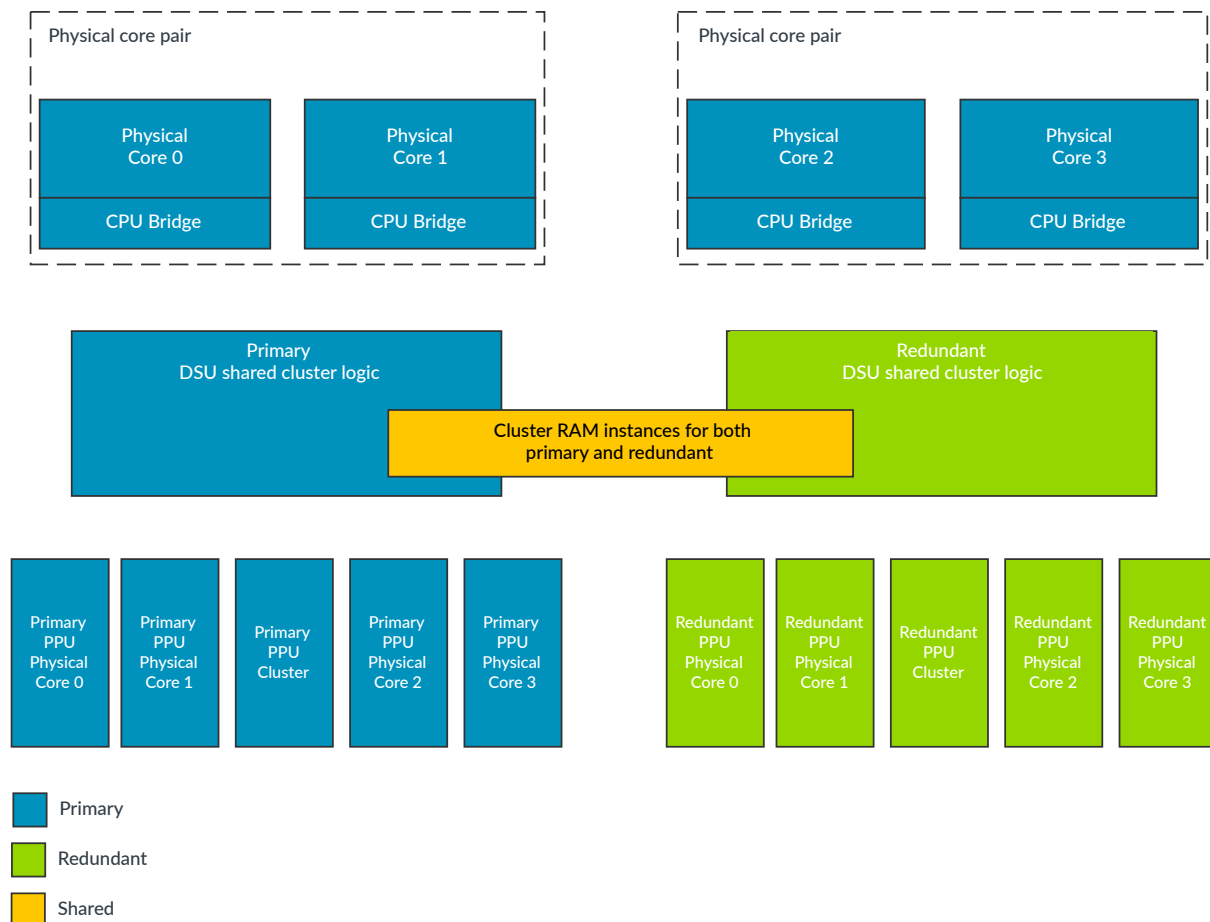
In Hybrid-mode, the cluster provides the following partial *Dual-Core Lock-Step* (DCLS) solutions:

- The Hybrid-mode allows better cluster performance relative to the Lock-mode because there are more cores operating independently of each other.
- The Hybrid-mode allows better fault detection in the cluster relative to Split-mode as the DSU-120AE is configured to execute in Lock-Step.

The Hybrid-mode can help the cluster availability in some applications. These applications might require a certain level of fault detection which is beyond what Split-mode can provide, but do not need the full cost of Lock-mode. This level of fault detection could be achieved by mechanisms such as logic *Built-In Self Test* (BIST) or Software Test Libraries (STLs). It is possible to perform these activities on one core at a time, which means that the remaining cores in the cluster are still available for running the workload. However, when the shared cluster logic requires these activities, the whole cluster becomes unavailable for the duration. This unavailability of all cores in the cluster might be unacceptable for some use cases, therefore Hybrid-mode allows the shared logic to have DCLS to give sufficient fault detection capabilities without requiring *Logic Built-In Self Test* (LBIST) or STLs to run on the shared logic, while the cores can still run in Split-mode. The selection of Hybrid-mode is controlled from the CLUSTERAE\_CLUSTERSLCTLR and CLUSTERAE\_CORESLCTLR registers, where the values of the registers are set using the CLUSTERSLDEFAULT and CORESLDEFAULT input signals. See the [B.1.6 External cluster AE registers summary](#) on page 600.

The following figure shows an example arrangement of the DSU-120AE cluster and the core instances in Hybrid-mode.

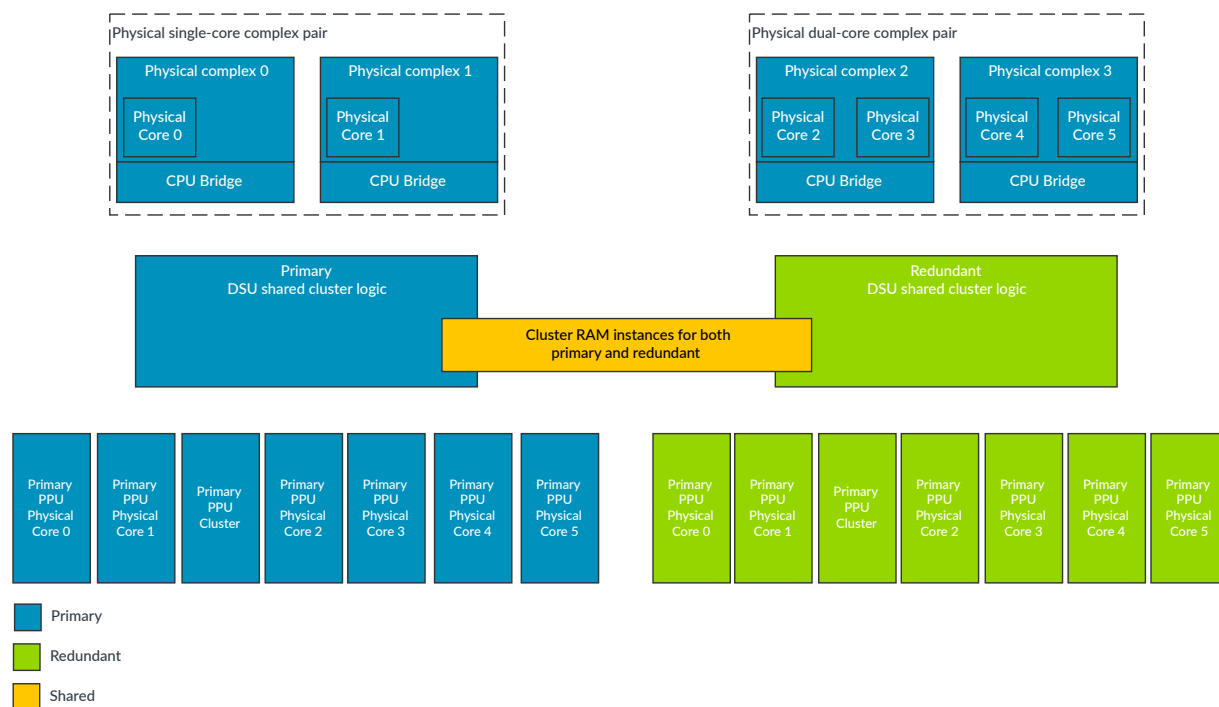
**Figure 1-15: Mixed-configuration Hybrid-mode: core example**



The following figure shows an example arrangement of the Hybrid-mode cluster overview and complex instances.



**Figure 1-16: Mixed-configuration Hybrid-mode: complex example**

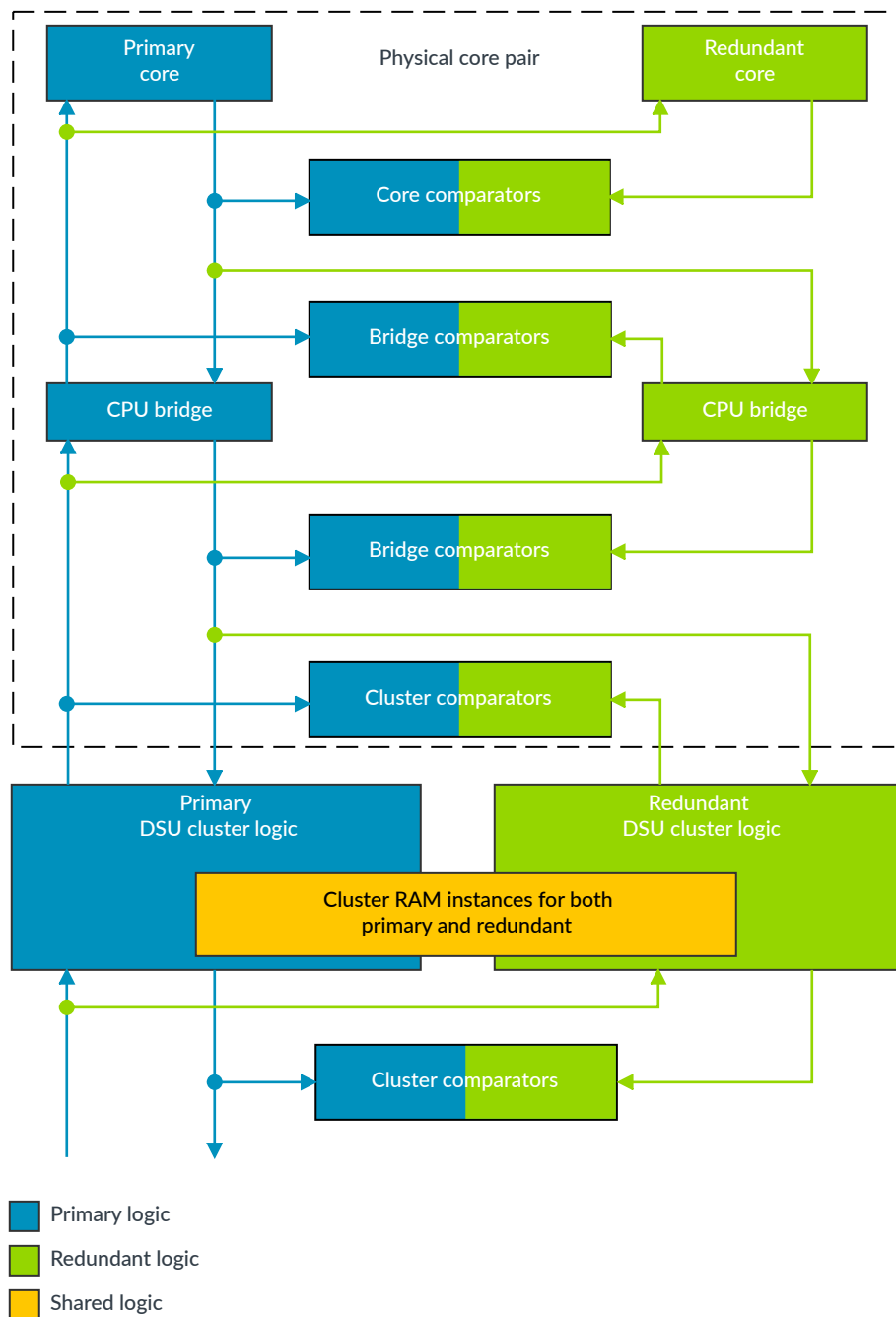


### 1.3.4 Primary and redundant logic comparison

When the DSU-120AE is configured in Lock-configuration or Mixed-configuration and is using either Lock-mode or Hybrid-mode, then both the primary and redundant parts of the DSU-120AE logic are active. Both copies of the DSU-120AE logic outputs feed comparators to check for any

behavioral difference between the primary and redundant logic. The following figure shows the comparators with the cluster logic.

**Figure 1-17: Comparators overview**



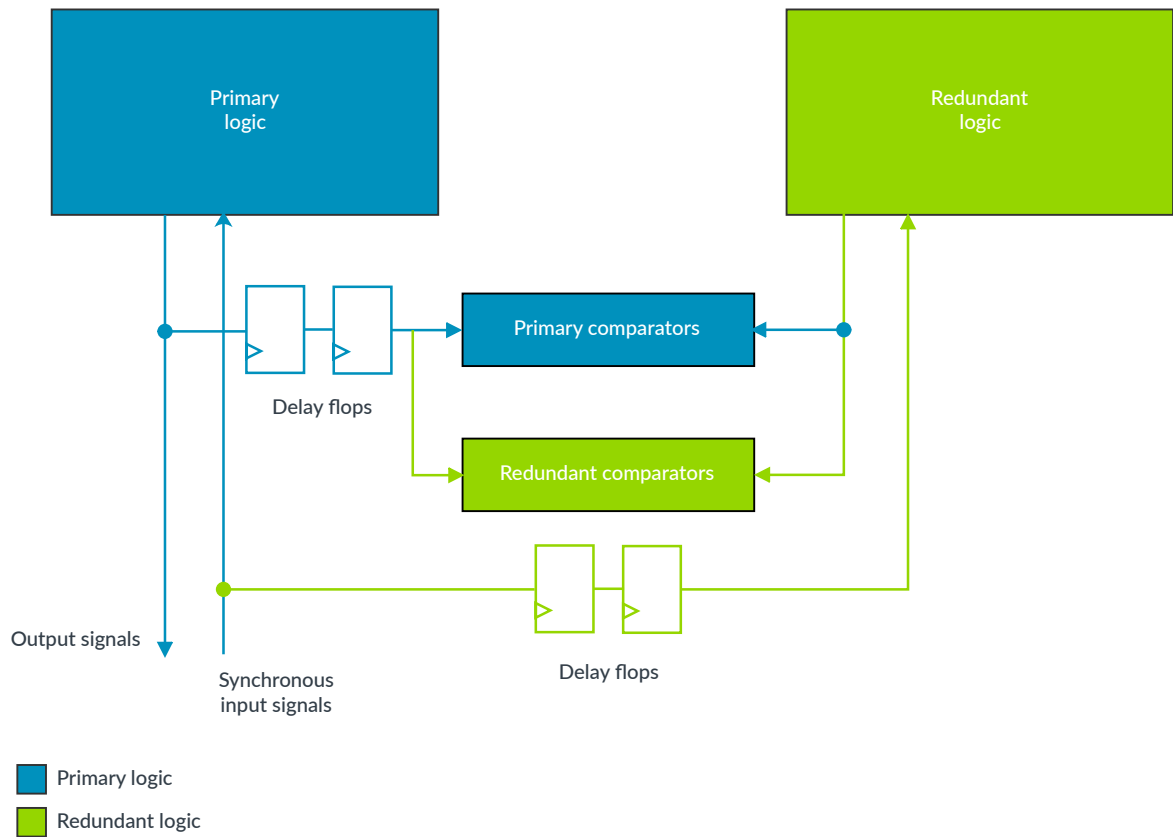


Note that this figure is a conceptual representation of how the primary and redundant logic is compared using the comparators. It does not represent the exact hierarchical implementation or connectivity of the comparators.

---

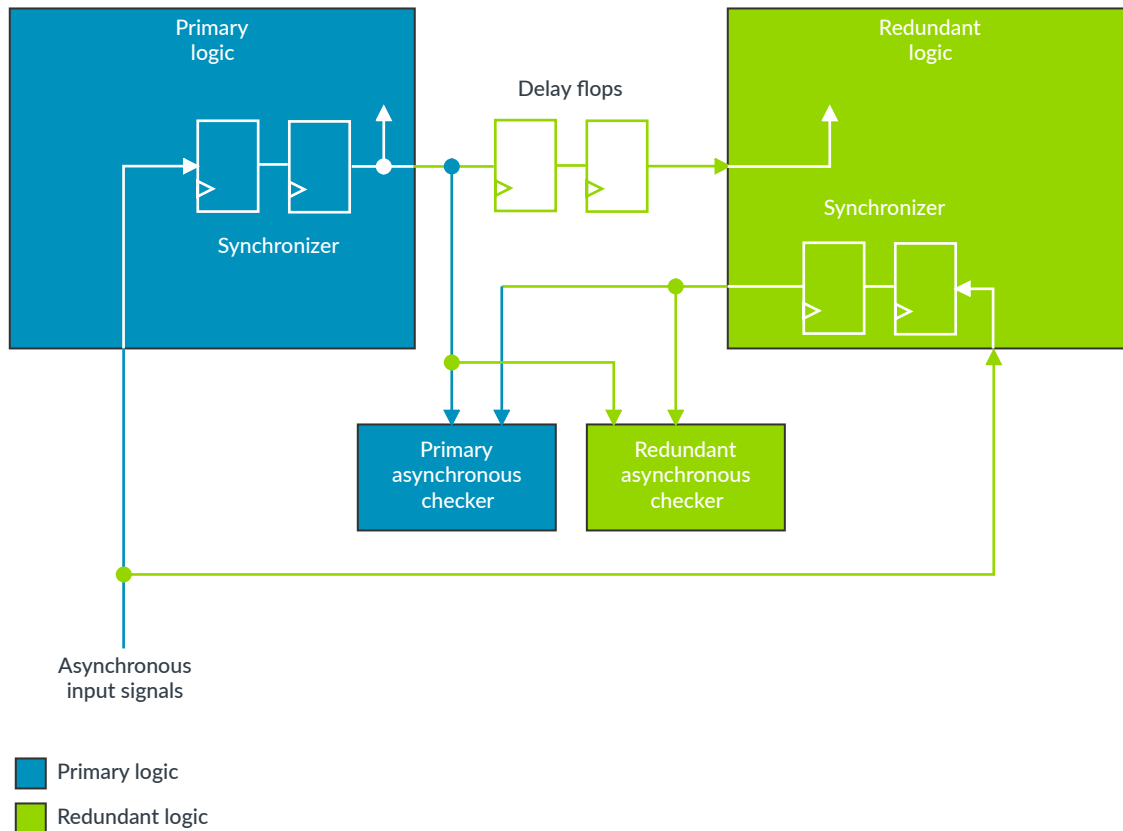
The same inputs are driven to both the primary and to the redundant logic. However, the signals driving the redundant logic are delayed by a configurable number of clock cycles compared to the signals driving the primary logic. This delay is added so that when there is a common physical event that affects both the primary and the redundant logic, the delay ensures that the primary and redundant logic behavior is more likely to differ. For example, in the case of a common physical event such as a significant drop or rise in the power supply that generates a fault in the behavior of the logic, then the delay between the primary and the redundant logic operation ensures that the generated fault has a different impact on the primary and redundant logic behavior, so the primary and redundant logic behavior will differ. For synchronous inputs to the logic, this is done by adding additional register stages on the signals driving the redundant logic. The primary logic outputs are delayed by the same number of delay stages as the redundant logic inputs, so the comparators observe identical behavior in both the primary and in the redundant logic as long as no physical faults occur in either the primary or redundant logic. Then the primary logic outputs can be compared with the redundant logic outputs. The following figure shows the timing delays for the synchronous signals.

**Figure 1-18: Timing delays for the synchronous signals**



When there are asynchronous inputs both to the primary and redundant logic, it is necessary that the delay between the primary and redundant operation is identical to the synchronous signal delay. This means that the asynchronous signal must be synchronized once and then the synchronizer output is used to drive both the primary and the redundant logic. However, the redundant logic includes a duplicate synchronizer for the asynchronous signal and the duplicated synchronizer output can be compared against the primary synchronizer output to check synchronization faults. The asynchronous comparison checks for identical outputs from both of the synchronizers, but it tolerates the differences in the cycle accuracy behavior due to the differences in the asynchronous signal synchronization. The following figure shows the timing delays for the asynchronous signals.

**Figure 1-19: Timing delays for the asynchronous signals**



The comparison logic drives output error signals from the cluster to indicate if there is any divergence between the primary and redundant logic behavior.



**Note**

Note that in case of a fault, there is no way to define if the fault arises from the primary or from the redundant logic. Comparators can only detect behavior divergences between the primary and redundant logic, but comparators cannot identify the correct behavior.

The comparison logic itself is also fully duplicated, where one set of comparison logic is running in the primary logic clock domain and the other set of comparison logic is running in the redundant logic clock domain. This means that there are two separate comparisons made for each output signal. The results of both comparators are output from the cluster, so the system-level logic can detect any behavior divergence in between the two sets of comparison logic.

The following types of control exist for the comparator logic:

#### Enable signal

There is an enable signal for the comparators to control whether the comparators report any detected errors on the comparator output signals.

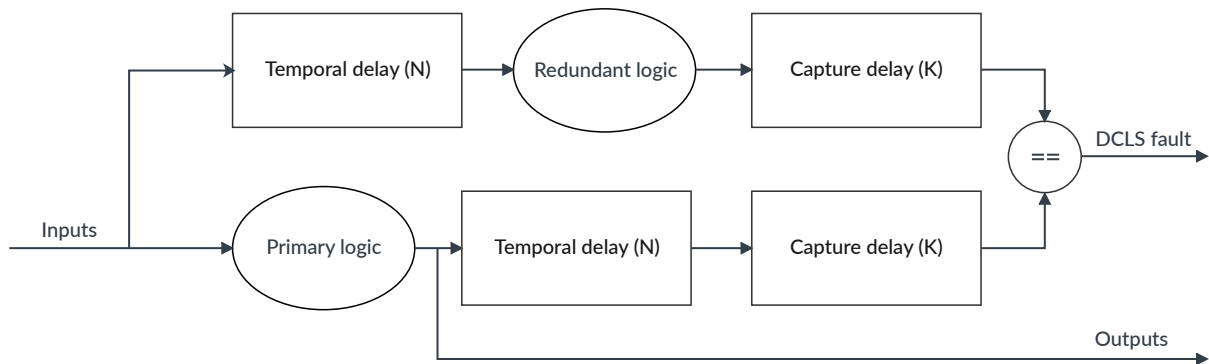
## Error injection control signals

There are error injection control signals to force each comparator output to report an error.

This permits the testing of system-level logic that monitors the output signals to confirm that the monitoring logic will detect if an error is reported from the comparator.

The delay between the primary and redundant logic is set by a configuration option when the cluster is implemented. In addition to the delay between the primary and redundant logic, the configuration option also controls the number of additional register stages to assist with timing closure for the physical design. The following figure shows the primary and redundant logic temporal and capture delays.

**Figure 1-20: Primary and redundant logic Temporal delays and Capture delays**



## 1.4 Cluster configurations

A cluster can be configured with up to two different types of cores in the same cluster, irrespective of whether the cluster is configured either for Lock-configuration or Mixed-configuration. Each core can target different power efficiency and performance levels. The cluster also supports complexes.

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

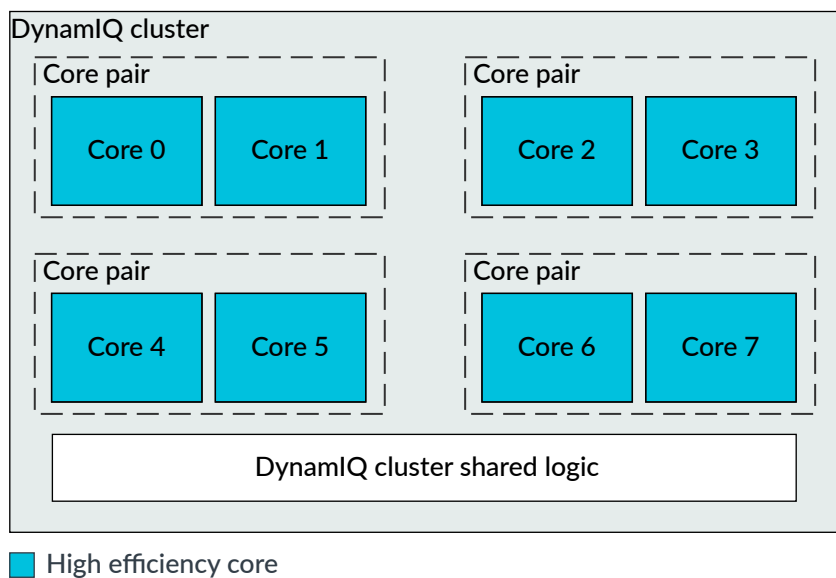
- One or more cores of the same type.
- Various arrangements of two types of cores. For example, one or more cores targeting either a high-performance level or a higher power efficiency level.
- Various arrangements of three of cores. For example, one or more high-performance cores, power-efficient cores, and intermediate cores.
- One or more complexes and no individual cores. For information on complexes, see [1.4.1 What is a complex?](#) on page 42.
- One or more complexes and individual cores.



For clusters, configured in Lock-configuration, a maximum of seven cores can be configured.

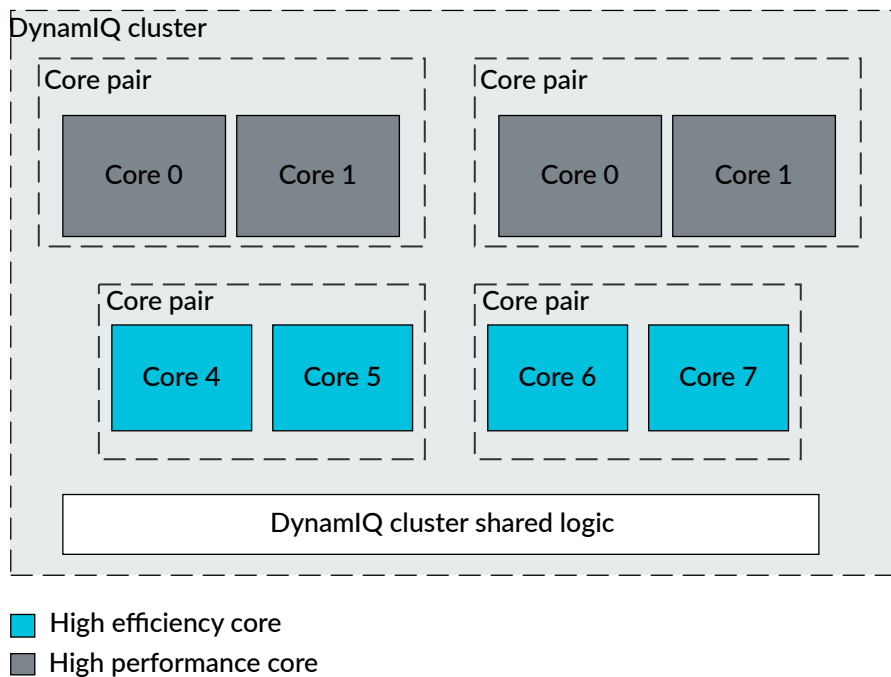
The following figure shows a cluster that is configured with all the same type of core.

**Figure 1-21: DynamIQ cluster with one type of core**



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

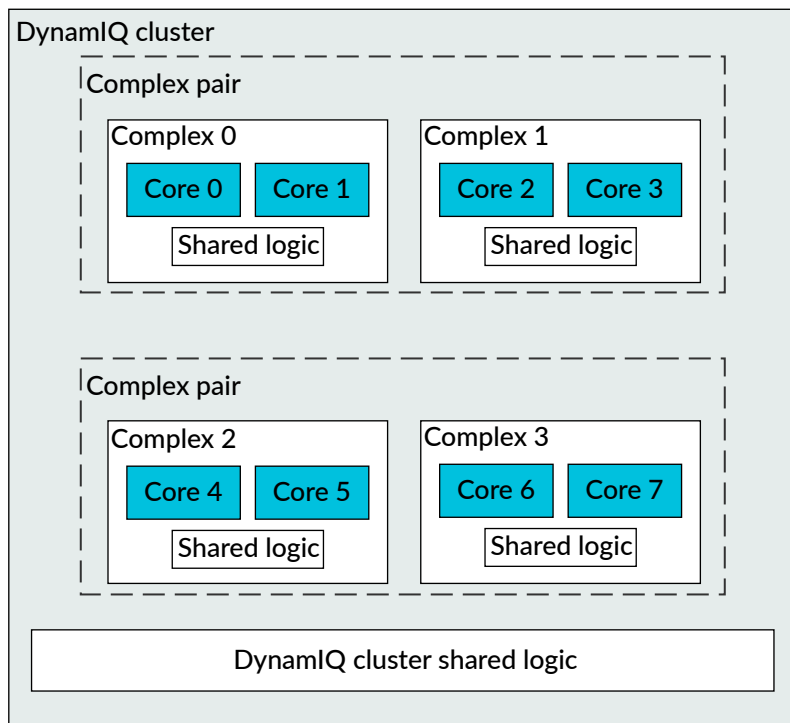
**Figure 1-22: DynamIQ cluster with two types of cores**



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



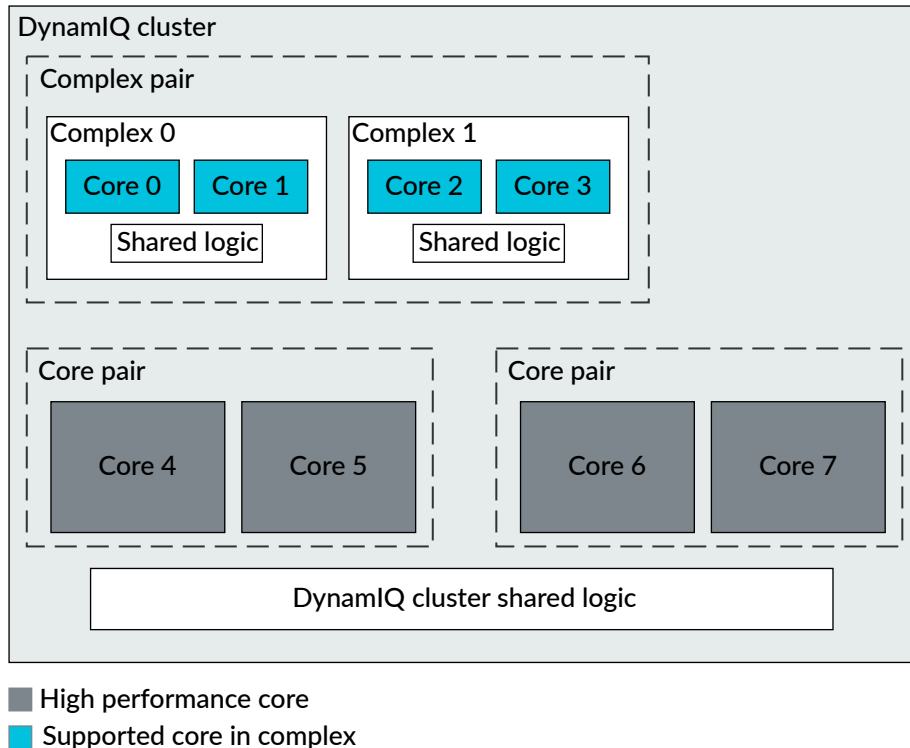
**Figure 1-23: DynamIQ cluster with four complexes**



■ Supported core in complex

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

**Figure 1-24: DynamIQ cluster with two complexes and four high-performance cores**



**Note**

All DSU-120AE-compatible cores support use in a multi-core cluster. Any combination of these cores should be configurable in the cluster provided that:

- In Split-configuration and Split-mode, the maximum number of cores is 14.
- In Lock-configuration and Lock-mode, the maximum number of cores is seven.
- There are no more than two different types of core in the cluster.

### 1.4.1 What is a complex?

The DSU-120AE 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.

The DSU-120AE DynamIQ™ cluster also supports complex pairs, where one of the two complexes is a primary complex and the other one a redundant complex. Both the primary and redundant complexes must be identical to one another. A complex pair contains two complexes where each complex can have either a single core or a dual core as shown in the figure below.



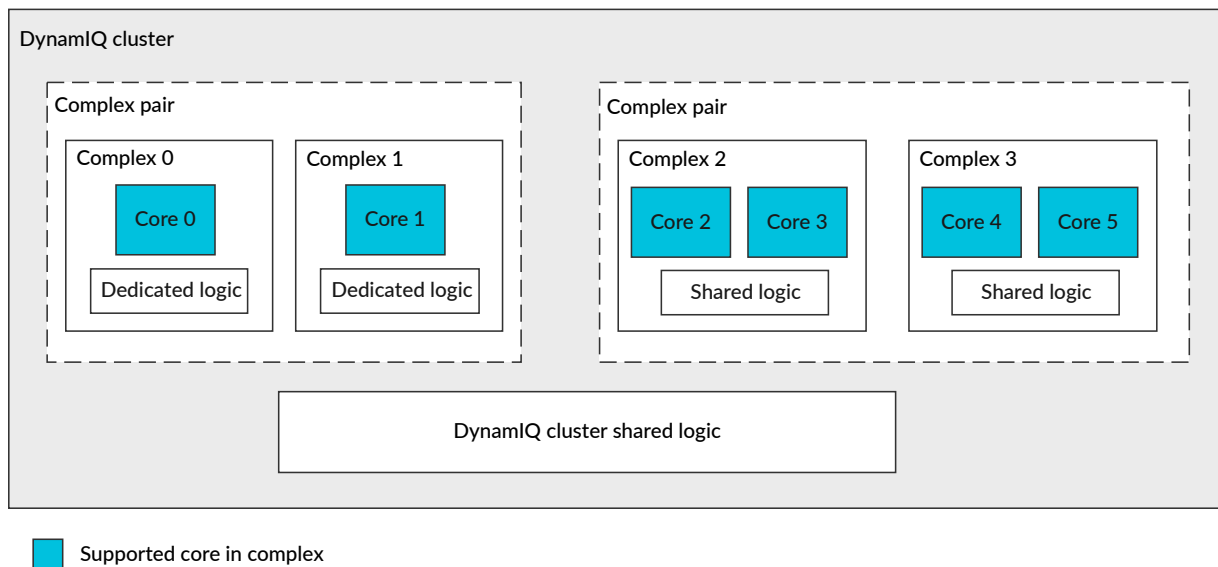
Only certain types of cores which have a merged-core microarchitecture can be used in a complex. To see if your core is supported in a complex and for further details of complexes, see your core *Technical Reference Manual* (TRM).

The maximum number of cores instantiated in the cluster is 14. 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 dual core complexes.

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

**Figure 1-25: Cluster with a dual-core complex pair and a single-core complex pair**



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 functionality of the cores, 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, appropriate for emphasis, where functionality applies to both standalone cores and complexed cores it is also stated.

---

## Related information

[1.4 Cluster configurations](#) on page 38

[1.8 Core, complex, and processing element numbering](#) on page 47

### 1.4.2 L3 memory system variants

By default the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) is implemented with an L3 cache. Depending on your requirements, you can instead implement the DSU-120AE without an L3 cache.

There are two 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* requester 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.

For more information on how to implement the DSU-120AE with one of the L3 memory system variants, see the *Configuration Guidelines* chapter in *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

## Related information

[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

[1.1 DynamIQ Shared Unit-120AE features](#) on page 16

## 1.5 Supported standards and specifications

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) complies with the Arm®v9.2-A architecture and all previous Arm®v8-A architectures up to Arm®v8.7-A.



The DSU-120AE is compatible with the architecture for the supported cores in the cluster. See the section *Supported standards and specifications* in your core *Technical Reference Manual* (TRM) for a list of specific architectural versions and features that the cores support.

The DSU-120AE complies with the architectures listed in the following table.

**Table 1-1: Standards and specifications support in the DSU-120AE**

Standard of specification	Version	Notes
Arm architecture	Arm®v9.2-A	The DSU-120AE supports cores based on the Arm®v9.2-A architecture. These cores also support previous Arm®v8-A architectures up to Arm®v8.7-A, dependent on the core.  See the section <i>Supported standards and specifications</i> in your core <i>Technical Reference Manual</i> (TRM) for details.
FEAT_RAS, Reliability, Availability, and Serviceability (RAS)	RAS v8.4	The DSU-120AE supports RAS features that conform to the v8.4 RAS architecture, see <a href="#">11. RAS extension support</a> on page 199.
Advanced Microcontroller Bus Architecture (AMBA)	<ul style="list-style-type: none"> <li>AMBA 5 CHI Issue E</li> <li>AMBA AXI5 Issue H</li> <li>AMBA APB5 Issue D</li> </ul>	For more information on the AMBA protocols supported by the DSU-120AE interfaces, see <a href="#">2.4 Interfaces</a> on page 57.
CoreSight™ architecture	v3.0	For more information on CoreSight™ architecture, see the <a href="#">Arm® CoreSight™ Architecture Specification v3.0</a> .
Debug	Arm®v9.2-A	Arm®v9.2-A architecture that is implemented with Arm®v8.4-A Debug architecture support and Arm®v8.3-A FEAT_DoPD, Debug over PowerDown support.  See FEAT_DoPD in the <a href="#">Arm® Architecture Reference Manual for A-profile architecture</a> for information on this architectural feature.

Standard of specification	Version	Notes
FEAT_GICv4p1, Generic Interrupt Controller (GIC) architecture CPU interface and Stream Protocol interface.	GICv4.1	The DSU-120AE uses Affinity level 1 to distinguish between different cores. This level is not supported by some interrupt controllers, such as GIC-500.  For information on FEAT_GICv4p1, see <a href="#">Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4</a> .
Performance Monitoring Unit (PMU)	PMUv3	-

## Related information

[2.2 DynamIQ cluster shared logic components](#) on page 52

[2.4 Interfaces](#) on page 57

## 1.6 Test features

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE. Additionally, internal *Memory Built-In Self Test* (MBIST) interfaces are provided to test the L3 cache and other memory arrays of the DSU-120AE.

The DSU-120AE includes an ATPG test interface that provides signals to control the *Design for Test* (DFT) features of the DSU-120AE, 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-120AE 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 the *Design for Test integration guidelines* chapter in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*. For information about the test signals related to your core, see your core *Configuration and Integration Manual*.

## 1.7 Design Tasks

Both the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) and the cores in the cluster are delivered as synthesizable RTL descriptions in Verilog HDL. Before you can use the DSU-120AE 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-120AE 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-120AE 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-120AE 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-120AE and the cores in the cluster by programming values into registers. The configuration choices affect the behavior of the DSU-120AE 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-120AE Configuration and Integration Manual*

## Related information

[13. System control registers](#) on page 215

# 1.8 Core, complex, and processing element numbering

A cluster contains two or more cores. The cluster can also contain two 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. The instance numbering in different configurations and modes are explained as follows:

- In Lock-configuration the core numbering is based on the number of logical core instances. A Lock-configuration cluster that consists of two core pairs has two logical cores and so the cores are numbered 0 and 1.
- In Split-configuration and Mixed-configuration the core numbering is based on the number of physical core instances.
- In Mixed-configuration, Lock-mode there is only one logical core active in each core pair and so the logical core numbering matches the physical numbering of the primary core in each core pair. In a Mixed-configuration, Lock-mode with two core pairs the numbering of the logical cores will be 0 and 2.

## 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. The instance numbering in different configurations and modes are explained as the follows:

- In Lock-configuration the complex numbering is based on the number of logical complex instances. A Lock-configuration cluster that consists of two dual core complex pairs has two logical complexes and four logical cores and so the complexes are numbered 0 and 1 and the cores are numbered 0, 1, 2, 3.
- In Split-configuration and Mixed-configuration the complex numbering is based on the number of physical complex instances.
- In Mixed-configuration, Lock-mode there is only one logical complex active in each complex pair and so the logical complex numbering matches the physical numbering of the primary complex in each complex pair. In a Mixed-configuration, Lock-mode with 2 dual-core complex pairs the numbering of the logical complexes will be 0 and 2 and the numbering of the logical cores will be 0, 1, 4, 5.

## Processing element

The Arm architecture allows for cores to support multiple *Processing Elements* (PEs).



The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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).

The parameter PE is the total number of PEs in the cluster, starting from one. This numbering also includes cores within complexes. For example, PE = 6 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 PE-1. For example, when referring to the second PE, *z* = 1.



In the current DSU-120AE, each core only has one PE. Therefore, PE = CN+1.

For more information on the instance numbering for cores and complexes in the cluster, see *RTL configuration process* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

## 1.9 Product revisions

The product revision increments at each release.

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

**Table 1-2: Product revisions**

Revision	Notes
r0p0	First release
r0p1	Maintenance updates

Changes in functionality that have an impact on the documentation also appear in [Revision history](#) on page 1008.

## 2. Technical overview

A *DynamIQ™ Shared Unit-120AE* (DSU-120AE) cluster-based system is also known as a DSU-120AE.

The DSU-120AE comprises two top-level modules, these are:

### **A module to form a DSU-120AE DynamIQ™ cluster**

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

### **A separate module for the DebugBlock**

Separating the debug components from the DSU-120AE 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.

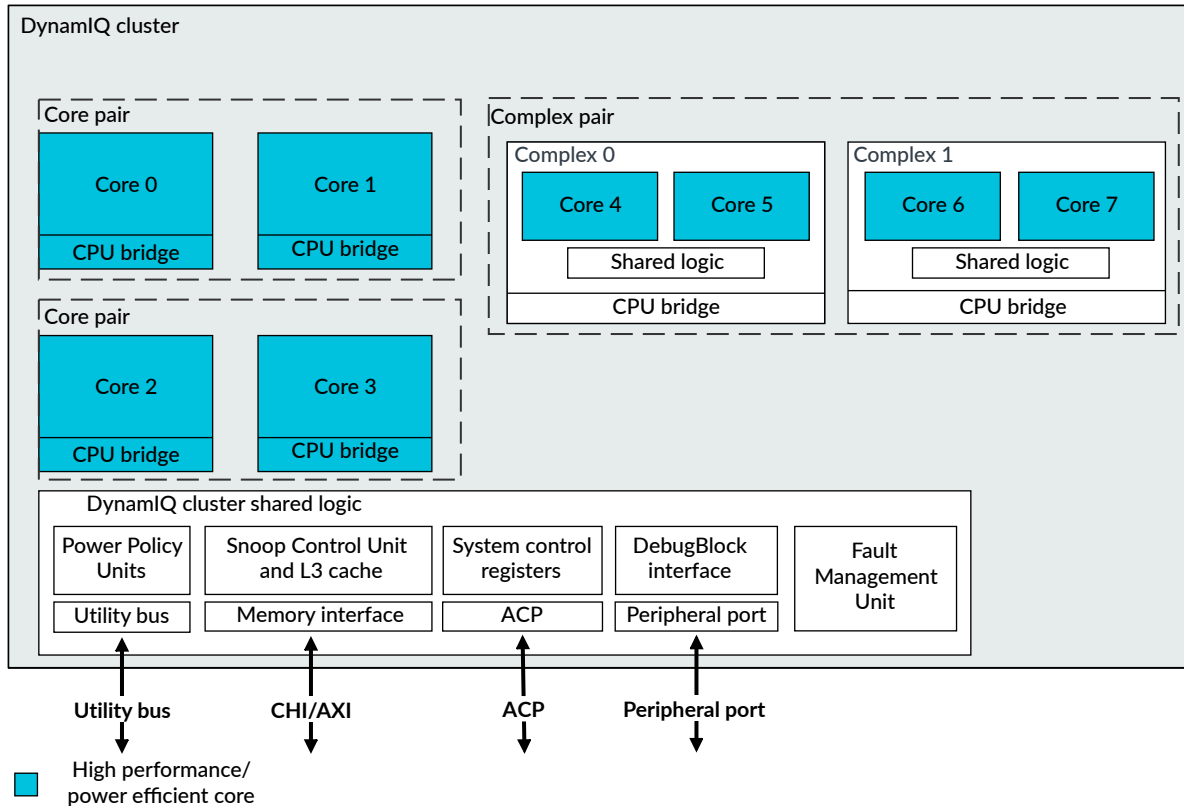
All the main *System on Chip* (SoC) interfaces appear at the top level of the DSU-120AE. The DSU-120AE connects the cores and complexes to an external memory system and the rest of the SoC.

## 2.1 DynamIQ cluster components

The DSU-120AE 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.

The following figure shows the main components that make up the DSU-120AE DynamIQ™ cluster within the DSU-120AE.

**Figure 2-1: DSU-120AE DynamIQ™ cluster components**



## Cores

The DSU-120AE DynamIQ™ cluster supports up to two different types of cores. The maximum number of supported cores depends on the build-time Dual-Core Lock-Step (DCLS) configuration as follows:

- For Lock-configuration, a maximum of seven cores
- For Split-configuration or Mixed-configuration, a maximum of 14 cores.

For information on the behavior and features of each core, see the *Technical Reference Manual* (TRM) of each core.

## Complexes

The DSU-120AE DynamIQ™ cluster supports complexes a maximum of 14 cores organized in complex-pairs. These complex-pairs can make up a pair of single-core complexes or a pair of dual-core complexes. Both of the complexes in the complex-pair have to be in identical arrangement.

A single cluster can have a mixture of both cores and complexes. The maximum number of cores in a cluster is 14, regardless if the cores are in the form of single cores or cores within a complex. Complexes are made up of specialized cores. See [1.4.1 What is a complex?](#) on page 42. See the

*DSU-120AE dependent features* 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-120AE DynamIQ™ cluster. See [2.2 DynamIQ cluster shared logic components](#) on page 52.

### Related information

[2.2 DynamIQ cluster shared logic components](#) on page 52

[2.3 DebugBlock components](#) on page 56

[1.4 Cluster configurations](#) on page 38

[1.4.1 What is a complex?](#) on page 42

## 2.1.1 Integration of the cores in the cluster

When you implement a DSU-120AE DynamIQ™ cluster, all interfacing between the cores, complexes, and the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) is implemented automatically. All the external signal inputs and outputs pass through the DSU-120AE. The DSU-120AE 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-120AE L3 memory system. Where necessary, the DSU-120AE implements additional buffering to compensate for different clock rates of the core and DSU-120AE L3 memory system.

Each core has an external clock interface, which is routed through the DSU-120AE to the respective core.

## 2.2 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.

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 the 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-120AE. 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

### Main memory requester

The main memory requester provides an interface between the DynamIQ™ Shared Unit-120AE 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 bus requester ports.

### Accelerator Coherency Port

The *Accelerator Coherency Port* (ACP) is an optional subordinate 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. Up to two ACP interfaces can be configured, with each interface configured to either 128-bit wide or 256-bit wide.

### Peripheral port

The peripheral port is an optional requester interface. It provides accesses to tightly coupled accelerators and it separates the traffic to the system. The port implements the AXI5 or CHI Issue E requester interface protocol.

### Utility bus

The utility bus is a 64-bit AXI5 subordinate 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:

- *Power Policy Units* (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 2-1: 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
24MB	12-way
32MB	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 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-120AE 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-120AE
- L3 cache hit and miss count information

- Status of the pin-configured modes of the Mixed-configuration for the DSU-120AE logic and cores.

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.

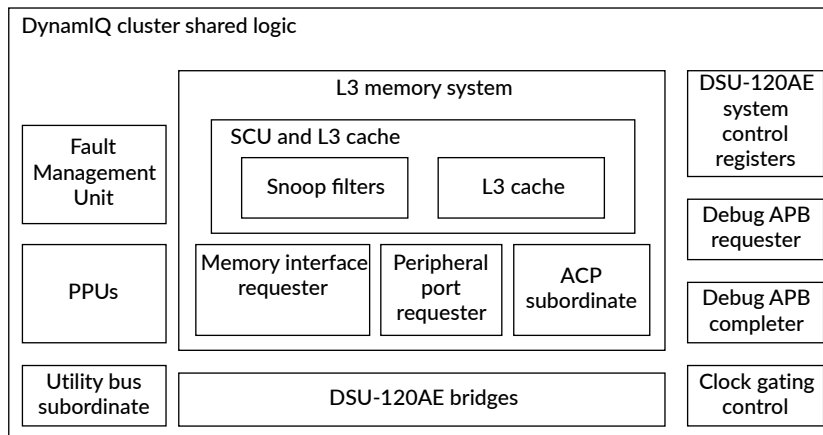
Trigger events from the cores are combined and output to the DebugBlock. Trigger events to the cores, and Debug register accesses, are received in the DebugBlock.

## Fault Management Unit

The *Fault Management Unit* (FMU) is utilized as an organizational structure to manage faults that are generated in the underlying units. There are comparators in various places of the DSU logic. The FMUs are responsible for bringing together all the faults and combining them into the output signals.

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

**Figure 2-2: DynamIQ™ cluster shared logic components**



## Related information

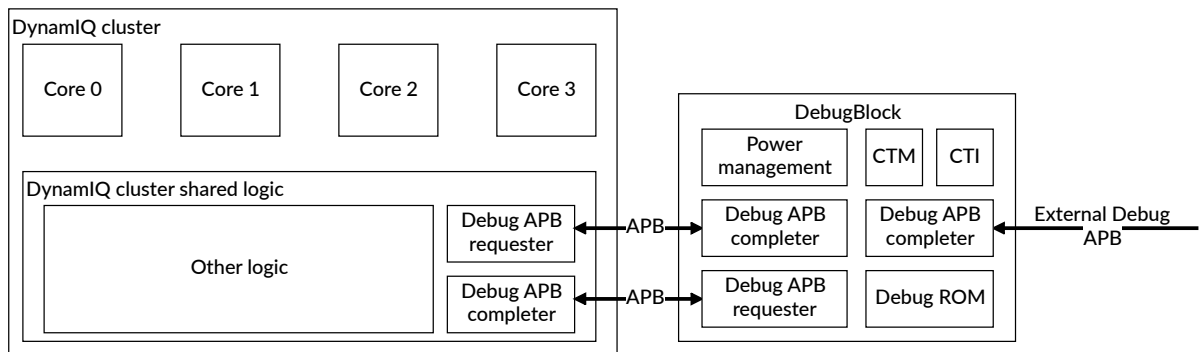
3. [Clocks and resets](#) on page 64
4. [Power management](#) on page 70
6. [L3 cache](#) on page 136
7. [CHI requester interface](#) on page 146
8. [AXI manager interface](#) on page 162
10. [AXI or CHI requester peripheral port](#) on page 183
13. [System control registers](#) on page 215
12. [Utility bus](#) on page 211
14. [Debug](#) on page 217

## 2.3 DebugBlock components

The DebugBlock is a dedicated debug component for the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) but is instantiated as a separate unit to support debug over powerdown.

The following figure shows the main components of the DebugBlock.

**Figure 2-3: DebugBlock components**



### Cluster (requester) to DebugBlock APB (completer)

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

### DebugBlock (requester) to cluster APB (completer)

Trigger events to the cores are transferred as APB writes to the DSU-120AE. Register accesses from the system debug APB are transferred to the DSU-120AE.

### System debug APB

The system debug APB completer 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.

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

[14. Debug](#) on page 217

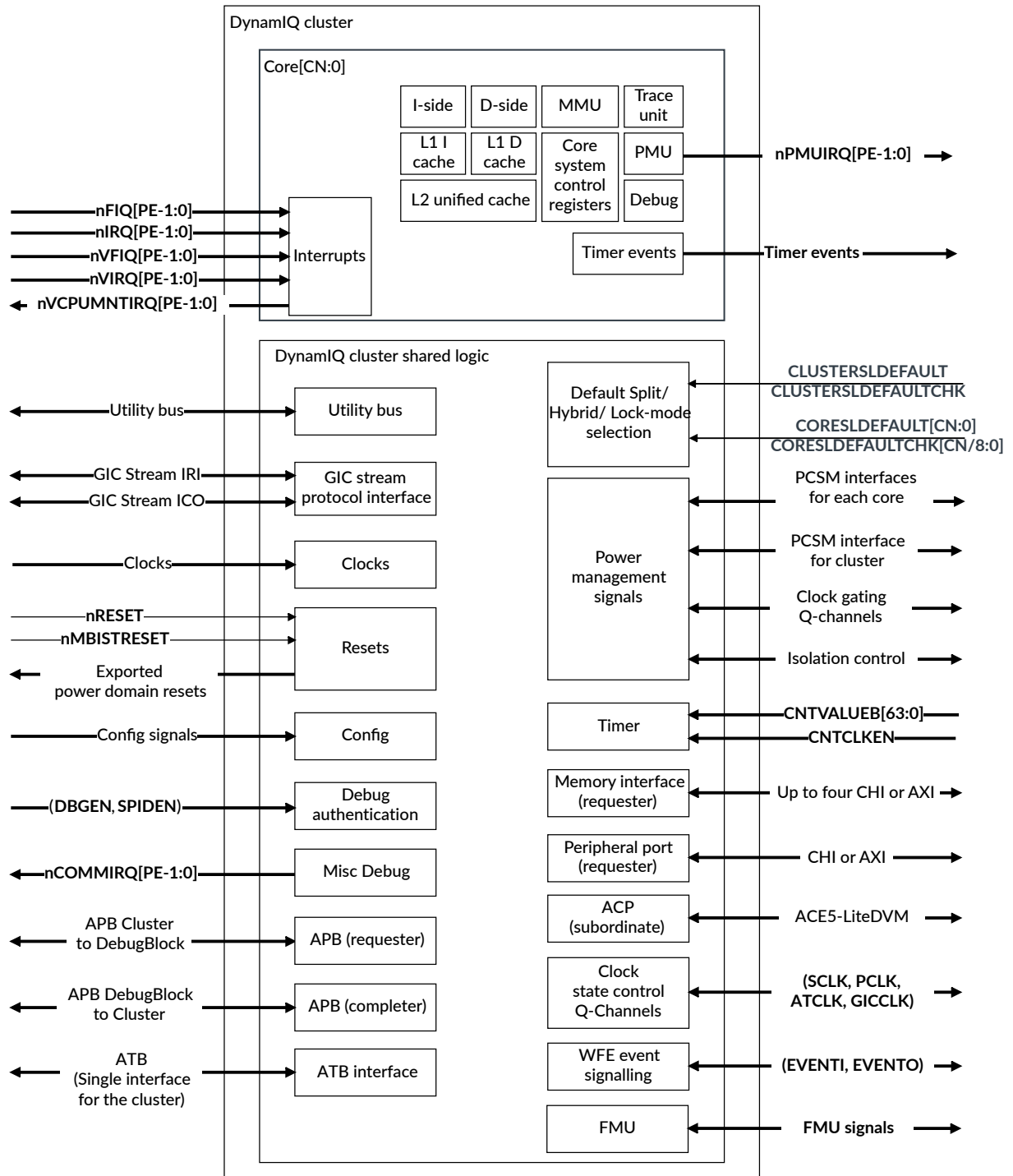
# 2.4 Interfaces

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) manages all the external interfaces to the *System on Chip* (SoC) including those from the cores and complexes in the cluster.

## DSU-120AE interfaces

The following figure shows the major external interfaces of the DSU-120AE DynamIQ™ cluster.

**Figure 2-4: DSU-120AE DynamIQ™ cluster interfaces**



The following table describes the external interfaces of the DSU-120AE DynamIQ™ cluster.

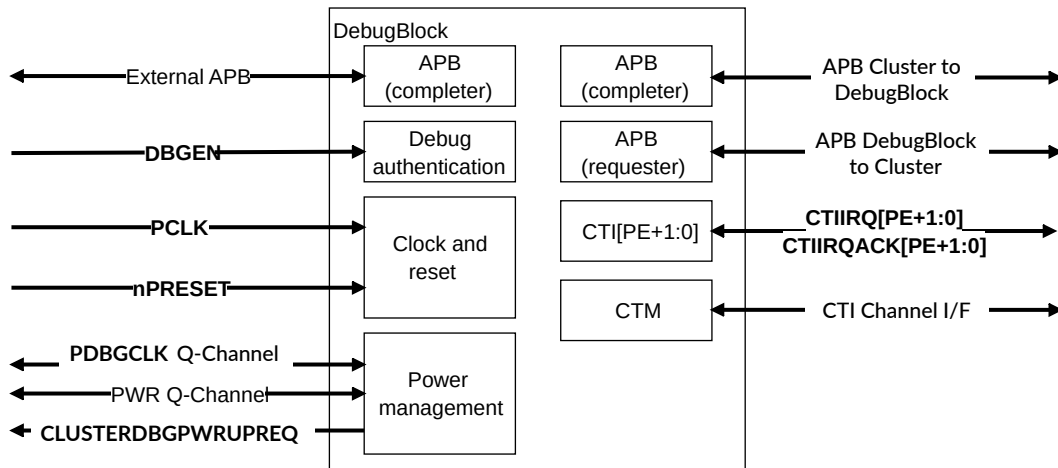
**Table 2-2: DSU-120AE DynamIQ™ cluster interfaces**

Purpose	Protocol	Notes
Trace	ATB	Transmitter ATB interface. This is a single interface for the whole cluster.
Memory	AMBA AXI5 or CHI Issue E	Requester interface to main memory. You can configure the DSU-120AE with either: <ul style="list-style-type: none"> <li>1, 2, 3, or 4 CHI bus requester ports; or</li> <li>1, 2, 3, or 4 AXI bus manager ports</li> </ul> For more information, see <a href="#">1.2 DynamIQ Shared Unit-120AE configuration options</a> on page 18
Accelerator Coherency Port (optional)	AMBA ACE5-Lite	Subordinate interface allowing an external manager to make coherent requests to cacheable memory. You can configure the DSU-120AE to have one or two ACP interfaces.
Peripheral port (optional)	AMBA AXI5 Issue H or CHI Issue E	The peripheral port provides the capability to separate the external system into a main memory subsystem accessed through the main requester interfaces and a subsystem for peripheral devices accessed through the peripheral port.
Utility bus	AMBA AXI5 Issue H	Memory-mapped port for accessing the following: <ul style="list-style-type: none"> <li>Power Policy Units (PPUs)</li> <li>Activity monitors</li> <li>Maximum Power Mitigation Mechanism (MPMM) registers</li> <li>RAS registers</li> <li>Memory System Resource Partitioning and Monitoring (MPAM) registers</li> </ul>
Cluster to DebugBlock	AMBA APB5 Issue D	APB interface from the cluster (requester) to the DebugBlock (completer)
DebugBlock to cluster	AMBA APB5 Issue D	APB interface from the DebugBlock (requester) to the cluster (completer)
Power state control	P-Channel	P-Channels for DSU-120AE and core power management
Clock state control	Q-Channel	Q-Channels for clock gating control
Wait For Event (WFE) event signaling	-	Signals for <i>Wait For Event</i> (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.
GIC interfaces	GIC Stream Protocol	Interrupts to individual cores. A single GIC Stream Protocol interface is shared by all cores.
Design for Test (DFT)	-	Interface to allow access for <i>Automatic Test Pattern Generation</i> (ATPG) scan-path testing.
Memory Built-In Self Test (MBIST)	Arm MBIST	Internal interface that supports the manufacturing test of the L3 cache and <i>Snoop Control Unit</i> (SCU) memories embedded in the DSU-120AE. Each core has its own internal MBIST interface.

## DebugBlock interfaces

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

**Figure 2-5: DebugBlock interfaces**



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

**Table 2-3: DebugBlock interfaces**

Purpose	Protocol	Notes
External debug	AMBA APB5 Issue D	Completer interface to external debug component, for example a <i>Debug Access Port</i> (DAP). It allows access to Debug registers and resources.
Cluster to DebugBlock	AMBA APB5 Issue D	APB interface from the cluster (requester) to the DebugBlock (completer).
DebugBlock to cluster	AMBA APB5 Issue D	APB interface from the DebugBlock (requester) to the cluster (completer).
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

- 3. [Clocks and resets](#) on page 64
- 7. [CHI requester interface](#) on page 146
- 9. [ACP subordinate interface](#) on page 173
- 8. [AXI manager interface](#) on page 162
- 10. [AXI or CHI requester peripheral port](#) on page 183
- 14. [Debug](#) on page 217

## 2.4.1 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-120AE* (DSU-120AE). PBHA is supported on all the manager ports, the AXI or CHI Peripheral port, and all the *Accelerator Coherency Ports* (ACPs).

The PBHA bits are provided externally as sideband signals on each of the supported PBHA busses, alongside manager 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 *Distributed Virtual Memory* (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.



For information on PBHA signals, see *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

---

## Related information

[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

## 2.4.2 Sequential hint

The bus manager ports provide speculative information about the probability of a sequential access to the other half of an aligned 128-byte range, close in time to this access. Your DRAM

controller could use this hint to optimize accesses. When this bit is high, it indicates that the core has identified that there is a high probability that a sequential access might be requested soon.

The sequential hint information is provided as a source sideband signal on either the CHI or AXI bus manager ports.

## 2.4.3 Interface protection

The DSU-120AE protects all the external interfaces to the cluster, except the non-safety related debug logic.

Interface protection is always present when *Dual-Core Lock-Step* (DCLS) is enabled, and is not present when DCLS is not configured.

Each signal that interface protection affects, has an associated check signal. This check signal has the identical signal name with "CHK" appended to the end of it. For input signals, the value of the signal (<signal>) is constantly compared to its corresponding CHK signal (<signal>CHK) to ensure that they agree. When these input signals differ, an interface protection fault is raised through the Fault Management Unit (FMU). For output signals, again a check signal is provided, but the comparison between the signals must be done at system-level. The protection mechanism that is specific to the underlying architecture of the relevant external interface determines the valid value of the associated check signal.

The following table describes the mechanisms, and their underlying architectures, that protect each of the supported external interfaces of the DSU-120AE cluster.

**Table 2-4: DSU-120AE interfaces and their protection mechanisms**

Interface	Protection mechanism	Description
CHIO, CHI1	Odd-parity	<a href="#">AMBA® AXI Protocol Specification</a>
ACP (ACE5-Lite)	Odd-parity	<a href="#">AMBA® AXI Protocol Specification</a>
Peripheral port (AXI4)	Odd-parity	<a href="#">AMBA® AXI Protocol Specification</a>
GIC (AXI4 Stream)	Odd-parity	<a href="#">AMBA® AXI Protocol Specification</a>
P/Q Channels	LPI redundancy (inverse polarity)	<a href="#">AMBA® Low Power Interface Specification</a>
Interrupts and events	Duplicated with inverse polarity	Duplicate signal that is provided as reference on inverse polarity.
Configuration	Duplicated with inverse polarity	Duplicate signal that is provided as reference on inverse polarity.
Clocks	Duplicated with in-phase clock-check	In-phase check clock that is provided on redundant clock tree. Primary clock is sourced to the primary logic; while the check clock is sourced to the corresponding redundant logic.
Resets	Duplicated with in-phase reset-check	In-phase check reset that is provided for all resets. Reset checker with redundant checking provides stable resolved resets.

Interface	Protection mechanism	Description
Distributed Time Interface	Odd-Parity on Status field	Status field of distributed scaled timer is protected by odd-parity. The DSU-120AE only protects the bits that are in-use in the DSU-120AE.
ATB, APB Requester (APBCD)	Not protected	Not protected

## 3. Clocks and resets

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE.

The DSU-120AE 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).

### 3.1 Clocks

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE.

**Table 3-1: DSU-120AE 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 <i>Snoop Control Unit</i> (SCU), L3 memory system, and all the external interfaces, including AXI, CHI, and <i>Accelerator Coherency Port</i> (ACP).  It is also used for cores and complexes that are configured to run synchronously with the DSU-120AE.
PCLK (DebugBlock)	The clock for the DebugBlock <b>Note:</b> The DebugBlock and the DSU-120AE 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-120AE DynamIQ™ cluster, ensure that you place an asynchronous bridge between the two clock domains.
PCLK (DSU-120AE cluster)	The clock for the debug interface in the DSU-120AE DynamIQ™ cluster <b>Note:</b> The DebugBlock and the DSU-120AE 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-120AE 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-120AE
GICCLK	The clock for the <i>Generic Interrupt Controller</i> (GIC) AXI-Stream interface between the DSU-120AE and an external GIC



Signal	Description
PERIPHCLK	The clock for the peripheral logic inside the DSU-120AE such as clock and power management logic and timers
PPUCLK	The clock for the <i>Power Policy Units</i> (PPUs). The PPUs reside in their own clock domain, see <a href="#">3.2 Clock domains</a> on page 66.

For more information on core and complex clock signaling, see *Functional integration* in the Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual.

All clocks can be driven fully asynchronously to each other. The DSU-120AE 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.



- 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 COREyCLK 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 bus manager port, 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-120AE is designed with the following expectations to achieve an acceptable performance:

- The COREyCLK or COMPLEXxCLK can be dynamically scaled to match the performance requirements of that core.
- The two cores or complexes in a pair are driven either by the COREyCLK and COREyCLKCHK clocks, or they are driven by the COMPLEXxCLK and COMPLEXxCLKCHK clocks. The primary clocks and their CHK clocks must be the same frequency and in phase at all times. This means that any dynamic frequency scaling must be done to both cores or complexes in the pair.
- SCLK is recommended to run between the maximum COREyCLK or COMPLEXxCLK frequency and approximately half of the maximum COREyCLK 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.
- The frequency of ATCLK must be determined based on the trace bandwidth of the system.
- 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 COREyCLK 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 COREyCLK 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 COREyCLK or COMPLEXxCLK frequency.

- Arm® recommends that the PPUCLK clock frequency is at least 25% of the maximum COREyCLK 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

[3.2 Clock domains](#) on page 66

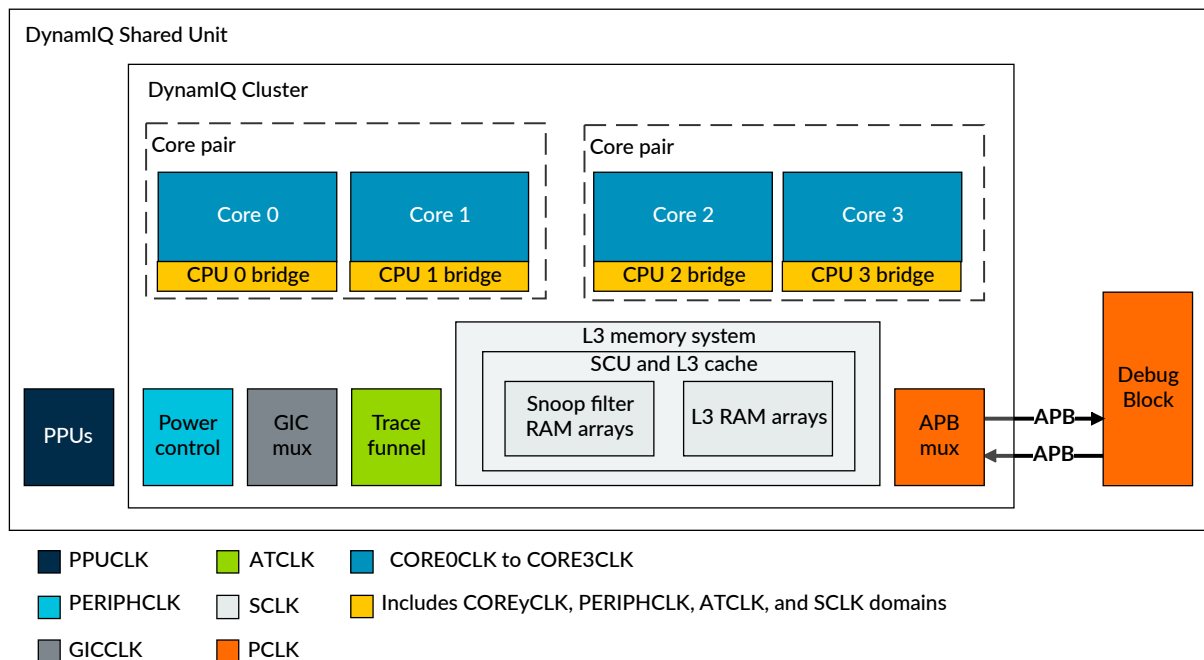
[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

## 3.2 Clock domains

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) has multiple clock domains. Each core-pair or complex-pair can be implemented in a separate clock domain.

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

**Figure 3-1: DSU-120AE 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 cluster shared logic 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

[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

[3.1 Clocks](#) on page 64

## 3.3 Resets

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) has five external reset signals, a cluster-wide Cold reset, a cluster-wide MBIST reset, and a reset for the DebugBlock. Other resets, such as Warm resets to the cores are controlled, inside the cluster, by the *Power Policy Units* (PPUs).

The following table describes the DSU-120AE reset signals.

**Table 3-2: DSU reset signals**

Signal	Description
nRESET	A DSU-120AE 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.
nRESETCHK	A redundant copy of the nRESET signal.
nMBISTRESET	A DSU-120AE DynamIQ™ cluster reset. nMBISTRESET is a single cluster-wide reset signal that resets all necessary logic in the cores and cluster for <i>Memory Built-In Self Test</i> (MBIST) testing.
nMBISTRESETCHK	A redundant copy of the nMBISTRESET signal.
nPRESET	The DebugBlock reset. A reset for all resettable registers in the DebugBlock.

The nMBISTRESET signal is active low and must be driven high for functional mode. A single top level nMBISTRESET is utilized to reset the IP in preparation for MBIST operations.

All reset inputs can be asserted (HIGH to LOW) and deasserted (LOW to HIGH) asynchronously. Reset synchronization logic inside the DSU-120AE 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.

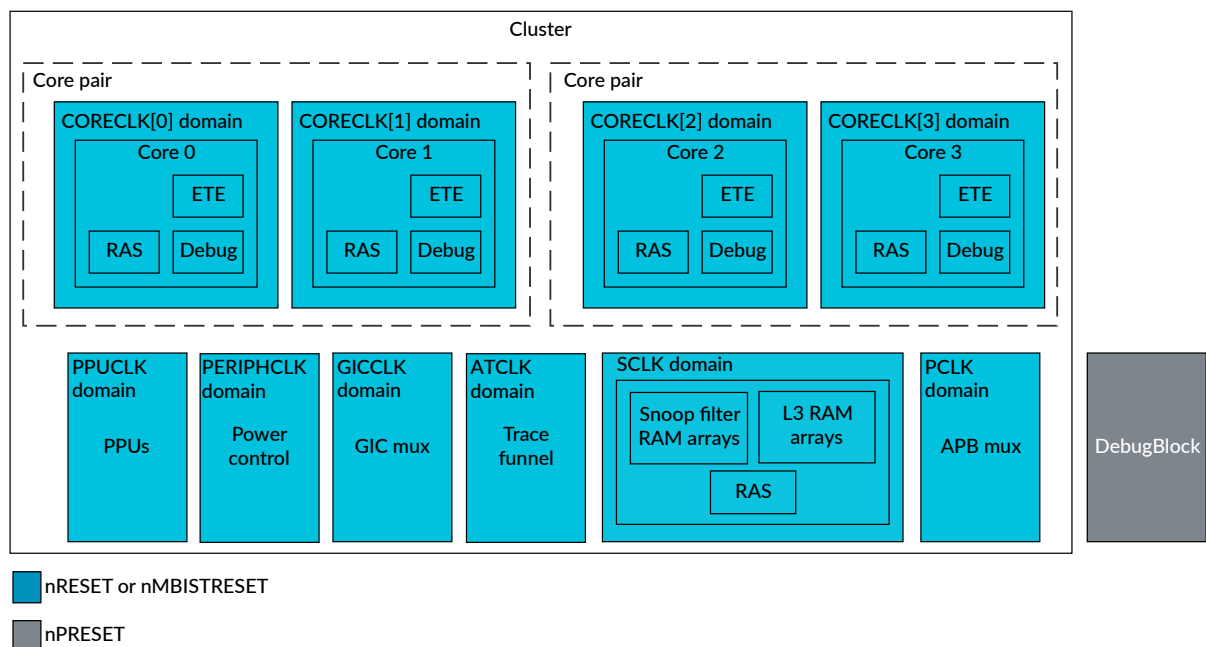
Resetting individual cores and other cluster logic can be performed by programming the appropriate integrated PPU, see [5. Power and reset control with Power Policy Units](#) on page 103. When the cluster internal resets are asserted, the PPUs drive the reset output signals that correspond to the power domain logic that is being reset.



You can use the DSU-120AE 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 clock domain and therefore must be synchronized before use in the destination component.

The following figure shows the pin-controlled reset domains.

**Figure 3-2: DSU-120AE pin-controlled reset domains**



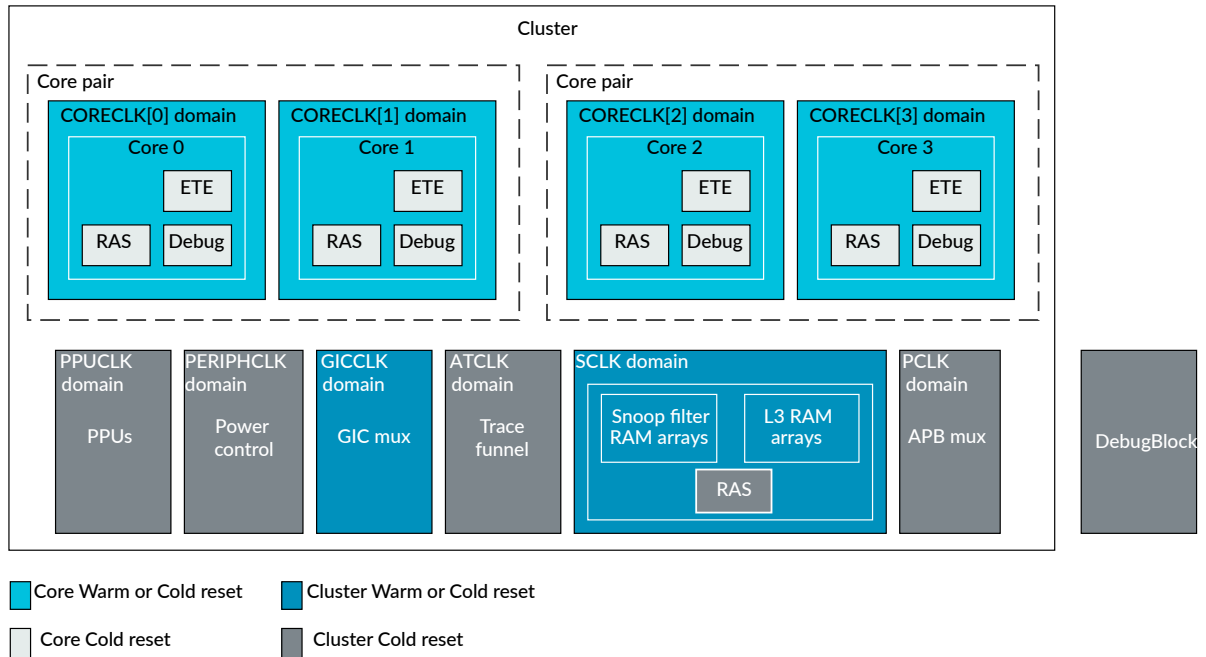
## 3.4 Resetting with Power Policy Units

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 [5. Power and reset control with Power Policy Units](#) on page 103.

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

**Figure 3-3: DSU-120AE 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 [5.3.2 nRESET sequence](#) on page 109.

## 4. Power management

This chapter describes the power domains, power modes, and operating modes for the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE.



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 [5. Power and reset control with Power Policy Units](#) on page 103.

### 4.1 Power management in the DSU-120AE

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE and cores in the cluster include:

- Internal core clock gating where different internal parts of the core are clock idle
- Per-core-pair *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 [4.4.1 L3 cache RAM powerdown](#) on page 78 and [4.4.2 L3 cache slice powerdown](#) on page 83.
- 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



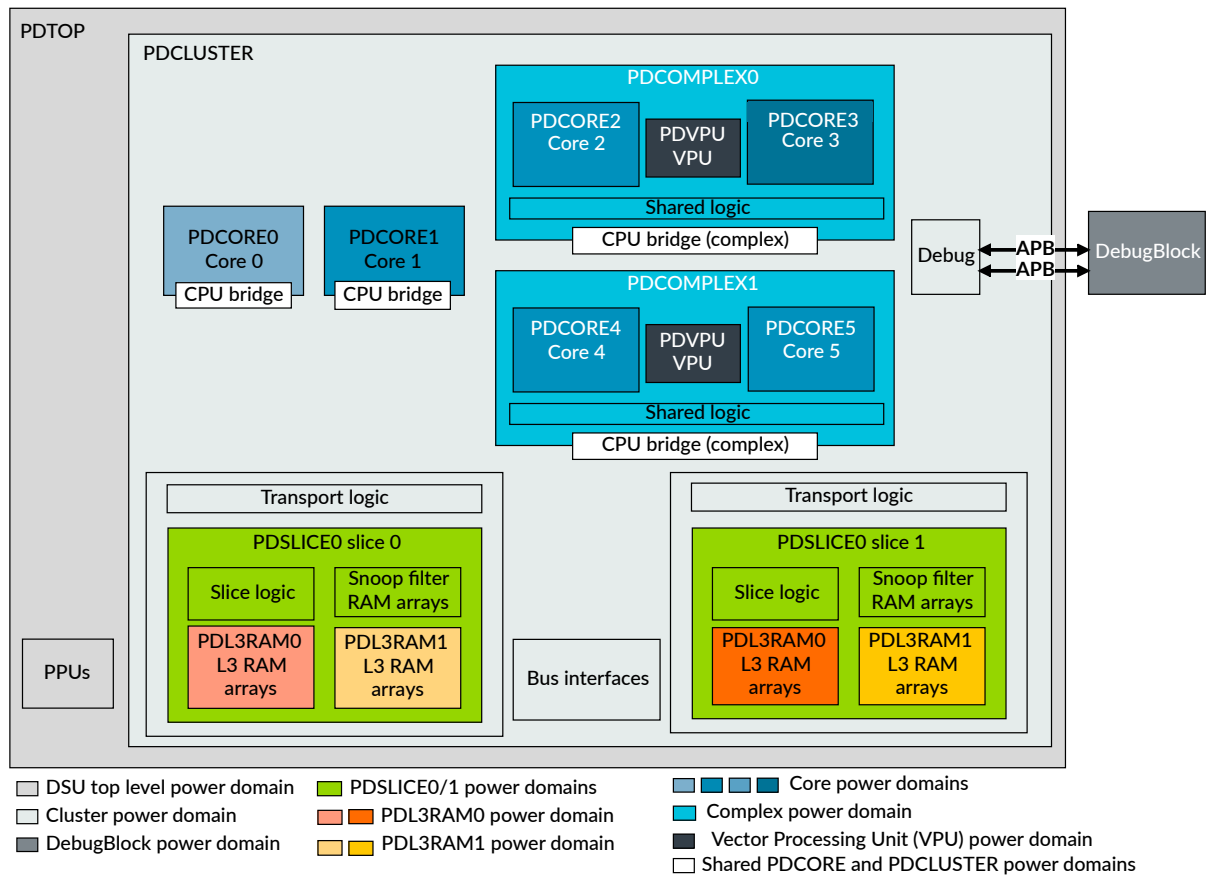
- The DSU-120AE power domain architecture, power modes, and operational modes, are based on the Arm Power Control System Architecture, see [Arm® Power Control System Architecture](#).
- This chapter does not describe how to use the PPUs for the cluster or the cores, see instead [5. Power and reset control with Power Policy Units](#) on page 103.

## 4.2 DSU-120AE supported power domains

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE-based cluster.

**Figure 4-1: DSU-120AE power domains**



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 PPU's 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 PPU's reside allows the PPU's 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 4-1: DSU-120AE power domains](#) on page 71. 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 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 or to be placed in retention 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.





Note

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, the only configuration that is powered up is either, a single L3 cache slice, half the cache slices, or all L3 cache slices.

For example, a DSU-120AE configured with four L3 cache slices, cache slices 1-3 are powered down together while cache slice 0 remains powered up.

## 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.

## 4.3 Cluster power modes

The DSU-120AE 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-120AE DynamIQ™ cluster.

**Table 4-1: DSU-120AE 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.
Full retention mode	FULL_RET	In Full retention mode, the L3 cache RAMs and snoop filter RAMs are placed in a retention state, while the cache slice logic is powered down. The rest of the cluster logic such as the bus requesters, and transport, remains powered.
Memory retention mode	MEM_RET	In Memory retention mode, only the L3 cache RAMs are placed in retention. The rest of the DSU-120AE 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.

Power mode	Short name	Description
Debug recovery mode	DBG_RECOV	Debug recovery mode is used for applying a Warm reset to the cluster, while preserving memory and <i>Reliability, Availability, and Serviceability</i> (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.

### 4.3.1 On mode (ON)

In the On mode, the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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.

### 4.3.2 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-120AE DynamIQ™ cluster), but the PDTOP domain is still powered up including all the *Power Policy Units* (PPUs).

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) can be initialized into this mode on a Cold reset.

### 4.3.3 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 FUNC\_RET field of the IMP\_CLUSTERPWRCTLR\_EL1 register, see [A.1.6 IMP\\_CLUSTERPWRCTLR\\_EL1, Cluster Power Control Register](#) on page 279.

The *Power Policy Units* (PPUs) can be programmed to automatically control entry and exit from this mode without software intervention, see [5. Power and reset control with Power Policy Units](#) on page 103.

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.

### 4.3.4 Cluster full retention mode (FULL\_RET)

Full retention mode (FULL\_RET) allows the L3 cache and snoop filter RAMs to be placed in retention state and the cache slice logic is powered down, 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 and the slice logic is powered down, 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 FULL\_RET field of the CLUSTERPWRCTLR register.

The *Power Policy Units* (PPUs) can be programmed to automatically control entry and exit from this mode without software intervention, see [5. Power and reset control with Power Policy Units](#) on page 103.

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 FULL\_RET mode to an ON mode to service the access.

Similar to functional retention mode (FUNC\_RET), in this mode the contents of the L3 cache RAMs are retained, whilst much of the cluster logic is powered up. However, in FULL\_RET mode additional power is saved because the cache slice logic is powered down.

### 4.3.5 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-120AE* (DSU-120AE) remains coherent when in Memory retention mode. Any snoop arriving is stalled while the DSU-120AE 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-120AE 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.

### 4.3.6 Emulated off mode (OFF\_EMU)

In the Emulated off mode, the cluster behaves logically as 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.

### 4.3.7 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-120AE* (DSU-120AE) logic remains powered. Therefore, debug accesses to the cluster can be made.

### 4.3.8 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 be 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.
- Warm reset mode performs a warm reset on both the primary and redundant logic. Some of the *Dual-Core Lock-Step* (DCLS) comparators are on cold reset, so they do not perform warm reset. Because the cluster state can change during

the reset, it can rarely lead to a case, when the comparators report a fake fault during Warm reset.

### 4.3.9 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-120AE 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.



Note

- 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:
  - 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 or trying to debug.
- The cluster is in Warm reset.
- 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.
- When the PPU\_PTCR.DBG\_RECOV\_PORST\_EN register bit is 0, the Debug recovery mode performs a warm reset on both the primary and redundant logic. Some of the *Dual-Core Lock-Step* (DCLS) comparators are on cold reset, so they do not perform warm reset. Because the cluster state can change during the reset, it can rarely lead to a case, when the comparators report a fake fault during Debug recovery.

---

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.

## 4.4 L3 RAM power control

In addition to retention features, the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) can further reduce static leakage power, using three powerdown features.

- Optionally power down half, or all except one, of the L3 cache slices.
- Within each L3 cache slice, power down a portion of the L3 cache RAM that the cache slice contains.
- Use Quick Nap with L3 data RAMs, for fine-grained automatic transitions to a low-leakage power mode.

## 4.4.1 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.



Note

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:

- [4.4.1.1 Setting automatic L3 cache power portion control](#) on page 79
- [4.4.1.2 Setting CLUSTERPWRCTLR\\_EL1.PRTNRQ power portion control](#) on page 80
- [4.4.1.3 L3 RAM power control using PPU static transactions](#) on page 81



Note

For information on operating modes, see [4.5 Cluster operating modes](#) on page 87.

---

#### 4.4.1.1 Setting automatic L3 cache power portion control

The DSU-120AE 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
- IMP\_CLUSTERL3UPTH2\_EL1, CLUSTERL3UPTH2



This register is only for use in conjunction with automated slice powerdown. See [4.4.2.2 Automated slice powerdown](#) on page 84.

---

See [4.4.1.4 Calculating values for threshold registers](#) on page 81 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.

#### 4.4.1.2 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.

### 4.4.1.3 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.

### 4.4.1.4 Calculating values for threshold registers

The DSU-120AE 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, and 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 hardware periodically calculates the hit and miss rates, based on the setting in the `IMP_CLUSTERPWRCTLR_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.

The hardware contains internal copies of the IMP\_CLUSTERL3HIT\_EL1 and IMP\_CLUSTERL3MISS\_EL1 registers that count the same events. At the end of each time period, the value in these internal counter 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
- IMP\_CLUSTERL3UPTH2\_EL1, CLUSTERL3UPTH2

After the calculations are complete, the internal version of the IMP\_CLUSTERL3HIT\_EL1 and IMP\_CLUSTERL3MISS\_EL1 registers are reset to zero. Depending on the number of L3 cache ways powered up, and the values in the hit and miss registers, and threshold registers, the following happens:

- If all L3 cache ways are powered up, then when IMP\_CLUSTERL3HIT\_EL1 is less than IMP\_CLUSTERL3DNTH0\_EL1, the cluster signals to the cluster *Power Policy Unit* (PPU) that it should request a power down of half of the ways of the L3 cache.
- If half of the L3 cache ways are powered, then:
  - When IMP\_CLUSTERL3HIT\_EL1 is less than IMP\_CLUSTERL3DNTH1\_EL1 and IMP\_CLUSTERL3MISS\_EL1 is less than IMP\_CLUSTERL3UPTH1\_EL1, the cluster signals to the cluster PPU that it should request a power down of all the L3 cache ways.
  - When IMP\_CLUSTERL3MISS\_EL1 is greater than IMP\_CLUSTERL3UPTH1\_EL1 and IMP\_CLUSTERL3HIT\_EL1 is greater than IMP\_CLUSTERL3DNTH1\_EL1, the cluster signals to the cluster PPU that it should request a power up of all the L3 cache ways.
- If no L3 cache ways are powered, then when IMP\_CLUSTERL3MISS\_EL1 is greater than IMP\_CLUSTERL3UPTH0\_EL1, the cluster requests to the cluster PPU to power up half the ways of the L3 cache.

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. The recommended set of values shown below and are based on having 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\_CLUSTERPWRCTLR\_EL1.AUTOPRTN register**

```
IMP_CLUSTERL3DNTH0_EL1 = 12288 * T * L / D
IMP_CLUSTERL3DNTH1_EL1 = 4096 * T * L / D
IMP_CLUSTERL3UPTH0_EL1 = 4096 * T * L / D
IMP_CLUSTERL3UPTH1_EL1 = 4096 * T * L / D
IMP_CLUSTERL3UPTH2_EL1 = 24576 * T * L / D
```

## 4.4.2 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. When four or eight slices are configured, then half of the slices can be powered off, leaving the remaining half of the slices handling all of the addresses.

If N cache slices are implemented, then in ONE SLICE operating 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 ONE 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 ONE 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.

### 4.4.2.1 L3 cache slice power control

The DSU-120AE contains hardware to automatically schedule L3 cache slice powerdown and powerup requests. The cluster provides a number of configurable mechanisms to determine when the L3 cache slices need to be powered. If any enabled mechanism determines that more slices need to be powered up, the cluster automatically requests this power transition. If all enabled

mechanisms determine that fewer slices are required, and this remains the case for a configurable period of time, the cluster automatically requests this power transition.

### Enable automatic L3 cache slice power control

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.

- The System Control Processor (SCP) programs the cluster Power Policy Unit (PPU) for dynamic transitions.
- Software running on the core programs the IMP\_CLUSTERPWRCTLR\_EL1 to configure the automatic powerdown mechanisms. The controls for the different mechanisms are in the following register fields. See [A.1.6 IMP\\_CLUSTERPWRCTLR\\_EL1, Cluster Power Control Register](#) on page 279 for more information.
  - SLCRQ. You can use this field to program a minimum number of L3 cache slices that should be kept on.
  - SLCPRTN. This field enables a mechanism that keeps L3 cache slices powered on if the "automatic L3 cache power portion control" mechanism determines that all their L3 cache is useful. The mechanism can also power up more L3 cache slices based on the value of IMP\_CLUSTERL3UPTH2\_EL1 / CLUSTERL3UPTH2. See [4.4.1.1 Setting automatic L3 cache power portion control](#) on page 79 for how to program this register.
  - SLCBW. This mechanism ensures that more slices are powered up if the current number of slices cannot provide the required bandwidth. You can use this field to control the sensitivity of this mechanism or disable it.
  - SLCSF. This mechanism ensures that the number of slices powered up is sufficient to provide a large enough snoop filter for the powered on cores' caches. You can use this field to disable the mechanism, however having enough snoop filter capacity is important for performance.
  - HSLCMASK, OSLCMASK, HSLCCNT, OSLCCNT. You can use these fields to guarantee a minimum slice operating mode when a certain number of specific cores is on.
- Software running on the core sets IMP\_CLUSTERPWRCTLR\_EL1.AUTOSLC to a non-zero value. Alternatively the SCP can program the CLUSTERPWRCTLR register through the utility bus.

#### 4.4.2.2 Automated slice powerdown

This feature provides more hardware automation of the slice powerdown decision.

When the CLUSTERPWRCTLR.AUTOSLC field is zero, only the SLCRQ field controls the slice powerdown. When the AUTOSLC field is nonzero, it enables the automation and indicates the time period with which the slice powerdown decisions are made.

The decision to change the slice power mode depends on these conditions:

- SLCRQ bits. These indicate a minimum mode, below which the slices are not powered off. For example, this allows the use of HALF SLICES mode while preventing the L3 cache from entering ONE SLICE mode.

- If the SLCSF bit is set and the number of cores powered on requires a snoop filter size greater than the current slice power mode, the slice power mode is increased.
- If the CLUSTERPWRDN.SHORTSLP bit for a core is set high, this indicates that the core must be treated as if it were ON for the automated slice power down calculation, even if the core is OFF.
- If the SLCBW bit is set and the slice bandwidth is higher than the threshold for this time period, the slice power mode is increased.
- If the number of cores powered on is greater than the HSLCCNT or OSLCCNT fields, after masking with the HSLCMASK or OSLCMASK fields, the slice power mode is increased. This allows configuring behaviors based on the number and types of core. For example, if you have a configuration with 4 high-performance cores and 4 high-efficiency cores, you can ensure that:
  - If any high-performance core is powered on, it uses ALL SLICE mode.
  - If no high-performance cores are powered on, they use HALF SLICE mode.
  - If only two or less high-efficiency cores are powered on, ONE SLICE mode can be entered.

This example can be configured with these settings:

- Set HSLCMASK to include all the high-performance cores
- Set HSLCCNT to 0.
- Set OSLCMASK to include all the high-efficiency cores.
- Set OSLCCNT to 2.



If the SLCPRTN bit is set, then the AUTOPRTN mechanism can indicate if more cache capacity is required.

---

These conditions are checked continuously, and if any become true indicating that more slices are needed, then the SLCPRTN bit operating mode immediately transitions to a higher number of slices. Depending on the current conditions, in ONE SLICE mode, the transition can be to HALF SLICES, or directly to ALL SLICES. This occurs because a single direct transition is more efficient than going through HALF SLICES mode with two separate transitions.

For transitions to a smaller number of slices, the decision is more conservative. This decision is made only at the end of each configured time period, and all of the conditions must indicate that a lower number of slices is suitable. This must also be true for the whole of the time period. The transitions to a smaller number of slices is always done in steps, from ALL to HALF, and then only HALF to ONE, after at least one more time period has passed.

Conditions that include the core power status by default treat the core as ON if it is in any power mode other than OFF or OFF\_EMU.

However, in some cases the operating system might have more information about how long the core is likely to remain off. If the OS has information that the core is only likely to remain off for a short time, it can set the CLUSTERPWRDN.SHORTSLP bit high before powering off the

core. If the operating system anticipates that the core will be powered off for longer, it sets the CLUSTERPWRDN.SHORTSLP bit low before powering off the core.

The AUTOSLC logic takes this into account, and treats any core with the SHORTSLP bit set to the same value as if it was still on. This means that the slice power off decisions only take place if the core is likely to remain in that state for a long period.

When the SLCPRTN bit is clear, the slice powerdown decisions are made independently of the RAM powerdown decisions. However, when the SLCPRTN bit is set, the two mechanisms interact with each other to give a complementary sequence of steps. Starting from ALL SLICE FULL RAM, the AUTOPRTN mechanism decides in the usual way whether to power down half the RAMs, giving half the cache capacity.

When the DSU has been in ALL SLICE HALF RAM mode for at least the AUTOSLC timeout period, and all the other conditions are met, then a transition starts to HALF SLICES mode, followed immediately by a transition to FULL RAM. This means that the bandwidth and power are reduced in HALF SLICE mode, but the cache capacity remains unchanged at half the overall capacity. The AUTOPRTN can then continue and choose to go back to HALF RAM, giving a quarter of the overall cache capacity.

When the DSU has been in HALF SLICE HALF RAM mode for at least the AUTOSLC timeout period, and all the other conditions are met, then a transition starts to ONE SLICE, followed immediately by a transition to FULL RAM. This keeps the capacity at a quarter if four slices are configured, or reduces to one eighth if 8 slices are configured. The AUTOPRTN mechanism can then continue in the usual manner and enter HALF RAM, or eventually SFONLY mode, while remaining in ONE SLICE mode.

If at any time in a FULL RAM mode, the AUTOPRTN mechanism indicates that more cache capacity is required, the slice power state changes to enable more capacity, if it is not already in ALL SLICE mode. The IMP\_CLUSTERL3UPTH2\_EL1 / CLUSTERL3UPTH2 upsize threshold 2 register configures this transition threshold.

### 4.4.3 L3 cache RAM Quick Nap

Some RAMs, such as the Arm POP RAMs, provide a quick nap (or light sleep) mode. This allows powering down some of the RAM periphery logic to reduce leakage, and also being able to power up again within a small number of cycles so that normal operation can be resumed without impacting performance.

Quick nap is enabled by default, if the RAMs that are implemented support it. There is no need for additional software control.

The L3 cache enters Quick Nap mode automatically after a short period of no activity. Quick Nap is controlled at a granular level for each slice. Therefore, access to one slice does not need to wake the RAMs in another slice. The wakeup is requested when a new access enters the tag pipeline, and so the wakeup can happen in parallel with the tag access. There is no mechanism for stalling the access, so the RAM must be awake by the time the data RAM access occurs. There is no performance impact from this behavior.

## 4.5 Cluster operating modes

An operating mode is a component-specific configuration of the power modes. For the *DynamIQ™ Shared Unit-120AE* (DSU-120AE), 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-120AE supports several operating modes to control two groups of modes. One mode from each group can be combined together in any combination.

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 4-2: Operating modes for L3 cache slices**

Operating mode name	Description
ONE SLICE	One slice is active (powered up). This slice resides in its own power domain.
HALF SLICES	Half the total number of slices are powered up.
ALL SLICES	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 4-3: Operating modes for L3 cache RAMs**

Operating mode name	Description
SFONLY	The L3 cache data and tag RAMs in each cache slice are powered down.
HALF RAM	One half of the L3 cache data and tag RAMs in each active slice are powered up.
FULL RAM	All of the L3 cache data and tag RAMs in each active slice are powered up.



**Note**

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

## 4.6 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.

This table shows which combinations of slices are powered on and active in four different slice configurations.

**Table 4-4: DSU-120AE Slice activity for 1, 2, 4, and 8 slice configurations**

	1 slice configuration		2 slice configuration		4 slice configuration		8 slice configuration	
	Active	Inactive	Active	Inactive	Active	Inactive	Active	Inactive
One Slice	0	None	0	1	0	1, 2, 3	0	1, 2, 3, 4, 5, 6, 7
Half Slices	0	None	0	1	0, 1	2, 3	0, 1, 2, 3	4, 5, 6, 7
All Slices	0	None	0, 1	None	0, 1, 2, 3	None	0, 1, 2, 3, 4, 5, 6, 7	None

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.

In the following two tables:

- N is the number of L3 cache slices.
- The internal power modes, for example, off and retention, are written in lowercase lettering as compared to cluster power modes which are written in uppercase lettering.
- The cluster power mode ON state includes the WARM\_RST, DBG\_RECOV, and MEM\_RET\_EMU states.
- The term retention is abbreviated ( ret.).



Note

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 4-5: DSU-120AE RAM power states for an active cache slice**

Cluster power mode	ON, OFF_EMU			FUNC_RET			FULL_RET			MEM_RET			OFF
CLUSTER PCSM PSTATE power mode	ON			FUNC_RET			FULL_RET			MEM_RET			OFF
Operating mode	FULL RAM	HALF RAM	SF ONLY	FULL RAM	HALF RAM	SF ONLY	FULL RAM	HALF RAM	SF ONLY	FULL RAM	HALF RAM	SF ONLY	N/A
Power portion 1 L3 data and tag RAMs	on	off	off	ret.	off	off	ret.	off	off	ret.	off	off	off
Power portion 0 L3 data and tag RAMs. Also victim RAMs	on	on	off	ret.	ret.	off	ret.	ret.	off	ret.	ret.	off	off
Snoop Filter SF RAM	on	on	on	ret.	ret.	ret.	ret.	ret.	ret.	off	off	off	off
LTDB RAM	on	on	on	off	off	off	off	off	off	off	off	off	off
Associated PDSLICE power domain state	on	on	on	on	on	on	off	off	off	off	off	off	off
PDCLUSTER power domain logic	on	on	on	on	on	on	on	on	on	off	off	off	off





The power portions 0 and 1 each consist of eight cache ways with one cache way for each of the eight *Memory system resource Partitioning And Monitoring* (MPAM) cache partitions. Therefore, powering down power-portion 1 powers down one cache way in each MPAM partition. For more information, see [6.4 L3 cache partitioning](#) on page 138.

The following table shows the power state dependencies for an inactive cache slice:

**Table 4-6: DSU-120AE RAM power states for an inactive cache slice**

Cluster power mode	ON, OFF_EMU			FUNC_RET			FULL_RET			MEM_RET			OFF
CLUSTER PCSM PSTATE power mode	ON			FUNC_RET			FULL_RET			MEM_RET			OFF
Operating mode	FULL RAM	HALF RAM	SF ONLY	FULL RAM	HALF RAM	SF ONLY	FULL RAM	HALF RAM	SF ONLY	FULL RAM	HALF RAM	SF ONLY	N/A
Power portion 1 L3 data and tag RAMs	off	off	off	off	off	off	off	off	off	off	off	off	off
Power portion 0 L3 data and tag RAMs. Also victim RAMs	off	off	off	off	off	off	off	off	off	off	off	off	off
Snoop Filter SF RAM	off	off	off	off	off	off	off	off	off	off	off	off	off
LTDB RAM	off	off	off	off	off	off	off	off	off	off	off	off	off
Associated PDSLICE power domain state	off	off	off	off	off	off	off	off	off	off	off	off	off
PDCLUSTER power domain logic	off	off	off	off	off	off	off	off	off	off	off	off	off

## 4.7 Cluster PPU mode transitions

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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.

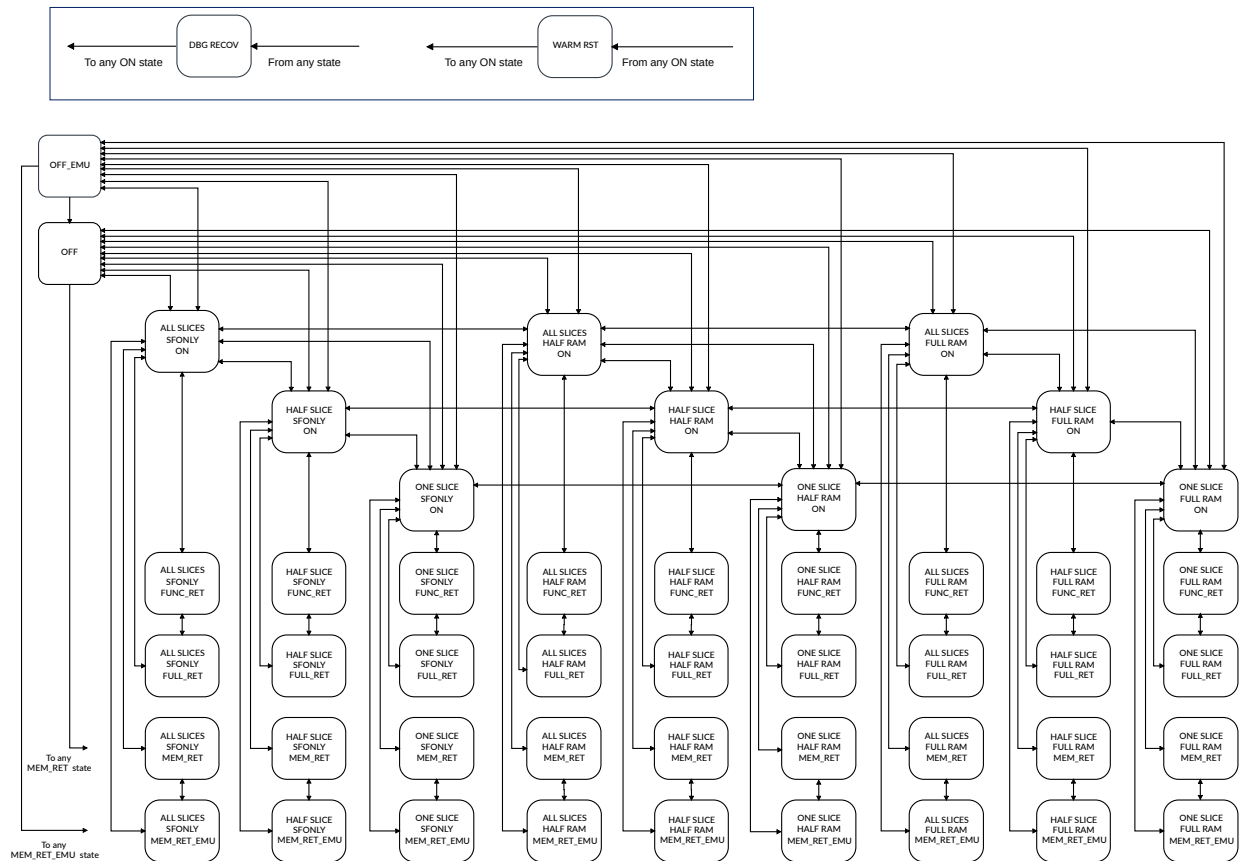
The cluster PPU controls transitions between the cluster PPU modes. Therefore, a *System Control Processor* (SCP) can program the PPU to go to any allowed PPU mode, and the PPU automatically makes the necessary transitions to reach the requested PPU mode.

The following figure shows the supported PPU mode transitions for the DSU-120AE DynamIQ™ cluster.



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

**Figure 4-2: DSU-120AE DynamIQ™ cluster PPU mode transitions**

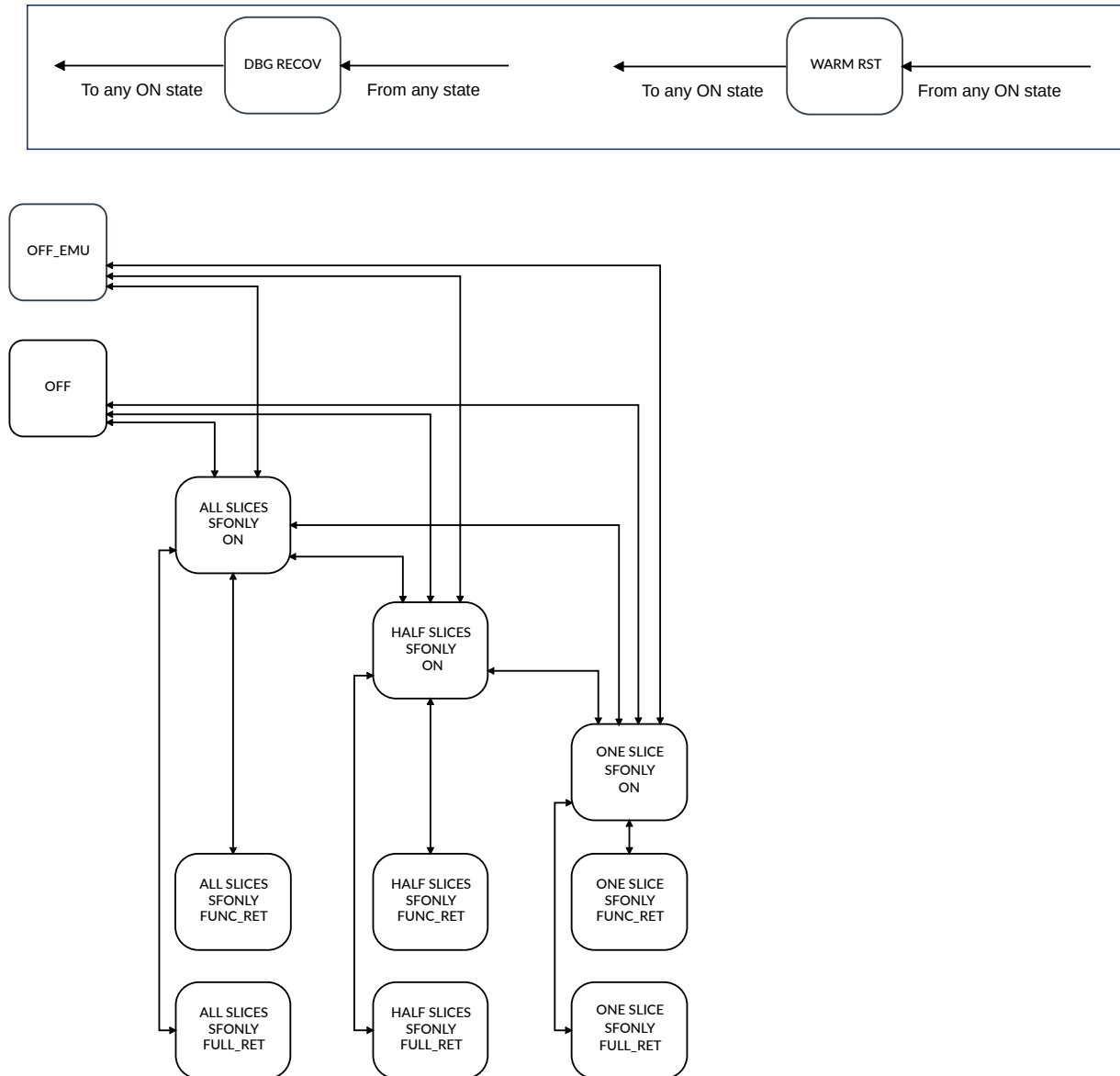


The following figure shows the supported PPU mode transitions for the DSU-120AE DynamIQ™ cluster where L3 cache is not implemented.



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

**Figure 4-3: DSU-120AE DynamIQ™ cluster PPU mode transitions, no L3 cache**



## 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 SLICES SFONLY ON and ALL SLICES HALF RAM ON

In these PPU modes, the DynamIQ™ cluster shared logic, including snoop filter RAMs, is powered up but half or all the L3 cache RAMs remain powered down. If the DSU-120AE is implemented with no L3 cache, then only the ALL SLICES SFONLY ON mode is supported.

## HALF SLICES SFONLY ON, HALF SLICES HALF RAM ON, and HALF SLICES FULL RAM ON

In these PPU modes, the DSU-120AE DynamIQ™ cluster is on and operational in the same way as the equivalent ALL SLICES operational mode, except that only half of the total number of cache slices configured are powered up and are active. The other slices are inactive and can be powered down.



Note

If the cluster is configured with two slices, then these PPU modes are identical to the ONE SLICE related PPU modes.

---

## ONE SLICE SFONLY ON, ONE SLICE HALF 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\_SLICES 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, HALF 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 FULL\_RET, HALF RAM FULL\_RET, and FULL RAM FULL\_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 L3 cache slice logic is powered down. 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, HALF 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.

### 4.7.1 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 HALF ON.
- The operating mode is maintained when moving from ON to FUNC\_RET, FULL\_RET, or MEM\_RET power modes.

**Note**

For more information, see [Arm® Power Policy Unit Architecture Specification](#).

### 4.7.2 PPU mode transition behavior

Where there is a transition between PPU modes, the DSU-120AE 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 [5.3 Power policy unit operation](#) on page 107.

**Table 4-7: Cluster domain PPU mode transition behavior**

Start PPU mode	End PPU mode	DSU-120AE behavior
OFF	ON	The L3 cache and snoop filter are initialized, and the cluster is brought into coherency with the rest of the system.
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	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.
ON	FULL_RET	If there is any ongoing memory access, the request is denied. Access to the L3 cache RAMs and slice logic is blocked. Once in FULL_RET power mode, any new transaction to the L3 cache is stalled until there is a return to ON mode.
FULL_RET	ON	Access to the L3 cache is allowed.
FUNC_RET	FULL_RET	-
FULL_RET	FUNC_RET	-
ON	MEM_RET	If there is any ongoing core or cluster activity, the request is denied.
MEM_RET	ON	The snoop filter is initialized.
FULL RAM ON	HALF RAM ON	Relevant ways in L3 cache are cleaned and invalidated.
HALF RAM ON	SFONLY ON	Relevant ways in L3 cache are cleaned and invalidated.
SFONLY ON	HALF RAM ON	Relevant ways in L3 cache are initialized.
HALF RAM ON	FULL RAM ON	Relevant ways in L3 cache are initialized.
HALF SLICES 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.
HALF SLICES ON	ALL SLICES ON	Relevant lines in L3 cache are cleaned and invalidated. Relevant snoop filter entries are emptied, causing back-invalidations to the cores if necessary.
ALL SLICES ON	HALF SLICES ON	Relevant lines in L3 cache are cleaned and invalidated. Relevant snoop filter entries are emptied, causing back-invalidations to the cores if necessary.
ALL SLICES 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	HALF SLICES 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 SLICES ON	Relevant lines in L3 cache are cleaned and invalidated. Relevant snoop filter entries are emptied, causing back-invalidations to the cores if necessary.

For information and guidelines on implementing your PCSM, see *System Design* in Arm® DynamIQ™ *Shared Unit-120AE Configuration and Integration Manual*.

### 4.7.3 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-120AE 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.

## 4.8 Core PPU modes

Each core or complex in the DSU-120AE 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.



Note

- As there are no operating modes for the cores in the DSU-120AE 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 4-8: 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 <i>Vector Processing Unit</i> (VPU), if present, is OFF.
Full retention	FULL_RET	<p>The core is in retention state.</p> <p>In this mode, only power that is required to retain register and RAM state is available. The core is non-operational.</p> <p>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.</p>
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	<p>The RAM and logic are powered up.</p> <p>This mode is for applying a Warm reset to the cluster, while preserving memory and RAS registers for debug purposes. Both cache and <i>Reliability, Availability, and Serviceability</i> (RAS) state are preserved when transitioning from DBG_RECOV to ON.</p> <p><b>Caution:</b> This mode must not be used during normal system operation.</p>
Warm reset	WARM_RST	<p>This Warm reset mode is used to reset the core. For more information about what is reset, see your core <i>Technical Reference Manual</i> (TRM).</p>

### 4.8.1 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.



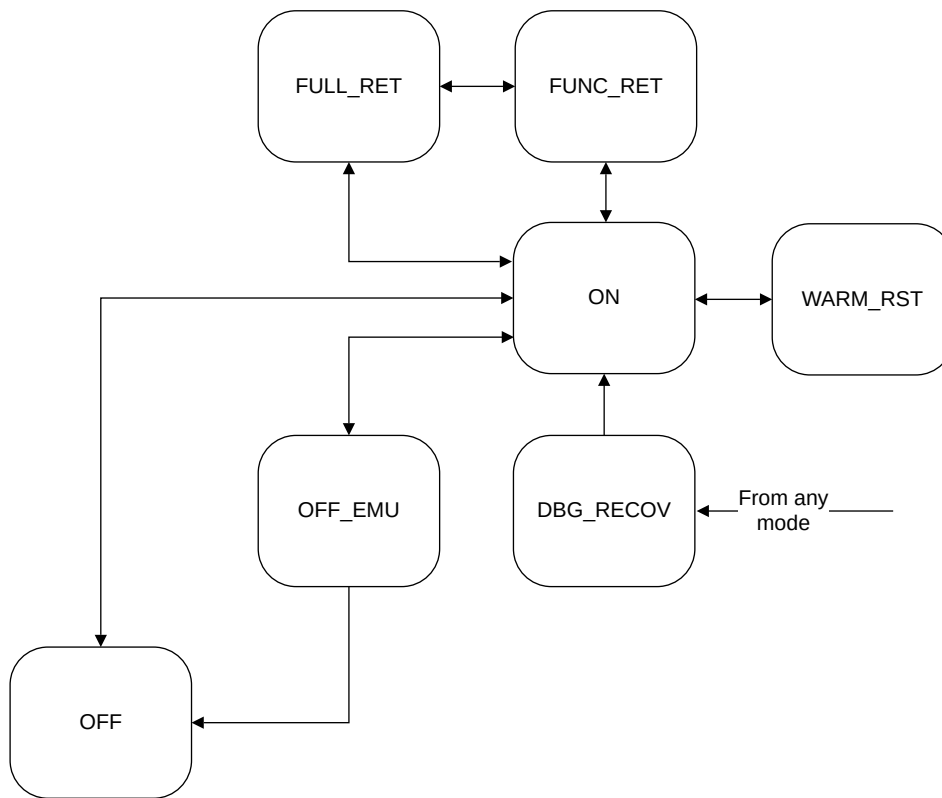
Note

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

The following figure shows the permitted core PPU mode transitions.



**Figure 4-4: 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.

### Warm reset mode (WARM\_RST)

Warm reset mode applies a Warm reset to the core logic. When in Warm reset, the following applies:

- If any core is put into Warm reset mode, then the cluster must also be 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.

## 4.9 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).

### 4.9.1 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 4-9: 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

Requested PPU mode		PCSM channel		
Core0	Core1	Core0	Core1	Shared logic
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
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



Caution

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.

## 4.9.2 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 4-10: 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: <ul style="list-style-type: none"> <li>Makes a request from ON mode to OFF mode.</li> <li>Makes a request from ON mode to OFF_EMU mode.</li> <li>Requests a reset using the RMR.RR register bit field.</li> </ul>	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 DEVACTIVE\* 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 [5.13 Core Full retention mode and static mode restrictions](#) on page 135.

## 4.10 Maximum Power Mitigation Mechanism

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) implements a *Maximum Power Mitigation Mechanism* (MPMM) feature that can be used to limit high activity events within the cluster, or trade off bandwidth versus power.

Larger configurations of the DSU-120AE support a very large bandwidth, and this can cause a lot of dynamic power to be consumed. It might be impractical or too expensive for a system implementer to build the *System On Chip* (SoC) power supply to support the maximum current draw from the DSU-120AE at the same time as the cores, *Graphics Processing Unit* (GPU), and any other components are also consuming their maximum current. To assist with overall power mitigation, the DSU-120AE implements a cluster MPMM.

The MPMM mechanism provides a number of gears. Each gear restricts the bandwidth available by an increasing amount. The restriction is implemented by limiting the number of transactions that can access the tag pipeline and therefore the RAMs in the L3 cache slice. The gear in use can either be programmed using the MPMM registers through the utility bus, or controlled through input signals. The system can then trade off bandwidth versus power based on information of what other system components are doing.

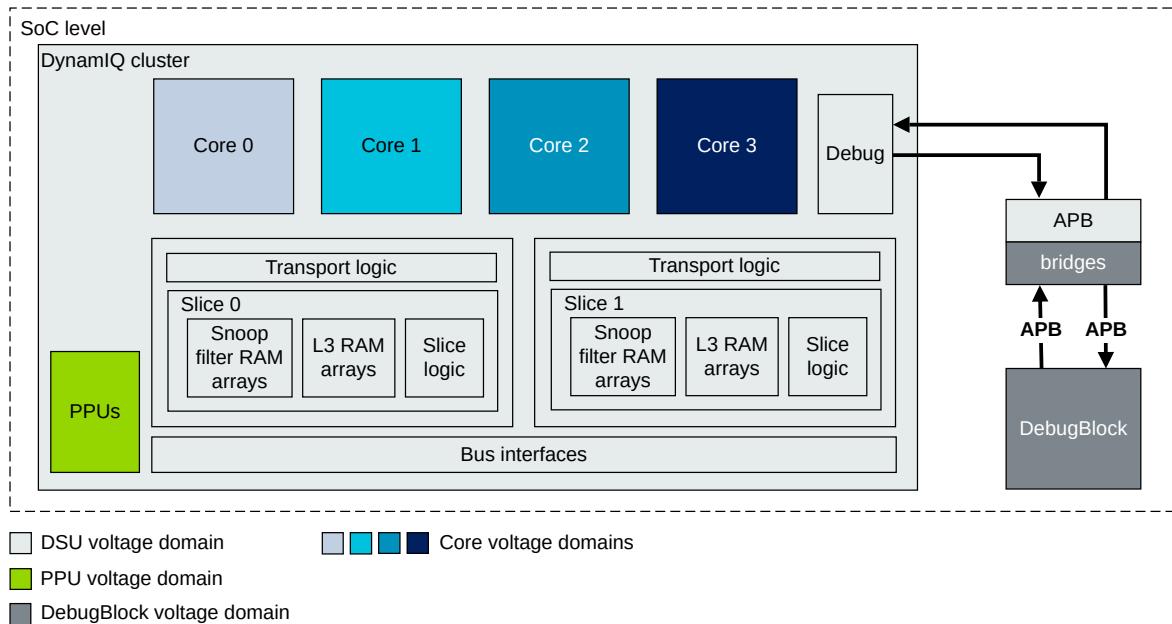
For more information on MPMM, see the *Maximum Power Mitigation Mechanism* section of the *System design* chapter in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

## 4.11 DSU-120AE voltage domains

The DSU-120AE supports each core-pair in the DSU-120AE DynamIQ™ cluster being implemented in a separate voltage domain. There is also a separate voltage domain for the DSU-120AE DynamIQ™ cluster itself.

The following figure shows the voltage domains in the cluster.

**Figure 4-5: DSU-120AE voltage domains**



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



**Note**

Implementing each core-pair in a separate voltage domain is optional, but depends on an asynchronous bridge configuration. When the bridge configuration is synchronous, all cores must be in the DSU-120AE voltage domain. 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-120AE 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.

## 5. Power and reset control with Power Policy Units

This chapter describes how to control the power mode and reset behavior for the DSU-120AE DynamIQ™ cluster, cores, and complexes using the *Power Policy Units* (PPUs).

### 5.1 The Power Policy Unit

Power mode control for the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) is provided by the *Power Policy Units* (PPUs) that are integrated into the cluster. These PPU 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-120AE 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 PPU through the utility bus to set the required power policy. The PPU 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.



Note

- 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 PPU for the cores it contains, see [Table 4-9: PPU mode and power domain states for a dual-core complex](#) on page 99.
- The cluster and all the core PPU are provided as part of the DSU-120AE.
- The implementation process automatically creates the PPU for the cluster and each core PPU, and connects these into the DSU-120AE DynamIQ™ cluster. Each PPU has a set of memory-mapped control registers which is accessed using the utility bus.

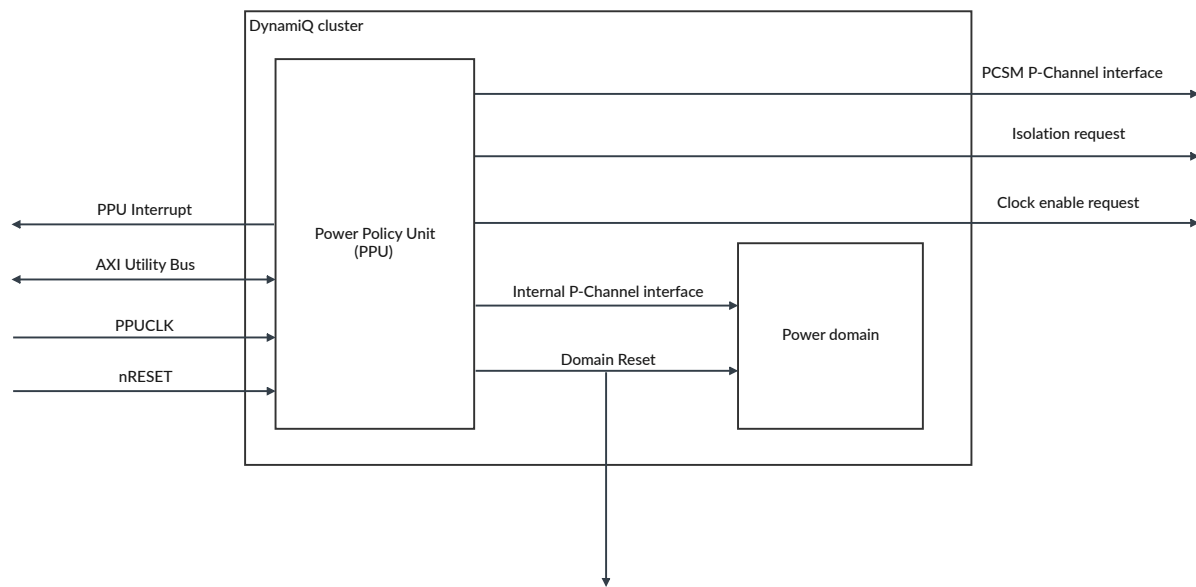
The PPU:

- Abstract away the underlying mechanics of power state machine control of the DSU-120AE. 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-120AE PPU interfaces. All interfaces are external to the DSU-120AE 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 5-1: DSU-120AE 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.



Some of the device control interface signals are exported outside of the DSU-120AE to allow control of other components that might be in the same power domain.

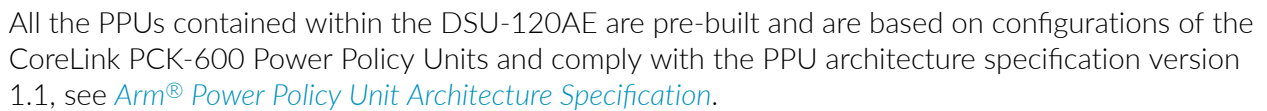
The interface provides low-level device control and ensures device quiescence. The interface comprises:

- The device interface, which consists of a P-Channel interface, see [AMBA® Low Power Interface Specification](#).
- The device control interface, which includes clock enables, resets, and isolation control.



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-120AE implementation. There are separate PCSM interfaces for each core instantiated in the cluster, and a separate PCSM for the DSU-120AE DynamIQ™ cluster itself.

**Figure 5-2: DSU-120AE PPU connections to a power-gated domain**



## 5.2 Power Policy Unit mode control for Lock-configuration and Mixed-configuration

When using Lock-configuration or Mixed-configuration, a single *Power Policy Unit* (PPU) can be used to control the power transitions for two cores.

### PPU power mode control for Lock-configuration

In Lock-configuration a single logical PPU controls the power mode requests for both the primary and redundant core. There is a single *Power Control State Machine* (PCSM) interface for the primary and redundant core. The single core PCSM is responsible for controlling the power domains for both the primary and redundant core. The single complex PCSM is responsible for both the primary and redundant complex power domains. In Lock-configuration a single logical PPU controls the power mode requests for both the primary and redundant cluster logic. There is a single PCSM interface for the primary and redundant cluster logic. The single core PCSM is responsible for controlling the power domains for both the primary and redundant cluster logic.

### PPU power mode control for Mixed-configuration

In Mixed-configuration, there is a separate logical PPU instance for each physical core.

#### PPU power mode control for Mixed-configuration, where the cores are in Split-mode

In core Split-mode each physical core is controlled by a separate PPU instance.

#### PPU power mode control for Mixed-configuration, where the cores are in Lock-mode

In core Lock-mode a single logical PPU controls the power mode requests for both the primary and redundant core. The PPU is identified by the primary core physical core numbering. For example, if the primary core is physical core 2 then the PPU 2 instance is used to control the power modes for the primary core 2 and its equivalent redundant core. The single PPU drives the PCSM interfaces for both the primary and redundant physical cores. The external core PCSMs must respond to both physical core PCSM interfaces so that the power control for the primary and redundant core is identical. The external complex PCSMs must respond to both physical complex PCSM interfaces so that the power control for the primary and redundant complex domain is identical.

#### PPU power mode control for Mixed-configuration Split-mode cluster

In cluster Split-mode a single logical PPU instance controls the power mode requests for both the primary and redundant cluster logic. The redundant logic is clock gated and remains inactive, but powered up. There is a single PCSM interface for the primary and redundant cluster logic. The single cluster PCSM is responsible for controlling the power domains for both the primary and redundant cluster logic. The redundant core is not available for use as a separate logical core in this mode, therefore it will appear to its logical PPU as if it were powered off. If you program a logical PPU that corresponds to a redundant core, then any power transition will be denied.

#### PPU power mode control for Mixed-configuration Lock-mode cluster

In cluster Lock-mode a single logical PPU instance controls the power mode requests for both the primary and redundant cluster logic. There is a single PCSM interface for the

primary and redundant cluster logic. The single cluster PCSM is responsible for controlling the power domains for both the primary and redundant cluster logic.

## 5.3 Power policy unit operation

The *Power Policy Unit* (PPU) supports all the DSU-120AE DynamIQ™ cluster power modes (ON, OFF, FUNC\_RET, FULL\_RET, MEM\_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-120AE 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 management.

### 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.



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 [4.7 Cluster PPU mode transitions](#) on page 89. 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[<y>], where y is the core instance number.

For the *DynamIQ™ Shared Unit-120AE* (DSU-120AE), a PPU is programmed by the *System Control Processor* (SCP) through the DSU-120AE utility bus. The SCP programs the PPU mode or range of

PPU modes that it wants the DSU-120AE 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.



Before performing a write to any of the PPU registers, if the DSU-120AE is configured for Lock-configuration or Mixed-configuration, the registers must be unlocked by writing to the CLUSTERAE\_CLUSTERWRITEKEY register. This must be done for each write access to the PPU registers. For more information, see [5.4 PPU and CLUSTERAE register protection](#) on page 110.

The PPU and the power domain communicate through the device P-Channel. This device P-Channel is internal to the DSU-120AE. 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-120AE Configuration and Integration Manual](#).

### 5.3.1 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.

### 5.3.2 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.

### 5.3.3 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.

**Note**

By default, the cluster powers on in the ALL SLICE, FULL RAM operating mode. However, if the OPRES field in the CLUSTERPWRCTLR register is set, then when the cluster next powers on, the operating mode is set instead to the operating mode that was in use at the time the cluster was powered down.

This state is stored in the PPU logic, and therefore can only be used if the PPUs remain powered on while the cluster is powered OFF.

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.

## 5.4 PPU and CLUSTERAE register protection

When *DynamIQ™ Shared Unit-120AE* (DSU-120AE) is configured for Lock-configuration or Mixed-configuration (including Split-mode), the *Power Policy Unit* (PPU) and clusterAE programming registers, that reside in the PDTOP power domain, are protected against incorrect writes to these registers.

The protection is implemented by using a write key register (CLUSTERAE\_CLUSTERWRITEKEY), which has to be written to before any write access can be performed in any of the core PPU registers, cluster PPU registers, or clusterAE registers.

**Note**

- To unlock CLUSTERAE registers and *Power Policy Unit* (PPU) registers, write to the CLUSTERAE\_CLUSTERWRITEKEY register, offset address 0x050060, value 0x000000BA.
- Each time you want to write to any of the PPU and CLUSTERAE registers, you must first write to the CLUSTERAE\_CLUSTERWRITEKEY register. This is because the key is reset after any successful write to the PPU or CLUSTERAE registers. Therefore, the key grants access for only a single write.

If an incorrect value is written to the CLUSTERAE\_CLUSTERWRITEKEY register, then the subsequent write access to the PPU or CLUSTERAE register receives a SLVERR response and the write does not take effect. Similarly, if a PPU or CLUSTERAE register is written to without previously writing to the CLUSTERAE\_CLUSTERWRITEKEY register, then the register receives a SLVERR response and again the write is ignored.



If the DSU-120AE is configured for Split-configuration, there is no register protection, and therefore no requirement to write to the CLUSTERAE\_CLUSTERWRITEKEY register.

For the CLUSTERAE\_CLUSTERWRITEKEY register description, see [B.1.6.8 CLUSTERAE\\_CLUSTERWRITEKEY, Cluster Write Key Register](#) on page 621.

## 5.5 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 subordinate port.

Accesses to PPU registers over the utility bus must be 32-bits long. 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 access on the utility bus. Accesses to these registers with the Non-secure bit set are treated as **RAZ/WI**.

## 5.6 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-120AE 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.

### 5.6.1 External cluster PPU registers

The *Power Policy Unit* (PPU) registers for the DSU-120AE DynamIQ™ cluster are only accessible from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the cluster PPU registers that are accessed externally (memory-mapped) from the utility bus of the DSU-120AE. For more information about a register, click on the register name in the table.



- You must access the cluster system control registers from the Secure state.
- The cluster PPU registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- 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 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 5-1: Cluster PPU register summary**

Offset	Name	Reset	Width	Description
0x000	<a href="#">PPU_PWPR</a>	See individual bit resets.	32-bit	Power Policy Register
0x004	<a href="#">PPU_PMER</a>	See individual bit resets.	32-bit	Power Mode Emulation Enable Register
0x008	<a href="#">PPU_PWSR</a>	See individual bit resets.	32-bit	Power Status Register
0x010	<a href="#">PPU_DISR</a>	See individual bit resets.	32-bit	Device Interface Input Current Status Register
0x014	<a href="#">PPU_MISR</a>	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register
0x018	<a href="#">PPU_STSR</a>	See individual bit resets.	32-bit	Stored Status Register
0x01C	<a href="#">PPU_UNLK</a>	See individual bit resets.	32-bit	Unlock Register
0x020	<a href="#">PPU_PWCR</a>	See individual bit resets.	32-bit	Power Configuration Register
0x024	<a href="#">PPU_PTCR</a>	See individual bit resets.	32-bit	Power Mode Transition Register
0x030	<a href="#">PPU_IMR</a>	See individual bit resets.	32-bit	Interrupt Mask Register
0x034	<a href="#">PPU_AIMR</a>	See individual bit resets.	32-bit	Additional Interrupt Mask Register
0x038	<a href="#">PPU_ISR</a>	See individual bit resets.	32-bit	Interrupt Status Register
0x03C	<a href="#">PPU_AISR</a>	See individual bit resets.	32-bit	Additional Interrupt Status Register
0x040	<a href="#">PPU_IESR</a>	See individual bit resets.	32-bit	Input Edge Sensitivity Register
0x044	<a href="#">PPU_OPSR</a>	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register
0x050	<a href="#">PPU_FUNRR</a>	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register
0x054	<a href="#">PPU_FULRR</a>	See individual bit resets.	32-bit	Full Retention RAM Configuration Register
0x058	<a href="#">PPU_MEMRR</a>	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register
0x170	<a href="#">PPU_DCDR0</a>	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0
0x174	<a href="#">PPU_DCDR1</a>	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1
0xFB0	<a href="#">PPU_IDR0</a>	See individual bit resets.	32-bit	PPU Identification Register 0
0xFB4	<a href="#">PPU_IDR1</a>	See individual bit resets.	32-bit	PPU Identification Register 1
0xFC8	<a href="#">PPU_IIDR</a>	See individual bit resets.	32-bit	Implementation Identification Register
0xFCC	<a href="#">PPU_AIDR</a>	See individual bit resets.	32-bit	Architecture Identification Register



Offset	Name	Reset	Width	Description
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1
0xFE8	PPU_PIDR2	See individual bit resets.	32-bit	PPU Peripheral Identification Register 2
0xFEC	PPU_PIDR3	See individual bit resets.	32-bit	PPU Peripheral Identification Register 3
0xFF0	PPU_CIDR0	See individual bit resets.	32-bit	PPU Component Identification Register 0
0xFF4	PPU_CIDR1	See individual bit resets.	32-bit	PPU Component Identification Register 1
0xFF8	PPU_CIDR2	See individual bit resets.	32-bit	PPU Component Identification Register 2
0xFFC	PPU_CIDR3	See individual bit resets.	32-bit	PPU Component Identification Register 3

## 5.6.2 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-120AE DynamIQ™ cluster.



In the following table:

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

**Table 5-2: 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
FULL_RET	0x5	0x5	0x0020
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-120AE DynamIQ™ cluster operating mode encodings for PPU\_PWPR.OP\_POLICY bit field.

**Table 5-3: Operating mode encodings for PPU\_PWPR.OP\_POLICY bit field**

Active slices	Active RAMs		
	Snoop Filter Only (SFONLY)	HALF RAM	FULL RAM
ONE SLICE	0x0	0x1	0x3
HALF SLICES	0x8	0x9	0xB
ALL SLICES	0x4	0x5	0x7

The following table shows the DSU-120AE DynamIQ™ cluster operating mode encodings for CLUSTERPCSMPSTATE[7:4] and CLUSTERPPUHWSTAT[31:16].

In the following table:



**Note**

- PCSMPSTATE[7:4] refers to CLUSTERPCSMPSTATE[7:4]
- PPUHWSTAT[31:16] refers to CLUSTERPPUHWSTAT[31:16]
- For ALL\_SLICES, there are pairs of values of PCSMPSTATE[7:4] that are equivalent. There is no significance in meaning between each of the two different encodings. In some situations, the DSU-120AE might generate PCSM transition requests between equivalent encodings.

**Table 5-4: Operating mode enumeration for the DSU-120AE cluster**

Operating mode name	PPU_PWPR.OP_POLICY	PCSMPSTATE[7:4]	PPUHWSTAT[31:16]
ONE SLICE, SFONLY	0x0	0x0	0x01
ONE SLICE, HALF RAM	0x1	0x1	0x02
ONE SLICE, FULL RAM	0x3	0x3	0x08
HALF SLICES, SFONLY	0x8	0x8	0x100
HALF SLICES, HALF RAM	0x9	0x9	0x200
HALF SLICES, FULL RAM	0xB	0xB	0x800
ALL SLICES, SFONLY	0x4	0x4 or 0xC	0x10
ALL SLICES, HALF RAM	0x5	0x5 or 0xD	0x20
ALL SLICES, FULL RAM	0x7	0x7 or 0xF	0x80

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

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

Operating mode name	PPU_PWPR.OP_POLICY	Default configuration (with L3 cache and SCU)
ONE SLICE, SFONLY	0x0	Supported
ONE SLICE, HALF RAM	0x1	
ONE SLICE, FULL RAM	0x3	
HALF SLICES, SFONLY	0x8	
HALF SLICES, HALF RAM	0x9	
HALF SLICES, FULL RAM	0xB	

Operating mode name	PPU_PWPR.OP_POLICY	Default configuration (with L3 cache and SCU)
ALL SLICES, SFONLY	0x4	
ALL SLICES, HALF RAM	0x5	
ALL SLICES, FULL RAM	0x7	

**Note**

When programming the OP\_POLICY field in static mode, ensure that the new value is a single valid transition from the current value, otherwise the request will be denied. Changing the slice bits and the RAM bits at the same time might also be denied.

When using dynamic mode, Arm recommends that you set the the OP\_POLICY field to 0 and use the CLUSTERPWRCTLR register to request changes to the operating mode. Setting OP\_POLICY to any other value might prevent transitions to some operating modes even if they are considered higher than the programmed value.

## 5.7 Core power mode control

There are separate *Power Policy Units* (PPUs) for each of the cores in the DSU-120AE DynamIQ™ cluster.

A component such as a *System Control Processor* (SCP) can program each of the core PPU's 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.

### 5.7.1 External core PPU registers

Each core *Power Policy Unit* (PPU) in the DSU-120AE 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.

The summary table provides an overview of all the PPU registers for a single core in the DSU-120AE. For more information about a register, click on the register name in the table.

**Note**

- You must access the cluster system control registers from the Secure state.
- The core PPU registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.

- These register descriptions are configuration of the PPU architecture, see [Arm® Power Policy Unit Architecture Specification](#) for more details.
- 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 base address for the core PPU registers is  $0x\langle n \rangle 80000$ , 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 5-6: Core PPU register summary**

Offset	Name	Reset	Width	Description
0x000	PPU_PWPR	See individual bit resets.	32-bit	Power Policy Register
0x004	PPU_PMER	See individual bit resets.	32-bit	Power Mode Emulation Enable Register
0x008	PPU_PWSR	See individual bit resets.	32-bit	Power Status Register
0x010	PPU_DISR	See individual bit resets.	32-bit	Device Interface Input Current Status Register
0x014	PPU_MISR	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register
0x018	PPU_STSR	See individual bit resets.	32-bit	Stored Status Register
0x01C	PPU_UNLK	See individual bit resets.	32-bit	Unlock Register
0x020	PPU_PWCR	See individual bit resets.	32-bit	Power Configuration Register
0x024	PPU_PTCR	See individual bit resets.	32-bit	Power Mode Transition Register
0x030	PPU_IMR	See individual bit resets.	32-bit	Interrupt Mask Register
0x034	PPU_AIMR	See individual bit resets.	32-bit	Additional Interrupt Mask Register
0x038	PPU_ISR	See individual bit resets.	32-bit	Interrupt Status Register
0x03C	PPU_AISR	See individual bit resets.	32-bit	Additional Interrupt Status Register
0x040	PPU_IESR	See individual bit resets.	32-bit	Input Edge Sensitivity Register
0x044	PPU_OPSR	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register
0x050	PPU_FUNRR	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register
0x054	PPU_FULRR	See individual bit resets.	32-bit	Full Retention RAM Configuration Register
0x058	PPU_MEMRR	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register
0x170	PPU_DCDR0	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0
0x174	PPU_DCDR1	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1
0xFB0	PPU_IDR0	See individual bit resets.	32-bit	PPU Identification Register 0
0xFB4	PPU_IDR1	See individual bit resets.	32-bit	PPU Identification Register 1
0xFC8	PPU_IIDR	See individual bit resets.	32-bit	Implementation Identification Register
0xFCC	PPU_AIDR	See individual bit resets.	32-bit	Architecture Identification Register
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0

Offset	Name	Reset	Width	Description
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1
0xFE8	PPU_PIDR2	See individual bit resets.	32-bit	PPU Peripheral Identification Register 2
0xFEC	PPU_PIDR3	See individual bit resets.	32-bit	PPU Peripheral Identification Register 3
0xFF0	PPU_CIDR0	See individual bit resets.	32-bit	PPU Component Identification Register 0
0xFF4	PPU_CIDR1	See individual bit resets.	32-bit	PPU Component Identification Register 1
0xFF8	PPU_CIDR2	See individual bit resets.	32-bit	PPU Component Identification Register 2
0xFFC	PPU_CIDR3	See individual bit resets.	32-bit	PPU Component Identification Register 3

## 5.7.2 Encodings for core power modes

The core *Power Policy Unit* (PPU) register bitfield PPU\_PWPR.PWR\_POLICY encodes the supported power modes for the cores.

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



**Note**

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 5-7: Power mode enumeration for the cores in the DSU-120AE 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
ON	0x8	0x8	0x0100
WARM_RST	0x9	0x8	0x0200
DBG_RECOV	0xA	0x8	0x0400

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

For information on the PPU registers, see [5.6.1 External cluster PPU registers](#) on page 111 and [5.7.1 External core PPU registers](#) on page 115.

### Related information

[1.8 Core, complex, and processing element numbering](#) on page 47

## 5.8 Selecting different modes in Mixed-configuration

In Mixed-configuration, the three different modes, namely Lock-mode, Split-mode, and Hybrid-mode are selected at cluster reset time using pins. The CORESLDEFAULT and CLUSTERSLDEFAULT signals are used to select the mode, and they only take effect when the nRESET pin is deasserted. The mode selected is shown in the CLUSTERAE\_CLUSTERSLCTLR and CLUSTERAE\_CORESLCTLR registers which are read only.

Updates to the CORESLDEFAULT and CLUSTERSLDEFAULT signals take effect when the related cluster or core logic exits reset.

The following table shows for the mapping between the Mixed-configuration modes and the register programming.

**Table 5-8: Permitted Mixed-configuration mode selection**

Mixed-configuration mode	CLUSTERSLDEFAULT and CLUSTERAE_CLUSTERSLCTLR	CORESLDEFAULT and CLUSTERAE_CORESLCTLR
Split-mode	Split	Split
Lock-mode	Lock	Lock
Hybrid-mode	Lock	Split

## 5.9 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.

### 5.9.1 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-120AE 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 [5.9.3 Programming sequence for an interrupt controller to control transitions between On and Off mode](#) on page 120.



Note

- Steps 2 and 4 are only required if you need to know when the power transition has completed. Otherwise they can be omitted.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.
- 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.

## 5.9.2 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-120AE 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.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value

0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

## 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.

## 5.9.3 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-120AE 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.



- In this task, <y> is the core instance number.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

## 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 or power down core <y>.



- To power up core <y>, assert the COREWAKEREQUEST[<y>] signal.
- To power down core <y>:
  - 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.



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.

## 5.10 Explicit reset of the cluster and cores

There are several reset scenarios for part, or all, of the DSU-120AE DynamIQ™ cluster and the cores.

In each of the reset scenarios you must ensure that the sequences are followed exactly.



Note

- In two of the reset scenarios ([5.10.4 Warm reset of the cluster, excluding the PPU](#)s on page 123 and [5.10.6 Reset of the cluster, excluding the PPU](#)s, [retaining the cache contents for debug](#) on page 128) four different methods are given for performing the same reset. Arm strongly recommends the methods of either programming the CLUSTERAE\_CLUSTERRECOV register, or a signal reset using the input/output domain P-Channel signals as compared to using *Power Policy Unit* (PPU) programming. This is because they involve less programming steps and they also automatically restore the cluster operating mode before the reset was taken, unlike the PPU programming method.
- When using the methods of either programming the CLUSTERAE\_CLUSTERRECOV register, or a signal reset using the input/output domain P-Channel signals, Arm strongly recommends that the PPU's are placed in dynamic mode. This enables the cluster operating mode that was present before the reset to be automatically restored.
- In the reset scenarios, if using the PPU programming method, the current cluster operating mode must be saved. This is because WARM\_RST and DBG\_RECOV power modes do not have an associated operating mode and therefore transitioning to the modes the operating state is not preserved. By saving the operating mode, this ensures that the same operating mode can be restored when leaving the power states.
- The PPU's and associated logic prevents unsupported transactions from occurring.

The scenarios for resetting all or part of the DSU-120AE DynamIQ™ cluster are:

- [5.10.1 Powerup \(Cold\) reset](#) on page 122
- [5.10.2 Core software initiated Warm reset of an individual core](#) on page 122
- [5.10.3 Core software initiated Cold or Warm reset of the cluster, excluding the PPU](#)s on page 123
- [5.10.4 Warm reset of the cluster, excluding the PPU](#)s on page 123
- [5.10.5 Cold reset of the whole cluster, including the PPU](#)s, retaining cache contents for debug on page 127
- [5.10.6 Reset of the cluster, excluding the PPU](#)s, retaining the cache contents for debug on page 128

### 5.10.1 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-120AE* (DSU-120AE) including the *Power Policy Units* (PPUs).

#### Procedure

1. Assert the nRESET signal for a minimum of ten PPUCLK cycles.
2. Deassert the nRESET signal.
3. Program the PPU for the cluster to On power mode, see [5.9.1 Programming sequence to bring the cluster and cores from Off to On mode](#) on page 118.
4. Program the PPU for each required core to On power mode, see [5.9.1 Programming sequence to bring the cluster and cores from Off to On mode](#) on page 118.

### 5.10.2 Core software initiated Warm reset of an individual core

Software running on a core in the cluster can use the Reset Management Register (RMR) to program a core to Warm reset. This mechanism was provided to switch between execution states, however all cores in this generation only support a single execution state, AArch64. Therefore, Arm recommends that this mechanism is not used as it may be removed in future generations.

#### About this task

The core RMR register bit field RR (RMR.RR) can be used by software to request a Warm reset of a core independent of the *Power Policy Unit* (PPU) control. However, an interrupt, debug access, or an *Error Correcting Code* (ECC) error detected during the automatic cache clean triggered by the reset request can prevent the Warm reset from being asserted.

If software requires the use of RMR.RR then the following actions must be taken to ensure the reset completes:

#### Procedure

1. Ensure that ECC fault detection is disabled before writing to the RMR.RR register bit field.
2. The core software must use the following the code sequence to guarantee the request for Warm reset.

```
; In addition, interrupts and debug requests for the PE should be disabled
```

```
; in the system before running this sequence to ensure the WFI suspends
execution,
    MOV Wy, #3 ; y is any register
    DSB ; ensure all stores etc are complete
    MSR RMR_ELx, Wy ; request the reset
    ISB ; synchronize change to the RMR
Loop
    WFI ; enter a quiescent state
    B Loop
```

See [Arm® Architecture Reference Manual for A-profile architecture](#), Issue K, section D 1.5.2 Reset types, Rule: RSGXSW for more information.

### 5.10.3 Core software initiated Cold or Warm reset of the cluster, excluding the PPU

To reset the cores and cluster, not including the *Power Policy Units* (PPUs), follow the sequence below. For a Cold reset program, the PPU to Off power mode. For a Warm reset, program the PPU to Emulated Off power mode.

#### About this task

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



Note

If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register, you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not-required.

#### Procedure

1. Use software running on each core to set the IMP\_CPUPWRCTLR\_EL1.CORE\_PWRDN\_EN bit.
2. Use software running on each core to execute a `WFI` instruction.
3. Program the *Power Policy Unit* (PPU) for each core to Off mode (Cold reset) or Emulated off mode (Warm reset).
4. Program the PPU for the cluster to Off power mode or Emulated off mode.



Note

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

5. Program the PPU for the cluster to On power mode.
6. For each of the cores, program their corresponding PPU to On power mode.

## 5.10.4 Warm reset of the cluster, excluding the PPUs

To provide a Warm reset to all the cores and the cluster use one of the following four methods. This type of reset can be used to recover from a watchdog timeout or a RAS error.

Arm recommends using either methods 2, 3, or 4 when resetting the cores and cluster rather than method 1.



Note

Because the WARM\_RST and DBG\_RECOV power modes do not wait for transactions to reach a quiescent state before entry, the cluster might be in any power state. Any external component that is communicating with the power domains being reset, for example the system interconnect, must also be reset to ensure any outstanding transactions are terminated. If there is a power transition or a clock gating transition in progress at the time, then the transition might depend on other transactions completing. Therefore, this can prevent the completion of the power or clock transition which in turn can prevent the entry into WARM\_RST or DBG\_RECOV mode. Therefore, Arm recommends resetting the cluster by using the CLUSTERRECOV register or the input/output domain P-Channel signals (methods 2 to 4) over direct programming of the PPUs (method 1), as it increases the chances of a successful reset as well as being a simpler sequence.

### Method 1: Warm reset of the cluster using PPU programming

To apply a Warm reset to the cores and cluster use the following sequence.



Note

If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

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.

## Method 2: Using the CLUSTERRECOV register, and not held in reset.

To apply a Warm reset to the cores and the cluster, and immediately come out of reset, use the following sequence.



Note

Arm strongly recommends that the reset sequence is done automatically. However, if you decide to leave the cluster and core PPU in static mode, for example for debugging, then you must ensure the *System Control Processor* (SCP) responds to the DEVACTIVE\* signal requests. This can be done either by polling the PPUs or by programming the PPUs to raise interrupts to notify your system power controller when the cluster or a core needs to transition to a new power mode. Your system power controller must respond appropriately. For more information on programming the PPUs for static power management and raising interrupts, see section *Configure the PPU for static power management* in chapter *System design* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

1. If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode), write to the cluster register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA, to unlock access to the CLUSTERAE\_CLUSTERRECOV register. If the DSU-120AE is configured for Split-configuration, then this step is not required.
2. Write to the cluster register CLUSTERAE\_CLUSTERRECOV, offset address 0x050050, value 0x0000\_0000\_0000\_0010. This places the cores and cluster in WARM\_RST power mode.

Once the reset is applied, the hardware automatically places the cluster and cores in the ON power mode. If the PPUs are in dynamic mode, then the cluster operating mode is also preserved and so no programming of the PPUs is required.

## Method 3: Using CLUSTERRECOV register, and held in reset

To apply a Warm reset to the cores and the cluster, and to control the exit from reset, use the following sequence:



Note

- Arm strongly recommends that the reset sequence is done automatically. However, if you decide to leave the cluster and core PPUs in static mode, for example for debugging, then you must ensure the *System Control Processor* (SCP) responds to the DEVACTIVE\* signal requests. This can be done either by polling the PPUs or by programming the PPUs to raise interrupts to notify your system power controller when the cluster or a core needs to transition to a new power mode. Your system power controller must respond appropriately. For more information on programming the PPUs for static power management and raising interrupts, see section *Configure the PPU for static power management* in chapter *System design* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to the CLUSTERAE\_CLUSTERRECOV register, you must first unlock access to this register. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY,

offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

1. Write to the cluster register CLUSTERAE\_CLUSTERRECOV, offset address 0x050050 , value 0x0000\_0000\_0000\_0030 . This requests that the cores and cluster are placed in the WARM\_RST power mode.
2. Wait until the cluster register CLUSTERAE\_CLUSTERRECOV, offset address 0x050050 , reads value 0x0000\_0000\_0000\_0070. This indicates all the cores and cluster are held in the WARM\_RST power mode.
3. When required, exit from WARM\_RST and transition the cores and cluster to the ON power mode by writing to the CLUSTERAE\_CLUSTERRECOV register, offset address 0x050050 , value 0x0000\_0000\_0000\_0000.

When the cluster and cores are placed in the ON power mode, if the PPU's are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPU's is required.

#### Method 4: Warm reset of the cluster using INPUTDOMAINPACTIVE and OUTPUTDOMAINPACTIVE signals

To apply a Warm reset to the cores and cluster using either the input or output P-Channel interfaces, and to control when the cores and cluster exit from reset, use the following sequence:



Note

Arm strongly recommends that the reset sequence is done automatically. However, if you decide to leave the cluster and core PPU's in static mode, for example for debugging, then you must ensure the *System Control Processor* (SCP) responds to the DEVPACTIVE\* signal requests. This can be done either by polling the PPU's or by programming the PPU's to raise interrupts to notify your system power controller when the cluster or a core needs to transition to a new power mode. Your system power controller must respond appropriately. For more information on programming the PPU's for static power management and raising interrupts, see section *Configure the PPU for static power management* in chapter *System design* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

1. Drive the INPUTDOMAINPACTIVE[9] signal or the OUTPUTDOMAINPACTIVE[9] signal HIGH (depending on what interface you are using) to request WARM\_RST power mode for the cluster and cores. This ensures the cores and cluster are placed in the WARM\_RST power mode.
2. Wait until the signal CLUSTERPPUHWSTAT[9] is driven HIGH. This indicates that the cluster and cores have been placed in the WARM\_RST power mode.



Note

Optionally, keeping the INPUTDOMAINPACTIVE[9] or OUTPUTDOMAINPACTIVE[9] driven HIGH, after the CLUSTERPPUHWSTAT[9] indicates HIGH, holds the cores and the cluster in Warm reset.

3. When required to exit from WARM\_RST power mode, drive either the INPUTDOMAINPACTIVE[9] signal or OUTPUTDOMAINPACTIVE[9] signal LOW depending on what interface you are using.

Once the reset is applied, the hardware automatically places the cluster and cores to the ON power mode. If the PPU's are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPU's is required.

### 5.10.5 Cold reset of the whole cluster, including the PPU's, retaining cache contents for debug

To provide a Cold reset to the whole cluster, including the *Power Policy Units* (PPU's) but retaining all cache contents for debug, use the following method.

#### About this task



Note

- This method is only suitable for configurations with the `PPU_RST_STATE` configuration parameter set to `FALSE`. For more information on the build-time configuration parameters, see section *hayden.yaml configuration parameters* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.
- Because this method resets the PPU's, the *Power Control State Machine* (PCSM) interfaces are also reset to the Off power mode. Therefore, you must ensure that your implemented PCSM logic must be aware of this sequence and not remove power during this period.
- Because the WARM\_RST and DBG\_RECOV power modes do not wait for transactions to reach a quiescent state before entry, the cluster might be in any power state. Any external component that is communicating with the power domains being reset, for example the system interconnect, must also be reset to ensure any outstanding transactions are terminated. If there is a power transition or a clock gating transition in progress at the time, then the transition might depend on other transactions completing. Therefore, this can prevent the completion of the power or clock transition which in turn can prevent the entry into WARM\_RST or DBG\_RECOV mode.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register `CLUSTERAE_CLUSTERWRITEKEY`, offset address `0x050060`, value `0x0000_0000_0000_00BA`. If the DSU-120AE is configured for Split-configuration, then this step is not required.

#### Procedure

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. Drive the signal nRESET LOW for a minimum of ten PPUCLK cycles.

3. For each of the cores, either leave them in the OFF power mode, or change them to DBG\_RECOV power mode by writing to the core PPU\_PWPR for core <y>, address 0x<y>80000, value 0x0000000A.
4. Write to the cluster PPU\_PWPR, address 0x030000, value 0x0000000A. This sets the cluster to the DBG\_RECOV power mode.
5. 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.
6. For each core, 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.

### 5.10.6 Reset of the cluster, excluding the PPUs, retaining the cache contents for debug

To provide a Cold reset or Warm reset the cluster and cores, excluding the *Power Policy Units* (PPUs) and retaining the cache contents for debug, use one of the following four methods.

- Arm recommends using either methods 2, 3, or 4 when resetting the cores and cluster rather than method 1.



Note

Because the WARM\_RST and DBG\_RECOV power modes do not wait for transactions to reach a quiescent state before entry, the cluster might be in any power state. Any external component that is communicating with the power domains being reset, for example the system interconnect, must also be reset to ensure any outstanding transactions are terminated. If there is a power transition or a clock gating transition in progress at the time, then the transition might depend on other transactions completing. Therefore, this can prevent the completion of the power or clock transition which in turn can prevent the entry into WARM\_RST or DBG\_RECOV mode. Arm recommends resetting the cluster by using the CLUSTERRECOV register or the input/output domain P-Channel signals (methods 2 to 4) over direct programming of the PPUs (method 1), as it increases the chances of a successful reset as well as being a simpler sequence.

- The setting of the DBG\_RECOV\_PORST\_EN bit in the PPU\_PTCR register determines if the reset is a Warm reset or a Cold reset.
- The setting of the PPU\_PTCR.DBG\_RECOV\_PORST\_EN register bit must be consistent across all PPUs (the cluster and all the cores) otherwise the results are **UNPREDICTABLE**.

#### Method 1: Using PPU programming

Arm does not recommend using this method, because it does not guarantee all outstanding power or clock transactions are completed, before the cluster and cores are placed in DBG\_RECOV power mode.





Note

If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

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 in OFF mode, 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. For each core, and the cluster, program up the corresponding PPU register bit PPU\_PTCR.DBG\_RECOV\_PORST\_EN, for either Cold or Warm reset:

0

Warm reset

1

Cold reset

This must be done for all cores unless in the OFF power mode.

5. Write to the core PPU\_PWPR for core <y>, address 0x<y>80000, value 0x0000000A, which sets the core to the DBG\_RECOV power mode. This must be done for all cores unless:
  - The core is in OFF power mode.
  - PPU\_PTCR.DBG\_RECOV\_PORST\_EN = 0 and
    - the core is either in OFF\_EMU power mode or
    - the core is in WARM\_RST power mode.
6. Write to the cluster PPU\_PWPR, address 0x030000, value 0x0000000A. This sets the cluster to the DBG\_RECOV power mode.
7. 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.
8. 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.

## Method 2: Using the CLUSTERRECOV register, and not held in reset

To apply a Cold or Warm reset to the cores and the cluster, and immediately come out of reset, use the following sequence.



- For the reset sequence to take effect automatically, ensure steps 1 and 2 are followed. This sets the cluster and core PPU to dynamic mode, by setting the PWR\_DYN\_EN bit in PPU\_PWPR registers.
- Arm strongly recommends that the reset sequence is done automatically. However, if you decide to leave the cluster and core PPU in static mode, for example for debugging, then you must ensure the *System Control Processor* (SCP) responds to the DEVACTIVE\* requests. This can be done either by polling the PPU or by programming the PPU to raise interrupts to notify your system power controller when the cluster or a core needs to transition to a new power mode. Your system power controller must respond appropriately. For more information on programming the PPU for static power management and raising interrupts, see section *Configure the PPU for static power management* in chapter *System design* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to the CLUSTERAE\_CLUSTERRECOV register or PPU registers, you must first unlock access to these registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

1. Write to the core PPU\_PWPR for core <y>, address 0x<y>80000, value 0x100, which sets the core PPU to dynamic mode. This must be done for all cores even if they are in the OFF power mode.
2. Write to the cluster PPU\_PWPR, address 0x030000, value 0x100. This sets the cluster PPU to dynamic mode.
3. For each core, and the cluster program up the corresponding PPU register bit PPU\_PTCR.DBG\_RECOV\_PORST\_EN, for either Cold or Warm reset:

**0**

Warm reset

**1**

Cold reset

This must be done for all cores unless in the OFF power mode.

4. Write to the cluster register CLUSTERAE\_CLUSTERRECOV, offset address 0x050050, value 0x0000\_0000\_0000\_0001. This ensures any outstanding power or clock gating transactions are terminated before placing the cores and cluster in DBG\_RECOV power mode.

Once the reset is applied, the hardware automatically places the cluster and cores in the ON power mode. If the PPU are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPU is required.

### Method 3: Using the CLUSTERRECOV register, and held in reset

To apply a Cold or Warm reset to the cores and the cluster, and to control the exit from reset, use the following sequence:



- For the reset sequence to take effect automatically, ensure steps 1 and 2 are followed. This sets the cluster and core PPU to dynamic mode, by setting the PWR\_DYN\_EN bit in PPU\_PWPR registers.
- Arm strongly recommends that the reset sequence is done automatically. However, if you decide to leave the cluster and core PPU in static mode, for example for debugging, then you must ensure the *System Control Processor* (SCP) responds to the DEVPACTIVE\* requests. This can be done either by polling the PPU or by programming the PPU to raise interrupts to notify your system power controller when the cluster or a core needs to transition to a new power mode. Your system power controller must respond appropriately. For more information on programming the PPU for static power management and raising interrupts, see section *Configure the PPU for static power management* in chapter *System design* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to the CLUSTERAE\_CLUSTERRECOV register or PPU registers, you must first unlock access to these registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

1. Write to the core PPU\_PWPR for core <y>, address 0x<y>80000, value 0x100, which sets the core PPU to dynamic mode. This must be done for all cores even if they are in the OFF power mode.
2. Write to the cluster PPU\_PWPR, address 0x030000, value 0x100. This sets the cluster PPU to dynamic mode.
3. For each core, and the cluster program up the corresponding PPU register bit PPU\_PTCR.DBG\_RECOV\_PORST\_EN, for either Cold or Warm reset:

0

Warm reset

1

Cold reset

This must be done for all cores unless in the OFF power mode.

4. Write to the cluster register CLUSTERAE\_CLUSTERRECOV, offset address 0x050050, value 0x0000\_0000\_0000\_0003. This ensures any outstanding power or clock gating transactions are terminated before requesting the cores and cluster are placed in the DBG\_RECOV power mode.

5. Wait until the cluster register CLUSTERAE\_CLUSTERRECOV, offset address 0x050050, reads value 0x0000\_0000\_0000\_0007. This indicates all the cores and cluster are held in the DBG\_RECOV power mode.
6. When required, exit from the DBG\_RECOV power mode and transition the cores and cluster to the ON power mode by writing to the CLUSTERAE\_CLUSTERRECOV register, offset address 0x050050, value 0x0000\_0000\_0000\_0000.

When the cluster and cores are placed in the ON power mode, if the PPU's are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPU's is required.

#### Method 4: Reset of the cluster using INPUTDOMAINACTIVE and OUTPUTDOMAINACTIVE signals

To apply a Cold reset to the cores and cluster using either the input or output P-Channel interfaces, and to control the exit from reset, use the following sequence:



- Arm strongly recommends that the reset sequence is done automatically. However, if you decide to leave the cluster and core PPU's in static mode, for example for debugging, then you must ensure the *System Control Processor* (SCP) responds to the DEVPACTIVE\* requests. This can be done either by polling the PPU's or by programming the PPU's to raise interrupts to notify your system power controller when the cluster or a core needs to transition to a new power mode. Your system power controller must respond appropriately. For more information on programming the PPU's for static power management and raising interrupts, see section *Configure the PPU for static power management* in chapter *System design* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.
- If the DSU-120AE is configured for Lock-configuration or Mixed-configuration (including Split-mode) then each time before writing to a PPU register you must first unlock the PPU registers. To do this, write to the register CLUSTERAE\_CLUSTERWRITEKEY, offset address 0x050060, value 0x0000\_0000\_0000\_00BA. If the DSU-120AE is configured for Split-configuration, then this step is not required.

1. Write to the core PPU\_PWPR for core <y>, address 0x<y>80000, value 0x100, which sets the core PPU to dynamic mode. This must be done for all cores even if they are in the OFF power mode.
2. Write to the cluster PPU\_PWPR, address 0x030000, value 0x100. This sets the cluster PPU to dynamic mode.
3. For each core, and the cluster program up the corresponding PPU register bit PPU\_PTCR.DBG\_RECOV\_PORST\_EN, for either Cold or Warm reset:

0

Warm reset

1

Cold reset

This must be done for all cores unless in the OFF power mode.

4. Drive the INPUTDOMAINPAIVE[10] signal or the OUTPUTDOMAINPAIVE[10] signal HIGH (depending on what interface you are using) to request DBG\_RECOV power mode for the cluster and cores. This ensures any outstanding power or clock gating transactions are terminated before requesting the cores and cluster are placed in the DBG\_RECOV power mode.
5. Wait until the signal CLUSTERPPUHWSTAT[10] is driven HIGH. This indicates that the cluster and cores have been placed in the DBG\_RECOV power mode.



Optionally, keeping the INPUTDOMAINPAIVE[10] or OUTPUTDOMAINPAIVE[10] driven HIGH, after the CLUSTERPPUHWSTAT[10] indicates HIGH, holds the cores and the cluster in Cold reset.

6. When required to exit from WARM\_RST power mode, drive either the INPUTDOMAINPAIVE[10] signal or OUTPUTDOMAINPAIVE[10] signal LOW depending on what interface you are using.

Once the reset is applied, the hardware automatically places the cluster and cores to the ON power mode. If the PPUs are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPUs is required.

## 5.11 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-120AE DynamIQ™ cluster to ensure that the correct operation is maintained.

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

**Table 5-9: 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, cluster FULL_RET 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-120AE DynamIQ™ cluster PPU modes.

**Table 5-10: 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 <i>Accelerator Coherency Port</i> (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 whilst 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



Note

For information on power mode dependencies between cores in a dual-core complex, see [4.9.2 Power mode transition dependencies for a dual-core complex](#) on page 100.

## 5.12 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 for OFF or MEM\_RET power modes.

It is possible for *Error Correcting Code* (ECC) errors to occur in the RAMs during a power transition to OFF or MEM\_RET power modes. For example, this could happen during the software sequence shortly before the hardware sequence starts. Another example of where errors could occur is during the powerdown sequence when the L3 cache is cleaned and invalidated. Although these errors are reported in the RAS error record registers, once the cluster or core is powered down the RAS registers are no longer accessible.

If the RAS registers are reporting an error, the following sequence happens:

1. The RAS interrupt signals for the appropriate core or cluster are asserted. The RAS interrupt signals are n<type>ERRIRQ, n<type>FAULTIRQ, and n<type>CTITIRQ, where type can be CORE, CLUSTER, or COMPLEX. For example, nCLUSTERFAULTIRQ, nCOREFAULTIRQ[CN:0], and nCOMPLEXFAULTIRQ[CX:0].
2. If the *Power Policy Unit* (PPU) is currently transitioning to an OFF or MEM\_RET power modes, then these requests to the OFF or MEM\_RET power modes are denied.

3. If the error is detected in a core RAM, then the core wakes up from the powerdown `WFI` instruction.
4. If the error is detected in the shared L2 cache of a complex after the last core in that complex has completed its powerdown sequence, then that core will wake up and start executing code from the reset vector.

The error record registers must be read and cleared before the power domain will accept the power domain request from the PPU. This can be done by either using software running on the core, or accesses through the utility bus.

For more information about numbering conventions, see [1.8 Core, complex, and processing element numbering](#) on page 47

## 5.13 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.



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.

---

## 5.14 Minimum mode and dynamic mode restrictions

When the Power Policy Unit (PPU) is programmed in dynamic mode, Arm recommends that the minimum power mode (`PWR_POLICY`) and minimum operating mode (`OP_POLICY`) are `0x0`. This is because they can prevent power transitions in a way that is hard to predict. In dynamic mode, the PPU determines the target power mode from the maximum of `PWR_POLICY` and the dynamically hinted power mode.

The target operating mode is the maximum of `OP_POLICY` and the dynamically hinted operating mode. The dynamically hinted values always indicate a power state that can be reached directly from the current power state. However, when the `PWR_POLICY` and `OP_POLICY` are nonzero, the target power state might not be reachable from the current power state.

If this is the case, the PPU might request an unsupported power transition and the request will be denied. As a result, the power state will not change.

## 6. L3 cache

All the cores and complexes in the DSU-120AE DynamIQ™ cluster share the L3 cache.

The shared L3 cache of the DSU-120AE (applies to non-Direct 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<sup>1</sup> 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-120AE automatically performs cache cleaning, eliminating the need for software-controlled cache cleaning.

---

### 6.1 L3 cache allocation policy

The DSU-120AE L3 cache only caches Cacheable, shareable memory locations. Non-shareable memory locations do not allocate into 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 not accessible to ACP and will not be allocated to the L3 cache.

The DSU-120AE 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.

---

<sup>1</sup> 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 and is allocated to the core 1 cache, but not the L3 cache.

### Related information

[6. L3 cache](#) on page 136

## 6.2 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, 16MB, or 32MB each cache slice has 16 ways.

When selecting a non-power-of-two L3 cache size of 1536KB, 3MB, 6MB, 12MB, or 24MB each cache slice only has 12 ways.

### Related information

[6.8 Cache slices and power portions](#) on page 144

[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

## 6.3 Memory System Resource Partitioning and Monitoring control

The DSU-120AE uses the *Memory system resource Partitioning And Monitoring* (MPAM) architecture extension to control L3 cache partitioning and bandwidth partitioning.

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-120AE only uses MPAM to partition the L3 cache capacity and bandwidth. For more details about this architecture extension, see [Arm® Memory System Resource Partitioning and Monitoring \(MPAM\) System Component Specification](#).

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. Therefore, 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-120AE uses this MPAM ID structure as follows:

**MPAM\_NS field, 1 bit**

This field 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.

**PARTID field, 6 bits**

This field 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 field, depending indicates if the 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.

**PMG field, 1 bit**

This field 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-120AE does not use.

## 6.4 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-120AE* (DSU-120AE) 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-120AE uses the *Memory System Resource Partitioning and Monitoring* (MPAM) architecture extension to control the partitioning of the L3 cache. For more information on the MPAM controls used and the structure of the MPAM partition ID (MPAM ID) for the DSU-120AE, see [6.3 Memory System Resource Partitioning and Monitoring control](#) on page 137.

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



Note

Storing the MPAM ID value in the L3 cache is typically only required if there is a downstream cache, 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-120AE 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 [4.4.1 L3 cache RAM powerdown](#) on page 78) 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

[4.4.1 L3 cache RAM powerdown](#) on page 78

[6. L3 cache](#) on page 136

## 6.5 Bandwidth partitioning

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) provides an optional mechanism to share bandwidth differently between different sources, based on the *Memory System Resource Partitioning and Monitoring* (MPAM) bandwidth partitioning. This is controlled using MPAM.

By default, the bandwidth available within the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) should be distributed approximately fairly between all cores making requests. However, there might be circumstances when more control is required. For example, in a dual core cluster with two *Accelerator Coherency Port* (ACP) interfaces, each core and each ACP interface would get one quarter of the bandwidth. But, allowing both ACP interface, collectively to use up half of the overall bandwidth might impact on the performance of the cores. Therefore, the ACP could be restricted to using only a smaller proportion of the overall bandwidth.

A memory-bandwidth proportional-stride partitioning scheme is used, see [Arm® Memory System Resource Partitioning and Monitoring \(MPAM\) System Component Specification](#).

Bandwidth partitioning allows you to control how bandwidth is split when the demand for bandwidth is greater than the bandwidth available.

Each MPAM PARTID has a separate MPAMCFG\_MBW\_PROP register, which contains an enable bit and the STRIDEM1 field. If the bandwidth partitioning is enabled for that MPAM PARTID, the 6-bit STRIDEM1 value controls how much bandwidth to give to ACP transactions and cores that are using that PARTID.

The STRIDEM1 value is the reciprocal of the relative bandwidth required, minus one. For example, if three PARTIDs are all contending for bandwidth and you want to assign bandwidths in the ratio 100:125:1000, you could program STRIDEM1 values of 9, 7, and 0, respectively. This is because  $1/(9+1) : 1/(7+1) : 1/(0+1)$  gives the required ratio. As the numbers are relative, other values can also be used to give the same bandwidth ratio, such as 19, 15, and 1.

The bandwidth partitioning mechanism is work-conserving, which means that enabling it does not reduce the total bandwidth that the cluster uses. The scheme only regulates PARTIDs that are using more than their fair share of bandwidth. Therefore, if a PARTID is not attempting to use much bandwidth then this does not reduce the ability of other PARTIDs to use that bandwidth.



Because of the following two reasons, the ratio of the bandwidth for certain PARTIDs might not be in the programmed ratio:

- A PARTID that is already getting all the bandwidth that it wants does not gain more bandwidth with a lower STRIDEM1 value.
- If there is spare bandwidth, the bandwidth partitioning does not regulate the bandwidth of any PARTIDs.

The STRIDEM1 value also affects the transaction latency in a congested system. This is because if a process has been given a small share of the bandwidth and it is attempting to use more bandwidth than it is allowed, its memory requests will have to wait to be arbitrated. You can give processes that are low bandwidth but high priority a very low STRIDEM1 value so that they have the lowest possible latency. As the scheme is work-conserving, the large bandwidth available to the process is not wasted if the process does not use it.

You can use a single PARTID for a software process that spans multiple cores or generates ACP transactions. The bandwidth mechanism considers the total bandwidth from all sources when regulating the bandwidth of a PARTID.

Where possible, software should avoid either:

- Using a mixture of PARTIDs with very different STRIDEM1 values on two cores in the same complex.
- Using a mixture of PARTIDs with very different STRIDEM1 values on an ACP interface.

When programmed like this, in certain situations the bandwidth that is achieved is a compromise. Therefore, some partitions might get more bandwidth than expected and others might get less bandwidth than expected.

For the best functioning of the mechanism, if a CHI system interconnect is not able to accept new transactions from the DSU-120AE, the interconnect should stop returning link-layer credits on the CHI REQ channel. The DSU-120AE will pick the most important transaction to send next. The interconnect should avoid generating large numbers of RetryAck responses in this situation because that reduces the ability of the DSU-120AE to control the order transactions are processed.

The cluster MPAM registers are used to configure the bandwidth QoS, see [B.1.2 External MPAM registers summary](#) on page 426.

## 6.6 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 *Accelerator Coherency Port* (ACP) interface or the CHI requester 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.



- 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.

The *Performance Monitoring Unit* (PMU) events, in particular those events from `0x0500` to `0x0524`, indicates to software how successful the stashing has been. This includes information on how many stash requests were received and how many of the received requests were dropped. For information on PMU events, see [16.2 PMU events](#) on page 250.

### Related information

[6. L3 cache](#) on page 136

[16.2 PMU events](#) on page 250

## 6.7 L3 cache data RAM latency

The DSU-120AE 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



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.

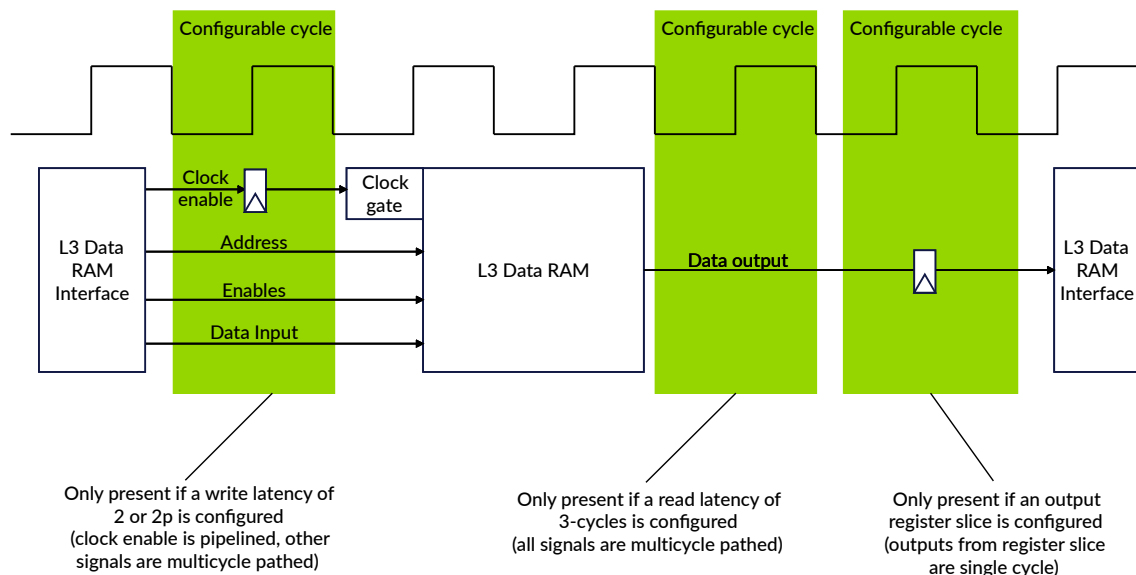
- An optional register slice on the output of the L3 data RAMs.

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 figure shows the L3 data RAM timing.

**Figure 6-1: 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.



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 6-1: 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

[1.2 DynamIQ Shared Unit-120AE configuration options](#) on page 18

[6. L3 cache](#) on page 136

## 6.8 Cache slices and power portions

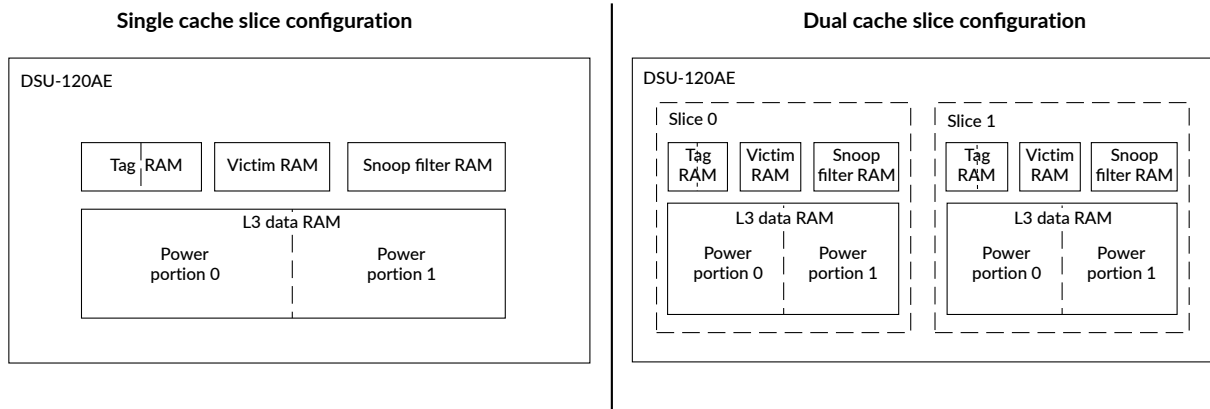
The L3 cache of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) can be divided up into identical slices, up to a limit of eight slices, each containing between 256KB and 4MB of the cache. A cache slice consists of the data, tag, victim, and snoop filter RAMs and associated logic. A power portion is a further subdivision of RAM in a cache slice.

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

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



**Figure 6-2: 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

[6.2 Available number of cache ways](#) on page 137

[6.8.1 Cache slice and requester port selection](#) on page 145

[4.4.2 L3 cache slice powerdown](#) on page 83

### 6.8.1 Cache slice and requester 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 requester port is configurable and can be independent from the slice mapping.

## 7. CHI requester 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-120AE* (DSU-120AE) to have either one, two, three, or four bus requester interface ports that use the AMBA 5 CHI Issue E protocol.



All L3 memory system variants for the DSU-120AE support the CHI Issue E protocol.

### 7.1 Multiple CHI bus requester port configurations

You can configure the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) to have one, two, three, or four CHI bus requester ports, at build time configuration, to give a range of bandwidth options. Transactions from the cores are routed to one of the CHI bus requester ports based on the transaction type, memory type, and transaction address.

The DSU-120AE also supports a configurable address target group methodology for the CHI bus requester ports. The address target groups are used to optimize the interconnect connectivity between the bus requester ports and the system.

Transactions are grouped into designated address target groups based on the target address, the memory type, and a set of configuration signals. Assigning a particular transaction to a group depends on the memory type targeted, for example Device transactions might be assigned to address target group 0. The address target groups are then assigned to different physical bus requester ports based on a pre-defined mapping. At reset time, any of the bus requester ports can optionally be disabled using a configuration signal which then alters the mapping between the address target groups and the remaining bus requester ports accordingly. Once the address target groups are mapped to bus requester ports, the address target groups are managed through the bus requester ports.

### 7.2 Configure CHI bus requester ports to use address target groups

Configuring requester ports to use address target groups involves a three-step process. After configuring the number of bus requester ports required, the hashing for the address target groups

must be defined. Finally, you must set the REQUESTERDISABLE signal to define the mapping between the address target groups and the bus requester ports.

## Procedure

1. Configure the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) for the number of bus requester ports required.  
Use the build time configuration parameter, `NUM_REQUESTERS` to specify the number of bus requester ports. See *Configuring the RTL* chapter in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual* on how to configure the RTL for the DSU-120AE.
2. Set up the address hashing for the number of the address target groups required. For hashing algorithms, see [7.2.1 Hashing for CHI transaction distribution](#) on page 147.  
The number of address target groups defined depends on the number of bus requester ports that have been configured at build time, as shown in the following table.

**Table 7-1: Combinations of requesters and address target groups supported**

Number of bus requester ports configured at build time	Number of address target groups
1	2
2	4
3	6
4	8

3. Set the REQUESTERDISABLE signal, to define the mapping between the address target groups to the bus requester ports. For the table of address target group mappings, see [7.2.2 Mapping for address target groups to CHI bus requester ports](#) on page 149.  
Once the mapping has been set up, the DSU-120AE automatically sets the id in the transaction address based on the allocation of the address target group numbers, see [7.2.3 CHI id bit setting](#) on page 150.

## 7.2.1 Hashing for CHI transaction distribution

When more than one bus requester port is implemented, the hashing to decide which transaction goes to which address target group is based on the *Physical Address* (PA) of the transaction, and the number of requester ports configured. There is a 1-bit, 2-bit, or 3-bit value that is used to identify the address target group number for each transaction depending on the number of bus requester ports configured. This gives a maximum of eight groups.

### Hashing for two, four, or eight address target groups

The hash function determines which address target group the PA of the transaction is sent to. The hash masks the transaction PA with a configurable mask, and then XORs all the resultant bits together. The configurable mask is set using the REQUESTERINTERLEAVE\* signals before the cluster leaves reset. In the following functions:

- REQUESTERINTERLEAVE0 is the configurable mask value set by REQUESTERINTERLEAVE0 input signal.
- REQUESTERINTERLEAVE1 is the configurable mask value set by REQUESTERINTERLEAVE1 input signal.

- REQUESTERINTERLEAVE2 is the configurable mask value set by REQUESTERINTERLEAVE2 input signal.
- ADDRESS is the PA of the transaction.

### Hashing for two address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE0[39:6])
```

### Hashing for four address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE0[39:6])
ADDRESS TARGET GROUP bit[1] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE1[39:6])
```

### Hashing for eight address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE0[39:6])
ADDRESS TARGET GROUP bit[1] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE1[39:6])
ADDRESS TARGET GROUP bit[2] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE2[39:6])
```

### Hashing for six address target groups

In the following function:

- ADDRESS is the PA of the transaction.
- REQUESTERADDRBITSELBOTTOM is the value set by REQUESTERADDRBITSELBOTTOM input signal.
- REQUESTERADDRBITSELTOP0 is the value set by REQUESTERADDRBITSELTOP0 input signal.
- REQUESTERADDRBITSELTOP1 is the value set by REQUESTERADDRBITSELTOP1 input signal.
- REQUESTERADDRBITSELTOP2 is the value set by REQUESTERADDRBITSELTOP2 input signal.
- REQUESTERTOPADDRBITINV is the value set by REQUESTERTOPADDRBITINV input signal.

The hash is:

```
ADDRESS TARGET GROUP[2:0] =
  (ADDRESS[REQUESTERADDRBITSELBOTTOM[3:0] +: 3]
  + ADDRESS[REQUESTERADDRBITSELBOTTOM[3:0]+3 +: 3]
  + ADDRESS[REQUESTERADDRBITSELBOTTOM[3:0]+6 +: 3]
  + (((REQUESTERTOPADDRBITINV ^ ADDRESS[REQUESTERADDRBITSELTOP2[5:0]]) << 2)
  | (ADDRESS[REQUESTERADDRBITSELTOP1[5:0]] << 1)
  | ADDRESS[REQUESTERADDRBITSELTOP0[5:0]])) % 6
```

The maximum value that can be output from this function is 0b101 (six groups).



For information on the functionality of the signals used in the hash functions, see the *CHI clock and configuration signals* section in the *Functional integration* chapter of the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

## 7.2.2 Mapping for address target groups to CHI bus requester ports

The mapping between the address target groups and the bus requester ports is determined by which bus requester ports are disabled at reset time. This is done by setting the signal REQUESTERDISABLE[*CMP*-1:0], where *CMP* is the number of bus requester ports configured.

The following table shows the mapping between the address target groups (groups) and the bus requester ports, where *MP* is the bus requester port number.

**Table 7-2: Mapping between address target groups and bus requester ports**

Number of bus requester ports ( <i>CMP</i> )	REQUESTERDISABLE[ <i>CMP</i> -1:0]	Address target group mapping
1	-	All traffic to MP 0
2	0b00	Traffic for groups 0,2 to MP 0 Traffic for groups 1,3 to MP 1
	0b10	All traffic to MP 0
	0b01	All traffic to MP 1
3	0b000	Traffic for groups 0,3 to MP 0 Traffic for groups 1,4 to MP 1 Traffic for groups 2,5 to MP 2
	0b110	All traffic to MP 0
	0b101	All traffic to MP 1
	0b011	All traffic to MP 2
4	0b0000	Traffic for groups 0,4 to MP 0 Traffic for groups 1,5 to MP 1 Traffic for groups 2,6 to MP 2 Traffic for groups 3,7 to MP 3
	0b1100	Traffic for groups 0, 2, 4, 6 to MP 0. Traffic for groups 1, 3, 5, 7 to MP 1.
	0b0011	Traffic for groups 0, 2, 4, 6 to MP 2. Traffic for groups 1, 3, 5, 7 to MP 3.
	0b1110	All traffic to MP 0
	0b1101	All traffic to MP 1

Number of bus requester ports (CMP)	REQUESTERDISABLE[CMP-1:0]	Address target group mapping
	0b1011	All traffic to MP 2
	0b0111	All traffic to MP 3

### 7.2.3 CHI id bit setting

The allocation of address target groups numbers is also used to set the id, by the *DynamIQ™ Shared Unit-120AE* (DSU-120AE), in the transaction address.

The following table shows how the address target id of the transaction, TgtID, is set depending on what address target group the transaction has been assigned.

**Table 7-3: Address target ID value dependency on address target groups**

Number of groups	TgtID = 0	TgtID = 1
2	Group 0	Group 1
4	Groups 0, 1	Groups 2, 3
6	Groups 0, 1, 2	Groups 3, 4, 5
8	Groups 0, 1, 2, 3	Groups 4, 5, 6, 7

## 7.3 CHI transaction routing with multiple requester ports

Transactions from the cores are routed, using the address target groups, to one of the CHI bus requester ports based on the transaction type, memory type, and transaction address.



**Note**

Address target group[0] has special functionality. For example, *Distributed Virtual Memory* (DVM) transactions always use address target group 0 unless the DEFAULTP configuration signal is set. Depending on your configuration signals, Device transactions are assigned to address target group 0. Typically, address target group 0 is mapped to bus interface port 0, but there might be special circumstances where bus interface port 0 is disabled. See [Table 7-2: Mapping between address target groups and bus requester ports](#) on page 149 for details.

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

**Table 7-4: CHI transaction routing**

Transaction type	Routed to
Cacheable transactions	Bus requester port number that is based on the address target group.
Normal Non-cacheable transactions	Bus requester port number that is based on the target address group.

Transaction type	Routed to
Device non-reorderable transactions	These are sent to either: <ul style="list-style-type: none"> <li>The bus requester port, which is assigned to address target group 0.</li> <li>All bus requester interfaces.</li> </ul>
Device reorderable transactions	Bus requester port number that is based on the target address group.
External snoop transactions	Snoop responses are routed to back to the same bus requester port that received the snoop.
DVM transactions	DVM transaction routing is controlled by the signal DEFAULTP: <ul style="list-style-type: none"> <li>Either sent to the bus requester port assigned to address target group 0; or</li> <li>Not used on this interface and these transactions are managed by the peripheral port.</li> </ul>



Note

- In the preceding table, you can find the bus requester port that corresponds to the target group given from the look-up table, see [Table 7-2: Mapping between address target groups and bus requester ports](#) on page 149.
- By default, transactions described in the preceding table are directed to one of the requester interface ports unless they match one of the peripheral port address ranges. However, if the DEFAULTP 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 requester 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 target group of the transaction. A configurable hash of the transaction address selects which requester interface port is used. See [7.2.1 Hashing for CHI transaction distribution](#) on page 147.

## Device non-reorderable transactions

Device non-reorderable transactions are either always routed to address target group 0 or are routed based on the calculated address target group and on the value of the DEVNRINTERLEAVE[1:0] input signal as follows:

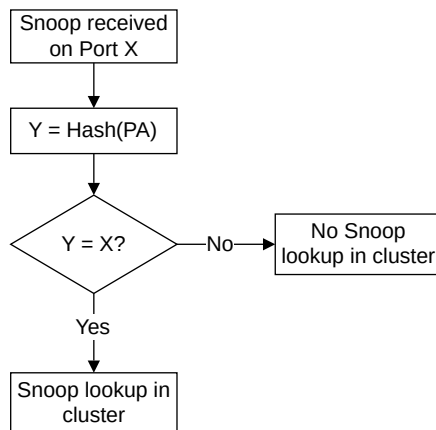
<b>0b00</b>	All Device non-reorderable transactions are sent to address target group 0.
<b>0b01</b>	Device non-reorderable transactions are sent to any requester interface port that is based on the same address interleaving as for non-Device transactions.
<b>0b10</b>	Reserved
<b>0b11</b>	There is no downstream convergence of traffic from different ports. This includes transactions sent on the same physical port using a different TgtID due to having a different address target group. This means that the system interconnect design must guarantee that

the transactions from different ports or from the same port but with different TgtID values are not routed to the same endpoint and 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 7-1: External snoop handling on CHI requester 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 requester 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 requester 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 requesters.

The second method is more efficient, and therefore if multiple requester 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 requester interface port.
- Calculate the correct requester interface port based on the snoop transaction address.

## DVM message transactions

The DSU-120AE only issues DVM messages on one port. When the DEFAULTP signal is asserted, all outgoing DVM messages are sent on the peripheral port. When the DEFAULTP signal is de-asserted, all outgoing DVM messages are sent on the port that is allocated to address target group 0.

The DSU-120AE only processes incoming DVM messages that are sent to the port used for outgoing DVM messages. In systems with multiple bus requester interface ports, DVM



messages sent to the other bus requester ports receive a response but have no effect inside the cluster.

For information about mappings for outgoing messages, see [Table 7-2: Mapping between address target groups and bus requester ports](#) on page 149.

## 7.4 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-120AE* (DSU-120AE) supports, or requires the interconnect and system to support.

**Table 7-5: CHI interconnect properties for the DSU-120AE**

CHI property	Supported by the DSU-120AE	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-120AE supports this feature if required by your interconnect.
Direct_Cache_Transfer	Yes	OPTIONAL. The DSU-120AE supports this feature if required by your interconnect.
Data_Poison	Yes	Yes
Data_Check	No	No
CCF_Wrap_Order	No. The DSU-120AE sends data packets in any order.	No
Barrier_Transactions	No	No. The DSU-120AE 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-120AE 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 requester interface values for the DSU-120AE.

**Table 7-6: CHI requester interface values for the DSU-120AE**

CHI property	Value	Comment
Req_Addr_Width	52	If the cluster only contains cores that have a <i>Physical Address</i> (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](#).



The DSU-120AE does not use the Streaming Ordered Writes feature of CHI. This means that for WriteNoSnp and WriteUnique transactions, the DSU never uses the combination of Order 0b10 and ExpCompAck HIGH.

## 7.5 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-120AE* (DSU-120AE), with a CHI bus.

**Table 7-7: 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 7-8: 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 7-9: 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.

## 7.6 Attributes of the CHI requester interface

The read and write issuing capabilities of the CHI requester interface depend on the configuration of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 requester port.

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

**Table 7-10: Attributes of the CHI requester memory interface**

Attribute	Value	Comment
Write issuing capability	Configuration dependent	This can range up to a maximum of 128 per requester port, depending on configuration.
Read issuing capability	Configuration dependent	This can range up to a maximum of 128 per requester 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. <b>Note:</b> The source of the transaction is encoded in the LPID field, see <a href="#">Table 7-12: CHI LPID[4:0] bitfields</a> on page 158.
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	-

Attribute	Value	Comment
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_LTBDS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. For multiple-requester configurations the percentage of total outstanding transactions each requester can support is as follows:
  - If there is only one requester port configured, then it can support the total number of outstanding transactions.
  - If there are two requester ports configured, then each port can support up to 50% of the total outstanding transactions.
  - If there are three requester ports configured, then each port can support up to 33% of the total outstanding transactions.
  - If there are four requester 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.
- The value assigned to the `TgtID` field, as set by the address target group function, does not affect the per-port total outstanding transaction capability.

## 7.7 CHI channel properties

The CHI requester interface supports snoops from your external memory system. The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE.

**Table 7-11: CHI channel properties**

Property	Value	Comment
Snoop acceptance capability	Configuration dependent	<p>The total snoop acceptance capability of the cluster is the value of the NUM_LTDBS configuration parameter multiplied by the NUM_L3_SLICES parameter.</p> <p>Each requester 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.</p>
DVM acceptance capability	Four per requester port	<p>The SCU can accept and process a maximum of four DVM transactions per requester port from the system. Each of these four transactions can be a two part DVM message.</p> <p>The interconnect must be configured to never send more than four DVM messages to a CHI requester interface port, otherwise the system might deadlock.</p>
Snoop latency	Hit and miss latencies depend on configuration	<p>Snoop latencies depend on how many requester 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.</p> <p>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.</p> <p>Latencies can be higher if hazards occur or if there are not enough buffers to absorb requests.</p>
	Miss	Dependent on build-time configuration
	DVM	Dependent on build-time configuration
Snoop filter	Supported	<p>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.</p> <p>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.</p> <p>Examples of case where evicts are not produced include:</p> <ul style="list-style-type: none"> <li>• Linefills that take External aborts.</li> <li>• Store exclusives that fail.</li> <li>• Mis-matched aliases.</li> </ul>
Supported transactions	-	The DSU-120AE supports all transaction types produced by the CHI protocol.

## 7.8 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:



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 7-12: CHI LPID[4:0] bitfields**

LPID[4:0] bit field	Accelerator Coherency Port (ACP) not implemented	One ACP port implemented	Two ACP ports implemented
[4]	Reserved	<p>The possible values are:</p> <p><b>0</b> If LPID[3:0] is 0xE</p> <p><b>Reserved</b> If LPID[3:0] is not 0xE</p>	<p>The possible values are:</p> <p><b>0</b> If LPID[3:0] is 0xE and ACP interface 0</p> <p><b>1</b> If LPID[3:0] is 0xE and ACP interface 1</p> <p><b>Reserved</b> If LPID[3:0] is not 0xE</p>
[3:0]	<p><b>0x0-0xD</b> Core instance number</p> <p><b>0xF</b> Cache copyback</p> <p><b>0xE</b> Accelerator Coherency Port (ACP) interface 0</p> <p><b>0xE</b> Accelerator Coherency Port (ACP) interface 1</p>		

The following table shows the CHI read and write transaction types supported by the CHI-configured requester port on the *DynamIQ™ Shared Unit-120AE* (DSU-120AE).

**Table 7-13: CHI read and write transactions supported by CHI-configured requester port**

Transaction	Operation	Produced by DSU-120AE
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 <i>Data Cache Clean to the Point of Persistence</i> (DC_CVAP) cache maintenance instruction generates this transaction when the BROADCASTPERSIST input signal is HIGH.	Yes
CleanUnique	Not used	No

Transaction	Operation	Produced by DSU-120AE
DVMOp	Branch predictor maintenance instructions, and <i>Translation Lookaside Buffer</i> (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 <i>Memory Tagging Extension</i> (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
WriteNoSnpPtlCMO	Not used	No

Transaction	Operation	Produced by DSU-120AE
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 <i>Accelerator Coherency Port</i> (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 requester interface.

**Table 7-14: 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				

The DSU-120AE never sends SnpRespDataPtl, NCBWrdDataCompAck, or WriteDataCancel packets.

## 7.9 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.

## 7.10 Support for memory types

The cores in the DSU-120AE 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 requester interface as Normal Non-cacheable.

## 8. AXI manager interface

You can configure the DSU-120AE to have an AMBA® AXI5 manager interface to your memory system, at a build-time configuration. This provides a non-coherent connection to your memory system. You can configure the DSU-120AE to have either one, two, three, or four AXI manager interface ports.

### 8.1 Multiple AXI bus manager port configurations

You can configure the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) to have one, two, three, or four AXI bus manager ports, at build time configuration, to give a range of bandwidth options. Transactions from the cores are routed to one of the AXI bus manager ports based on the transaction type, memory type, and transaction address.

The DSU-120AE also supports a configurable address target group methodology for the AXI bus manager ports. The address target groups are used to optimize the interconnect connectivity between the bus manager ports and the system.

Transactions are grouped into designated address target groups based on the target address, the memory type, and a set of configuration signals. In assigning a particular transaction to a group, the memory type targeted is taken into account, for example Device transactions might be assigned to address target group 0. The address target groups are then assigned to different physical bus manager ports based on a pre-defined mapping. At reset time, any of the bus manager ports can optionally be disabled using a configuration signal which then alters the mapping between the address target groups and the remaining bus manager ports accordingly. Once the address target groups are mapped to bus manager ports, the address target groups are managed through the bus manager ports.

### 8.2 Configure AXI bus manager ports to use address target groups

Configuring manager ports to use address target groups involves a three-step process. After configuring the number of bus manager ports required, the hashing for the address target groups must be defined. Finally, you must set the REQUESTERDISABLE signal to define the mapping between the address target groups and the bus manager ports.

#### Procedure

1. Configure the DynamIQ™ Shared Unit-120AE for the number of bus manager ports required. Use the build time configuration parameter, `NUM_REQUESTERS` to specify the number of bus manager ports. See *Configuring the RTL* chapter in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual* on how to configure the RTL for the DSU-120AE.
2. Set up the address hashing for the number of the address target groups required. See [8.2.1 Hashing for AXI transaction distribution](#) on page 163.

The number of address target groups defined depends on the number of bus manager ports that have been configured at build time, as shown in the following table.

**Table 8-1: Combinations of managers and address target groups supported**

Number of bus manager ports configured at build time	Number of address target groups
1	2
2	4
3	6
4	8

- Set the REQUESTERDISABLE signal, to define the mapping between the address target groups to the bus manager ports. For the table of address target group mappings, see [8.2.2 Mapping for address target groups to AXI bus manager ports](#) on page 164.  
Once the mapping has been set up, the DSU-120AE automatically sets the id in the transaction address based on the allocation of the address target group numbers, see [8.2.3 AXI id bit setting](#) on page 165.

## 8.2.1 Hashing for AXI transaction distribution

When more than one bus manager port is implemented, the hashing to decide which transaction goes to which address target group is based on the *Physical Address* (PA) of the transaction, and the number of manager ports configured. There is a 1-bit, 2-bit, or 3-bit value that is used to identify the address target group number for each transaction, depending on the number of bus manager ports configured. This gives a maximum of eight groups.

### Hashing for two, four, or eight address target groups

The hash function determines which address target group the PA of the transaction is sent to. The hash masks the transaction PA with a configurable mask, and then XORs all the resultant bits together. The configurable mask is set using the REQUESTERINTERLEAVE\* signals before the cluster leaves reset. In the following functions:

- REQUESTERINTERLEAVE0 is the configurable mask value set by REQUESTERINTERLEAVE0 input signal.
- REQUESTERINTERLEAVE1 is the configurable mask value set by REQUESTERINTERLEAVE1 input signal.
- REQUESTERINTERLEAVE2 is the configurable mask value set by REQUESTERINTERLEAVE2 input signal.
- ADDRESS is the PA of the transaction.

### Hashing for two address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE0[39:6])
```

## Hashing for four address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE0[39:6])
ADDRESS TARGET GROUP bit[1] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE1[39:6])
```

## Hashing for eight address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE0[39:6])
ADDRESS TARGET GROUP bit[1] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE1[39:6])
ADDRESS TARGET GROUP bit[2] = ^(ADDRESS[39:6] & REQUESTERINTERLEAVE2[39:6])
```

## Hashing for six address target groups

In the following function:

- ADDRESS is the PA of the transaction.
- REQUESTERADDRBITSELBOTTOM is the value set by REQUESTERADDRBITSELBOTTOM input signal.
- REQUESTERADDRBITSELTOP0 is the value set by REQUESTERADDRBITSELTOP0 input signal.
- REQUESTERADDRBITSELTOP1 is the value set by REQUESTERADDRBITSELTOP1 input signal.
- REQUESTERADDRBITSELTOP2 is the value set by REQUESTERADDRBITSELTOP2 input signal.
- REQUESTERTOPADDRBITINV is the value set by REQUESTERTOPADDRBITINV input signal.

The hash is:

```
ADDRESS TARGET GROUP[2:0] =
(ADDRESS[REQUESTERADDRBITSELBOTTOM[3:0] +: 3]
+ ADDRESS[REQUESTERADDRBITSELBOTTOM[3:0]+3 +: 3]
+ ADDRESS[REQUESTERADDRBITSELBOTTOM[3:0]+6 +: 3]
+ (((REQUESTERTOPADDRBITINV ^ ADDRESS[REQUESTERADDRBITSELTOP2[5:0]]) << 2)
| (ADDRESS[REQUESTERADDRBITSELTOP1[5:0]] << 1)
| ADDRESS[REQUESTERADDRBITSELTOP0[5:0]])) % 6
```

The maximum value that can be output from this function is 0b101 (six groups).



For information on the functionality of the signals used in the hash functions, see the *CHI clock and configuration signals* section in the *Functional integration* chapter of the Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual.

## 8.2.2 Mapping for address target groups to AXI bus manager ports

The mapping between the address target groups and the bus manager ports is determined by which bus manager ports are disabled at reset time. This is done by setting the signal REQUESTERDISABLE[CMP-1:0], where CMP is the number of bus manager ports configured.

The following table shows the mapping between the address target groups (groups) and the bus manager ports, where MP is the bus manager port number.

**Table 8-2: Mapping between address target groups and bus manager ports**

Number of bus manager ports (CMP)	REQUESTERDISABLE[CMP-1:0]	Address target group mapping
1	-	All traffic to MP 0
2	0b00	Traffic for groups 0,2 to MP 0 Traffic for groups 1,3 to MP 1
	0b10	All traffic to MP 0
	0b01	All traffic to MP 1
3	0b000	Traffic for groups 0,3 to MP 0 Traffic for groups 1,4 to MP 1 Traffic for groups 2,5 to MP 2
	0b110	All traffic to MP 0
	0b101	All traffic to MP 1
	0b011	All traffic to MP 2
4	0b0000	Traffic for groups 0,4 to MP 0 Traffic for groups 1,5 to MP 1 Traffic for groups 2,6 to MP 2 Traffic for groups 3,7 to MP 3
	0b1100	Traffic for groups 0, 2, 4, 6 to MP 0. Traffic for groups 1, 3, 5, 7 to MP 1.
	0b0011	Traffic for groups 0, 2, 4, 6 to MP 2. Traffic for groups 1, 3, 5, 7 to MP 3.
	0b1110	All traffic to MP 0
	0b1101	All traffic to MP 1
	0b1011	All traffic to MP 2
	0b0111	All traffic to MP 3

### 8.2.3 AXI id bit setting

The allocation of address target groups numbers is also used to set the id bit, by the *DynamIQ™ Shared Unit-120AE* (DSU-120AE), in the transaction address.

The following table shows how the address target id bit of the transaction, bit[0] of the AXI read and write address IDs (TgtID[0]), is set depending on what address target group the transaction has been assigned.

**Table 8-3: Address target ID value dependency on address target groups**

Number of groups	TgtID[0] = 0	TgtID[0] = 1
2	Group 0	Group 1
4	Groups 0, 1	Groups 2, 3
6	Groups 0, 1, 2	Groups 3, 4, 5
8	Groups 0, 1, 2, 3	Groups 4, 5, 6, 7

## 8.3 AXI transaction routing with multiple manager ports

Transactions from the cores are routed, using the address target groups, to one of the CHI bus manager ports based on the transaction type, memory type, and transaction address.



Note

Address target group[0] has special functionality. Depending on your configuration signals, Device transactions are assigned to address target group 0. Typically, address target group 0 is mapped to bus interface port 0, but there might be special circumstances where bus interface port 0 is disabled. See [Table 8-2: Mapping between address target groups and bus manager ports](#) on page 165 for details.

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

**Table 8-4: CHI transaction routing**

Transaction type	Routed to
Cacheable transactions	Bus manager port number that is based on the address target group.
Normal Non-cacheable transactions	Bus manager port number that is based on the target address group.
Device non-reorderable transactions	These transactions are sent to either: <ul style="list-style-type: none"> <li>The bus manager port which is assigned to address target group 0.</li> <li>All bus manager interfaces.</li> </ul>
Device reorderable transactions	Bus manager port number that is based on the target address group.



Note

- In the preceding table, you can find the bus manager port that corresponds to the target group given from the look-up table, see [Table 7-2: Mapping between address target groups and bus requester ports](#) on page 149.

- By default, transactions described in the preceding table are directed to one of the manager interface ports unless they match one of the peripheral port address ranges. However, if the DEFAULTP signal is asserted at reset, then the mapping is inverted. Therefore all transactions, 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 manager 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 target group of the transaction. A configurable hash of the transaction address selects which manager interface port is used. See [8.2.1 Hashing for AXI transaction distribution](#) on page 163.

### Device non-reorderable transactions

Device non-reorderable transactions are either always routed to address target group 0 or are routed based on the calculated address target group and on the value of the DEVNRINTERLEAVE[1:0] input signal as follows:

<b>0b00</b>	All Device non-reorderable transactions are sent to address target group 0.
<b>0b01</b>	Device non-reorderable transactions are sent to any manager interface port that is based on the same address interleaving as for non-Device transactions.
<b>0b10</b>	Reserved
<b>0b11</b>	There is no downstream convergence of traffic from different ports. This includes transactions sent on the same physical port using a different TgtID due to having a different address target group. This means that the system interconnect design must guarantee that the transactions from different ports or from the same port but with different TgtID values are not routed to the same endpoint and therefore the ReadReceipt or Data Buffer ID (DBID) is enough to guarantee global ordering.

## 8.4 AXI manager port interface properties

AMBA defines a set of interface properties for the AXI interconnect. The AXI manager port of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) only supports some of these interface properties.

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

**Table 8-5: AXI interconnect properties for the DSU-120AE**

AXI property	Supported by the DSU-120AE	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
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-120AE	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



## 8.5 AXI configurations

The AXI manager interface of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) by default supports the AXI5 protocol but you can configure it to support the AXI4 protocol.

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

## 8.6 AXI 256-bit manager interface attributes

The read and write issuing capabilities of the AXI manager interface depend on the configuration of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 manager interface attributes.

**Table 8-6: AXI 256-bit manager 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 68 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_LTBDS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. If there is only one manager port configured, then it can support the total number of outstanding transactions. For multiple-manager configurations the percentage of total outstanding transactions each manager can support is as follows:

- If there is only one manager port configured, then it can support the total number of outstanding transactions.
- If there are two manager ports configured, then each port can support up to 50% of the total outstanding transactions.

- If there are three manager ports configured, then each port can support up to 33% of the total outstanding transactions.
- If there are four manager 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.

For more information about the AXI signals described in this manual, see the [AMBA® AXI Protocol Specification](#).

## 8.7 AXI transactions

The AXI manager interface of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 8-7: 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-120AE does not generate any FIXED bursts and a burst does not cross a cache line boundary.

The DSU-120AE generates only a subset of all possible AXI transactions on the manager interface.

The following transaction types are supported:

- INCR 2 256-bit read transfers
- INCR 2 256-bit write transfers
- WRAP 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.

- WRAP 1 16-bit, 32-bit, 64-bit, and 128-bit, and 256-bit atomic compare 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:

- 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.

## 8.8 Support for memory types

The cores in the DSU-120AE 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 requester interface as Normal Non-cacheable.

## 8.9 Read response

The AXI manager 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.

## 8.10 Write response

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



For interoperability reasons, Arm recommends that system components fully comply with the AXI specification and do not rely on the DSU-120AE behavior described here.

---

## 8.11 Barriers

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 managers in the system. Arm expects most peripherals to meet this requirement.

## 8.12 AXI privilege information

AXI provides information about the privilege level of accesses on the ARPROTM<p>[0] and AWPROTM<p>[0] signals, where <p> is the manager port interface number. 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.

## 9. ACP subordinate interface

The *Accelerator Coherency Port* (ACP) is an optional subordinate interface that provides coherent transaction support between the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) and external accelerators such as a *Direct Memory Access* (DMA) engine. Up to two ACP interfaces can be configured during build time configuration, with each ACP interface being implemented as either a 128-bit or 256-bit port.

The ACP subordinate interface allows an external manager to access memory through the main memory interface of the DSU-120AE. 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-120AE 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.
- For information on the configuring the number of ACP interfaces, the ACP port width, and the placement of ACP interfaces in the DSU-120AE, see the *RTL configuration process* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

### 9.1 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 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-120AE* (DSU-120AE).

**Table 9-1: ACP interface properties for the DSU-120AE**

ACP property	Supported by the DSU-120AE
Port_Type	Accelerator
Continuous_Cache_Line_Read_Data	Yes
Multi_Copy_Atomicity	Yes. System support is required.
Ordered_Write_Observation	No
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
MPAM_support	Yes

## Related information

[9.2 ACP ACE5-LiteDVM protocol subset](#) on page 174

[9.3 ACP transactions](#) on page 175

## 9.2 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.



For information on how to connect the ACP interface to your system, see *Functional Integration* in the *Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual*.

---

### Related information

[9.1 ACP features](#) on page 173

[9.3 ACP transactions](#) on page 175

[9.4 ACP performance](#) on page 179

## 9.3 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 9-2: ACP supported transaction types**

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



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 9-3: 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)



Read request type	ARSIZE	ARLEN	ARBURST	Address alignment	
				INCR	WRAP
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 9-4: 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 9-5: 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).

Write request type	AWSIZE	AWLEN	AWBURST	Address alignment		Comment
				INCR	WRAP	
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 9-6: 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] = 0b000000)	If AWSNOOP is WriteUniquePtl, then any combination of bytes is valid. If AWSNOOP is WriteUniqueFull, then all bytes must be valid.
32-byte	0x5 (32-bytes)	0x0 (1-beat)	INCR or WRAP	32-byte boundary (AWADDR[4:0] = 0b000000)	32-byte boundary (AWADDR[4:0] = 0b000000)	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 [1.8 Core, complex, and processing element numbering](#) on page 47 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.

The DSU-120AE 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.



Note

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

## 9.4 ACP performance

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

### AXI ID guidelines

The ACP manager 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 manager 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 manager port instead.
- Stash requests that target a core (AWSTASHLPIDENS is HIGH) always attempt to allocate to the core L2 cache. The value of AWCACHES[3] does not affect the allocation.

### 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 9-7: ACP acceptance capabilities for each ACP port**

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 across the whole cluster is limited to the NUM_LTDBS parameter multiplied by the NUM_L3_SLICES parameter.

### Related information

[9.1 ACP features](#) on page 173

[9.2 ACP ACE5-LiteDVM protocol subset](#) on page 174

[9.3 ACP transactions](#) on page 175

## 9.5 DVM snoop transaction support

The *Accelerator Coherency Port* (ACP) of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) supports *Distributed Virtual Messages* (DVM) snoop transactions. DVM snoop transactions are only sent from the ACP port to the ACP manager.

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 subordinate 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 [9.5.1 Control the receiving of DVM snoop transactions](#) on page 181.



Note

- If your ACP manager 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.
- 

### 9.5.1 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 manager.

#### About this task

How to control the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 Protocol Specification](#).

---

#### Procedure

1. Instruct your ACP manager 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-120AE has acknowledged the request.  
When both the SYSCOREQS and SYSCOACKS signals are HIGH, the DSU-120AE 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 manager to deassert SYSCOREQS signal (LOW).
4. Wait for the signal SYSCOACKS to go LOW.  
When the signal SYSCOACKS has gone LOW, the DSU-120AE 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.

---

## 10. AXI or CHI requester 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 requester port to support accesses to the rest of the system whilst the main requester ports connect to main memory.

Using the peripheral port as alternative requester 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 requester interface
- A 256-bit AXI5 non-coherent requester interface
- A 256-bit CHI Issue E requester 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-120AE Configuration and Integration Manual* for more details on configuring the peripheral port.

### 10.1 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 instructions



Note

- Cacheable memory transactions are only coherent outside the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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.

## 10.2 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:

### **ASTARTOMP[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.

### **AENDOMP[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  $\text{ASTARTOMP[PA-1:20]} \leq \text{peripheral port address range} < \text{AENDOMP[PA-1:20]}$ .

The first address range is configurable to granularity of 1MB. The values for the ASTARTOMP[PA-1:20] and AENDOMP[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]
```



- 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 DEFAULTP signal is LOW, and both ASTARTOMP and AENDOMP are tied LOW then traffic is sent to the main requester 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 requester port. See [10.2.1 Changing peripheral port address range](#) on page 185 for more details.

By default, all incoming transaction addresses go to the main requester 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.

If *Distributed Virtual Memory* (DVM) operations are supported, these go to the requester port assigned to address target group 0. See [7.3 CHI transaction routing with multiple requester ports](#) on page 150 for more details.

However, if the signal DEFAULTP 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 requester ports.
- DVM operations are sent to the peripheral port if supported.



Note

To avoid system deadlocks, the peripheral port and main requester 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.

## 10.2.1 Changing peripheral port address range

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE* 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.



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.



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.

## 10.3 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-120AE* (DSU-120AE) 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 10-1: AXI 64-bit peripheral port interface properties**

AXI property	Supported by the DSU-120AE	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

AXI property	Supported by the DSU-120AE	Interconnect or system support required
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	Yes, the DSU-120AE 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

## 10.4 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-120AE* (DSU-120AE) 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 10-2: AXI 256-bit peripheral port interface properties**

AXI property	Supported by the DSU-120AE	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

AXI property	Supported by the DSU-120AE	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	Yes, the DSU-120AE 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

## 10.5 AXI 64-bit peripheral port transactions

The AXI 64-bit configured peripheral port of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 10-3: 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-120AE does not generate any FIXED bursts, and a burst does not cross a cache line boundary.

The DSU-120AE generates only a subset of all possible AXI transactions on the manager interface.

The following transaction types are supported:

- INCR N (N:2,4,8) 64-bit read transfers
- INCR N (N:2,4,8) 64-bit write transfers
- WRAP 8 64-bit read transfers
- WRAP N (N:2,4) 64-bit atomic compare write transfers.
- WRAP 1 16-bit, 32-bit, and 64-bit atomic compare write 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

The following points apply to AXI transactions:

- 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.

## 10.6 AXI 256-bit peripheral port transactions

The AXI 256-bit configured peripheral port of the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 10-4: 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	-

Transaction	Operation
AtomicSwap	-
AtomicCompare	-

The cache linefill fetch length is always 64 bytes. The DSU-120AE does not generate any FIXED bursts and a burst does not cross a cache line boundary.

The DSU-120AE generates only a subset of all possible AXI transactions on the manager interface.

The following transaction types are supported:

- INCR 2 256-bit read transfers
- INCR 2 256-bit write transfers
- WRAP 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.
- WRAP 1 16-bit, 32-bit, 64-bit, and 128-bit, and 256-bit atomic compare 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:

- 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.

## 10.7 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-120AE* (DSU-120AE) 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 10-5: 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. <b>Note:</b> The source of the transaction is encoded in the LPID field, see <a href="#">Table 7-12: CHI LPID[4:0] bitfields</a> on page 158.
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_LTBDS` 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.



**Note**

- 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.

## 10.8 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-120AE* (DSU-120AE) 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 10-6: CHI peripheral port interface properties**

CHI property	Supported by the DSU-120AE cluster	Interconnect support required
Atomic_Transactions	The DSU-120AE 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	Yes	Yes
Data_Check	No	No
CCF_Wrap_Order	No	No
Enhanced_Features	<p>The DSU-120AE cluster supports data return from SC state.</p> <p>The DSU-120AE cluster does not support input/output deallocation transactions, for example <i>ReadOnceMakeInvalid</i> (ROMI) and <i>ReadOnceCleanInvalid</i> (ROCI).</p> <p>The DSU-120AE cluster supports ReadNotSharedDirty transactions and requires interconnect support.</p> <p>If BROADCASTPERSIST is HIGH, the DSU-120AE cluster supports CleanSharedPersist transactions and requires interconnect support.</p>	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 10-7: 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



## 10.9 CHI peripheral port transactions

The CHI configured peripheral port of DynamIQ™ Shared Unit-120AE (DSU-120AE) supports the same CHI transactions as the CHI configured main requester interface.

The following table shows the read and write transactions supported by the CHI-configured peripheral port of the DSU-120AE.

**Table 10-8: CHI read and write transactions supported by DSU-120AE**

Transaction	Operation	Produced by DSU-120AE
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	<i>Translation Lookaside Buffer</i> (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 <i>Memory Tagging Extension</i> (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-120AE
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 <i>Accelerator Coherency Port</i> (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

## 10.10 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 and the number of *Accelerator Coherency Port* (ACP) interfaces 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 each ACP. Therefore there can be a maximum of two LPIDs if two ACP interfaces are configured.

Therefore, the maximum number of LPIDs for cluster of 14 cores is 17 LPIDs.

The following table describes the read and write issuing capabilities of the peripheral port when configured as AXI 64-bit mode.

**Table 10-9: AXI issuing capabilities**

Attribute	Value	Comments
Write issuing capability	Up to a maximum of 4 times total number of LPIDs	<p>For the cluster of 14 cores, the maximum issuing capability is 68 outstanding reads or writes.</p> <p><b>Note:</b> These values 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.</p>
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 following table lists the encoding for AXI transaction IDs for the Peripheral port when configured as AXI 64-bit mode.

**Table 10-10: AXI transaction ID encoding**

Attribute	Value	Comments
All IDs	0b0r_yyyy	<p>The value comprises the following fields:</p> <p><b>r</b> Can be 0 or 1 <b>yyyy</b> LPID number</p>



**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 [8.6 AXI 256-bit manager interface attributes](#) on page 169 for the read and write issuing capabilities for a peripheral port configured in 256-bit mode.

See the [AMBA® AXI Protocol Specification](#) for more information about the AXI signals described in this manual.

## 10.11 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 manager 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:

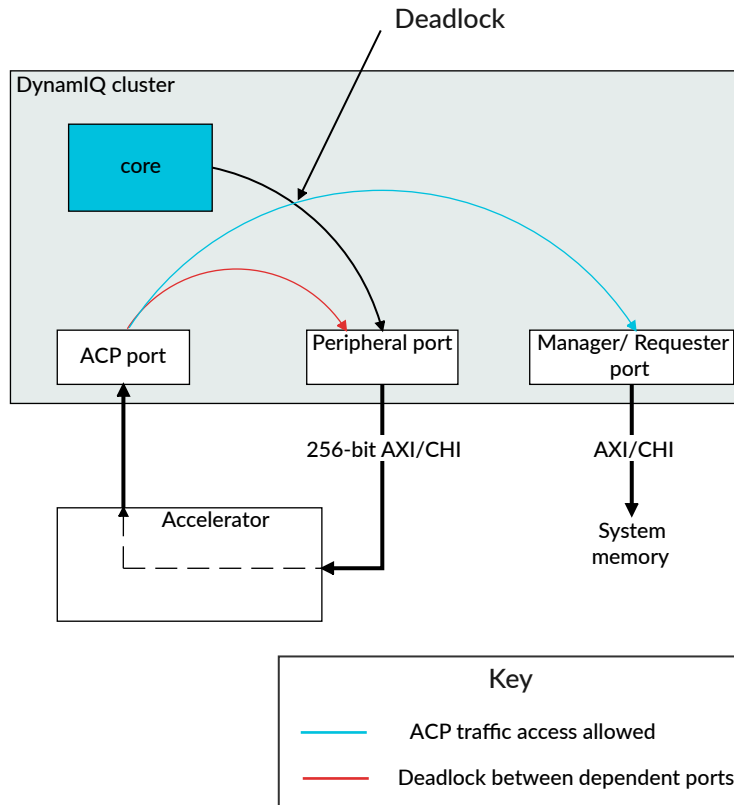
- For connecting to a tightly-coupled accelerator.
- For use as a more general system port.

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-120AE* (DSU-120AE) is connected to a tightly-coupled accelerator.

**Figure 10-1: Deadlock scenario when peripheral port is connected to an accelerator**



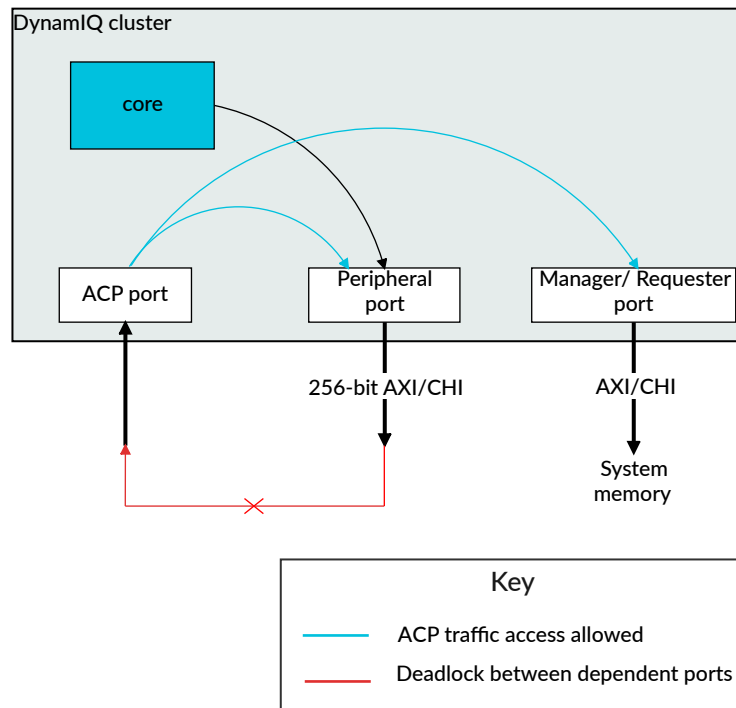
When the peripheral port is connected to a tightly-coupled accelerator, the accelerator might have an internal dependency, this is shown by the dashed line in the preceding figure. This means, a read or write to the registers of the accelerator through the peripheral port cannot complete until an outstanding 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 figure) 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 manager 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 figure. 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 10-2: 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 figure. 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.

## 10.12 AXI privilege information for the AXI-configured peripheral port

AXI provides information about the privilege level of accesses on the ARPROTMP0[0] and AWPRTMP0[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.

# 11. RAS extension support

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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-120AE 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-120AE. 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 *Reliability, Availability, and Serviceability (RAS) Extension* appendix in the [Arm® Architecture Reference Manual for A-profile architecture](#).

## 11.1 Cache protection behavior

The configuration of the *Reliability, Availability, Serviceability* (RAS) Extension that is implemented in the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) includes *Error Correcting Code* (ECC) cache protection. In this case, the DSU-120AE protects against errors that result in a RAM bitcell holding the incorrect value.

The RAMs in the DSU-120AE support *Single Error Correct Double Error Detect* (SECEDED). SECEDED 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-120AE can progress and remain functionally correct when there is a single bit error in any RAM.

**Table 11-1: RAM cache protection**

RAM	Protection	Description
L3 data cache data	ECC, SECEDED	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: <ul style="list-style-type: none"> <li>If the tag entry is 58 bits wide, there are 8 ECC bits.</li> <li>If the tag entry is 57 bits wide or less, there are 7 ECC bits.</li> </ul>
L3 cache victim	None	-
<i>Long-Term Data Buffer</i> (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-120AE remains functionally correct.

If there is a double bit error in a single RAM within the same protection granule, the DSU-120AE 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-120AE might or might not detect the errors. The cache protection feature of the DSU-120AE 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-120AE 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-120AE 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 Cluster Extended Control Register. See either [A.1.5 IMP\\_CLUSTERECTLR\\_EL1, Cluster Extended Control Register](#) on page 274 or [B.1.1.11 CLUSTERECTLR, Cluster Extended Control Register](#) on page 417, 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-120AE 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-120AE 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.

The DSU-120AE does not poison the data transaction of a DVM operation it initiates because the data payload of a DVM operation is not derived from cached data.

If the DSU-120AE receives a poisoned DVM operation from a core, it consumes the error and reports it as an uncorrectable error (`producer error`). If the DSU must broadcast the poisoned DVM operation, it sets `RespErr=DErr` and `Poison=0` so that the broadcast DVM operation is not marked as poisoned.

## 11.2 Error containment

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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.

## 11.3 Fault detection and reporting

When the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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 `ERROCTLR.FI` is set, all Deferred errors and Uncorrected errors that the DSU-120AE detects generate an FHI through the `nCLUSTERFAULTIRQ` signal.

When `ERROCTLR.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 `ERROCTLR.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 `ERROCTLR.CI` is set, all critical errors that the DSU-120AE 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 `ERROSTATUS` register.

## 11.4 Error detection and reporting

When the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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\_ERROFR. This is a read-only register that specifies various error record settings.
- The cluster Error Record Control Register, CLUSTERRAS\_ERROCTLR.
- The cluster Error Record Miscellaneous Register 0-3, CLUSTERRAS\_ERROMISCO-3. These registers record details of the error location and counts.
- The cluster Pseudo-fault Generation Feature register, CLUSTERRAS\_ERROPFGF. Read-only register.
- The cluster Error Record Primary Status Register, CLUSTERRAS\_ERROSTATUS.

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-120AE 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.



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\_ERROSTATUS register.

When a dirty cache line with an error on the data RAMs is evicted from the cluster, the write on the requester interface still takes place. However, if the error is uncorrectable then:

- If the DSU-120AE is configured with an AXI manager-port, the uncorrected data is written and the error is reported in the RAS registers.
- If the DSU-120AE is configured with a CHI requester-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-120AE is configured with a CHI requester-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 subordinate 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.

### 11.4.1 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

[16.2 PMU events](#) on page 250

### 11.4.2 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.

### 11.4.3 Double error reporting

If the DSU-120AE detects an *Error Correcting Code (ECC)* error in the L3 data RAM, the DSU-120AE 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-120AE detects an error in the L3 data RAM, the DSU-120AE 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-120AE 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 ERRORMISC0.CECO due to the reporting of a second Correctable Error. The contents of the ERROSTATUS accurately shows that the error came from the L3 Data RAM. For example, ERROSTATUS.SERR=6, ERROSTATUS.V=1 and ERROSTATUS.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, ERROSTATUS.V=1, ERROSTATUS.DE=1 and ERROSTATUS.SERR=6. However, the extra error from the LTDB RAM also sets ERROSTATUS.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, ERROSTATUS.V=1, ERROSTATUS.UE=1, ERROSTATUS.SERR=2. The extra error also means that ERROSTATUS.OF=1. Also even though L3 data RAM poisoned the data, ERROSTATUS.PN=0.

## 11.5 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-120AE* (DSU-120AE) 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 will trigger the pseudo fault. The pseudo fault can be deliberately triggered 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 LTDB 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 will trigger 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.



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.

---

## 11.6 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 in OFF or MEM\_RET power modes.

It is possible for *Error Correcting Code* (ECC) errors to occur in the RAMs during a power transition to OFF or MEM\_RET power modes. For example, this could happen during the software sequence shortly before the hardware sequence starts. Another example of where errors could occur is during the powerdown sequence when the L3 cache is cleaned and invalidated. Although these errors are reported in the RAS error record registers, once the cluster or core is powered down the RAS registers are no longer accessible.

If the RAS registers are reporting an error, the following sequence happens:

1. The RAS interrupt signals for the appropriate core or cluster are asserted. The RAS interrupt signals are n<type>ERRIRQ, n<type>FAULTIRQ, and n<type>CTITIRQ, where type can be CORE, CLUSTER, or COMPLEX. For example, nCLUSTERFAULTIRQ, nCOREFAULTIRQ[CN:0], and nCOMPLEXFAULTIRQ[CX:0].
2. If the *Power Policy Unit* (PPU) is currently transitioning to an OFF or MEM\_RET power modes, then these requests to the OFF or MEM\_RET power modes are denied.
3. If the error is detected in a core RAM, then the core wakes up from the powerdown `WFI` instruction.

4. If the error is detected in the shared L2 cache of a complex after the last core in that complex has completed its powerdown sequence, then that core will wake up and start executing code from the reset vector.

The error record registers must be read and cleared before the power domain will accept the power domain request from the PPU. This can be done by either using software running on the core, or accesses through the utility bus.

For more information about numbering conventions, see [1.8 Core, complex, and processing element numbering](#) on page 47

## 11.7 Core RAS registers

The cores support *Error Correcting Code* (ECC) and parity protection on their RAMs, which normally updates the core RAS registers, when a fault is detected. However, when running in Lock-Step, the core's RAMs are duplicated, therefore a fault in one set of the RAMs causes the fault to be reported in one copy of the core. If the core or complex RAS registers are read using memory mapped accesses on the utility bus, then the value read from the two cores or complexes are different, which causes a *Dual-Core Lock-Step* (DCLS) fault to be reported. To avoid this case, the DSU provides two address ranges for the RAS registers of each core or complex.

The two address ranges are as follows:

### First address range

- For the core: 0x<n>A0000
- For the complex: 0x<n>C0000

When performing a read from the first address, it reads the data from the primary core or from the primary complex. It does not report a fault if the redundant register content does not match.

Writing to the first address range updates the registers in both the primary and redundant core or complexes.

### Second address range

- For the core: 0x<n>F0000
- For the complex: 0x<n>E0000

When performing a read from the second address, it reads the data from the redundant core or from the redundant complex. It does not report a fault, if the primary register content does not match.

Writing to the second address range is ignored.

**Note**

Note that, there is a single set of fault reporting interrupt pin from the cores, which indicates if the primary or redundant core reported a fault in the RAS registers. Therefore, software responding to these interrupt pins must read from both the primary and redundant address ranges to determine the true location of the reported fault.

## 11.8 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 accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.

### 11.8.1 AArch64 RAS registers

The **IMPLEMENTATION DEFINED** cluster RAS registers are accessible either from System register accesses from the cores or from memory-mapped accesses on the utility bus.

The summary table provides an overview of the **IMPLEMENTATION DEFINED** AArch64 cluster RAS registers in the DSU-120AE. For more information about a register, click on the register name in the table.

**Note**

For registers without a listed reset value refer to the individual field resets documented on the register description pages.

For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table 11-2: RAS registers summary**

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
<a href="#">ERXFR_EL1</a>	3	0	C5	C4	0	See individual bit resets.	64-bit	Selected Error Record Feature Register
<a href="#">ERXCTLR_EL1</a>	3	0	C5	C4	1	See individual bit resets.	64-bit	Selected Error Record Control Register



Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
<a href="#">ERXSTATUS_EL1</a>	3	0	C5	C4	2	See individual bit resets.	64-bit	Selected Error Record Primary Status Register
<a href="#">ERXPFGF_EL1</a>	3	0	C5	C4	4	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Feature Register
<a href="#">ERXPFGCTL_EL1</a>	3	0	C5	C4	5	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Control Register
<a href="#">ERXPFGCDN_EL1</a>	3	0	C5	C4	6	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Countdown Register
<a href="#">ERXMISCO_EL1</a>	3	0	C5	C5	0	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 0
<a href="#">ERXMISC1_EL1</a>	3	0	C5	C5	1	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 1
<a href="#">ERXMISC2_EL1</a>	3	0	C5	C5	2	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 2
<a href="#">ERXMISC3_EL1</a>	3	0	C5	C5	3	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 3

## 11.8.2 External cluster RAS registers

The cluster RAS registers are accessible either from memory-mapped accesses on the utility bus or from System register accesses from the cores.

The summary table provides an overview of all the cluster RAS registers in DSU-120AE. For more information about a register, click on the register name in the table.



Note

- The cluster RAS registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is 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 11-3: CLUSTERRAS registers summary**

Offset	Name	Reset	Width	Description
0x000	<a href="#">CLUSTERRAS_ERROFR</a>	See individual bit resets.	64-bit	Error Record Feature Register
0x008	<a href="#">CLUSTERRAS_ERROCTL</a>	See individual bit resets.	64-bit	Error Record Control Register
0x010	<a href="#">CLUSTERRAS_ERROSTATUS</a>	See individual bit resets.	64-bit	Error Record Primary Status Register
0x018	<a href="#">CLUSTERRAS_ERROADDR</a>	See individual bit resets.	64-bit	Error Record Address Register
0x020	<a href="#">CLUSTERRAS_ERROMISCO</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 0
0x028	<a href="#">CLUSTERRAS_ERROMISC1</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 1
0x030	<a href="#">CLUSTERRAS_ERROMISC2</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 2

Offset	Name	Reset	Width	Description
0x038	CLUSTERRAS_ERRROMISC3	See individual bit resets.	64-bit	Error Record Miscellaneous Register 3
0x800	CLUSTERRAS_ERR0PFGF	See individual bit resets.	64-bit	Pseudo-fault Generation Feature Register
0x808	CLUSTERRAS_ERR0PFGCTL	See individual bit resets.	64-bit	Pseudo-fault Generation Control Register
0x810	CLUSTERRAS_ERR0PFGCDN	See individual bit resets.	64-bit	Pseudo-fault Generation Countdown Register
0xE00	CLUSTERRAS_ERRGSR	See individual bit resets.	64-bit	Error Group Status Register
0xE10	CLUSTERRAS_ERRIIDR	See individual bit resets.	32-bit	Implementation Identification Register
0xFA8	CLUSTERRAS_ERRDEVAFF	See individual bit resets.	64-bit	Device Affinity Register
0xFBC	CLUSTERRAS_ERRDEVARCH	See individual bit resets.	32-bit	Device Architecture Register
0xFC8	CLUSTERRAS_ERRDEVID	See individual bit resets.	32-bit	Device Configuration Register
0xFD0	CLUSTERRAS_ERRPIDR4	See individual bit resets.	32-bit	Peripheral Identification Register 4
0xFD4	CLUSTERRAS_ERRPIDR5	See individual bit resets.	32-bit	Peripheral Identification Register 5
0xFD8	CLUSTERRAS_ERRPIDR6	See individual bit resets.	32-bit	Peripheral Identification Register 6
0xFDC	CLUSTERRAS_ERRPIDR7	See individual bit resets.	32-bit	Peripheral Identification Register 7
0xFE0	CLUSTERRAS_ERRPIDR0	See individual bit resets.	32-bit	Peripheral Identification Register 0
0xFE4	CLUSTERRAS_ERRPIDR1	See individual bit resets.	32-bit	Peripheral Identification Register 1
0xFE8	CLUSTERRAS_ERRPIDR2	See individual bit resets.	32-bit	Peripheral Identification Register 2
0xFEC	CLUSTERRAS_ERRPIDR3	See individual bit resets.	32-bit	Peripheral Identification Register 3
0xFF0	CLUSTERRAS_ERRCIDR0	See individual bit resets.	32-bit	Component Identification Register 0
0xFF4	CLUSTERRAS_ERRCIDR1	See individual bit resets.	32-bit	Component Identification Register 1
0xFF8	CLUSTERRAS_ERRCIDR2	See individual bit resets.	32-bit	Component Identification Register 2
0xFFC	CLUSTERRAS_ERRCIDR3	See individual bit resets.	32-bit	Component Identification Register 3

## 12. Utility bus

The utility bus provides access to control registers for various system components in the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) and the cores within the DSU-120AE DynamIQ™ cluster. The utility bus is implemented as a 64-bit AMBA AXI5 subordinate 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 cores and cluster
- *Memory Partitioning and Monitoring* (MPAM) registers for the cluster
- *Activity Monitor Unit* (AMU) registers in the cores and cluster
- *Maximum Power Mitigation Mechanism* (MPMM) registers in the cores and cluster



Note

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).

---

### 12.1 Utility bus accesses

Transactions on the utility bus comply with a subset of the AXI5 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 state or Root state accesses on the utility bus, see [Table 12-3: Utility bus base addresses for system-accessible component registers](#) on page 214. Ensure that you access any system component register with the security 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 12-1: 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.

### 12.1.1 Core access to system component registers

Some of the system-accessible 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. Note that all the registers are accessible through the utility bus.

**Table 12-2: System component registers accessible from cores**

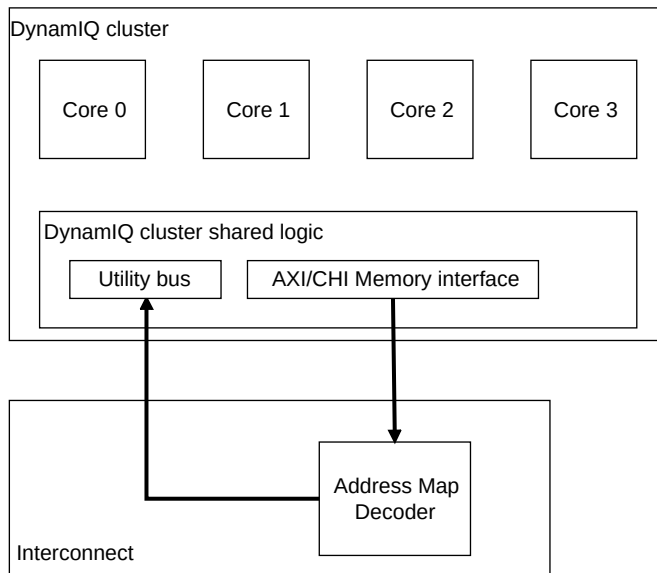
Registers	Directly accessible from cores
Cluster power control	Yes
Cluster <i>Memory System Resource Partitioning and Monitoring</i> (MPAM)	No
Cluster <i>Reliability, Availability, and Serviceability</i> (RAS)	Yes
Cluster <i>Power Policy Unit</i> (PPU)	No
Cluster <i>Activity Monitor Unit</i> (AMU)	No
Core PPU	No
Cluster-AE	No



For accessibility information on the core registers, other than PPU registers, see your core *Technical Reference Manual* (TRM).

The following figure shows an example of memory-mapped addressing for the cores to access the utility bus through the interconnect.

**Figure 12-1: Memory-mapped access from the cores to the utility bus**



### 12.1.2 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.

## 12.2 Base addresses for system-accessible components

Each set of System registers is grouped on separate 64 KB page boundaries allowing access to be enforced by a *Memory Management Unit* (MMU).

The following table shows the base addresses for each set of registers for system-accessible components and what Security state they should be accessed from.



Note

- The base address for each set of registers for the core Power Policy Units (PPUs) depends on the core instance number  $\langle n \rangle$ , 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  $0x\langle n \rangle 90000$  -  $0x\langle n \rangle F0000$ , 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 12-3: Utility bus base addresses for system-accessible component registers**

Base address, n is core instance number	Registers	Security state	Memory map
0x000000	Cluster control	Secure state	<a href="#">B.1.1 External cluster system control registers summary</a> on page 396
0x010000	Cluster MPAM	Any state	<a href="#">B.1.2 External MPAM registers summary</a> on page 426
0x020000	Cluster RAS	Secure state	<a href="#">B.1.3 External cluster RAS registers summary</a> on page 452
0x030000	Cluster PPU	Secure state	<a href="#">B.1.4 External cluster PPU registers summary</a> on page 503
0x040000	Activity Monitors	Secure state	<a href="#">B.1.5 External cluster AMU registers summary</a> on page 564
0x050000	Cluster AE registers	Secure state	<a href="#">B.1.6 External cluster AE registers summary</a> on page 600
0x060000 - 0x070000	Reserved for future cluster registers	-	-
$0x\langle n \rangle 80000$	Core $\langle n \rangle$ PPU	Secure state	<a href="#">B.1.7 External core PPU registers summary</a> on page 623
$0x\langle n \rangle 90000$ - $0x\langle n \rangle F0000$	Core $\langle n \rangle$ registers	See your core TRM	See your core TRM

## 13. System control registers

The system control registers control and provide status information for the functions that the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) implements. They can be accessed from the cores directly or externally through the utility bus.

### 13.1 AArch64 generic-system-control registers

The cluster Generic System Control registers are accessible either from System register accesses from the cores or from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the AArch64 Generic System Control registers in the DSU-120AE. For more information about a register, click on the register name in the table.



For registers with a listed reset value refer to the individual field resets documented on the register description pages.

**Table 13-1: Generic System Control registers summary**

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
<a href="#">IMP_CLUSTERCFR_EL1</a>	3	0	C15	C3	0	See individual bit resets.	64-bit	Cluster Configuration Register
<a href="#">IMP_CLUSTERIDR_EL1</a>	3	0	C15	C3	1	See individual bit resets.	64-bit	Cluster Main Revision Register
<a href="#">IMP_CLUSTERREVIDR_EL1</a>	3	0	C15	C3	2	See individual bit resets.	64-bit	Cluster ECO ID Register
<a href="#">IMP_CLUSTERACTLR_EL1</a>	3	0	C15	C3	3	See individual bit resets.	64-bit	Cluster Auxiliary Control Register
<a href="#">IMP_CLUSTERECTLR_EL1</a>	3	0	C15	C3	4	See individual bit resets.	64-bit	Cluster Extended Control Register
<a href="#">IMP_CLUSTERPWRCTLR_EL1</a>	3	0	C15	C3	5	See individual bit resets.	64-bit	Cluster Power Control Register
<a href="#">IMP_CLUSTERPWRDN_EL1</a>	3	0	C15	C3	6	See individual bit resets.	64-bit	Cluster Power Down Register
<a href="#">IMP_CLUSTERPWRSTAT_EL1</a>	3	0	C15	C3	7	See individual bit resets.	64-bit	Cluster Power Status Register
<a href="#">IMP_CLUSTERL3DNTH0_EL1</a>	3	0	C15	C4	0	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register
<a href="#">IMP_CLUSTERL3DNTH1_EL1</a>	3	0	C15	C4	1	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register
<a href="#">IMP_CLUSTERL3UPTH0_EL1</a>	3	0	C15	C4	2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
IMP_CLUSTERL3UPTH1_EL1	3	0	C15	C4	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register
IMP_CLUSTERBUSQOS_EL1	3	0	C15	C4	4	See individual bit resets.	64-bit	Cluster Bus QoS Control Register
IMP_CLUSTERL3HIT_EL1	3	0	C15	C4	5	See individual bit resets.	64-bit	Cluster L3 Hit Counter Register
IMP_CLUSTERL3MISS_EL1	3	0	C15	C4	6	See individual bit resets.	64-bit	Cluster L3 Miss Counter Register
IMP_CLUSTERPPSTART_EL1	3	0	C15	C9	0	See individual bit resets.	64-bit	Cluster Peripheral Port Start Address Register
IMP_CLUSTERPPEND_EL1	3	0	C15	C9	1	See individual bit resets.	64-bit	Cluster Peripheral Port End Address Register
IMP_CLUSTERRCFR2_EL1	3	0	C15	C9	2	See individual bit resets.	64-bit	Cluster Configuration Register 2
IMP_CLUSTERL3UPTH2_EL1	3	0	C15	C9	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register
IMP_CLUSTERCDBG_EL3	3	6	C15	C4	7	See individual bit resets.	64-bit	Cluster Cache Debug Register
IMP_CLUSTERPMMDCR_EL3	3	6	C15	C6	3	See individual bit resets.	64-bit	Monitor Debug Configuration Register (EL3)



## 14. Debug

The DSU-120AE 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-120AE. 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:

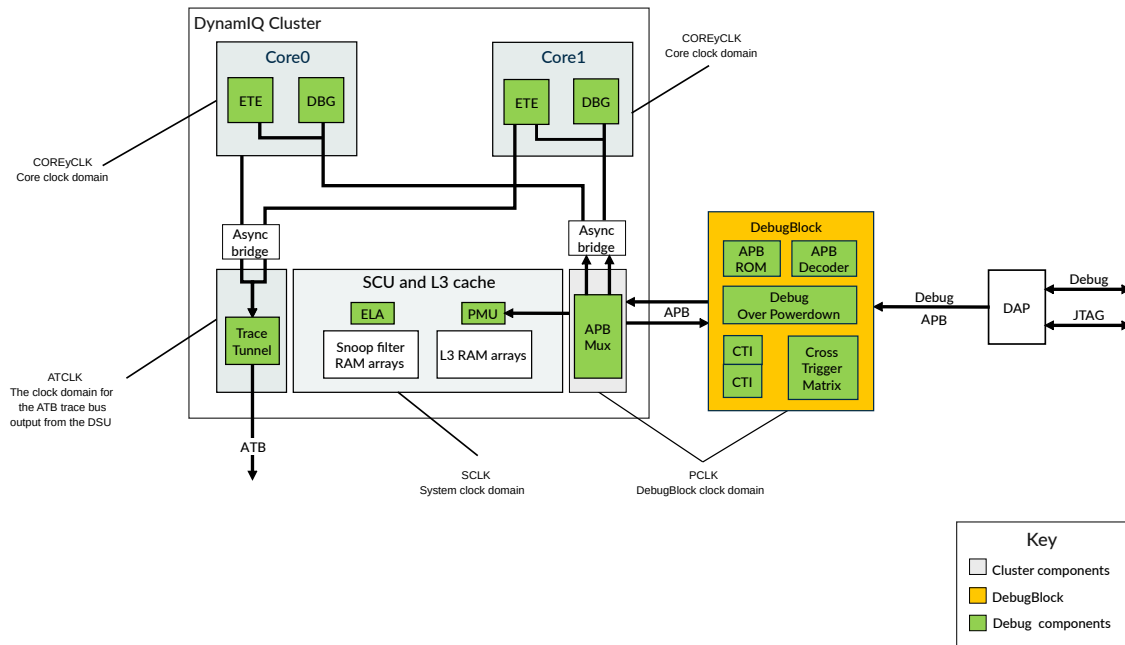
- The DebugBlock is placed in a separate power domain, to ensure that it is possible to maintain the connection to a debugger while the cores and cluster are powered down.
- The DebugBlock is physically placed with the other CoreSight logic in the SoC, rather than close to the cluster.

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-120AE implements the following CoreSight debug components:

- Per-core *Embedded Trace Extension* (ETE). Although the ETE is supplied with the core, the DSU-120AE 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 14-1: 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 unit in each core outputs trace, which is funneled in the cluster down to a single AMBA 5 ATB-C interface, which is 32 bits wide in small clusters and 64 bits wide in larger clusters.

## Cache debug

Cache debug of the DSU-120AE 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-120AE with the physical location to read, and the DSU-120AE returns the RAM contents at that location. The core then exposes this information in a system register.

## 14.1 Cache debug

Cache debug of the DSU-120AE 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-120AE cache debug information is provided through the DSU-120AE 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 bit fields for the physical location it wants to retrieve the data from. See the following table for bit field values.
2. The DSU-120AE returns the RAM contents in the CLUSTERCDBG register in an encoded form.
3. The core reads this information from the CLUSTERCDBG register.



Note

- 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 reads the cache contents when the cluster is in the ON power mode. If the cluster is in FUNC\_RET power mode or FULL\_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.FUNCRET and IMP\_CLUSTERPWRCTLR\_EL1.FULLRET values to zero.

The following table describes the bit fields for the CLUSTERCDBG register when writing to the register:

**Table 14-1: 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	<p>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.</p> <p>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 <a href="#">Table 14-3: Tag index width for L3 RAM accesses</a> on page 221. For details on slice ID widths see <a href="#">Table 14-2: Slice ID width</a> on page 220.</p> <p>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 <a href="#">Table 14-4: SF index width for SF RAM accesses</a> on page 221.</p>
[5:3]	CHUNK	<p>Select of 64-bit data chunk to read from 512-bit Data RAM cache line. Only used when accessing Data RAM data.</p> <p><b>0b000</b> Data[63:0]</p> <p><b>0b001</b> Data[127:64]</p> <p><b>0b010</b> Data[191:128]</p> <p><b>0b011</b> Data[255:192]</p> <p><b>0b100</b> Data[319:256]</p> <p><b>0b101</b> Data[383:320]</p> <p><b>0b110</b> Data[447:384]</p> <p><b>0b111</b> Data[511:448]</p>
[2:0]	RAM	<p>RAM to be accessed. All other values are reserved.</p> <p><b>0b001</b> Snoop Filter RAM</p> <p><b>0b010</b> Tag RAM</p> <p><b>0b011</b> Data RAM - accessing cacheline data</p> <p><b>0b111</b> Data RAM - accessing cacheline <i>Memory Tagging Extension</i> (MTE) tags</p>

The following table shows how to determine the slice ID width from the number of cache slices configured:

**Table 14-2: 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:



**Note**

In the following table, the L3 Tag RAM address width for the 3072KB and 4096KB 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 14-3: 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
3072KB-4096KB	12

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. The snoop filter RAM address width is also 11-bits for the 3072kB and 4096kB sizes, but there are four banks, so the effective width is 13-bits. For these configurations, the *Least Significant Bit* (LSB) of the SLCID\_IDX field selects which bank to access.

**Table 14-4: 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
3072KB, 4096KB	13

The following table describes how to interpret RAM data read back from the DSU-120AE CLUSTERCDBG register, for a snoop filter access:

**Table 14-5: 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.
[39:38]	2	This field has the following values:  <b>0b00</b> Cache line is invalid  <b>0b01</b> Cluster received a shared copy of the cache line. The cores know it is shared.  <b>0b10</b> 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).  <b>0b11</b> 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:  <b>0</b> The cache line is Secure <b>1</b> The cache line is Non-secure
[24:0]	25	Physical address tag. The encoding of these bits depends on IMP_CLUSTERCFR_EL1.SFIDX as follows:  <b>0x9</b> { PA[39:15] <b>0xA</b> { PA[39:16], 1'b0} <b>0xB</b> { PA[39:17], 2'b00} <b>0xC</b> { PA[39:18], 3'b000}

The following table describes how to interpret RAM data read back from the DSU-120AE CLUSTERCDBG register, for a tag RAM access:

**Table 14-6: CLUSTERCDBG bit descriptions for a tag RAM access**

Bits	Width (bits)	Description
[63:58]	6	RAZ

Bits	Width (bits)	Description
[57]	1	<i>Memory System Resource Partitioning and Monitoring (MPAM)</i> - 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 PARTID or a Secure state PARTID.
[49:46]	4	<i>Page-Based Hardware Attribute (PBHA)</i> . 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:  <b>0b00</b> MTE tag is Invalid  <b>0b01</b> MTE tag is Clean  <b>0b10</b> 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:  <b>0</b> This hints that the system should not allocate the cache line.  <b>1</b> This hints that the system should allocate the cache line.
[42]	1	PF (PreFetch). The possible values are:  <b>0</b> This hints that the cache line is not considered to be a prefetch (the cache line has been used).  <b>1</b> This hints that the cache line is an unused prefetch.

Bits	Width (bits)	Description
[41]	1	<p>CP (CPU Presence). The possible values are:</p> <p><b>0</b></p> <p>Indicates that there is no snoop filter entry for this cache line.</p> <p><b>1</b></p> <p>Indicates a snoop filter entry for this cache line.</p>
[40:39]	2	<p>Tag RAM State. The possible values are:</p> <p><b>0b00</b></p> <p>Cache line entry Invalid</p> <p><b>0b01</b></p> <p>UniqueDirty. The cluster has a unique, modified (dirty) copy of the cache line.</p> <p><b>0b10</b></p> <p>SharedClean. The cluster has a shared copy of the cache line that is coherent with the external memory location.</p> <p><b>0b11</b></p> <p>UniqueClean. The cluster has a unique copy of the cache line.</p>
[38:27]	12	RAZ
[26]	12	<p>NS (Non-Secure). The possible values are:</p> <p><b>0</b></p> <p>The cache line is Secure.</p> <p><b>0</b></p> <p>The cache line is Non-secure.</p>



Bits	Width (bits)	Description
[25:0]	26	<p>Physical Address Tag</p> <p>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.</p> <p>For (L3 sets/L3 cache slices), the possible values are:</p> <p><b>256</b> {PA[39:14]}</p> <p><b>512</b> {PA[39:15],1'b0}</p> <p><b>1024</b> {PA[39:16],2'b00}</p> <p><b>2048</b> {PA[39:17],3'b000}</p> <p>Where PA is the Physical Address width.</p>

The following table describes how to interpret the data read back from the DSU-120AE CLUSTERCDBG register, for a Data RAM data access:

**Table 14-7: 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-120AE CLUSTERCDBG register, for a Data RAM tag value access:

**Table 14-8: CLUSTERCDBG bit descriptions for a Data RAM MTE tag value access**

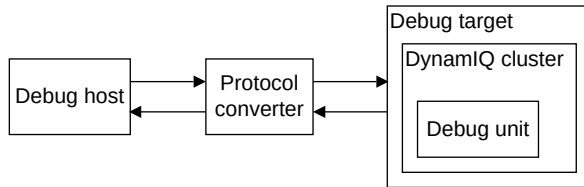
Bits	Width (bits)	Description
[63:16]	-	RAZ
[15:12]	4	MTE tag for selected cache line bits [511:384]
[11:8]	4	MTE tag for selected cache line bits [383:256]
[7:4]	4	MTE tag for selected cache line bits [255:128]
[3:0]	4	MTE tag for selected cache line bits [127:0]

## 14.2 Supported debug methods

The DSU-120AE 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 14-2: 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-120AE* (DSU-120AE) based devices, the mechanism used to access the debug unit is based on the CoreSight architecture. The DSU-120AE itself is accessed using an APB completer interface. An example of a debug target is a development system with a test chip or a silicon part with a DSU-120AE.

### Debug unit

Helps debugging software that is running on the core:

- DSU-120AE 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.

## 14.3 Terminology

The DSU-120AE 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

[1.8 Core, complex, and processing element numbering](#) on page 47

## 14.4 Cluster and core Debug power control

The cores included in the DSU-120AE, support the FEAT\_DoPD *Debug over PowerDown* architectural extension.

The FEAT\_DoPD provides a Debug programmers model where:

- The Debug, *Performance Monitoring Unit* (PMU), and *Embedded Trace Extension* (ETE) registers are all in the core power domain.
- The *Cross Trigger Interface* (CTI) registers for the cores and cluster are all in the Debug power domain.



Note

In the DSU, the CTI registers for both the cores and the cluster are included in the DebugBlock.

---

The CoreSight Granular Power Requesters, in the DebugBlock ROM table and cluster ROM table, provide the Debug power control for the cluster and core power domains.

### Requesting powerup of the core and cluster Debug registers with the Granular Power Requester registers

In order to access the debug registers included in the core and cluster, the debugger makes power control requests for the cluster power domain, using the DebugBlock ROM table power control (DBROM\_DBGPCRO) register. After the cluster is confirmed as powered up, the debugger makes debug power requests for the core power domains using the cluster ROM table power control registers (CLUSTERROM\_DBGPCRO-CLUSTERROM\_DBGPCR13).



- If the Debugger attempts to access the core Debug registers before the core domain is powered up, it receives an error response.
- Support for reset catch, to make the cores enter Debug state immediately upon exiting reset, is provided with the cross trigger interface Device Control (CTIDEVCTL) register *Reset Catch Enable* (RCE) field.

For example, for a cluster configured with a single core, the powerup request sequence is:

1. The debugger makes a power control request to the cluster power domain, for the cluster to be powered up, using the DBROM\_DBGPCR0 register.
2. The Debugger must poll the Cluster Power Status Register (DBROM\_DBGPSR0) to confirm that the cluster is powered up.
3. Once confirmed that the cluster is powered up, the debugger makes a power control request to the core 0 power domain for core 0 to be powered up, using the CLUSTERROM\_DBGPCR0 register.
4. The Debugger must poll the Cluster ROM table Debug Power Status Register0 (CLUSTERROM\_DBGPSR0) for core 0 to confirm that the core is powered up.

### Related information

[B.2.3.17 DBROM\\_DBGPCR0, DebugBlock ROM table Debug Power Control Register 0](#) on page 896

[B.2.3.18 DBROM\\_DBGPSR0, DebugBlock ROM table Debug Power Status Register 0](#) on page 897

[B.2.2.17 CLUSTERROM\\_DBGPCR0, Cluster ROM table Debug Power Control Register 0](#) on page 818

[B.2.2.31 CLUSTERROM\\_DBGPSR0, Cluster ROM table Debug Power Status Register 0](#) on page 837

[B.2.1.31 CTIDEVCTL, CTI Device Control register](#) on page 744

[B.2.3 External debug ROM registers summary](#) on page 873

[B.2.2 External cluster ROM registers summary](#) on page 771

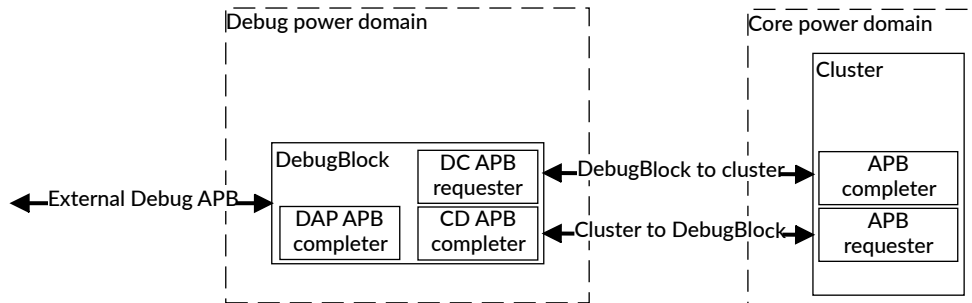
## 14.5 DebugBlock overview

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-120AE (DSU-120AE) also allows powering down the DebugBlock when debug is not in process.

The following figure shows how the DebugBlock is connected to the cluster.

**Figure 14-3: Debug APB connections**



The DebugBlock has the following APB interfaces:

#### External Debug APB (DAP APB)

An APB completer 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 requester 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 completer 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® Architecture Reference Manual for A-profile architecture](#).

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

[14. Debug](#) on page 217

[14.6 DebugBlock subcomponents](#) on page 230

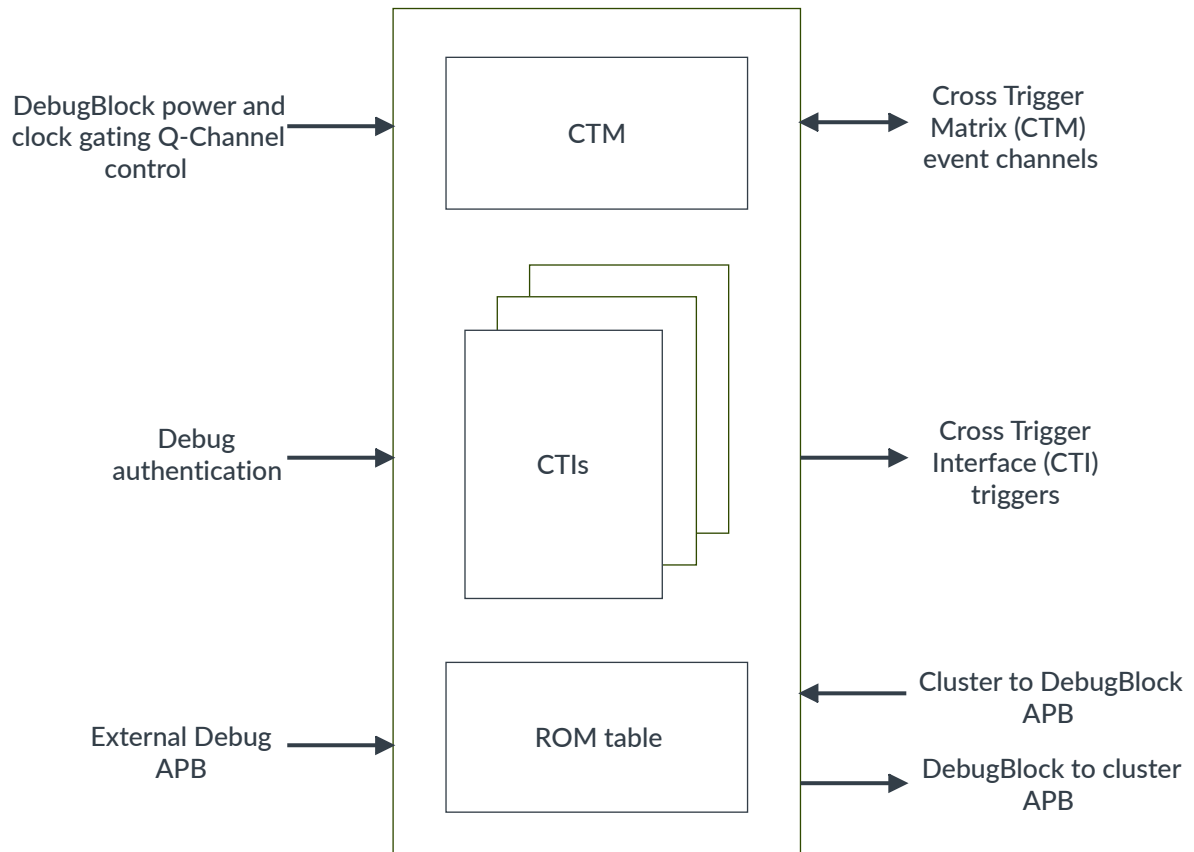
[14.7 Embedded Cross Trigger overview](#) on page 232

## 14.6 DebugBlock subcomponents

The DebugBlock component consists of various subcomponents that facilitate the debugging of the DSU-120AE DynamIQ™ cluster while the cores, complexes, and cluster are powered down.

The following figure shows the DebugBlock.

**Figure 14-4: DebugBlock block diagram**



The CTIs shown in the diagram include the CTI attached to each core and the cluster CTI.

### Cross Trigger Matrix (CTM)

The CTM distributes trigger events between the CTI instances inside the DebugBlock. The CTM event channels connect the DebugBlock CTM to the system-level CTM.

### Cross Trigger Interface (CTI)

The CTIs generate and receive debug trigger events. The trigger events are transmitted to and from the cluster using the cluster and DebugBlock APB interfaces.

### APB ROM table

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 [15. ROM tables](#) on page 239 for more information on ROM tables.

## Related information

[14.7 Embedded Cross Trigger overview](#) on page 232

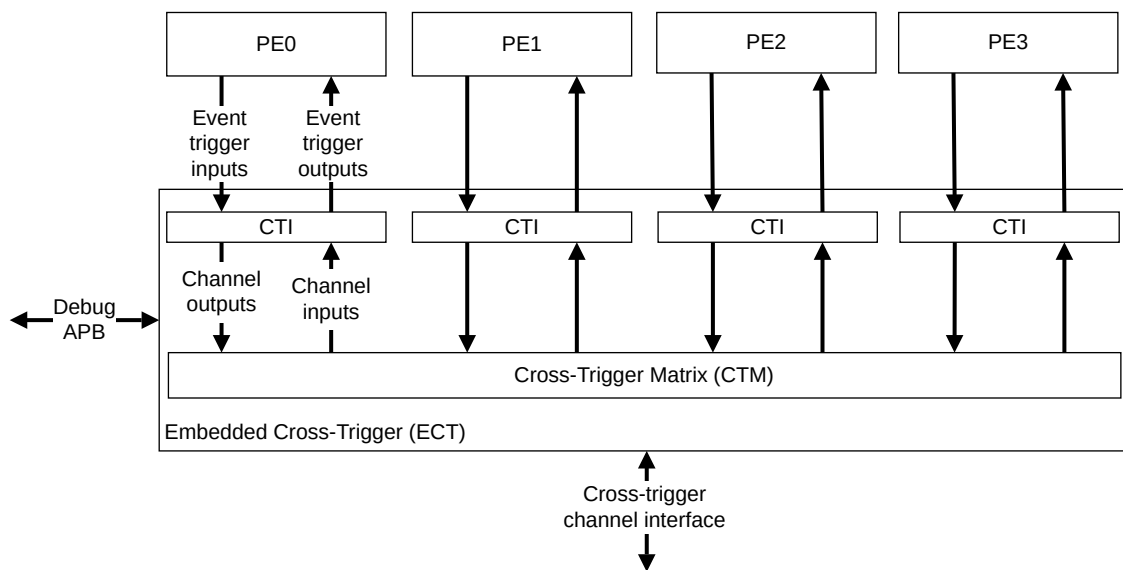
# 14.7 Embedded Cross Trigger overview

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. The CTIs are interconnected through a *Cross Trigger Matrix* (CTM) to send debug and trace events between PEs.

The following figure shows a conceptual view of the trigger event inputs and outputs between the PEs and the ECT.

**Figure 14-5: 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

[14.7.1 CTI triggers](#) on page 233

[14.6 DebugBlock subcomponents](#) on page 230

## 14.7.1 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 14-9: 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 event 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 14-10: 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 <i>Generic Interrupt Controller</i> (GIC) for the PE.
3	Reserved	-	-	-	Reserved
4-7	ETE trace external input	CTI	ETE	Pulse	This trigger event is sent to the <i>Embedded Trace Extension</i> (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 14-11: 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 14-12: 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 <i>Generic Interrupt Controller</i> (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

[14.7 Embedded Cross Trigger overview](#) on page 232

## 14.8 External CTI registers

The cluster *Cross Trigger Interface* (CTI) registers and core CTI registers are only accessible using memory-mapped accesses over the Debug APB interface.

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**.
- 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 14-13: CTI registers summary**

Offset	Name	Reset	Width	Description	Identical core CTI
0x000	<a href="#">CTICONTROL</a>	See individual bit resets.	32-bit	CTI Control register	Yes
0x010	<a href="#">CTIINTACK</a>	See individual bit resets.	32-bit	CTI Output Trigger Acknowledge register	Yes
0x014	<a href="#">CTIAPPSET</a>	See individual bit resets.	32-bit	CTI Application Trigger Set register	Yes
0x018	<a href="#">CTIAPPCLEAR</a>	See individual bit resets.	32-bit	CTI Application Trigger Clear register	Yes
0x01C	<a href="#">CTIAPPPULSE</a>	See individual bit resets.	32-bit	CTI Application Pulse register	Yes
0x20	<a href="#">CTIINEN0</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x24	<a href="#">CTIINEN1</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x28	<a href="#">CTIINEN2</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x2C	<a href="#">CTIINEN3</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x30	<a href="#">CTIINEN4</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x34	<a href="#">CTIINEN5</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x38	<a href="#">CTIINEN6</a>	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes

Offset	Name	Reset	Width	Description	Identical core CTI
0x3C	CTIINEN7	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x40	CTIINEN8	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x44	CTIINEN9	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0xA0	CTIOUTEN0	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xA4	CTIOUTEN1	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xA8	CTIOUTEN2	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xAC	CTIOUTEN3	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xB0	CTIOUTEN4	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xB4	CTIOUTEN5	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xB8	CTIOUTEN6	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xBC	CTIOUTEN7	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xC0	CTIOUTEN8	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xC4	CTIOUTEN9	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0x130	CTITRIGINSTATUS	See individual bit resets.	32-bit	CTI Trigger In Status register	Yes
0x134	CTITRIGOUTSTATUS	See individual bit resets.	32-bit	CTI Trigger Out Status register	Yes
0x138	CTICHINSTATUS	See individual bit resets.	32-bit	CTI Channel In Status register	Yes
0x13C	CTICHOUTSTATUS	See individual bit resets.	32-bit	CTI Channel Out Status register	Yes
0x140	CTIGATE	See individual bit resets.	32-bit	CTI Channel Gate Enable register	Yes
0x150	CTIDEVCTL	See individual bit resets.	32-bit	CTI Device Control register	Yes
0xFA0	CTICLAIMSET	See individual bit resets.	32-bit	CTI Claim Tag Set register	Yes
0xFA4	CTICLAIMCLR	See individual bit resets.	32-bit	CTI Claim Tag Clear register	Yes
0xFA8	CTIDEVAFF0	See individual bit resets.	32-bit	CTI Device Affinity register 0	No, see individual register
0xFAC	CTIDEVAFF1	See individual bit resets.	32-bit	CTI Device Affinity register 1	No, see individual register

Offset	Name	Reset	Width	Description	Identical core CTI
0xFB8	CTIAUTHSTATUS	See individual bit resets.	32-bit	CTI Authentication Status register	Yes
0xFBC	CTIDEVARCH	See individual bit resets.	32-bit	CTI Device Architecture register	Yes
0xFC0	CTIDEVID2	See individual bit resets.	32-bit	CTI Device ID register 2	Yes
0xFC4	CTIDEVID1	See individual bit resets.	32-bit	CTI Device ID register 1	Yes
0xFC8	CTIDEVID	See individual bit resets.	32-bit	CTI Device ID register 0	Yes
0xFCC	CTIDEVTYPE	See individual bit resets.	32-bit	CTI Device Type register	Yes
0xFD0	CTIPIDR4	See individual bit resets.	32-bit	CTI Peripheral Identification Register 4	No, see individual register
0xFE0	CTIPIDR0	See individual bit resets.	32-bit	CTI Peripheral Identification Register 0	No, see individual register
0xFE4	CTIPIDR1	See individual bit resets.	32-bit	CTI Peripheral Identification Register 1	No, see individual register
0xFE8	CTIPIDR2	See individual bit resets.	32-bit	CTI Peripheral Identification Register 2	No, see individual register
0xFEC	CTIPIDR3	See individual bit resets.	32-bit	CTI Peripheral Identification Register 3	No, see individual register
0xFF0	CTICIDR0	See individual bit resets.	32-bit	CTI Component Identification Register 0	Yes
0xFF4	CTICIDR1	See individual bit resets.	32-bit	CTI Component Identification Register 1	Yes
0xFF8	CTICIDR2	See individual bit resets.	32-bit	CTI Component Identification Register 2	Yes
0xFFC	CTICIDR3	See individual bit resets.	32-bit	CTI Component Identification Register 3	Yes

## 14.9 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 5 ATB-C interface, which is 32-bits wide in small clusters and 64-bits wide in larger clusters.



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.

## 14.10 CoreSight component identification

The following table lists the CoreSight ID values for the components present within the DSU-120AE.

For details of the CoreSight ID scheme, see the [Arm® CoreSight™ Architecture Specification v3.0](#).

**Table 14-14: CoreSight component identification**

Component	PID	CID	DevType	DevArch	Revision
DebugBlock ROM table	0x04001BB4EB	0xB105900D	0x00000000	0x47700AF7	rOp1
Cluster ROM	0x04001BB4EC	0xB105900D	0x00000000	0x47711A14	rOp1
Cluster CTI	0x04001BB4EC	0xB105900D	0x14	0x47700AF7	rOp1
Cluster PMU	0x04001BB4EC	0xB105900D	0x00000016	0x47702A16	rOp1

For details on the CoreSight component identification for the cluster ELA, see the [Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual](#).

## 15. 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-120AE* (DSU-120AE) 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-120AE 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-120AE include components for each core in the cluster, for example a *Cross Trigger Interface* (CTI) for each core in the cluster.



For a cluster comprised of complexes or cores, the core numbering follows the core instance numbering, see [1.8 Core, complex, and processing element numbering](#) on page 47.

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-120AE.

**Table 15-1: Types of components listed in the ROM tables for the DSU-120AE**

ROM table	ROM table located in	Components
DebugBlock	DebugBlock	<ul style="list-style-type: none"><li>Cluster CTI</li><li>CTI for each core</li><li>Power control and status registers for the cluster</li><li>Peripheral and component identification registers</li><li>ROM table entry point for the Cluster ROM table</li></ul>
Cluster	DebugBlock	<ul style="list-style-type: none"><li>Cluster <i>Performance Monitoring Unit</i> (PMU)</li><li>Cluster <i>Embedded Logic Analyzer</i> (ELA)</li><li>ROM table entry points for each standalone core or complex.</li><li>Power control and status registers for each standalone core or complex in the cluster.</li><li>Peripheral and component identification registers</li></ul>
Standalone core	Core	See the <i>Technical Reference Manual</i> (TRM) for your core
Complex	Complex	See the TRM for your core

## 15.1 Debug system address map

The debug system address map for the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) cluster depends on the specific implementation of your cluster, for example the number of cores configured in the cluster.

The following describes the conditions for certain entries being present in the address map:

### Core or Complex <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 is not 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.

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 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 ETE for the first core of the complex.

### Components

Components are only present for cores that are included in the cluster.

### Debug APB system address map

The following table shows the debug system address map for the DSU-120AE DynamIQ™ cluster.

**Table 15-2: Debug APB system address map**

Debug component (if present)	Debug APB address offset
DebugBlock ROM Table	0x0
Cluster ROM Table	0x10000
Cluster PMU	0x20000
Cluster ELA	0x30000
Cluster CTI	0x40000
Complex or core 0 ROM Table	0x80000
Core 0 Debug	0x90000
Core 0 PMU	0xA0000
Core 0 ETE	0xB0000
Core 0 ELA	0xC0000



Debug component (if present)	Debug APB address offset
Core 0 CTI	0xF0000
Complex or core 1 ROM Table	0x100000
Core 1 Debug	0x110000
Core 1 PMU	0x120000
Core 1 ETE	0x130000
Core 1 ELA	0x140000
Core 1 CTI	0x170000
Complex or core 2 ROM Table	0x180000
Core 2 Debug	0x190000
Core 2 PMU	0x1A0000
Core 2 ETE	0x1B0000
Core 2 ELA	0x1C0000
Core 2 CTI	0x1F0000
Complex or core 3 ROM Table	0x200000
Core 3 Debug	0x210000
Core 3 PMU	0x220000
Core 3 ETE	0x230000
Core 3 ELA	0x240000
Core 3 CTI	0x270000
Complex or core 4 ROM Table	0x280000
Core 4 Debug	0x290000
Core 4 PMU	0x2A0000
Core 4 ETE	0x2B0000
Core 4 ELA	0x2C0000
Core 4 CTI	0x2F0000
Complex or core 5 ROM Table	0x300000
Core 5 Debug	0x310000
Core 5 PMU	0x320000
Core 5 ETE	0x330000
Core 5 ELA	0x340000
Core 5 CTI	0x370000
Complex or core 6 ROM Table	0x380000
Core 6 Debug	0x390000
Core 6 PMU	0x3A0000
Core 6 ETE	0x3B0000
Core 6 ELA	0x3C0000
Core 6 CTI	0x3F0000
Complex or core 7 ROM Table	0x400000
Core 7 Debug	0x410000
Core 7 PMU	0x420000

Debug component (if present)	Debug APB address offset
Core 7 ETE	0x430000
Core 7 ELA	0x440000
Core 7 CTI	0x470000
Complex or core 8 ROM Table	0x480000
Core 8 Debug	0x490000
Core 8 PMU	0x4A0000
Core 8 ETE	0x4B0000
Core 8 ELA	0x4C0000
Core 8 CTI	0x4F0000
Complex or core 9 ROM Table	0x500000
Core 9 Debug	0x510000
Core 9 PMU	0x520000
Core 9 ETE	0x530000
Core 9 ELA	0x540000
Core 9 CTI	0x570000
Complex or core 10 ROM Table	0x580000
Core 10 Debug	0x590000
Core 10 PMU	0x5A0000
Core 10 ETE	0x5B0000
Core 10 ELA	0x5C0000
Core 10 CTI	0x5F0000
Complex or core 11 ROM Table	0x600000
Core 11 Debug	0x610000
Core 11 PMU	0x620000
Core 11 ETE	0x630000
Core 11 ELA	0x640000
Core 11 CTI	0x670000
Complex or core 12 ROM Table	0x680000
Core 12 Debug	0x690000
Core 12 PMU	0x6A0000
Core 12 ETE	0x6B0000
Core 12 ELA	0x6C0000
Core 12 CTI	0x6F0000
Complex or core 13 ROM Table	0x700000
Core 13 Debug	0x710000
Core 13 PMU	0x720000
Core 13 ETE	0x730000
Core 13 ELA	0x740000
Core 13 CTI	0x770000

## 15.2 DebugBlock ROM table

The DebugBlock ROM table contents depends 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 core.
- Entry point for the cluster ROM table
- Power control register to allow a cluster powerup request, see [15.4 ROM table power request registers for cluster and cores](#) on page 246.

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](#).



The DebugBlock ROM table part number is 0x4E9.

**Table 15-3: DebugBlock ROM table**

Offset	Name	Description
0x0000	ROMENTRY0	Cluster ROM table entry point
0x0004	ROMENTRY1	Cluster CTI
0x0008	ROMENTRY2	CTI for core 0
0x000C	ROMENTRY3	CTI for core 1
0x0010	ROMENTRY4	CTI for core 2
0x0014	ROMENTRY5	CTI for core 3
0x0018	ROMENTRY6	CTI for core 4
0x001C	ROMENTRY7	CTI for core 5
0x0020	ROMENTRY8	CTI for core 6
0x0024	ROMENTRY9	CTI for core 7
0x0028	ROMENTRY10	CTI for core 8
0x002C	ROMENTRY11	CTI for core 9
0x0030	ROMENTRY12	CTI for core 10
0x0034	ROMENTRY13	CTI for core 11
0x0038	ROMENTRY14	CTI for core 12
0x003C	ROMENTRY15	CTI for core 13
0x0048-0x09FC	-	Reserved
0x0A00	DBGPCRO	Debug Power Control Register
0x0A04-0x0A7C	-	Reserved
0x0A80	DBGPSRO	Debug Power Status Register

Offset	Name	Description
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

## 15.3 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 [15.4 ROM table power request registers for cluster and cores](#) on page 246.

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 0x4EA.

**Table 15-4: 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	ROMENTRY10	Core 8 ROM table
0x002C	ROMENTRY11	Core 9 ROM table
0x0030	ROMENTRY12	Core 10 ROM table
0x0034	ROMENTRY13	Core 11 ROM table
0x0038	ROMENTRY14	Core 12 ROM table
0x003C	ROMENTRY15	Core 13 ROM table
0x0040-0x09FC	-	Reserved
0x0A00-0x0A34	DBGPCR<n>	Debug Power Control Register for core <n>
0x0A80-0x0AB4	DBGPSR<n>	Debug Power Status Register for core <n>
0x0AB8-0x0BFC	-	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

Offset	Name	Description
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

## 15.4 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-120AE DynamIQ™ cluster.

Each corresponding core or cluster PPU then reacts to the request that was made as appropriate.

## 15.5 External cluster ROM registers

The cluster ROM table registers are only accessible using memory-mapped accesses over the debug APB interface.

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**.
- The number of registers that contain valid entries depends on the number of cores configured for the cluster.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table 15-5: CLUSTERROM registers summary**

Offset	Name	Reset	Width	Description
0x000	CLUSTERROM_ROMENTRY0	See individual bit resets.	32-bit	Cluster ROM table entry 0
0x004	CLUSTERROM_ROMENTRY1	See individual bit resets.	32-bit	Cluster ROM table entry 1
0x008	CLUSTERROM_ROMENTRY2	See individual bit resets.	32-bit	Cluster ROM table entry 2
0x00C	CLUSTERROM_ROMENTRY3	See individual bit resets.	32-bit	Cluster ROM table entry 3
0x010	CLUSTERROM_ROMENTRY4	See individual bit resets.	32-bit	Cluster ROM table entry 4
0x014	CLUSTERROM_ROMENTRY5	See individual bit resets.	32-bit	Cluster ROM table entry 5
0x018	CLUSTERROM_ROMENTRY6	See individual bit resets.	32-bit	Cluster ROM table entry 6
0x01C	CLUSTERROM_ROMENTRY7	See individual bit resets.	32-bit	Cluster ROM table entry 7
0x020	CLUSTERROM_ROMENTRY8	See individual bit resets.	32-bit	Cluster ROM table entry 8
0x024	CLUSTERROM_ROMENTRY9	See individual bit resets.	32-bit	Cluster ROM table entry 9
0x028	CLUSTERROM_ROMENTRY10	See individual bit resets.	32-bit	Cluster ROM table entry 10
0x02C	CLUSTERROM_ROMENTRY11	See individual bit resets.	32-bit	Cluster ROM table entry 11
0x030	CLUSTERROM_ROMENTRY12	See individual bit resets.	32-bit	Cluster ROM table entry 12
0x034	CLUSTERROM_ROMENTRY13	See individual bit resets.	32-bit	Cluster ROM table entry 13
0x038	CLUSTERROM_ROMENTRY14	See individual bit resets.	32-bit	Cluster ROM table entry 14
0x03C	CLUSTERROM_ROMENTRY15	See individual bit resets.	32-bit	Cluster ROM table entry 15
0xA00	CLUSTERROM_DBGPCR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 0
0xA04	CLUSTERROM_DBGPCR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 1
0xA08	CLUSTERROM_DBGPCR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 2
0xA0C	CLUSTERROM_DBGPCR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 3
0xA10	CLUSTERROM_DBGPCR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 4
0xA14	CLUSTERROM_DBGPCR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 5
0xA18	CLUSTERROM_DBGPCR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 6
0xA1C	CLUSTERROM_DBGPCR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 7
0xA20	CLUSTERROM_DBGPCR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 8
0xA24	CLUSTERROM_DBGPCR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 9
0xA28	CLUSTERROM_DBGPCR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 10
0xA2C	CLUSTERROM_DBGPCR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 11
0xA30	CLUSTERROM_DBGPCR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 12
0xA34	CLUSTERROM_DBGPCR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 13
0xA80	CLUSTERROM_DBGPSR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 0
0xA84	CLUSTERROM_DBGPSR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 1
0xA88	CLUSTERROM_DBGPSR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 2
0xA8C	CLUSTERROM_DBGPSR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 3
0xA90	CLUSTERROM_DBGPSR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 4
0xA94	CLUSTERROM_DBGPSR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 5
0xA98	CLUSTERROM_DBGPSR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 6
0xA9C	CLUSTERROM_DBGPSR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 7
0xAA0	CLUSTERROM_DBGPSR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 8

Offset	Name	Reset	Width	Description
0xAA4	<a href="#">CLUSTERROM_DBGPSR9</a>	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 9
0xAA8	<a href="#">CLUSTERROM_DBGPSR10</a>	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 10
0xAAC	<a href="#">CLUSTERROM_DBGPSR11</a>	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 11
0xAB0	<a href="#">CLUSTERROM_DBGPSR12</a>	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 12
0xAB4	<a href="#">CLUSTERROM_DBGPSR13</a>	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 13
0xC00	<a href="#">CLUSTERROM_PRIDR0</a>	See individual bit resets.	32-bit	Cluster ROM table Power Request ID Register 0
0xFB8	<a href="#">CLUSTERROM_AUTHSTATUS</a>	See individual bit resets.	32-bit	Cluster ROM table Authentication Status Register
0xFBC	<a href="#">CLUSTERROM_DEVARCH</a>	See individual bit resets.	32-bit	Cluster ROM table Device Architecture Register
0xFC8	<a href="#">CLUSTERROM_DEVID</a>	See individual bit resets.	32-bit	Cluster ROM table Device Configuration Register
0xFCC	<a href="#">CLUSTERROM_DEVTYPE</a>	See individual bit resets.	32-bit	Cluster ROM table Device Type Register
0xFD0	<a href="#">CLUSTERROM_PIDR4</a>	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 4
0xFE0	<a href="#">CLUSTERROM_PIDR0</a>	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 0
0xFE4	<a href="#">CLUSTERROM_PIDR1</a>	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 1
0xFE8	<a href="#">CLUSTERROM_PIDR2</a>	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 2
0xFEC	<a href="#">CLUSTERROM_PIDR3</a>	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 3
0xFF0	<a href="#">CLUSTERROM_CIDR0</a>	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 0
0xFF4	<a href="#">CLUSTERROM_CIDR1</a>	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 1
0xFF8	<a href="#">CLUSTERROM_CIDR2</a>	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 2
0xFFC	<a href="#">CLUSTERROM_CIDR3</a>	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 3

## 15.6 External debug ROM registers

The debug ROM table registers are only accessible using memory-mapped accesses over the debug APB interface.

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 debug ROM table register values are based on a cluster, where DSU-120AE NUM\_CORES parameter is set to 14.
- 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**.
- The number of registers that contain valid entries depends on the number of cores configured for the cluster.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.



**Table 15-6: DBROM registers summary**

Offset	Name	Reset	Width	Description
0x000	DBROM_ROMENTRY0	See individual bit resets.	32-bit	DebugBlock ROM table Entry 0
0x004	DBROM_ROMENTRY1	See individual bit resets.	32-bit	DebugBlock ROM table Entry 1
0x008	DBROM_ROMENTRY2	See individual bit resets.	32-bit	DebugBlock ROM table Entry 2
0x00C	DBROM_ROMENTRY3	See individual bit resets.	32-bit	DebugBlock ROM table Entry 3
0x010	DBROM_ROMENTRY4	See individual bit resets.	32-bit	DebugBlock ROM table Entry 4
0x014	DBROM_ROMENTRY5	See individual bit resets.	32-bit	DebugBlock ROM table Entry 5
0x018	DBROM_ROMENTRY6	See individual bit resets.	32-bit	DebugBlock ROM table Entry 6
0x01C	DBROM_ROMENTRY7	See individual bit resets.	32-bit	DebugBlock ROM table Entry 7
0x020	DBROM_ROMENTRY8	See individual bit resets.	32-bit	DebugBlock ROM table Entry 8
0x024	DBROM_ROMENTRY9	See individual bit resets.	32-bit	DebugBlock ROM table Entry 9
0x028	DBROM_ROMENTRY10	See individual bit resets.	32-bit	DebugBlock ROM table Entry 10
0x02C	DBROM_ROMENTRY11	See individual bit resets.	32-bit	DebugBlock ROM table Entry 11
0x030	DBROM_ROMENTRY12	See individual bit resets.	32-bit	DebugBlock ROM table Entry 12
0x034	DBROM_ROMENTRY13	See individual bit resets.	32-bit	DebugBlock ROM table Entry 13
0x038	DBROM_ROMENTRY14	See individual bit resets.	32-bit	DebugBlock ROM table Entry 14
0x03C	DBROM_ROMENTRY15	See individual bit resets.	32-bit	DebugBlock ROM table Entry 15
0xA00	DBROM_DBGPCRO	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Control Register 0
0xA80	DBROM_DBGPSRO	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Status Register 0
0xC00	DBROM_PRIDRO	See individual bit resets.	32-bit	DebugBlock ROM table Power Request ID Register 0
0xFB8	DBROM_AUTHSTATUS	See individual bit resets.	32-bit	DebugBlock ROM table Authentication Status Register
0xFBC	DBROM_DEVARCH	See individual bit resets.	32-bit	DebugBlock ROM table Device Architecture Register
0xFC8	DBROM_DEVID	See individual bit resets.	32-bit	DebugBlock ROM table Device Configuration Register
0xFCC	DBROM_DEVTYPE	See individual bit resets.	32-bit	DebugBlock ROM table Device Type Register
0xFD0	DBROM_PIDR4	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 4
0xFE0	DBROM_PIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 0
0xFE4	DBROM_PIDR1	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 1
0xFE8	DBROM_PIDR2	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 2
0xFEC	DBROM_PIDR3	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 3
0xFF0	DBROM_CIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 0
0xFF4	DBROM_CIDR1	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 1
0xFF8	DBROM_CIDR2	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 2
0xFFC	DBROM_CIDR3	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 3

# 16. Performance Monitors Extension support

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) 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.

## 16.1 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.

### PMU register interfaces

The *DynamIQ™ Shared Unit-120AE* (DSU-120AE) supports access to the performance monitor registers from the internal System register interface. The DSU-120AE also supports access to the PMU through the memory-mapped Debug APB interface.

## 16.2 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 16-1: PMU events**

PMU event number	Event mnemonic	Event description
0x0011	CYCLES	Cycle counter
0x0019	BUS_ACCESS	Bus access counter  Counts every beat of data that is transferred over the data channels between the <i>Snoop Control Unit</i> (SCU) and the interconnect. If both read and write beats are transferred on a given cycle, this event is counted twice on that cycle.  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 ( <i>Error Correcting Code</i> (ECC) or parity) in the protected RAMs.
0x001D	BUS_CYCLES	ACE or CHI bus cycle counter.
0x0029	L3D_CACHE_ALLOCATE	L3 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	L3 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	L3 unified cache access counter  Counts every Cacheable read or write transaction issued to the <i>Snoop Control Unit</i> (SCU).  This event counts the sum of L3D_CACHE_RD and L3D_CACHE_WR.
0x002C	L3D_CACHE_WB	L3 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 SCU and the interconnect.  <b>Note:</b> 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.

PMU event number	Event mnemonic	Event description
0x0061	BUS_ACCESS_WR	<p>Bus access, write counter</p> <p>Counts every beat of data transferred over the write data channel between the SCU and the interconnect.</p> <p><b>Note:</b> 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.</p>
0x0062	BUS_ACCESS_SHARED	<p>Bus access, shared counter</p> <p>Counts every beat of shared data transferred over the data channels between the SCU and the interconnect.</p>
0x0063	BUS_ACCESS_NOT_SHARED	<p>Bus access, not shared counter</p> <p>Counts every beat of not shared data transferred over the write data channel between the SCU and the interconnect.</p>
0x0064	BUS_ACCESS_NORMAL	<p>Bus access, normal counter</p> <p>Counts every beat of normal data transferred over the write data channel between the SCU and the interconnect.</p>
0x0065	BUS_ACCESS_PERIPH	<p>Bus access, periph counter</p> <p>Counts every beat of Device data transferred over the write data channel between the SCU and the interconnect.</p>
0x00A0	L3D_CACHE_RD	<p>L3 unified cache access, read counter</p> <p>Counts every Cacheable shareable read transaction that is issued to the SCU. Prefetches and stashes are not counted.</p>
0x00A1	L3D_CACHE_WR	<p>L3 unified cache access, write counter</p> <p>Counts every Cacheable write transaction issued to the SCU.</p>
0x00A2	L3D_CACHE_REFILL_RD	<p>L3 unified cache refill, read counter</p> <p>Counts every Cacheable read transaction issued to the interconnect caused by a Cacheable shareable read transaction. Prefetches and stashes are not counted.</p>
0x00A3	L3D_CACHE_REFILL_WR	<p>L3 unified cache refill, write counter</p> <p>Counts every Cacheable read transaction issued to the interconnect caused by a write transaction.</p>
0x0119	ACP_ACCESS	<p><i>Accelerator Coherency Port (ACP) access counter</i></p> <p>Counts every beat of data transferred over the data channels between the SCU and the ACP. If both read and write data beats are transferred on a given cycle, this event is counted twice on that cycle.</p> <p>This event counts the sum of ACP_ACCESS_RD and ACP_ACCESS_WR.</p>
0x011D	ACP_CYCLES	ACP cycle counter

PMU event number	Event mnemonic	Event description
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. If both read and write data beats are transferred on a given cycle, this event is counted twice on that cycle.  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	Snoop access counter  Counts every 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 either: <ul style="list-style-type: none"> <li>• The stash hit in the cluster or;</li> <li>• The stash is not performed due to the L3 cache being off.</li> </ul>
0x0510	SCU_STASH_ICN_ACCESS	Stash access, ICN counter  Counts every stash transaction originating from the interconnect.

PMU event number	Event mnemonic	Event description
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.
0x0520	SCU_STASH_ACP_ACCESS	Stash access, ACP counter Counter for every stash-supported transaction originating from an ACP.
0x0521	SCU_STASH_ACP_MISS	Stash data miss, ACP counter. Counter for every dataless stash 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 either: <ul style="list-style-type: none"> <li>The stash hit in the cluster or;</li> <li>The stash was not performed due to L3 cache being off.</li> </ul>
0x00D0	SCU_HZD_ADDRESS	Arbitration hazard, address counter  Counts every flush caused by an address hazard.
0x00F3	SCU_BIB_ACCESS	Counts every snoop filter access due to snoop filter maintenance activity.
0x00F4	SCU_BACK_INVALIDATE	Back invalidation counter  Counts when a core must be snooped to invalidate a line because of not enough capacity in the snoop filter.
0x00F5	SCU_BIB_ECC	BIB ECC errors counter  ECC errors detected on a back invalidation accesses that cause a way to be avoided but are not corrected or reported in the <i>Reliability, Availability, and Serviceability</i> (RAS) registers.

## 16.3 PMU interrupt

The DSU-120AE 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).

## 16.4 External cluster PMU registers

The cluster *Performance Monitoring Unit* (PMU) registers are accessible either from memory-mapped accesses over the debug APB interface or from System register accesses from the cores.

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**.
- 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 16-2: CLUSTERPMU registers summary**

Offset	Name	Reset	Width	Description
0x0	<a href="#">CLUSTERPMU_PMEVCNTR0</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x8	<a href="#">CLUSTERPMU_PMEVCNTR1</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x10	<a href="#">CLUSTERPMU_PMEVCNTR2</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x18	<a href="#">CLUSTERPMU_PMEVCNTR3</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x20	<a href="#">CLUSTERPMU_PMEVCNTR4</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x28	<a href="#">CLUSTERPMU_PMEVCNTR5</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x400	<a href="#">CLUSTERPMU_PMEVTYPER0</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x404	<a href="#">CLUSTERPMU_PMEVTYPER1</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x408	<a href="#">CLUSTERPMU_PMEVTYPER2</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x40C	<a href="#">CLUSTERPMU_PMEVTYPER3</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x410	<a href="#">CLUSTERPMU_PMEVTYPER4</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x414	<a href="#">CLUSTERPMU_PMEVTYPER5</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x600	<a href="#">CLUSTERPMU_PMEVCNTRSRO</a>	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers

Offset	Name	Reset	Width	Description
0x608	CLUSTERPMU_PMEVCNTR1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x610	CLUSTERPMU_PMEVCNTR2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x618	CLUSTERPMU_PMEVCNTR3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x620	CLUSTERPMU_PMEVCNTR4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x628	CLUSTERPMU_PMEVCNTR5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x638	CLUSTERPMU_PMSSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Status register
0x640	CLUSTERPMU_PMOVSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Status Snapshot register
0xC00	CLUSTERPMU_PMCNTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Set register
0xC20	CLUSTERPMU_PMCNTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Clear register
0xC40	CLUSTERPMU_PMINTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Set register
0xC60	CLUSTERPMU_PMINTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Clear register
0xC80	CLUSTERPMU_PMOVSCCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Clear register
0xCC0	CLUSTERPMU_PMOVSSSET	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Set register
0xE00	CLUSTERPMU_PMCFGR	See individual bit resets.	32-bit	Cluster Performance Monitors Configuration Register
0xE04	CLUSTERPMU_PMCR	See individual bit resets.	32-bit	Cluster Performance Monitors Control Register
0xE08	CLUSTERPMU_PMIIDR	See individual bit resets.	32-bit	Cluster Performance Monitors Implementation Identification register
0xE20	CLUSTERPMU_PMCEID0	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 0
0xE24	CLUSTERPMU_PMCEID1	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 1
0xE28	CLUSTERPMU_PMCEID2	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 2
0xE2C	CLUSTERPMU_PMCEID3	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 3
0xE30	CLUSTERPMU_PMSSCR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Capture register
0xE38	CLUSTERPMU_PMSSRR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Reset register
0xFA8	CLUSTERPMU_PMDEVAFFO	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 0



Offset	Name	Reset	Width	Description
0xFAC	CLUSTERPMU_PMDEVAFF1	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 1
0xFB8	CLUSTERPMU_PMAUTHSTATUS	See individual bit resets.	32-bit	Cluster Performance Monitors Authentication Status register
0xFBC	CLUSTERPMU_PMDEVARCH	See individual bit resets.	32-bit	Cluster Performance Monitors Device Architecture register
0xFC8	CLUSTERPMU_PMDEVID	See individual bit resets.	32-bit	Cluster Performance Monitors Device ID register
0xFCC	CLUSTERPMU_PMDEVTYPE	See individual bit resets.	32-bit	Cluster Performance Monitors Device Type register
0xFD0	CLUSTERPMU_PMPIDR4	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 4
0xFE0	CLUSTERPMU_PMPIDR0	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 0
0xFE4	CLUSTERPMU_PMPIDR1	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 1
0xFE8	CLUSTERPMU_PMPIDR2	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 2
0xFEC	CLUSTERPMU_PMPIDR3	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 3
0xFF0	CLUSTERPMU_PMCIDR0	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 0
0xFF4	CLUSTERPMU_PMCIDR1	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 1
0xFF8	CLUSTERPMU_PMCIDR2	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 2
0xFFC	CLUSTERPMU_PMCIDR3	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 3

# 17. Activity Monitors Extension support

The DynamIQ™ Shared Unit-120AE core implements the Activity Monitors Extension to the Arm®v8.4-A architecture. Activity monitoring has features similar to performance monitoring features, but is intended for system management use whereas performance monitoring is aimed at user and debug applications.

The activity monitors provide useful information for system power management and persistent monitoring. The activity monitors are read-only in operation and accessed from the utility bus.

The DynamIQ™ Shared Unit-120AE implements five counters in one group, each of which is a 64-bit counter that counts a fixed event.

## 17.1 Activity monitors access

The DynamIQ™ Shared Unit-120AE supports memory-mapped access to activity monitors from the utility bus interface.

The base address for the cluster *Activity Monitor Unit* (AMU) registers on the utility bus interface is 0x040000. These registers are accessed from Secure state.

These registers are treated as RAZ/WI if either:

- The register is marked as Reserved.
- The register is accessed in the wrong Security state.
- The cluster is powered down.

See the [Arm® Architecture Reference Manual for A-profile architecture](#) for information on the memory mapping of these registers.

## 17.2 Activity monitors counters

The DynamIQ™ Shared Unit-120AE implements five activity monitors counters, 0-4.

Each counter has the following characteristics:

- All events are counted in 64-bit wrapping counters that wrap when they overflow. There is no support for overflow status indication or interrupts.
- Any change in clock frequency, including when a `WFI` and `WFE` instruction stops the clock, can affect any counter.
- All events, 0-4, are fixed. For the list of the cluster activity monitor events, see [17.4 Activity monitors events](#) on page 260.
- The activity monitor counters are reset to zero on a Warm or Cold reset of the power domain of the cluster. When the cluster is not in reset, activity monitoring is available.

## 17.3 External cluster AMU registers

The cluster *Activity Monitor Unit* (AMU) registers are only accessible from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the cluster AMU registers that are accessed externally (memory-mapped) from the utility bus of the DSU-120AE. For more information about a register, click on the register name in the table. For more information on the architecture of the AMU registers, see [Arm® CoreSight™ Performance Monitoring Unit Architecture](#).



Note

- The cluster AMU registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- The part number is 0x04EA.
- The base address for the cluster AMU registers is 0x040000.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table 17-1: CLUSTERAMU registers summary**

Offset	Name	Reset	Width	Description
0x0	<a href="#">CLUSTERAMU_AMEVCNTR0</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x8	<a href="#">CLUSTERAMU_AMEVCNTR1</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x10	<a href="#">CLUSTERAMU_AMEVCNTR2</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x18	<a href="#">CLUSTERAMU_AMEVCNTR3</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x20	<a href="#">CLUSTERAMU_AMEVCNTR4</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x400	<a href="#">CLUSTERAMU_AMEVTYPE0</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x404	<a href="#">CLUSTERAMU_AMEVTYPE1</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x408	<a href="#">CLUSTERAMU_AMEVTYPE2</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x40C	<a href="#">CLUSTERAMU_AMEVTYPE3</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x410	<a href="#">CLUSTERAMU_AMEVTYPE4</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0xC00	<a href="#">CLUSTERAMU_AMCNTENSET</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Count Enable Set register
0xC20	<a href="#">CLUSTERAMU_AMCNTENCLR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Count Enable Clear register
0xE00	<a href="#">CLUSTERAMU_AMCFGR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Configuration Register
0xE04	<a href="#">CLUSTERAMU_AMCR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Control Register
0xE08	<a href="#">CLUSTERAMU_AMIIDR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Implementation Identification register
0xFA8	<a href="#">CLUSTERAMU_AMDEVAFF</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Device Affinity register
0xFBC	<a href="#">CLUSTERAMU_AMDEVARCH</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Device Architecture register
0xFC8	<a href="#">CLUSTERAMU_AMDEVID</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Device ID register

Offset	Name	Reset	Width	Description
0xFCC	CLUSTERAMU_AMDEVTYPE	See individual bit resets.	32-bit	Cluster Activity Monitors Device Type register
0xFD0	CLUSTERAMU_AMPIDR4	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 4
0xFE0	CLUSTERAMU_AMPIDR0	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 0
0xFE4	CLUSTERAMU_AMPIDR1	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 1
0xFE8	CLUSTERAMU_AMPIDR2	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 2
0xFEC	CLUSTERAMU_AMPIDR3	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 3
0xFF0	CLUSTERAMU_AMCIDR0	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 0
0xFF4	CLUSTERAMU_AMCIDR1	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 1
0xFF8	CLUSTERAMU_AMCIDR2	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 2
0xFFC	CLUSTERAMU_AMCIDR3	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 3

## 17.4 Activity monitors events

Activity monitors events in the DynamIQ™ Shared Unit-120AE are all fixed, and they map to the activity monitors counters.

The following table shows the mapping of counters to fixed events.

**Table 17-2: Mapping of counters to fixed events**

Activity monitor counter <n>	Associated register	Event	Event number	Description
AMEVCNTR0	CLUSTERAMU_AMEVCNTR0	L3_CACHE_READ_HIT	0x0	L3 cache read hit  <b>Note:</b> This event counts the same information as the IMP_CLUSTERL3HIT_EL1 System register.
AMEVCNTR1	CLUSTERAMU_AMEVCNTR1	L3_CACHE_READ_MISS	0x1	L3 cache read miss  <b>Note:</b> This event counts the same information as the IMP_CLUSTERL3MISS_EL1 System register.
AMEVCNTR2	CLUSTERAMU_AMEVCNTR2	POST_L3_READ_OCCUPANCY	0x2	Post L3 read occupancy  Increments by n every cycle, where n is the number of Cacheable read transactions outstanding to the bus requester ports and peripheral port for that cycle.  You can use the value n to determine the average latency of a read by dividing by the post-L3 read transaction count.

Activity monitor counter <n>	Associated register	Event	Event number	Description
AMEVCNTR3	CLUSTERAMU_AMEVCNTR3	POST_L3_WRITE_TRANSACTIONS	0x3	Post L3 write transactions  Counts the number of Cacheable write transactions that are sent to the bus requester ports and peripheral port.
AMEVCNTR4	CLUSTERAMU_AMEVCNTR4	POST_L3_READ_TRANSACTIONS	0x4	Post L3 read transactions  Counts the number of Cacheable read transactions that are sent to the bus requester ports and peripheral port.



Transactions caused by atomic instructions that perform a read and a write are only counted once, as a read, for the activity monitors. Examples of these instructions include, atomic load, swap, and compare and swap instructions. Atomic store instructions are counted only as a write.

# Appendix A AArch64 registers

This appendix contains the descriptions for all the AArch64 registers in the *DynamIQ™ Shared Unit-120AE* (DSU-120AE).

## A.1 AArch64 generic system control registers summary

The cluster Generic System Control registers are accessible either from System register accesses from the cores or from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the AArch64 Generic System Control registers in the DSU-120AE. For more information about a register, click on the register name in the table.



For registers with 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
<a href="#">IMP_CLUSTERCFR_EL1</a>	3	0	C15	C3	0	See individual bit resets.	64-bit	Cluster Configuration Register
<a href="#">IMP_CLUSTERIDR_EL1</a>	3	0	C15	C3	1	See individual bit resets.	64-bit	Cluster Main Revision Register
<a href="#">IMP_CLUSTERREVIDR_EL1</a>	3	0	C15	C3	2	See individual bit resets.	64-bit	Cluster ECO ID Register
<a href="#">IMP_CLUSTERACTLR_EL1</a>	3	0	C15	C3	3	See individual bit resets.	64-bit	Cluster Auxiliary Control Register
<a href="#">IMP_CLUSTERECTLR_EL1</a>	3	0	C15	C3	4	See individual bit resets.	64-bit	Cluster Extended Control Register
<a href="#">IMP_CLUSTERPWRCTLR_EL1</a>	3	0	C15	C3	5	See individual bit resets.	64-bit	Cluster Power Control Register
<a href="#">IMP_CLUSTERPWRDN_EL1</a>	3	0	C15	C3	6	See individual bit resets.	64-bit	Cluster Power Down Register
<a href="#">IMP_CLUSTERPWRSTAT_EL1</a>	3	0	C15	C3	7	See individual bit resets.	64-bit	Cluster Power Status Register
<a href="#">IMP_CLUSTERL3DNTH0_EL1</a>	3	0	C15	C4	0	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register
<a href="#">IMP_CLUSTERL3DNTH1_EL1</a>	3	0	C15	C4	1	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register
<a href="#">IMP_CLUSTERL3UPTH0_EL1</a>	3	0	C15	C4	2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register
<a href="#">IMP_CLUSTERL3UPTH1_EL1</a>	3	0	C15	C4	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
IMP_CLUSTERBUSQOS_EL1	3	0	C15	C4	4	See individual bit resets.	64-bit	Cluster Bus QoS Control Register
IMP_CLUSTERL3HIT_EL1	3	0	C15	C4	5	See individual bit resets.	64-bit	Cluster L3 Hit Counter Register
IMP_CLUSTERL3MISS_EL1	3	0	C15	C4	6	See individual bit resets.	64-bit	Cluster L3 Miss Counter Register
IMP_CLUSTERPPSTART_EL1	3	0	C15	C9	0	See individual bit resets.	64-bit	Cluster Peripheral Port Start Address Register
IMP_CLUSTERPPEND_EL1	3	0	C15	C9	1	See individual bit resets.	64-bit	Cluster Peripheral Port End Address Register
IMP_CLUSTERRCFR2_EL1	3	0	C15	C9	2	See individual bit resets.	64-bit	Cluster Configuration Register 2
IMP_CLUSTERL3UPTH2_EL1	3	0	C15	C9	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register
IMP_CLUSTERCDBG_EL3	3	6	C15	C4	7	See individual bit resets.	64-bit	Cluster Cache Debug Register
IMP_CLUSTERPMMDCR_EL3	3	6	C15	C6	3	See individual bit resets.	64-bit	Monitor Debug Configuration Register (EL3)

## A.1.1 IMP\_CLUSTERCFR\_EL1, Cluster Configuration Register

Contains details of the hardware configuration of the cluster.

### Configurations

AArch64 register IMP\_CLUSTERCFR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.9 CLUSTERCFR, Cluster Configuration Register](#) on page 410 bits [63:0].

### 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	xxxx	x0xx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0

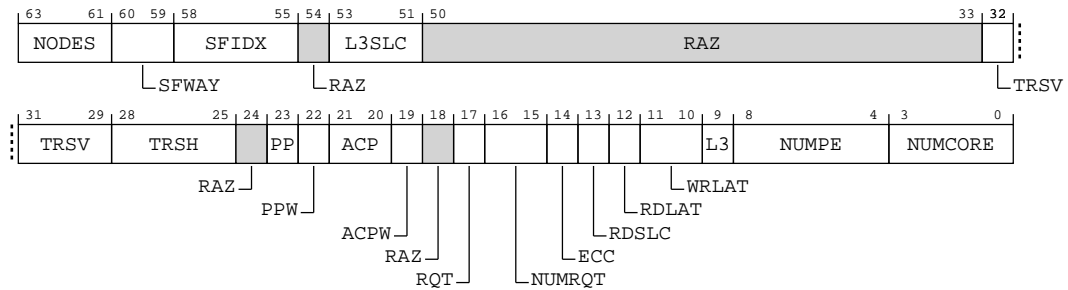


Note

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.  <b>0b001</b> One node.  <b>0b010</b> Two nodes.  <b>0b011</b> Three nodes.  <b>0b100</b> Four nodes.  <b>0b101</b> Eight nodes.	xxx
[60:59]	SFWAY	Number of Snoop Filter ways.  <b>0b00</b> 4 ways  <b>0b01</b> 6 ways  <b>0b10</b> 8 ways  <b>0b11</b> 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	<p>Number of L3 cache slices.</p> <p><b>0b000</b> Eight L3 cache slices.</p> <p><b>0b001</b> One L3 cache slice.</p> <p><b>0b010</b> Two L3 cache slices.</p> <p><b>0b100</b> Four L3 cache slices.</p>	xxx
[50:33]	RAZ	Reserved	RAZ
[32:29]	TRSV	<p>Transport register slices, vertical.</p> <p><b>0b0000</b> No register slices</p> <p><b>0b0001</b> One register slice</p> <p><b>0b0010</b> Two register slices</p> <p><b>0b0011</b> Three register slices</p> <p><b>0b0100</b> Four register slices</p> <p><b>0b0101</b> Five register slices</p> <p><b>0b0110</b> Six register slices</p> <p><b>0b0111</b> Seven register slices</p> <p><b>0b1000</b> Eight register slices</p>	xxxx

Bits	Name	Description	Reset
[28:25]	TRSH	<p>Transport register slices, horizontal.</p> <p><b>0b0000</b> No register slices</p> <p><b>0b0001</b> One register slice</p> <p><b>0b0010</b> Two register slices</p> <p><b>0b0011</b> Three register slices</p> <p><b>0b0100</b> Four register slices</p> <p><b>0b0101</b> Five register slices</p> <p><b>0b0110</b> Six register slices</p> <p><b>0b0111</b> Seven register slices</p> <p><b>0b1000</b> Eight register slices</p>	xxxx
[24]	RAZ	Reserved	RAZ
[23]	PP	<p>Peripheral port presence.</p> <p><b>0b0</b> No peripheral port present</p> <p><b>0b1</b> Peripheral port present</p>	x
[22]	PPW	<p>Peripheral port width.</p> <p><b>0b0</b> 64 bit data width</p> <p><b>0b1</b> 256 bit data width</p>	x
[21:20]	ACP	<p>ACP interface presence.</p> <p><b>0b00</b> No ACP interface present</p> <p><b>0b01</b> One ACP interface present</p> <p><b>0b10</b> Two ACP interface present</p>	xx
[19]	ACPW	<p>ACP interface width.</p> <p><b>0b0</b> 128 bit data width</p> <p><b>0b1</b> 256 bit data width</p>	x

Bits	Name	Description	Reset
[18]	RAZ	Reserved	RAZ
[17]	RQT	Requester bus interface type.  <b>0b0</b> AXI interface  <b>0b1</b> CHI interface	x
[16:15]	NUMRQT	Number of Requester interfaces.  <b>0b00</b> One requester  <b>0b01</b> Two requesters  <b>0b10</b> Three requesters  <b>0b11</b> Four requesters	xx
[14]	ECC	SCU-L3 ECC configuration.  <b>0b0</b> SCU-L3 is configured with no ECC  <b>0b1</b> SCU-L3 is configured with ECC	x
[13]	RDSLC	L3 data RAM read register slice.  <b>0b0</b> No register slice present  <b>0b1</b> Register slice present	x
[12]	RDLAT	L3 Data RAM read latency.  <b>0b0</b> Two cycle output delay from L3 data RAMs  <b>0b1</b> Three cycle output delay from L3 data RAMs	x
[11:10]	WRLAT	L3 Data RAM write latency.  <b>0b00</b> One cycle input delay from L3 data RAMs  <b>0b01</b> Two cycle input delay from L3 data RAMs  <b>0b10</b> Two cycle input delay plus a one cycle hold	xx
[9]	L3	L3 cache presence.  <b>0b0</b> No L3 cache present  <b>0b1</b> 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	<p>Number of cores present in the cluster.</p> <p><b>0b0000</b> One core</p> <p><b>0b0001</b> Two cores</p> <p><b>0b0010</b> Three cores</p> <p><b>0b0011</b> Four cores</p> <p><b>0b0100</b> Five cores</p> <p><b>0b0101</b> Six cores</p> <p><b>0b0110</b> Seven cores</p> <p><b>0b0111</b> Eight cores</p> <p><b>0b1000</b> Nine core</p> <p><b>0b1001</b> Ten cores</p> <p><b>0b1010</b> Eleven cores</p> <p><b>0b1011</b> Twelve cores</p> <p><b>0b1100</b> Thirteen cores</p> <p><b>0b1101</b> Fourteen cores</p>	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

AArch64 register IMP\_CLUSTERIDR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.1 CLUSTERIDR, Cluster Main Revision Register](#) on page 397 bits [63:0].

### 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	0000	0001
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-2: AArch64\_imp\_clusteridr\_el1 bit assignments

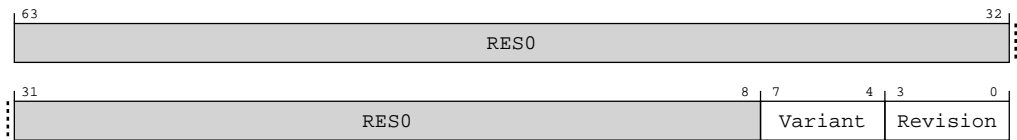


Table A-5: IMP\_CLUSTERIDR\_EL1 bit descriptions

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 rpxy description of the product revision status.  <b>0b0000</b> Cluster major revision 0.	0b0000
[3:0]	Revision	Indicates the minor revision number of the DSU. This is the minor revision number y in the py part of the rpxy description of the product revision status.  <b>0b0000</b> Cluster minor revision 0.  <b>0b0001</b> Cluster minor revision 1.	0b0001

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;
elseif PSTATE.EL == EL1 then
```

```

if EL2Enabled() && HCR_EL2.TIDCP == '1' then
    AArch64.SystemAccessTrap(EL2, 0x18);
else
    return IMP_CLUSTERIDR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERIDR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERIDR_EL1;

```

MSR S3\_0\_C15\_C3\_1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERIDR_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    IMP_CLUSTERIDR_EL1 = X[t];
elseif 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

AArch64 register IMP\_CLUSTERREVIDR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.2 CLUSTERREVIDR, Cluster ECO ID Register](#) on page 398 bits [63:0].

#### Attributes

##### Width

64

##### Functional group

Generic System Control

##### Access type

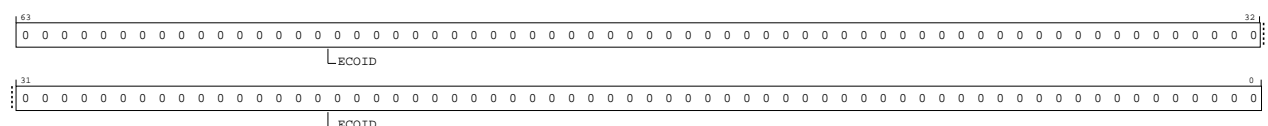
See bit descriptions

##### 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**



[illegible]

MRS <Xt>, S3\_0\_C15\_C3\_2

MSR S3\_0\_C15\_C3\_2, <Xt>

MRS <Xt>, S3\_0\_C15\_C3\_2

MSR S3\_0\_C15\_C3\_2, &lt;Xt&gt;

Page 272 of 1012



### 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

AArch64 register IMP\_CLUSTERACTLR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.10 CLUSTERACTLR, Cluster Auxiliary Control Register](#) on page 415 bits [63:0].

#### 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-4: AArch64\_imp\_clusteractlr\_el1 bit assignments

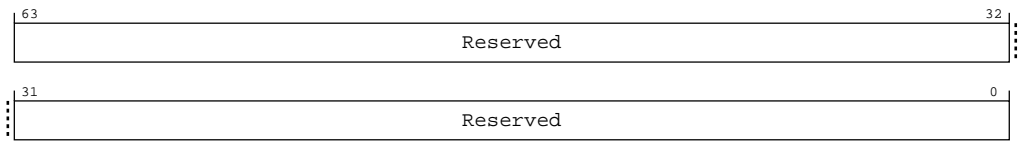


Table A-11: IMP\_CLUSTERACTLR\_EL1 bit descriptions

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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERACTLR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERACTLR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERACTLR_EL1;

```

MSR S3\_0\_C15\_C3\_3, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.ACTLREN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.ACTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.ACTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERACTLR_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.ACTLREN == '0' then
        UNDEFINED;
    elseif ACTLR_EL3.ACTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERACTLR_EL1 = X[t];
elseif 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

AArch64 register IMP\_CLUSTERECTLR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.11 CLUSTERECTLR, Cluster Extended Control Register](#) on page 417 bits [63:0].

### 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0



Note

Where the reset reads xxxx, see individual bits.

### Bit descriptions

Figure A-5: AArch64\_imp\_clusterectlr\_el1 bit assignments

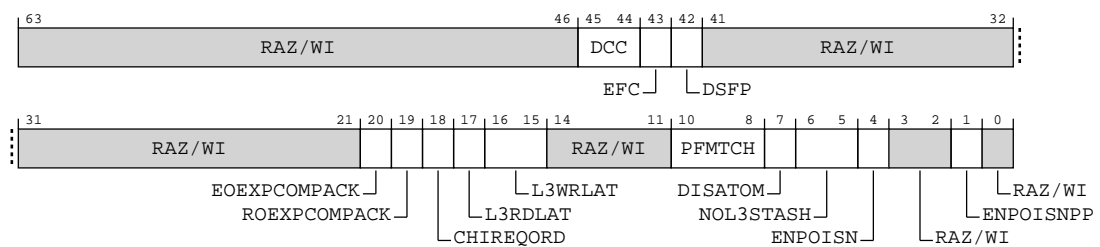


Table A-14: IMP\_CLUSTERECTLR\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:46]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[45:44]	DCC	<p>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.</p> <p><b>0b00</b></p> <p>Disables sending data when clean cachelines are evicted.</p> <p><b>0b01</b></p> <p>Enables sending WriteEvictFull transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data.</p> <p><b>0b10</b></p> <p>Enables sending WriteEvictOrEvict transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data.</p> <p><b>0b11</b></p> <p>Enables sending WriteEvictOrEvict transactions when Unique Clean or Shared Clean cachelines are evicted. This is the reset value.</p>	0b11
[43]	EFC	<p>Eviction flush control. Controls whether hardware cache flushes and DC CISCW instructions send data when evicting clean cachelines on the CHI interface.</p> <p><b>0b0</b></p> <p>Disables sending data when hardware cache flushes or DC CISCW instructions evict a clean cacheline. Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP). This is the reset value.</p> <p><b>0b1</b></p> <p>Sending of data when hardware cache flushes or DC CISCW instructions evict clean cachelines is controlled by Downstream Cache Control (DCC). Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP).</p>	0b0
[42]	DSFP	<p>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.</p> <p><b>0b0</b></p> <p>Disables sending Evict transactions when clean cachelines are evicted without data.</p> <p><b>0b1</b></p> <p>Enables sending of Evict transactions when clean cachelines are evicted without data. This is the reset value.</p>	0b1
[41:21]	RAZ/WI	Reserved	RAZ/WI
[20]	EOEXPCOMPACT	<p>Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Endpoint Order transactions to the system</p> <p><b>0b0</b></p> <p>CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=0</p> <p><b>0b1</b></p> <p>CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=1</p>	0b0
[19]	ROEXPCOMPACT	<p>Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Request Order transactions to the system</p> <p><b>0b0</b></p> <p>CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=0</p> <p><b>0b1</b></p> <p>CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=1</p>	0b0

Bits	Name	Description	Reset
[18]	CHIREQORD	<p>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.</p> <p><b>0b0</b> Disables sending Request Order on the CHI interface to the system. Will send No Order instead.</p> <p><b>0b1</b> Enables sending Request Order on the CHI interface to the system for Non-snoopable transactions.</p>	0b0
[17]	L3RDLAT	<p>L3 data RAM read (output) latency.</p> <p><b>0b0</b> The L3 data RAM output latency is 2 cycles.</p> <p><b>0b1</b> The L3 data RAM output latency is 3 cycles.</p>	x <sup>2</sup>
[16:15]	L3WRLAT	<p>L3 data RAM write (input) latency.</p> <p><b>0b00</b> The L3 data RAM input latency is 1 cycle with an additional hold cycle.</p> <p><b>0b01</b> The L3 data RAM input latency is 2 cycles without an additional hold cycle.</p> <p><b>0b10</b> 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.</p>	xx <sup>3</sup>
[14:11]	RAZ/WI	Reserved	RAZ/WI
[10:8]	PFMTCH	<p>Prefetch matching delay. Controls the amount of time a prefetch waits for a possible match with a later read. Encoded as powers of 2, from 1-128.</p> <p><b>0b000</b> Wait for 1 cycle.</p> <p><b>0b001</b> Wait for 2 cycles.</p> <p><b>0b010</b> Wait for 4 cycles.</p> <p><b>0b011</b> Wait for 8 cycles.</p> <p><b>0b100</b> Wait for 16 cycles.</p> <p><b>0b101</b> Wait for 32 cycles.</p> <p><b>0b110</b> Wait for 64 cycles.</p> <p><b>0b111</b> Wait for 128 cycles.</p>	0b101

<sup>2</sup> This field resets to the value of the L3\_DATA\_RD\_LATENCY configuration parameter.

<sup>3</sup> This field resets to the value of the L3\_DATA\_WR\_LATENCY configuration parameter.

Bits	Name	Description	Reset
[7]	DISATOM	<p>Disable cacheable shareable atomics being sent to the interconnect.</p> <p><b>0b0</b></p> <p>Cacheable shareable atomics will be sent to the interconnect if the BROADCASTATOMIC pin is set.</p> <p><b>0b1</b></p> <p>Cacheable shareable atomics will be handled inside the cluster.</p>	0b0
[6:5]	NOL3STASH	<p>CPU StashOnce request behaviour when L3 is not present or powered down.</p> <p><b>0b00</b></p> <p>Stashes are sent out to the interconnect, if supported.</p> <p><b>0b01</b></p> <p>Normal read request sent to interconnect.</p> <p><b>0b10</b></p> <p>StashOnce has no effect.</p>	0b10
[4]	ENPOISN	<p>Interconnect data poisoning support for the CHI Requester(s). This bit is ignored for AXI configurations, which never support poisoning.</p> <p><b>0b0</b></p> <p>Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop.</p> <p><b>0b1</b></p> <p>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.</p>	0b1
[3:2]	RAZ/WI	Reserved	RAZ/WI
[1]	ENPOISNPP	<p>Interconnect data poisoning support for the CHI Peripheral Port. This bit is ignored for AXI configurations, which never support poisoning.</p> <p><b>0b0</b></p> <p>Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop.</p> <p><b>0b1</b></p> <p>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.</p>	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

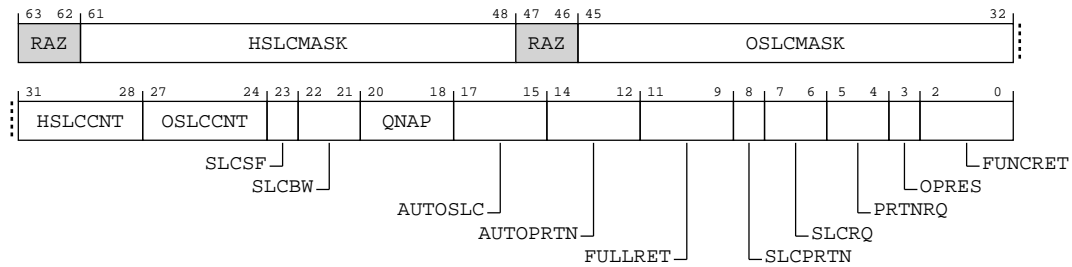
See bit descriptions

### Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1010 0100 0000 0001 0111  
0000

## Bit descriptions

**Figure A-6: AArch64\_imp\_clusterpwrtlr\_el1 bit assignments**



**Table A-17: IMP\_CLUSTERPWRTLR\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:62]	RAZ	Reserved	RAZ
[61:48]	HSLCMASK	Half slice core mask. This contains one bit per core, and if that bit is set then the core is included in the count that is compared with the HSLCCNT field. Bits above the number of cores implemented are <b>RAZ/WI</b> .	0b00000000000000
[47:46]	RAZ	Reserved	RAZ
[45:32]	OSLCMASK	One slice core mask. This contains one bit per core, and if that bit is set then the core is included in the count that is compared with the OSLCCNT field. Bits above the number of cores implemented are <b>RAZ/WI</b> .	0b00000000000000
[31:28]	HSLCCNT	Half slice core count. When AUTOSLC is non-zero, if the number of cores powered on in the group selected by the HSLCMASK field is greater than this field then it will prevent the transition from ALL SLICE to HALF SLICE mode, or cause a transition from HALF SLICE to ALL SLICE. Note that a core that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this count.	0b0000
[27:24]	OSLCCNT	One slice core count. When AUTOSLC is non-zero, if the number of cores powered on in the group selected by the OSLCMASK field is greater than this field then it will prevent the transition from ALL SLICE or HALF SLICE to ONE SLICE mode, or cause a transition from ONE SLICE to HALF SLICE. Note that a core that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this count.	0b0000



Bits	Name	Description	Reset
[23]	SLCSF	<p>Include snoop filter capacity in the AUTOSLC decision. If powering down slices would reduce the amount of snoop filter below the amount needed for the currently powered on cores then this will prevent the slice powerdown. Note that a core that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this calculation.</p> <p><b>0b0</b> Ignore the snoop filter in AUTOSLC decisions</p> <p><b>0b1</b> AUTOSLC will ensure enough slices are powered on for the currently powered on cores</p>	0b1
[22:21]	SLCBW	<p>Include slice bandwidth in the AUTOSLC decision. Prevent the slice powerdown if powering down slices would reduce the available bandwidth below the currently used bandwidth. Power up more slices if the slice bandwidth is close to saturating with the current number of slices.</p> <p><b>0b00</b> Ignore the slice bandwidth in AUTOSLC decisions</p> <p><b>0b01</b> AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 3% of the AUTOSLC time period</p> <p><b>0b11</b> AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 12% of the AUTOSLC time period</p>	0b01
[20:18]	QNAP	<p>Control L3 data RAM quick nap enable time</p> <p><b>0b000</b> Quick nap disabled</p> <p><b>0b001</b> Enabled with 8 cycle timeout</p> <p><b>0b010</b> Enabled with 12 cycle timeout</p> <p><b>0b011</b> Enabled with 16 cycle timeout</p> <p><b>0b100</b> Enabled with 24 cycle timeout</p> <p><b>0b101</b> Enabled with 32 cycle timeout</p> <p><b>0b110</b> Enabled with 64 cycle timeout</p> <p><b>0b111</b> Enabled with 128 cycle timeout</p>	0b001

Bits	Name	Description	Reset
[17:15]	AUTOSLC	<p>Enable automatic slice 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.</p> <p><b>0b000</b> Disabled</p> <p><b>0b001</b> 524,288 architectural timer ticks, time period of 524us</p> <p><b>0b010</b> 1,048,576 architectural timer ticks, time period of 1ms</p> <p><b>0b011</b> 2,097,152 architectural timer ticks, time period of 2.1ms</p> <p><b>0b100</b> 4,194,304 architectural timer ticks, time period of 4.2ms</p> <p><b>0b101</b> 8,388,608 architectural timer ticks, time period of 8.4ms</p> <p><b>0b110</b> 16,777,216 architectural timer ticks, time period of 16.8ms</p> <p><b>0b111</b> 33,554,432 architectural timer ticks, time period of 33.6ms</p>	0b000
[14:12]	AUTOPRTN	<p>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.</p> <p><b>0b000</b> Disabled</p> <p><b>0b001</b> 524,288 architectural timer ticks, time period of 524us</p> <p><b>0b010</b> 1,048,576 architectural timer ticks, time period of 1ms</p> <p><b>0b011</b> 2,097,152 architectural timer ticks, time period of 2.1ms</p> <p><b>0b100</b> 4,194,304 architectural timer ticks, time period of 4.2ms</p> <p><b>0b101</b> 8,388,608 architectural timer ticks, time period of 8.4ms</p> <p><b>0b110</b> 16,777,216 architectural timer ticks, time period of 16.8ms</p> <p><b>0b111</b> 33,554,432 architectural timer ticks, time period of 33.6ms</p>	0b000

Bits	Name	Description	Reset
[11:9]	FULLRET	<p>Enable the FULL_RET slice powerdown mode and time period. Note that while this would typically be a longer period than the FUNC_RET field, to allow entry into FUNC_RET first, but if it is shorter then FULL_RET will be entered directly rather than via FUNC_RET.</p> <p><b>0b000</b> Disabled</p> <p><b>0b001</b> 128 architectural timer ticks, time period of 128ns</p> <p><b>0b010</b> 512 architectural timer ticks, time period of 512ns</p> <p><b>0b011</b> 2,048 architectural timer ticks, time period of 2us</p> <p><b>0b100</b> 4,096 architectural timer ticks, time period of 4.1us</p> <p><b>0b101</b> 8,192 architectural timer ticks, time period of 8.2us</p> <p><b>0b110</b> 16,384 architectural timer ticks, time period of 16.4us</p> <p><b>0b111</b> 32,768 architectural timer ticks, time period of 32.8us</p>	0b000
[8]	SLCPRTN	<p>The AUTOSLC logic should make slice operating mode decisions considering the AUTOPRTN status. If enabled, AUTOSLC will not power down more slices if AUTOPRTN is keeping all the L3 cache portions powered on. The AUTOPRTN mechanism can also request to power up more slices if it determines that more cache is needed and all the L3 cache portions are powered on</p> <p><b>0b0</b> AUTOSLC decisions should ignore the AUTOPRTN status</p> <p><b>0b1</b> AUTOSLC decisions should include the AUTOPRTN status</p>	0b1
[7:6]	SLCRQ	<p>Cache slice power request. These bits are passed to the PPU as an advisory request for which slices to power. Note that when the AUTOSLC field is not 0b000, higher modes might be requested by the AUTOSLC mechanism and this field acts as a minimum operating mode.</p> <p><b>0b00</b> Request that one L3 cache slice is powered on</p> <p><b>0b01</b> Request that all L3 cache slices are powered on</p> <p><b>0b10</b> Request that half the L3 cache slices are powered on</p>	0b01

Bits	Name	Description	Reset
[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.  <b>0b00</b> Request that none of the L3 cache portions in each slice is powered on  <b>0b01</b> Request that half of the L3 cache portions in each slice are powered on  <b>0b11</b> Request that both of the L3 cache portions in each slice are powered on	0b11
[3]	OPRES	Restore previous operating mode after cluster power off  <b>0b0</b> Operating mode defaults to ALL RAM ALL SLICE when powering on the cluster  <b>0b1</b> Enable restoring of previous operating mode	0b0
[2:0]	FUNCRET	L3 Data RAM retention control.  <b>0b000</b> Disable the retention circuit.  <b>0b001</b> 128 architectural timer ticks, time period of 128ns minimum delay before retention  <b>0b010</b> 512 architectural timer ticks, time period of 512ns minimum delay before retention  <b>0b011</b> 2,048 architectural timer ticks, time period of 2us minimum delay before retention  <b>0b100</b> 4,096 architectural timer ticks, time period of 4.1us minimum delay before retention  <b>0b101</b> 8,192 architectural timer ticks, time period of 8.2us minimum delay before retention  <b>0b110</b> 16,384 architectural timer ticks, time period of 16.4us minimum delay before retention  <b>0b111</b> 32,768 architectural timer ticks, time period of 32.8us 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\_clusterpwrn\_el1 bit assignments

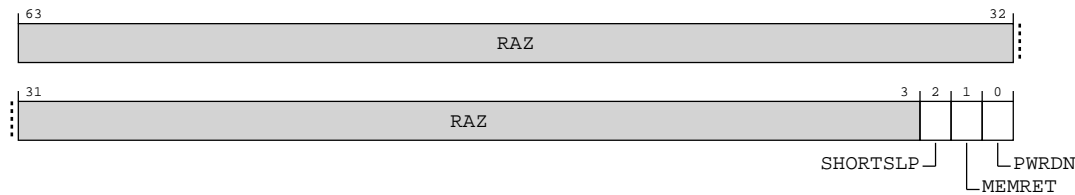


Table A-20: IMP\_CLUSTERPWRDN\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:3]	RAZ	Reserved	RAZ
[2]	SHORTSLP	Indicates that the core thinks it will only be powered off for a short period before powering on again. This can be used by the automated slice powerdown logic.	0b0
[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	xxxx	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-8: AArch64\_imp\_clusterpwrstat\_el1 bit assignments

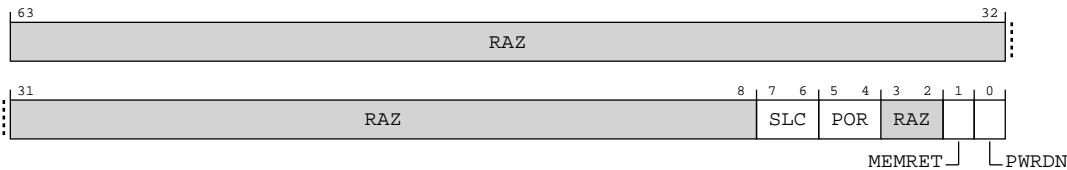


Table A-23: IMP\_CLUSTERPWRSTAT\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:8]	RAZ	Reserved	RAZ
[7:6]	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 bits [7:6] of the IMP_CLUSTERPWRCTLR_EL1 has taken effect.	xx
[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, &lt;Xt&gt;

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b111

## Accessibility

MRS &lt;Xt&gt;, S3\_0\_C15\_C3\_7

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPWRSTAT_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPWRSTAT_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPWRSTAT_EL1;

```

MSR S3\_0\_C15\_C3\_7, &lt;Xt&gt;

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPWRSTAT_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elseif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPWRSTAT_EL1 = X[t];
elseif 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

AArch64 register IMP\_CLUSTERL3DNTH0\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.4 CLUSTERL3DNTH0, Cluster L3 Downsize Threshold0 Register](#) on page 404 bits [63:0].

#### 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

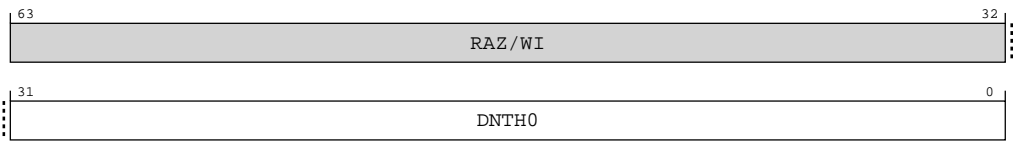


Table A-26: IMP\_CLUSTERL3DNTH0\_EL1 bit descriptions

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

AArch64 register IMP\_CLUSTERL3DNTH1\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.5 CLUSTERL3DNTH1, Cluster L3 Downsize Threshold1 Register](#) on page 406 bits [63:0].

#### 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

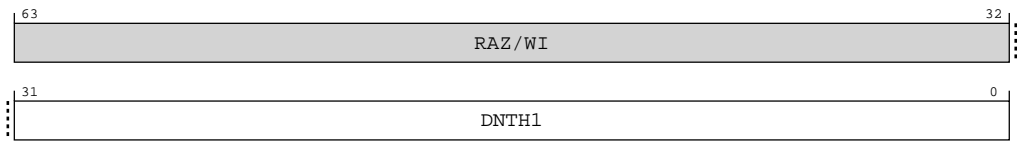


Table A-29: IMP\_CLUSTERL3DNTH1\_EL1 bit descriptions

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

AArch64 register IMP\_CLUSTERL3UPTH0\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.6 CLUSTERL3UPTH0, Cluster L3 Upsize Threshold0 Register](#) on page 407 bits [63:0].

#### 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

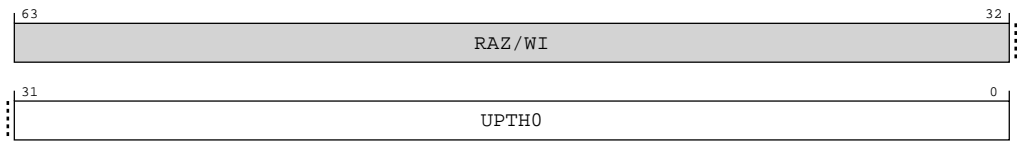


Table A-32: IMP\_CLUSTERL3UPTH0\_EL1 bit descriptions

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

AArch64 register IMP\_CLUSTERL3UPTH1\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.7 CLUSTERL3UPTH1, Cluster L3 Upsize Threshold1 Register](#) on page 408 bits [63:0].

#### 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

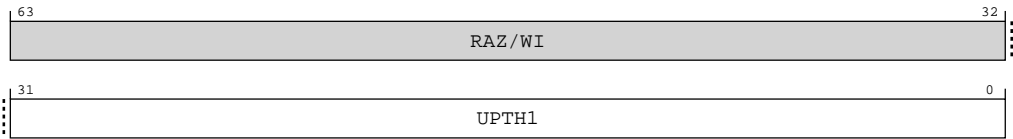


Table A-35: IMP\_CLUSTERL3UPTH1\_EL1 bit descriptions

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

AArch64 register IMP\_CLUSTERBUSQOS\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.8 CLUSTERBUSQOS, Cluster Bus QoS Control Register](#) on page 409 bits [63:0].

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 1110 1110 1011 1110

Bit descriptions

Figure A-13: AArch64\_imp\_clusterbusqos\_el1 bit assignments

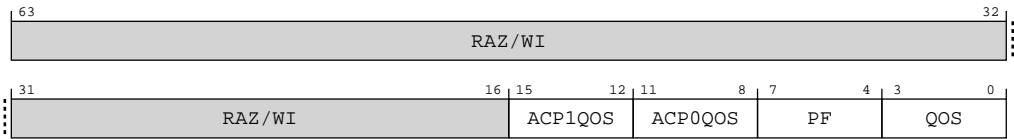


Table A-38: IMP\_CLUSTERBUSQOS\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:16]	RAZ/WI	Reserved	RAZ/WI
[15:12]	ACP1QOS	Valid driven on the CHI bus QoS field for acp 1 accesses.	0b1110
[11:8]	ACP0QOS	Valid driven on the CHI bus QoS field for acp 0 accesses.	0b1110
[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

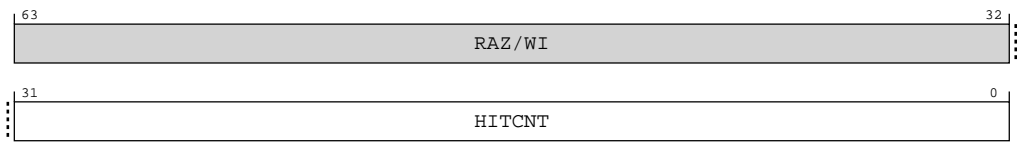


Table A-41: IMP\_CLUSTERL3HIT\_EL1 bit descriptions

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

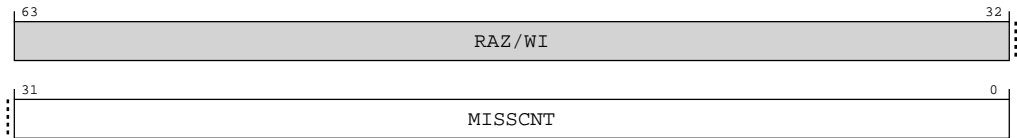


Table A-44: IMP\_CLUSTERL3MISS\_EL1 bit descriptions

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;
elseif 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	xxxx	0000	0000	0000	0000	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-16: AArch64\_imp\_clusterppstart\_el1 bit assignments

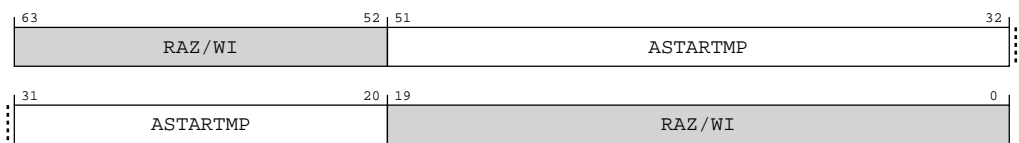


Table A-47: IMP\_CLUSTERPPSTART\_EL1 bit descriptions

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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPPSTART_EL1;
elseif PSTATE.EL == EL2 then
```



```
return IMP_CLUSTERPPSTART_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPPSTART_EL1;
```

MSR S3\_0\_C15\_C9\_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPPSTART_EL1 = X[t];
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
            UNDEFINED;
        elseif ACTLR_EL3.ECTLREN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
            else
                IMP_CLUSTERPPSTART_EL1 = X[t];
    elseif 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 0000 0000 0000 0000 0000
```



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-17: AArch64\_imp\_clusterppend\_el1 bit assignments

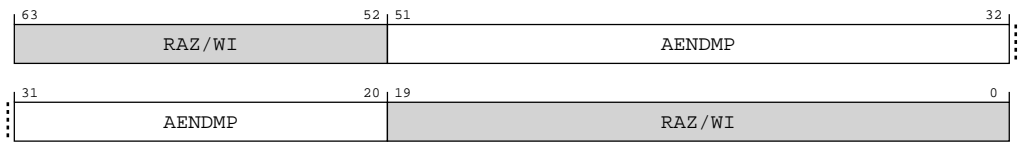


Table A-50: IMP\_CLUSTERPPEND\_EL1 bit descriptions

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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPPEND_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPPEND_EL1;
```

```
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPPEND_EL1;
```

MSR S3\_0\_C15\_C9\_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPEND_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elseif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPEND_EL1 = X[t];
elseif 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

AArch64 register IMP\_CLUSTERCFR2\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.12 CLUSTERCFR2, Cluster Configuration Register 2](#) on page 421 bits [63:0].

### Attributes

#### Width

64

#### Functional group

Generic System Control

#### Access type

See bit descriptions

Reset value

0000	0000	0000	0000	0000	00xx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-18: AArch64\_imp\_clustercfr2\_el1 bit assignments

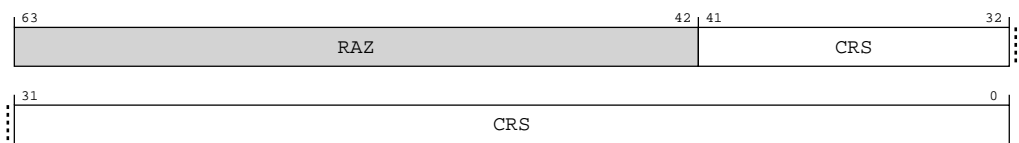


Table A-53: IMP\_CLUSTERCFR2\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:42]	RAZ	Reserved	RAZ
[41:0]	CRS	Core register slices. Each three bits represents a core, with [2:0] for core 0 up to [41:39] for core 13.  0b000 No register slices  0b001 One register slice  0b00010 Two register slices  0b00011 Three register slices  0b00100 Four register slices  0b00101 Five register slices  0b00110 Six register slices  0b00111 Seven register slices	42 { 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, &lt;Xt&gt;

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b010

### Accessibility

MRS &lt;Xt&gt;, 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, &lt;Xt&gt;

```

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\_CLUSTERL3UPTH2\_EL1, Cluster L3 Upsize Threshold2 Register

This register is intended for use in algorithms for determining when to power up slices.

### Configurations

AArch64 register IMP\_CLUSTERL3UPTH2\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.1.15 CLUSTERL3UPTH2, Cluster L3 Upsize Threshold2 Register](#) on page 425 bits [63:0].

### 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\_clusterl3upth2\_el1 bit assignments

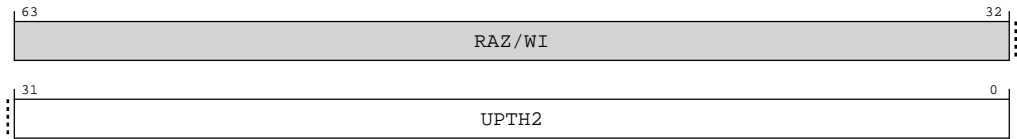


Table A-56: IMP\_CLUSTERL3UPTH2\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	UPTH2	If all the L3 ways are powered, but not all of the slices are powered, and the cache miss bandwidth rises above this threshold then the number of slices is upsized. 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\_C9\_3

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b011

MSR S3\_0\_C15\_C9\_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b011

Accessibility  
MRS <Xt>, S3\_0\_C15\_C9\_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERL3UPTH2_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERL3UPTH2_EL1;
elseif PSTATE.EL == EL3 then
```

```
return IMP_CLUSTERL3UPTH2_EL1;
```

MSR S3\_0\_C15\_C9\_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
        IMP_CLUSTERL3UPTH2_EL1 = X[t];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERL3UPTH2_EL1 = X[t];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3UPTH2_EL1 = X[t];
```

A.1.20 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

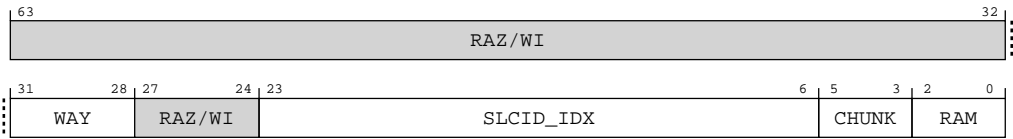
See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-20: AArch64\_imp\_clustercdbg\_el3 bit assignments



**Table A-59: IMP\_CLUSTERDBG\_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
[23:6]	SLCID_IDX	<p>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.</p> <p>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.</p> <p>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.</p> <p>For a Tag RAM or Data RAM access this field will encode as {0, SLICE_ID_W, TagRAM_IDX_W}</p> <p>For a Snoop Filter RAM access this field will encode as {0, SLICE_ID_W, SFRAM_IDX_W}.</p>	0b00000000000000000000
[5:3]	CHUNK	<p>Select of 64-bit data chunk to read from 512-bit Data RAM cache line. Only used when accessing Data RAM data.</p> <p><b>0b000</b> Data[63:0]</p> <p><b>0b001</b> Data[127:64]</p> <p><b>0b010</b> Data[191:128]</p> <p><b>0b011</b> Data[255:192]</p> <p><b>0b100</b> Data[319:256]</p> <p><b>0b101</b> Data[383:320]</p> <p><b>0b110</b> Data[447:384]</p> <p><b>0b111</b> Data[511:448]</p>	0b000



Bits	Name	Description	Reset
[2:0]	RAM	<p>RAM to be accessed. All other values are reserved.</p> <p><b>0b001</b> Snoop Filter RAM</p> <p><b>0b010</b> Tag RAM</p> <p><b>0b011</b> Data RAM - accessing cacheline data</p> <p><b>0b111</b> Data RAM - accessing cacheline MTE tags</p>	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_CLUSTERDBG_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_CLUSTERDBG_EL3 = X[t];

```

### A.1.21 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

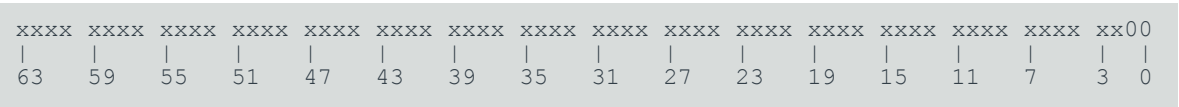
##### Functional group

Generic System Control

##### Access type

See bit descriptions

##### Reset value



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-21: AArch64\_imp\_clusterpmmdcr\_el3 bit assignments

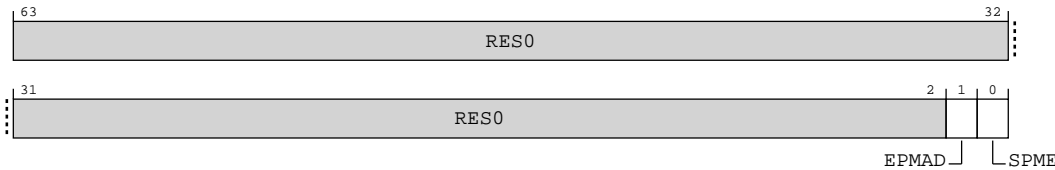


Table A-62: IMP\_CLUSTERPMMDCR\_EL3 bit descriptions

Bits	Name	Description	Reset
[63:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1]	EPMAD	External Performance Monitors Non-secure Access Disable. Controls Non-secure access to Performance Monitor registers by an external debugger.  <b>0b0</b> Non-secure access to Performance Monitor registers from external debugger is permitted.  <b>0b1</b> 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.  <b>0b0</b> Event counting prohibited in Secure state.  <b>0b1</b> 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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elseif PSTATE.EL == EL2 then
    UNDEFINED;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMMDCR_EL3;

```

MSR S3\_6\_C15\_C6\_3, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elseif PSTATE.EL == EL2 then
    UNDEFINED;
elseif PSTATE.EL == EL3 then

```

```
IMP_CLUSTERPMMDCR_EL3 = X[t];
```

## A.2 AArch64 performance monitors registers summary

The cluster Performance Monitors registers are accessible either from System register accesses from the cores or from memory-mapped accesses on the utility bus.

The summary table provides an overview of all AArch64 cluster Performance Monitors registers in the DSU-120AE. For more information about a register, click on the register name in the table.



- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table A-65: Performance Monitors registers summary**

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
<a href="#">IMP_CLUSTERPMCR_EL1</a>	3	0	C15	C5	0	See individual bit resets.	64-bit	Performance Monitors Control Register
<a href="#">IMP_CLUSTERPMCNTENSET_EL1</a>	3	0	C15	C5	1	See individual bit resets.	64-bit	Performance Monitors Count Enable Set Register
<a href="#">IMP_CLUSTERPMCNTENCLR_EL1</a>	3	0	C15	C5	2	See individual bit resets.	64-bit	Performance Monitors Count Enable Clear Register
<a href="#">IMP_CLUSTERPMOVSSSET_EL1</a>	3	0	C15	C5	3	See individual bit resets.	64-bit	Performance Monitors Overflow Flag Status Set Register
<a href="#">IMP_CLUSTERPMOVSCCLR_EL1</a>	3	0	C15	C5	4	See individual bit resets.	64-bit	Performance Monitors Overflow Flag Status Clear Register
<a href="#">IMP_CLUSTERPMSELR_EL1</a>	3	0	C15	C5	5	See individual bit resets.	64-bit	Performance Monitors Event Counter Selection Register
<a href="#">IMP_CLUSTERPMINTENSET_EL1</a>	3	0	C15	C5	6	See individual bit resets.	64-bit	Performance Monitors Interrupt Enable Set Register
<a href="#">IMP_CLUSTERPMINTENCLR_EL1</a>	3	0	C15	C5	7	See individual bit resets.	64-bit	Performance Monitors Interrupt Enable Clear Register
<a href="#">IMP_CLUSTERPMCCNTR_EL1</a>	3	0	C15	C6	0	See individual bit resets.	64-bit	Performance Monitors Cycle Count Register
<a href="#">IMP_CLUSTERPMXEVTYPER_EL1</a>	3	0	C15	C6	1	See individual bit resets.	64-bit	Performance Monitors Selected Event Type Register
<a href="#">IMP_CLUSTERPMXVCNTR_EL1</a>	3	0	C15	C6	2	See individual bit resets.	64-bit	Performance Monitors Selected Event Count Register
<a href="#">IMP_CLUSTERPMCEID0_EL1</a>	3	0	C15	C6	4	See individual bit resets.	64-bit	Performance Monitors Common Event Identification Register 0
<a href="#">IMP_CLUSTERPMCEID1_EL1</a>	3	0	C15	C6	5	See individual bit resets.	64-bit	Performance Monitors Common Event Identification Register 1

### 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	xx0x	xxxx	xxx0
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-22: AArch64\_imp\_clusterpmcr\_el1 bit assignments

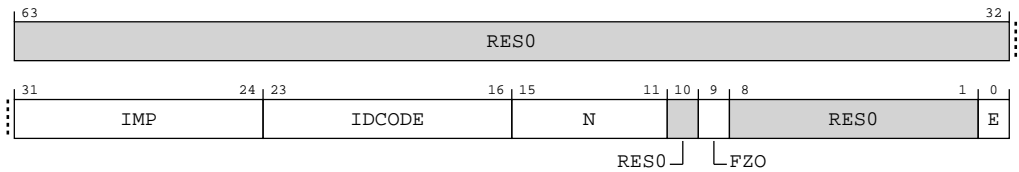


Table A-66: IMP\_CLUSTERPMCR\_EL1 bit descriptions

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 <b>RES0</b> and software must use the AArch64-MIDR_EL1 to identify the PE.  <b>0b00000000</b> Software must use the AArch64-MIDR_EL1 to identify the PE.	8 {x}

Bits	Name	Description	Reset
[23:16]	IDCODE	Identification code. This field is RO with a value of 0x00.  <b>0b00000000</b> 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.  <b>0b00110</b> Indicates that 6 event counters are implemented.  Access to this field is: RO	5{x}
[10]	RES0	Reserved	RES0
[9]	FZO	Freeze on overflow.  <b>0b0</b> Freeze on overflow disabled.  <b>0b1</b> Freeze on overflow enabled.	0b0
[8:1]	RES0	Reserved	RES0
[0]	E	Enable.  <b>0b0</b> All event counters in the range [0..(PMN-1)] are disabled.  <b>0b1</b> All event counters in the range [0..(PMN-1)] 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_CLUSTERPMCRL_EL1;
    endif
elsif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCRL_EL1;
elsif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCRL_EL1;
endif

```

MSR S3\_0\_C15\_C5\_0, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCRL_EL1 = X[t];
    elseif PSTATE.EL == EL2 then
        if ACTLR_EL3.CLUSTERPMUEN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
            else
                IMP_CLUSTERPMCRL_EL1 = X[t];
    elseif PSTATE.EL == EL3 then
        IMP_CLUSTERPMCRL_EL1 = X[t];

```

## A.2.2 IMP\_CLUSTERPMCNTENSET\_EL1, Performance Monitors Count Enable Set Register

Enables 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

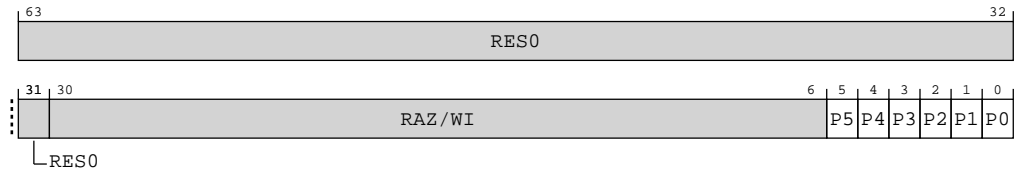
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	x000	0000	0000	0000	0000	0000	00xx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure A-23: AArch64\_imp\_clusterpmcntenset\_el1 bit assignments**



**Table A-69: IMP\_CLUSTERPMCNTENSET\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	Event counter enable bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXVCNTR_EL1.	x
[4]	P4	Event counter enable bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXVCNTR_EL1.	x
[3]	P3	Event counter enable bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXVCNTR_EL1.	x



Bits	Name	Description	Reset
[2]	P2	<p>Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p>	x
[1]	P1	<p>Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p>	x
[0]	P0	<p>Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p>	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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCNTENSET_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCNTENSET_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCNTENSET_EL1;

```

MSR S3\_0\_C15\_C5\_1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCNTENSET_EL1 = X[t];
    elseif 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];
    elseif PSTATE.EL == EL3 then
        IMP_CLUSTERPMCNTENSET_EL1 = X[t];

```

## A.2.3 IMP\_CLUSTERPMCNTENCLR\_EL1, Performance Monitors Count Enable Clear Register

Disables 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	0



Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure A-24: AArch64\_imp\_clusterpmcntenclr\_el1 bit assignments

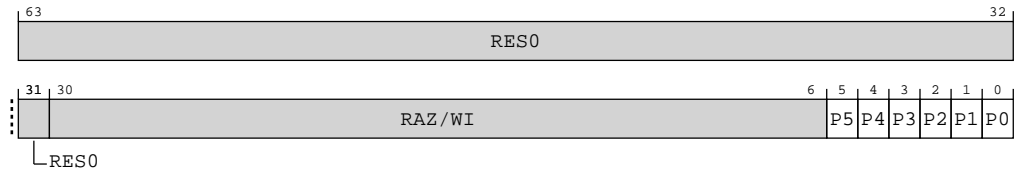


Table A-72: IMP\_CLUSTERPMCNTENCLR\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	Event counter disable bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXVCNTR_EL1.	x
[4]	P4	Event counter disable bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXVCNTR_EL1.	x
[3]	P3	Event counter disable bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXVCNTR_EL1.	x

Bits	Name	Description	Reset
[2]	P2	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVNTR_EL1.	x
[1]	P1	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVNTR_EL1.	x
[0]	P0	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 is disabled. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVNTR_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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCNTENCLR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCNTENCLR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCNTENCLR_EL1;

```

MSR S3\_0\_C15\_C5\_2, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCNTENCLR_EL1 = X[t];
    elseif 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];
    elseif 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 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

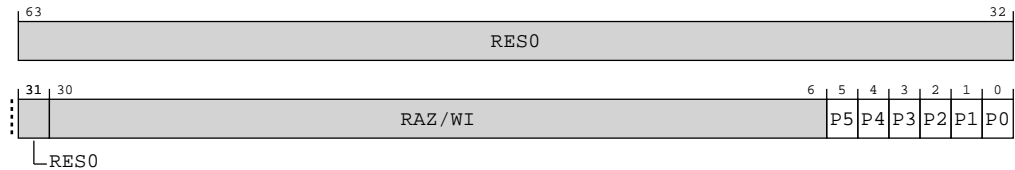
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	x000	0000	0000	0000	0000	0000	00xx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure A-25: AArch64\_imp\_clusterpmovsset\_el1 bit assignments**



**Table A-75: IMP\_CLUSTERPMOVSSET\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXVCNTR_EL1 overflow bit to 1.	x
[4]	P4	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXVCNTR_EL1 overflow bit to 1.	x
[3]	P3	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXVCNTR_EL1.  <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> When read, means that AArch64-IMP_CLUSTERPMXVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXVCNTR_EL1 overflow bit to 1.	x

Bits	Name	Description	Reset
[2]	P2	<p>Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>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.</p>	x
[1]	P1	<p>Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>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.</p>	x
[0]	P0	<p>Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>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.</p>	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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMOVSSSET_EL1 = X[t];
    elseif PSTATE.EL == EL2 then
        if ACTLR_EL3.CLUSTERPMUEN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
            else
                IMP_CLUSTERPMOVSSSET_EL1 = X[t];
    elseif PSTATE.EL == EL3 then
        IMP_CLUSTERPMOVSSSET_EL1 = X[t];

```

## A.2.5 IMP\_CLUSTERPMOVSSCLR\_EL1, Performance Monitors Overflow Flag Status Clear Register

Contains the state of the overflow bit for 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0

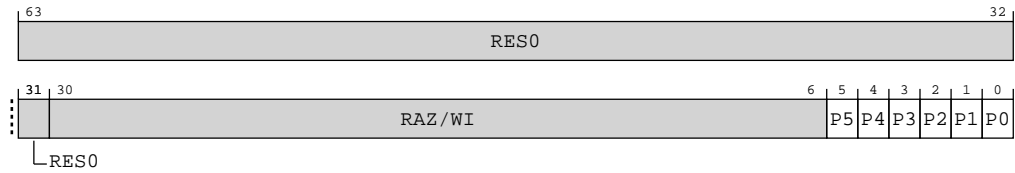




Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure A-26: AArch64\_imp\_clustermovsclr\_el1 bit assignments**



**Table A-78: IMP\_CLUSTERPMOVSLR\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[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 <b>RAZ/WI</b> . N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. <b>0b1</b> 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 <b>RAZ/WI</b> . N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. <b>0b0</b> When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. <b>0b1</b> 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	<p>Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.</p> <p>If N is less than 31, then bits [30:N] are <b>RAZ/WI</b>. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVNTR_EL1 overflow bit to 0.</p>	x
[2]	P2	<p>Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.</p> <p>If N is less than 31, then bits [30:N] are <b>RAZ/WI</b>. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVNTR_EL1 overflow bit to 0.</p>	x
[1]	P1	<p>Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.</p> <p>If N is less than 31, then bits [30:N] are <b>RAZ/WI</b>. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVNTR_EL1 overflow bit to 0.</p>	x
[0]	P0	<p>Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.</p> <p>If N is less than 31, then bits [30:N] are <b>RAZ/WI</b>. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N.</p> <p><b>0b0</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that AArch64-IMP_CLUSTERPMXEVNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVNTR_EL1 overflow bit to 0.</p>	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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMOVSLR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMOVSLR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMOVSLR_EL1;

```

MSR S3\_0\_C15\_C5\_4, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSLR_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSLR_EL1 = X[t];
elseif PSTATE.EL == EL3 then
    IMP_CLUSTERPMOVSLR_EL1 = X[t];

```

## A.2.6 IMP\_CLUSTERPMSELR\_EL1, Performance Monitors Event Counter Selection Register

Selects the current event counter AArch64-IMP\_CLUSTERPMEVCNTR\_EL1.

IMP\_CLUSTERPMSELR\_EL1 is used in conjunction with AArch64-IMP\_CLUSTERPMXEVTYPYPER\_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

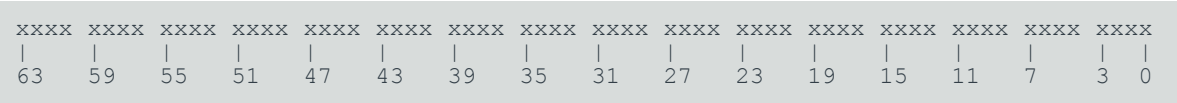
Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-27: AArch64\_imp\_clusterpmselr\_el1 bit assignments

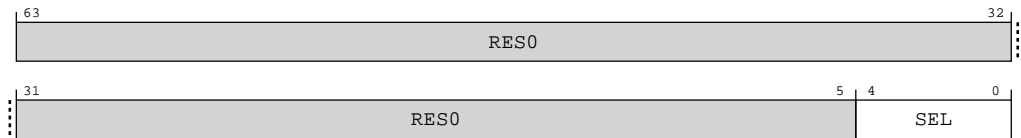


Table A-81: IMP\_CLUSTERPMSELR\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:5]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[4:0]	SEL	<p>Selects event counter, AArch64-IMP_CLUSTERPMXEVNTR_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_CLUSTERPMXEVTYPEPER_EL1 or AArch64-IMP_CLUSTERPMXEVNTR_EL1 occurs.</p> <p>This field can take any value from 0 (0b000000) to (PMCR.N)-1.</p> <ul style="list-style-type: none"> <li>A read or write of AArch64-IMP_CLUSTERPMXEVTYPEPER_EL1 is treated as <b>RAZ/WI</b>.</li> <li>A read or write of AArch64-IMP_CLUSTERPMXEVNTR_EL1 is treated as <b>RAZ/WI</b>.</li> </ul> <p>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:</p> <ul style="list-style-type: none"> <li>Direct reads of this field are treated as <b>RAZ/WI</b>.</li> <li>The results of access to AArch64-IMP_CLUSTERPMXEVTYPEPER_EL1 or AArch64-IMP_CLUSTERPMXEVNTR_EL1 are treated as <b>RAZ/WI</b>.</li> </ul>	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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMSELR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMSELR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMSELR_EL1;

```

MSR S3\_0\_C15\_C5\_5, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);

```

```
elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
    if Halted() && EDSCR.SDD == '1' then
        UNDEFINED;
    else
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMSELR_EL1 = X[t];
elseif 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];
elseif 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 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

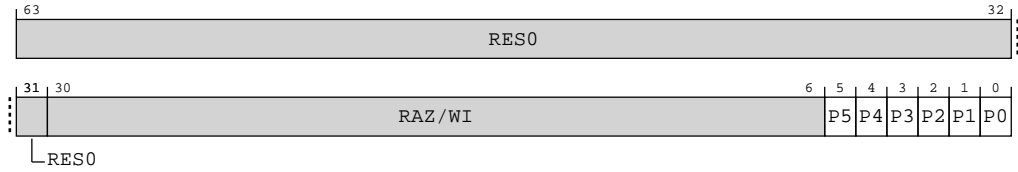
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	x000	0000	0000	0000	0000	0000	00xx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure A-28: AArch64\_imp\_clusterpmintenset\_el1 bit assignments**



**Table A-84: IMP\_CLUSTERPMINTENSET\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:31]	<b>RES0</b>	Reserved	<b>RES0</b>
[30:6]	<b>RAZ/WI</b>	Reserved	<b>RAZ/WI</b>
[5]	<b>P5</b>	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	<b>x</b>
[4]	<b>P4</b>	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	<b>x</b>
[3]	<b>P3</b>	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	<b>x</b>
[2]	<b>P2</b>	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	<b>x</b>

Bits	Name	Description	Reset
[1]	P1	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	x
[0]	P0	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVNTR_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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMINTENSET_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMINTENSET_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMINTENSET_EL1;

```

MSR S3\_0\_C15\_C5\_6, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;

```



```

elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
    AArch64.SystemAccessTrap(EL2, 0x18);
elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
    if Halted() && EDSCR.SDD == '1' then
        UNDEFINED;
    else
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENSET_EL1 = X[t];
elseif 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];
elseif 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 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

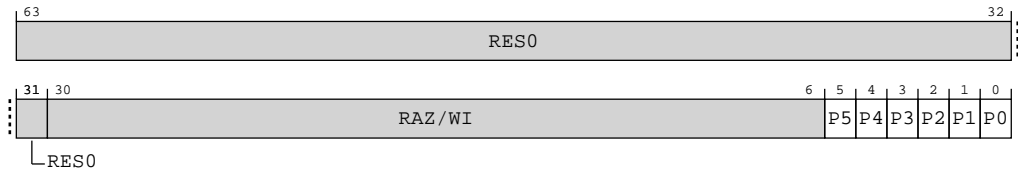
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	x000	0000	0000	0000	0000	0000	00xx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure A-29: AArch64\_imp\_clusterpmintencr\_el1 bit assignments**



**Table A-87: IMP\_CLUSTERPMINTENCLR\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	x
[4]	P4	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	x
[3]	P3	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	x
[2]	P2	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	x

Bits	Name	Description	Reset
[1]	P1	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVNTR_EL1 interrupt request.	x
[0]	P0	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVNTR_EL1.  <b>0b0</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is disabled. When written, has no effect.  <b>0b1</b> When read, means that the AArch64-IMP_CLUSTERPMXEVNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVNTR_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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMINTENCLR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMINTENCLR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMINTENCLR_EL1;

```

MSR S3\_0\_C15\_C5\_7, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;

```

```

elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
    AArch64.SystemAccessTrap(EL2, 0x18);
elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
    if Halted() && EDSCR.SDD == '1' then
        UNDEFINED;
    else
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENCLR_EL1 = X[t];
elseif 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];
elseif 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. **RES0** if not implemented.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

64

#### Functional group

Performance Monitors registers

#### Access type

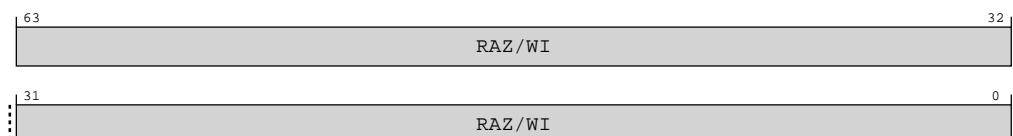
See bit descriptions

#### Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

### Bit descriptions

Figure A-30: AArch64\_imp\_clusterpmccntr\_el1 bit assignments



**Table A-90: IMP\_CLUSTERPMCCNTR\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

### 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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCCNTR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCCNTR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCCNTR_EL1;

```

MSR S3\_0\_C15\_C6\_0, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCCNTR_EL1 = X[t];
elseif 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];

```

```
elseif 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

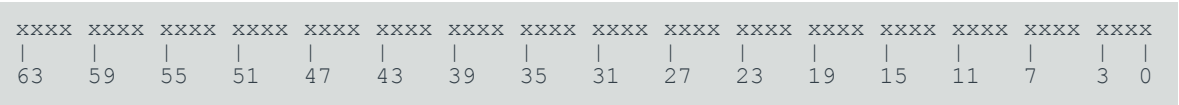
##### Functional group

Performance Monitors registers

##### Access type

See bit descriptions

##### Reset value



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-31: AArch64\_imp\_clusterpmxevtyper\_el1 bit assignments

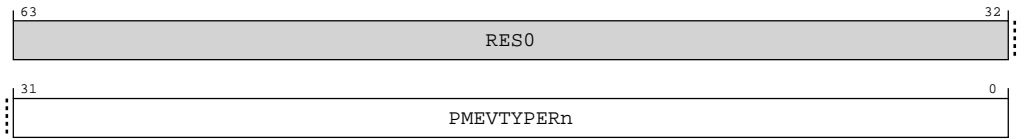


Table A-93: IMP\_CLUSTERPMXEVTYPER\_EL1 bit descriptions

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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMXEVTYPER_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMXEVTYPER_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMXEVTYPER_EL1;

```

MSR S3\_0\_C15\_C6\_1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMXEVTYPER_EL1 = X[t];
elseif 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];
elseif 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	
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Note

Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-32: AArch64\_imp\_clusterpmxevcntr\_el1 bit assignments

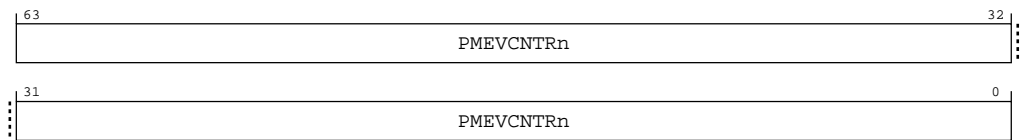


Table A-96: IMP\_CLUSTERPMXEVCNTR\_EL1 bit descriptions

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;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMXVCNTR_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMXVCNTR_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMXVCNTR_EL1;
```

MSR S3\_0\_C15\_C6\_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXVCNTR_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXVCNTR_EL1 = X[t];
elseif PSTATE.EL == EL3 then
    IMP_CLUSTERPMXVCNTR_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

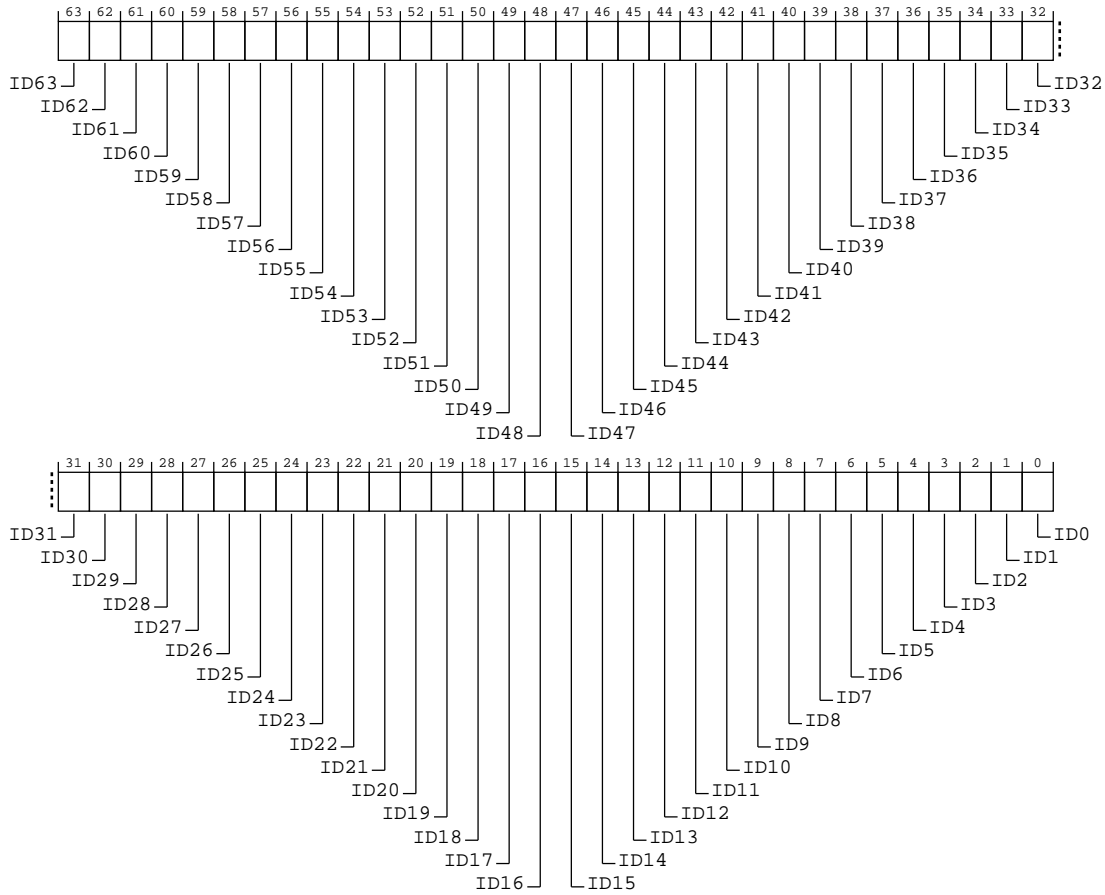
See bit descriptions

#### Reset value

0000 0000 0000 0000 0001 1110 0000 0000 0010 0110 0000 0010 0000 0000 0000  
0000

## Bit descriptions

**Figure A-33: AArch64\_imp\_clusterpmceid0\_el1 bit assignments**



**Table A-99: IMP\_CLUSTERPMCEID0\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63]	ID63	Common event 0x003F implemented. <b>0b0</b> Event 0x003F not implemented.	0b0
[62]	ID62	Common event 0x003E implemented. <b>0b0</b> Event 0x003E not implemented.	0b0
[61]	ID61	Common event 0x003D implemented. <b>0b0</b> Event 0x003D not implemented.	0b0
[60]	ID60	Common event 0x003C implemented. <b>0b0</b> Event 0x003C not implemented.	0b0

Bits	Name	Description	Reset
[59]	ID59	Common event 0x003B implemented. <b>0b0</b> Event 0x003B not implemented.	0b0
[58]	ID58	Common event 0x003A implemented. <b>0b0</b> Event 0x003A not implemented.	0b0
[57]	ID57	Common event 0x0039 implemented. <b>0b0</b> Event 0x0039 not implemented.	0b0
[56]	ID56	Common event 0x0038 implemented. <b>0b0</b> Event 0x0038 not implemented.	0b0
[55]	ID55	Common event 0x0037 implemented. <b>0b0</b> Event 0x0037 not implemented.	0b0
[54]	ID54	Common event 0x0036 implemented. <b>0b0</b> Event 0x0036 not implemented.	0b0
[53]	ID53	Common event 0x0035 implemented. <b>0b0</b> Event 0x0035 not implemented.	0b0
[52]	ID52	Common event 0x0034 implemented. <b>0b0</b> Event 0x0034 not implemented.	0b0
[51]	ID51	Common event 0x0033 implemented. <b>0b0</b> Event 0x0033 not implemented.	0b0
[50]	ID50	Common event 0x0032 implemented. <b>0b0</b> Event 0x0032 not implemented.	0b0
[49]	ID49	Common event 0x0031 implemented. <b>0b0</b> Event 0x0031 not implemented.	0b0
[48]	ID48	Common event 0x0030 implemented. <b>0b0</b> Event 0x0030 not implemented.	0b0
[47]	ID47	Common event 0x002F implemented. <b>0b0</b> Event 0x002F not implemented.	0b0
[46]	ID46	Common event 0x002E implemented. <b>0b0</b> Event 0x002E not implemented.	0b0

Bits	Name	Description	Reset
[45]	ID45	Common event 0x002D implemented. <b>0b0</b> Event 0x002D not implemented.	0b0
[44]	ID44	Common event 0x002C implemented. <b>0b1</b> L3D_CACHE_WB event implemented.	0b1
[43]	ID43	Common event 0x002B implemented. <b>0b1</b> L3D_CACHE event implemented.	0b1
[42]	ID42	Common event 0x002A implemented. <b>0b1</b> L3D_CACHE_REFILL event implemented.	0b1
[41]	ID41	Common event 0x0029 implemented. <b>0b1</b> L3D_CACHE_ALLOCATE event implemented.	0b1
[40]	ID40	Common event 0x0028 implemented. <b>0b0</b> Event 0x0028 not implemented.	0b0
[39]	ID39	Common event 0x0027 implemented. <b>0b0</b> Event 0x0027 not implemented.	0b0
[38]	ID38	Common event 0x0026 implemented. <b>0b0</b> Event 0x0026 not implemented.	0b0
[37]	ID37	Common event 0x0025 implemented. <b>0b0</b> Event 0x0025 not implemented.	0b0
[36]	ID36	Common event 0x0024 implemented. <b>0b0</b> Event 0x0024 not implemented.	0b0
[35]	ID35	Common event 0x0023 implemented. <b>0b0</b> Event 0x0023 not implemented.	0b0
[34]	ID34	Common event 0x0022 implemented. <b>0b0</b> Event 0x0022 not implemented.	0b0
[33]	ID33	Common event 0x0021 implemented. <b>0b0</b> Event 0x0021 not implemented.	0b0
[32]	ID32	Common event 0x0020 implemented. <b>0b0</b> Event 0x0020 not implemented.	0b0

Bits	Name	Description	Reset
[31]	ID31	Common event 0x001F implemented. <b>0b0</b> Event 0x001F not implemented.	0b0
[30]	ID30	Common event 0x001E implemented. <b>0b0</b> CHAIN event implemented.	0b0
[29]	ID29	Common event 0x001D implemented. <b>0b1</b> BUS_CYCLES event implemented.	0b1
[28]	ID28	Common event 0x001C implemented. <b>0b0</b> Event 0x001C not implemented.	0b0
[27]	ID27	Common event 0x001B implemented. <b>0b0</b> Event 0x001B not implemented.	0b0
[26]	ID26	Common event 0x001A implemented. <b>0b1</b> MEMORY_ERROR event implemented.	0b1
[25]	ID25	Common event 0x0019 implemented. <b>0b1</b> BUS_ACCESS event implemented.	0b1
[24]	ID24	Common event 0x0018 implemented. <b>0b0</b> Event 0x0018 not implemented.	0b0
[23]	ID23	Common event 0x0017 implemented. <b>0b0</b> Event 0x0017 not implemented.	0b0
[22]	ID22	Common event 0x0016 implemented. <b>0b0</b> Event 0x0016 not implemented.	0b0
[21]	ID21	Common event 0x0015 implemented. <b>0b0</b> Event 0x0015 not implemented.	0b0
[20]	ID20	Common event 0x0014 implemented. <b>0b0</b> Event 0x0014 not implemented.	0b0
[19]	ID19	Common event 0x0013 implemented. <b>0b0</b> Event 0x0013 not implemented.	0b0
[18]	ID18	Common event 0x0012 implemented. <b>0b0</b> Event 0x0012 not implemented.	0b0

Bits	Name	Description	Reset
[17]	ID17	Common event 0x0011 implemented. <b>0b1</b> CYCLES event implemented.	0b1
[16]	ID16	Common event 0x0010 implemented. <b>0b0</b> Event 0x0010 not implemented.	0b0
[15]	ID15	Common event 0x000F implemented. <b>0b0</b> Event 0x000F not implemented.	0b0
[14]	ID14	Common event 0x000E implemented. <b>0b0</b> Event 0x000E not implemented.	0b0
[13]	ID13	Common event 0x000D implemented. <b>0b0</b> Event 0x000D not implemented.	0b0
[12]	ID12	Common event 0x000C implemented. <b>0b0</b> Event 0x000C not implemented.	0b0
[11]	ID11	Common event 0x000B implemented. <b>0b0</b> Event 0x000B not implemented.	0b0
[10]	ID10	Common event 0x000A implemented. <b>0b0</b> Event 0x000A not implemented.	0b0
[9]	ID9	Common event 0x0009 implemented. <b>0b0</b> Event 0x0009 not implemented.	0b0
[8]	ID8	Common event 0x0008 implemented. <b>0b0</b> Event 0x0008 not implemented.	0b0
[7]	ID7	Common event 0x0007 implemented. <b>0b0</b> Event 0x0007 not implemented.	0b0
[6]	ID6	Common event 0x0006 implemented. <b>0b0</b> Event 0x0006 not implemented.	0b0
[5]	ID5	Common event 0x0005 implemented. <b>0b0</b> Event 0x0005 not implemented.	0b0
[4]	ID4	Common event 0x0004 implemented. <b>0b0</b> Event 0x0004 not implemented.	0b0

Bits	Name	Description	Reset
[3]	ID3	Common event 0x0003 implemented. <b>0b0</b> Event 0x0003 not implemented.	0b0
[2]	ID2	Common event 0x0002 implemented. <b>0b0</b> Event 0x0002 not implemented.	0b0
[1]	ID1	Common event 0x0001 implemented. <b>0b0</b> Event 0x0001 not implemented.	0b0
[0]	ID0	Common event 0x0000 implemented. <b>0b0</b> 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);
    elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCEID0_EL1 = X[t];
    elseif 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];
    elseif 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

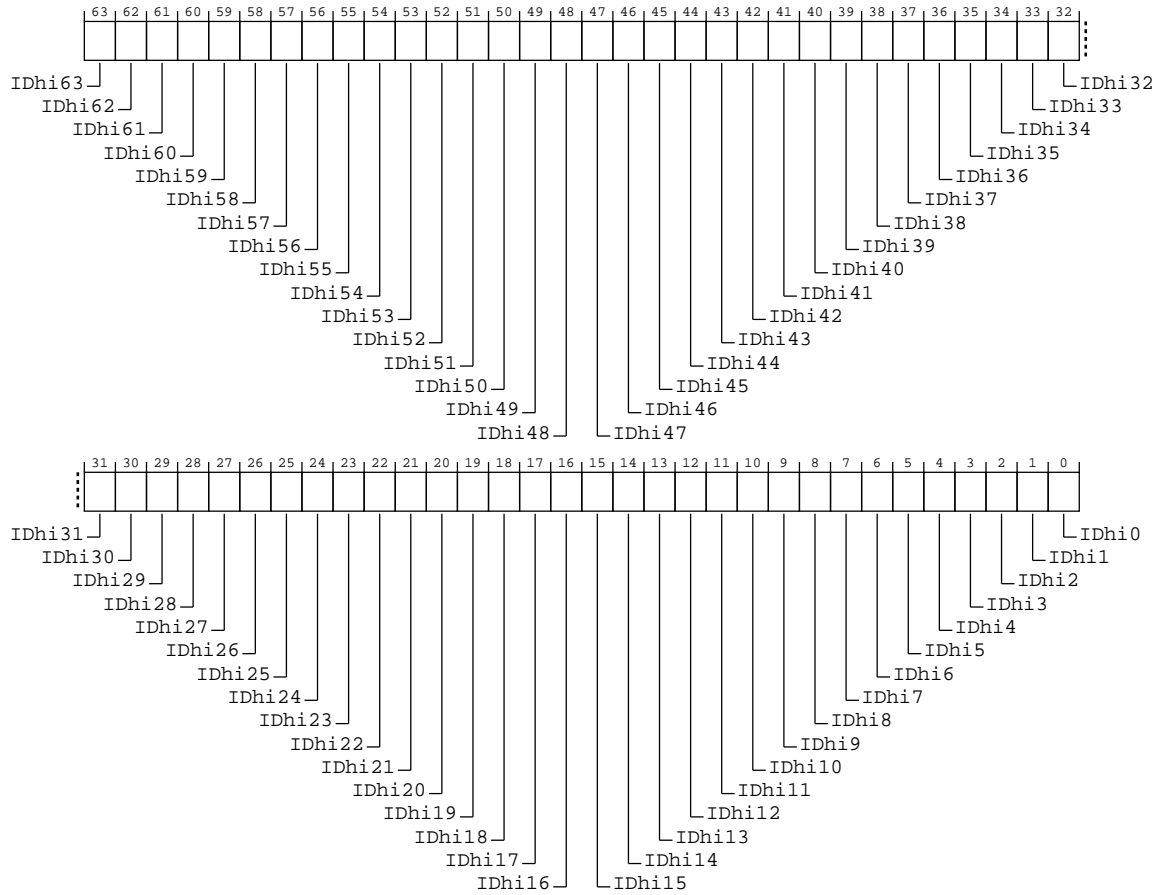
See bit descriptions

#### Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000  
0000

## Bit descriptions

**Figure A-34: AArch64\_imp\_clusterpmceid1\_el1 bit assignments**



**Table A-102: IMP\_CLUSTERPMCEID1\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63]	IDHi63	Common event 0x403F implemented. <b>0b0</b> Event 0x403F not implemented.	0b0
[62]	IDHi62	Common event 0x403E implemented. <b>0b0</b> Event 0x403E not implemented.	0b0
[61]	IDHi61	Common event 0x403D implemented. <b>0b0</b> Event 0x403D not implemented.	0b0
[60]	IDHi60	Common event 0x403C implemented. <b>0b0</b> Event 0x403C not implemented.	0b0

Bits	Name	Description	Reset
[59]	IDhi59	Common event 0x403B implemented. <b>0b0</b> Event 0x403B not implemented.	0b0
[58]	IDhi58	Common event 0x403A implemented. <b>0b0</b> Event 0x403A not implemented.	0b0
[57]	IDhi57	Common event 0x4039 implemented. <b>0b0</b> Event 0x4039 not implemented.	0b0
[56]	IDhi56	Common event 0x4038 implemented. <b>0b0</b> Event 0x4038 not implemented.	0b0
[55]	IDhi55	Common event 0x4037 implemented. <b>0b0</b> Event 0x4037 not implemented.	0b0
[54]	IDhi54	Common event 0x4036 implemented. <b>0b0</b> Event 0x4036 not implemented.	0b0
[53]	IDhi53	Common event 0x4035 implemented. <b>0b0</b> Event 0x4035 not implemented.	0b0
[52]	IDhi52	Common event 0x4034 implemented. <b>0b0</b> Event 0x4034 not implemented.	0b0
[51]	IDhi51	Common event 0x4033 implemented. <b>0b0</b> Event 0x4033 not implemented.	0b0
[50]	IDhi50	Common event 0x4032 implemented. <b>0b0</b> Event 0x4032 not implemented.	0b0
[49]	IDhi49	Common event 0x4031 implemented. <b>0b0</b> Event 0x4031 not implemented.	0b0
[48]	IDhi48	Common event 0x4030 implemented. <b>0b0</b> Event 0x4030 not implemented.	0b0
[47]	IDhi47	Common event 0x402F implemented. <b>0b0</b> Event 0x402F not implemented.	0b0
[46]	IDhi46	Common event 0x402E implemented. <b>0b0</b> Event 0x402E not implemented.	0b0

Bits	Name	Description	Reset
[45]	IDhi45	Common event 0x402D implemented. <b>0b0</b> Event 0x402D not implemented.	0b0
[44]	IDhi44	Common event 0x402C implemented. <b>0b0</b> Event 0x402C not implemented.	0b0
[43]	IDhi43	Common event 0x402B implemented. <b>0b0</b> Event 0x402B not implemented.	0b0
[42]	IDhi42	Common event 0x402A implemented. <b>0b0</b> Event 0x402A not implemented.	0b0
[41]	IDhi41	Common event 0x4029 implemented. <b>0b0</b> Event 0x4029 not implemented.	0b0
[40]	IDhi40	Common event 0x4028 implemented. <b>0b0</b> Event 0x4028 not implemented.	0b0
[39]	IDhi39	Common event 0x4027 implemented. <b>0b0</b> Event 0x4027 not implemented.	0b0
[38]	IDhi38	Common event 0x4026 implemented. <b>0b0</b> Event 0x4026 not implemented.	0b0
[37]	IDhi37	Common event 0x4025 implemented. <b>0b0</b> Event 0x4025 not implemented.	0b0
[36]	IDhi36	Common event 0x4024 implemented. <b>0b0</b> Event 0x4024 not implemented.	0b0
[35]	IDhi35	Common event 0x4023 implemented. <b>0b0</b> Event 0x4023 not implemented.	0b0
[34]	IDhi34	Common event 0x4022 implemented. <b>0b0</b> Event 0x4022 not implemented.	0b0
[33]	IDhi33	Common event 0x4021 implemented. <b>0b0</b> Event 0x4021 not implemented.	0b0
[32]	IDhi32	Common event 0x4020 implemented. <b>0b0</b> Event 0x4020 not implemented.	0b0

Bits	Name	Description	Reset
[31]	IDhi31	Common event 0x401F implemented. <b>0b0</b> Event 0x401F not implemented.	0b0
[30]	IDhi30	Common event 0x401E implemented. <b>0b0</b> Event 0x401E not implemented.	0b0
[29]	IDhi29	Common event 0x401D implemented. <b>0b0</b> Event 0x401D not implemented.	0b0
[28]	IDhi28	Common event 0x401C implemented. <b>0b0</b> Event 0x401C not implemented.	0b0
[27]	IDhi27	Common event 0x401B implemented. <b>0b0</b> Event 0x401B not implemented.	0b0
[26]	IDhi26	Common event 0x401A implemented. <b>0b0</b> Event 0x401A not implemented.	0b0
[25]	IDhi25	Common event 0x4019 implemented. <b>0b0</b> Event 0x4019 not implemented.	0b0
[24]	IDhi24	Common event 0x4018 implemented. <b>0b0</b> Event 0x4018 not implemented.	0b0
[23]	IDhi23	Common event 0x4017 implemented. <b>0b0</b> Event 0x4017 not implemented.	0b0
[22]	IDhi22	Common event 0x4016 implemented. <b>0b0</b> Event 0x4016 not implemented.	0b0
[21]	IDhi21	Common event 0x4015 implemented. <b>0b0</b> Event 0x4015 not implemented.	0b0
[20]	IDhi20	Common event 0x4014 implemented. <b>0b0</b> Event 0x4014 not implemented.	0b0
[19]	IDhi19	Common event 0x4013 implemented. <b>0b0</b> Event 0x4013 not implemented.	0b0
[18]	IDhi18	Common event 0x4012 implemented. <b>0b0</b> Event 0x4012 not implemented.	0b0

Bits	Name	Description	Reset
[17]	IDhi17	Common event 0x4011 implemented. <b>0b0</b> Event 0x4011 not implemented.	0b0
[16]	IDhi16	Common event 0x4010 implemented. <b>0b0</b> Event 0x4010 not implemented.	0b0
[15]	IDhi15	Common event 0x400F implemented. <b>0b0</b> Event 0x400F not implemented.	0b0
[14]	IDhi14	Common event 0x400E implemented. <b>0b0</b> Event 0x400E not implemented.	0b0
[13]	IDhi13	Common event 0x400D implemented. <b>0b0</b> Event 0x400D not implemented.	0b0
[12]	IDhi12	Common event 0x400C implemented. <b>0b0</b> Event 0x400C not implemented.	0b0
[11]	IDhi11	Common event 0x400B implemented. <b>0b0</b> Event 0x400B not implemented.	0b0
[10]	IDhi10	Common event 0x400A implemented. <b>0b0</b> Event 0x400A not implemented.	0b0
[9]	IDhi9	Common event 0x4009 implemented. <b>0b0</b> Event 0x4009 not implemented.	0b0
[8]	IDhi8	Common event 0x4008 implemented. <b>0b0</b> Event 0x4008 not implemented.	0b0
[7]	IDhi7	Common event 0x4007 implemented. <b>0b0</b> Event 0x4007 not implemented.	0b0
[6]	IDhi6	Common event 0x4006 implemented. <b>0b0</b> Event 0x4006 not implemented.	0b0
[5]	IDhi5	Common event 0x4005 implemented. <b>0b0</b> Event 0x4005 not implemented.	0b0
[4]	IDhi4	Common event 0x4004 implemented. <b>0b0</b> Event 0x4004 not implemented.	0b0

Bits	Name	Description	Reset
[3]	IDhi3	Common event 0x4003 implemented. <b>0b0</b> Event 0x4003 not implemented.	0b0
[2]	IDhi2	Common event 0x4002 implemented. <b>0b0</b> Event 0x4002 not implemented.	0b0
[1]	IDhi1	Common event 0x4001 implemented. <b>0b0</b> Event 0x4001 not implemented.	0b0
[0]	IDhi0	Common event 0x4000 implemented. <b>0b0</b> Event 0x4000 not implemented.	0b0

### Access

MRS &lt;Xt&gt;, S3\_0\_C15\_C6\_5

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b101

MSR S3\_0\_C15\_C6\_5, &lt;Xt&gt;

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b101

### Accessibility

MRS &lt;Xt&gt;, S3\_0\_C15\_C6\_5

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        return IMP_CLUSTERPMCEID1_EL1;
elseif PSTATE.EL == EL2 then
    return IMP_CLUSTERPMCEID1_EL1;
elseif PSTATE.EL == EL3 then
    return IMP_CLUSTERPMCEID1_EL1;

```

MSR S3\_0\_C15\_C6\_5, &lt;Xt&gt;

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then

```

```

    AArch64.SystemAccessTrap(EL2, 0x18);
elseif ACTLR_EL3.CLUSTERPMUEN == '0' then
    if Halted() && EDSCR.SDD == '1' then
        UNDEFINED;
    else
        AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID1_EL1 = X[t];
elseif 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];
elseif PSTATE_EL == EL3 then
    IMP_CLUSTERPMCEID1_EL1 = X[t];

```

## A.3 AArch64 RAS registers summary

The **IMPLEMENTATION DEFINED** cluster RAS registers are accessible either from System register accesses from the cores or from memory-mapped accesses on the utility bus.

The summary table provides an overview of the **IMPLEMENTATION DEFINED** AArch64 cluster RAS registers in the DSU-120AE. For more information about a register, click on the register name in the table.



Note

For registers without a listed reset value refer to the individual field resets documented on the register description pages.

For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table A-105: RAS registers summary**

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
<a href="#">ERXFR_EL1</a>	3	0	C5	C4	0	See individual bit resets.	64-bit	Selected Error Record Feature Register
<a href="#">ERXCTLR_EL1</a>	3	0	C5	C4	1	See individual bit resets.	64-bit	Selected Error Record Control Register
<a href="#">ERXSTATUS_EL1</a>	3	0	C5	C4	2	See individual bit resets.	64-bit	Selected Error Record Primary Status Register
<a href="#">ERXPFGF_EL1</a>	3	0	C5	C4	4	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Feature Register
<a href="#">ERXPFGCTL_EL1</a>	3	0	C5	C4	5	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Control Register
<a href="#">ERXPFGCDN_EL1</a>	3	0	C5	C4	6	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Countdown Register



Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description
ERXMISCO_EL1	3	0	C5	C5	0	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 0
ERXMISC1_EL1	3	0	C5	C5	1	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 1
ERXMISC2_EL1	3	0	C5	C5	2	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 2
ERXMISC3_EL1	3	0	C5	C5	3	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 3

### 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

AArch64 register ERXFR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.1 CLUSTERRAS\\_ERR0FR, Error Record Feature Register](#) on page 453 bits [63:0].

#### Attributes

##### Width

64

##### Functional group

RAS registers

##### Access type

See bit descriptions

##### Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xx00	10xx	0000	1010	1001	1010	0110
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0

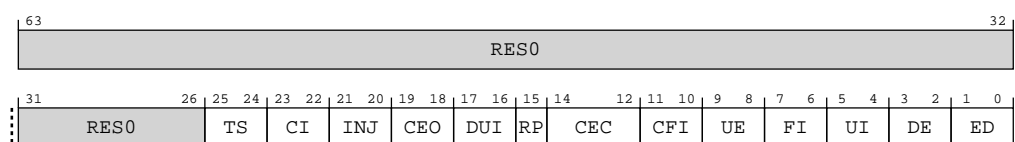


Note

Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-35: AArch64\_erxfr\_el1 bit assignments



**Table A-106: ERXFR\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:26]	RES0	Reserved	RES0
[25:24]	TS	Timestamp Extension.  <b>0b00</b> The node does not support a timestamp register.	0b00
[23:22]	CI	Critical error interrupt.  Indicates whether the critical error interrupt and associated controls are implemented.  <b>0b10</b> 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.  <b>0b00</b> The node does not implement the RAS Common Fault Injection Model Extension.  <b>0b01</b> The node implements the RAS Common Fault Injection Model Extension. See ext-CLUSTERRAS_ERROPFGF for more information.  All other values are reserved.	xx
[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.  <b>0b00</b> Counts Corrected errors. Keeps the previous error syndrome. If the counter overflows then CLUSTERRAS_ERROSTATUS.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.  <b>0b00</b> Does not support feature. ext-CLUSTERRAS_ERROCTL.DUI is <b>RES0</b> .  All other values are reserved.	0b00
[15]	RP	Repeat counter.  Indicates whether the node implements a repeat Corrected error counter in CLUSTERRAS_ERRROMISCO for each error record <m> owned by the node that implements a standard Corrected error counter.  <b>0b1</b> A first (repeat) counter and a second (other) counter are implemented. The repeat counter is the same size as the primary error counter.	0b1

Bits	Name	Description	Reset
[14:12]	CEC	Corrected Error Counter.  Indicates whether the node implements standard Corrected error counter (CE counter) mechanisms in CLUSTERRAS_ERRORMISCO for each error record <m> owned by the node that can record countable errors. <b>0b010</b> Implements an 8-bit Corrected error counter in CLUSTERRAS_ERRORMISCO[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. <b>0b10</b> Feature is controllable using ext-CLUSTERRAS_ERROCTL.R.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. <b>0b01</b> Feature always enabled. ext-CLUSTERRAS_ERROCTL.R.UE is <b>RES0</b> .	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. <b>0b10</b> Feature is controllable using ext-CLUSTERRAS_ERROCTL.R.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. <b>0b10</b> Feature is controllable using ext-CLUSTERRAS_ERROCTL.R.UI.	0b10
[3:2]	DE	Deferred error enable. <b>0b01</b> 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. <b>0b10</b> Feature is controllable using ext-CLUSTERRAS_ERROCTL.R.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' && HFGTR_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\_ERROCTLR when the value in AArch64-ERRSELR\_EL1.SEL is set to 0.

### Configurations

AArch64 register ERXCTLR\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.2 CLUSTERRAS\\_ERROCTLR, Error Record Control Register](#) on page 456 bits [63:0].

### 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure A-36: AArch64\_erxctlr\_el1 bit assignments

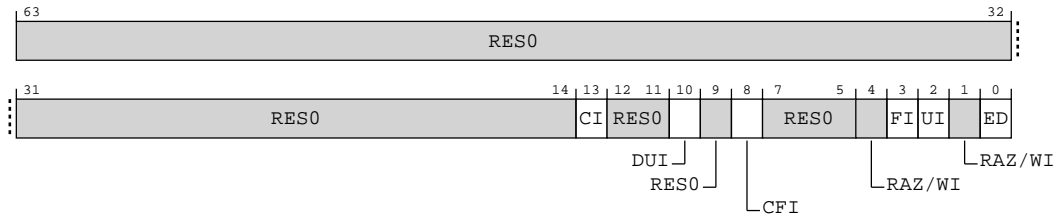


Table A-108: ERXCTLR\_EL1 bit descriptions

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.  <b>0b0</b> Critical error interrupt not generated for critical errors. Critical errors are treated as Uncontained errors.  <b>0b1</b> 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.  <b>0b0</b> 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	<p>Fault handling interrupt for Corrected errors enable. This control applies to errors arising from both reads and writes.</p> <p>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&lt;n&gt;MISCO.</p> <p><b>0b0</b></p> <p>Fault handling interrupt not generated for Corrected errors.</p> <p><b>0b1</b></p> <p>Fault handling interrupt generated for Corrected errors.</p> <p>The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.</p>	x
[7:5]	RES0	Reserved	RES0
[4]	RAZ/ WI	Reserved	RAZ/ WI
[3]	FI	<p>Fault handling interrupt enable. This control applies to errors arising from both reads and writes.</p> <p>When enabled, the fault handling interrupt is generated for all detected Corrected errors, Deferred errors, and Uncorrected errors.</p> <p><b>0b0</b></p> <p>Fault handling interrupt disabled.</p> <p><b>0b1</b></p> <p>Fault handling interrupt enabled.</p> <p>The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.</p>	x
[2]	UI	<p>Uncorrected error recovery interrupt enable. This control applies to errors arising from both reads and writes.</p> <p>When enabled, the error recovery interrupt is generated for all detected Uncorrected errors that are not deferred.</p> <p><b>0b0</b></p> <p>Error recovery interrupt disabled.</p> <p><b>0b1</b></p> <p>Error recovery interrupt enabled.</p> <p>The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.</p>	x
[1]	RAZ/ WI	Reserved	RAZ/ WI
[0]	ED	<p>Error reporting and logging enable.</p> <p>When disabled, the node behaves as if error detection and correction are disabled, and no errors are recorded or signaled by the node.</p> <p><b>0b0</b></p> <p>Error reporting disabled.</p> <p><b>0b1</b></p> <p>Error reporting enabled.</p>	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;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXCTLR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXCTLR_EL1;
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        return ERXCTLR_EL1;
elseif PSTATE.EL == EL3 then
    return ERXCTLR_EL1;

```

MSR ERXCTLR\_EL1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXCTLR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;

```

```
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXCTLR_EL1 = X[t];
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.TERR == '1' then
            UNDEFINED;
        elseif SCR_EL3.TERR == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXCTLR_EL1 = X[t];
    elseif PSTATE.EL == EL3 then
        ERXCTLR_EL1 = X[t];
```

### A.3.3 ERXSTATUS\_EL1, Selected Error Record Primary Status Register

Accesses ext-CLUSTERRAS\_ERROSTATUS when the value in AArch64-ERRSELR\_EL1.SEL is set to 0.

#### Configurations

AArch64 register ERXSTATUS\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.3 CLUSTERRAS\\_ERROSTATUS, Error Record Primary Status Register](#) on page 459 bits [63:0].

#### 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



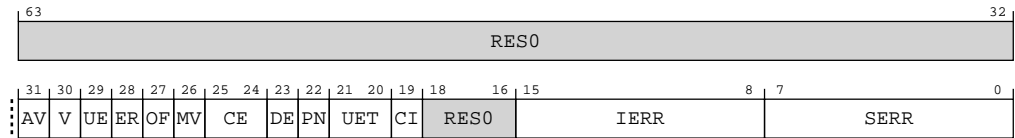
Note

Where the reset reads xxxx, see individual bits.



## Bit descriptions

**Figure A-37: AArch64\_erxstatus\_el1 bit assignments**



**Table A-111: ERXSTATUS\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31]	AV	Address Valid. <b>0b0</b> ext-CLUSTERRAS_ERROADDR not valid.  This bit is unimplemented and treated as <b>RAZ/WI</b> .	0b0
[30]	V	Status Register Valid. <b>0b0</b> CLUSTERRAS_ERROSTATUS not valid.  <b>0b1</b> CLUSTERRAS_ERROSTATUS valid. At least one error has been recorded.  This bit is read/write-one-to-clear.	0b0
[29]	UE	Uncorrected error. <b>0b0</b> No errors have been detected, or all detected errors have been either corrected or deferred.  <b>0b1</b> At least one detected error was not corrected and not deferred.  When clearing CLUSTERRAS_ERROSTATUS.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 <b>UNKNOWN</b> if CLUSTERRAS_ERROSTATUS.V is set to 0.  This bit is read/write-one-to-clear.	0b0
[28]	ER	Error Reported. <b>0b0</b> No in-band error (External abort) reported.  This bit is unimplemented and treated as <b>RAZ/WI</b> .	0b0

Bits	Name	Description	Reset
[27]	OF	<p>Overflow.</p> <p>Indicates that multiple errors have been detected. This bit is set to 1 when one of the following occurs:</p> <ul style="list-style-type: none"> <li>A Corrected error is counted and the counter overflows.</li> <li>CLUSTERRAS_ERROSTATUS.V was previously set to 1 and a type of error other than a Corrected error is recorded.</li> </ul> <p>Otherwise, this bit is unchanged when an error is recorded.</p> <p>A direct write that modifies the counter overflow flag indirectly might set this bit to an <b>UNKNOWN</b> value.</p> <p>A direct write to this bit that clears this bit to zero might indirectly set the counter overflow flag to an <b>UNKNOWN</b> value.</p> <p><b>0b0</b></p> <p>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.</p> <p><b>0b1</b></p> <p>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.</p> <p>If this bit is nonzero, then software must write 1 to this bit, to clear this bit to zero, when clearing CLUSTERRAS_ERROSTATUS.V to 0.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERROSTATUS.V is set to 0.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[26]	MV	<p>Miscellaneous Registers (CLUSTERRAS_ERRROMISCO) Valid.</p> <p><b>0b0</b></p> <p>CLUSTERRAS_ERRROMISCO is not valid.</p> <p><b>0b1</b></p> <p>The contents of CLUSTERRAS_ERRROMISCO contains additional information for an error recorded by this record.</p> <p>Only CLUSTERRAS_ERRROMISCO is implemented. CLUSTERRAS_ERRROMISC1,2,3 are treated as <b>RAZ/WI</b>.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[25:24]	CE	<p>Corrected Error.</p> <p><b>0b00</b></p> <p>No errors were corrected.</p> <p><b>0b10</b></p> <p>At least one error was corrected.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this field is nonzero, then software must write ones to this field to clear this field to zero.</p> <p>If CLUSTERRAS_ERROSTATUS.V is set to 0, this field is not valid and reads <b>UNKNOWN</b>.</p> <p>This field is read/write-one-to-clear. Writing a value other than all-zeros or all-ones sets this field to an <b>UNKNOWN</b> value.</p>	0b00

Bits	Name	Description	Reset
[23]	DE	<p>Deferred Error.</p> <p><b>0b0</b> No errors were deferred.</p> <p><b>0b1</b> At least one error was not corrected and deferred.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero.</p> <p>If CLUSTERRAS_ERROSTATUS.V is set to 0, this bit is not valid and reads <b>UNKNOWN</b>.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[22]	PN	<p>Poison.</p> <p><b>0b0</b> Uncorrected error or Deferred error recorded because a corrupt value was detected.</p> <p><b>0b1</b> Uncorrected error or Deferred error recorded because a poison value was detected.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if any of the following are true:</p> <ul style="list-style-type: none"> <li>CLUSTERRAS_ERROSTATUS.V is set to 0.</li> <li>CLUSTERRAS_ERROSTATUS.{DE, UE} are both set to 0.</li> </ul> <p>This bit is read/write-one-to-clear.</p>	0b0
[21:20]	UET	<p>Uncorrected Error Type.</p> <p>Describes the state of the component after detecting or consuming an Uncorrected error.</p> <p><b>0b00</b> Uncorrected error, Uncontainable error (UC).</p> <p>This field is not implemented and is treated as <b>RAZ/WI</b>.</p>	0b00
[19]	CI	<p>Critical error.</p> <p>Indicates whether a critical error condition has been recorded.</p> <p><b>0b0</b> No critical error condition recorded.</p> <p><b>0b1</b> Critical error condition recorded.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERROSTATUS.V is set to 0.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[18:16]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[15:8]	IERR	<p><b>IMPLEMENTATION DEFINED</b> Extended error code.</p> <p>Used with any primary error code SERR value. Additional information is placed in the CLUSTERRAS_ERRROMISCO register.</p> <p><b>0b00000000</b> If SERR == 0x7, indicates a Tag RAM error. Not used with other SERR values.</p> <p><b>0b00000010</b> If SERR == 0x7, indicates a Snoop Filter RAM error. Not used with other SERR values.</p> <p>This field is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERRROSTATUS.V is set to 0.</p>	0x00
[7:0]	SERR	<p>Primary error code.</p> <p>Indicates the type of Primary error.</p> <p><b>0b00000000</b> No error.</p> <p><b>0b00000001</b> <b>IMPLEMENTATION DEFINED</b> error.</p> <p><b>0b00000010</b> Data value from (non-associative) internal memory. For example, ECC from on-chip SRAM or buffer.</p> <p><b>0b00000011</b> <b>IMPLEMENTATION DEFINED</b> pin. For example, nSEI pin.</p> <p><b>0b00000100</b> Assertion failure. For example, consistency failure.</p> <p><b>0b00000101</b> Error detected on internal data path. For example, parity on ALU result.</p> <p><b>0b00000110</b> Data value from associative memory. For example, ECC error on cache data.</p> <p><b>0b00000111</b> Address/control value from associative memory. For example, ECC error on cache tag.</p> <p><b>0b00001000</b> Data value from a TLB. For example, ECC error on TLB data.</p> <p><b>0b00001001</b> Address/control value from a TLB. For example, ECC error on TLB tag.</p>	0x00

Bits	Name	Description	Reset
[7:0] continued	SERR	<p><b>0b00001010</b> Data value from producer. For example, parity error on write data bus.</p> <p><b>0b00001011</b> Address/control value from producer. For example, parity error on address bus.</p> <p><b>0b00001100</b> Data value from (non-associative) external memory. For example, ECC error in SDRAM.</p> <p><b>0b00001101</b> Illegal address (software fault). For example, access to unpopulated memory.</p> <p><b>0b00001110</b> Illegal access (software fault). For example, byte write to word register.</p> <p><b>0b00001111</b> Illegal state (software fault). For example, device not ready.</p> <p><b>0b00010000</b> Internal data register. For example, parity on a SIMD&amp;FP register. For a PE, all general-purpose, stack pointer, SIMD&amp;FP, and SVE registers are data registers.</p> <p><b>0b00010001</b> Internal control register. For example, Parity on a System register. For a PE, all registers other than general-purpose, stack pointer, SIMD&amp;FP, and SVE registers are control registers.</p> <p><b>0b00010010</b> Error response from subordinate. For example, error response from cache write-back.</p> <p><b>0b00010011</b> External timeout. For example, timeout on interaction with another node.</p>	0x00
[7:0] continued	SERR	<p><b>0b00010100</b> Internal timeout. For example, timeout on interface within the node.</p> <p><b>0b00010101</b> Deferred error from subordinate not supported at requester. For example, poisoned data received from a subordinate by a requester that cannot defer the error further.</p> <p>All other values are reserved.</p> <p>This field is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERR0STATUS.V is set to 0.</p>	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;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXSTATUS_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXSTATUS_EL1;
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXSTATUS_EL1;
elseif PSTATE.EL == EL3 then
    return ERXSTATUS_EL1;

```

MSR ERXSTATUS\_EL1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXSTATUS_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXSTATUS_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXSTATUS_EL1 = X[t];

```

```
elseif 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

AArch64 register ERXPFGF\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.9 CLUSTERRAS\\_ERR0PFGF, Pseudo-fault Generation Feature Register](#) on page 473 bits [63:0].

#### Attributes

##### Width

64

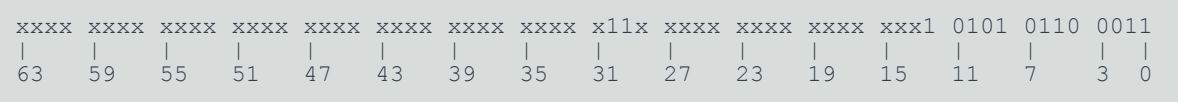
##### Functional group

RAS registers

##### Access type

See bit descriptions

##### Reset value



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-38: AArch64\_erxpfgr\_el1 bit assignments

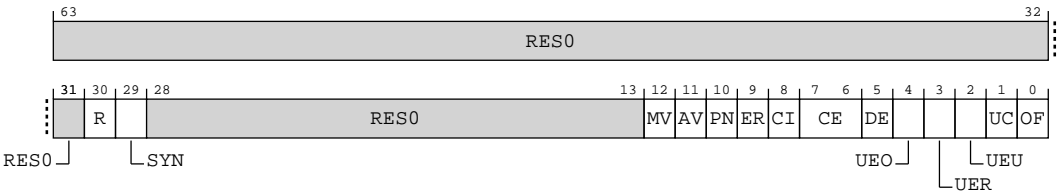


Table A-114: ERXPFGF\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[30]	R	Restartable. Support for Error Generation Counter restart mode.  <b>0b1</b> Feature controllable.	0b1
[29]	SYN	Syndrome. Fault syndrome injection.  <b>0b1</b> When an injected error is recorded, the node does not update the ext-CLUSTERRAS_ERROSTATUS.{IERR, SERR} fields. ext-CLUSTERRAS_ERROSTATUS.{IERR, SERR} are writable when ext-CLUSTERRAS_ERROSTATUS.V == 0.  <b>Note:</b> Software can write intended values into the ext-CLUSTERRAS_ERROSTATUS.{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_ERR0MISCO register when an injected error is recorded.  CLUSTERRAS_ERR0MISC1-3 registers are reserved and unused for this purpose.  <b>0b1</b> When an injected error is recorded, the node does not update all the syndrome fields in CLUSTERRAS_ERR0MISCO.  The node records syndrome in CLUSTERRAS_ERR0MISCO OFO, CECO, OFR, CECR, WAY, INDX, LVL, and IND fields and sets ext-CLUSTERRAS_ERROSTATUS.MV to 1. CLUSTERRAS_ERROPGFCTL.MV is <b>RAO</b> .  <b>Note:</b> Software can write intended values into the CLUSTERRAS_ERR0MISCO register when setting up a fault injection event.	0b1
[11]	AV	Address syndrome. Address syndrome injection. Always <b>RAZ/WI</b> .  <b>0b0</b> The node does not support ext-CLUSTERRAS_ERROADDR and does not set ext-CLUSTERRAS_ERROSTATUS.AV.	0b0
[10]	PN	Poison flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERROSTATUS.PN status flag.  <b>0b1</b> When an injected error is recorded, ext-CLUSTERRAS_ERROSTATUS.PN is set to ext-CLUSTERRAS_ERROPGFCTL.PN.	0b1
[9]	ER	Error Reported flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERROSTATUS.ER status flag.  <b>0b0</b> When an injected error is recorded, the node does not set ext-CLUSTERRAS_ERROSTATUS.ER.  This bit reads-as-zero.	0b0



Bits	Name	Description	Reset
[8]	CI	<p>Critical Error flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.CI status flag.</p> <p><b>0b1</b></p> <p>When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.CI is set to ext-CLUSTERRAS_ERROPFGCTL.CI.</p>	0b1
[7:6]	CE	<p>Corrected Error generation. Describes the types of Corrected Error that the fault generation feature of the node can generate.</p> <p><b>0b01</b></p> <p>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.</p>	0b01
[5]	DE	<p>Deferred Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b1</b></p> <p>The fault generation feature of the node allows generation of this type of error.</p>	0b1
[4]	UEO	<p>Latent or Restartable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b0</b></p> <p>The fault generation feature of the node cannot generate this type of error.</p> <p>This bit reads-as-zero.</p>	0b0
[3]	UER	<p>Signaled or Recoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b0</b></p> <p>The fault generation feature of the node cannot generate this type of error.</p> <p>This bit reads-as-zero.</p>	0b0
[2]	UEU	<p>Unrecoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b0</b></p> <p>The fault generation feature of the node cannot generate this type of error.</p> <p>This bit reads-as-zero.</p>	0b0
[1]	UC	<p>Uncontainable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b1</b></p> <p>The fault generation feature of the node allows generation of this type of error.</p>	0b1
[0]	OF	<p>Overflow flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.OF status flag.</p> <p><b>0b1</b></p> <p>When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.OF is set to ext-CLUSTERRAS_ERROPFGCTL.OF.</p>	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;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXPFGF_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXPFGF_EL1;
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
            UNDEFINED;
        elseif SCR_EL3.FIEN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
            else
                return ERXPFGF_EL1;
    elseif PSTATE.EL == EL3 then
        return ERXPFGF_EL1;

```

## A.3.5 ERXPFGCTL\_EL1, Selected Pseudo-fault Generation Control Register

Accesses the ext-CLUSTERRAS\_ERROPFGCTL register when the value in AArch64-ERRSEL\_EL1.SEL is set to 0.

### Configurations

AArch64 register ERXPFGCTL\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.10 CLUSTERRAS\\_ERROPFGCTL, Pseudo-fault Generation Control Register](#) on page 476 bits [63:0].

### Attributes

#### Width

64

#### Functional group

RAS registers

## Access type

See bit descriptions

## Reset value

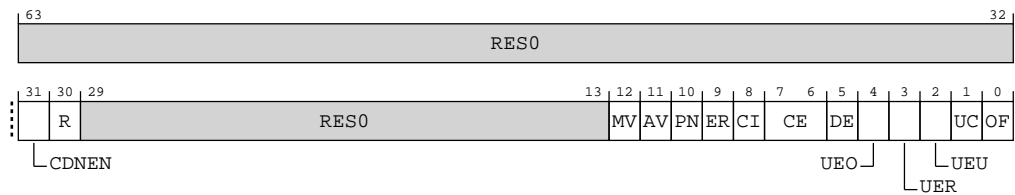
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0xxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure A-39: AArch64\_erxpgctl\_el1 bit assignments**



**Table A-116: ERXPGCTL\_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_ERRORPFGCDN into the Error Generation Counter, and enables this counter.  <b>0b0</b> The Error Generation Counter is disabled.  <b>0b1</b> The Error Generation Counter is enabled. On a write of 1 to this bit, the Error Generation Counter is set to ext-CLUSTERRAS_ERRORPFGCDN.CDN.	0b0
[30]	R	Restart. Controls whether, on reaching zero, the Error Generation Counter restarts from the ext-CLUSTERRAS_ERRORPFGCDN value, or stops.  <b>0b0</b> On reaching 0, the Error Generation Counter stops.  <b>0b1</b> On reaching 0, the Error Generation Counter is set to ext-CLUSTERRAS_ERRORPFGCDN.CDN.	x
[29:13]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[12]	MV	Miscellaneous syndrome. The value that is written to ext-CLUSTERRAS_ERROSTATUS.MV when an injected error is recorded.  <b>0b0</b> ext-CLUSTERRAS_ERROSTATUS.MV is set to 0 when an injected error is recorded.  <b>0b1</b> ext-CLUSTERRAS_ERROSTATUS.MV is set to 1 when an injected error is recorded.	x
[11]	AV	Address syndrome. The value that is written to ext-CLUSTERRAS_ERROSTATUS.AV when an injected error is recorded.  <b>0b0</b> ext-CLUSTERRAS_ERROSTATUS.AV is set to 0 when an injected error is recorded.  This bit is <b>RES0</b> .  Access to this field is: <b>RES0</b>	x
[10]	PN	Poison flag. The value that is written to ext-CLUSTERRAS_ERROSTATUS.PN when an injected error is recorded.  <b>0b0</b> ext-CLUSTERRAS_ERROSTATUS.PN is set to 0 when an injected error is recorded.  <b>0b1</b> ext-CLUSTERRAS_ERROSTATUS.PN is set to 1 when an injected error is recorded.	x
[9]	ER	<b>0b0</b> ext-CLUSTERRAS_ERROSTATUS.ER is set to 0 when an injected error is recorded.  This bit is <b>RES0</b> .  Access to this field is: <b>RES0</b>	x
[8]	CI	Critical Error flag. The value that is written to ext-CLUSTERRAS_ERROSTATUS.CI when an injected error is recorded.  <b>0b0</b> ext-CLUSTERRAS_ERROSTATUS.CI is set to 0 when an injected error is recorded.  <b>0b1</b> ext-CLUSTERRAS_ERROSTATUS.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.  <b>0b00</b> No error of this type is generated.  <b>0b01</b> A non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERROSTATUS.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_ERROPFGF.CE.	xx
[5]	DE	Deferred Error generation enable. Controls whether this type of error condition might be generated.  <b>0b0</b> No error of this type is generated.  <b>0b1</b> 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.  <b>0b0</b> No error of this type is generated.  This bit is <b>RES0</b> .  Access to this field is: <b>RES0</b>	x
[3]	UER	Signaled or Recoverable Error generation enable. Controls whether this type of error condition might be generated.  <b>0b0</b> No error of this type is generated.  This bit is <b>RES0</b> .  Access to this field is: <b>RES0</b>	x
[2]	UEU	Unrecoverable Error generation enable. Controls whether this type of error condition might be generated.  <b>0b0</b> No error of this type is generated.  This bit is <b>RES0</b> .  Access to this field is: <b>RES0</b>	x
[1]	UC	Uncontainable Error generation enable. Controls whether this type of error condition might be generated.  <b>0b0</b> No error of this type is generated.  <b>0b1</b> 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.  <b>0b0</b> ext-CLUSTERRAS_ERR0STATUS.OF is set to 0 when an injected error is recorded.  <b>0b1</b> 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>, ERXPFPGCTL\_EL1

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXPFPGCTL_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXPFPGCTL_EL1;
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elseif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXPFPGCTL_EL1;
elseif PSTATE.EL == EL3 then
    return ERXPFPGCTL_EL1;

```

MSR ERXPFPGCTL\_EL1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXPFPGCTL_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXPFPGCTL_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elseif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXPFPGCTL_EL1 = X[t];

```

```
elseif 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

AArch64 register ERXPFGCDN\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.11 CLUSTERRAS\\_ERR0PFGCDN, Pseudo-fault Generation Countdown Register](#) on page 479 bits [63:0].

#### Attributes

##### Width

64

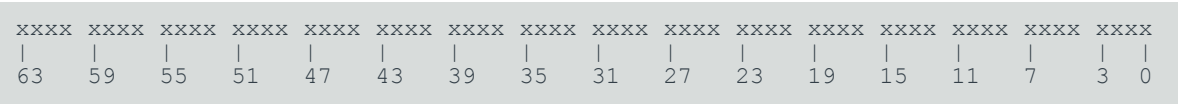
##### Functional group

RAS registers

##### Access type

See bit descriptions

##### Reset value



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure A-40: AArch64\_erxpfgcdn\_el1 bit assignments

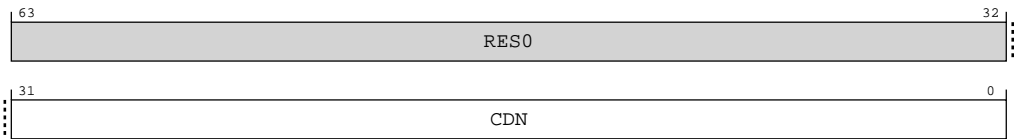


Table A-119: ERXPFGCDN\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[31:0]	CDN	<p>Countdown value.</p> <p>This field is copied to Error Generation Counter when either:</p> <ul style="list-style-type: none"> <li>Software writes ext-CLUSTERRAS_ERROPFGCTL.CDNEN with 1.</li> <li>The Error Generation Counter decrements to zero and ext-CLUSTERRAS_ERROPFGCTL.R == 1.</li> </ul> <p>While ext-CLUSTERRAS_ERROPFGCTL.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_ERROPFGCTL register is generated.</p> <p><b>Note:</b> The current Error Generation Counter value is not visible to software.</p>	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;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEN == '1' && HFGTR_EL2.ERXPFGCDN_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXPFGCDN_EL1;
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
            UNDEFINED;
        elseif SCR_EL3.FIEN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);

```



```

    else
        return ERXPFPGCDN_EL1;
    elsif PSTATE.EL == EL3 then
        return ERXPFPGCDN_EL1;

```

MSR ERXPFPGCDN\_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.ERXPFPGCDN_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
            ERXPFPGCDN_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
                ERXPFPGCDN_EL1 = X[t];
    elsif PSTATE.EL == EL3 then
        ERXPFPGCDN_EL1 = X[t];

```

### A.3.7 ERXMISCO\_EL1, Selected Error Record Miscellaneous Register 0

Accesses ext-CLUSTERRAS\_ERROMISCO 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

AArch64 register ERXMISCO\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.5 CLUSTERRAS\\_ERROMISCO, Error Record Miscellaneous Register 0](#) on page 467 bits [63:0].

#### 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	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure A-41: AArch64\_ermisc0\_el1 bit assignments

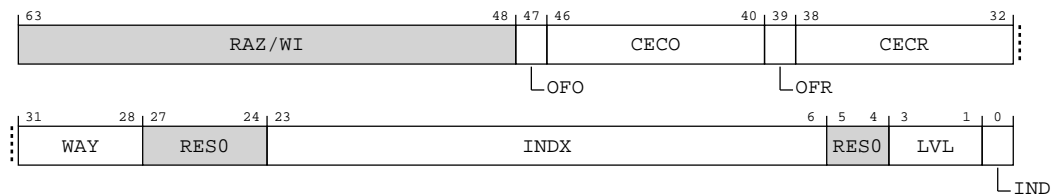


Table A-122: ERXMISC0\_EL1 bit descriptions

Bits	Name	Description	Reset
[63:48]	RAZ/WI	Reserved	RAZ/WI
[47]	OFO	<p>Sticky overflow bit for Other errors.</p> <p>Set to 1 when the Corrected error count Other (CECO) field is incremented and wraps through zero.</p> <p><b>0b0</b></p> <p>Other counter has not overflowed.</p> <p><b>0b1</b></p> <p>Other counter has overflowed.</p> <p>A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an <b>UNKNOWN</b> value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an <b>UNKNOWN</b> value.</p>	x
[46:40]	CECO	<p>Corrected error count for Other errors.</p> <p>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.</p> <p>At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.</p>	7 {x}

Bits	Name	Description	Reset
[39]	OFR	<p>Sticky overflow bit for Repeat errors.</p> <p>Set to 1 when the Corrected error count Repeat (CECR) field is incremented and wraps through zero.</p> <p><b>0b0</b> Repeat counter has not overflowed.</p> <p><b>0b1</b> Repeat counter has overflowed.</p> <p>A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERROSTATUS.OF to an <b>UNKNOWN</b> value and a direct write to ext-CLUSTERRAS_ERROSTATUS.OF that clears it to zero might indirectly set this bit to an <b>UNKNOWN</b> value.</p>	x
[38:32]	CECR	<p>Corrected error count for Repeat errors.</p> <p>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.</p> <p>The syndrome is recorded in the following fields:</p> <ul style="list-style-type: none"> <li>ext-CLUSTERRAS_ERROSTATUS.IERR</li> <li>ext-CLUSTERRAS_ERROSTATUS.SERR</li> <li>ext-CLUSTERRAS_ERRROMISCO.INDX</li> <li>ext-CLUSTERRAS_ERRROMISCO.WAY</li> </ul> <p>The syndrome is matched on a new Corrected error if all of the following are true:</p> <ul style="list-style-type: none"> <li>ext-CLUSTERRAS_ERROSTATUS.MV bit is set,</li> <li>ext-CLUSTERRAS_ERROSTATUS.IERR matches the new error,</li> <li>ext-CLUSTERRAS_ERROSTATUS.SERR matches the new error,</li> <li>ext-CLUSTERRAS_ERRROMISCO.INDX matches the new error,</li> <li>ext-CLUSTERRAS_ERRROMISCO.WAY matches the new error.</li> </ul> <p>CLUSTERRAS_ERROSTATUS.MV indicates the validity of the INDX and WAY fields of the CLUSTERRAS_ERRROMISCO register</p> <p>At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.</p>	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	<p>L3 Cache-level that contained the error. Always 0x2.</p> <p><b>0b010</b> L3 cache.</p>	xxx
[0]	IND	<p>L3 Cache instruction vs. data cache that contained the error. Always data (0x0).</p> <p><b>0b0</b> Data cache error.</p>	x

## Access

MRS <Xt>, ERXMISCO\_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b000

MSR ERXMISCO\_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b000

## Accessibility

MRS <Xt>, ERXMISCO\_EL1

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXMISCN_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXMISCO_EL1;
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.TERR == '1' then
            UNDEFINED;
        elseif SCR_EL3.TERR == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
            else
                return ERXMISCO_EL1;
    elseif PSTATE.EL == EL3 then
        return ERXMISCO_EL1;

```

MSR ERXMISCO\_EL1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCN_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then

```

```

        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXMISC0_EL1 = X[t];
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.TERR == '1' then
            UNDEFINED;
        elseif SCR_EL3.TERR == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXMISC0_EL1 = X[t];
    elseif 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

AArch64 register ERXMISC1\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.6 CLUSTERRAS\\_ERR0MISC1, Error Record Miscellaneous Register 1](#) on page 469 bits [63:0].

#### Attributes

##### Width

64

##### Functional group

RAS registers

##### Access type

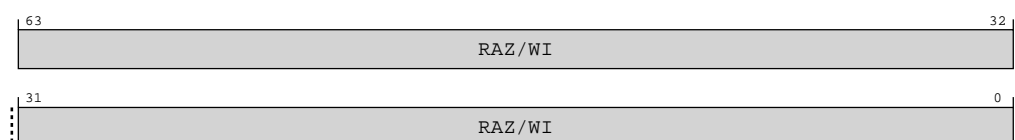
See bit descriptions

##### Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000  
0000

#### Bit descriptions

**Figure A-42: AArch64\_erxmisc1\_el1 bit assignments**



**Table A-125: ERXMISC1\_EL1 bit descriptions**

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

**Access**

MRS &lt;Xt&gt;, ERXMISC1\_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b001

MSR ERXMISC1\_EL1, &lt;Xt&gt;

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b001

**Accessibility**

MRS &lt;Xt&gt;, ERXMISC1\_EL1

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXMIScN_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXMISC1_EL1;
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.TERR == '1' then
            UNDEFINED;
        elseif SCR_EL3.TERR == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
            else
                return ERXMISC1_EL1;
    elseif PSTATE.EL == EL3 then
        return ERXMISC1_EL1;

```

MSR ERXMISC1\_EL1, &lt;Xt&gt;

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif 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

AArch64 register ERXMISC2\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.7 CLUSTERRAS\\_ERR0MISC2, Error Record Miscellaneous Register 2](#) on page 471 bits [63:0].

#### Attributes

##### Width

64

##### Functional group

RAS registers

##### Access type

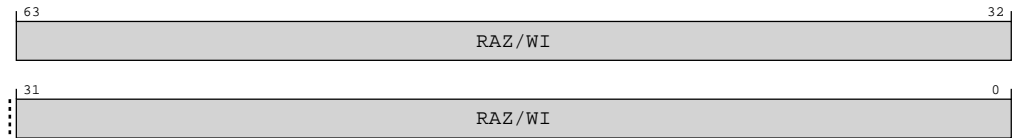
See bit descriptions

##### Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000  
0000

## Bit descriptions

**Figure A-43: AArch64\_erxmisc2\_el1 bit assignments**



**Table A-128: 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;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGTR_EL2.ERXMISCn_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXMISC2_EL1;
    elseif PSTATE.EL == EL2 then
        if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
        priority when SDD == '1'" && SCR_EL3.TERR == '1' then
            UNDEFINED;
        elseif SCR_EL3.TERR == '1' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);

```



```

else
    return ERXMISC2_EL1;
elseif PSTATE.EL == EL3 then
    return ERXMISC2_EL1;

```

MSR ERXMISC2\_EL1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCN_EL1 == '1'
then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC2_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC2_EL1 = X[t];
elseif PSTATE.EL == EL3 then
    ERXMISC2_EL1 = X[t];

```

### A.3.10 ERXMISC3\_EL1, Selected Error Record Miscellaneous Register 3

Accesses ext-CLUSTERRAS\_ERRORMISC3 when the value in AArch64-ERRSELR\_EL1.SEL is set to 0.

Unimplemented error syndrome register.

#### Configurations

AArch64 register ERXMISC3\_EL1 bits [63:0] are architecturally mapped to External System register [B.1.3.8 CLUSTERRAS\\_ERRORMISC3, Error Record Miscellaneous Register 3](#) on page 472 bits [63:0].

#### 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\_erxmisc3\_el1 bit assignments

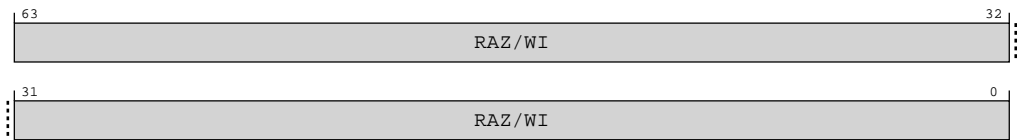


Table A-131: ERXMISC3\_EL1 bit descriptions

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;
```

```

elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            return ERXMISC3_EL1;
elseif PSTATE.EL == EL3 then
    return ERXMISC3_EL1;

```

## MSR ERXMISC3\_EL1, <Xt>

```

if PSTATE.EL == EL0 then
    UNDEFINED;
elseif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
    then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXMISC3_EL1 = X[t];
elseif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
    priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elseif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
        else
            ERXMISC3_EL1 = X[t];
elseif 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-120AE* (DSU-120AE).

## B.1 Registers accessed over the utility bus

This section contains the descriptions for all the external registers in the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) accessed over the utility bus.

### B.1.1 External cluster system control registers 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.

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-120AE. For more information about a register, click on the register name in the table.



Note

- You must access the cluster system control registers from the Secure state.
- The cluster system control registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- The base address for the cluster system 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
0x0000	<a href="#">CLUSTERIDR</a>	See individual bit resets.	64-bit	Cluster Main Revision Register
0x0008	<a href="#">CLUSTERREVIDR</a>	See individual bit resets.	64-bit	Cluster ECO ID Register
0x0010	<a href="#">CLUSTERPWRCtrlR</a>	See individual bit resets.	64-bit	Cluster Power Control Register
0x0028	<a href="#">CLUSTERL3DNTH0</a>	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register
0x0030	<a href="#">CLUSTERL3DNTH1</a>	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register
0x0038	<a href="#">CLUSTERL3UPTH0</a>	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register
0x0040	<a href="#">CLUSTERL3UPTH1</a>	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register
0x0048	<a href="#">CLUSTERBUSQOS</a>	See individual bit resets.	64-bit	Cluster Bus QoS Control Register
0x0050	<a href="#">CLUSTERCFR</a>	See individual bit resets.	64-bit	Cluster Configuration Register

Offset	Name	Reset	Width	Description
0x0058	CLUSTERACTLR	See individual bit resets.	64-bit	Cluster Auxiliary Control Register
0x0060	CLUSTERECTLR	See individual bit resets.	64-bit	Cluster Extended Control Register
0x0068	CLUSTERCFR2	See individual bit resets.	64-bit	Cluster Configuration Register 2
0x0080	CLUSTERPPMCR	See individual bit resets.	64-bit	Cluster PPM Control Register
0x0088	CLUSTERMPMMCR	See individual bit resets.	64-bit	Cluster MPMM Control Register
0x0090	CLUSTERL3UPTH2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register

B.1.1.1 CLUSTERIDR, Cluster Main Revision Register

Holds the revision and patch level of the cluster.

Configurations

External register CLUSTERIDR bits [63:0] are architecturally mapped to AArch64 System register [A.1.2 IMP\\_CLUSTERIDR\\_EL1, Cluster Main Revision Register](#) on page 269 bits [63:0].

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	0000	0001
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-1: ext\_clusteridr bit assignments

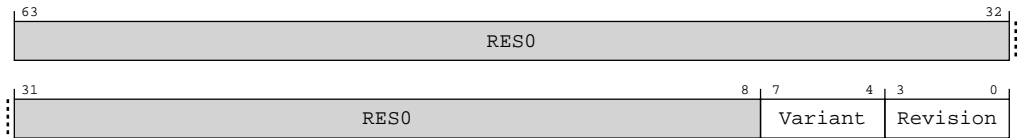


Table B-2: CLUSTERIDR bit descriptions

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.	0b0000
[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.	0b0001

Accessibility

Component	Offset	Instance	Range
Cluster	0x0000	CLUSTERIDR	None

This interface is accessible as follows:

RO

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

External register CLUSTERREVIDR bits [63:0] are architecturally mapped to AArch64 System register A.1.3 IMP\_CLUSTERREVIDR\_EL1, Cluster ECO ID Register on page 271 bits [63:0].

Attributes

Width

64

## Component

Cluster

## Register offset

0x0008

## Access type

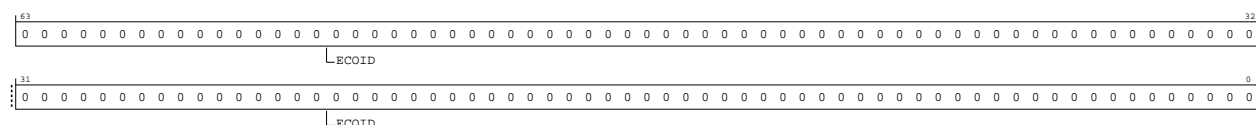
RO

## Reset value

[illegible]

## Bit descriptions

### Figure B-2: ext\_clusterrevidr bit assignments



### Table B-4: CLUSTERREVIDR bit descriptions

Bits	Name	Description	Reset
[63:0]	ECOID	Contains ECO information. Refer to the errata documentation for any bit allocations.  Customer ECO ID	0x0000000000000000

## Accessibility

Component	Offset	Instance	Range
Cluster	0x0008	CLUSTERREVIDR	None

This interface is accessible as follows:

RO

### B.1.1.3 CLUSTERPWRCTRL, 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 1010 0100 0000 0001 0111  
0000

## Bit descriptions

Figure B-3: ext\_clusterpwrctlr bit assignments

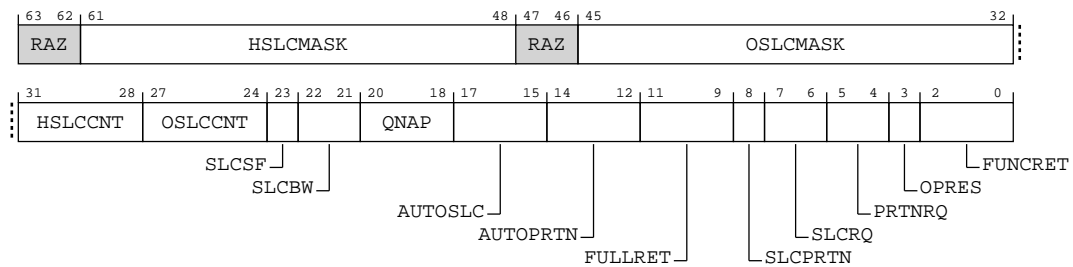


Table B-6: CLUSTERPWRCTLR bit descriptions

Bits	Name	Description	Reset
[63:62]	RAZ	Reserved	RAZ
[61:48]	HSLCMASK	Half slice core mask. This contains one bit per core, and if that bit is set then the core is included in the count that is compared with the HSLCCNT field. Bits above the number of cores implemented are <b>RAZ/WI</b> .	0b00000000000000
[47:46]	RAZ	Reserved	RAZ
[45:32]	OSLCMASK	One slice core mask. This contains one bit per core, and if that bit is set then the core is included in the count that is compared with the OSLCCNT field. Bits above the number of cores implemented are <b>RAZ/WI</b> .	0b00000000000000
[31:28]	HSLCCNT	Half slice core count. When AUTOSLC is non-zero, if the number of cores powered on in the group selected by the HSLCMASK field is greater than this field then it will prevent the transition from ALL SLICE to HALF SLICE mode, or cause a transition from HALF SLICE to ALL SLICE. Note that a core that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this count.	0b0000
[27:24]	OSLCCNT	One slice core count. When AUTOSLC is non-zero, if the number of cores powered on in the group selected by the OSLCMASK field is greater than this field then it will prevent the transition from ALL SLICE or HALF SLICE to ONE SLICE mode, or cause a transition from ONE SLICE to HALF SLICE. Note that a core that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this count.	0b0000



Bits	Name	Description	Reset
[23]	SLCSF	<p>Include snoop filter capacity in the AUTOSLC decision. If powering down slices would reduce the amount of snoop filter below the amount needed for the currently powered on cores then this will prevent the slice powerdown. Note that a core that is off but has the IMP_CLUSTERPWDRN_EL1.SHORTSLP bit set will be treated as if it was on for this calculation.</p> <p><b>0b0</b> Ignore the snoop filter in AUTOSLC decisions</p> <p><b>0b1</b> AUTOSLC will ensure enough slices are powered on for the currently powered on cores</p>	0b1
[22:21]	SLCBW	<p>Include slice bandwidth in the AUTOSLC decision. Prevent the slice powerdown if powering down slices would reduce the available bandwidth below the currently used bandwidth. Power up more slices if the slice bandwidth is close to saturating with the current number of slices.</p> <p><b>0b00</b> Ignore the slice bandwidth in AUTOSLC decisions</p> <p><b>0b01</b> AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 3% of the AUTOSLC time period</p> <p><b>0b11</b> AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 12% of the AUTOSLC time period</p>	0b01
[20:18]	QNAP	<p>Control L3 data RAM quick nap enable time</p> <p><b>0b000</b> Quick nap disabled</p> <p><b>0b001</b> Enabled with 8 cycle timeout</p> <p><b>0b010</b> Enabled with 12 cycle timeout</p> <p><b>0b011</b> Enabled with 16 cycle timeout</p> <p><b>0b100</b> Enabled with 24 cycle timeout</p> <p><b>0b101</b> Enabled with 32 cycle timeout</p> <p><b>0b110</b> Enabled with 64 cycle timeout</p> <p><b>0b111</b> Enabled with 128 cycle timeout</p>	0b001

Bits	Name	Description	Reset
[17:15]	AUTOSLC	<p>Enable automatic slice 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.</p> <p><b>0b000</b> Disabled</p> <p><b>0b001</b> 524,288 architectural timer ticks, time period of 524us</p> <p><b>0b010</b> 1,048,576 architectural timer ticks, time period of 1ms</p> <p><b>0b011</b> 2,097,152 architectural timer ticks, time period of 2.1ms</p> <p><b>0b100</b> 4,194,304 architectural timer ticks, time period of 4.2ms</p> <p><b>0b101</b> 8,388,608 architectural timer ticks, time period of 8.4ms</p> <p><b>0b110</b> 16,777,216 architectural timer ticks, time period of 16.8ms</p> <p><b>0b111</b> 33,554,432 architectural timer ticks, time period of 33.6ms</p>	0b000
[14:12]	AUTOPRTN	<p>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.</p> <p><b>0b000</b> Disabled</p> <p><b>0b001</b> 524,288 architectural timer ticks, time period of 524us</p> <p><b>0b010</b> 1,048,576 architectural timer ticks, time period of 1ms</p> <p><b>0b011</b> 2,097,152 architectural timer ticks, time period of 2.1ms</p> <p><b>0b100</b> 4,194,304 architectural timer ticks, time period of 4.2ms</p> <p><b>0b101</b> 8,388,608 architectural timer ticks, time period of 8.4ms</p> <p><b>0b110</b> 16,777,216 architectural timer ticks, time period of 16.8ms</p> <p><b>0b111</b> 33,554,432 architectural timer ticks, time period of 33.6ms</p>	0b000

Bits	Name	Description	Reset
[11:9]	FULLRET	<p>Enable the FULL_RET slice powerdown mode and time period. Note that while this would typically be a longer period than the FUNC_RET field, to allow entry into FUNC_RET first, but if it is shorter then FULL_RET will be entered directly rather than via FUNC_RET.</p> <p><b>0b000</b> Disabled</p> <p><b>0b001</b> 128 architectural timer ticks, time period of 128ns</p> <p><b>0b010</b> 512 architectural timer ticks, time period of 512ns</p> <p><b>0b011</b> 2,048 architectural timer ticks, time period of 2us</p> <p><b>0b100</b> 4,096 architectural timer ticks, time period of 4.1us</p> <p><b>0b101</b> 8,192 architectural timer ticks, time period of 8.2us</p> <p><b>0b110</b> 16,384 architectural timer ticks, time period of 16.4us</p> <p><b>0b111</b> 32,768 architectural timer ticks, time period of 32.8us</p>	0b000
[8]	SLCPRTN	<p>The AUTOSLC logic should make slice operating mode decisions considering the AUTOPRTN status. If enabled, AUTOSLC will not power down more slices if AUTOPRTN is keeping all the L3 cache portions powered on. The AUTOPRTN mechanism can also request to power up more slices if it determines that more cache is needed and all the L3 cache portions are powered on</p> <p><b>0b0</b> AUTOSLC decisions should ignore the AUTOPRTN status</p> <p><b>0b1</b> AUTOSLC decisions should include the AUTOPRTN status</p>	0b1
[7:6]	SLCRQ	<p>Cache slice power request. These bits are passed to the PPU as an advisory request for which slices to power. Note that when the AUTOSLC field is not 0b000, higher modes might be requested by the AUTOSLC mechanism and this field acts as a minimum operating mode.</p> <p><b>0b00</b> Request that one L3 cache slice is powered on</p> <p><b>0b01</b> Request that all L3 cache slices are powered on</p> <p><b>0b10</b> Request that half the L3 cache slices are powered on</p>	0b01

Bits	Name	Description	Reset
[5:4]	PRTNRQ	<p>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.</p> <p><b>0b00</b> Request that none of the L3 cache portions in each slice is powered on</p> <p><b>0b01</b> Request that half of the L3 cache portions in each slice are powered on</p> <p><b>0b11</b> Request that both of the L3 cache portions in each slice are powered on</p>	0b11
[3]	OPRES	<p>Restore previous operating mode after cluster power off</p> <p><b>0b0</b> Operating mode defaults to ALL RAM ALL SLICE when powering on the cluster</p> <p><b>0b1</b> Enable restoring of previous operating mode</p>	0b0
[2:0]	FUNCRET	<p>L3 Data RAM retention control.</p> <p><b>0b000</b> Disable the retention circuit.</p> <p><b>0b001</b> 128 architectural timer ticks, time period of 128ns minimum delay before retention</p> <p><b>0b010</b> 512 architectural timer ticks, time period of 512ns minimum delay before retention</p> <p><b>0b011</b> 2,048 architectural timer ticks, time period of 2us minimum delay before retention</p> <p><b>0b100</b> 4,096 architectural timer ticks, time period of 4.1us minimum delay before retention</p> <p><b>0b101</b> 8,192 architectural timer ticks, time period of 8.2us minimum delay before retention</p> <p><b>0b110</b> 16,384 architectural timer ticks, time period of 16.4us minimum delay before retention</p> <p><b>0b111</b> 32,768 architectural timer ticks, time period of 32.8us minimum delay before retention</p>	0b000

## Accessibility

Component	Offset	Instance	Range
Cluster	0x0010	CLUSTERPWRCTLR	None

This interface is accessible as follows:

RW

B.1.1.4 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

External register CLUSTERL3DNTH0 bits [63:0] are architecturally mapped to AArch64 System register [A.1.9 IMP\\_CLUSTERL3DNTH0\\_EL1, Cluster L3 Downsize Threshold0 Register](#) on page 289 bits [63:0].

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-4: ext\_clusterl3dnth0 bit assignments

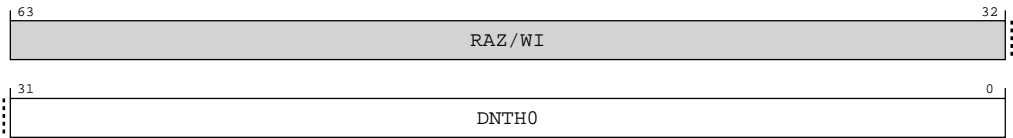


Table B-8: CLUSTERL3DNTH0 bit descriptions

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

Accessibility

Component	Offset	Instance	Range
Cluster	0x0028	CLUSTERL3DNTH0	None

This interface is accessible as follows:

RW

B.1.1.5 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  
0000

Bit descriptions

Figure B-5: ext\_clusterl3dnth1 bit assignments

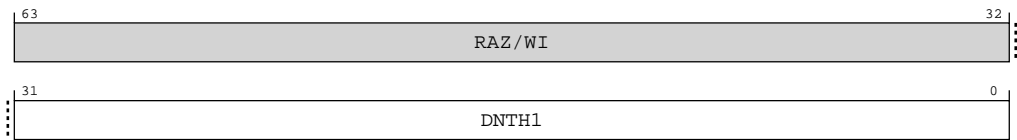


Table B-10: CLUSTERL3DNTH1 bit descriptions

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

Accessibility

Component	Offset	Instance	Range
Cluster	0x0030	CLUSTERL3DNTH1	None

This interface is accessible as follows:

RW

B.1.1.6 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  
0000

Bit descriptions

Figure B-6: ext\_clusterl3upth0 bit assignments

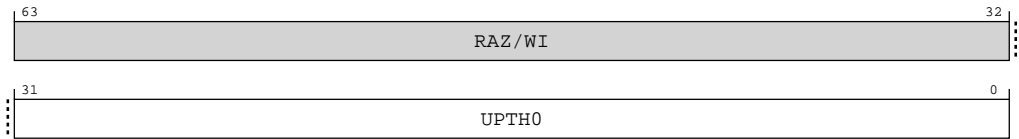


Table B-12: CLUSTERL3UPTH0 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[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

Accessibility

Component	Offset	Instance	Range
Cluster	0x0038	CLUSTERL3UPTH0	None

This interface is accessible as follows:

RW

B.1.1.7 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-7: ext\_clusterl3upth1 bit assignments

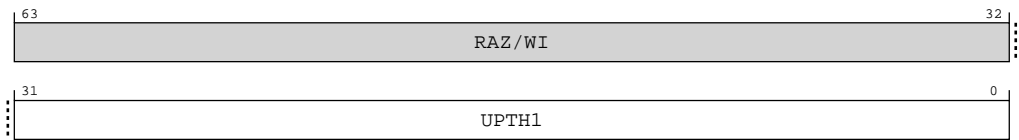




Table B-14: CLUSTERL3UPTH1 bit descriptions

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

Accessibility

Component	Offset	Instance	Range
Cluster	0x0040	CLUSTERL3UPTH1	None

This interface is accessible as follows:

RW

B.1.1.8 CLUSTERBUSQOS, Cluster Bus QoS Control Register

Determines the value driven on the CHI bus QoS field.

Configurations

External register CLUSTERBUSQOS bits [63:0] are architecturally mapped to AArch64 System register [A.1.13 IMP\\_CLUSTERBUSQOS\\_EL1, Cluster Bus QoS Control Register](#) on page 297 bits [63:0].

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 1110 1110 1011 1110

Bit descriptions

Figure B-8: ext\_clusterbusqos bit assignments

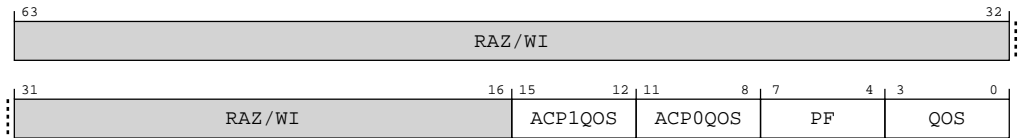


Table B-16: CLUSTERBUSQOS bit descriptions

Bits	Name	Description	Reset
[63:16]	RAZ/WI	Reserved	RAZ/WI
[15:12]	ACP1QOS	Valid driven on the CHI bus QoS field for acp 1 accesses.	0b1110
[11:8]	ACP0QOS	Valid driven on the CHI bus QoS field for acp 0 accesses.	0b1110
[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

Accessibility

Component	Offset	Instance	Range
Cluster	0x0048	CLUSTERBUSQOS	None

This interface is accessible as follows:

RW

B.1.1.9 CLUSTERCFR, Cluster Configuration Register

Contains details of the hardware configuration of the cluster.

Configurations

External register CLUSTERCFR bits [63:0] are architecturally mapped to AArch64 System register [A.1.1 IMP\\_CLUSTERCFR\\_EL1, Cluster Configuration Register](#) on page 263 bits [63:0].

Attributes

Width

64

Component

Cluster

Register offset

0x0050

Access type

RO

Reset value

xxxx	xxxx	x0xx	x000	0000	0000	0000	000x	xxxx	xxx0	xxxx	x0xx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-9: ext\_clustercfr bit assignments

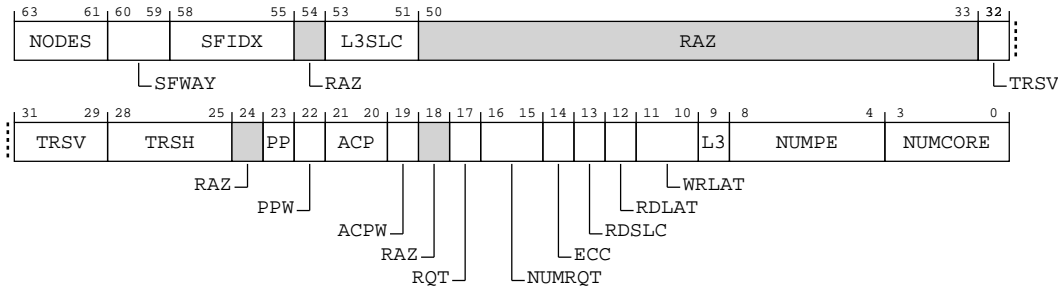


Table B-18: CLUSTERCFR bit descriptions

Bits	Name	Description	Reset
[63:61]	NODES	Number of transport nodes.  <b>0b001</b> One node.  <b>0b010</b> Two nodes.  <b>0b011</b> Three nodes.  <b>0b100</b> Four nodes.  <b>0b101</b> Eight nodes.	xxx

Bits	Name	Description	Reset
[60:59]	SFWAY	<p>Number of Snoop Filter ways.</p> <p><b>0b00</b> 4 ways</p> <p><b>0b01</b> 6 ways</p> <p><b>0b10</b> 8 ways</p> <p><b>0b11</b> 12 ways</p>	xx
[58:55]	SFIDX	Log2 of the number of snoop filter indexes.	xxxx
[54]	RAZ	Reserved	RAZ
[53:51]	L3SLC	<p>Number of L3 cache slices.</p> <p><b>0b000</b> Eight L3 cache slices.</p> <p><b>0b001</b> One L3 cache slice.</p> <p><b>0b010</b> Two L3 cache slices.</p> <p><b>0b100</b> Four L3 cache slices.</p>	xxx
[50:33]	RAZ	Reserved	RAZ
[32:29]	TRSV	<p>Transport register slices, vertical.</p> <p><b>0b0000</b> No register slices</p> <p><b>0b0001</b> One register slice</p> <p><b>0b0010</b> Two register slices</p> <p><b>0b0011</b> Three register slices</p> <p><b>0b0100</b> Four register slices</p> <p><b>0b0101</b> Five register slices</p> <p><b>0b0110</b> Six register slices</p> <p><b>0b0111</b> Seven register slices</p> <p><b>0b1000</b> Eight register slices</p>	xxxx

Bits	Name	Description	Reset
[28:25]	TRSH	<p>Transport register slices, horizontal.</p> <p><b>0b0000</b> No register slices</p> <p><b>0b0001</b> One register slice</p> <p><b>0b0010</b> Two register slices</p> <p><b>0b0011</b> Three register slices</p> <p><b>0b0100</b> Four register slices</p> <p><b>0b0101</b> Five register slices</p> <p><b>0b0110</b> Six register slices</p> <p><b>0b0111</b> Seven register slices</p> <p><b>0b1000</b> Eight register slices</p>	xxxx
[24]	RAZ	Reserved	RAZ
[23]	PP	<p>Peripheral port presence.</p> <p><b>0b0</b> No peripheral port present</p> <p><b>0b1</b> Peripheral port present</p>	x
[22]	PPW	<p>Peripheral port width.</p> <p><b>0b0</b> 64 bit data width</p> <p><b>0b1</b> 256 bit data width</p>	x
[21:20]	ACP	<p>ACP interface presence.</p> <p><b>0b00</b> No ACP interface present</p> <p><b>0b01</b> One ACP interface present</p> <p><b>0b10</b> Two ACP interface present</p>	xx
[19]	ACPW	<p>ACP interface width.</p> <p><b>0b0</b> 128 bit data width</p> <p><b>0b1</b> 256 bit data width</p>	x

Bits	Name	Description	Reset
[18]	RAZ	Reserved	RAZ
[17]	RQT	Requester bus interface type.  <b>0b0</b> AXI interface  <b>0b1</b> CHI interface	x
[16:15]	NUMRQT	Number of Requester interfaces.  <b>0b00</b> One requester  <b>0b01</b> Two requesters  <b>0b10</b> Three requesters  <b>0b11</b> Four requesters	xx
[14]	ECC	SCU-L3 ECC configuration.  <b>0b0</b> SCU-L3 is configured with no ECC  <b>0b1</b> SCU-L3 is configured with ECC	x
[13]	RDSLC	L3 data RAM read register slice.  <b>0b0</b> No register slice present  <b>0b1</b> Register slice present	x
[12]	RDLAT	L3 Data RAM read latency.  <b>0b0</b> Two cycle output delay from L3 data RAMs  <b>0b1</b> Three cycle output delay from L3 data RAMs	x
[11:10]	WRLAT	L3 Data RAM write latency.  <b>0b00</b> One cycle input delay from L3 data RAMs  <b>0b01</b> Two cycle input delay from L3 data RAMs  <b>0b10</b> Two cycle input delay plus a one cycle hold	xx
[9]	L3	L3 cache presence.  <b>0b0</b> No L3 cache present  <b>0b1</b> 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.  <b>0b0000</b> One core  <b>0b0001</b> Two cores  <b>0b0010</b> Three cores  <b>0b0011</b> Four cores  <b>0b0100</b> Five cores  <b>0b0101</b> Six cores  <b>0b0110</b> Seven cores  <b>0b0111</b> Eight cores  <b>0b1000</b> Nine core  <b>0b1001</b> Ten cores  <b>0b1010</b> Eleven cores  <b>0b1011</b> Twelve cores  <b>0b1100</b> Thirteen cores  <b>0b1101</b> Fourteen cores	xxxx

### Accessibility

Component	Offset	Instance	Range
Cluster	0x0050	CLUSTERCFR	None

This interface is accessible as follows:

RO

B.1.1.10 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

External register CLUSTERACTLR bits [63:0] are architecturally mapped to AArch64 System register [A.1.4 IMP\\_CLUSTERACTLR\\_EL1, Cluster Auxiliary Control Register](#) on page 272 bits [63:0].

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	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-10: ext\_clusteractlr bit assignments

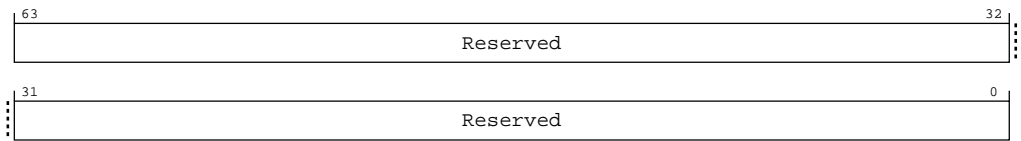


Table B-20: CLUSTERACTLR bit descriptions

Bits	Name	Description	Reset
[63:0]	Reserved	Reserved for Arm internal use	64 { x }



Accessibility

Component	Offset	Instance	Range
Cluster	0x0058	CLUSTERACTLR	None

This interface is accessible as follows:

RW

B.1.1.11 CLUSTERECTLR, Cluster Extended Control Register

This register should be used for dynamically changing implementation specific control bits.

Configurations

External register CLUSTERECTLR bits [63:0] are architecturally mapped to AArch64 System register [A.1.5 IMP\\_CLUSTERECTLR\\_EL1, Cluster Extended Control Register](#) on page 274 bits [63:0].

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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0

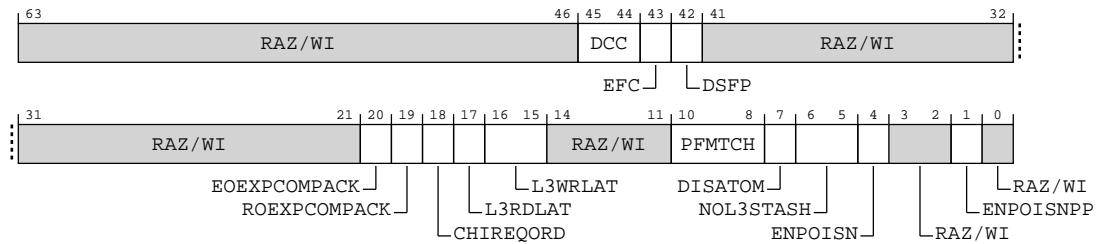


Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-11: ext\_clusterectlr bit assignments**



**Table B-22: CLUSTERECTLR bit descriptions**

Bits	Name	Description	Reset
[63:46]	RAZ/WI	Reserved	RAZ/WI
[45:44]	DCC	<p>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.</p> <p><b>0b00</b></p> <p>Disables sending data when clean cachelines are evicted.</p> <p><b>0b01</b></p> <p>Enables sending WriteEvictFull transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data.</p> <p><b>0b10</b></p> <p>Enables sending WriteEvictOrEvict transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data.</p> <p><b>0b11</b></p> <p>Enables sending WriteEvictOrEvict transactions when Unique Clean or Shared Clean cachelines are evicted. This is the reset value.</p>	0b11
[43]	EFC	<p>Eviction flush control. Controls whether hardware cache flushes and DC CISW instructions send data when evicting clean cachelines on the CHI interface.</p> <p><b>0b0</b></p> <p>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.</p> <p><b>0b1</b></p> <p>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).</p>	0b0
[42]	DSFP	<p>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.</p> <p><b>0b0</b></p> <p>Disables sending Evict transactions when clean cachelines are evicted without data.</p> <p><b>0b1</b></p> <p>Enables sending of Evict transactions when clean cachelines are evicted without data. This is the reset value.</p>	0b1
[41:21]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[20]	EOEXPCOMPACT	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Endpoint Order transactions to the system  <b>0b0</b> CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=0  <b>0b1</b> CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=1	0b0
[19]	ROEXPCOMPACT	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Request Order transactions to the system  <b>0b0</b> CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=0  <b>0b1</b> 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.  <b>0b0</b> Disables sending Request Order on the CHI interface to the system. Will send No Order instead.  <b>0b1</b> Enables sending Request Order on the CHI interface to the system for Non-snoopable transactions.	0b0
[17]	L3RDLAT	L3 data RAM read (output) latency.  <b>0b0</b> The L3 data RAM output latency is 2 cycles.  <b>0b1</b> The L3 data RAM output latency is 3 cycles.	x <sup>4</sup>
[16:15]	L3WRLAT	L3 data RAM write (input) latency.  <b>0b00</b> The L3 data RAM input latency is 1 cycle with an additional hold cycle.  <b>0b01</b> The L3 data RAM input latency is 2 cycles without an additional hold cycle.  <b>0b10</b> 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 <sup>5</sup>
[14:11]	RAZ/WI	Reserved	RAZ/WI

<sup>4</sup> This field resets to the value of the L3\_DATA\_RD\_LATENCY configuration parameter.

<sup>5</sup> This field resets to the value of the L3\_DATA\_WR\_LATENCY configuration parameter.

Bits	Name	Description	Reset
[10:8]	PFMTCH	<p>Prefetch matching delay. Controls the amount of time a prefetch waits for a possible match with a later read. Encoded as powers of 2, from 1-128.</p> <p><b>0b000</b> Wait for 1 cycle.</p> <p><b>0b001</b> Wait for 2 cycles.</p> <p><b>0b010</b> Wait for 4 cycles.</p> <p><b>0b011</b> Wait for 8 cycles.</p> <p><b>0b100</b> Wait for 16 cycles.</p> <p><b>0b101</b> Wait for 32 cycles.</p> <p><b>0b110</b> Wait for 64 cycles.</p> <p><b>0b111</b> Wait for 128 cycles.</p>	0b101
[7]	DISATOM	<p>Disable cacheable shareable atomics being sent to the interconnect.</p> <p><b>0b0</b> Cacheable shareable atomics will be sent to the interconnect if the BROADCASTATOMIC pin is set.</p> <p><b>0b1</b> Cacheable shareable atomics will be handled inside the cluster.</p>	0b0
[6:5]	NOL3STASH	<p>CPU StashOnce request behaviour when L3 is not present or powered down.</p> <p><b>0b00</b> Stashes are sent out to the interconnect, if supported.</p> <p><b>0b01</b> Normal read request sent to interconnect.</p> <p><b>0b10</b> StashOnce has no effect.</p>	0b10
[4]	ENPOISN	<p>Interconnect data poisoning support for the CHI Requester(s). This bit is ignored for AXI configurations, which never support poisoning.</p> <p><b>0b0</b> Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop.</p> <p><b>0b1</b> 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.</p>	0b1
[3:2]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[1]	ENPOISNPP	Interconnect data poisoning support for the CHI Peripheral Port. This bit is ignored for AXI configurations, which never support poisoning.  <b>0b0</b> Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop.  <b>0b1</b> 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

Accessibility

Component	Offset	Instance	Range
Cluster	0x0060	CLUSTERECTLR	None

This interface is accessible as follows:

RW

B.1.1.12 CLUSTERCFR2, Cluster Configuration Register 2

Contains details of the hardware configuration of the cluster.

Configurations

External register CLUSTERCFR2 bits [63:0] are architecturally mapped to AArch64 System register [A.1.18 IMP\\_CLUSTERCFR2\\_EL1, Cluster Configuration Register 2](#) on page 307 bits [63:0].

Attributes

Width

64

Component

Cluster

Register offset

0x0068

Access type

RO

Reset value

0000	0000	0000	0000	0000	00xx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-12: ext\_clustercfr2 bit assignments

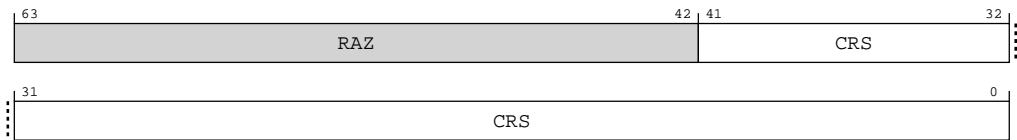


Table B-24: CLUSTERCFR2 bit descriptions

Bits	Name	Description	Reset
[63:42]	RAZ	Reserved	RAZ
[41:0]	CRS	Core register slices. Each three bits represents a core, with [2:0] for core 0 up to [41:39] for core 13.  0b000 No register slices  0b001 One register slice  0b00010 Two register slices  0b00011 Three register slices  0b00100 Four register slices  0b00101 Five register slices  0b00110 Six register slices  0b00111 Seven register slices	42 { x }

Accessibility

Component	Offset	Instance	Range
Cluster	0x0068	CLUSTERCFR2	None

This interface is accessible as follows:

RO

B.1.1.13 CLUSTERPPMCR, Cluster PPM Control Register

Provides controls to enable/disable pin control to MPMM gears.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0080

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 0000  
0000

Bit descriptions

Figure B-13: ext\_clusterppmcr bit assignments

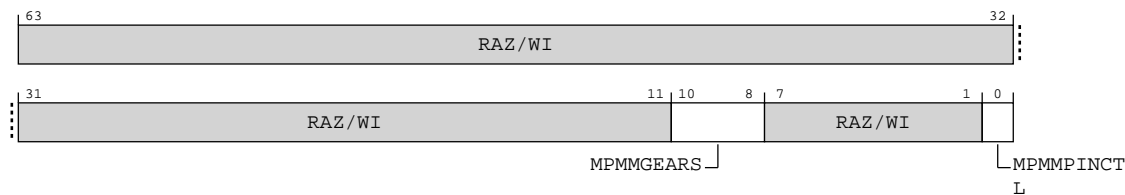


Table B-26: CLUSTERPPMCR bit descriptions

Bits	Name	Description	Reset
[63:11]	RAZ/WI	Reserved	RAZ/WI
[10:8]	MPMMGEARS	Number of MPMM gears implemented.  <b>0b100</b> Four MPMM gears implemented  Access to this field is: RO	0b100
[7:1]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[0]	MPMMPINCTL	MPMM pin control enable.  <b>0b0</b> MPMM controlled by the CLUSTERMPMMCR register, the external pins are ignored  <b>0b1</b> MPMM controlled by external pins, the CLUSTERMPMMCR register is ignored	0b0

Accessibility

Component	Offset	Instance	Range
Cluster	0x0080	CLUSTERPPMCR	None

This interface is accessible as follows:

RW

B.1.1.14 CLUSTERMPMMCR, Cluster MPMM Control Register

Provides controls to enable/disable MPMM and configure the MPMM gears.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0088

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-14: ext\_clustermppmmcr bit assignments

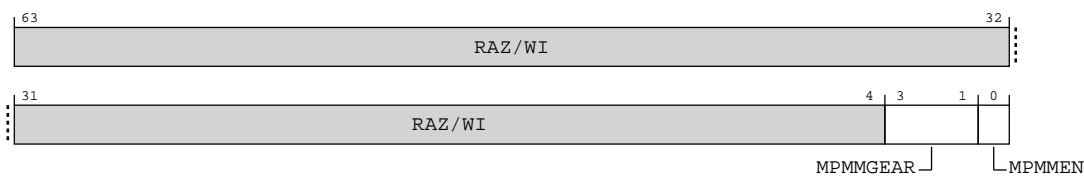


Table B-28: CLUSTERMPMMCR bit descriptions

Bits	Name	Description	Reset
[63:4]	RAZ/WI	Reserved	RAZ/WI
[3:1]	MPMMGEAR	MPMM Gear select.	0b000
[0]	MPMMEN	Enable MPMM.  0b0 MPMM disabled  0b1 MPMM enabled	0b0

Accessibility

Component	Offset	Instance	Range
Cluster	0x0088	CLUSTERMPMMCR	None

This interface is accessible as follows:

RW

B.1.1.15 CLUSTERL3UPTH2, Cluster L3 Upsize Threshold2 Register

This register is intended for use in algorithms for determining when to power up slices.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0090

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-15: ext\_clusterl3upth2 bit assignments

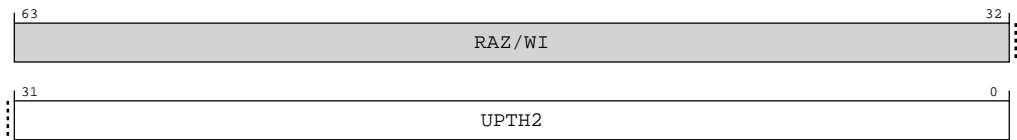


Table B-30: CLUSTERL3UPTH2 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	UPTH2	If all the L3 ways are powered, but not all of the slices are powered, and the cache miss bandwidth rises above this threshold then the number of slices is upsized. The value in this register is compared with the change in the cluster L3 miss counter since the last time period.	0x00000000

Accessibility

Component	Offset	Instance	Range
Cluster	0x0090	CLUSTERL3UPTH2	None

This interface is accessible as follows:

RW

B.1.2 External MPAM registers summary

The cluster *Memory System Resource Partitioning and Monitoring* (MPAM) registers are only accessible from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the cluster MPAM registers that are accessed externally (memory-mapped) from the utility bus of the DSU-120AE. For more information about a register, click on the register name in the table.



Note

- The cluster MPAM registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- The base address for the cluster 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-32: MPAM registers summary**

Offset	Name	Reset	Width	Description
0x0000	MPAMF_IDR	See individual bit resets.	64-bit	MPAM Features Identification Register
0x0000	MPAMF_IDR	See individual bit resets.	64-bit	MPAM Features Identification Register
0x0008	MPAMF_SIDR	See individual bit resets.	32-bit	MPAM Features Secure Identification Register
0x0018	MPAMF_IIDR	See individual bit resets.	32-bit	MPAM Implementation Identification Register
0x0018	MPAMF_IIDR	See individual bit resets.	32-bit	MPAM Implementation Identification Register
0x0020	MPAMF_AIDR	See individual bit resets.	32-bit	MPAM Architecture Identification Register
0x0020	MPAMF_AIDR	See individual bit resets.	32-bit	MPAM Architecture Identification Register
0x0030	MPAMF_CPOR_IDR	See individual bit resets.	32-bit	MPAM Features Cache Portion Partitioning ID register
0x0030	MPAMF_CPOR_IDR	See individual bit resets.	32-bit	MPAM Features Cache Portion Partitioning ID register
0x0040	MPAMF_MBW_IDR	See individual bit resets.	32-bit	MPAM Memory Bandwidth Partitioning Identification Register
0x0040	MPAMF_MBW_IDR	See individual bit resets.	32-bit	MPAM Memory Bandwidth Partitioning Identification Register
0x00F0	MPAMF_ECR	See individual bit resets.	32-bit	MPAM Error Control Register
0x00F0	MPAMF_ECR	See individual bit resets.	32-bit	MPAM Error Control Register
0x00F8	MPAMF_ESR	See individual bit resets.	32-bit	MPAM Error Status Register
0x00F8	MPAMF_ESR	See individual bit resets.	32-bit	MPAM Error Status Register
0x0100	MPAMCFG_PART_SEL	See individual bit resets.	32-bit	MPAM Partition Configuration Selection Register
0x0100	MPAMCFG_PART_SEL	See individual bit resets.	32-bit	MPAM Partition Configuration Selection Register
0x0500	MPAMCFG_MBW_PROP_ns	See individual bit resets.	32-bit	MPAM Memory Bandwidth Proportional Stride Partition Configuration for Non-Secure PARTIDs
0x0500	MPAMCFG_MBW_PROP_s	See individual bit resets.	32-bit	MPAM Memory Bandwidth Proportional Stride Partition Configuration for Secure PARTIDs
0x1000	MPAMCFG_CPBM_ns	See individual bit resets.	32-bit	MPAM Cache Portion Bitmap Partition Configuration Register for Non-secure PARTIDs
0x1000	MPAMCFG_CPBM_s	See individual bit resets.	32-bit	MPAM Cache Portion Bitmap Partition Configuration Register for Secure PARTIDs

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

64

Component

MPAM

Register offsets (2)

0x0000,0x0000

Access type

RO

Reset value

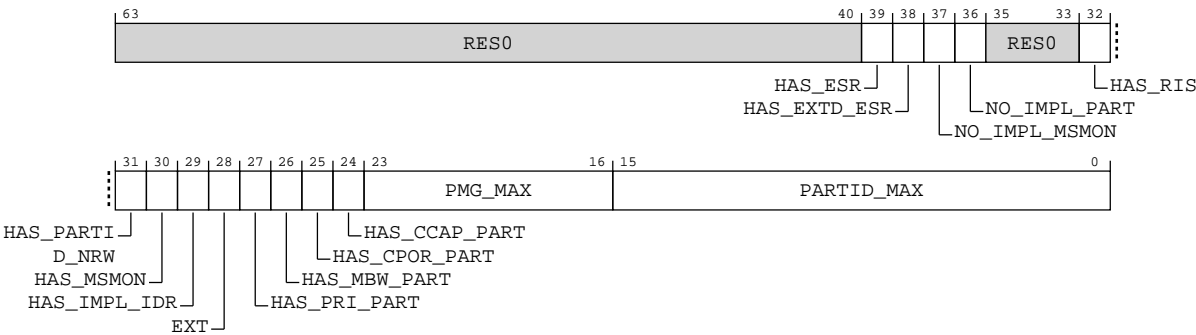
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	1011	xxx0	0001	0110	0000	0001	0000	0000	0011	1111
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-16: ext\_mpamf\_idr bit assignments



**Table B-33: MPAMF\_IDR bit descriptions**

Bits	Name	Description	Reset
[63:40]	RES0	Reserved	RES0
[39]	HAS_ESR	Has 32-bit ESR. <b>0b1</b> Has a 32-bit ESR register implemented.	0b1
[38]	HAS_EXTD_ESR	Support for extended ESR. <b>0b0</b> Doesn't support an extended ESR register.	0b0
[37]	NO_IMPL_MSMON	Support for IMPL_MSMON register. <b>0b1</b> Doesn't have an IMPL_MSMON register implemented.	0b1
[36]	NO_IMPL_PART	Support for IMPL_PART register. <b>0b1</b> Doesn't have an IMPL_PART register implemented.	0b1
[35:33]	RES0	Reserved	RES0
[32]	HAS_RIS	Support for resource instance selection. <b>0b0</b> Doesn't support resource instance selection.	0b0
[31]	HAS_PARTID_NRW	Has PARTID narrowing. <b>0b0</b> Does not have ext-MPAMF_PARTID_NRW_IDR, ext-MPAMCFG_INTPARTID or intPARTID mapping support.	0b0
[30]	HAS_MSMON	Has resource monitors. Indicates whether this MSC has MPAM resource monitors. <b>0b0</b> 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. <b>0b0</b> Does not have ext-MPAMF_IMPL_IDR.	0b0
[28]	EXT	IDR is 64-bit. <b>0b1</b> IDR is 64-bit.	0b1
[27]	HAS_PRI_PART	Has priority partitioning. Indicates whether this MSC implements MPAM priority partitioning and ext-MPAMF_PRI_IDR. <b>0b0</b> 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. <b>0b1</b> Does support memory bandwidth partitioning features and has ext-MPAMF_MBW_IDR register.	0b1

Bits	Name	Description	Reset
[25]	HAS_CPOR_PART	Has cache portion partitioning. Indicates whether this MSC implements MPAM cache portion partitioning and ext-MPAMF_CPOR_IDR.  <b>0b1</b> 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.  <b>0b0</b> 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.  <b>0b00000001</b> Supports 2 Non-secure PMGs.	0x01
[15:0]	PARTID_MAX	Maximum value of Non-secure PARTID supported by this component.  <b>0b0000000000011111</b> Supports 64 Non-secure PARTIDs.	0x003F

## Accessibility

Component	Offset	Instance	Range
MPAM	0x0000	MPAMF_IDR_s	None

This interface is accessible as follows:

RO

Component	Offset	Instance	Range
MPAM	0x0000	MPAMF_IDR_ns	None

This interface is accessible as follows:

RO

### 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

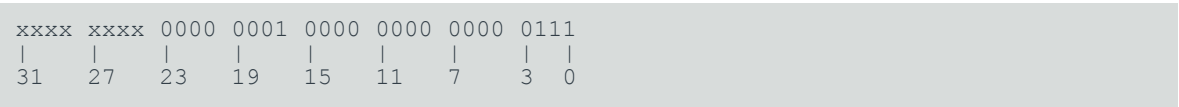
Register offset

0x0008

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-17: ext\_mpamf\_sidr bit assignments

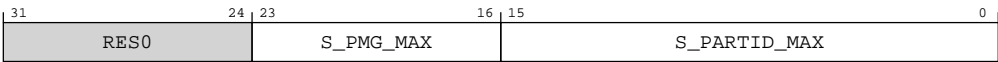


Table B-36: MPAMF\_SIDR bit descriptions

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.  0b000000000000000111 Supports 8 Secure PARTIDs.	0x0007

Accessibility

Component	Offset	Instance	Range
MPAM	0x0008	MPAMF_SIDR_s	None

This interface is accessible as follows:

RO

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 1100 0000 0001 0100 0011 1011

Bit descriptions

Figure B-18: ext\_mpamf\_iidr bit assignments

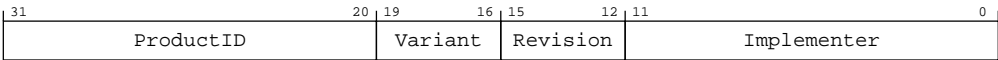


Table B-38: MPAMF\_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Value identifying the MPAM Memory System Component.  0b010011101100 DSU-120AE Cluster MPAM.	0x4EC
[19:16]	Variant	Value used to distinguish product variants, or major revisions of the product.  0b0000 Product variant 0.	0b0000
[15:12]	Revision	Value used to distinguish minor revisions of the product.  0b0000 Product revision 0.  0b0001 Product revision 1.	0b0001



Bits	Name	Description	Reset
[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.  <b>0b010000111011</b> Arm implementation.	0x43B

Accessibility

Component	Offset	Instance	Range
MPAM	0x0018	MPAMF_IIDR	None

This interface is accessible as follows:

RO

Component	Offset	Instance	Range
MPAM	0x0018	MPAMF_IIDR	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-19: ext\_mpamf\_aidr bit assignments

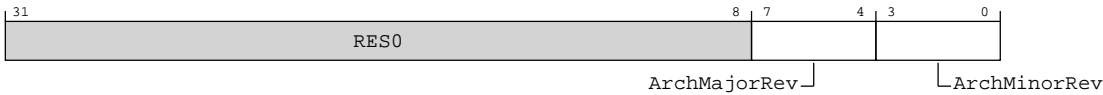


Table B-41: MPAMF\_AIDR bit descriptions

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.  0b0001 MPAM minor version 1.	0b0001

Accessibility

Component	Offset	Instance	Range
MPAM	0x0020	MPAMF_AIDR	None

This interface is accessible as follows:

RO

Component	Offset	Instance	Range
MPAM	0x0020	MPAMF_AIDR	None

This interface is accessible as follows:

RO

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

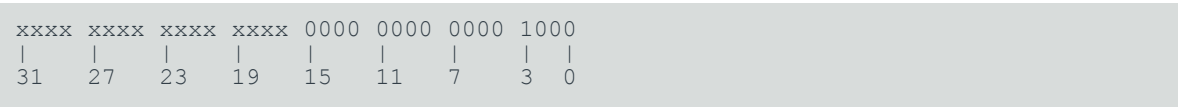
Register offsets (2)

0x0030,0x0030

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-20: ext\_mpamf\_cpor\_idr bit assignments

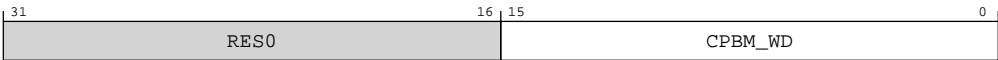


Table B-44: MPAMF\_CPOR\_IDR bit descriptions

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.  0b00000000000001000 Supports 8 cache portion partitioning bits.	0x0008

Accessibility

Component	Offset	Instance	Range
MPAM	0x0030	MPAMF_CPOR_IDR_s	None

This interface is accessible as follows:

RO

Component	Offset	Instance	Range
MPAM	0x0030	MPAMF_CPOR_IDR_ns	None

This interface is accessible as follows:

RO

B.1.2.6 MPAMF\_MBW\_IDR, MPAM Memory Bandwidth Partitioning Identification Register

Indicates which MPAM bandwidth partitioning features are present on this MSC.  
MPAMF\_MBW\_IDR\_s indicates bandwidth partitioning features accessed from the Secure MPAM feature page. MPAMF\_MBW\_IDR\_ns indicates bandwidth partitioning 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)

0x0040,0x0040

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	x010	00xx	xx00	0110
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-21: ext\_mpamf\_mbw\_idr bit assignments

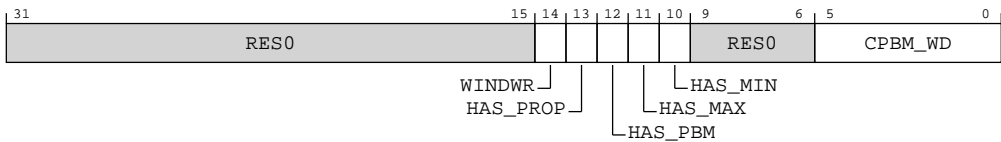


Table B-47: MPAMF\_MBW\_IDR bit descriptions

Bits	Name	Description	Reset
[31:15]	RES0	Reserved	RES0
[14]	WINDWR	The bandwidth accounting period <b>0b0</b> The bandwidth accounting period is not writeable.	0b0
[13]	HAS_PROP	Proportional stride bandwidth partitioning <b>0b1</b> Supports proportional stride bandwidth partitioning.	0b1
[12]	HAS_PBM	Bandwidth portion partitioning <b>0b0</b> Does not support bandwidth portion partitioning.	0b0
[11]	HAS_MAX	Maximum bandwidth partitioning <b>0b0</b> Does not support maximum bandwidth partitioning.	0b0
[10]	HAS_MIN	Minimum bandwidth partitioning <b>0b0</b> Does not support minimum bandwidth partitioning.	0b0
[9:6]	RES0	Reserved	RES0
[5:0]	CPBM_WD	Number of implemented bits in the bandwidth allocation fields <b>0b000110</b> Supports 6 bits in the cache bandwidth allocation fields.	0b000110

Accessibility

Component	Offset	Instance	Range
MPAM	0x0040	MPAMF_MBW_IDR_s	None

This interface is accessible as follows:

RO

Component	Offset	Instance	Range
MPAM	0x0040	MPAMF_MBW_IDR_ns	None

This interface is accessible as follows:

RO

B.1.2.7 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-22: ext\_mpamf\_ecr bit assignments

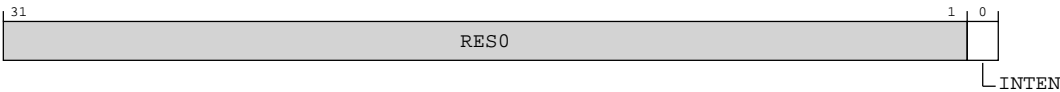


Table B-50: MPAMF\_ECR bit descriptions

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

Accessibility

Component	Offset	Instance	Range
MPAM	0x00F0	MPAMF_ECR_s	None

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
MPAM	0x00F0	MPAMF_ECR_ns	None

This interface is accessible as follows:

RW

B.1.2.8 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-23: ext\_mpamf\_esr bit assignments

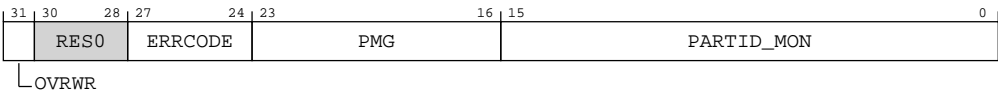


Table B-53: MPAMF\_ESR bit descriptions

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	<p>Error code.</p> <p><b>0b0000</b> No error.</p> <p><b>0b0001</b> PARTID_SEL_Range.</p> <p><b>0b0010</b> Req_PARTID_Range.</p> <p><b>0b0011</b> MSMONCFG_ID_RANGE.</p> <p><b>0b0100</b> Req_PMG_Range.</p> <p><b>0b0101</b> Monitor_Range.</p> <p><b>0b0110</b> intPARTID_Range.</p> <p><b>0b0111</b> Unexpected_INTERNAL.</p> <p><b>0b1000</b> Reserved.</p> <p><b>0b1001</b> Reserved.</p> <p><b>0b1010</b> Reserved.</p> <p><b>0b1011</b> Reserved.</p> <p><b>0b1100</b> Reserved.</p> <p><b>0b1101</b> Reserved.</p> <p><b>0b1110</b> Reserved.</p> <p><b>0b1111</b> Reserved.</p>	xxxx
[23:16]	PMG	<p>Program monitoring group.</p> <p>Set to the PMG on an error that captures PMG. Otherwise, set to 0x00 on an error that does not capture PMG.</p>	8 {x}
[15:0]	PARTID_MON	<p>PARTID or monitor.</p> <p>Set to the PARTID on an error that captures PARTID.</p> <p>Set to the monitor index on an error that captures MON.</p> <p>On an error that captures neither PARTID nor MON, this field is set to 0x0000.</p>	16 {x}

Accessibility

Component	Offset	Instance	Range
MPAM	0x00F8	MPAMF_ESR_s	None

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
MPAM	0x00F8	MPAMF_ESR_ns	None

This interface is accessible as follows:

RW

B.1.2.9 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

See bit descriptions

Reset value

xxxx	xxxx	xxxx	xxx0	xxxx	xxxx	xxxx	xxxx
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-24: ext\_mpamcfg\_part\_sel bit assignments

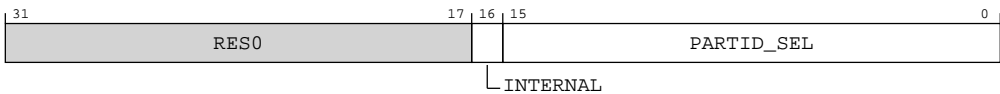


Table B-56: MPAMCFG\_PART\_SEL bit descriptions

Bits	Name	Description	Reset
[31:17]	RES0	Reserved	RES0
[16]	INTERNAL	Internal PARTID. This field is <b>RAZ/WI</b> .  <b>0b0</b>  PARTID_SEL is interpreted as a request PARTID and ignored except for use with ext-MPAMCFG_INTPARTID register access.  Access to this field is: <b>RAZ/WI</b>	0b0
[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}

Accessibility

Component	Offset	Instance	Range
MPAM	0x0100	MPAMCFG_PART_SEL_s	None

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
MPAM	0x0100	MPAMCFG_PART_SEL_ns	None

This interface is accessible as follows:

RW

#### B.1.2.10 MPAMCFG\_MBW\_PROP\_ns, MPAM Memory Bandwidth Proportional Stride Partition Configuration for Non-Secure PARTIDs

Controls the proportional stride of memory bandwidth that the PARTID selected by MPAMCFG\_PART\_SEL uses. MPAMCFG\_MBW\_PROP\_ns controls the bandwidth proportional stride for the Secure PARTID selected by the Secure instance of MPAMCFG\_PART\_SEL. MPAMCFG\_MBW\_PROP\_ns controls the bandwidth proportional stride for the Non-secure PARTID selected by the Non-secure instance of MPAMCFG\_PART\_SEL.

Proportional stride is a relative cost of bandwidth requested by one PARTID in relation to the costs of the bandwidths requested by each other PARTID also competing to use the bandwidth.

## Configurations

This register is available in all configurations.

## Attributes

## Width

32

## Component

MPAM

## Register offset

0x0500

## Access type

RW

## Reset value

0xxx	xxxx	xxxx	xxxx	xxxx	xxxx	xx00	0000
31	27	23	19	15	11	7	3 0



### Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-25: ext\_mpamcfg\_mbw\_prop\_ns bit assignments**



**Table B-59: MPAMCFG\_MBW\_PROP\_ns bit descriptions**

Bits	Name	Description	Reset
[31]	EN	Enable proportional stride bandwidth partitioning.  <b>0b0</b> The selected partition is not regulated by proportional stride bandwidth partitioning.  <b>0b1</b> The selected partition has bandwidth usage regulated by proportional stride bandwidth partitioning as controlled by STRIDEM1.	0b0
[30:6]	RES0	Reserved	RES0
[5:0]	STRIDEM1	Memory bandwidth stride minus 1 allocated to the partition selected by MPAMCFG_PART_SEL. STRIDEM1 represents the normalized cost of bandwidth consumption by the partition. The default value of 0 gives the maximum fair share of the bandwidth available to this partition. Larger values in this field indicate that this partition should receive a lower share of the overall bandwidth, relative to other partitions that have smaller values in this field.	0b000000

### Accessibility

Component	Offset	Instance	Range
MPAM	0x0500	MPAMCFG_MBW_PROP_ns	None

This interface is accessible as follows:

RW

#### B.1.2.11 MPAMCFG\_MBW\_PROP\_s, MPAM Memory Bandwidth Proportional Stride Partition Configuration for Secure PARTIDs

Controls the proportional stride of memory bandwidth that the PARTID selected by MPAMCFG\_PART\_SEL uses. MPAMCFG\_MBW\_PROP\_s controls the bandwidth proportional stride for the Secure PARTID selected by the Secure instance of MPAMCFG\_PART\_SEL. MPAMCFG\_MBW\_PROP\_ns controls the bandwidth proportional stride for the Non-secure PARTID selected by the Non-secure instance of MPAMCFG\_PART\_SEL.

Proportional stride is a relative cost of bandwidth requested by one PARTID in relation to the costs of the bandwidths requested by each other PARTID also competing to use the bandwidth.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32

#### Component

MPAM

Register offset

0x0500

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-26: ext\_mpamcfg\_mbw\_prop\_s bit assignments



Table B-61: MPAMCFG\_MBW\_PROP\_s bit descriptions

Bits	Name	Description	Reset
[31]	EN	Enable proportional stride bandwidth partitioning.  <b>0b0</b> The selected partition is not regulated by proportional stride bandwidth partitioning.  <b>0b1</b> The selected partition has bandwidth usage regulated by proportional stride bandwidth partitioning as controlled by STRIDEM1.	0b0
[30:6]	RES0	Reserved	RES0
[5:0]	STRIDEM1	Memory bandwidth stride minus 1 allocated to the partition selected by MPAMCFG_PART_SEL. STRIDEM1 represents the normalized cost of bandwidth consumption by the partition. The default value of 0 gives the maximum fair share of the bandwidth available to this partition. Larger values in this field indicate that this partition should receive a lower share of the overall bandwidth, relative to other partitions that have smaller values in this field.	0b000000

Accessibility

Component	Offset	Instance	Range
MPAM	0x0500	MPAMCFG_MBW_PROP_s	None

This interface is accessible as follows:

RW

B.1.2.12 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-27: ext\_mpamcfg\_cpbm\_ns bit assignments

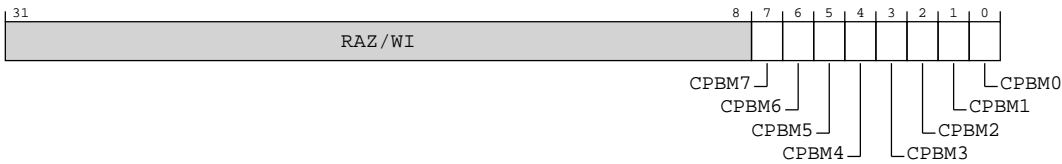


Table B-63: MPAMCFG\_CPBM\_ns bit descriptions

Bits	Name	Description	Reset
[31:8]	RAZ/WI	Reserved	RAZ/ WI

Bits	Name	Description	Reset
[7]	CPBM7	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b> The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b> The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[6]	CPBM6	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b> The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b> The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[5]	CPBM5	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b> The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b> The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[4]	CPBM4	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b> The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b> The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[3]	CPBM3	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b> The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b> The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1



Bits	Name	Description	Reset
[2]	CPBM2	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n.  <b>0b0</b> The PARTID is not permitted to allocate into cache portion n.  <b>0b1</b> 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.  <b>0b0</b> The PARTID is not permitted to allocate into cache portion n.  <b>0b1</b> 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.  <b>0b0</b> The PARTID is not permitted to allocate into cache portion n.  <b>0b1</b> 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

## Accessibility

Component	Offset	Instance	Range
MPAM	0x1000	MPAMCFG_CPBM_ns	None

This interface is accessible as follows:

RW

### B.1.2.13 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-28: ext\_mpamcfg\_cpbm\_s bit assignments

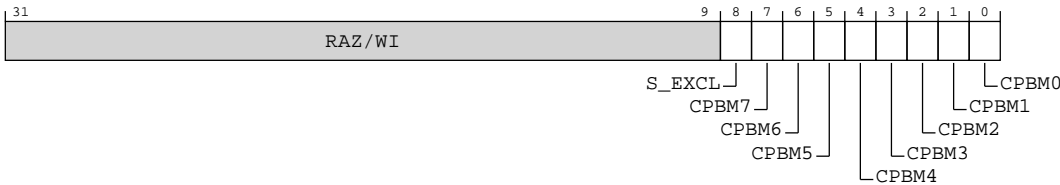


Table B-65: MPAMCFG\_CPBM\_s bit descriptions

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.  <b>0b0</b> Each set MPAMCFG_CPBM_s bit has no effect on the corresponding MPAMCFG_CPBM_ns bit.  <b>0b1</b> 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.  <b>0b0</b> The PARTID is not permitted to allocate into cache portion n.  <b>0b1</b> 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
[6]	CPBM6	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b></p> <p>The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b></p> <p>The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[5]	CPBM5	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b></p> <p>The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b></p> <p>The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[4]	CPBM4	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b></p> <p>The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b></p> <p>The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[3]	CPBM3	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b></p> <p>The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b></p> <p>The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1
[2]	CPBM2	<p>Each bit, CPBM&lt;n&gt;, grants permission to the PARTID to allocate cache lines within cache portion n.</p> <p><b>0b0</b></p> <p>The PARTID is not permitted to allocate into cache portion n.</p> <p><b>0b1</b></p> <p>The PARTID is permitted to allocate within cache portion n.</p> <p>The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.</p>	0b1

Bits	Name	Description	Reset
[1]	CPBM1	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n.  <b>0b0</b> The PARTID is not permitted to allocate into cache portion n.  <b>0b1</b> 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.  <b>0b0</b> The PARTID is not permitted to allocate into cache portion n.  <b>0b1</b> 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

## Accessibility

Component	Offset	Instance	Range
MPAM	0x1000	MPAMCFG_CPBM_s	None

This interface is accessible as follows:

RW

## B.1.3 External cluster RAS registers summary

The cluster RAS registers are accessible either from memory-mapped accesses on the utility bus or from System register accesses from the cores.

The summary table provides an overview of all the cluster RAS registers in DSU-120AE. For more information about a register, click on the register name in the table.



Note

- The cluster RAS registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is 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-67: CLUSTERRAS registers summary**

Offset	Name	Reset	Width	Description
0x000	<a href="#">CLUSTERRAS_ERROFR</a>	See individual bit resets.	64-bit	Error Record Feature Register
0x008	<a href="#">CLUSTERRAS_ERROCTLR</a>	See individual bit resets.	64-bit	Error Record Control Register
0x010	<a href="#">CLUSTERRAS_ERROSTATUS</a>	See individual bit resets.	64-bit	Error Record Primary Status Register
0x018	<a href="#">CLUSTERRAS_ERROADDR</a>	See individual bit resets.	64-bit	Error Record Address Register
0x020	<a href="#">CLUSTERRAS_ERRROMISCO</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 0
0x028	<a href="#">CLUSTERRAS_ERRROMISC1</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 1
0x030	<a href="#">CLUSTERRAS_ERRROMISC2</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 2
0x038	<a href="#">CLUSTERRAS_ERRROMISC3</a>	See individual bit resets.	64-bit	Error Record Miscellaneous Register 3
0x800	<a href="#">CLUSTERRAS_ERROPFGF</a>	See individual bit resets.	64-bit	Pseudo-fault Generation Feature Register
0x808	<a href="#">CLUSTERRAS_ERROPFGCTL</a>	See individual bit resets.	64-bit	Pseudo-fault Generation Control Register
0x810	<a href="#">CLUSTERRAS_ERROPFGCDN</a>	See individual bit resets.	64-bit	Pseudo-fault Generation Countdown Register
0xE00	<a href="#">CLUSTERRAS_ERRGSR</a>	See individual bit resets.	64-bit	Error Group Status Register
0xE10	<a href="#">CLUSTERRAS_ERRIIDR</a>	See individual bit resets.	32-bit	Implementation Identification Register
0xFA8	<a href="#">CLUSTERRAS_ERRDEVAFF</a>	See individual bit resets.	64-bit	Device Affinity Register
0xFBC	<a href="#">CLUSTERRAS_ERRDEVARCH</a>	See individual bit resets.	32-bit	Device Architecture Register
0xFC8	<a href="#">CLUSTERRAS_ERRDEVID</a>	See individual bit resets.	32-bit	Device Configuration Register
0xFD0	<a href="#">CLUSTERRAS_ERRPIDR4</a>	See individual bit resets.	32-bit	Peripheral Identification Register 4
0xFD4	<a href="#">CLUSTERRAS_ERRPIDR5</a>	See individual bit resets.	32-bit	Peripheral Identification Register 5
0xFD8	<a href="#">CLUSTERRAS_ERRPIDR6</a>	See individual bit resets.	32-bit	Peripheral Identification Register 6
0xFDC	<a href="#">CLUSTERRAS_ERRPIDR7</a>	See individual bit resets.	32-bit	Peripheral Identification Register 7
0xFE0	<a href="#">CLUSTERRAS_ERRPIDR0</a>	See individual bit resets.	32-bit	Peripheral Identification Register 0
0xFE4	<a href="#">CLUSTERRAS_ERRPIDR1</a>	See individual bit resets.	32-bit	Peripheral Identification Register 1
0xFE8	<a href="#">CLUSTERRAS_ERRPIDR2</a>	See individual bit resets.	32-bit	Peripheral Identification Register 2
0xFEC	<a href="#">CLUSTERRAS_ERRPIDR3</a>	See individual bit resets.	32-bit	Peripheral Identification Register 3
0xFF0	<a href="#">CLUSTERRAS_ERRCIDR0</a>	See individual bit resets.	32-bit	Component Identification Register 0
0xFF4	<a href="#">CLUSTERRAS_ERRCIDR1</a>	See individual bit resets.	32-bit	Component Identification Register 1
0xFF8	<a href="#">CLUSTERRAS_ERRCIDR2</a>	See individual bit resets.	32-bit	Component Identification Register 2
0xFFC	<a href="#">CLUSTERRAS_ERRCIDR3</a>	See individual bit resets.	32-bit	Component Identification Register 3

### B.1.3.1 CLUSTERRAS\_ERROFR, 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

External register CLUSTERRAS\_ERROFR bits [63:0] are architecturally mapped to AArch64 System register [A.3.1 ERXFR\\_EL1, Selected Error Record Feature Register](#) on page 361 bits [63:0].

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x000

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xx00	10xx	0000	1010	1001	1010	0110
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-29: ext\_clusterras\_err0fr bit assignments

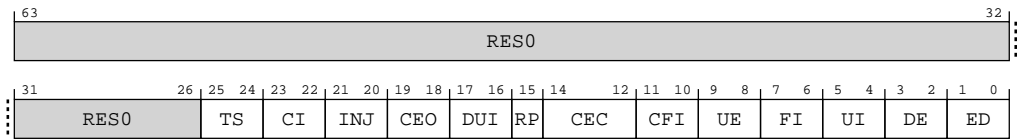


Table B-68: CLUSTERRAS\_ERR0FR bit descriptions

Bits	Name	Description	Reset
[63:26]	RES0	Reserved	RES0
[25:24]	TS	Timestamp Extension. Not implemented and treated as <b>RAZ/WI</b> . <b>0b00</b> 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. <b>0b10</b> 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	<p>Fault Injection Extension.</p> <p>Indicates whether the RAS Common Fault Injection Model Extension is implemented.</p> <p><b>0b00</b></p> <p>The node does not implement the RAS Common Fault Injection Model Extension.</p> <p><b>0b01</b></p> <p>The node implements the RAS Common Fault Injection Model Extension. See ext-CLUSTERRAS_ERROPFGF for more information.</p> <p>All other values are reserved.</p>	xx
[19:18]	CEO	<p>Corrected Error overwrite.</p> <p>Indicates the behavior when a second Corrected error is detected after a first Corrected error has been recorded by an error record &lt;m&gt; owned by the node.</p> <p><b>0b00</b></p> <p>Counts Corrected errors. Keeps the previous error syndrome. If the counter overflows then CLUSTERRAS_ERROSTATUS.OF is set to 1.</p> <p>All other values are reserved.</p>	0b00
[17:16]	DUI	<p>Error recovery interrupt for deferred errors.</p> <p>Indicates whether the node implements a control for enabling error recovery interrupts on deferred errors.</p> <p><b>0b00</b></p> <p>Does not support feature. ext-CLUSTERRAS_ERROCTL.R.DUI is <b>RES0</b>.</p> <p>All other values are reserved.</p>	0b00
[15]	RP	<p>Repeat counter.</p> <p>Indicates whether the node implements a repeat Corrected error counter in CLUSTERRAS_ERROMISCO.</p> <p><b>0b1</b></p> <p>A first (repeat) counter and a second (other) counter are implemented. The repeat counter is the same size as the primary error counter.</p>	0b1
[14:12]	CEC	<p>Corrected Error Counter.</p> <p>Indicates whether the node implements standard Corrected error counter (CE counter) mechanisms in CLUSTERRAS_ERROMISCO.</p> <p><b>0b010</b></p> <p>Implements an 8-bit Corrected error counter in CLUSTERRAS_ERROMISCO[39:32].</p> <p>All other values are reserved.</p>	0b010
[11:10]	CFI	<p>Fault handling interrupt for corrected errors.</p> <p>Indicates whether the node implements a control for enabling fault handling interrupts on corrected errors.</p> <p><b>0b10</b></p> <p>Feature is controllable using ext-CLUSTERRAS_ERROCTL.R.CFI.</p> <p>All other values are reserved.</p>	0b10

Bits	Name	Description	Reset
[9:8]	UE	<p>In-band uncorrected error reporting.</p> <p>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.</p> <p><b>0b01</b></p> <p>Feature always enabled. ext-CLUSTERRAS_ERROCTL.UE is <b>RES0</b>.</p>	0b01
[7:6]	FI	<p>Fault handling interrupt.</p> <p>Indicates whether the node implements a fault handling interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt.</p> <p><b>0b10</b></p> <p>Feature is controllable using ext-CLUSTERRAS_ERROCTL.FI.</p>	0b10
[5:4]	UI	<p>Error recovery interrupt for uncorrected errors.</p> <p>Indicates whether the node implements an error recovery interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt.</p> <p><b>0b10</b></p> <p>Feature is controllable using ext-CLUSTERRAS_ERROCTL.UI.</p>	0b10
[3:2]	DE	<p>Deferred error enable.</p> <p><b>0b01</b></p> <p>Deferred errors is always enabled.</p>	0b01
[1:0]	ED	<p>Error reporting and logging.</p> <p>Indicates this is the first record owned by the cluster. The cluster implements controls for enabling and disabling error reporting and logging.</p> <p><b>0b10</b></p> <p>Feature is controllable using ext-CLUSTERRAS_ERROCTL.ED.</p> <p>The value 0b11 is reserved.</p>	0b10

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x000	ERROFR	None

This interface is accessible as follows:

RO

### B.1.3.2 CLUSTERRAS\_ERROCTL, 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

External register CLUSTERRAS\_ERR0CTLR bits [63:0] are architecturally mapped to AArch64 System register [A.3.2 ERXCTLR\\_EL1, Selected Error Record Control Register](#) on page 364 bits [63:0].

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	xx0x	x0x0	xxx0	0001
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-30: ext\_clusterras\_err0ctlr bit assignments

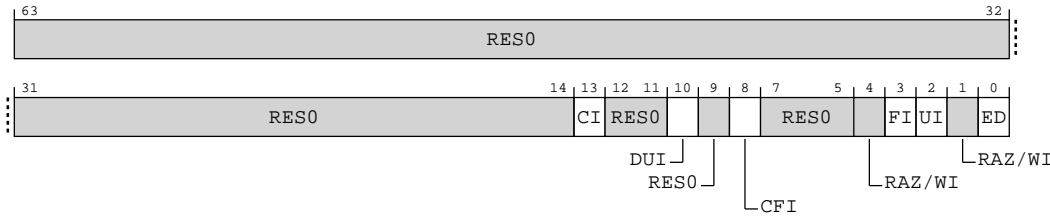


Table B-70: CLUSTERRAS\_ERR0CTLR bit descriptions

Bits	Name	Description	Reset
[63:14]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[13]	CI	<p>Critical error interrupt enable.</p> <p>When enabled, the critical error interrupt is generated for a critical error condition.</p> <p><b>0b0</b></p> <p>Critical error interrupt not generated for critical errors. Critical errors are treated as Uncontained errors.</p> <p><b>0b1</b></p> <p>Critical error interrupt generated for critical errors.</p>	0b0
[12:11]	RES0	Reserved	RES0
[10]	DUI	<p>Error recovery interrupt for deferred errors enable. This control applies to errors arising from both reads and writes.</p> <p>When enabled, an error recovery interrupt is generated for all detected Deferred errors.</p> <p><b>0b0</b></p> <p>Error recovery interrupt not generated for deferred errors.</p> <p>The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.</p> <p>Access to this field is: RO</p>	0b0
[9]	RES0	Reserved	RES0
[8]	CFI	<p>Fault handling interrupt for Corrected errors enable. This control applies to errors arising from both reads and writes.</p> <p>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&lt;n&gt;MISCO.</p> <p><b>0b0</b></p> <p>Fault handling interrupt not generated for Corrected errors.</p> <p><b>0b1</b></p> <p>Fault handling interrupt generated for Corrected errors.</p> <p>The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.</p>	0b0
[7:5]	RES0	Reserved	RES0
[4]	RAZ/ WI	Reserved	RAZ/ WI
[3]	FI	<p>Fault handling interrupt enable. This control applies to errors arising from both reads and writes.</p> <p>When enabled, the fault handling interrupt is generated for all detected Corrected errors, Deferred errors, and Uncorrected errors.</p> <p><b>0b0</b></p> <p>Fault handling interrupt disabled.</p> <p><b>0b1</b></p> <p>Fault handling interrupt enabled.</p> <p>The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.</p>	0b0

Bits	Name	Description	Reset
[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.  <b>0b0</b> Error recovery interrupt disabled.  <b>0b1</b> 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.	0b0
[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.  <b>0b0</b> Error reporting disabled.  <b>0b1</b> Error reporting enabled.	0b1

### Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x008	ERROCTLR	None

This interface is accessible as follows:

RW

#### B.1.3.3 CLUSTERRAS\_ERROSTATUS, 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 requester.
- 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

External register CLUSTERRAS\_ERROSTATUS bits [63:0] are architecturally mapped to AArch64 System register [A.3.3 ERXSTATUS\\_EL1, Selected Error Record Primary Status Register](#) on page 368 bits [63:0].

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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0

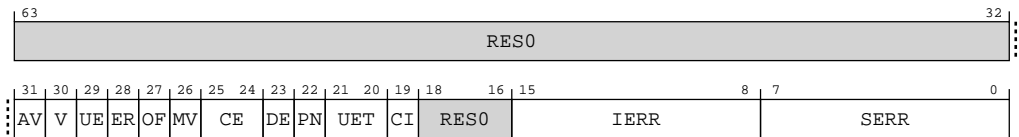


Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-31: ext\_clusterras\_err0status bit assignments



**Table B-72: CLUSTERRAS\_ERR0STATUS bit descriptions**

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31]	AV	<p>Address Valid.</p> <p><b>0b0</b> ext-CLUSTERRAS_ERR0ADDR not valid.</p> <p>This bit is unimplemented and treated as <b>RAZ/WI</b>.</p>	0b0
[30]	V	<p>Status Register Valid.</p> <p><b>0b0</b> CLUSTERRAS_ERR0STATUS not valid.</p> <p><b>0b1</b> CLUSTERRAS_ERR0STATUS valid. At least one error has been recorded.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[29]	UE	<p>Uncorrected error.</p> <p><b>0b0</b> No errors have been detected, or all detected errors have been either corrected or deferred.</p> <p><b>0b1</b> At least one detected error was not corrected and not deferred.</p> <p>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.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERR0STATUS.V is set to 0.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[28]	ER	<p>Error Reported.</p> <p><b>0b0</b> No in-band error (External abort) reported.</p> <p>This bit is unimplemented and treated as <b>RAZ/WI</b>.</p>	0b0

Bits	Name	Description	Reset
[27]	OF	<p>Overflow.</p> <p>Indicates that multiple errors have been detected. This bit is set to 1 when one of the following occurs:</p> <ul style="list-style-type: none"> <li>A Corrected error is counted and the counter overflows.</li> <li>CLUSTERRAS_ERROSTATUS.V was previously set to 1 and a type of error other than a Corrected error is recorded.</li> </ul> <p>Otherwise, this bit is unchanged when an error is recorded.</p> <p>A direct write that modifies the counter overflow flag indirectly might set this bit to an <b>UNKNOWN</b> value.</p> <p>A direct write to this bit that clears this bit to zero might indirectly set the counter overflow flag to an <b>UNKNOWN</b> value.</p> <p><b>0b0</b></p> <p>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.</p> <p><b>0b1</b></p> <p>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.</p> <p>If this bit is nonzero, then software must write 1 to this bit, to clear this bit to zero, when clearing CLUSTERRAS_ERROSTATUS.V to 0.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERROSTATUS.V is set to 0.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[26]	MV	<p>Miscellaneous Registers (CLUSTERRAS_ERRROMISCO) Valid.</p> <p><b>0b0</b></p> <p>CLUSTERRAS_ERRROMISCO is not valid.</p> <p><b>0b1</b></p> <p>The contents of CLUSTERRAS_ERRROMISCO contains additional information for an error recorded by this record.</p> <p>Only CLUSTERRAS_ERRROMISCO is implemented. CLUSTERRAS_ERRROMISC1,2,3 are treated as <b>RAZ/WI</b>.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[25:24]	CE	<p>Corrected Error.</p> <p><b>0b00</b></p> <p>No errors were corrected.</p> <p><b>0b10</b></p> <p>At least one error was corrected.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this field is nonzero, then software must write ones to this field to clear this field to zero.</p> <p>If CLUSTERRAS_ERROSTATUS.V is set to 0, this field is not valid and reads <b>UNKNOWN</b>.</p> <p>This field is read/write-one-to-clear. Writing a value other than all-zeros or all-ones sets this field to an <b>UNKNOWN</b> value.</p>	0b00

Bits	Name	Description	Reset
[23]	DE	<p>Deferred Error.</p> <p><b>0b0</b> No errors were deferred.</p> <p><b>0b1</b> At least one error was not corrected and deferred.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero.</p> <p>If CLUSTERRAS_ERROSTATUS.V is set to 0, this bit is not valid and reads <b>UNKNOWN</b>.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[22]	PN	<p>Poison.</p> <p><b>0b0</b> Uncorrected error or Deferred error recorded because a corrupt value was detected.</p> <p><b>0b1</b> Uncorrected error or Deferred error recorded because a poison value was detected.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if any of the following are true:</p> <ul style="list-style-type: none"> <li>CLUSTERRAS_ERROSTATUS.V is set to 0.</li> <li>CLUSTERRAS_ERROSTATUS.{DE, UE} are both set to 0.</li> </ul> <p>This bit is read/write-one-to-clear.</p>	0b0
[21:20]	UET	<p>Uncorrected Error Type.</p> <p>Describes the state of the component after detecting or consuming an Uncorrected error.</p> <p><b>0b00</b> Uncorrected error, Uncontainable error (UC).</p> <p>This field is not implemented and is treated as <b>RAZ/WI</b>.</p>	0b00
[19]	CI	<p>Critical error.</p> <p>Indicates whether a critical error condition has been recorded.</p> <p><b>0b0</b> No critical error condition recorded.</p> <p><b>0b1</b> Critical error condition recorded.</p> <p>When clearing CLUSTERRAS_ERROSTATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero.</p> <p>This bit is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERROSTATUS.V is set to 0.</p> <p>This bit is read/write-one-to-clear.</p>	0b0
[18:16]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[15:8]	IERR	<p><b>IMPLEMENTATION DEFINED</b> Extended error code.</p> <p>Used with any primary error code SERR value. Additional information is placed in the CLUSTERRAS_ERRROMISCO register.</p> <p><b>0b00000000</b> If SERR == 0x7, indicates a Tag RAM error. Not used with other SERR values.</p> <p><b>0b00000010</b> If SERR == 0x7, indicates a Snoop Filter RAM error. Not used with other SERR values.</p> <p>This field is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERR0STATUS.V is set to 0.</p>	0x00
[7:0]	SERR	<p>Primary error code.</p> <p>Indicates the type of Primary error.</p> <p><b>0b00000000</b> No error.</p> <p><b>0b00000001</b> <b>IMPLEMENTATION DEFINED</b> error.</p> <p><b>0b00000010</b> Data value from (non-associative) internal memory. For example, ECC from on-chip SRAM or buffer.</p> <p><b>0b00000011</b> <b>IMPLEMENTATION DEFINED</b> pin. For example, nSEI pin.</p> <p><b>0b00000100</b> Assertion failure. For example, consistency failure.</p> <p><b>0b00000101</b> Error detected on internal data path. For example, parity on ALU result.</p> <p><b>0b00000110</b> Data value from associative memory. For example, ECC error on cache data.</p> <p><b>0b00000111</b> Address/control value from associative memory. For example, ECC error on cache tag.</p> <p><b>0b00001000</b> Data value from a TLB. For example, ECC error on TLB data.</p> <p><b>0b00001001</b> Address/control value from a TLB. For example, ECC error on TLB tag.</p>	0x00



Bits	Name	Description	Reset
[7:0] continued	SERR	<p><b>0b00001010</b> Data value from producer. For example, parity error on write data bus.</p> <p><b>0b00001011</b> Address/control value from producer. For example, parity error on address bus.</p> <p><b>0b00001100</b> Data value from (non-associative) external memory. For example, ECC error in SDRAM.</p> <p><b>0b00001101</b> Illegal address (software fault). For example, access to unpopulated memory.</p> <p><b>0b00001110</b> Illegal access (software fault). For example, byte write to word register.</p> <p><b>0b00001111</b> Illegal state (software fault). For example, device not ready.</p> <p><b>0b00010000</b> Internal data register. For example, parity on a SIMD&amp;FP register. For a PE, all general-purpose, stack pointer, SIMD&amp;FP, and SVE registers are data registers.</p> <p><b>0b00010001</b> Internal control register. For example, Parity on a System register. For a PE, all registers other than general-purpose, stack pointer, SIMD&amp;FP, and SVE registers are control registers.</p> <p><b>0b00010010</b> Error response from subordinate. For example, error response from cache write-back.</p> <p><b>0b00010011</b> External timeout. For example, timeout on interaction with another node.</p>	0x00
[7:0] continued	SERR	<p><b>0b00010100</b> Internal timeout. For example, timeout on interface within the node.</p> <p><b>0b00010101</b> Deferred error from subordinate not supported at requester. For example, poisoned data received from a subordinate by a requester that cannot defer the error further.</p> <p>All other values are reserved.</p> <p>This field is not valid and reads <b>UNKNOWN</b> if CLUSTERRAS_ERR0STATUS.V is set to 0.</p>	0x00

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x010	ERR0STATUS	None

This interface is accessible as follows:

RW

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

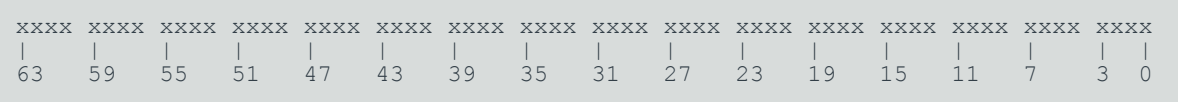
Register offset

0x018

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-32: ext\_clusterras\_err0addr bit assignments

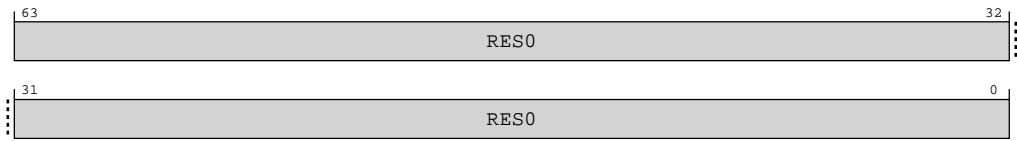


Table B-74: CLUSTERRAS\_ERR0ADDR bit descriptions

Bits	Name	Description	Reset
[63:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x018	ERR0ADDR	None

This interface is accessible as follows:

RO

B.1.3.5 CLUSTERRAS\_ERROMISCO, 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

External register CLUSTERRAS\_ERROMISCO bits [63:0] are architecturally mapped to AArch64 System register [A.3.7 ERXMISCO\\_EL1, Selected Error Record Miscellaneous Register 0](#) on page 385 bits [63:0].

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	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0

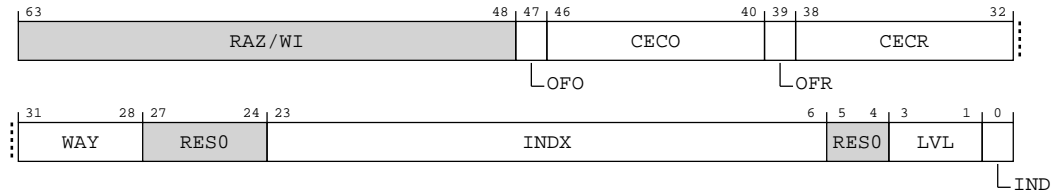


Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-33: ext\_clusterras\_err0misc0 bit assignments**



**Table B-76: CLUSTERRAS\_ERR0MISC0 bit descriptions**

Bits	Name	Description	Reset
[63:48]	RAZ/WI	Reserved	RAZ/WI
[47]	OFO	<p>Sticky overflow bit for Other errors.</p> <p>Set to 1 when the Corrected error count Other (CECO) field is incremented and wraps through zero.</p> <p><b>0b0</b></p> <p>Other counter has not overflowed.</p> <p><b>0b1</b></p> <p>Other counter has overflowed.</p> <p>A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an <b>UNKNOWN</b> value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an <b>UNKNOWN</b> value.</p>	x
[46:40]	CECO	<p>Corrected error count for Other errors.</p> <p>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.</p> <p>At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.</p>	7 {x}
[39]	OFR	<p>Sticky overflow bit for Repeat errors.</p> <p>Set to 1 when the Corrected error count Repeat (CECR) field is incremented and wraps through zero.</p> <p><b>0b0</b></p> <p>Repeat counter has not overflowed.</p> <p><b>0b1</b></p> <p>Repeat counter has overflowed.</p> <p>A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an <b>UNKNOWN</b> value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an <b>UNKNOWN</b> value.</p>	x

Bits	Name	Description	Reset
[38:32]	CECR	<p>Corrected error count for Repeat errors.</p> <p>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.</p> <p>The syndrome is recorded in the following fields:</p> <ul style="list-style-type: none"> <li>ext-CLUSTERRAS_ERROSTATUS.IERR</li> <li>ext-CLUSTERRAS_ERROSTATUS.SERR</li> <li>ext-CLUSTERRAS_ERRROMISCO.INDX</li> <li>ext-CLUSTERRAS_ERRROMISCO.WAY</li> </ul> <p>The syndrome is matched on a new Corrected error if all of the following are true:</p> <ul style="list-style-type: none"> <li>ext-CLUSTERRAS_ERROSTATUS.MV bit is set,</li> <li>ext-CLUSTERRAS_ERROSTATUS.IERR matches the new error,</li> <li>ext-CLUSTERRAS_ERROSTATUS.SERR matches the new error,</li> <li>ext-CLUSTERRAS_ERRROMISCO.INDX matches the new error,</li> <li>ext-CLUSTERRAS_ERRROMISCO.WAY matches the new error.</li> </ul> <p>CLUSTERRAS_ERROSTATUS.MV indicates the validity of the INDX and WAY fields of the CLUSTERRAS_ERRROMISCO register</p> <p>At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.</p>	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	<p>L3 Cache-level that contained the error. Always 0x2.</p> <p><b>0b010</b> L3 cache.</p>	xxx
[0]	IND	<p>L3 Cache instruction vs. data cache that contained the error. Always data (0x0).</p> <p><b>0b0</b> Data cache error.</p>	x

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x020	ERRROMISCO	None

This interface is accessible as follows:

RW

B.1.3.6 CLUSTERRAS\_ERR0MISC1, Error Record Miscellaneous Register 1

Unimplemented error syndrome register.

Configurations

External register CLUSTERRAS\_ERR0MISC1 bits [63:0] are architecturally mapped to AArch64 System register A.3.8 ERXMISC1\_EL1, Selected Error Record Miscellaneous Register 1 on page 389 bits [63:0].

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-34: ext\_clusterras\_err0misc1 bit assignments

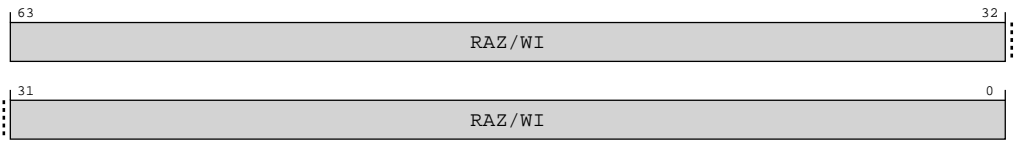


Table B-78: CLUSTERRAS\_ERR0MISC1 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x028	ERR0MISC1	None

This interface is accessible as follows:

RW

B.1.3.7 CLUSTERRAS\_ERR0MISC2, Error Record Miscellaneous Register 2

Unimplemented error syndrome register.

Configurations

External register CLUSTERRAS\_ERR0MISC2 bits [63:0] are architecturally mapped to AArch64 System register [A.3.9 ERXMISC2\\_EL1, Selected Error Record Miscellaneous Register 2](#) on page 391 bits [63:0].

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-35: ext\_clusterras\_err0misc2 bit assignments

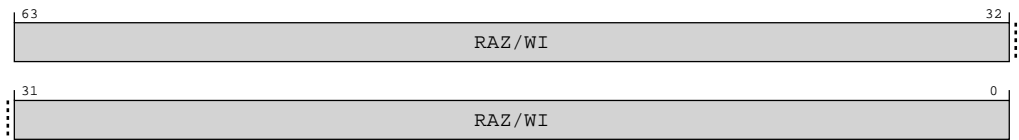


Table B-80: CLUSTERRAS\_ERR0MISC2 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x030	ERR0MISC2	None

This interface is accessible as follows:

RW

B.1.3.8 CLUSTERRAS\_ERR0MISC3, Error Record Miscellaneous Register 3

Unimplemented error syndrome register.

Configurations

External register CLUSTERRAS\_ERR0MISC3 bits [63:0] are architecturally mapped to AArch64 System register [A.3.10 ERXMISC3\\_EL1, Selected Error Record Miscellaneous Register 3](#) on page 393 bits [63:0].

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-36: ext\_clusterras\_err0misc3 bit assignments

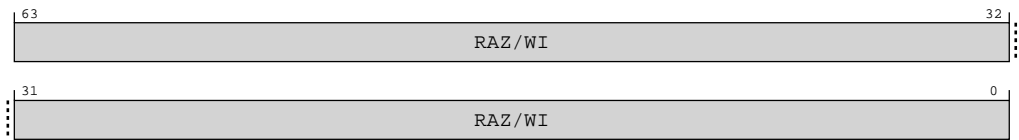


Table B-82: CLUSTERRAS\_ERR0MISC3 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x038	ERR0MISC3	None

This interface is accessible as follows:

RW



B.1.3.9 CLUSTERRAS\_ERROPFGF, Pseudo-fault Generation Feature Register

Defines which common architecturally-defined fault generation features are implemented.

Configurations

External register CLUSTERRAS\_ERROPFGF bits [63:0] are architecturally mapped to AArch64 System register [A.3.4 ERXPFGF\\_EL1, Selected Pseudo-fault Generation Feature Register](#) on page 375 bits [63:0].

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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-37: ext\_clusterras\_err0pfgf bit assignments

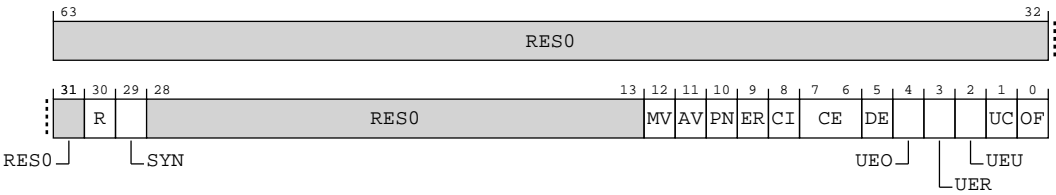


Table B-84: CLUSTERRAS\_ERROPFGF bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[30]	R	Restartable. Support for Error Generation Counter restart mode.  <b>0b1</b> Feature controllable.	0b1
[29]	SYN	Syndrome. Fault syndrome injection.  <b>0b1</b> When an injected error is recorded, the node does not update the ext-CLUSTERRAS_ERROSTATUS.{IERR, SERR} fields. ext-CLUSTERRAS_ERROSTATUS.{IERR, SERR} are writable when ext-CLUSTERRAS_ERROSTATUS.V == 0.  <b>Note:</b> Software can write intended values into the ext-CLUSTERRAS_ERROSTATUS.{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_ERR0MISCO register when an injected error is recorded.  CLUSTERRAS_ERR0MISC1-3 registers are reserved and unused for this purpose.  <b>0b1</b> When an injected error is recorded, the node does not update all the syndrome fields in CLUSTERRAS_ERR0MISCO.  The node records syndrome in CLUSTERRAS_ERR0MISCO OFO, CECO, OFR, CECR, WAY, INDX, LVL, and IND fields and sets ext-CLUSTERRAS_ERROSTATUS.MV to 1. CLUSTERRAS_ERR0PGFCTL.MV is <b>RAO</b> .  <b>Note:</b> Software can write intended values into the CLUSTERRAS_ERR0MISCO register when setting up a fault injection event.	0b1
[11]	AV	Address syndrome. Address syndrome injection. Always <b>RAZ/WI</b> .  <b>0b0</b> The node does not support ext-CLUSTERRAS_ERROADDR and does not set ext-CLUSTERRAS_ERROSTATUS.AV.	0b0
[10]	PN	Poison flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERROSTATUS.PN status flag.  <b>0b1</b> When an injected error is recorded, ext-CLUSTERRAS_ERROSTATUS.PN is set to ext-CLUSTERRAS_ERR0PGFCTL.PN.	0b1
[9]	ER	Error Reported flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERROSTATUS.ER status flag.  <b>0b0</b> When an injected error is recorded, the node does not set ext-CLUSTERRAS_ERROSTATUS.ER.  This bit reads-as-zero.	0b0

Bits	Name	Description	Reset
[8]	CI	<p>Critical Error flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.CI status flag.</p> <p><b>0b1</b></p> <p>When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.CI is set to ext-CLUSTERRAS_ERR0PFGCTL.CI.</p>	0b1
[7:6]	CE	<p>Corrected Error generation. Describes the types of Corrected Error that the fault generation feature of the node can generate.</p> <p><b>0b01</b></p> <p>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.</p>	0b01
[5]	DE	<p>Deferred Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b1</b></p> <p>The fault generation feature of the node allows generation of this type of error.</p>	0b1
[4]	UEO	<p>Latent or Restartable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b0</b></p> <p>The fault generation feature of the node cannot generate this type of error.</p> <p>This bit reads-as-zero.</p>	0b0
[3]	UER	<p>Signaled or Recoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b0</b></p> <p>The fault generation feature of the node cannot generate this type of error.</p> <p>This bit reads-as-zero.</p>	0b0
[2]	UEU	<p>Unrecoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b0</b></p> <p>The fault generation feature of the node cannot generate this type of error.</p> <p>This bit reads-as-zero.</p>	0b0
[1]	UC	<p>Uncontainable Error generation. Describes whether the fault generation feature of the node can generate this type of error.</p> <p><b>0b1</b></p> <p>The fault generation feature of the node allows generation of this type of error.</p>	0b1
[0]	OF	<p>Overflow flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.OF status flag.</p> <p><b>0b1</b></p> <p>When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.OF is set to ext-CLUSTERRAS_ERR0PFGCTL.OF.</p>	0b1

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x800	ERR0PFGF	None

This interface is accessible as follows:

RO

B.1.3.10 CLUSTERRAS\_ERR0PFGCTL, Pseudo-fault Generation Control Register

Enables controlled fault generation.

Configurations

External register CLUSTERRAS\_ERR0PFGCTL bits [63:0] are architecturally mapped to AArch64 System register [A.3.5 ERXPFGCTL\\_EL1, Selected Pseudo-fault Generation Control Register](#) on page 378 bits [63:0].

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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0

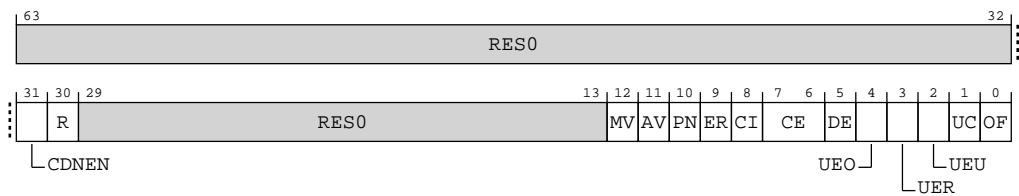


Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-38: ext\_clusterras\_err0pfgctl bit assignments



**Table B-86: CLUSTERRAS\_ERR0PFGCTL bit descriptions**

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31]	CDNEN	<p>Countdown Enable. Controls transfers from the value that is held in the ext-CLUSTERRAS_ERR0PFGCDN into the Error Generation Counter, and enables this counter.</p> <p><b>0b0</b></p> <p>The Error Generation Counter is disabled.</p> <p><b>0b1</b></p> <p>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.</p>	0b0
[30]	R	<p>Restart. Controls whether, on reaching zero, the Error Generation Counter restarts from the ext-CLUSTERRAS_ERR0PFGCDN value, or stops.</p> <p><b>0b0</b></p> <p>On reaching 0, the Error Generation Counter stops.</p> <p><b>0b1</b></p> <p>On reaching 0, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.</p>	x
[29:13]	RES0	Reserved	RES0
[12]	MV	<p>Miscellaneous syndrome. The value that is written to ext-CLUSTERRAS_ERROSTATUS.MV when an injected error is recorded.</p> <p><b>0b0</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.MV is set to 0 when an injected error is recorded.</p> <p><b>0b1</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.MV is set to 1 when an injected error is recorded.</p>	x
[11]	AV	<p>Address syndrome. The value that is written to ext-CLUSTERRAS_ERROSTATUS.AV when an injected error is recorded.</p> <p><b>0b0</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.AV is set to 0 when an injected error is recorded.</p> <p>This bit is <b>RES0</b>.</p> <p>Access to this field is: <b>RES0</b></p>	x
[10]	PN	<p>Poison flag. The value that is written to ext-CLUSTERRAS_ERROSTATUS.PN when an injected error is recorded.</p> <p><b>0b0</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.PN is set to 0 when an injected error is recorded.</p> <p><b>0b1</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.PN is set to 1 when an injected error is recorded.</p>	x
[9]	ER	<p>Error Reported flag. The value that is written to ext-CLUSTERRAS_ERROSTATUS.ER when an injected error is recorded.</p> <p><b>0b0</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.ER is set to 0 when an injected error is recorded.</p> <p><b>0b1</b></p> <p>ext-CLUSTERRAS_ERROSTATUS.ER is set to 1 when an injected error is recorded.</p> <p>This bit is <b>RES0</b>.</p> <p>Access to this field is: <b>RES0</b></p>	x

Bits	Name	Description	Reset
[8]	CI	<p>Critical Error flag. The value that is written to ext-CLUSTERRAS_ERRSTATUS.CI when an injected error is recorded.</p> <p><b>0b0</b> ext-CLUSTERRAS_ERRSTATUS.CI is set to 0 when an injected error is recorded.</p> <p><b>0b1</b> ext-CLUSTERRAS_ERRSTATUS.CI is set to 1 when an injected error is recorded.</p>	x
[7:6]	CE	<p>Corrected Error generation enable. Controls the type of Corrected Error condition that might be generated.</p> <p><b>0b00</b> No error of this type is generated.</p> <p><b>0b01</b> A non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERRSTATUS.CE == 0b10, might be generated when the Error Generation Counter decrements to zero.</p> <p>The set of permitted values for this field is defined by ext-CLUSTERRAS_ERROPFGF.CE.</p>	xx
[5]	DE	<p>Deferred Error generation enable. Controls whether this type of error condition might be generated.</p> <p><b>0b0</b> No error of this type is generated.</p> <p><b>0b1</b> An error of this type might be generated when the Error Generation Counter decrements to zero.</p>	x
[4]	UEO	<p>Latent or Restartable Error generation enable. Controls whether this type of error condition might be generated.</p> <p><b>0b0</b> No error of this type is generated.</p> <p>This bit is <b>RES0</b>.</p> <p>Access to this field is: <b>RES0</b></p>	x
[3]	UER	<p>Signaled or Recoverable Error generation enable. Controls whether this type of error condition might be generated.</p> <p><b>0b0</b> No error of this type is generated.</p> <p>This bit is <b>RES0</b>.</p> <p>Access to this field is: <b>RES0</b></p>	x
[2]	UEU	<p>Unrecoverable Error generation enable. Controls whether this type of error condition might be generated.</p> <p><b>0b0</b> No error of this type is generated.</p> <p>This bit is <b>RES0</b>.</p> <p>Access to this field is: <b>RES0</b></p>	x

Bits	Name	Description	Reset
[1]	UC	Uncontainable Error generation enable. Controls whether this type of error condition might be generated.  <b>0b0</b> No error of this type is generated.  <b>0b1</b> 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.  <b>0b0</b> ext-CLUSTERRAS_ERR0STATUS.OF is set to 0 when an injected error is recorded.  <b>0b1</b> ext-CLUSTERRAS_ERR0STATUS.OF is set to 1 when an injected error is recorded.	x

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x808	ERR0PFGCTL	None

This interface is accessible as follows:

RW

### 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

External register CLUSTERRAS\_ERR0PFGCDN bits [63:0] are architecturally mapped to AArch64 System register [A.3.6 ERXPFGCDN\\_EL1, Selected Pseudo-fault Generation Countdown Register](#) on page 383 bits [63:0].

## Attributes

### Width

64

### Component

CLUSTERRAS

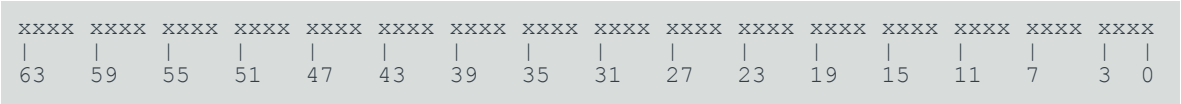
### Register offset

0x810

### Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-39: ext\_clusterras\_err0pfgcdn bit assignments

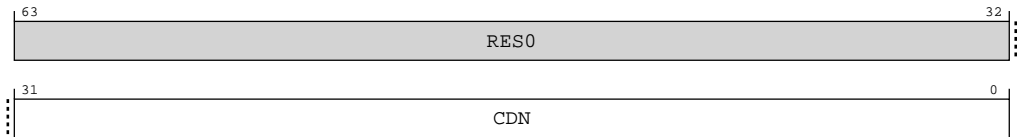


Table B-88: CLUSTERRAS\_ERR0PFGCDN bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31:0]	CDN	<p>Countdown value.</p> <p>This field is copied to Error Generation Counter when either:</p> <ul style="list-style-type: none"><li>Software writes ext-CLUSTERRAS_ERR0PFGCTL.CDNEN with 1.</li><li>The Error Generation Counter decrements to zero and ext-CLUSTERRAS_ERR0PFGCTL.R == 1.</li></ul> <p>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.</p> <p><b>Note:</b> The current Error Generation Counter value is not visible to software.</p>	32 {x}

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x810	ERR0PFGCDN	None

This interface is accessible as follows:

RW



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

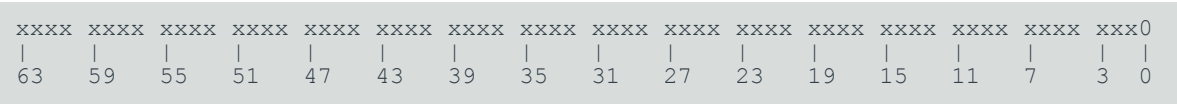
Register offset

0xE00

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-40: ext\_clusterras\_errgsr bit assignments

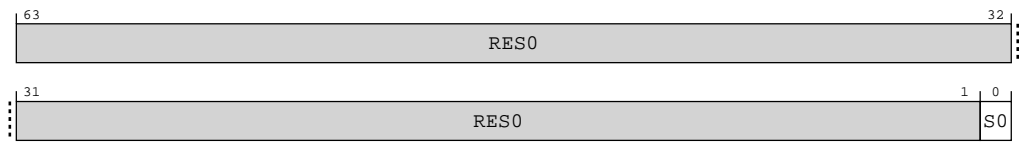


Table B-90: CLUSTERRAS\_ERRGSR bit descriptions

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.  <b>0b0</b> No error.  <b>0b1</b> One or more errors.	0b0

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xE00	ERRGSR	None

This interface is accessible as follows:

RO

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 1100 0000 0001 0100 0011 1011

Bit descriptions

Figure B-41: ext\_clusterras\_erriidr bit assignments

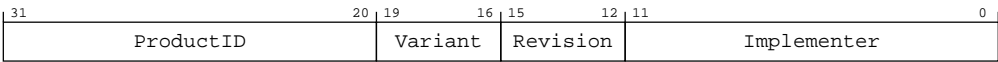


Table B-92: CLUSTERRAS\_ERRIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Part number, bits [11:0]. The part number is selected by the designer of the product.  <b>0b010011101100</b> DSU-120AE 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.	0x4EC

Bits	Name	Description	Reset
[19:16]	Variant	Product major revision.  This field distinguishes product variants or major revisions of the product.  <b>0b0000</b> Product variant 0.  ext-CLUSTERRAS_ERRPIDR2.REVISION matches CLUSTERRAS_ERRIIDR.Variant.	0b0000
[15:12]	Revision	Product minor revision.  This field distinguishes minor revisions of the product.  <b>0b0000</b> Product revision 0.  <b>0b0001</b> Product revision 1.  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.  <b>0b010000111011</b> 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 <b>RES0</b> .  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

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xE10	ERRIIDR	None

This interface is accessible as follows:

RO

### 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	0000	0000	
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-42: ext\_clusterras\_errdevaff bit assignments

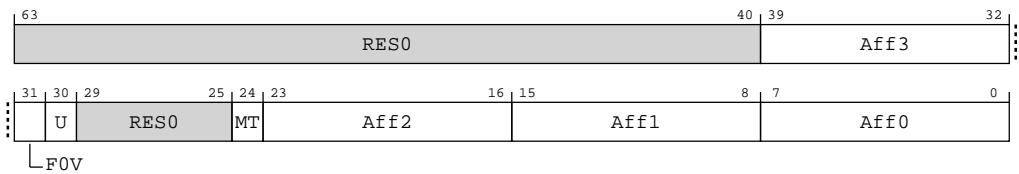


Table B-94: CLUSTERRAS\_ERRDEVAFF bit descriptions

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]	FOV	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
[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

Bits	Name	Description	Reset
[24]	MT	Multithreaded. The AArch64-MPIDR_EL1.MT bit viewed from the highest Exception level of the associated PE. <b>0b0</b> 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 <b>UNKNOWN</b> .	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. <b>0b10000000</b> ERRDEVAFF.Aff2 is valid, and the PE affinity level is 2.	0x80
[7:0]	Aff0	Affinity level 0. <b>0b00000000</b> The PE affinity is above level 1 or a subset of level 1. All other values are reserved.	0x00

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFA8	ERRDEVAFF	None

This interface is accessible as follows:

RO

### 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

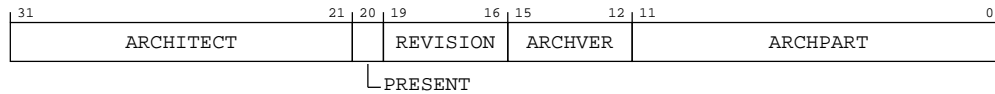
RO

## Reset value

0100 0111 0111 0001 0000 1010 0000 0000

## Bit descriptions

**Figure B-43: ext\_clusterras\_errdevarch bit assignments**



**Table B-96: CLUSTERRAS\_ERRDEVARCH bit descriptions**

Bits	Name	Description	Reset
[31:21]	ARCHITECT	<p>Architect.</p> <p>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.</p> <p><b>0b01000111011</b></p> <p>JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.</p>	0b01000111011
[20]	PRESENT	<p>DEVARCH Present.</p> <p>Defines that the DEVARCH register is present.</p> <p><b>0b1</b></p> <p>Device Architecture information present.</p> <p>This bit is <b>RAO</b>.</p>	0b1
[19:16]	REVISION	<p>Revision.</p> <p>Defines the architecture revision of the component. The defined values of this field are:</p> <p><b>0b0001</b></p> <p>RAS System Architecture v1.1</p> <p>All other values are reserved.</p>	0b0001
[15:12]	ARCHVER	<p>Architecture Version.</p> <p>Defines the architecture version of the component. The defined values of this field are:</p> <p><b>0b0000</b></p> <p>RAS System Architecture v1.</p> <p>This field reads as 0b0000.</p> <p>All other values are reserved.</p> <p>ARCHVER and ARCHPART are also defined as a single field, ARCHID, so that ARCHVER is ARCHID[15:12].</p>	0b0000

Bits	Name	Description	Reset
[11:0]	ARCHPART	Architecture Part.  Defines the architecture of the component.  <b>0b101000000000</b> 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

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFBC	ERRDEVARCH	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-44: ext\_clusterras\_errdevid bit assignments

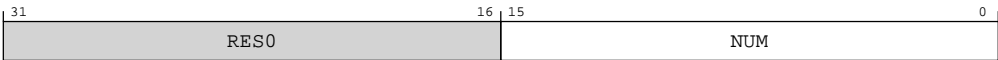


Table B-98: CLUSTERRAS\_ERRDEVID bit descriptions

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

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFC8	ERRDEVID	None

This interface is accessible as follows:

RO

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

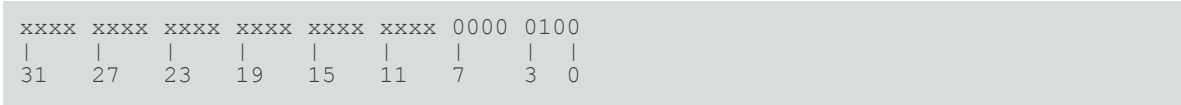
Register offset

0xFD0



Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-45: ext\_clusterras\_errpidr4 bit assignments



Table B-100: CLUSTERRAS\_ERRPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[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.  <b>0b0000</b> 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.  <b>0b0100</b> Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.  <b>Note:</b> For a component designed by Arm Limited, the JEP106 bank is 5, meaning this field has the value 0x4.	0b0100

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFD0	ERRPIDR4	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-46: ext\_clusterras\_errpidr5 bit assignments

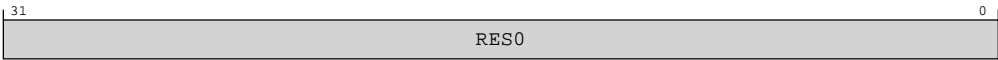


Table B-102: CLUSTERRAS\_ERRPIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFD4	ERRPIDR5	None

This interface is accessible as follows:

RO

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

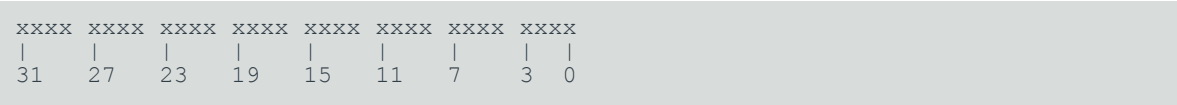
Register offset

0xFD8

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-47: ext\_clusterras\_errpidr6 bit assignments

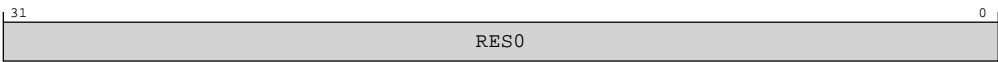


Table B-104: CLUSTERRAS\_ERRPIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFD8	ERRPIDR6	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-48: ext\_clusterras\_errpidr7 bit assignments

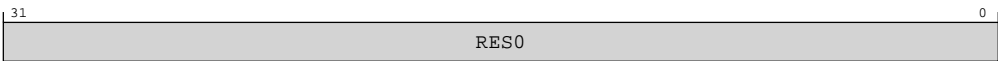


Table B-106: CLUSTERRAS\_ERRPIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFDC	ERRPIDR7	None

This interface is accessible as follows:

RO

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	1100
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-49: ext\_clusterras\_errpidr0 bit assignments

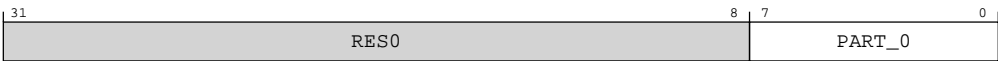


Table B-108: CLUSTERRAS\_ERRPIDR0 bit descriptions

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. <b>0b11101100</b> DSU-120AE Cluster RAS. Bits [7:0] of part number 0x4EC.	0xEC

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFE0	ERRPIDR0	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-50: ext\_clusterras\_errpidr1 bit assignments

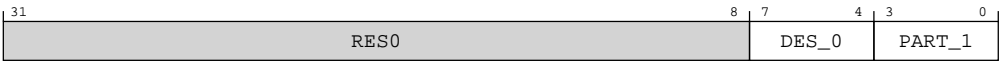


Table B-110: CLUSTERRAS\_ERRPIDR1 bit descriptions

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.  <b>0b1011</b> Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.  <b>Note:</b> For a component designed by Arm Limited, the JEP106 identification code is 0x3B.	0b1011
[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.  <b>0b0100</b> DSU-120AE Cluster RAS. Bits [11:8] of part number 0x4EC.	0b0100

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFE4	ERRPIDR1	None

This interface is accessible as follows:

RO

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	0001	1011
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-51: ext\_clusterras\_errpidr2 bit assignments

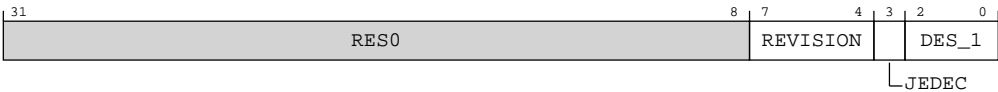


Table B-112: CLUSTERRAS\_ERRPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[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.  <b>0b0000</b> Component major revision 0.  <b>0b0001</b> Component major revision 1.  For DSU-120AE: <ul style="list-style-type: none"><li>Major revision 0 corresponds to r0p0.</li><li>Major revision 1 corresponds to r0p1.</li></ul>	0b0001
[3]	JEDEC	JEDEC-assigned JEP106 implementer code is used. This bit is <b>RAO</b> .  <b>0b1</b> JEDEC-assignee values is used.	0b1



Bits	Name	Description	Reset
[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.  <b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.  <b>Note:</b> For a component designed by Arm Limited, the JEP106 identification code is 0x3B.	0b011

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFE8	ERRPIDR2	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure B-52: ext\_clusterras\_errpidr3 bit assignments



Table B-114: CLUSTERRAS\_ERRPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVISION	Component minor revision. <b>0b0000</b> Component minor revision 0.	0b0000
[3:0]	CMOD	Customer Modified. <b>0b0000</b> The component is not modified from the original design.  For any two components with the same Unique Component Identifier: <ul style="list-style-type: none"><li>If the value of the CMOD fields of both components equals zero, the components are identical.</li><li>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.</li><li>If the CMOD fields of both components have the same non-zero value, it does not necessarily mean that they have the same modifications.</li></ul>	0b0000

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFEC	ERRPIDR3	None

This interface is accessible as follows:

RO

### 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

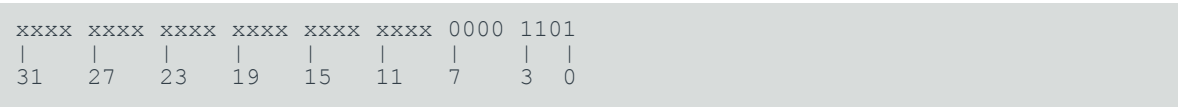
Register offset

0xFF0

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-53: ext\_clusterras\_errcidr0 bit assignments



Table B-116: 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

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFF0	ERRCIDR0	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-54: ext\_clusterras\_errcidr1 bit assignments



Table B-118: 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

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFF4	ERRCIDR1	None

This interface is accessible as follows:

RO

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

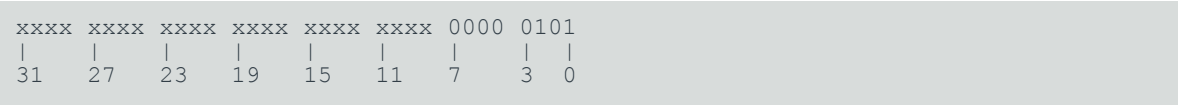
Register offset

0xFF8

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-55: ext\_clusterras\_errcidr2 bit assignments



Table B-120: 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

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFF8	ERRCIDR2	None

This interface is accessible as follows:

RO

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

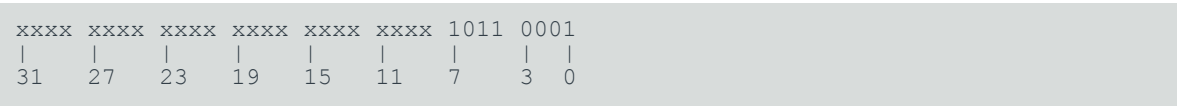
Register offset

0xFFC

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-56: ext\_clusterras\_errcidr3 bit assignments



Table B-122: 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

## Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0xFFC	ERRCIDR3	None

This interface is accessible as follows:

RO

### B.1.4 External cluster PPU registers summary

The *Power Policy Unit* (PPU) registers for the DSU-120AE DynamIQ™ cluster are only accessible from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the cluster PPU registers that are accessed externally (memory-mapped) from the utility bus of the DSU-120AE. For more information about a register, click on the register name in the table.



Note

- You must access the cluster system control registers from the Secure state.
- The cluster PPU registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- 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 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-124: Cluster PPU register summary**

Offset	Name	Reset	Width	Description
0x000	<a href="#">PPU_PWPR</a>	See individual bit resets.	32-bit	Power Policy Register
0x004	<a href="#">PPU_PMER</a>	See individual bit resets.	32-bit	Power Mode Emulation Enable Register
0x008	<a href="#">PPU_PWSR</a>	See individual bit resets.	32-bit	Power Status Register
0x010	<a href="#">PPU_DISR</a>	See individual bit resets.	32-bit	Device Interface Input Current Status Register
0x014	<a href="#">PPU_MISR</a>	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register
0x018	<a href="#">PPU_STSR</a>	See individual bit resets.	32-bit	Stored Status Register
0x01C	<a href="#">PPU_UNLK</a>	See individual bit resets.	32-bit	Unlock Register
0x020	<a href="#">PPU_PWCR</a>	See individual bit resets.	32-bit	Power Configuration Register
0x024	<a href="#">PPU_PTCR</a>	See individual bit resets.	32-bit	Power Mode Transition Register

Offset	Name	Reset	Width	Description
0x030	PPU_IMR	See individual bit resets.	32-bit	Interrupt Mask Register
0x034	PPU_AIMR	See individual bit resets.	32-bit	Additional Interrupt Mask Register
0x038	PPU_ISR	See individual bit resets.	32-bit	Interrupt Status Register
0x03C	PPU_AISR	See individual bit resets.	32-bit	Additional Interrupt Status Register
0x040	PPU_IESR	See individual bit resets.	32-bit	Input Edge Sensitivity Register
0x044	PPU_OPSR	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register
0x050	PPU_FUNRR	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register
0x054	PPU_FULRR	See individual bit resets.	32-bit	Full Retention RAM Configuration Register
0x058	PPU_MEMRR	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register
0x170	PPU_DCDR0	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0
0x174	PPU_DCDR1	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1
0xFB0	PPU_IDR0	See individual bit resets.	32-bit	PPU Identification Register 0
0xFB4	PPU_IDR1	See individual bit resets.	32-bit	PPU Identification Register 1
0xFC8	PPU_IIDR	See individual bit resets.	32-bit	Implementation Identification Register
0xFCC	PPU_AIDR	See individual bit resets.	32-bit	Architecture Identification Register
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1
0xFE8	PPU_PIDR2	See individual bit resets.	32-bit	PPU Peripheral Identification Register 2
0xFEC	PPU_PIDR3	See individual bit resets.	32-bit	PPU Peripheral Identification Register 3
0xFF0	PPU_CIDR0	See individual bit resets.	32-bit	PPU Component Identification Register 0
0xFF4	PPU_CIDR1	See individual bit resets.	32-bit	PPU Component Identification Register 1
0xFF8	PPU_CIDR2	See individual bit resets.	32-bit	PPU Component Identification Register 2
0xFFC	PPU_CIDR3	See individual bit resets.	32-bit	PPU Component Identification Register 3

#### 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



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-57: ext\_ppu\_pwpr bit assignments

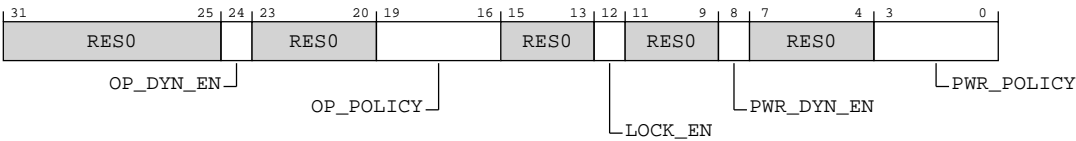


Table B-125: PPU\_PWPR bit descriptions

Bits	Name	Description	Reset
[31:25]	RES0	Reserved	RES0
[24]	OP_DYN_EN	Operating mode dynamic transition enable.  <b>0b0</b> Dynamic transitions disabled for operating modes.  <b>0b1</b> Dynamic transitions enabled for operating modes, allowing transitions to be initiated by changes on operating mode DEACTIVE inputs.	0b0
[23:20]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[19:16]	OP_POLICY	<p>Operating mode policy.</p> <p>When static operating mode transitions are enabled, OP_DYN_EN is set to 0b0, then this is the target operating mode for the PPU.</p> <p>When dynamic operating mode transitions are enabled, OP_DYN_EN is set to 0b1, then this is the minimum operating mode for the PPU.</p> <p>All other values are reserved.</p> <p><b>0b0000</b> OPMODE_00: ONE_SLICE_SF_ONLY_ON: One L3 Cache slice is operational, the Cache RAM is powered down.</p> <p><b>0b0001</b> OPMODE_01: ONE_SLICE_HALF_RAM_ON: One L3 Cache slice is operational, half of the Cache RAMs are powered on.</p> <p><b>0b0011</b> OPMODE_03: ONE_SLICE_FULL_RAM_ON: One L3 Cache slice is operational, all of the Cache RAMs are powered on.</p> <p><b>0b0100</b> OPMODE_04: ALL_SLICE_SF_ONLY_ON: All L3 Cache slices are operational, the Cache RAMs in each slice are powered down.</p> <p><b>0b0101</b> OPMODE_05: ALL_SLICE_HALF_RAM_ON: All L3 Cache slices are operational, half of the Cache RAMs are powered on.</p> <p><b>0b0111</b> OPMODE_07: ALL_SLICE_FULL_RAM_ON: All L3 Cache slices are operational, all of the Cache RAMs are powered on.</p> <p><b>0b1000</b> OPMODE_08: HALF_SLICE_SF_ONLY_ON: Half L3 Cache slices are operational, the Cache RAMs in each slice are powered down.</p> <p><b>0b1001</b> OPMODE_09: HALF_SLICE_HALF_RAM_ON: Half L3 Cache slices are operational, half of the Cache RAMs are powered on.</p> <p><b>0b1011</b> OPMODE_0B: HALF_SLICE_FULL_RAM_ON: Half L3 Cache slices are operational, all of the Cache RAMs are powered on.</p>	0b0000
[15:13]	RES0	Reserved	RES0
[12]	LOCK_EN	<p>Lock enable bit for OFF, OFF_EMU, MEM_RET and MEM_RET_EMU power modes.</p> <p><b>0b0</b> Lock feature disabled.</p> <p><b>0b1</b> Lock feature enabled.</p>	0b0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8]	PWR_DYN_EN	<p>Power mode dynamic transition enable.</p> <p><b>0b0</b> Dynamic transitions disabled for power modes.</p> <p><b>0b1</b> Dynamic transitions enabled for power modes, allowing transitions to be initiated by changes on power mode DEACTIVE inputs.</p>	0b0
[7:4]	RES0	Reserved	RES0
[3:0]	PWR_POLICY	<p>Power mode policy.</p> <p>When static power mode transitions are enabled, PWR_DYN_EN is set to 0b0, this is the target power mode for the PPU.</p> <p>When dynamic power mode transitions are enabled, PWR_DYN_EN is set to 0b1, this is the minimum power mode for the PPU.</p> <p>All other values are reserved.</p> <p><b>0b0000</b> OFF. Logic off and RAM off.</p> <p><b>0b0001</b> OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power.</p> <p><b>0b0010</b> MEM_RET. Memory Retention. Logic off with RAM retained.</p> <p><b>0b0011</b> 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.</p> <p><b>0b0101</b> FULL_RET. Full Retention. Slice logic off with RAM contents retained.</p> <p><b>0b0111</b> FUNC_RET. Functional Retention. Logic on with L3 Cache and Snoop Filter retained.</p> <p><b>0b1000</b> ON. Logic on with RAM on, cluster is functional.</p> <p><b>0b1001</b> WARM_RST. Warm Reset. Warm reset application with logic and RAM on.</p> <p><b>0b1010</b> DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.</p>	0b0000

## Accessibility

This interface is accessible as follows:

RW

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

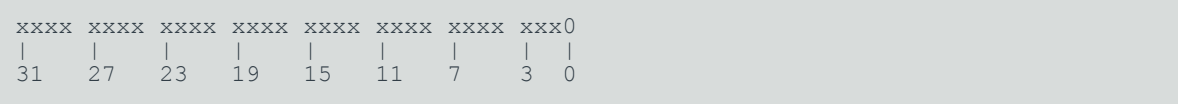
Register offset

0x004

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-58: ext\_ppu\_pmer bit assignments

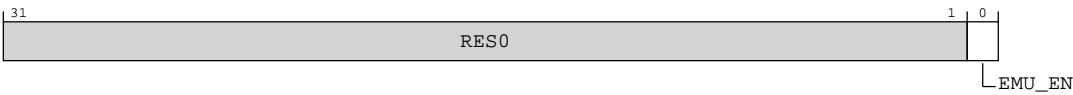


Table B-126: PPU\_PMER bit descriptions

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

Accessibility

This interface is accessible as follows:

RW

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

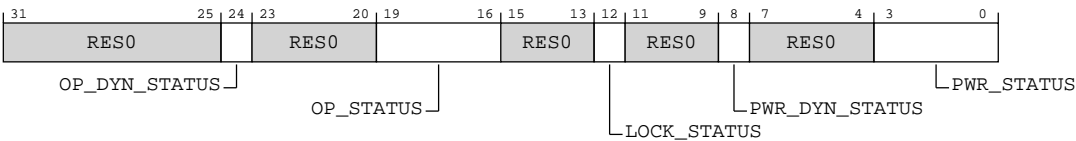
xxxx	xxx0	xxxx	0000	xxx0	xxx0	xxxx	0000
31	27	23	19	15	11	7	3
							0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-59: ext\_ppu\_pwsr bit assignments



**Table B-127: PPU\_PWSR bit descriptions**

Bits	Name	Description	Reset
[31:25]	RES0	Reserved	RES0
[24]	OP_DYN_STATUS	<p>Operating mode dynamic transition status.</p> <p>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.</p> <p><b>0b0</b></p> <p>Dynamic transitions disabled for operating modes.</p> <p><b>0b1</b></p> <p>Dynamic transitions enabled for operating modes.</p>	0b0
[23:20]	RES0	Reserved	RES0
[19:16]	OP_STATUS	<p>Operating mode status.</p> <p>These bits reflect the current operating mode of the PPU.</p> <p>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 DEVSTATE output bits are set to zero.</p> <p>All other values are reserved.</p> <p><b>0b0000</b></p> <p>OPMODE_00: ONE_SLICE_SF_ONLY_ON: One L3 Cache slice is operational, only the snoop filter RAM instances are active in the slice</p> <p><b>0b0001</b></p> <p>OPMODE_01: ONE_SLICE_HALF_RAM_ON: One L3 Cache slice is operational, half of the Cache RAMs are powered on.</p> <p><b>0b0011</b></p> <p>OPMODE_03: ONE_SLICE_FULL_RAM_ON: One L3 Cache slice is operational, all of the Cache RAMs are powered on.</p> <p><b>0b0100</b></p> <p>OPMODE_04: ALL_SLICE_SF_ONLY_ON: All L3 Cache slices are operational, only the snoop filter RAM instances are active in each slice.</p> <p><b>0b0101</b></p> <p>OPMODE_05: ALL_SLICE_HALF_RAM_ON: All L3 Cache slices are operational, half of the Cache RAMs are powered on.</p> <p><b>0b0111</b></p> <p>OPMODE_07: ALL_SLICE_FULL_RAM_ON: All L3 Cache slices are operational, all of the Cache RAMs are powered on.</p> <p><b>0b1000</b></p> <p>OPMODE_08: HALF_SLICE_SF_ONLY_ON: Half L3 Cache slices are operational, the Cache RAMs in each slice are powered down.</p> <p><b>0b1001</b></p> <p>OPMODE_09: HALF_SLICE_HALF_RAM_ON: Half L3 Cache slices are operational, half of the Cache RAMs are powered on.</p> <p><b>0b1011</b></p> <p>OPMODE_0B: HALF_SLICE_FULL_RAM_ON: Half L3 Cache slices are operational, all of the Cache RAMs are powered on.</p>	0b0000

Bits	Name	Description	Reset
[15:13]	RES0	Reserved	RES0
[12]	LOCK_STATUS	<p>Lock status.</p> <p><b>0b0</b> The PPU is not locked in the current mode.</p> <p><b>0b1</b> The PPU is locked in the current mode.</p>	0b0
[11:9]	RES0	Reserved	RES0
[8]	PWR_DYN_STATUS	<p>Power mode dynamic transition status.</p> <p>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.</p> <p><b>0b0</b> Dynamic transitions disabled for power modes.</p> <p><b>0b1</b> Dynamic transitions enabled for power modes.</p>	0b0
[7:4]	RES0	Reserved	RES0
[3:0]	PWR_STATUS	<p>Power mode status.</p> <p>These bits reflect the current power mode of the PPU.</p> <p>All other values are reserved.</p> <p><b>0b0000</b> OFF. Logic off and RAM off.</p> <p><b>0b0001</b> OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power.</p> <p><b>0b0010</b> MEM_RET. Memory Retention. Logic off with RAM retained.</p> <p><b>0b0011</b> 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.</p> <p><b>0b0101</b> FULL_RET. Full Retention. Slice logic off with RAM contents retained.</p> <p><b>0b0111</b> FUNC_RET. Functional Retention. Logic on with L3 Cache and Snoop Filter retained.</p> <p><b>0b1000</b> ON. Logic on with RAM on, cluster is functional.</p> <p><b>0b1001</b> WARM_RST. Warm Reset. Warm reset application with logic and RAM on.</p> <p><b>0b1010</b> DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.</p>	0b0000

## Accessibility

This interface is accessible as follows:

RO

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

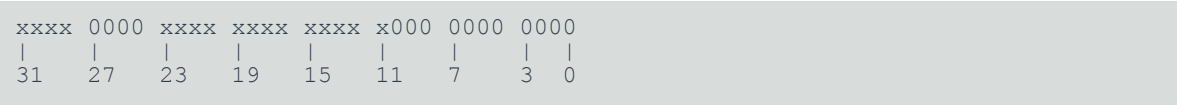
Register offset

0x010

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-60: ext\_ppu\_disr bit assignments

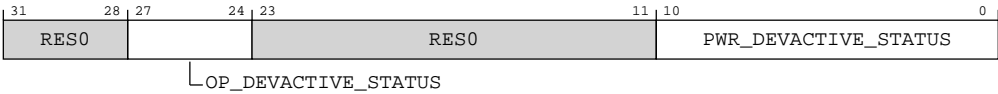


Table B-128: PPU\_DISR bit descriptions

Bits	Name	Description	Reset
[31:28]	RES0	Reserved	RES0



Bits	Name	Description	Reset
[27:24]	OP_DEVACTIVE_STATUS	<p>Status of the operating mode DEVPACTIVE inputs.</p> <p>All other values are reserved.</p> <p><b>0b0000</b> Request for OPMODE_00, ONE_SLICE_SF_ONLY_ON.</p> <p><b>0b0001</b> Request for OPMODE_01, ONE_SLICE_HALF_RAM_ON.</p> <p><b>0b0011</b> Request for OPMODE_03, ONE_SLICE_FULL_RAM_ON.</p> <p><b>0b0100</b> Request for OPMODE_04, ALL_SLICE_SF_ONLY_ON.</p> <p><b>0b0101</b> Request for OPMODE_05, ALL_SLICE_HALF_RAM_ON.</p> <p><b>0b0111</b> Request for OPMODE_07, ALL_SLICE_FULL_RAM_ON.</p> <p><b>0b1000</b> Request for OPMODE_08, HALF_SLICE_SF_ONLY_ON.</p> <p><b>0b1001</b> Request for OPMODE_09, HALF_SLICE_HALF_RAM_ON.</p> <p><b>0b1011</b> Request for OPMODE_0B, HALF_SLICE_FULL_RAM_ON.</p>	0b0000
[23:11]	RES0	Reserved	RES0
[10:0]	PWR_DEVACTIVE_STATUS	<p>Status of the power mode DEVPACTIVE inputs.</p> <p><b>0b000000000000</b> Request for OFF.</p> <p><b>0000000001x</b> Request for OFF_EMU.</p> <p><b>000000001xx</b> Request for MEM_RET.</p> <p><b>00000001xxx</b> Request for MEM_RET_EMU.</p> <p><b>000001xxxxx</b> Request for FULL_RET.</p> <p><b>0001xxxxxxx</b> Request for FUNC_RET.</p> <p><b>001xxxxxxx</b> Request for ON.</p> <p><b>01xxxxxxx</b> Request for WARM_RST.</p> <p><b>1xxxxxxx</b> Request for DBG_RECOV.</p>	0b000000000000

Accessibility

This interface is accessible as follows:

RO

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

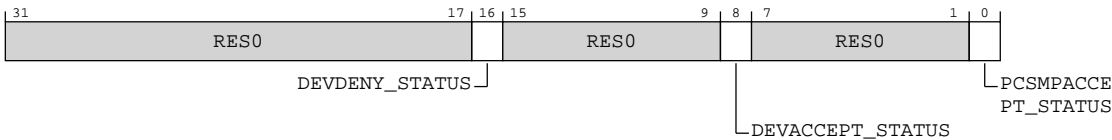
xxxx	xxxx	xxxx	xxx0	xxxx	xxx0	xxxx	xxx0
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-61: ext\_ppu\_misr bit assignments



**Table B-129: PPU\_MISR bit descriptions**

Bits	Name	Description	Reset
[31:17]	RES0	Reserved	RES0
[16]	DEVDPENY_STATUS	Status of the device interface DEVDPENY inputs.  <b>0b0</b> DEVDPENY deasserted.  <b>0b1</b> DEVDPENY asserted.	0b0
[15:9]	RES0	Reserved	RES0
[8]	DEVACCEPT_STATUS	Status of the device interface DEVACCEPT inputs.  <b>0b0</b> DEVACCEPT deasserted.  <b>0b1</b> DEVACCEPT asserted.	0b0
[7:1]	RES0	Reserved	RES0
[0]	PCSMPACCEPT_STATUS	Status of the PCSMPACCEPT inputs.  <b>0b0</b> PCSMPACCEPT deasserted.  <b>0b1</b> PCSMPACCEPT asserted.	0b0

## Accessibility

This interface is accessible as follows:

RO

## 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

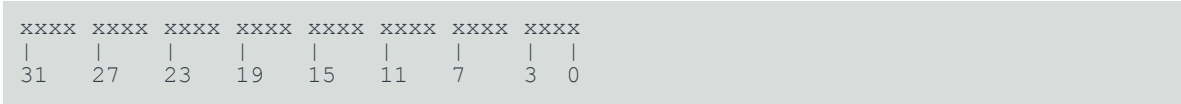
### Register offset

0x018

### Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-62: ext\_ppu\_stsr bit assignments

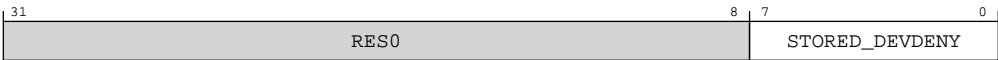


Table B-130: PPU\_STSR bit descriptions

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 }

Accessibility

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-63: ext\_ppu\_unlk bit assignments

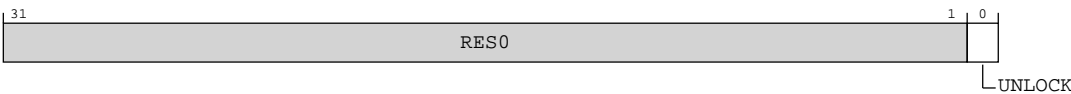


Table B-131: PPU\_UNLK bit descriptions

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

Accessibility

This interface is accessible as follows:

RW

B.1.4.8 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	1111	xxxx	x111	1111	111x	xxxx	xxx1
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure B-64: ext\_ppu\_pwcr bit assignments

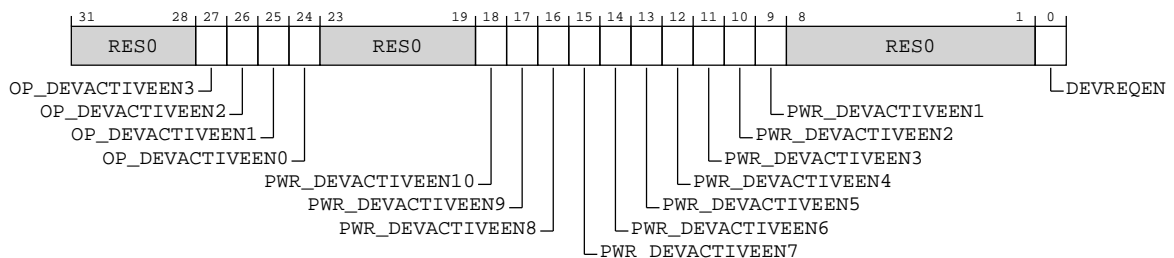


Table B-132: PPU\_PWCR bit descriptions

Bits	Name	Description	Reset
[31:28]	RES0	Reserved	RES0
[27]	OP_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[19] input.  <b>0b0</b> DEVPACTIVE[19] input (Half L3 Cache Slices active) disabled.  <b>0b1</b> DEVPACTIVE[19] input (Half L3 Cache Slices active) enabled.	0b1

Bits	Name	Description	Reset
[26]	OP_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[18] input.  <b>0b0</b> DEVPACTIVE[18] input (All L3 Cache Slices active) disabled.  <b>0b1</b> DEVPACTIVE[18] input (All L3 Cache Slices active) enabled.	0b1
[25]	OP_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[17] input.  <b>0b0</b> DEVPACTIVE[17] input (Upper L3 Cache RAMs active) disabled.  <b>0b1</b> DEVPACTIVE[17] input (Upper L3 Cache RAMs active) enabled.	0b1
[24]	OP_DEVACTIVEEN0	Enables the operating mode DEVPACTIVE[16] input.  <b>0b0</b> DEVPACTIVE[16] input (Lower L3 Cache RAMs active) disabled.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[10] input (DBG_RECOV) disabled.  <b>0b1</b> DEVPACTIVE[10] input (DBG_RECOV) enabled.	0b1
[17]	PWR_DEVACTIVEEN9	Enables the operating mode DEVPACTIVE[9] input.  <b>0b0</b> DEVPACTIVE[9] input (WARM_RST) disabled.  <b>0b1</b> DEVPACTIVE[9] input (WARM_RST) enabled.	0b1
[16]	PWR_DEVACTIVEEN8	Enables the operating mode DEVPACTIVE[8] input.  <b>0b0</b> DEVPACTIVE[8] input (ON) disabled.  <b>0b1</b> DEVPACTIVE[8] input (ON) enabled.	0b1
[15]	PWR_DEVACTIVEEN7	Enables the operating mode DEVPACTIVE[7] input.  <b>0b0</b> DEVPACTIVE[7] input (FUNC_RET) disabled.  <b>0b1</b> DEVPACTIVE[7] input (FUNC_RET) enabled.	0b1
[14]	PWR_DEVACTIVEEN6	Enables the operating mode DEVPACTIVE[6] input.  <b>0b1</b> DEVPACTIVE[6] input (MEM_OFF) enabled.	0b1

Bits	Name	Description	Reset
[13]	PWR_DEVACTIVEEN5	Enables the operating mode DEVPACTIVE[5] input. <b>0b0</b> DEVPACTIVE[5] input (FULL_RET) disabled. <b>0b1</b> DEVPACTIVE[5] input (FULL_RET) enabled.	0b1
[12]	PWR_DEVACTIVEEN4	Enables the operating mode DEVPACTIVE[4] input. <b>0b1</b> DEVPACTIVE[4] input (LOGIC_RET) enabled.	0b1
[11]	PWR_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[3] input. <b>0b0</b> DEVPACTIVE[3] input (MEM_RET_EMU) disabled. <b>0b1</b> DEVPACTIVE[3] input (MEM_RET_EMU) enabled.	0b1
[10]	PWR_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[2] input. <b>0b0</b> DEVPACTIVE[2] input (MEM_RET) disabled. <b>0b1</b> DEVPACTIVE[2] input (MEM_RET) enabled.	0b1
[9]	PWR_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[1] input. <b>0b0</b> DEVPACTIVE[1] input (OFF_EMU) disabled. <b>0b1</b> DEVPACTIVE[1] input (OFF_EMU) enabled.	0b1
[8:1]	RES0	Reserved	RES0
[0]	DEVREQEN	Device interface handshake enable. <b>0b0</b> Device interface handshake disabled for transitions. <b>0b1</b> Device interface handshake enabled for transitions.	0b1

## Accessibility

This interface is accessible as follows:

RW

### 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

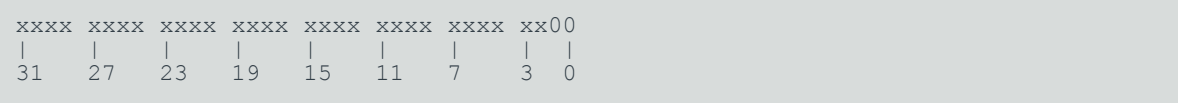
Register offset

0x024

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-65: ext\_ppu\_ptcr bit assignments

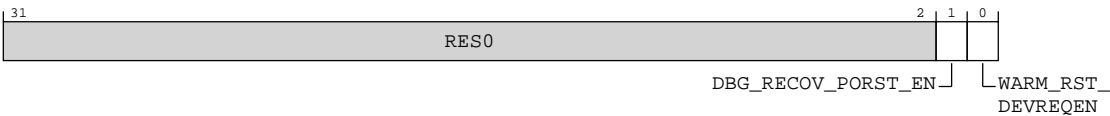


Table B-133: PPU\_PTCR bit descriptions

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 <b>UNPREDICTABLE</b> .  <b>0b0</b> DEVPORESETn is not asserted when in DBG_RECOV.  <b>0b1</b> DEVPORESETn is asserted when in DBG_RECOV.	0b0

Bits	Name	Description	Reset
[0]	WARM_RST_DEVREQEN	<div>Device interface handshake behavior.</div> <div>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 <b>UNPREDICTABLE</b>.</div> <div><b>0b0</b><div>The PPU does not perform a device interface handshake when transitioning between ON and WARM_RST.</div></div> <div><b>0b1</b><div>The PPU performs a device interface handshake when transitioning between ON and WARM_RST.</div></div>	0b0

Accessibility

This interface is accessible as follows:

RW

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure B-66: ext\_ppu\_imr bit assignments

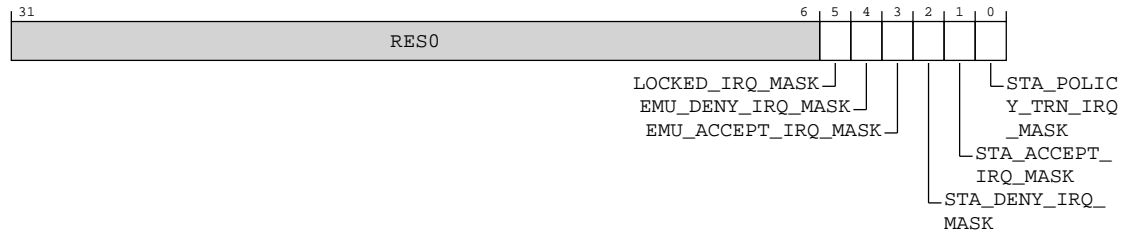


Table B-134: PPU\_IMR bit descriptions

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5]	LOCKED_IRQ_MASK	Locked event mask  <b>0b0</b> Locked event enabled.  <b>0b1</b> Locked event masked.	0b1
[4]	EMU_DENY_IRQ_MASK	Emulation transition denial event mask  <b>0b0</b> Emulation transition denial event enabled.  <b>0b1</b> Emulation transition denial event masked.	0b1
[3]	EMU_ACCEPT_IRQ_MASK	Emulation transition acceptance event mask  <b>0b0</b> Emulation transition acceptance event enabled.  <b>0b1</b> Emulation transition acceptance event masked.	0b1
[2]	STA_DENY_IRQ_MASK	Static transition denial event mask  <b>0b0</b> Static transition denial event enabled.  <b>0b1</b> Static transition denial event masked.	0b0

Bits	Name	Description	Reset
[1]	STA_ACCEPT_IRQ_MASK	Static transition acceptance event mask  <b>0b0</b> Static transition acceptance event enabled.  <b>0b1</b> Static transition acceptance event masked.	0b1
[0]	STA_POLICY_TRN_IRQ_MASK	Static full policy transition completion event mask  <b>0b0</b> Static full policy transition completion event enabled.  <b>0b1</b> Static full policy transition completion event masked.	0b0

Accessibility

This interface is accessible as follows:

RW

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

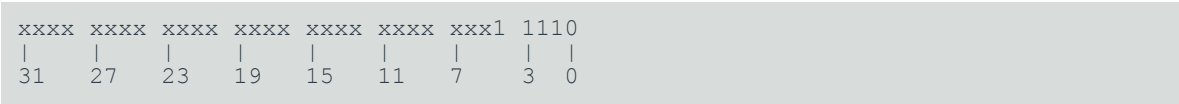
Register offset

0x034

Access type

RW

Reset value





Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure B-67: ext\_ppu\_aimr bit assignments

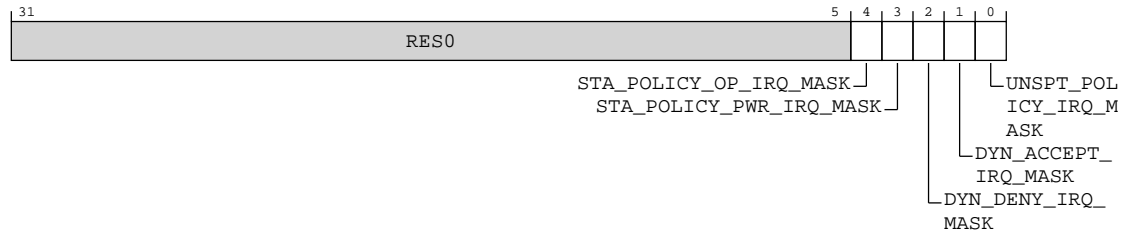


Table B-135: PPU\_AIMR bit descriptions

Bits	Name	Description	Reset
[31:5]	RES0	Reserved	RES0
[4]	STA_POLICY_OP_IRQ_MASK	Static operating policy transition completion event mask <b>0b0</b> Static operating policy transition completion event enabled. <b>0b1</b> Static operating policy transition completion event masked.	0b1
[3]	STA_POLICY_PWR_IRQ_MASK	Static power policy transition completion event mask <b>0b0</b> Static power policy transition completion event enabled. <b>0b1</b> Static power policy transition completion event masked.	0b1
[2]	DYN_DENY_IRQ_MASK	Dynamic transition denial event mask <b>0b0</b> Dynamic transition denial event enabled. <b>0b1</b> Dynamic transition denial event masked.	0b1
[1]	DYN_ACCEPT_IRQ_MASK	Dynamic transition acceptance event mask <b>0b0</b> Dynamic transition acceptance event enabled. <b>0b1</b> Dynamic transition acceptance event masked.	0b1

Bits	Name	Description	Reset
[0]	UNSPT_POLICY_IRQ_MASK	<p>Unsupported policy event mask</p> <p><b>0b0</b> Unsupported policy event enabled.</p> <p><b>0b1</b> Unsupported policy event masked.</p>	0b0

## Accessibility

This interface is accessible as follows:

RW

### 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	0000	xxxx	x000	0x0x	000x	0x00	0000
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

Figure B-68: ext\_ppu\_isr bit assignments

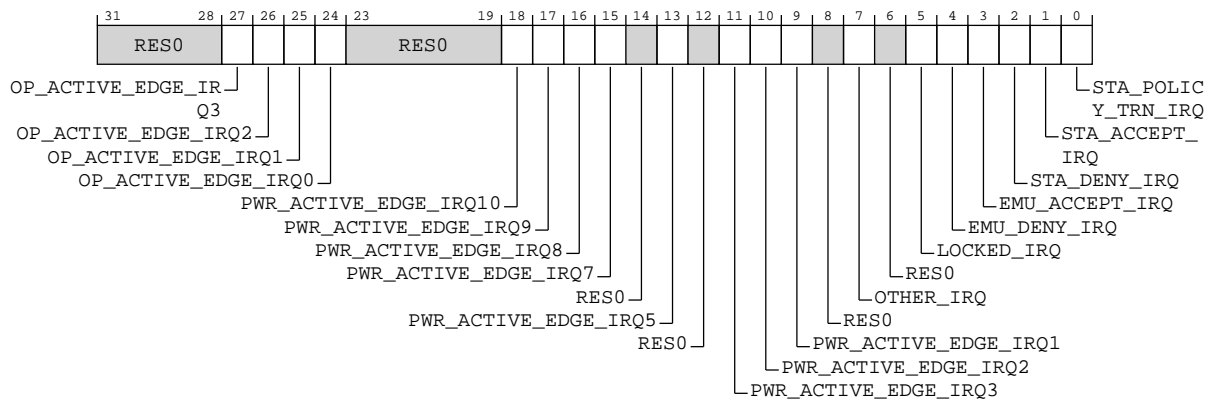


Table B-136: PPU\_ISR bit descriptions

Bits	Name	Description	Reset
[31:28]	RES0	Reserved	RES0
[27]	OP_ACTIVE_EDGE_IRQ3	Indicates if operating mode DEVPACTIVE[19] input caused the input edge event.  <b>0b0</b> DEVPACTIVE[19] input (Half L3 Cache Slices active) did not assert the interrupt output.  <b>0b1</b> DEVPACTIVE[19] input (Half L3 Cache Slices active) asserted the interrupt output.	0b0
[26]	OP_ACTIVE_EDGE_IRQ2	Indicates if operating mode DEVPACTIVE[18] input caused the input edge event.  <b>0b0</b> DEVPACTIVE[18] input (All L3 Cache Slices active) did not assert the interrupt output.  <b>0b1</b> DEVPACTIVE[18] input (All L3 Cache Slices active) asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[25]	OP_ACTIVE_EDGE_IRQ1	Indicates if operating mode DEVPACTIVE[17] input caused the input edge event.  <b>0b0</b> DEVPACTIVE[17] input (Upper L3 Cache RAMs active) did not assert the interrupt output.  <b>0b1</b> DEVPACTIVE[17] input (Upper L3 Cache RAMs active) asserted the interrupt output.	0b0
[24]	OP_ACTIVE_EDGE_IRQ0	Indicates if operating mode DEVPACTIVE[16] input caused the input edge event.  <b>0b0</b> DEVPACTIVE[16] input (Lower L3 Cache RAMs active) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[10] input (DBG_RECOV) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[9] input (WARM_RST) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[8] input (ON) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[7] input (FUNC_RET) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[5] input (FULL_RET) did not assert the interrupt output.  <b>0b1</b> DEVPACTIVE[5] input (FULL_RET) asserted the interrupt output.	0b0
[12]	RES0	Reserved	RES0



Bits	Name	Description	Reset
[11]	PWR_ACTIVE_EDGE_IRQ3	Indicates if power mode DEVPACTIVE[3] input caused the input edge event.  <b>0b0</b> DEVPACTIVE[3] input (MEM_RET_EMU) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[2] input (MEM_RET) did not assert the interrupt output.  <b>0b1</b> DEVPACTIVE[2] input (MEM_RET) asserted the interrupt output.	0b0
[9]	PWR_ACTIVE_EDGE_IRQ1	Indicates if power mode DEVPACTIVE[1] input caused the input edge event.  <b>0b0</b> DEVPACTIVE[1] input (OFF_EMU) did not assert the interrupt output.  <b>0b1</b> 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).  <b>0b0</b> No interrupt pending in ext-PPU_AISR.  <b>0b1</b> Interrupt pending in ext-PPU_AISR.	0b0
[6]	RES0	Reserved	RES0
[5]	LOCKED_IRQ	Locked event status.  <b>0b0</b> No locked event.  <b>0b1</b> A locked event asserted the interrupt output.	0b0
[4]	EMU_DENY_IRQ	Emulated transition denial event status.  <b>0b0</b> No emulated transition denial event.  <b>0b1</b> An emulated transition denial event asserted the interrupt output.	0b0
[3]	EMU_ACCEPT_IRQ	Emulated transition acceptance event status.  <b>0b0</b> No emulated transition acceptance event.  <b>0b1</b> An emulated transition acceptance event asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[2]	STA_DENY_IRQ	Static transition denial event status.  <b>0b0</b> No static transition denial event.  <b>0b1</b> An static transition denial event asserted the interrupt output.	0b0
[1]	STA_ACCEPT_IRQ	Static transition acceptance event status.  <b>0b0</b> No static transition acceptance event.  <b>0b1</b> An static transition acceptance event asserted the interrupt output.	0b0
[0]	STA_POLICY_TRN_IRQ	Static full policy transition completion event status.  <b>0b0</b> No static full policy transition completion event.  <b>0b1</b> An static full policy transition completion event asserted the interrupt output.	0b0

## Accessibility

This interface is accessible as follows:

RW

### 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

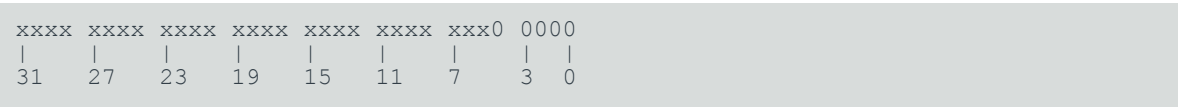
Register offset

0x03C

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-69: ext\_ppu\_aisr bit assignments

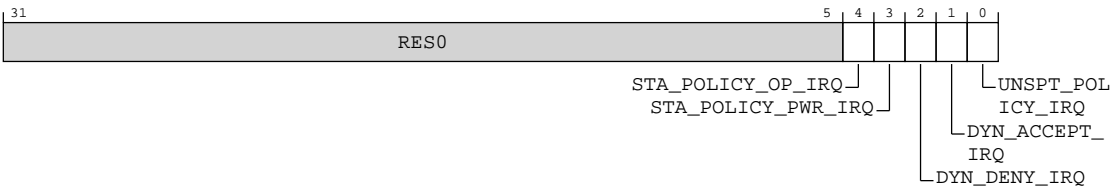


Table B-137: PPU\_AISR bit descriptions

Bits	Name	Description	Reset
[31:5]	RES0	Reserved	RES0
[4]	STA_POLICY_OP_IRQ	Static operating policy transition completion event status  <b>0b0</b> No static operating policy transition completion event.  <b>0b1</b> A static operating policy transition completion event asserted the interrupt output.	0b0
[3]	STA_POLICY_PWR_IRQ	Static power policy transition completion event status  <b>0b0</b> No static power policy transition completion event.  <b>0b1</b> A static power policy transition completion event asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[2]	DYN_DENY_IRQ	Dynamic transition denial event status  <b>0b0</b> No dynamic transition denial event.  <b>0b1</b> A dynamic transition denial event asserted the interrupt output.	0b0
[1]	DYN_ACCEPT_IRQ	Dynamic transition acceptance event status  <b>0b0</b> No dynamic transition acceptance event.  <b>0b1</b> A dynamic transition acceptance event asserted the interrupt output.	0b0
[0]	UNSPT_POLICY_IRQ	Unsupported policy event status  <b>0b0</b> No unsupported policy event.  <b>0b1</b> An unsupported policy event asserted the interrupt output.	0b0

## Accessibility

This interface is accessible as follows:

RW

### 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

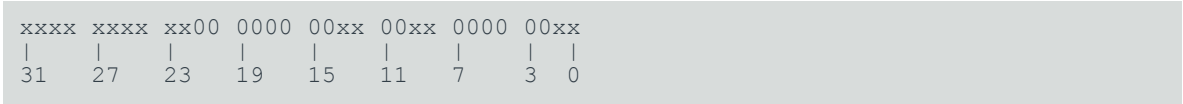
### Register offset

0x040

### Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-70: ext\_ppu\_iesr bit assignments

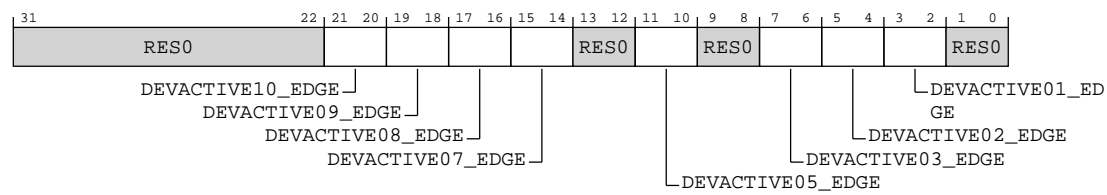


Table B-138: PPU\_IESR bit descriptions

Bits	Name	Description	Reset
[31:22]	RES0	Reserved	RES0
[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

Bits	Name	Description	Reset
[17:16]	DEVACTIVE08_EDGE	<p>Configures the transitions on the DEVPACTIVE[8] input (ON) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[15:14]	DEVACTIVE07_EDGE	<p>Configures the transitions on the DEVPACTIVE[7] input (FUNC_RET) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[13:12]	RES0	Reserved	RES0
[11:10]	DEVACTIVE05_EDGE	<p>Configures the transitions on the DEVPACTIVE[5] input (FULL_RET) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[9:8]	RES0	Reserved	RES0
[7:6]	DEVACTIVE03_EDGE	<p>Configures the transitions on the DEVPACTIVE[3] input (MEM_RET_EMU) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00

Bits	Name	Description	Reset
[5:4]	DEVACTIVE02_EDGE	<p>Configures the transitions on the DEVPACTIVE[2] input (MEM_RET) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[3:2]	DEVACTIVE01_EDGE	<p>Configures the transitions on the DEVPACTIVE[1] input (OFF_EMU) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[1:0]	RES0	Reserved	RES0

## Accessibility

This interface is accessible as follows:

RW

### 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	0000	0000
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-71: ext\_ppu\_opsr bit assignments

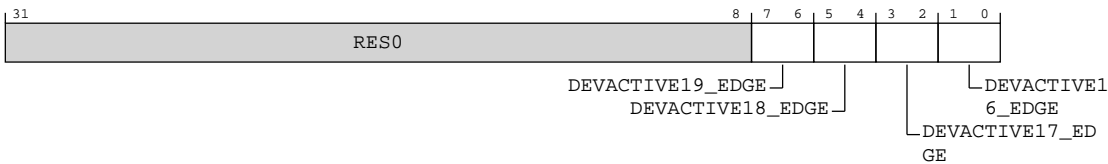


Table B-139: PPU\_OPSR bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:6]	DEVACTIVE19_EDGE	Configures the transitions on the DEVPACTIVE[19] input (Half L3 Cache Slices active) that generate an Input Edge interrupt event.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> Both edges of event generate an interrupt.	0b00



Bits	Name	Description	Reset
[5:4]	DEVACTIVE18_EDGE	<p>Configures the transitions on the DEVPACTIVE[18] input (All L3 Cache Slices active) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[3:2]	DEVACTIVE17_EDGE	<p>Configures the transitions on the DEVPACTIVE[17] input (Upper L3 Cache RAMs active) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00
[1:0]	DEVACTIVE16_EDGE	<p>Configures the transitions on the DEVPACTIVE[16] input (Lower L3 Cache RAMs active) that generate an Input Edge interrupt event.</p> <p><b>0b00</b> Event masked.</p> <p><b>0b01</b> Rising edge of event generates an interrupt.</p> <p><b>0b10</b> Falling edge of event generates an interrupt.</p> <p><b>0b11</b> Both edges of event generate an interrupt.</p>	0b00

## Accessibility

This interface is accessible as follows:

RW

## 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-72: ext\_ppu\_funrr bit assignments

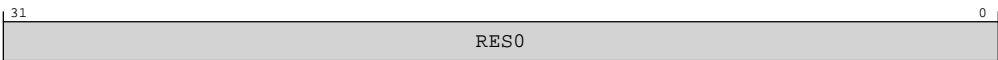


Table B-140: PPU\_FUNRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-73: ext\_ppu\_fulrr bit assignments

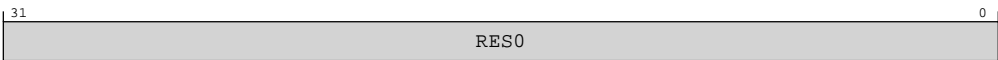


Table B-141: PPU\_FULRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

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

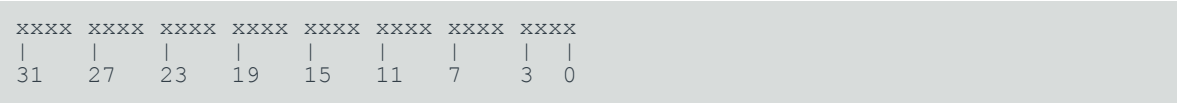
Register offset

0x058

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-74: ext\_ppu\_memrr bit assignments

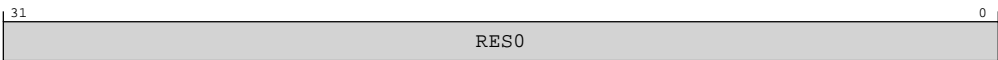


Table B-142: PPU\_MEMRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-75: ext\_ppu\_dcdr0 bit assignments

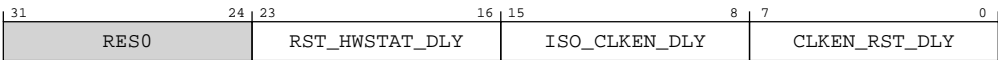


Table B-143: PPU\_DCDR0 bit descriptions

Bits	Name	Description	Reset
[31:24]	RES0	Reserved	RES0
[23:16]	RST_HWSTAT_DLY	Delay in PPUCLK clock cycles from reset de-assertion to HWSTAT update.Delay calculated as RST_HWSTAT_DLY + 1. Valid values for the field are in the range 0-255.	0x00
[15:8]	ISO_CLKEN_DLY	Delay in PPUCLK clock cycles from isolation enable de-assertion to clock enable assertion.Delay calculated as ISO_CLKEN_DLY + 1. Valid values for the field are in the range 0-255.	0x00
[7:0]	CLKEN_RST_DLY	Delay in PPUCLK clock cycles from clock enable assertion to reset de-assertion.Delay calculated as CLKEN_RST_DLY + 1. Valid values for the field are in the range 0-255.	0x00

Accessibility

This interface is accessible as follows:

RW

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-76: ext\_ppu\_dcdr1 bit assignments



Table B-144: PPU\_DCDR1 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:8]	CLKEN_ISO_DLY	Delay in PPUCLK clock cycles from clock enable de-assertion to isolation enable assertion.Delay calculated as CLKEN_ISO_DLY + 1. Valid values for the field are in the range 0-255.	0x00
[7:0]	ISO_RST_DLY	Delay in PPUCLK clock cycles from isolation enable assertion to reset assertion.Delay calculated as ISO_RST_DLY + 1. Valid values for the field are in the range 0-255.	0x00

Accessibility

This interface is accessible as follows:

RW

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

x111	1100	1111	x111	1100	1111	1111	0000
31	27	23	19	15	11	7	3 0

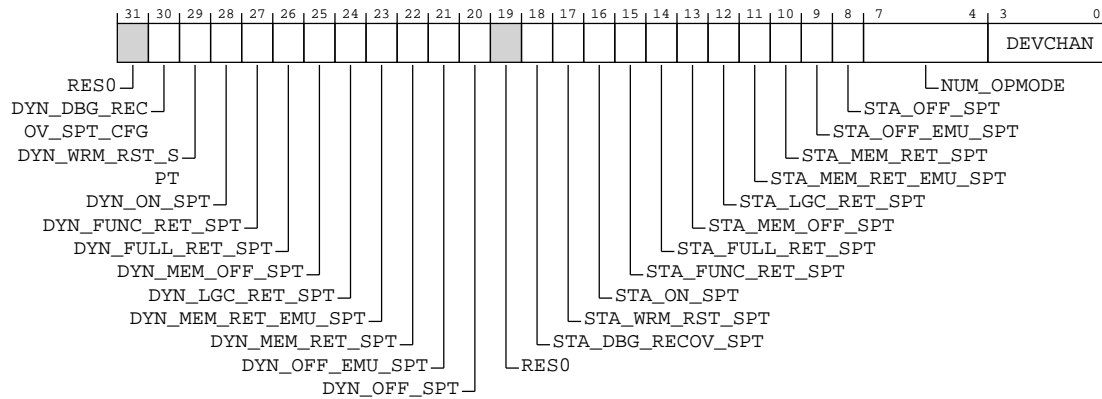


Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-77: ext\_ppu\_idr0 bit assignments**



**Table B-145: PPU\_IDR0 bit descriptions**

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30]	DYN_DBG_RECOV_SPT_CFG	Dynamic DBG_RECOV support. <b>0b1</b> Dynamic DBG_RECOV supported.	0b1
[29]	DYN_WRM_RST_SPT	Dynamic WARM_RST support. <b>0b1</b> Dynamic WARM_RST not supported.	0b1
[28]	DYN_ON_SPT	Dynamic ON support. <b>0b1</b> Dynamic ON supported.	0b1
[27]	DYN_FUNC_RET_SPT	Dynamic DYN_FUNC_RET_SPT support. <b>0b1</b> Dynamic DYN_FUNC_RET_SPT supported.	0b1
[26]	DYN_FULL_RET_SPT	Dynamic DYN_FULL_RET_SPT support. <b>0b0</b> Dynamic DYN_FULL_RET_SPT not supported.	0b1
[25]	DYN_MEM_OFF_SPT	Dynamic MEM_OFF support. <b>0b0</b> Dynamic MEM_OFF not supported.	0b0
[24]	DYN_LGC_RET_SPT	Dynamic LOGIC_RET support. <b>0b0</b> Dynamic LOGIC_RET not supported.	0b0
[23]	DYN_MEM_RET_EMU_SPT	Dynamic DYN_MEM_RET_EMU_SPT support. <b>0b1</b> Dynamic DYN_MEM_RET_EMU_SPT supported.	0b1



Bits	Name	Description	Reset
[22]	DYN_MEM_RET_SPT	Dynamic DYN_MEM_RET_SPT support. <b>0b1</b> Dynamic DYN_MEM_RET_SPT supported.	0b1
[21]	DYN_OFF_EMU_SPT	Dynamic OFF_EMU support. <b>0b1</b> Dynamic OFF_EMU supported.	0b1
[20]	DYN_OFF_SPT	Dynamic OFF support. <b>0b1</b> Dynamic OFF supported.	0b1
[19]	<b>RES0</b>	Reserved	<b>RES0</b>
[18]	STA_DBG_RECOV_SPT	DBG_RECOV support. <b>0b1</b> DBG_RECOV supported.	0b1
[17]	STA_WRM_RST_SPT	WARM_RST support. <b>0b1</b> WRM_RST supported.	0b1
[16]	STA_ON_SPT	ON support. <b>0b1</b> ON supported.	0b1
[15]	STA_FUNC_RET_SPT	FUNC_RET support. <b>0b1</b> FUNC_RET supported.	0b1
[14]	STA_FULL_RET_SPT	FULL_RET support. <b>0b0</b> FULL_RET not supported.	0b1
[13]	STA_MEM_OFF_SPT	MEM_OFF support. <b>0b0</b> MEM_OFF not supported.	0b0
[12]	STA_LGC_RET_SPT	LOGIC_RET support. <b>0b0</b> LOGIC_RET not supported.	0b0
[11]	STA_MEM_RET_EMU_SPT	MEM_RET_EMU support. <b>0b1</b> MEM_RET_EMU supported.	0b1
[10]	STA_MEM_RET_SPT	MEM_RET support. <b>0b1</b> MEM_RET supported.	0b1
[9]	STA_OFF_EMU_SPT	OFF_EMU support. <b>0b1</b> OFF_EMU supported.	0b1

Bits	Name	Description	Reset
[8]	STA_OFF_SPT	OFF support.  <b>0b1</b> OFF supported.	0b1
[7:4]	NUM_OPMODE	No. of operating modes supported, minus 1.  <b>0b1111</b> 16 operating modes supported.	0b1111
[3:0]	DEVCHAN	No. of Device Interface Channels.  <b>0b0000</b> 0 (P-channel PPU).	0b0000

Accessibility

This interface is accessible as follows:

RO

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

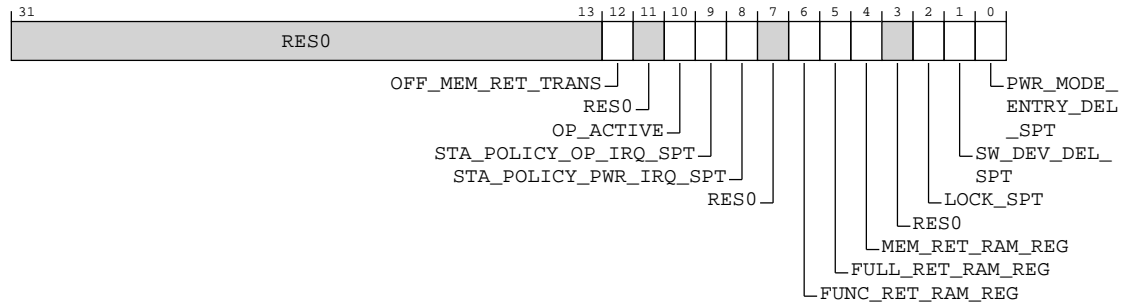
xxxx	xxxx	xxxx	xxxx	xxx1	x111	x000	x110
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-78: ext\_ppu\_idr1 bit assignments**



**Table B-146: 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. <b>0b1</b> OFF to MEM_RET direct transition supported.	0b1
[11]	RES0	Reserved	RES0
[10]	OP_ACTIVE	Operating mode use model for dynamic transitions. <b>0b1</b> Independent use model.	0b1
[9]	STA_POLICY_OP_IRQ_SPT	Operating policy transition completion event status. <b>0b1</b> Operating policy transition completion events supported.	0b1
[8]	STA_POLICY_PWR_IRQ_SPT	Power policy transition completion event status. <b>0b1</b> 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. <b>0b0</b> ext-PPU_FUNRR is reserved.	0b0
[5]	FULL_RET_RAM_REG	Indicates if the ext-PPU_FULRR register is present or reserved. <b>0b0</b> ext-PPU_FULRR is reserved.	0b0

Bits	Name	Description	Reset
[4]	MEM_RET_RAM_REG	Indicates if the ext-PPU_MEMRR register is present or reserved.  <b>0b0</b> ext-PPU_MEMRR is reserved.	0b0
[3]	<b>RES0</b>	Reserved	<b>RES0</b>
[2]	LOCK_SPT	Indicates if the lock and the lock interrupt event are supported.  <b>0b1</b> Lock and the lock interrupt event are supported.	0b1
[1]	SW_DEV_DEL_SPT	Software device delay control configuration support.  <b>0b1</b> Software device delay control configuration supported.	0b1
[0]	PWR_MODE_ENTRY_DEL_SPT	Power mode entry delay support.  <b>0b0</b> Power mode entry delay not supported.	0b0

### Accessibility

This interface is accessible as follows:

RO

### 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

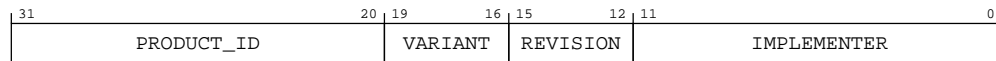
RO

#### Reset value

0000 1011 0110 0010 0000 0100 0011 1011

## Bit descriptions

**Figure B-79: ext\_ppu\_iidr bit assignments**



**Table B-147: PPU\_IIDR bit descriptions**

Bits	Name	Description	Reset
[31:20]	PRODUCT_ID	Value identifying the PPU part. <b>0b000010110110</b> Power Policy Unit.	0x0B6
[19:16]	VARIANT	Value used to distinguish PPU variants, or major revisions of the PPU. <b>0b0000</b> PPU variant 0. <b>0b0001</b> PPU variant 1. <b>0b0010</b> PPU variant 2. <b>0b0011</b> PPU variant 3. <b>0b0100</b> PPU variant 4.	0b0010
[15:12]	REVISION	Value used to distinguish minor revisions of the PPU. <b>0b0000</b> PPU revision 0. <b>0b0001</b> PPU revision 1. <b>0b0010</b> PPU revision 2. <b>0b0011</b> PPU revision 3. <b>0b0100</b> PPU revision 4.	0b0000
[11:0]	IMPLEMENTER	Implementer identification. <b>0b010000111011</b> Arm Limited.	0x43B

## Accessibility

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-80: ext\_ppu\_aidr bit assignments

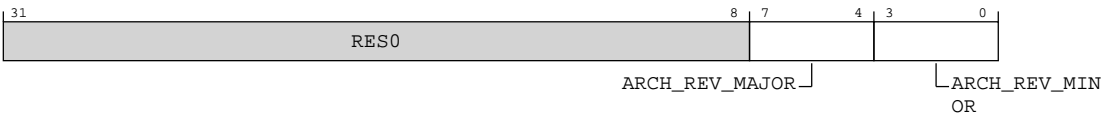


Table B-148: PPU\_AIDR bit descriptions

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

Bits	Name	Description	Reset
[3:0]	ARCH_REV_MINOR	PPU architecture minor revision.  <b>0b0010</b> PPU architecture minor revision 2.	0b0010

Accessibility

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-81: ext\_ppu\_pidr4 bit assignments

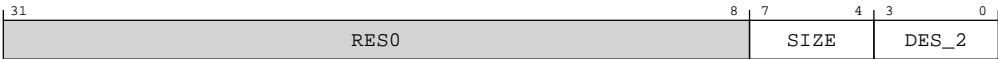


Table B-149: PPU\_PIDR4 bit descriptions

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

Accessibility

This interface is accessible as follows:

RO

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

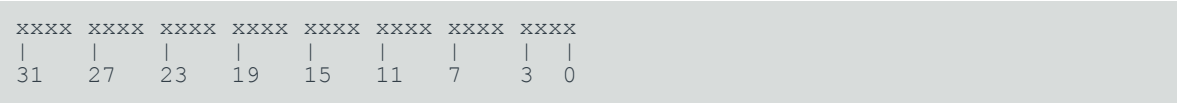
Register offset

0xFD4

Access type

RO

Reset value



Note

Where the reset reads xxxx, see individual bits.



Bit descriptions

Figure B-82: ext\_ppu\_pidr5 bit assignments

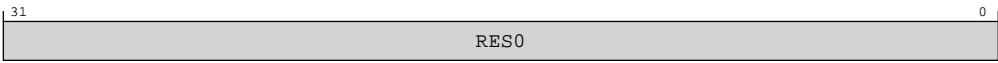


Table B-150: PPU\_PIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

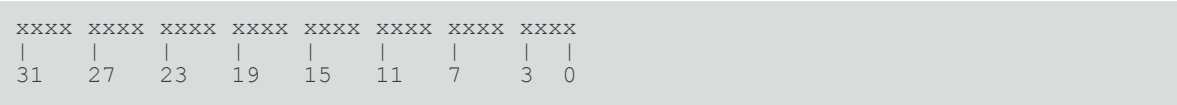
Register offset

0xFD8

Access type

RO

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-83: ext\_ppu\_pidr6 bit assignments

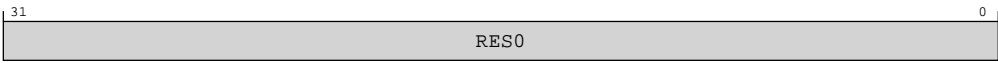


Table B-151: PPU\_PIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-84: ext\_ppu\_pidr7 bit assignments

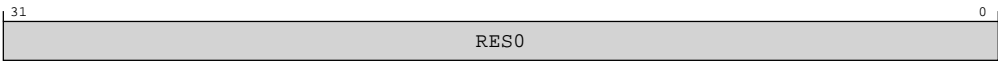


Table B-152: PPU\_PIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-85: ext\_ppu\_pidr0 bit assignments

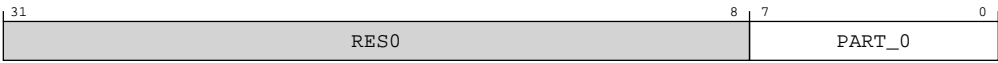


Table B-153: PPU\_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0].  0b10110110 DSU-120AE Power Policy Unit. Bits [7:0] of part number 0x0B6.	0xB6

Accessibility

This interface is accessible as follows:

RO

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





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-86: ext\_ppu\_pidr1 bit assignments

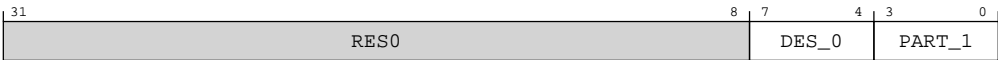


Table B-154: PPU\_PIDR1 bit descriptions

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-120AE Power Policy Unit. Bits [11:8] of part number 0x0B6.	0b0000

Accessibility

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-87: ext\_ppu\_pidr2 bit assignments



Table B-155: PPU\_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVISION	Component major revision. <b>0b0000</b> Component major revision 0. <b>0b0001</b> Component major revision 1. <b>0b0010</b> Component major revision 2. <b>0b0011</b> Component major revision 3. <b>0b0100</b> Component major revision 4.	0b0010
[3]	JEDEC	JEDEC assignee. <b>0b1</b> JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. <b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-88: ext\_ppu\_pidr3 bit assignments



Table B-156: PPU\_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVAND	Component minor revision.  <b>0b0000</b> Component minor revision 0.  <b>0b0001</b> Component minor revision 1.  <b>0b0010</b> Component minor revision 2.  <b>0b0011</b> Component minor revision 3.  <b>0b0100</b> Component minor revision 4.	0b0000
[3:0]	CMOD	Customer Modified.  <b>0b0000</b> The component is not modified from the original design.	0b0000

Accessibility

This interface is accessible as follows:

RO

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

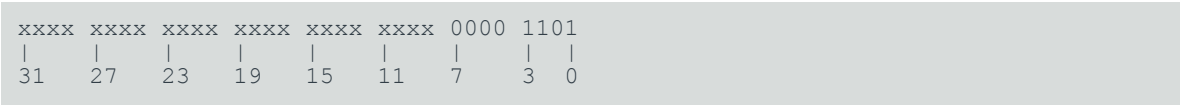
Register offset

0xFF0

Access type

RO

Reset value







Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-89: ext\_ppu\_cidr0 bit assignments

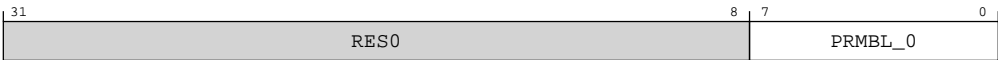


Table B-157: PPU\_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	CoreSight component identification preamble.  0b00001101 CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-90: ext\_ppu\_cidr1 bit assignments



Table B-158: PPU\_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	CLASS	CoreSight component class. <b>0b1111</b> CoreLink component.	0b1111
[3:0]	PRMBL_1	CoreSight component identification preamble. <b>0b0000</b> CoreSight component identification preamble.	0b0000

Accessibility

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-91: ext\_ppu\_cidr2 bit assignments



Table B-159: PPU\_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	CoreSight component identification preamble. <b>0b000000101</b> CoreSight component identification preamble.	0x05

Accessibility

This interface is accessible as follows:

RO

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

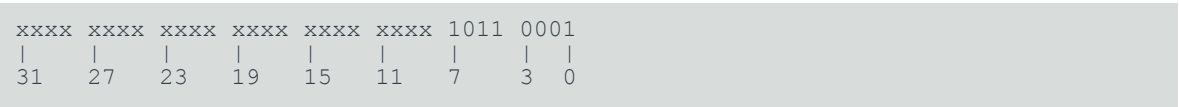
Register offset

0xFFC

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-92: ext\_ppu\_cidr3 bit assignments

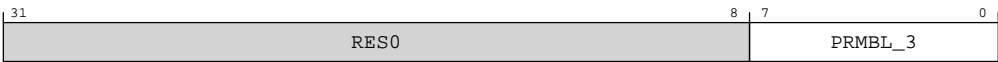


Table B-160: PPU\_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_3	CoreSight component identification preamble. <b>0b10110001</b> CoreSight component identification preamble.	0xB1

Accessibility

This interface is accessible as follows:

RO

B.1.5 External cluster AMU registers summary

The cluster *Activity Monitor Unit* (AMU) registers are only accessible from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the cluster AMU registers that are accessed externally (memory-mapped) from the utility bus of the DSU-120AE. For more information about

a register, click on the register name in the table. For more information on the architecture of the AMU registers, see [Arm® CoreSight™ Performance Monitoring Unit Architecture](#).



- The cluster AMU registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- The part number is 0x04EA.
- The base address for the cluster AMU registers is 0x040000.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table B-161: CLUSTERAMU registers summary**

Offset	Name	Reset	Width	Description
0x0	<a href="#">CLUSTERAMU_AMEVCNTR0</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x8	<a href="#">CLUSTERAMU_AMEVCNTR1</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x10	<a href="#">CLUSTERAMU_AMEVCNTR2</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x18	<a href="#">CLUSTERAMU_AMEVCNTR3</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x20	<a href="#">CLUSTERAMU_AMEVCNTR4</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Event Count Registers
0x400	<a href="#">CLUSTERAMU_AMEVTYPEPER0</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x404	<a href="#">CLUSTERAMU_AMEVTYPEPER1</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x408	<a href="#">CLUSTERAMU_AMEVTYPEPER2</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x40C	<a href="#">CLUSTERAMU_AMEVTYPEPER3</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0x410	<a href="#">CLUSTERAMU_AMEVTYPEPER4</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Event Type Registers
0xC00	<a href="#">CLUSTERAMU_AMCNTENSET</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Count Enable Set register
0xC20	<a href="#">CLUSTERAMU_AMCNTENCLR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Count Enable Clear register
0xE00	<a href="#">CLUSTERAMU_AMCFGR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Configuration Register
0xE04	<a href="#">CLUSTERAMU_AMCR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Control Register
0xE08	<a href="#">CLUSTERAMU_AMIIDR</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Implementation Identification register
0xFA8	<a href="#">CLUSTERAMU_AMDEVAFF</a>	See individual bit resets.	64-bit	Cluster Activity Monitors Device Affinity register
0xFBC	<a href="#">CLUSTERAMU_AMDEVARCH</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Device Architecture register
0xFC8	<a href="#">CLUSTERAMU_AMDEVID</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Device ID register
0xFCC	<a href="#">CLUSTERAMU_AMDEVTYPE</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Device Type register
0xFD0	<a href="#">CLUSTERAMU_AMPIDR4</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 4
0xFE0	<a href="#">CLUSTERAMU_AMPIDR0</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 0
0xFE4	<a href="#">CLUSTERAMU_AMPIDR1</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 1
0xFE8	<a href="#">CLUSTERAMU_AMPIDR2</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 2
0xFEC	<a href="#">CLUSTERAMU_AMPIDR3</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Peripheral Identification Register 3
0xFF0	<a href="#">CLUSTERAMU_AMCIDR0</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 0
0xFF4	<a href="#">CLUSTERAMU_AMCIDR1</a>	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 1

Offset	Name	Reset	Width	Description
0xFF8	CLUSTERAMU_AMCIDR2	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 2
0xFFC	CLUSTERAMU_AMCIDR3	See individual bit resets.	32-bit	Cluster Activity Monitors Component Identification Register 3

### B.1.5.1 CLUSTERAMU\_AMEVCNTR0, Cluster Activity Monitors Event Count Registers

Holds event counter 0, which counts events.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

64

##### Component

CLUSTERAMU

##### Register offset

0x0

##### Access type

RW

##### Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3



Note

Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure B-93: ext\_clusteramu\_amevcntr0 bit assignments

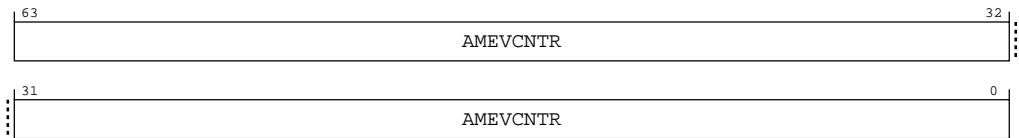


Table B-162: CLUSTERAMU\_AMEVCNTR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	AMEVCNTR	Event counter 0.	64 {x}

Accessibility

This interface is accessible as follows:

RW

B.1.5.2 CLUSTERAMU\_AMEVCNTR1, Cluster Activity Monitors Event Count Registers

Holds event counter 1, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offset

0x8

Access type

RW

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-94: ext\_clusteramu\_amevcntr1 bit assignments

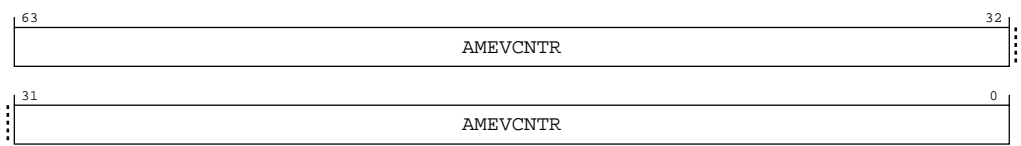


Table B-163: CLUSTERAMU\_AMEVCNTR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	AMEVCNTR	Event counter 1.	64 {x}

Accessibility

This interface is accessible as follows:

RW

B.1.5.3 CLUSTERAMU\_AMEVCNTR2, Cluster Activity Monitors Event Count Registers

Holds event counter 2, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offset

0x10

Access type

RW

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-95: ext\_clusteramu\_amevcntr2 bit assignments

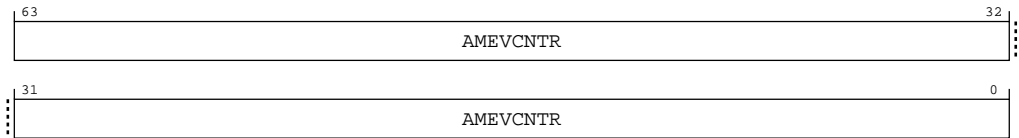


Table B-164: CLUSTERAMU\_AMEVCNTR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	AMEVCNTR	Event counter 2.	64 { x }

Accessibility

This interface is accessible as follows:

RW

B.1.5.4 CLUSTERAMU\_AMEVCNTR3, Cluster Activity Monitors Event Count Registers

Holds event counter 3, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offset

0x18

Access type

RW

Reset value

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-96: ext\_clusteramu\_amevcntr3 bit assignments

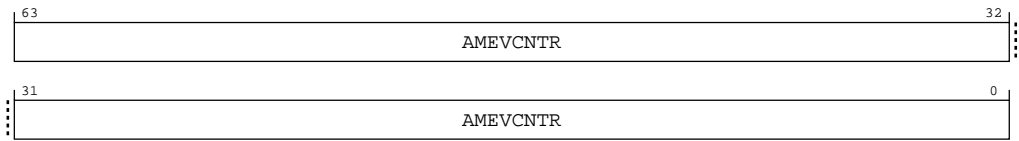


Table B-165: CLUSTERAMU\_AMEVCNTR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	AMEVCNTR	Event counter 3.	64 { x }

Accessibility

This interface is accessible as follows:

RW

B.1.5.5 CLUSTERAMU\_AMEVCNTR4, Cluster Activity Monitors Event Count Registers

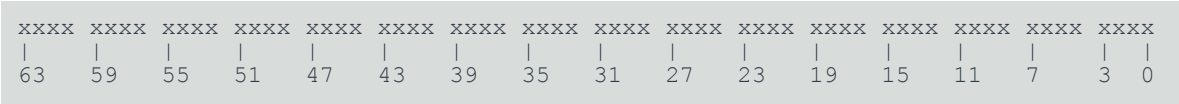
Holds event counter 4, which counts events.

Configurations

This register is available in all configurations.

- Attributes
- Width
- 64
- Component
- CLUSTERAMU
- Register offset
- 0x20
- Access type
- RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-97: ext\_clusteramu\_amevcntr4 bit assignments

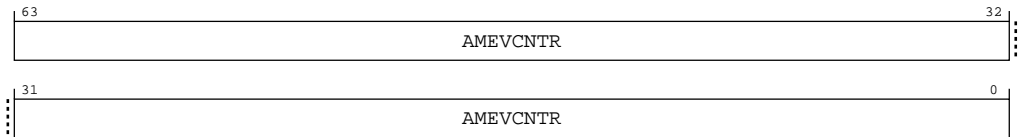


Table B-166: CLUSTERAMU\_AMEVCNTR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	AMEVCNTR	Event counter 4.	64 { x }

Accessibility

This interface is accessible as follows:

RW

B.1.5.6 CLUSTERAMU\_AMEVTYPER0, Cluster Activity Monitors Event Type Registers

Configures event counter n, where n is 0 to 31.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERAMU


Register offset

0x400

Access type  
RO

Reset value



 Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-98: ext\_clusteramu\_amevtyper0 bit assignments

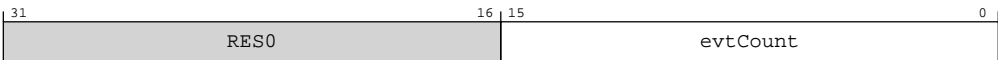


Table B-167: CLUSTERAMU\_AMEVTYPER0 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERAMU_AMEVCNTR<n>.	16 {x}

Accessibility

This interface is accessible as follows:

RO

B.1.5.7 CLUSTERAMU\_AMEVTYPER1, Cluster Activity Monitors Event Type Registers

Configures event counter n, where n is 0 to 31.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x404

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-99: ext\_clusteramu\_amevtyper1 bit assignments

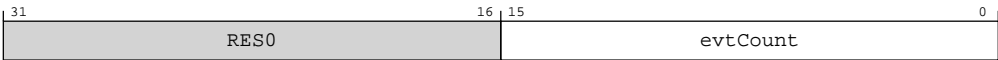


Table B-168: CLUSTERAMU\_AMEVTYPER1 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERAMU_AMEVCNTR<n>.	16 {x}

Accessibility

This interface is accessible as follows:

RO

B.1.5.8 CLUSTERAMU\_AMEVTYPER2, Cluster Activity Monitors Event Type Registers

Configures event counter n, where n is 0 to 31.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component  
CLUSTERAMU

Register offset  
0x408

Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-100: ext\_clusteramu\_amevtyper2 bit assignments

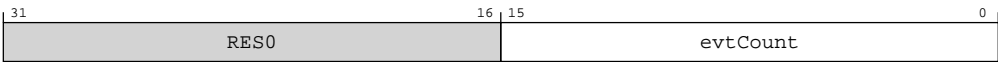


Table B-169: CLUSTERAMU\_AMEVTYPER2 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERAMU_AMEVCNTR<n>.	16 {x}

Accessibility

This interface is accessible as follows:

RO

B.1.5.9 CLUSTERAMU\_AMEVTYPER3, Cluster Activity Monitors Event Type Registers

Configures event counter n, where n is 0 to 31.

Configurations

If event counter n is not implemented then accesses to this register are RES0.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x40C

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-101: ext\_clusteramu\_amevtyper3 bit assignments

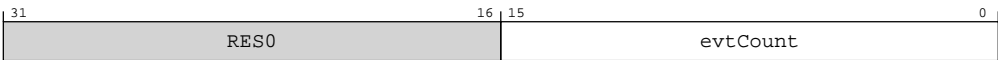


Table B-170: CLUSTERAMU\_AMEVTYPER3 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERAMU_AMEVCNTR<n>.	16 {x}

Accessibility

This interface is accessible as follows:

RO

### B.1.5.10 CLUSTERAMU\_AMEVTYPER4, Cluster Activity Monitors Event Type Registers

Configures event counter n, where n is 0 to 31.

#### Configurations

If event counter n is not implemented then accesses to this register are RES0.

#### Attributes

##### Width

32

##### Component

CLUSTERAMU

##### Register offset

0x410

##### Access type

RO

##### Reset value



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure B-102: ext\_clusteramu\_amevtyper4 bit assignments

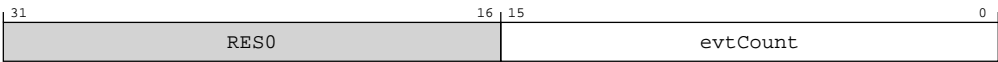


Table B-171: CLUSTERAMU\_AMEVTYPER4 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERAMU_AMEVCNTR<n>.	16 {x}

#### Accessibility

This interface is accessible as follows:



RO

B.1.5.11 CLUSTERAMU\_AMCNTENSET, Cluster Activity Monitors Count Enable Set register

Enables the implemented event counters CLUSTERAMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xC00

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-103: ext\_clusteramu\_amcntenset bit assignments

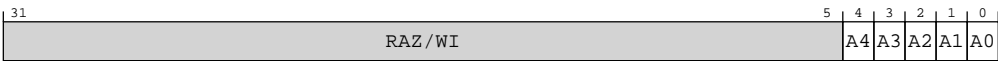


Table B-172: CLUSTERAMU\_AMCNTENSET bit descriptions

Bits	Name	Description	Reset
[31:5]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[4]	A4	<p>Event counter enable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[3]	A3	<p>Event counter enable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[2]	A2	<p>Event counter enable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[1]	A1	<p>Event counter enable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[0]	A0	<p>Event counter enable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x

## Accessibility

This interface is accessible as follows:

RW

B.1.5.12 CLUSTERAMU\_AMCNTENCLR, Cluster Activity Monitors Count Enable  
Clear register

Disables the implemented event counters CLUSTERAMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

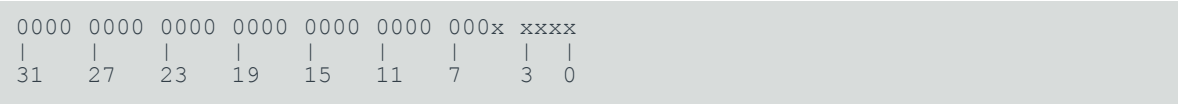
Register offset

0xC20

Access type

RW

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-104: ext\_clusteramu\_amcntenclr bit assignments



Table B-173: CLUSTERAMU\_AMCNTENCLR bit descriptions

Bits	Name	Description	Reset
[31:5]	RAZ/ WI	Reserved	RAZ/ WI

Bits	Name	Description	Reset
[4]	A4	<p>Event counter disable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[3]	A3	<p>Event counter disable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[2]	A2	<p>Event counter disable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[1]	A1	<p>Event counter disable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x
[0]	A0	<p>Event counter disable bit for ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERAMU_AMCFGR.N is less than 32, bits [31:ext-CLUSTERAMU_AMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERAMU_AMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERAMU_AMEVCNTR&lt;n&gt;.</p>	x

## Accessibility

This interface is accessible as follows:

RW

B.1.5.13 CLUSTERAMU\_AMCFGR, Cluster Activity Monitors Configuration Register

Contains AMU-specific configuration data.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xE00

Access type

RO

Reset value

0000	xxx0	00xx	xxxx	xx11	1111	0000	0100
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-105: ext\_clusteramu\_amcfgr bit assignments

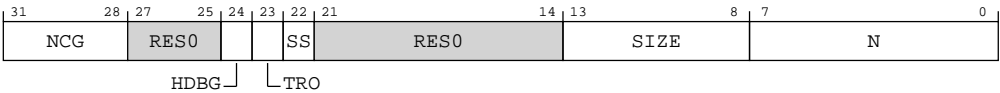


Table B-174: CLUSTERAMU\_AMCFGR bit descriptions

Bits	Name	Description	Reset
[31:28]	NCG	Number of Monitor Groups Implemented.  0b0000 Monitor Groups not implemented.	0b0000
[27:25]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[24]	HDBG	Halt on debug. <b>0b0</b> Halt on debug not supported	0b0
[23]	TRO	Trace feature support. <b>0b0</b> Trace features not supported.	0b0
[22]	SS	Snapshot support. <b>0b0</b> Snapshot not supported.	0b0
[21:14]	RES0	Reserved	RES0
[13:8]	SIZE	Size of counters, minus one. This field defines the size of the largest counter implemented by the Activity Monitors Unit.  This field is used by software to determine the spacing of the counters in the memory-map. <b>0b111111</b> The largest counter is 64-bits. Counters are at doubleword-aligned addresses.	0b111111
[7:0]	N	Number of counters implemented, minus one. <b>0b00000100</b> Five event counters implemented.	0x04

## Accessibility

This interface is accessible as follows:

RO

### B.1.5.14 CLUSTERAMU\_AMCR, Cluster Activity Monitors Control Register

Provides configuration details of the Activity Monitors implementation.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32

### Component

CLUSTERAMU

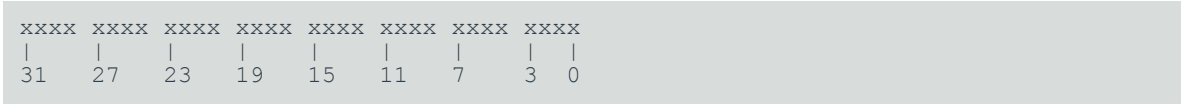
### Register offset

0xE04

### Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-106: ext\_clusteramu\_amcr bit assignments

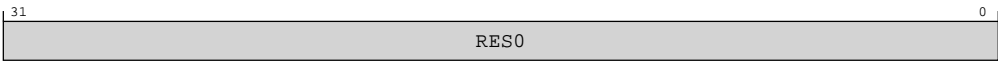


Table B-175: CLUSTERAMU\_AMCR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.1.5.15 CLUSTERAMU\_AMIIDR, Cluster Activity Monitors Implementation Identification register

Defines the implemented of the component..

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xE08

Access type

RO

Reset value

0100 1110 1100 0000 0001 0100 0011 1011

Bit descriptions

Figure B-107: ext\_clusteramu\_amiidr bit assignments

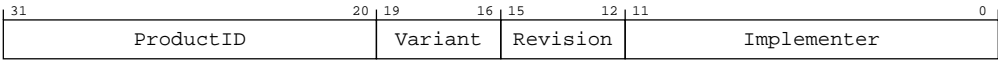


Table B-176: CLUSTERAMU\_AMIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Value identifying the AMU Component.  0b010011101100 DSU-120AE Cluster AMU.	0x4EC
[19:16]	Variant	Value used to distinguish product variants, or major revisions of the product.  0b0000 Product variant 0.	0b0000
[15:12]	Revision	Value used to distinguish minor revisions of the product.  0b0000 Product revision 0.  0b0001 Product revision 1.	0b0001
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the AMU Component.  For an Arm implementation, bits[11:0] are 0x43B.  0b010000111011 Arm implementation.	0x43B

Accessibility

This interface is accessible as follows:

RO

B.1.5.16 CLUSTERAMU\_AMDEVAFF, Cluster Activity Monitors Device Affinity register

Allows the external agent to determine which PE in a multiprocessor system the Activity Monitor component relates to.

Configurations

This register is available in all configurations.



Attributes

Width

64

Component

CLUSTERAMU

Register offset

0xFA8

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	x0xx	xxx0	xxxx	xxxx	1000	0000	0000	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-108: ext\_clusteramu\_amdevaff bit assignments

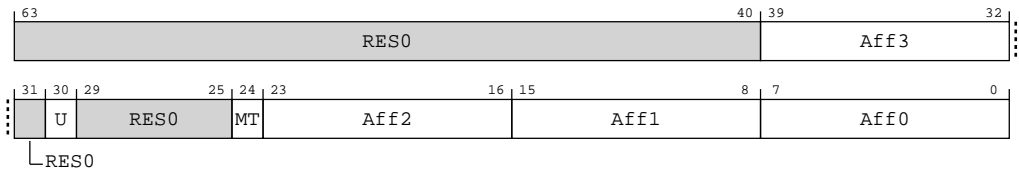


Table B-177: CLUSTERAMU\_AMDEVAFF bit descriptions

Bits	Name	Description	Reset
[63:40]	RES0	Reserved	RES0
[39:32]	Aff3	Affinity level 3. Value read from the CFGMPIDRAFF3 configuration pins.	8 {x}
[31]	RES0	Reserved	RES0
[30]	U	Uniprocessor/Multiprocessor system. <b>0b0</b> 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. <b>0b0</b> Activity of PEs at the lowest affinity level is largely independent.	0b0

Bits	Name	Description	Reset
[23:16]	Aff2	Affinity level 2. Value read from the CFGMPIDRAFF2 configuration pins.	8 {x}
[15:8]	Aff1	Affinity level 1. <b>0b10000000</b> Affinity with all cores in cluster.	0x80
[7:0]	Aff0	Affinity level 0. <b>0b00000000</b> Affinity with all core threads in cluster.	0x00

## Accessibility

This interface is accessible as follows:

RO

## B.1.5.17 CLUSTERAMU\_AMDEVARCH, Cluster Activity Monitors Device Architecture register

Identifies the programmers' model architecture of the Activity Monitor component.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32

### Component

CLUSTERAMU

### Register offset

0xFBC

### Access type

RO

### Reset value

0100 0111 0111 0000 0000 1010 0110 0110

## Bit descriptions

**Figure B-109: ext\_clusteramu\_amdevarch bit assignments**

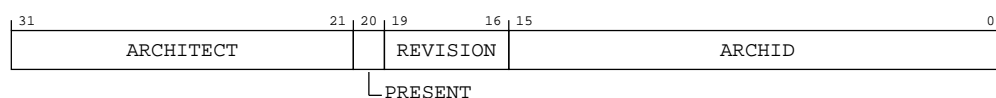


Table B-178: CLUSTERAMU\_AMDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. <b>0b01000111011</b> JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. <b>0b1</b> DEVARCH information present.	0b1
[19:16]	REVISION	Revision. <b>0b0000</b> Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. <b>0b00000101001100110</b> Activity Monitor (AMU) architecture AMUv1.	0x0A66

Accessibility

This interface is accessible as follows:

RO

B.1.5.18 CLUSTERAMU\_AMDEVID, Cluster Activity Monitors Device ID register

Provides information about features of the Activity Monitors implementation.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFC8

Access type

RO

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-110: ext\_clusteramu\_amdevvid bit assignments

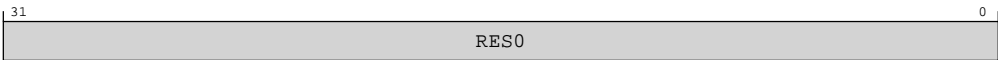


Table B-179: CLUSTERAMU\_AMDEVID bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.1.5.19 CLUSTERAMU\_AMDEVTYPE, Cluster Activity Monitors Device Type register

Indicates to a debugger that this component is part of a processor Activity monitor interface.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

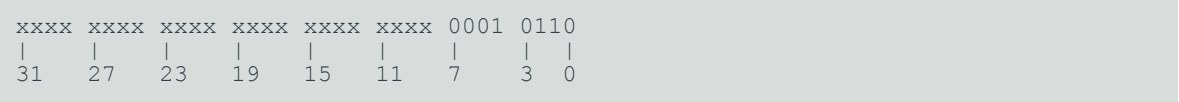
Register offset

0xFCC

Access type

RO

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-111: ext\_clusteramu\_amdevtype bit assignments

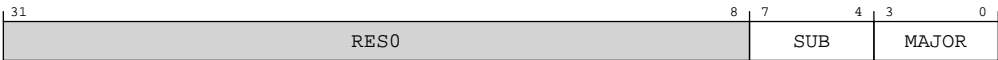


Table B-180: CLUSTERAMU\_AMDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SUB	Subtype. <b>0b0001</b> Associated with a processor.	0b0001
[3:0]	MAJOR	Major type. <b>0b0110</b> Activity Monitor.	0b0110

Accessibility

This interface is accessible as follows:

RO

B.1.5.20 CLUSTERAMU\_AMPIDR4, Cluster Activity Monitors Peripheral Identification Register 4

Provides information to identify a Activity Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFD0

Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-112: ext\_clusteramu\_ampidr4 bit assignments



Table B-181: CLUSTERAMU\_AMPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. <b>0b0000</b> The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. <b>0b0100</b> Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.1.5.21 CLUSTERAMU\_AMPIDR0, Cluster Activity Monitors Peripheral Identification Register 0

Provides information to identify a Activity Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFE0

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-113: ext\_clusteramu\_ampidr0 bit assignments

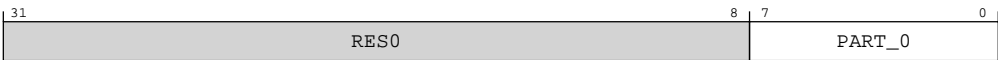


Table B-182: CLUSTERAMU\_AMPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0].  0b11101100 DSU-120AE Cluster AMU. Bits [7:0] of part number 0x4EC.	0xEC

Accessibility

This interface is accessible as follows:

RO

B.1.5.22 CLUSTERAMU\_AMPIDR1, Cluster Activity Monitors Peripheral Identification Register 1

Provides information to identify a Activity Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFE4

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-114: ext\_clusteramu\_ampidr1 bit assignments

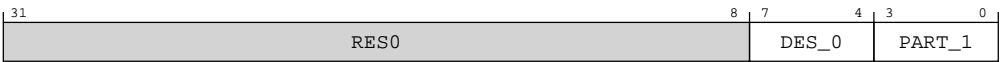


Table B-183: CLUSTERAMU\_AMPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0]. <b>0b1011</b> Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011



Bits	Name	Description	Reset
[3:0]	PART_1	Part number bits [11:8].  <b>0b0100</b> DSU-120AE Cluster AMU. Bits [11:8] of part number 0x4EC.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.1.5.23 CLUSTERAMU\_AMPIDR2, Cluster Activity Monitors Peripheral Identification Register 2

Provides information to identify a Activity Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFE8

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0000	1011
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-115: ext\_clusteramu\_ampidr2 bit assignments

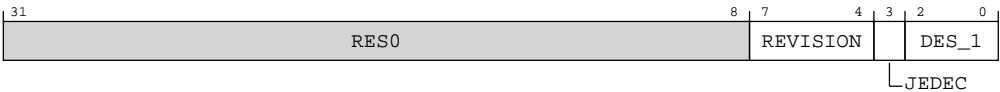


Table B-184: CLUSTERAMU\_AMPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVISION	Component major revision.  <b>0b0000</b> Component major revision 0.  <b>0b0001</b> Component major revision 1.  For DSU-120AE: <ul style="list-style-type: none"><li>Major revision 0 corresponds to r0p0.</li><li>Major revision 1 corresponds to r0p1.</li></ul>	0b0000
[3]	JEDEC	JEDEC assignee.  <b>0b1</b> JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4].  <b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

This interface is accessible as follows:

RO

B.1.5.24 CLUSTERAMU\_AMPIDR3, Cluster Activity Monitors Peripheral Identification Register 3

Provides information to identify a Activity Monitor component.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component  
CLUSTERAMU

Register offset  
0xFEC

Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-116: ext\_clusteramu\_ampidr3 bit assignments

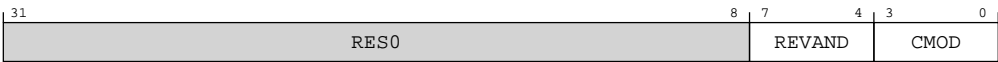


Table B-185: CLUSTERAMU\_AMPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVAND	Component minor revision. <b>0b0000</b> Component minor revision 0.	xxxx
[3:0]	CMOD	Customer Modified. <b>0b0000</b> The component is not modified from the original design.	xxxx

Accessibility

This interface is accessible as follows:

RO

B.1.5.25 CLUSTERAMU\_AMCIDR0, Cluster Activity Monitors Component Identification Register 0

Provides information to identify a Activity Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

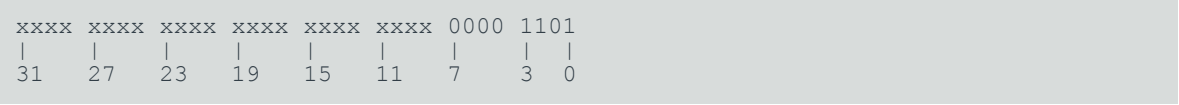
Register offset

0xFF0

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-117: ext\_clusteramu\_amcidr0 bit assignments

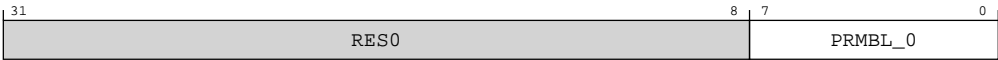


Table B-186: CLUSTERAMU\_AMCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	Preamble. <b>0b00001101</b> CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

B.1.5.26 CLUSTERAMU\_AMCIDR1, Cluster Activity Monitors Component Identification Register 1

Provides information to identify a Activity Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

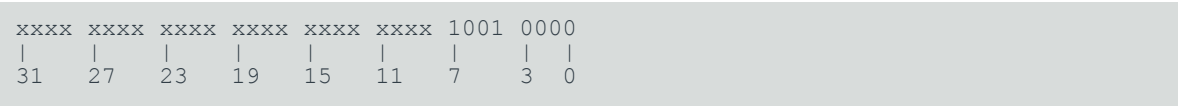
Register offset

0xFF4

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-118: ext\_clusteramu\_amcldr1 bit assignments



Table B-187: CLUSTERAMU\_AMCIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	CLASS	Component class.  0b1001 CoreSight debug component.	0b1001

Bits	Name	Description	Reset
[3:0]	PRMBL_1	Preamble.  <b>0b0000</b> CoreSight component identification preamble.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.5.27 CLUSTERAMU\_AMCIDR2, Cluster Activity Monitors Component Identification Register 2

Provides information to identify a Activity Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFF8

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0000	0101
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-119: ext\_clusteramu\_amcldr2 bit assignments

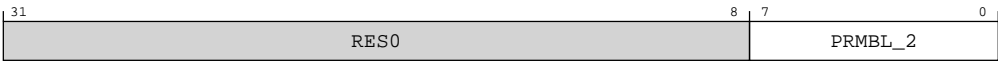


Table B-188: CLUSTERAMU\_AMCIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	Preamble.  0b00000101 CoreSight component identification preamble.	0x05

Accessibility

This interface is accessible as follows:

RO

B.1.5.28 CLUSTERAMU\_AMCIDR3, Cluster Activity Monitors Component Identification Register 3

Provides information to identify a Activity Monitor component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFFC

Access type

RO

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-120: ext\_clusteramu\_amcldr3 bit assignments

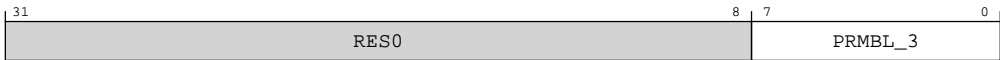


Table B-189: CLUSTERAMU\_AMCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_3	Preamble.  <b>0b10110001</b> CoreSight component identification preamble.	0xB1

Accessibility

This interface is accessible as follows:

RO

B.1.6 External cluster AE registers summary

The cluster *Automotive Enhanced* (AE) registers are only accessible from memory-mapped accesses on the utility bus.

The summary table provides an overview of all the cluster AE registers that are accessed externally (memory-mapped) from the utility bus of the DSU-120AE. For more information about a register, click on the register name in the table.



- The cluster AE registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- The base address for the cluster AE registers is 0x050000.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.



**Table B-190: CLUSTERAE registers summary**

Offset	Name	Reset	Width	Description
0x0000	CLUSTERAE_CLUSTERSLCFR	See individual bit resets.	64-bit	Cluster Split/Lock Configuration Feature Register
0x0008	CLUSTERAE_CLUSTERSLCTL	See individual bit resets.	64-bit	Cluster Split/Lock Configuration Control Register
0x0010	CLUSTERAE_CLUSTERSLSTAT	See individual bit resets.	64-bit	Cluster Split/Lock Configuration Status Register
0x0020	CLUSTERAE_CORESLCFR	See individual bit resets.	64-bit	Core Split/Lock Configuration Feature Register
0x0030	CLUSTERAE_CORESLCTL	See individual bit resets.	64-bit	Core Split/Lock Configuration Control Register
0x0040	CLUSTERAE_CORESLSTAT	See individual bit resets.	64-bit	Core Split/Lock Configuration Status Register
0x0050	CLUSTERAE_CLUSTERRECOV	See individual bit resets.	64-bit	Cluster Recovery Control Register
0x0060	CLUSTERAE_CLUSTERWRITEKEY	See individual bit resets.	64-bit	Cluster Write Key Register

### B.1.6.1 CLUSTERAE\_CLUSTERSLCFR, Cluster Split/Lock Configuration Feature Register

Describes the supported AE features of the Cluster and Cores.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

64

##### Component

CLUSTERAE

##### Register offset

0x0000

##### Access type

RO

##### Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-121: ext\_clusterae\_clusterslcfcr bit assignments

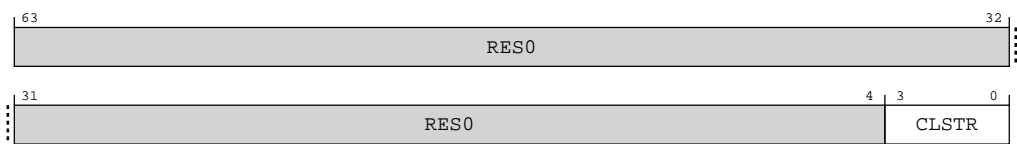


Table B-191: CLUSTERAE\_CLUSTERSLCFR bit descriptions

Bits	Name	Description	Reset
[63:4]	RES0	Reserved	RES0
[3:0]	CLSTR	Indicates the Cluster level support for Split/Lock.  <b>0b0001</b> Cluster only supports LOCKED mode (including HYBRID mode).  <b>0b0100</b> Cluster only supports SPLIT mode.  <b>0b0101</b> Cluster supports operation in SPLIT or MIXED mode.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.6.2 CLUSTERAE\_CLUSTERSLCTLR, Cluster Split/Lock Configuration Control Register

Control register for setting the Split/Lock mode of the Cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAE

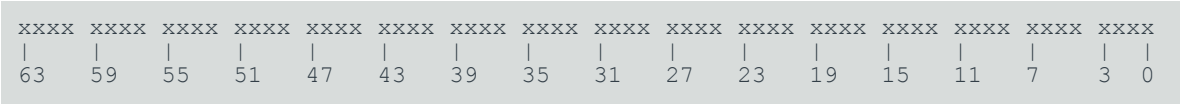
Register offset

0x0008

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-122: ext\_clusterae\_clusterslctlr bit assignments

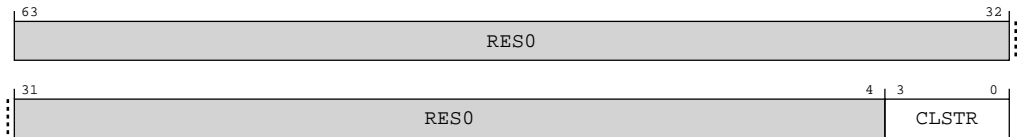


Table B-192: CLUSTERAE\_CLUSTERSLCTLR bit descriptions

Bits	Name	Description	Reset
[63:4]	RES0	Reserved	RES0
[3:0]	CLSTR	Indicates cluster support for Split/Lock.  The reset value is 0b0000 when DCLS_MODE is configured to split.  The reset value is 0b0001 when DCLS_MODE is configured to locked.  The reset value when DCLS_MODE is configured to mixed is the value of the CLUSTERSLDEFAULT pin.  <b>0b0000</b> Cluster is to be used in SPLIT mode.  <b>0b0001</b> Cluster is to be used in LOCKED mode (includes HYBRID mode).	xxxx

Accessibility

This interface is accessible as follows:

RO

B.1.6.3 CLUSTERAE\_CLUSTERSLSTAT, Cluster Split/Lock Configuration Status Register

Status register describing power and operating status of cluster and cores.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAE

Register offset

0x0010

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-123: ext\_clusterae\_clusterslstat bit assignments

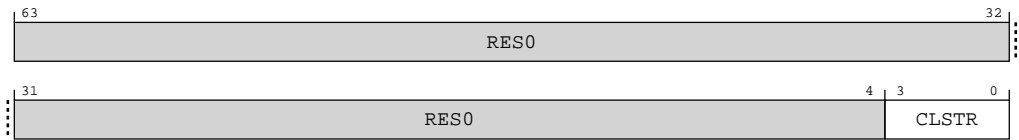


Table B-193: CLUSTERAE\_CLUSTERSLSTAT bit descriptions

Bits	Name	Description	Reset
[63:4]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[3:0]	CLSTR	Indicates the current Split/Lock status for the Cluster.  <b>0b0000</b> Cluster is powered off.  <b>0b0001</b> Cluster is powered on in LOCKED mode (includes HYBRID mode).  <b>0b0100</b> Cluster is powered on in SPLIT mode.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.6.4 CLUSTERAE\_CORESLCFR, Core Split/Lock Configuration Feature Register

Describes supported AE features of the cores.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAE

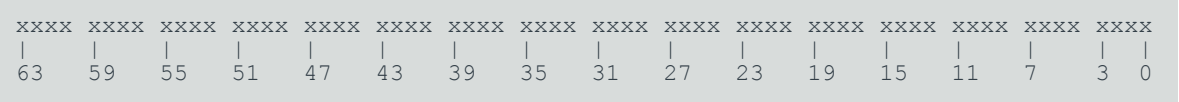
Register offset

0x0020

Access type

RO

Reset value

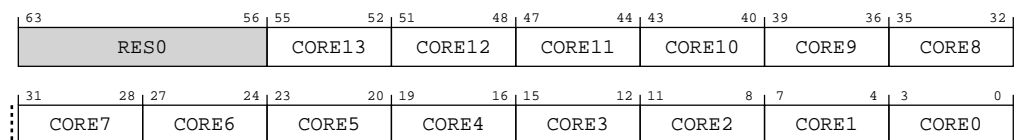


Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-124: ext\_clusterae\_coreslcfcr bit assignments**



**Table B-194: CLUSTERAE\_CORESLCFR bit descriptions**

Bits	Name	Description	Reset
[63:56]	RES0	Reserved	RES0
[55:52]	CORE13	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[51:48]	CORE12	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx

Bits	Name	Description	Reset
[47:44]	CORE11	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[43:40]	CORE10	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[39:36]	CORE9	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx

Bits	Name	Description	Reset
[35:32]	CORE8	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[31:28]	CORE7	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[27:24]	CORE6	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx



Bits	Name	Description	Reset
[23:20]	CORE5	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[19:16]	CORE4	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[15:12]	CORE3	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx

Bits	Name	Description	Reset
[11:8]	CORE2	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[7:4]	CORE1	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx
[3:0]	CORE0	<p>Indicates core N support for Split/Lock.</p> <p><b>0b0000</b> Core not present.</p> <p><b>0b0001</b> Core supports operation only in LOCKED mode as the primary core.</p> <p><b>0b0100</b> Core supports operation in SPLIT mode only.</p> <p><b>0b0101</b> Core supports both SPLIT and LOCKED mode, as the primary core.</p> <p><b>0b0111</b> Core supports both SPLIT and LOCKED mode, as the redundant core.</p>	xxxx

## Accessibility

This interface is accessible as follows:

RO

B.1.6.5 CLUSTERAE\_CORESLCTLR, Core Split/Lock Configuration Control Register

Control register for setting the Split/Lock mode of each core.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAE

Register offset

0x0030

Access type

RO

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
0															



Where the reset reads xxxx, see individual bits.

Bit descriptions

Indicates request settings for core N Split/Lock

Figure B-125: ext\_clusterae\_coreslctlr bit assignments

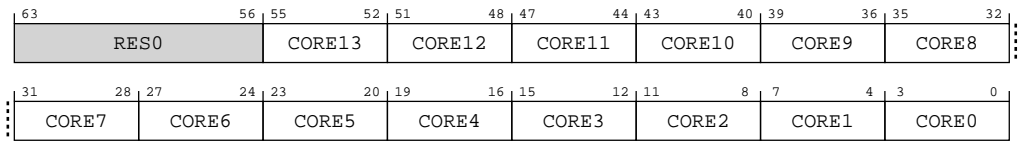


Table B-195: CLUSTERAE\_CORESLCTLR bit descriptions

Bits	Name	Description	Reset
[63:56]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[55:52]	CORE13	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[51:48]	CORE12	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[47:44]	CORE11	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[43:40]	CORE10	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx

Bits	Name	Description	Reset
[39:36]	CORE9	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[35:32]	CORE8	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[31:28]	CORE7	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[27:24]	CORE6	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx

Bits	Name	Description	Reset
[23:20]	CORE5	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[19:16]	CORE4	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[15:12]	CORE3	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[11:8]	CORE2	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx

Bits	Name	Description	Reset
[7:4]	CORE1	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx
[3:0]	CORE0	<p>Indicates core N support for Split/Lock.</p> <p>The reset value is 0b0000 when DCLS_MODE is configured to split.</p> <p>The reset value is 0b0001 when DCLS_MODE is configured to locked.</p> <p>The reset value when DCLS_MODE is configured to mixed is the relevant bit of CORESLDEFAULT pin.</p> <p><b>0b0000</b> Core is to be used in SPLIT mode.</p> <p><b>0b0001</b> Core is to be used in LOCKED mode (includes HYBRID mode).</p>	xxxx

## Accessibility

This interface is accessible as follows:

RO

## B.1.6.6 CLUSTERAE\_CORESLSTAT, Core Split/Lock Configuration Status Register

Status register showing power and operating status of each core.

## Configurations

This register is available in all configurations.

## Attributes

### Width

64

### Component

CLUSTERAE

### Register offset

0x0040

### Access type

RO

## Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0

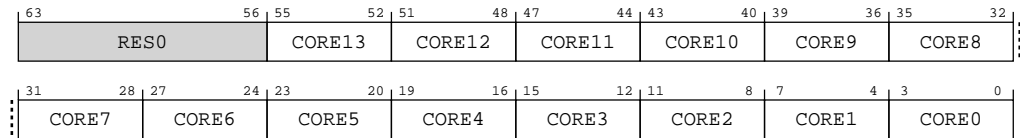


Where the reset reads xxxx, see individual bits.

## Bit descriptions

Indicates the current Split/Lock status for the core.

**Figure B-126: ext\_clusterae\_coreslstat bit assignments**



**Table B-196: CLUSTERAE\_CORESLSTAT bit descriptions**

Bits	Name	Description	Reset
[63:56]	RES0	Reserved	RES0
[55:52]	CORE13	Indicates the current Split/Lock status for core N.  <b>0b0000</b> Core not present or powered off.  <b>0b0001</b> Core powered in LOCKED mode as the primary core.  <b>0b0011</b> Core powered in LOCKED mode as the redundant core.  <b>0b0100</b> Core powered in SPLIT mode	xxxx
[51:48]	CORE12	Indicates the current Split/Lock status for core N.  <b>0b0000</b> Core not present or powered off.  <b>0b0001</b> Core powered in LOCKED mode as the primary core.  <b>0b0011</b> Core powered in LOCKED mode as the redundant core.  <b>0b0100</b> Core powered in SPLIT mode	xxxx



Bits	Name	Description	Reset
[47:44]	CORE11	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[43:40]	CORE10	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[39:36]	CORE9	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[35:32]	CORE8	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx

Bits	Name	Description	Reset
[31:28]	CORE7	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[27:24]	CORE6	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[23:20]	CORE5	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[19:16]	CORE4	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx

Bits	Name	Description	Reset
[15:12]	CORE3	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[11:8]	CORE2	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[7:4]	CORE1	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx
[3:0]	CORE0	<p>Indicates the current Split/Lock status for core N.</p> <p><b>0b0000</b> Core not present or powered off.</p> <p><b>0b0001</b> Core powered in LOCKED mode as the primary core.</p> <p><b>0b0011</b> Core powered in LOCKED mode as the redundant core.</p> <p><b>0b0100</b> Core powered in SPLIT mode</p>	xxxx

## Accessibility

This interface is accessible as follows:

RO

B.1.6.7 CLUSTERAE\_CLUSTERRECOV, Cluster Recovery Control Register

Debug recovery and warm reset request register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAE

Register offset

0x0050

Access type

RW

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0000	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-127: ext\_clusterae\_clusterrecov bit assignments

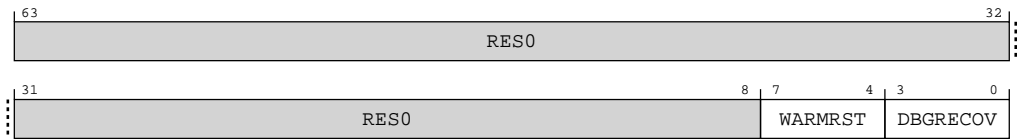


Table B-197: CLUSTERAE\_CLUSTERRECOV bit descriptions

Bits	Name	Description	Reset
[63:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	WARMRST	<p>This mode is used for resetting during debugging, after an unrecoverable RAS error, or after a watchdog timeout.</p> <p><b>0b0000</b> When read, returns the current status of Cores and Cluster in normal operation. When written requests a return to normal operation (or remain if already operating normally)</p> <p><b>0b0001</b> When read, returns the current status of Cores and Cluster entering WARM_RST. When written requests the Cluster to enter WARM_RST. Once the whole Cluster has entered, it automatically returns to normal operation.</p> <p><b>0b0011</b> When read, returns the current status of Cores and Cluster entering WARM_RST, ready to remain there. When written requests the Cluster to enter WARM_RST and keep the Cores and Cluster held in reset.</p> <p><b>0b0100</b> When read, returns the current status of Cores and Cluster leaving WARM_RST. Writes of this value are unsupported.</p> <p><b>0b0111</b> When read, returns the current status of Cores and Cluster all in WARM_RST (or OFF/OFF_EMU). Writes of this value are unsupported.</p>	0b0000
[3:0]	DBGRECOV	<p>This mode is used for recovering state for use in debugging, after a reset which could typically be from some kind of watchdog timeout. It is similar to WARM_RST, but additionally preserves the RAM contents.</p> <p><b>0b0000</b> When read, returns the current status of Cores and Cluster in normal operation. When written requests a return to normal operation (or remain if already operating normally)</p> <p><b>0b0001</b> When read, returns the current status of Cores and Cluster entering DBG_RECOV. When written requests the Cluster to enter DBG_RECOV. Once the whole Cluster has entered, it automatically returns to normal operation.</p> <p><b>0b0011</b> When read, returns the current status of Cores and Cluster entering DBG_RECOV, ready to remain there. When written requests the Cluster to enter DBG_RECOV and keep the Cores and Cluster held in reset.</p> <p><b>0b0100</b> When read, returns the current status of Cores and Cluster leaving DBG_RECOV. Writes of this value are unsupported.</p> <p><b>0b0111</b> When read, returns the current status of Cores and Cluster all in DBG_RECOV (or OFF/OFF_EMU). Writes of this value are unsupported.</p>	0b0000

## Accessibility

This interface is accessible as follows:

RW

B.1.6.8 CLUSTERAE\_CLUSTERWRITEKEY, Cluster Write Key Register

Control register for setting the Cluster KEY to enable writing of AE and PPU registers.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAE

Register offset

0x0060

Access type

RESERVEDW

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	0000	0000
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-128: ext\_clusterae\_clusterwritekey bit assignments

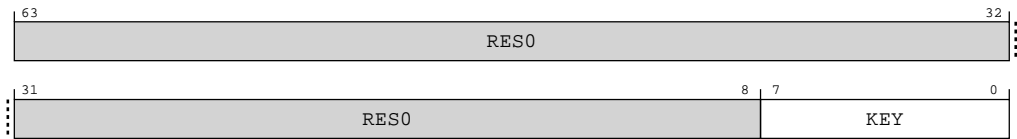


Table B-198: CLUSTERAE\_CLUSTERWRITEKEY bit descriptions

Bits	Name	Description	Reset
[63:8]	RES0	Reserved	RES0
[7:0]	KEY	Write protection key. The value 8'hBA must be written to the field to enable subsequent write to other registers in this page or any PPU registers.	0x00

Accessibility

This interface is accessible as follows:

WO

## B.1.7 External core PPU registers summary

Each core *Power Policy Unit* (PPU) in the DSU-120AE 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.

The summary table provides an overview of all the PPU registers for a single core in the DSU-120AE. For more information about a register, click on the register name in the table.



Note

- You must access the cluster system control registers from the Secure state.
- The core PPU registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- These register descriptions are configuration of the PPU architecture, see [Arm® Power Policy Unit Architecture Specification](#) for more details.
- 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 base address for the core PPU registers is  $0x\langle n \rangle 80000$ , 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-199: Core PPU register summary**

Offset	Name	Reset	Width	Description
0x000	<a href="#">PPU_PWPR</a>	See individual bit resets.	32-bit	Power Policy Register
0x004	<a href="#">PPU_PMER</a>	See individual bit resets.	32-bit	Power Mode Emulation Enable Register
0x008	<a href="#">PPU_PWSR</a>	See individual bit resets.	32-bit	Power Status Register
0x010	<a href="#">PPU_DISR</a>	See individual bit resets.	32-bit	Device Interface Input Current Status Register
0x014	<a href="#">PPU_MISR</a>	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register
0x018	<a href="#">PPU_STSR</a>	See individual bit resets.	32-bit	Stored Status Register
0x01C	<a href="#">PPU_UNLK</a>	See individual bit resets.	32-bit	Unlock Register
0x020	<a href="#">PPU_PWCR</a>	See individual bit resets.	32-bit	Power Configuration Register
0x024	<a href="#">PPU_PTCR</a>	See individual bit resets.	32-bit	Power Mode Transition Register
0x030	<a href="#">PPU_IMR</a>	See individual bit resets.	32-bit	Interrupt Mask Register
0x034	<a href="#">PPU_AIMR</a>	See individual bit resets.	32-bit	Additional Interrupt Mask Register
0x038	<a href="#">PPU_ISR</a>	See individual bit resets.	32-bit	Interrupt Status Register
0x03C	<a href="#">PPU_AISR</a>	See individual bit resets.	32-bit	Additional Interrupt Status Register

Offset	Name	Reset	Width	Description
0x040	PPU_IESR	See individual bit resets.	32-bit	Input Edge Sensitivity Register
0x044	PPU_OPSR	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register
0x050	PPU_FUNRR	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register
0x054	PPU_FULRR	See individual bit resets.	32-bit	Full Retention RAM Configuration Register
0x058	PPU_MEMRR	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register
0x170	PPU_DCDR0	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0
0x174	PPU_DCDR1	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1
0xFB0	PPU_IDR0	See individual bit resets.	32-bit	PPU Identification Register 0
0xFB4	PPU_IDR1	See individual bit resets.	32-bit	PPU Identification Register 1
0xFC8	PPU_IIDR	See individual bit resets.	32-bit	Implementation Identification Register
0xFCC	PPU_AIDR	See individual bit resets.	32-bit	Architecture Identification Register
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1
0xFE8	PPU_PIDR2	See individual bit resets.	32-bit	PPU Peripheral Identification Register 2
0xFEC	PPU_PIDR3	See individual bit resets.	32-bit	PPU Peripheral Identification Register 3
0xFF0	PPU_CIDR0	See individual bit resets.	32-bit	PPU Component Identification Register 0
0xFF4	PPU_CIDR1	See individual bit resets.	32-bit	PPU Component Identification Register 1
0xFF8	PPU_CIDR2	See individual bit resets.	32-bit	PPU Component Identification Register 2
0xFFC	PPU_CIDR3	See individual bit resets.	32-bit	PPU Component Identification Register 3

### B.1.7.1 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

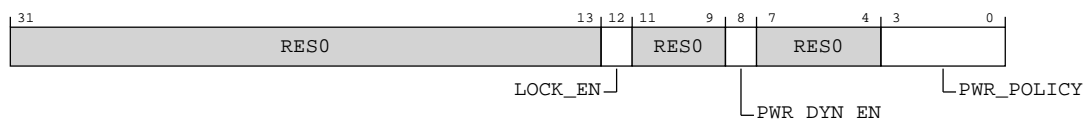
xxxx	xxxx	xxxx	xxxx	xxx0	xxx0	xxxx	0000
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

### Figure B-129: ext\_ppu\_pwpr bit assignments



### Table B-200: PPU\_PWPR bit descriptions

Bits	Name	Description	Reset
[31:13]	RES0	Reserved	RES0
[12]	LOCK_EN	<p>Lock enable bit for OFF and OFF_EMU power modes</p> <p><b>0b0</b> Lock feature disabled.</p> <p><b>0b1</b> Lock feature enabled.</p>	0b0
[11:9]	RES0	Reserved	RES0
[8]	PWR_DYN_EN	<p>Power mode dynamic transition enable.</p> <p><b>0b0</b> Dynamic transitions disabled for power modes.</p> <p><b>0b1</b> Dynamic transitions enabled for power modes, allowing transitions to be initiated by changes on power mode DEACTIVE inputs.</p>	0b0
[7:4]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[3:0]	PWR_POLICY	<p>Power mode policy.</p> <p>When static power mode transitions are enabled, PWR_DYN_EN is set to 0b0, this is the target power mode for the PPU.</p> <p>When dynamic power mode transitions are enabled, PWR_DYN_EN is set to 0b1, this is the minimum power mode for the PPU.</p> <p>All other values are reserved.</p> <p><b>0b0000</b> OFF. Logic off and RAM off.</p> <p><b>0b0001</b> OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power.</p> <p><b>0b0101</b> FULL_RET. Full Retention. Logic and RAM in retention.</p> <p><b>0b0111</b> FUNC_RET. Functional Retention. Floating-point/Vector logic retained, rest of the core logic and RAM on, core is functional.</p> <p><b>0b1000</b> ON. Logic on with RAM on, core is functional.</p> <p><b>0b1001</b> WARM_RST. Warm Reset. Warm reset application with logic and RAM on.</p> <p><b>0b1010</b> DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.</p>	0b0000

## Accessibility

This interface is accessible as follows:

RW

## B.1.7.2 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

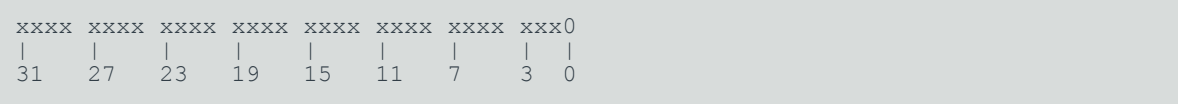
Register offset

0x004

Access type

RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-130: ext\_ppu\_pmer bit assignments

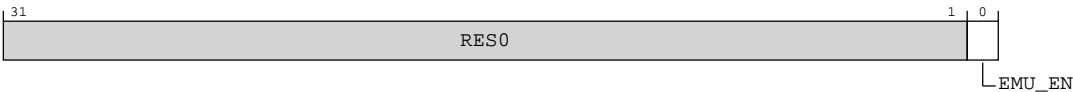


Table B-201: PPU\_PMER bit descriptions

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

Accessibility

This interface is accessible as follows:

RW

B.1.7.3 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-131: ext\_ppu\_pwsr bit assignments

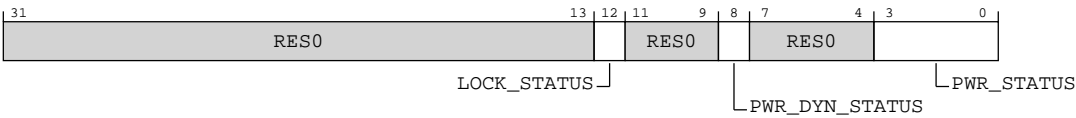


Table B-202: PPU\_PWSR bit descriptions

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	<p>Power mode dynamic transition status.</p> <p>There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when PPU_PWPR.DYN_EN is programmed.</p> <p><b>0b0</b> Dynamic transitions disabled for power modes.</p> <p><b>0b1</b> Dynamic transitions enabled for power modes.</p>	0b0
[7:4]	RES0	Reserved	RES0
[3:0]	PWR_STATUS	<p>Power mode status.</p> <p>These bits reflect the current power mode of the PPU.</p> <p>All other values are reserved.</p> <p><b>0b0000</b> OFF. Logic off and RAM off.</p> <p><b>0b0001</b> OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power.</p> <p><b>0b0101</b> FULL_RET. Full Retention. Logic and RAM in retention.</p> <p><b>0b0111</b> FUNC_RET. Functional Retention. Floating-point/Vector logic retained, rest of the core logic and RAM on, core is functional.</p> <p><b>0b1000</b> ON. Logic on with RAM on, core is functional.</p> <p><b>0b1001</b> WARM_RST. Warm Reset. Warm reset application with logic and RAM on.</p> <p><b>0b1010</b> DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.</p>	0b0000

## Accessibility

This interface is accessible as follows:

RO

### B.1.7.4 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

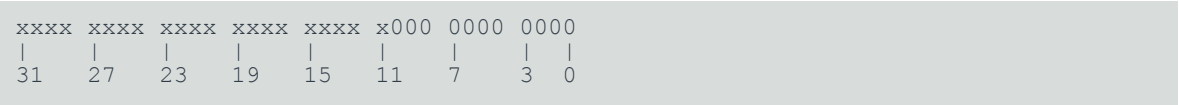
Register offset

0x010

Access type

RO

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-132: ext\_ppu\_disr bit assignments



Table B-203: PPU\_DISR bit descriptions

Bits	Name	Description	Reset
[31:11]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[10:0]	PWR_DEVACTIVE_STATUS	Status of the power mode DEVPACTIVE inputs.  <b>0b000000000000</b> Minimum mode OFF.  <b>0000000001x</b> Minimum mode OFF_EMU.  <b>000001xxxxx</b> Minimum mode FULL_RET.  <b>0001xxxxxxx</b> Minimum mode FUNC_RET.  <b>001xxxxxxxx</b> Minimum mode ON.  <b>01xxxxxxxxx</b> Minimum mode WARM_RST.  <b>1xxxxxxxxxx</b> Minimum mode DBG_RECOV.	0b000000000000

Accessibility

This interface is accessible as follows:

RO

B.1.7.5 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-133: ext\_ppu\_misr bit assignments

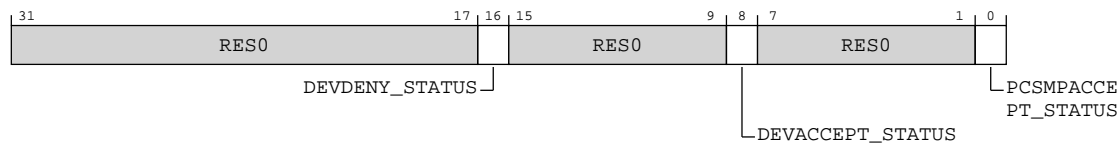


Table B-204: PPU\_MISR bit descriptions

Bits	Name	Description	Reset
[31:17]	RES0	Reserved	RES0
[16]	DEVDPDENY_STATUS	Status of the device interface DEVDPDENY inputs.  0b0 DEVDPDENY deasserted.  0b1 DEVDPDENY 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]	PCSMACCEPT_STATUS	Status of the PCSMAPCEPT inputs.  0b0 PCSMACCEPT deasserted.  0b1 PCSMACCEPT asserted.	0b0

Accessibility

This interface is accessible as follows:

RO



B.1.7.6 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-134: ext\_ppu\_stsr bit assignments

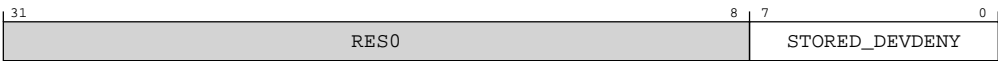


Table B-205: PPU\_STSR bit descriptions

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 }

Accessibility

This interface is accessible as follows:

RO

B.1.7.7 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-135: ext\_ppu\_unlk bit assignments

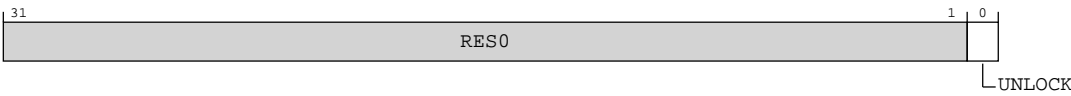


Table B-206: PPU\_UNLK bit descriptions

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

Accessibility

This interface is accessible as follows:

RW

B.1.7.8 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

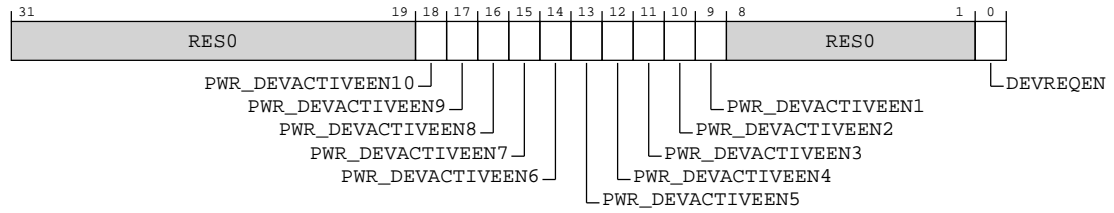
xxxx	xxxx	xxxx	x111	1111	111x	xxxx	xxx1
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-136: ext\_ppu\_pwcr bit assignments**



**Table B-207: PPU\_PWCR bit descriptions**

Bits	Name	Description	Reset
[31:19]	RES0	Reserved	RES0
[18]	PWR_DEVACTIVEEN10	Enables the operating mode DEVPACTIVE[10] input. <b>0b0</b> DEVPACTIVE[10] input (DBG_RECOV) disabled. <b>0b1</b> DEVPACTIVE[10] input (DBG_RECOV) enabled.	0b1
[17]	PWR_DEVACTIVEEN9	Enables the operating mode DEVPACTIVE[9] input. <b>0b0</b> DEVPACTIVE[9] input (WARM_RST) disabled. <b>0b1</b> DEVPACTIVE[9] input (WARM_RST) enabled.	0b1
[16]	PWR_DEVACTIVEEN8	Enables the operating mode DEVPACTIVE[8] input. <b>0b0</b> DEVPACTIVE[8] input (ON) disabled. <b>0b1</b> DEVPACTIVE[8] input (ON) enabled.	0b1
[15]	PWR_DEVACTIVEEN7	Enables the operating mode DEVPACTIVE[7] input. <b>0b0</b> DEVPACTIVE[7] input (FUNC_RET) disabled. <b>0b1</b> DEVPACTIVE[7] input (FUNC_RET) enabled.	0b1
[14]	PWR_DEVACTIVEEN6	Enables the operating mode DEVPACTIVE[6] input. <b>0b1</b> DEVPACTIVE[6] input (MEM_OFF) enabled.	0b1
[13]	PWR_DEVACTIVEEN5	Enables the operating mode DEVPACTIVE[5] input. <b>0b0</b> DEVPACTIVE[5] input (FULL_RET) disabled. <b>0b1</b> DEVPACTIVE[5] input (FULL_RET) enabled.	0b1

Bits	Name	Description	Reset
[12]	PWR_DEVACTIVEEN4	Enables the operating mode DEVPACTIVE[4] input. <b>0b1</b> DEVPACTIVE[4] input (LOGIC_RET) enabled.	0b1
[11]	PWR_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[3] input. <b>0b1</b> DEVPACTIVE[3] input (MEM_RET_EMU) enabled.	0b1
[10]	PWR_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[2] input. <b>0b1</b> DEVPACTIVE[2] input (MEM_RET) enabled.	0b1
[9]	PWR_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[1] input. <b>0b0</b> DEVPACTIVE[1] input (OFF_EMU) disabled. <b>0b1</b> DEVPACTIVE[1] input (OFF_EMU) enabled.	0b1
[8:1]	RES0	Reserved	RES0
[0]	DEVREQEN	Device interface handshake enable. <b>0b0</b> Device interface handshake disabled for transitions. <b>0b1</b> Device interface handshake enabled for transitions.	0b1

## Accessibility

This interface is accessible as follows:

RW

### B.1.7.9 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

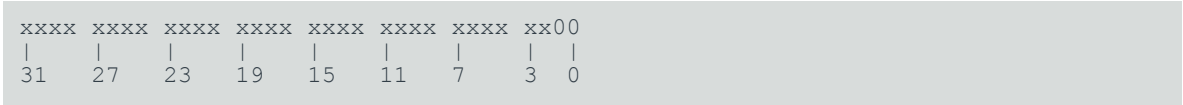
PPU

### Register offset

0x024

Access type  
RW

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-137: ext\_ppu\_ptcr bit assignments

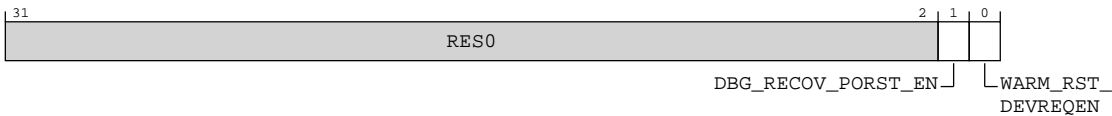


Table B-208: PPU\_PTCR bit descriptions

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 <b>UNPREDICTABLE</b> .  <b>0b0</b> DEVPORESETn is not asserted when in DBG_RECOV.  <b>0b1</b> 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 <b>UNPREDICTABLE</b> .  <b>0b0</b> The PPU does not perform a device interface handshake when transitioning between ON and WARM_RST.  <b>0b1</b> The PPU performs a device interface handshake when transitioning between ON and WARM_RST.	0b0

Accessibility

This interface is accessible as follows:

RW

B.1.7.10 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

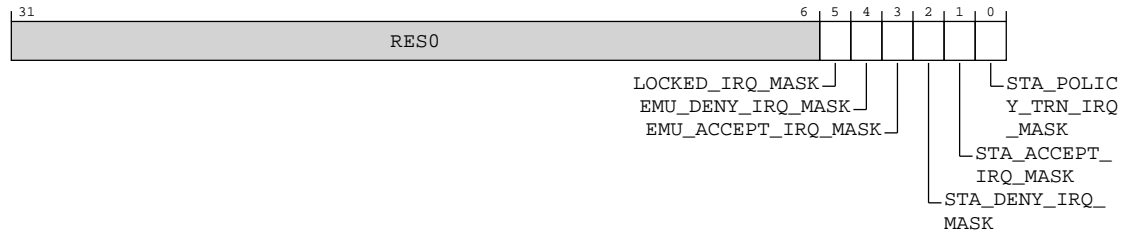
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xx11	1010
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-138: ext\_ppu\_imr bit assignments**



**Table B-209: PPU\_IMR bit descriptions**

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5]	LOCKED_IRQ_MASK	<p>Locked event mask</p> <p><b>0b0</b> Locked event enabled.</p> <p><b>0b1</b> Locked event masked.</p>	0b1
[4]	EMU_DENY_IRQ_MASK	<p>Emulation transition denial event mask</p> <p><b>0b0</b> Emulation transition denial event enabled.</p> <p><b>0b1</b> Emulation transition denial event masked.</p>	0b1
[3]	EMU_ACCEPT_IRQ_MASK	<p>Emulation transition acceptance event mask</p> <p><b>0b0</b> Emulation transition acceptance event enabled.</p> <p><b>0b1</b> Emulation transition acceptance event masked.</p>	0b1
[2]	STA_DENY_IRQ_MASK	<p>Static transition denial event mask</p> <p><b>0b0</b> Static transition denial event enabled.</p> <p><b>0b1</b> Static transition denial event masked.</p>	0b0
[1]	STA_ACCEPT_IRQ_MASK	<p>Static transition acceptance event mask</p> <p><b>0b0</b> Static transition acceptance event enabled.</p> <p><b>0b1</b> Static transition acceptance event masked.</p>	0b1



Bits	Name	Description	Reset
[0]	STA_POLICY_TRN_IRQ_MASK	Static full policy transition completion event mask  <b>0b0</b> Static full policy transition completion event enabled.  <b>0b1</b> Static full policy transition completion event masked.	0b0

Accessibility

This interface is accessible as follows:

RW

B.1.7.11 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	xxxx	x110
31	27	23	19	15	11	7	3	0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-139: ext\_ppu\_aimr bit assignments

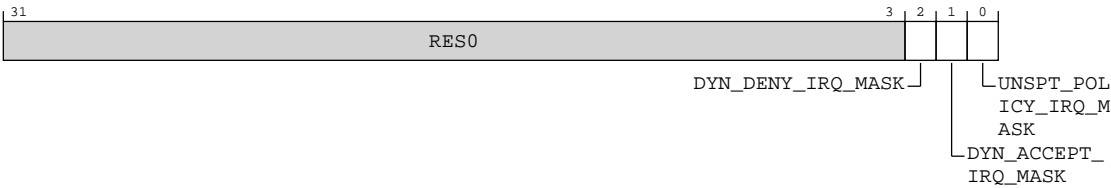


Table B-210: PPU\_AIMR bit descriptions

Bits	Name	Description	Reset
[31:3]	RES0	Reserved	RES0
[2]	DYN_DENY_IRQ_MASK	Dynamic transition denial event mask  <b>0b0</b> Dynamic transition denial event enabled.  <b>0b1</b> Dynamic transition denial event masked.	0b1
[1]	DYN_ACCEPT_IRQ_MASK	Dynamic transition acceptance event mask  <b>0b0</b> Dynamic transition acceptance event enabled.  <b>0b1</b> Dynamic transition acceptance event masked.	0b1
[0]	UNSPT_POLICY_IRQ_MASK	Unsupported policy event mask  <b>0b0</b> Unsupported policy event enabled.  <b>0b1</b> Unsupported policy event masked.	0b0

Accessibility

This interface is accessible as follows:

RW

B.1.7.12 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

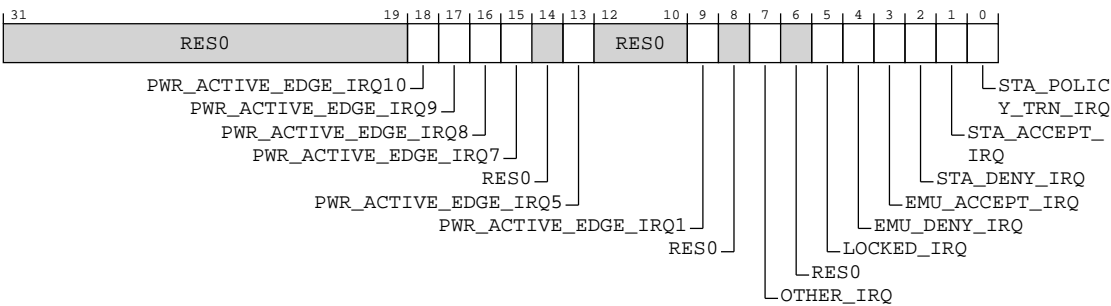
xxxx	xxxx	xxxx	x000	0x0x	xx0x	0x00	0000
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-140: ext\_ppu\_isr bit assignments



**Table B-211: 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.  <b>0b0</b> DEVPACTIVE[10] input (DBG_RECOV) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[9] input (WARM_RST) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[8] input (ON) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[7] input (FUNC_RET) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[5] input (FULL_RET) did not assert the interrupt output.  <b>0b1</b> 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.  <b>0b0</b> DEVPACTIVE[1] input (OFF_EMU) did not assert the interrupt output.  <b>0b1</b> 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).  <b>0b0</b> No interrupt pending in PPU_AISR.  <b>0b1</b> Interrupt pending in PPU_AISR.	0b0
[6]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[5]	LOCKED_IRQ	<p>Locked event status.</p> <p><b>0b0</b> No locked event.</p> <p><b>0b1</b> A locked event asserted the interrupt output.</p>	0b0
[4]	EMU_DENY_IRQ	<p>Emulated transition denial event status.</p> <p><b>0b0</b> No emulated transition denial event.</p> <p><b>0b1</b> An emulated transition denial event asserted the interrupt output.</p>	0b0
[3]	EMU_ACCEPT_IRQ	<p>Emulated transition acceptance event status.</p> <p><b>0b0</b> No emulated transition acceptance event.</p> <p><b>0b1</b> An emulated transition acceptance event asserted the interrupt output.</p>	0b0
[2]	STA_DENY_IRQ	<p>Static transition denial event status.</p> <p><b>0b0</b> No static transition denial event.</p> <p><b>0b1</b> A static transition denial event asserted the interrupt output.</p>	0b0
[1]	STA_ACCEPT_IRQ	<p>Static transition acceptance event status.</p> <p><b>0b0</b> No static transition acceptance event.</p> <p><b>0b1</b> A static transition acceptance event asserted the interrupt output.</p>	0b0
[0]	STA_POLICY_TRN_IRQ	<p>Static full policy transition completion event status.</p> <p><b>0b0</b> No static full policy transition completion event.</p> <p><b>0b1</b> A static full policy transition completion event asserted the interrupt output.</p>	0b0

## Accessibility

This interface is accessible as follows:

RW

B.1.7.13 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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-141: ext\_ppu\_aisr bit assignments

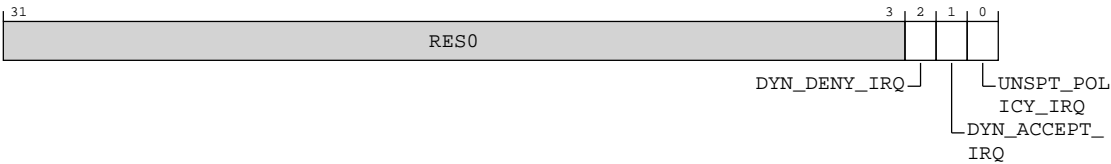


Table B-212: PPU\_AISR bit descriptions

Bits	Name	Description	Reset
[31:3]	RES0	Reserved	RES0
[2]	DYN_DENY_IRQ	Dynamic transition denial event status  <b>0b0</b> No dynamic transition denial event.  <b>0b1</b> A dynamic transition denial event asserted the interrupt output.	0b0
[1]	DYN_ACCEPT_IRQ	Dynamic transition acceptance event status  <b>0b0</b> No dynamic transition acceptance event.  <b>0b1</b> A dynamic transition acceptance event asserted the interrupt output.	0b0
[0]	UNSPPT_POLICY_IRQ	Unsupported policy event status  <b>0b0</b> No unsupported policy event.  <b>0b1</b> An unsupported policy event asserted the interrupt output.	0b0

Accessibility

This interface is accessible as follows:

RW

B.1.7.14 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



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-142: ext\_ppu\_iesr bit assignments

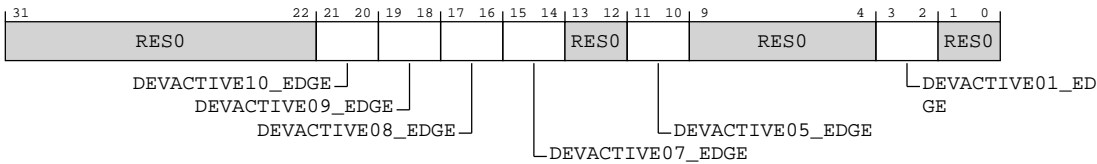


Table B-213: PPU\_IESR bit descriptions

Bits	Name	Description	Reset
[31:22]	RES0	Reserved	RES0
[21:20]	DEVPACTIVE10_EDGE	Configures the transitions on the DEVPACTIVE[10] input (DBG_RECOV) that generate an Input Edge interrupt event.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> Both edges of event generate an interrupt.	0b00



Bits	Name	Description	Reset
[19:18]	DEVACTIVE09_EDGE	Configures the transitions on the DEVPACTIVE[9] input (WARM_RST) that generate an Input Edge interrupt event.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> 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.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> Both edges of event generate an interrupt.	0b00
[15:14]	DEVACTIVE07_EDGE	Configures the transitions on the DEVPACTIVE[7] input (ON) that generate an Input Edge interrupt event.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> Both edges of event generate an interrupt.	0b00
[13:12]	RES0	Reserved	RES0
[11:10]	DEVACTIVE05_EDGE	Configures the transitions on the DEVPACTIVE[5] input (ON) that generate an Input Edge interrupt event.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> Both edges of event generate an interrupt.	0b00
[9:4]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[3:2]	DEVACTIVE01_EDGE	Configures the transitions on the DEVPACTIVE[1] input (OFF_EMU) that generate an Input Edge interrupt event.  <b>0b00</b> Event masked.  <b>0b01</b> Rising edge of event generates an interrupt.  <b>0b10</b> Falling edge of event generates an interrupt.  <b>0b11</b> Both edges of event generate an interrupt.	0b00
[1:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.1.7.15 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

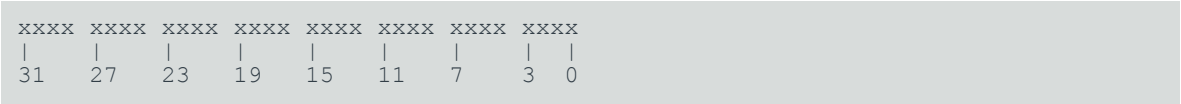
Register offset

0x044

Access type

RW

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-143: ext\_ppu\_opsr bit assignments

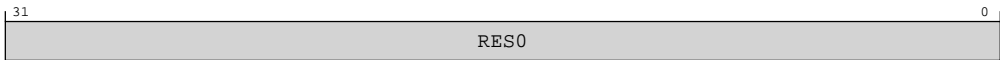


Table B-214: PPU\_OPSPR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.1.7.16 PPU\_FUNRR, Functional Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

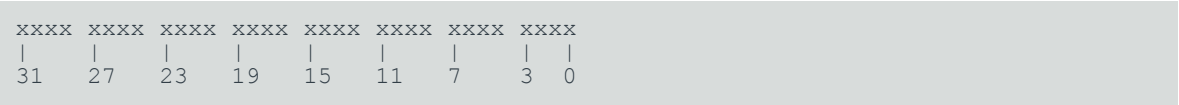
Register offset

0x050

Access type

RW

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-144: ext\_ppu\_funrr bit assignments

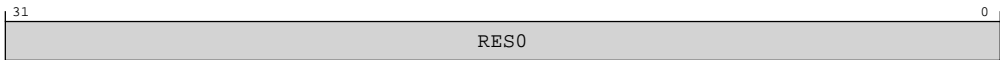


Table B-215: PPU\_FUNRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.1.7.17 PPU\_FULRR, Full Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

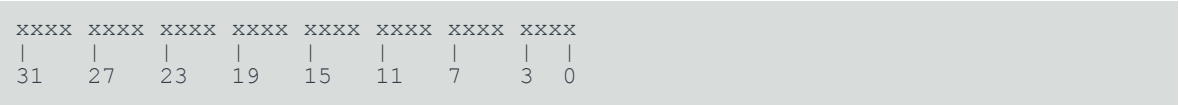
Register offset

0x054

Access type

RW

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-145: ext\_ppu\_fulrr bit assignments

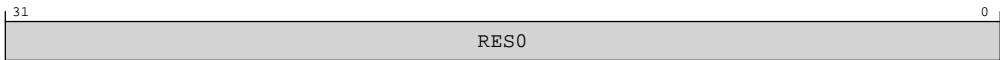


Table B-216: PPU\_FULRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.1.7.18 PPU\_MEMRR, Memory Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

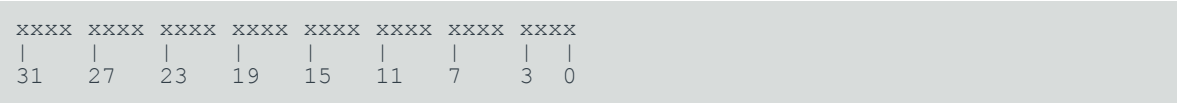
Register offset

0x058

Access type

RW

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-146: ext\_ppu\_memrr bit assignments

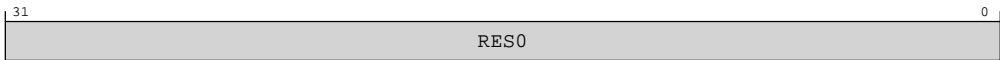


Table B-217: PPU\_MEMRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.1.7.19 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

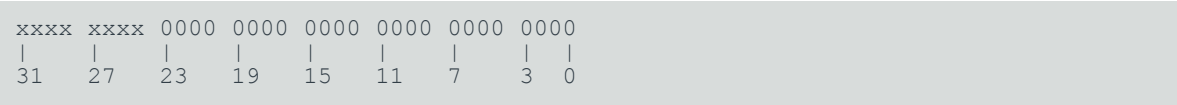
Register offset

0x170

Access type

RW

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-147: ext\_ppu\_dcdR0 bit assignments

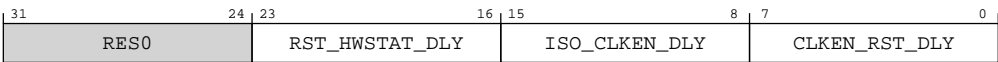


Table B-218: PPU\_DCDR0 bit descriptions

Bits	Name	Description	Reset
[31:24]	RES0	Reserved	RES0
[23:16]	RST_HWSTAT_DLY	Delay in PPUCLK clock cycles from reset de-assertion to HWSTAT update.Delay calculated as RST_HWSTAT_DLY + 1. Valid values for the field are in the range 0-255.	0x00
[15:8]	ISO_CLKEN_DLY	Delay in PPUCLK clock cycles from isolation enable de-assertion to clock enable assertion.Delay calculated as ISO_CLKEN_DLY + 1. Valid values for the field are in the range 0-255.	0x00
[7:0]	CLKEN_RST_DLY	Delay in PPUCLK clock cycles from clock enable assertion to reset de-assertion.Delay calculated as CLKEN_RST_DLY + 1. Valid values for the field are in the range 0-255.	0x00

Accessibility

This interface is accessible as follows:

RW

B.1.7.20 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-148: ext\_ppu\_dcdr1 bit assignments

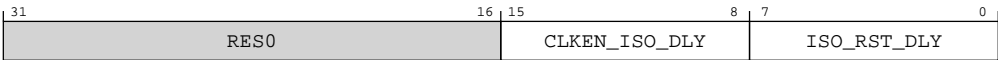


Table B-219: PPU\_DCDR1 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0
[15:8]	CLKEN_ISO_DLY	Delay in PPUCCLK clock cycles from clock enable de-assertion to isolation enable assertion.Delay calculated as CLKEN_ISO_DLY + 1. Valid values for the field are in the range 0-255.	0x00
[7:0]	ISO_RST_DLY	Delay in PPUCCLK clock cycles from isolation enable assertion to reset assertion.Delay calculated as ISO_RST_DLY + 1. Valid values for the field are in the range 0-255.	0x00

Accessibility

This interface is accessible as follows:

RW

B.1.7.21 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

x111	1100	0011	x111	1100	0011	0000	0000
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-149: ext\_ppu\_idr0 bit assignments

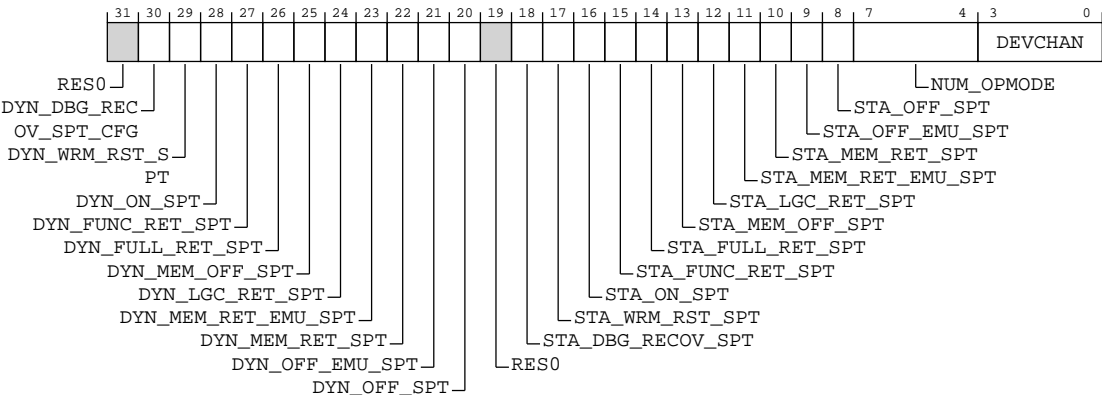


Table B-220: PPU\_IDR0 bit descriptions

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30]	DYN_DBG_RECOV_SPT_CFG	Dynamic DBG_RECOV support.  0b1 Dynamic DBG_RECOV supported.	0b1
[29]	DYN_WRM_RST_SPT	Dynamic WARM_RST support.  0b1 Dynamic WARM_RST supported.	0b1

Bits	Name	Description	Reset
[28]	DYN_ON_SPT	Dynamic ON support. <b>0b1</b> Dynamic ON supported.	0b1
[27]	DYN_FUNC_RET_SPT	Dynamic DYN_FUNC_RET_SPT support. <b>0b1</b> Dynamic DYN_FUNC_RET_SPT supported.	0b1
[26]	DYN_FULL_RET_SPT	Dynamic DYN_FULL_RET_SPT support. <b>0b1</b> Dynamic DYN_FULL_RET_SPT supported.	0b1
[25]	DYN_MEM_OFF_SPT	Dynamic MEM_OFF support. <b>0b0</b> Dynamic MEM_OFF not supported.	0b0
[24]	DYN_LGC_RET_SPT	Dynamic LOGIC_RET support. <b>0b0</b> Dynamic LOGIC_RET not supported.	0b0
[23]	DYN_MEM_RET_EMU_SPT	Dynamic DYN_MEM_RET_EMU_SPT support. <b>0b0</b> Dynamic DYN_MEM_RET_EMU_SPT not supported.	0b0
[22]	DYN_MEM_RET_SPT	Dynamic DYN_MEM_RET_SPT support. <b>0b0</b> Dynamic DYN_MEM_RET_SPT not supported.	0b0
[21]	DYN_OFF_EMU_SPT	Dynamic OFF_EMU support. <b>0b1</b> Dynamic OFF_EMU supported.	0b1
[20]	DYN_OFF_SPT	Dynamic OFF support. <b>0b1</b> Dynamic OFF supported.	0b1
[19]	RES0	Reserved	RES0
[18]	STA_DBG_RECOV_SPT	DBG_RECOV support. <b>0b1</b> DBG_RECOV supported.	0b1
[17]	STA_WRM_RST_SPT	WARM_RST support. <b>0b1</b> WRM_RST supported.	0b1
[16]	STA_ON_SPT	ON support. <b>0b1</b> ON supported.	0b1
[15]	STA_FUNC_RET_SPT	FUNC_RET support. <b>0b1</b> FUNC_RET supported.	0b1

Bits	Name	Description	Reset
[14]	STA_FULL_RET_SPT	FULL_RET support. <b>0b1</b> FULL_RET supported.	0b1
[13]	STA_MEM_OFF_SPT	MEM_OFF support. <b>0b0</b> MEM_OFF not supported.	0b0
[12]	STA_LGC_RET_SPT	LOGIC_RET support. <b>0b0</b> LOGIC_RET not supported.	0b0
[11]	STA_MEM_RET_EMU_SPT	MEM_RET_EMU support. <b>0b0</b> MEM_RET_EMU not supported.	0b0
[10]	STA_MEM_RET_SPT	MEM_RET support. <b>0b0</b> MEM_RET not supported.	0b0
[9]	STA_OFF_EMU_SPT	OFF_EMU support. <b>0b1</b> OFF_EMU supported.	0b1
[8]	STA_OFF_SPT	OFF support. <b>0b1</b> OFF supported.	0b1
[7:4]	NUM_OPMODE	No. of operating modes supported, minus 1. <b>0b0000</b> 1 operating mode supported.	0b0000
[3:0]	DEVCHAN	No. of Device Interface Channels. <b>0b0000</b> 0 (P-channel PPU).	0b0000

## Accessibility

This interface is accessible as follows:

RO

### B.1.7.22 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
31	27	23	19	15	11	7	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-150: ext\_ppu\_idr1 bit assignments

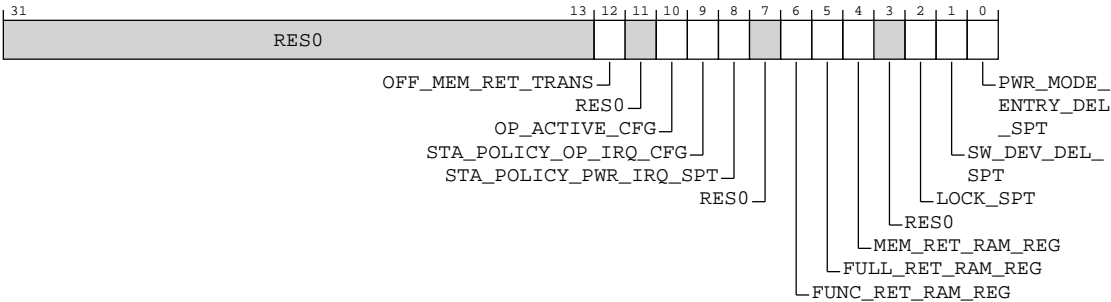


Table B-221: PPU\_IDR1 bit descriptions

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

Bits	Name	Description	Reset
[11]	RES0	Reserved	RES0
[10]	OP_ACTIVE_CFG	Operating mode use model for dynamic transitions. <b>0b0</b> Ladder use model.	0b0
[9]	STA_POLICY_OP_IRQ_CFG	Operating policy transition completion event status. <b>0b0</b> Operating policy transition completion events not supported.	0b0
[8]	STA_POLICY_PWR_IRQ_SPT	Power policy transition completion event status. <b>0b0</b> 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. <b>0b0</b> PPU_FUNRR is reserved.	0b0
[5]	FULL_RET_RAM_REG	Indicates if the PPU_FULRR register is present or reserved. <b>0b0</b> PPU_FULRR is reserved.	0b0
[4]	MEM_RET_RAM_REG	Indicates if the PPU_MEMRR register is present or reserved. <b>0b0</b> PPU_MEMRR is reserved.	0b0
[3]	RES0	Reserved	RES0
[2]	LOCK_SPT	Indicates if the lock and the lock interrupt event are supported. <b>0b1</b> Lock and the lock interrupt event are supported.	0b1
[1]	SW_DEV_DEL_SPT	Software device delay control configuration support. <b>0b1</b> Software device delay control configuration supported.	0b1
[0]	PWR_MODE_ENTRY_DEL_SPT	Power mode entry delay support. <b>0b0</b> Power mode entry delay not supported.	0b0

## Accessibility

This interface is accessible as follows:

RO

### B.1.7.23 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 0010 0000 0100 0011 1011

Bit descriptions

Figure B-151: ext\_ppu\_iidr bit assignments

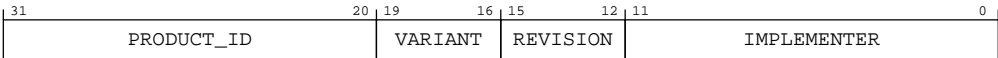


Table B-222: PPU\_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	PRODUCT_ID	Value identifying the PPU part.  <b>0b000010110110</b> Power Policy Unit.	0x0B6
[19:16]	VARIANT	Value used to distinguish PPU variants, or major revisions of the PPU.  <b>0b0000</b> PPU variant 0.  <b>0b0001</b> PPU variant 1.  <b>0b0010</b> PPU variant 2.  <b>0b0011</b> PPU variant 3.  <b>0b0100</b> PPU variant 4.	0b0010

Bits	Name	Description	Reset
[15:12]	REVISION	Value used to distinguish minor revisions of the PPU.  <b>0b0000</b> PPU revision 0.  <b>0b0001</b> PPU revision 1.  <b>0b0010</b> PPU revision 2.  <b>0b0011</b> PPU revision 3.  <b>0b0100</b> PPU revision 4.	0b0000
[11:0]	IMPLEMENTER	Implementer identification.  <b>0b010000111011</b> Arm Limited.	0x43B

Accessibility

This interface is accessible as follows:

RO

B.1.7.24 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





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-152: ext\_ppu\_aidr bit assignments

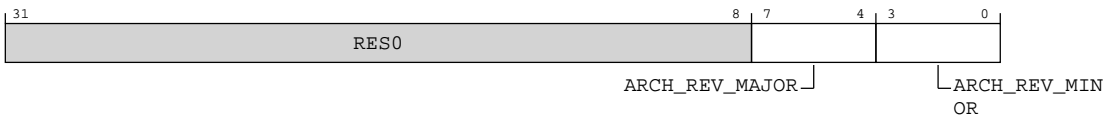


Table B-223: PPU\_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	ARCH_REV_MAJOR	PPU architecture major revision. <b>0b0001</b> PPU architecture major revision 1.	0b0001
[3:0]	ARCH_REV_MINOR	PPU architecture minor revision. <b>0b0010</b> PPU architecture minor revision 2.	0b0010

Accessibility

This interface is accessible as follows:

RO

B.1.7.25 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-153: ext\_ppu\_pidr4 bit assignments



Table B-224: 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

Accessibility

This interface is accessible as follows:

RO

B.1.7.26 PPU\_PIDR5, PPU Peripheral Identification Register 5

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

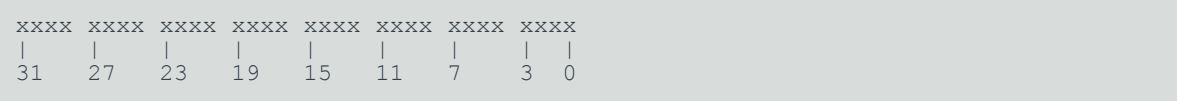
Register offset

0xFD4

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-154: ext\_ppu\_pidr5 bit assignments

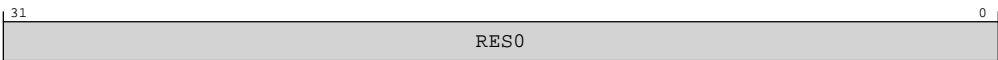


Table B-225: PPU\_PIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.1.7.27 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-155: ext\_ppu\_pidr6 bit assignments

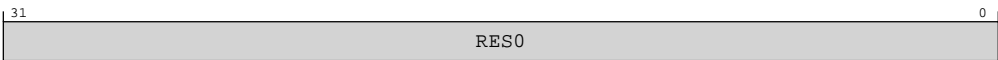


Table B-226: PPU\_PIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.1.7.28 PPU\_PIDR7, PPU Peripheral Identification Register 7

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

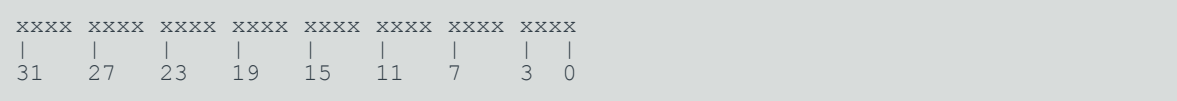
Register offset

0xFDC

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-156: ext\_ppu\_pidr7 bit assignments

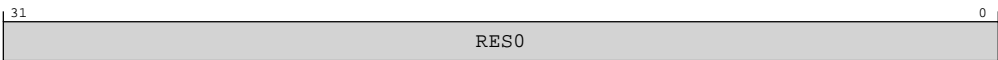


Table B-227: PPU\_PIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.1.7.29 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-157: ext\_ppu\_pidr0 bit assignments

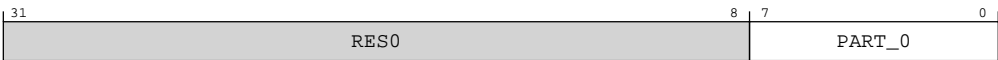


Table B-228: PPU\_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. <b>0b10110110</b> Core Power Policy Unit. Bits [7:0] of part number 0x0B6.	0xB6

Accessibility

This interface is accessible as follows:

RO

B.1.7.30 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-158: ext\_ppu\_pidr1 bit assignments



Table B-229: 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

Accessibility

This interface is accessible as follows:

RO

B.1.7.31 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-159: ext\_ppu\_pidr2 bit assignments



Table B-230: PPU\_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVISION	<p>Component major revision.</p> <p><b>0b0000</b> Component major revision 0.</p> <p><b>0b0001</b> Component major revision 1.</p> <p><b>0b0010</b> Component major revision 2.</p> <p><b>0b0011</b> Component major revision 3.</p> <p><b>0b0100</b> Component major revision 4.</p>	0b0010
[3]	JEDEC	<p>JEDEC assignee.</p> <p><b>0b1</b> JEDEC-assignee values is used.</p>	0b1
[2:0]	DES_1	<p>JEP106 identification code bits [6:4].</p> <p><b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.</p>	0b011

## Accessibility

This interface is accessible as follows:

RO

## B.1.7.32 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 0000 0000





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-160: ext\_ppu\_pidr3 bit assignments

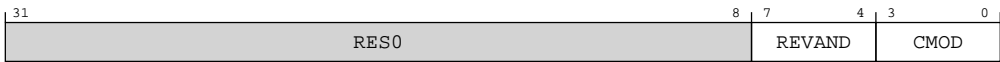


Table B-231: PPU\_PIDR3 bit descriptions

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.	0b0000
[3:0]	CMOD	Customer Modified.  0b0000 The component is not modified from the original design.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.7.33 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-161: ext\_ppu\_cidr0 bit assignments

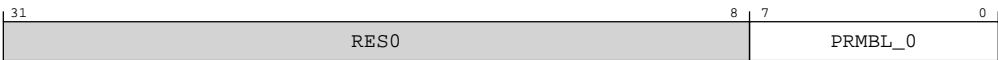


Table B-232: PPU\_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	CoreSight component identification preamble. <b>0b00001101</b> CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

B.1.7.34 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-162: ext\_ppu\_cidr1 bit assignments

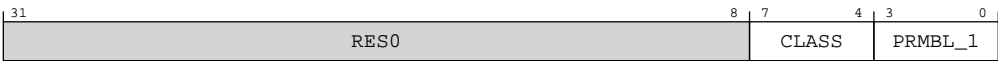


Table B-233: PPU\_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	CLASS	CoreSight component class. <b>0b1111</b> CoreLink component.	0b1111
[3:0]	PRMBL_1	CoreSight component identification preamble. <b>0b0000</b> CoreSight component identification preamble.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.7.35 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



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-163: ext\_ppu\_cidr2 bit assignments



Table B-234: PPU\_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:0]	PRMBL_2	CoreSight component identification preamble.  <b>0b000000101</b> CoreSight component identification preamble.	0x05

Accessibility

This interface is accessible as follows:

RO

B.1.7.36 PPU\_CIDR3, PPU Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

PPU

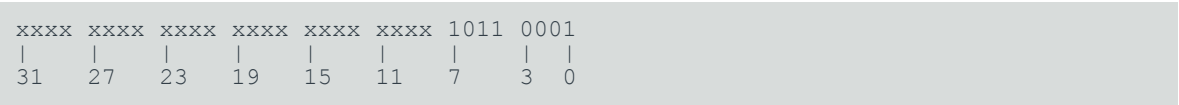
Register offset

0xFFC

Access type

RO

Reset value

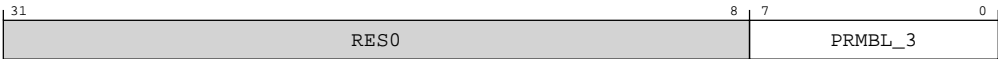


Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-164: ext\_ppu\_cidr3 bit assignments



**Table B-235: PPU\_CIDR3 bit descriptions**

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_3	CoreSight component identification preamble. <b>0b10110001</b> CoreSight component identification preamble.	0xB1

### Accessibility

This interface is accessible as follows:

RO

## B.2 Registers accessed over the Debug APB bus

This section contains the descriptions for all the external registers in the *DynamIQ™ Shared Unit-120AE* (DSU-120AE) accessed over the Debug APB bus.

### B.2.1 External cluster and core CTI registers summary

The cluster *Cross Trigger Interface* (CTI) registers and core CTI registers are only accessible using memory-mapped accesses over the Debug APB interface.

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**.
- 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-236: CTI registers summary**

Offset	Name	Reset	Width	Description	Identical core CTI
0x000	CTICONTROL	See individual bit resets.	32-bit	CTI Control register	Yes
0x010	CTIINTACK	See individual bit resets.	32-bit	CTI Output Trigger Acknowledge register	Yes

Offset	Name	Reset	Width	Description	Identical core CTI
0x014	CTIAPPSET	See individual bit resets.	32-bit	CTI Application Trigger Set register	Yes
0x018	CTIAPPCLEAR	See individual bit resets.	32-bit	CTI Application Trigger Clear register	Yes
0x01C	CTIAPPULSE	See individual bit resets.	32-bit	CTI Application Pulse register	Yes
0x20	CTIINEN0	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x24	CTIINEN1	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x28	CTIINEN2	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x2C	CTIINEN3	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x30	CTIINEN4	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x34	CTIINEN5	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x38	CTIINEN6	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x3C	CTIINEN7	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x40	CTIINEN8	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0x44	CTIINEN9	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes
0xA0	CTIOUTEN0	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xA4	CTIOUTEN1	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xA8	CTIOUTEN2	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xAC	CTIOUTEN3	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xB0	CTIOUTEN4	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xB4	CTIOUTEN5	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xB8	CTIOUTEN6	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xBC	CTIOUTEN7	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xC0	CTIOUTEN8	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes
0xC4	CTIOUTEN9	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes

Offset	Name	Reset	Width	Description	Identical core CTI
0x130	CTITRIGINSTATUS	See individual bit resets.	32-bit	CTI Trigger In Status register	Yes
0x134	CTITRIGOUTSTATUS	See individual bit resets.	32-bit	CTI Trigger Out Status register	Yes
0x138	CTICHINSTATUS	See individual bit resets.	32-bit	CTI Channel In Status register	Yes
0x13C	CTICHOUTSTATUS	See individual bit resets.	32-bit	CTI Channel Out Status register	Yes
0x140	CTIGATE	See individual bit resets.	32-bit	CTI Channel Gate Enable register	Yes
0x150	CTIDEVCTL	See individual bit resets.	32-bit	CTI Device Control register	Yes
0xFA0	CTICLAIMSET	See individual bit resets.	32-bit	CTI Claim Tag Set register	Yes
0xFA4	CTICLAIMCLR	See individual bit resets.	32-bit	CTI Claim Tag Clear register	Yes
0xFA8	CTIDEVAFF0	See individual bit resets.	32-bit	CTI Device Affinity register 0	No, see individual register
0xFAC	CTIDEVAFF1	See individual bit resets.	32-bit	CTI Device Affinity register 1	No, see individual register
0xFB8	CTIAUTHSTATUS	See individual bit resets.	32-bit	CTI Authentication Status register	Yes
0xFBC	CTIDEVARCH	See individual bit resets.	32-bit	CTI Device Architecture register	Yes
0xFC0	CTIDEVID2	See individual bit resets.	32-bit	CTI Device ID register 2	Yes
0xFC4	CTIDEVID1	See individual bit resets.	32-bit	CTI Device ID register 1	Yes
0xFC8	CTIDEVID	See individual bit resets.	32-bit	CTI Device ID register 0	Yes
0xFCC	CTIDEVTYPE	See individual bit resets.	32-bit	CTI Device Type register	Yes
0xFD0	CTIPIDR4	See individual bit resets.	32-bit	CTI Peripheral Identification Register 4	No, see individual register
0xFE0	CTIPIDR0	See individual bit resets.	32-bit	CTI Peripheral Identification Register 0	No, see individual register
0xFE4	CTIPIDR1	See individual bit resets.	32-bit	CTI Peripheral Identification Register 1	No, see individual register
0xFE8	CTIPIDR2	See individual bit resets.	32-bit	CTI Peripheral Identification Register 2	No, see individual register
0xFEC	CTIPIDR3	See individual bit resets.	32-bit	CTI Peripheral Identification Register 3	No, see individual register
0xFF0	CTICIDR0	See individual bit resets.	32-bit	CTI Component Identification Register 0	Yes
0xFF4	CTICIDR1	See individual bit resets.	32-bit	CTI Component Identification Register 1	Yes



Offset	Name	Reset	Width	Description	Identical core CTI
0xFF8	CTICIDR2	See individual bit resets.	32-bit	CTI Component Identification Register 2	Yes
0xFFC	CTICIDR3	See individual bit resets.	32-bit	CTI Component Identification Register 3	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

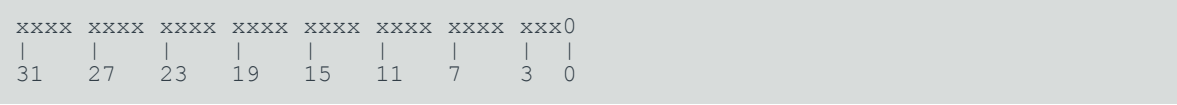
Register offset

0x000

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-165: ext\_cticontrol bit assignments

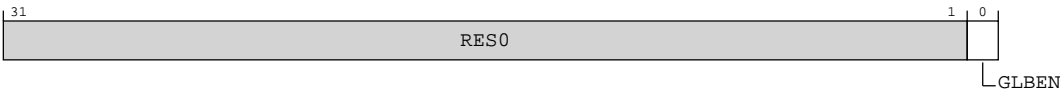


Table B-237: CTICONTROL bit descriptions

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:  <b>0b0</b> CTI mapping functions and application trigger disabled.  <b>0b1</b> 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	Range
CTI	0x000	CTICONTROL	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-166: ext\_ctiintack bit assignments

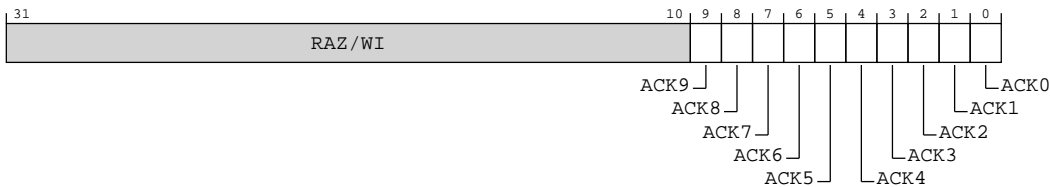


Table B-239: CTIINTACK bit descriptions

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: <ul style="list-style-type: none"><li>n &gt;= ext-CTIDEVID.NUMTRIG, the number of implemented triggers.</li><li>Output trigger n is not active.</li><li>The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li><li>Output trigger n is not implemented.</li><li>Output trigger n is not connected.</li><li>Output trigger n is self-acknowledging and does not require software acknowledge.</li></ul> Otherwise, the behavior on writes to ACK<n> is as follows: <b>0b0</b> No effect <b>0b1</b> Deactivate the trigger.	x

Bits	Name	Description	Reset
[8]	ACK8	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x
[7]	ACK7	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x

Bits	Name	Description	Reset
[6]	ACK6	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x
[5]	ACK5	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x

Bits	Name	Description	Reset
[4]	ACK4	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x
[3]	ACK3	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x

Bits	Name	Description	Reset
[2]	ACK2	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	x
[1]	ACK1	<p>Acknowledge for output trigger &lt;n&gt;.</p> <p>If any of the following is true, writes to ACK&lt;n&gt; are ignored:</p> <ul style="list-style-type: none"> <li>• <math>n \geq \text{ext-CTIDEVID.NUMTRIG}</math>, the number of implemented triggers.</li> <li>• Output trigger n is not active.</li> <li>• The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li> <li>• Output trigger n is not implemented.</li> <li>• Output trigger n is not connected.</li> <li>• Output trigger n is self-acknowledging and does not require software acknowledge.</li> </ul> <p>Otherwise, the behavior on writes to ACK&lt;n&gt; is as follows:</p> <p><b>0b0</b> No effect</p> <p><b>0b1</b> Deactivate the trigger.</p>	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: <ul style="list-style-type: none"><li>n &gt;= ext-CTIDEVID.NUMTRIG, the number of implemented triggers.</li><li>Output trigger n is not active.</li><li>The channel mapping function output, as controlled by ext-CTIOUTEN&lt;n&gt;, is still active.</li><li>Output trigger n is not implemented.</li><li>Output trigger n is not connected.</li><li>Output trigger n is self-acknowledging and does not require software acknowledge.</li></ul> Otherwise, the behavior on writes to ACK<n> is as follows: <b>0b0</b> No effect <b>0b1</b> Deactivate the trigger.	x

Accessibility

Component	Offset	Instance	Range
CTI	0x010	CTIINTACK	None

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-167: ext\_ctiappset bit assignments

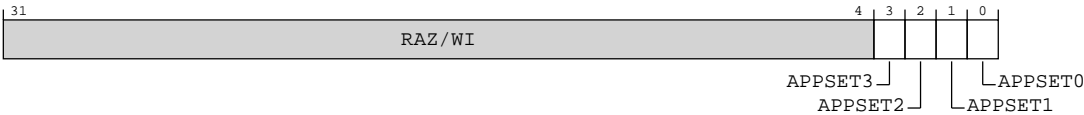


Table B-241: CTIAPPSET bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/ WI
[3]	APPSET3	Application trigger3 enable.  Possible values of this bit are:  <b>0b0</b> Reading this means the application trigger is inactive. Writing this has no effect.  <b>0b1</b> 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:  <b>0b0</b> Reading this means the application trigger is inactive. Writing this has no effect.  <b>0b1</b> 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:  <b>0b0</b> Reading this means the application trigger is inactive. Writing this has no effect.  <b>0b1</b> 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:  <b>0b0</b>  Reading this means the application trigger is inactive. Writing this has no effect.  <b>0b1</b>  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	Range
CTI	0x014	CTIAPPSET	None

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

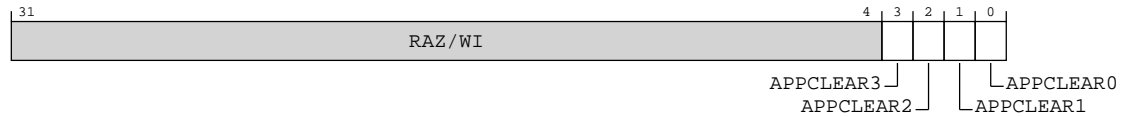
0000	0000	0000	0000	0000	0000	0000	xxxx
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-168: ext\_ctiappclear bit assignments**



**Table B-243: CTIAPPCLEAR bit descriptions**

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	APPCLEAR3	<p>Application trigger &lt;x&gt; disable.</p> <p>Writing to this bit has the following effect:</p> <p><b>0b0</b> No effect.</p> <p><b>0b1</b> Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.</p>	x
[2]	APPCLEAR2	<p>Application trigger &lt;x&gt; disable.</p> <p>Writing to this bit has the following effect:</p> <p><b>0b0</b> No effect.</p> <p><b>0b1</b> Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.</p>	x
[1]	APPCLEAR1	<p>Application trigger &lt;x&gt; disable.</p> <p>Writing to this bit has the following effect:</p> <p><b>0b0</b> No effect.</p> <p><b>0b1</b> Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.</p>	x
[0]	APPCLEAR0	<p>Application trigger &lt;x&gt; disable.</p> <p>Writing to this bit has the following effect:</p> <p><b>0b0</b> No effect.</p> <p><b>0b1</b> Clear corresponding bit in CTIAPPTRIG to 0 and clear the corresponding channel event.</p>	x

Accessibility

Component	Offset	Instance	Range
CTI	0x018	CTIAPPCLEAR	None

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-169: ext\_ctiapppulse bit assignments

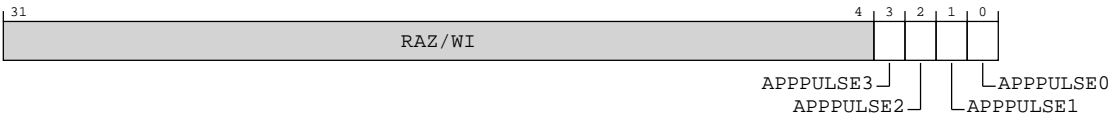


Table B-245: CTIAPPPULSE bit descriptions

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:  <b>0b0</b> No effect.  <b>0b1</b> Channel <x> event pulse generated.	x
[2]	APPPULSE2	Generate event pulse on ECT channel <x>.  Writing to this bit has the following effect:  <b>0b0</b> No effect.  <b>0b1</b> Channel <x> event pulse generated.	x
[1]	APPPULSE1	Generate event pulse on ECT channel <x>.  Writing to this bit has the following effect:  <b>0b0</b> No effect.  <b>0b1</b> Channel <x> event pulse generated.	x
[0]	APPPULSE0	Generate event pulse on ECT channel <x>.  Writing to this bit has the following effect:  <b>0b0</b> No effect.  <b>0b1</b> Channel <x> event pulse generated.	x

Accessibility

Component	Offset	Instance	Range
CTI	0x01C	CTIAPPPULSE	None

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

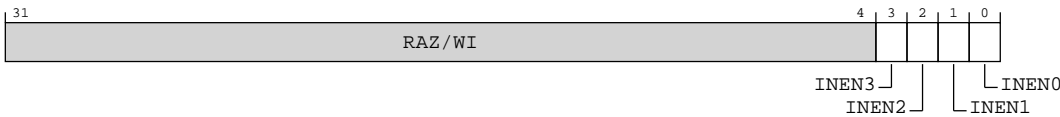
0000	0000	0000	0000	0000	0000	0000	xxxx
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-170: ext\_ctiinen0 bit assignments



**Table B-247: CTIINEN0 bit descriptions**

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	INEN3	Input trigger <n> to output channel <x> enable.  Possible values of this bit are:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

### Accessibility

Component	Offset	Instance	Range
CTI	0x20	CTIINEN0	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-171: ext\_ctiinen1 bit assignments

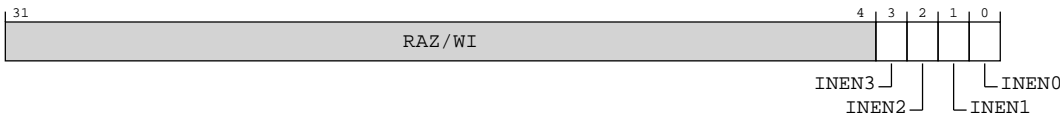


Table B-249: CTIINEN1 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x24	CTIINEN1	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-172: ext\_ctiinen2 bit assignments

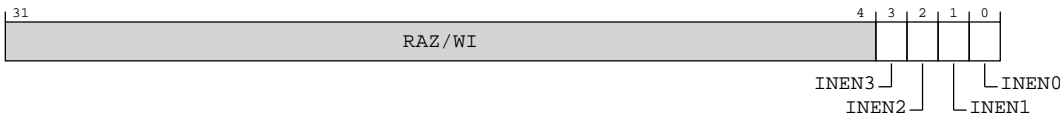


Table B-251: CTIINEN2 bit descriptions

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: <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>. <b>0b1</b> 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: <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>. <b>0b1</b> 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: <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>. <b>0b1</b> 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: <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>. <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x28	CTIINEN2	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-173: ext\_ctiinen3 bit assignments

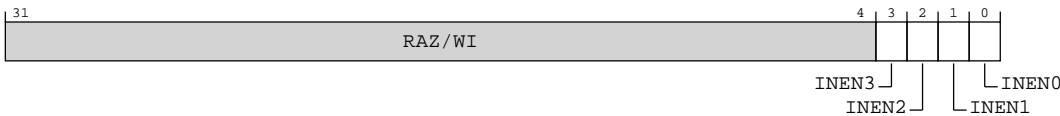


Table B-253: CTIINEN3 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x2C	CTIINEN3	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-174: ext\_ctiinen4 bit assignments

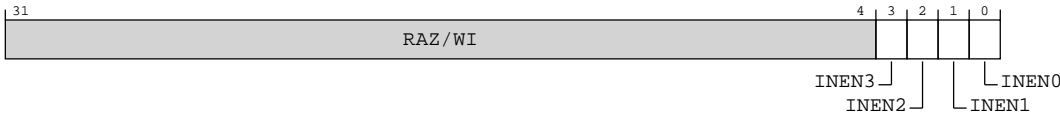


Table B-255: CTIINEN4 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x30	CTIINEN4	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-175: ext\_ctiinen5 bit assignments

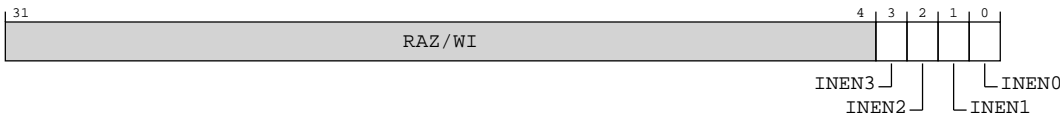


Table B-257: CTIINEN5 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x34	CTIINEN5	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-176: ext\_ctiinen6 bit assignments

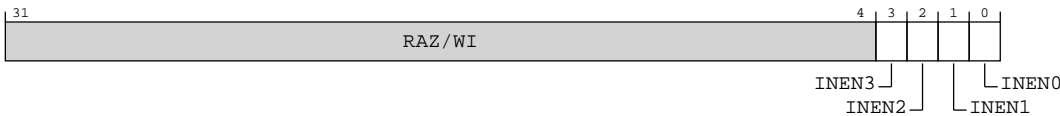


Table B-259: CTIINEN6 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x38	CTIINEN6	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-177: ext\_ctiinen7 bit assignments

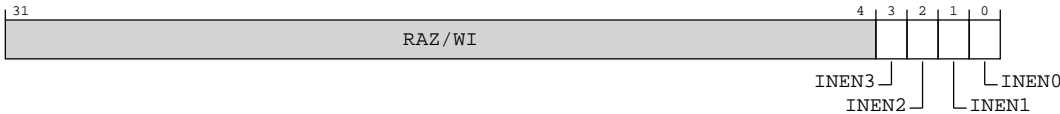


Table B-261: CTIINEN7 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x3C	CTIINEN7	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-178: ext\_ctiinen8 bit assignments

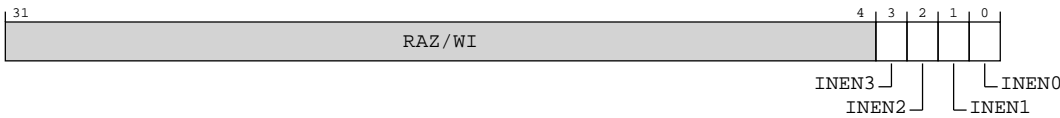


Table B-263: CTIINEN8 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x40	CTIINEN8	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-179: ext\_ctiinen9 bit assignments

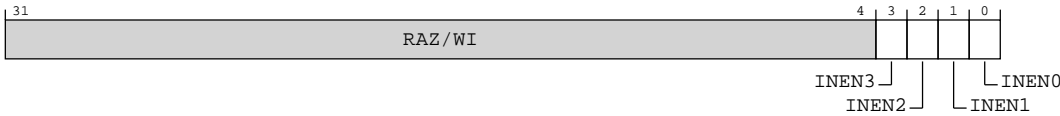


Table B-265: CTIINEN9 bit descriptions

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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> 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:  <b>0b0</b> Input trigger <n> will not generate an event on output channel <x>.  <b>0b1</b> Input trigger <n> will generate an event on output channel <x>.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0x44	CTIINEN9	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-180: ext\_ctiouten0 bit assignments

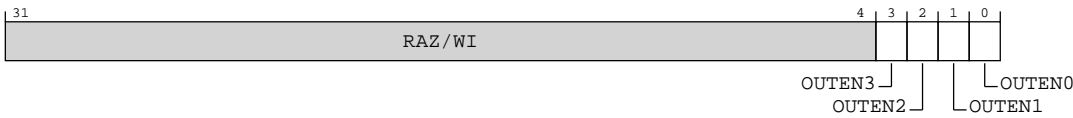


Table B-267: CTIOUTEN0 bit descriptions

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: <b>0b0</b> An event on input channel <x> will not cause output trigger <n> to be asserted. <b>0b1</b> 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: <b>0b0</b> An event on input channel <x> will not cause output trigger <n> to be asserted. <b>0b1</b> 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: <b>0b0</b> An event on input channel <x> will not cause output trigger <n> to be asserted. <b>0b1</b> 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: <b>0b0</b> An event on input channel <x> will not cause output trigger <n> to be asserted. <b>0b1</b> An event on input channel <x> will cause output trigger <n> to be asserted.	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xA0	CTIOUTEN0	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-181: ext\_ctiouten1 bit assignments

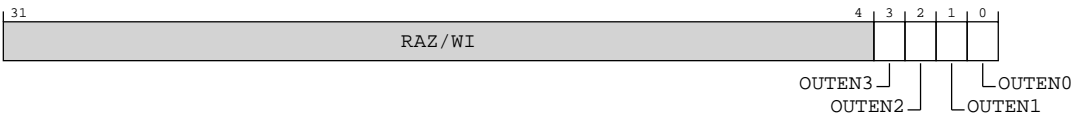


Table B-269: CTIOUTEN1 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xA4	CTIOUTEN1	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-182: ext\_ctiouten2 bit assignments

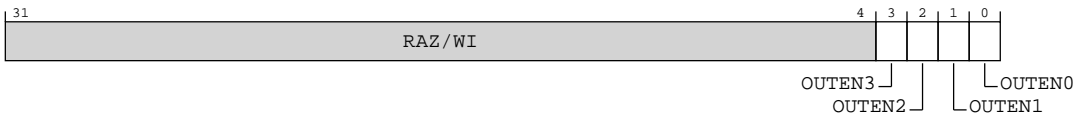


Table B-271: CTIOUTEN2 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xA8	CTIOUTEN2	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-183: ext\_ctiouten3 bit assignments

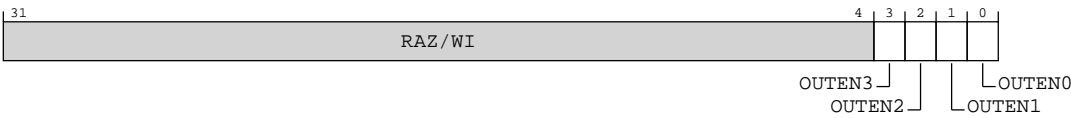


Table B-273: CTIOUTEN3 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI



Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xAC	CTIOUTEN3	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-184: ext\_ctiouten4 bit assignments

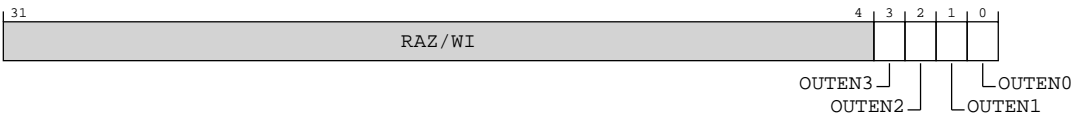


Table B-275: CTIOUTEN4 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xB0	CTIOUTEN4	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-185: ext\_ctiouten5 bit assignments

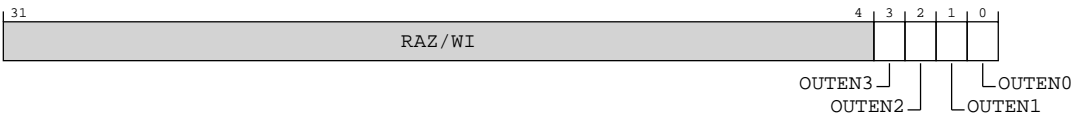


Table B-277: CTIOUTEN5 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xB4	CTIOUTEN5	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-186: ext\_ctiouten6 bit assignments

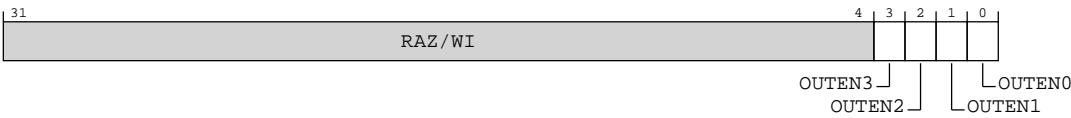


Table B-279: CTIOUTEN6 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xB8	CTIOUTEN6	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-187: ext\_ctiouten7 bit assignments

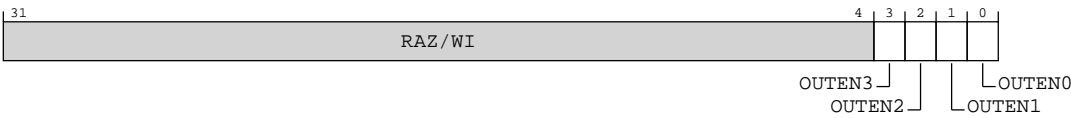


Table B-281: CTIOUTEN7 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI



Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xBC	CTIOUTEN7	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-188: ext\_ctiouten8 bit assignments

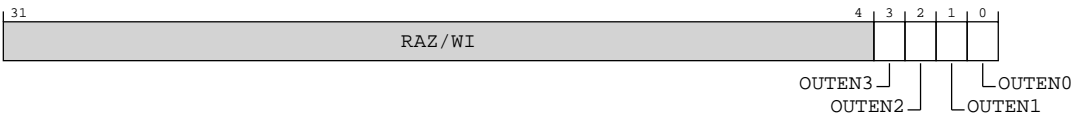


Table B-283: CTIOUTEN8 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xC0	CTIOUTEN8	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-189: ext\_ctiouten9 bit assignments

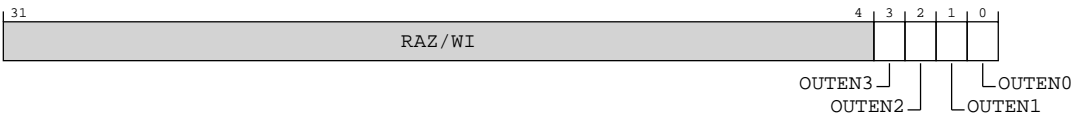


Table B-285: CTIOUTEN9 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	OUTEN3	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[2]	OUTEN2	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[1]	OUTEN1	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x
[0]	OUTEN0	<p>Input channel &lt;x&gt; to output trigger &lt;n&gt; enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> An event on input channel &lt;x&gt; will not cause output trigger &lt;n&gt; to be asserted.</p> <p><b>0b1</b> An event on input channel &lt;x&gt; will cause output trigger &lt;n&gt; to be asserted.</p>	x

## Accessibility

Component	Offset	Instance	Range
CTI	0xC4	CTIOUTEN9	None

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-190: ext\_ctitriginstatus bit assignments

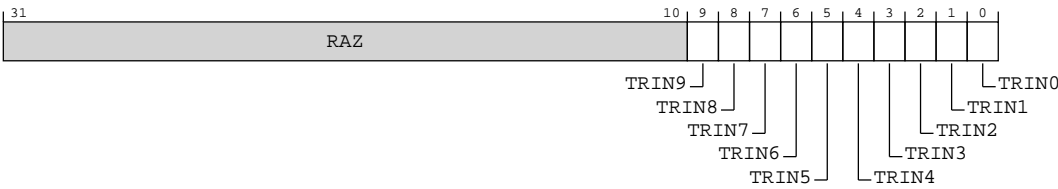


Table B-287: CTITRIGINSTATUS bit descriptions

Bits	Name	Description	Reset
[31:10]	RAZ	Reserved	RAZ
[9]	TRIN9	Trigger input <n> status.  Possible values of this bit are:  <b>0b0</b> Input trigger n is inactive.  <b>0b1</b> Input trigger n is active.	0b0

Bits	Name	Description	Reset
[8]	TRIN8	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[7]	TRIN7	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[6]	TRIN6	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[5]	TRIN5	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[4]	TRIN4	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[3]	TRIN3	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0

Bits	Name	Description	Reset
[2]	TRIN2	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[1]	TRIN1	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0
[0]	TRIN0	<p>Trigger input &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Input trigger n is inactive.</p> <p><b>0b1</b> Input trigger n is active.</p>	0b0

## Accessibility

Component	Offset	Instance	Range
CTI	0x130	CTITRIGINSTATUS	None

This interface is accessible as follows:

RO

### 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



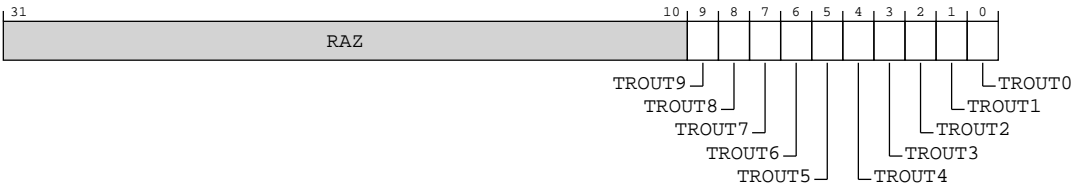
**Register offset**  
0x134

**Access type**  
RO

**Reset value**  
0000 0000 0000 0000 0000 0000 0000 0000

**Bit descriptions**

**Figure B-191: ext\_ctitrigoutstatus bit assignments**



**Table B-289: CTITRIGOUTSTATUS bit descriptions**

Bits	Name	Description	Reset
[31:10]	RAZ	Reserved	RAZ
[9]	TROUT9	Trigger output <n> status.  <b>0b0</b> Output trigger n is inactive.  <b>0b1</b> Output trigger n is active.  Otherwise when n < N TROUT<n> is <b>RAZ</b> .	0b0
[8]	TROUT8	Trigger output <n> status.  <b>0b0</b> Output trigger n is inactive.  <b>0b1</b> Output trigger n is active.  Otherwise when n < N TROUT<n> is <b>RAZ</b> .	0b0
[7]	TROUT7	Trigger output <n> status.  <b>0b0</b> Output trigger n is inactive.  <b>0b1</b> Output trigger n is active.  Otherwise when n < N TROUT<n> is <b>RAZ</b> .	0b0

Bits	Name	Description	Reset
[6]	TROUT6	<p>Trigger output &lt;n&gt; status.</p> <p><b>0b0</b></p> <p>Output trigger n is inactive.</p> <p><b>0b1</b></p> <p>Output trigger n is active.</p> <p>Otherwise when <math>n &lt; N</math> TROUT&lt;n&gt; is <b>RAZ</b>.</p>	0b0
[5]	TROUT5	<p>Trigger output &lt;n&gt; status.</p> <p><b>0b0</b></p> <p>Output trigger n is inactive.</p> <p><b>0b1</b></p> <p>Output trigger n is active.</p> <p>Otherwise when <math>n &lt; N</math> TROUT&lt;n&gt; is <b>RAZ</b>.</p>	0b0
[4]	TROUT4	<p>Trigger output &lt;n&gt; status.</p> <p><b>0b0</b></p> <p>Output trigger n is inactive.</p> <p><b>0b1</b></p> <p>Output trigger n is active.</p> <p>Otherwise when <math>n &lt; N</math> TROUT&lt;n&gt; is <b>RAZ</b>.</p>	0b0
[3]	TROUT3	<p>Trigger output &lt;n&gt; status.</p> <p><b>0b0</b></p> <p>Output trigger n is inactive.</p> <p><b>0b1</b></p> <p>Output trigger n is active.</p> <p>Otherwise when <math>n &lt; N</math> TROUT&lt;n&gt; is <b>RAZ</b>.</p>	0b0
[2]	TROUT2	<p>Trigger output &lt;n&gt; status.</p> <p><b>0b0</b></p> <p>Output trigger n is inactive.</p> <p><b>0b1</b></p> <p>Output trigger n is active.</p> <p>Otherwise when <math>n &lt; N</math> TROUT&lt;n&gt; is <b>RAZ</b>.</p>	0b0
[1]	TROUT1	<p>Trigger output &lt;n&gt; status.</p> <p><b>0b0</b></p> <p>Output trigger n is inactive.</p> <p><b>0b1</b></p> <p>Output trigger n is active.</p> <p>Otherwise when <math>n &lt; N</math> TROUT&lt;n&gt; is <b>RAZ</b>.</p>	0b0

Bits	Name	Description	Reset
[0]	TROUT0	Trigger output <n> status.  <b>0b0</b> Output trigger n is inactive.  <b>0b1</b> Output trigger n is active.  Otherwise when n < N TROUT<n> is <b>RAZ</b> .	0b0

Accessibility

Component	Offset	Instance	Range
CTI	0x134	CTITRIGOUTSTATUS	None

This interface is accessible as follows:

RO

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-192: ext\_ctichinstatus bit assignments

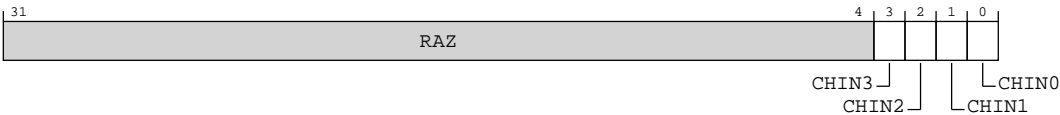


Table B-291: CTICHINSTATUS bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ	Reserved	RAZ
[3]	CHIN3	Input channel <n> status.  Possible values of this bit are:  <b>0b0</b> Input channel <n> is inactive.  <b>0b1</b> Input channel <n> is active.	0b0
[2]	CHIN2	Input channel <n> status.  Possible values of this bit are:  <b>0b0</b> Input channel <n> is inactive.  <b>0b1</b> Input channel <n> is active.	0b0
[1]	CHIN1	Input channel <n> status.  Possible values of this bit are:  <b>0b0</b> Input channel <n> is inactive.  <b>0b1</b> Input channel <n> is active.	0b0
[0]	CHIN0	Input channel <n> status.  Possible values of this bit are:  <b>0b0</b> Input channel <n> is inactive.  <b>0b1</b> Input channel <n> is active.	0b0

Accessibility

Component	Offset	Instance	Range
CTI	0x138	CTICHINSTATUS	None

This interface is accessible as follows:

RO

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-193: ext\_ctichoutstatus bit assignments

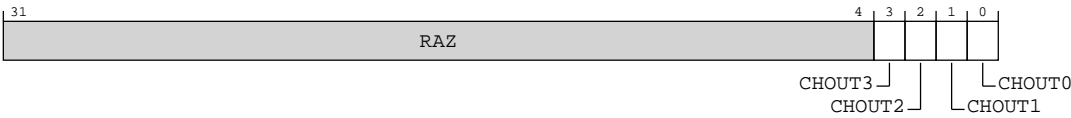


Table B-293: CTICHOUTSTATUS bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ	Reserved	RAZ
[3]	CHOUT3	Output channel <n> status.  Possible values of this bit are:  <b>0b0</b> Output channel <n> is inactive.  <b>0b1</b> Output channel <n> is active.  <b>Note:</b> The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	0b0

Bits	Name	Description	Reset
[2]	CHOUT2	<p>Output channel &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Output channel &lt;n&gt; is inactive.</p> <p><b>0b1</b> Output channel &lt;n&gt; is active.</p> <p><b>Note:</b> The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.</p>	0b0
[1]	CHOUT1	<p>Output channel &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Output channel &lt;n&gt; is inactive.</p> <p><b>0b1</b> Output channel &lt;n&gt; is active.</p> <p><b>Note:</b> The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.</p>	0b0
[0]	CHOUT0	<p>Output channel &lt;n&gt; status.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Output channel &lt;n&gt; is inactive.</p> <p><b>0b1</b> Output channel &lt;n&gt; is active.</p> <p><b>Note:</b> The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.</p>	0b0

## Accessibility

Component	Offset	Instance	Range
CTI	0x13C	CTICHOUTSTATUS	None

This interface is accessible as follows:

RO

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-194: ext\_ctigate bit assignments

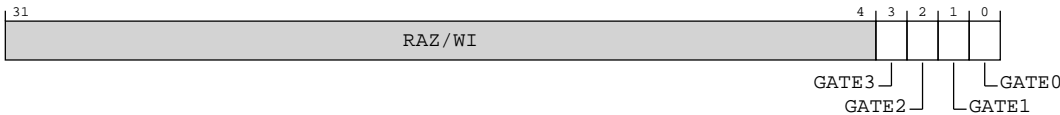


Table B-295: CTIGATE bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/ WI
[3]	GATE3	Channel <x> gate enable.  Possible values of this bit are:  <b>0b0</b> Disable output and, if CTIDEVID.INOUT == 0b01, input channel <x> propagation.  <b>0b1</b> 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

Bits	Name	Description	Reset
[2]	GATE2	<p>Channel &lt;x&gt; gate enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Disable output and, if CTIDEVID.INOUT == 0b01, input channel &lt;x&gt; propagation.</p> <p><b>0b1</b> Enable output and, if CTIDEVID.INOUT == 0b01, input channel &lt;x&gt; propagation.</p> <p>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.</p>	0b1
[1]	GATE1	<p>Channel &lt;x&gt; gate enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Disable output and, if CTIDEVID.INOUT == 0b01, input channel &lt;x&gt; propagation.</p> <p><b>0b1</b> Enable output and, if CTIDEVID.INOUT == 0b01, input channel &lt;x&gt; propagation.</p> <p>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.</p>	0b1
[0]	GATE0	<p>Channel &lt;x&gt; gate enable.</p> <p>Possible values of this bit are:</p> <p><b>0b0</b> Disable output and, if CTIDEVID.INOUT == 0b01, input channel &lt;x&gt; propagation.</p> <p><b>0b1</b> Enable output and, if CTIDEVID.INOUT == 0b01, input channel &lt;x&gt; propagation.</p> <p>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.</p>	0b1

## Accessibility

Component	Offset	Instance	Range
CTI	0x140	CTIGATE	None

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-195: ext\_ctidevctl bit assignments

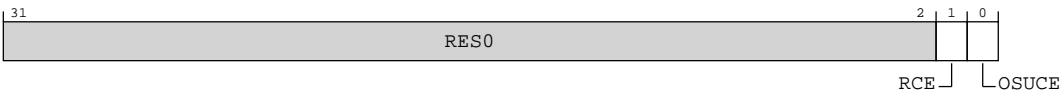


Table B-297: CTIDEVCTL bit descriptions

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

Bits	Name	Description	Reset
[0]	OSUCE	OS Unlock Catch Enable  <b>0b0</b> OS Unlock Catch debug event disabled.  <b>0b1</b> OS Unlock Catch debug event enabled.	0b0

Accessibility

Component	Offset	Instance	Range
CTI	0x150	CTIDEVCTL	None

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-196: ext\_ctclaimset bit assignments

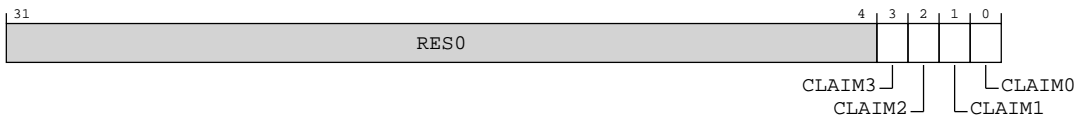


Table B-299: CTCLAIMSET bit descriptions

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 <sup>6</sup>
[2]	CLAIM2	CLAIM tag set bit.  0b0 No action.  0b1 Indirectly set claim bit to 1.	x <sup>7</sup>
[1]	CLAIM1	CLAIM tag set bit.  0b0 No action.  0b1 Indirectly set claim bit to 1.	x <sup>8</sup>
[0]	CLAIM0	CLAIM tag set bit.  0b0 No action.  0b1 Indirectly set claim bit to 1.	x <sup>9</sup>

<sup>6</sup> An External Debug reset clears the CLAIM tag bits to 0.  
<sup>7</sup> An External Debug reset clears the CLAIM tag bits to 0.  
<sup>8</sup> An External Debug reset clears the CLAIM tag bits to 0.  
<sup>9</sup> An External Debug reset clears the CLAIM tag bits to 0.

Accessibility

Component	Offset	Instance	Range
CTI	0xFA0	CTICLAIMSET	None

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-197: ext\_cticlaimer bit assignments

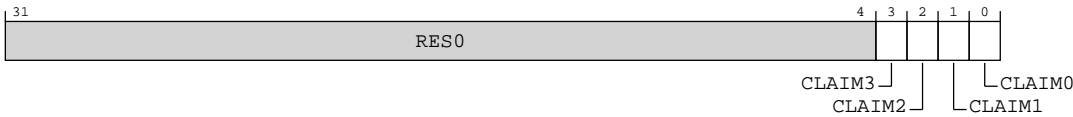


Table B-301: CTICLAIMCLR bit descriptions

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 <sup>10</sup>
[2]	CLAIM2	CLAIM tag clear bit.  0b0 No action.  0b1 Indirectly clear claim bit to 0.	x <sup>11</sup>
[1]	CLAIM1	CLAIM tag clear bit.  0b0 No action.  0b1 Indirectly clear claim bit to 0.	x <sup>12</sup>
[0]	CLAIM0	CLAIM tag clear bit.  0b0 No action.  0b1 Indirectly clear claim bit to 0.	x <sup>13</sup>

Accessibility

Component	Offset	Instance	Range
CTI	0xFA4	CTICLAIMCLR	None

This interface is accessible as follows:

When SoftwareLockStatus()

RO

<sup>10</sup> An External Debug reset clears the CLAIM tag bits to 0.  
<sup>11</sup> An External Debug reset clears the CLAIM tag bits to 0.  
<sup>12</sup> An External Debug reset clears the CLAIM tag bits to 0.  
<sup>13</sup> An External Debug reset clears the CLAIM tag bits to 0.

**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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-198: ext\_ctidevaff0 bit assignments

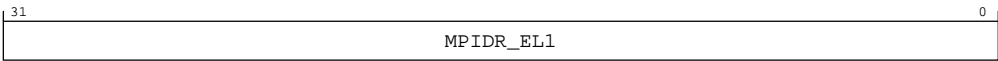


Table B-303: CTIDEVAFF0 bit descriptions

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.	<b>Cluster</b>  32 {x}  <b>Core</b>  See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.

Accessibility

Component	Offset	Instance	Range
CTI	0xFA8	CTIDEVAFF0	None

This interface is accessible as follows:

RO

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

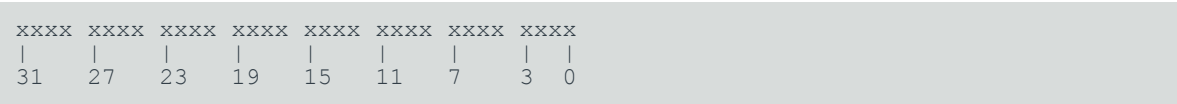
Register offset

0xFAC

Access type

RO

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-199: ext\_ctidevaff1 bit assignments

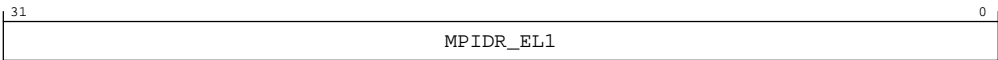


Table B-305: CTIDEVAFF1 bit descriptions

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.	<div>Cluster32 {x}</div> <div>CoreSee section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.</div>

Accessibility

Component	Offset	Instance	Range
CTI	0xFAC	CTIDEVAFF1	None

This interface is accessible as follows:

RO

B.2.1.36 CTIAUTHSTATUS, CTI Authentication Status register

Provides information about the state of the authentication interface for CTI.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

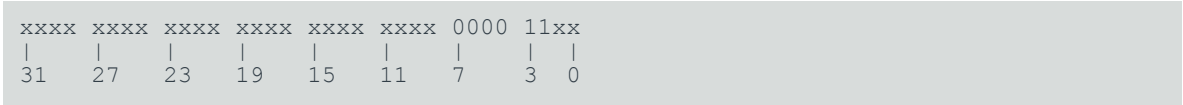
Register offset

0xFB8



Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-200: ext\_ctiauthstatus bit assignments

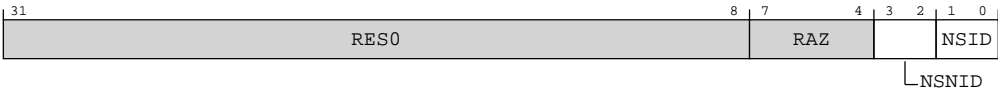


Table B-307: CTIAUTHSTATUS bit descriptions

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.  <b>0b11</b> Supported and enabled. DBGEN == TRUE.	0b11
[1:0]	NSID	Holds the same value as ext-DBGAUTHSTATUS_EL1.SID.  <b>0b10</b> Secure invasive debug disabled. DBGEN == FALSE.  <b>0b11</b> Secure invasive debug enabled. DBGEN == TRUE.	xx

Accessibility

Component	Offset	Instance	Range
CTI	0xFB8	CTIAUTHSTATUS	None

This interface is accessible as follows:

RO

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-201: ext\_ctidevarch bit assignments

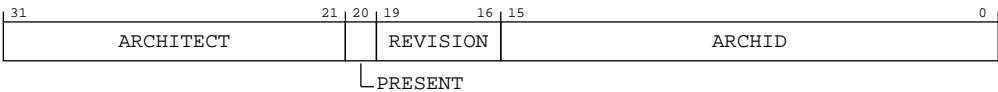


Table B-309: CTIDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. <b>0b01000111011</b> JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. <b>0b1</b> DEVARCH information present.	0b1
[19:16]	REVISION	Revision. Defines the architecture revision of the component. <b>0b0001</b> First revision, and also adds support for ext-CTIDEVCTL.	0b0001
[15:0]	ARCHID	Architecture ID. <b>0b0001101000010100</b> Cross Trigger Interface (CTI) architecture CTIv2.	0x1A14

Accessibility

Component	Offset	Instance	Range
CTI	0xFBC	CTIDEVARCH	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-202: ext\_ctidevid2 bit assignments



Table B-311: CTIDEVID2 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CTI	0xFC0	CTIDEVID2	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-203: ext\_ctidevid1 bit assignments

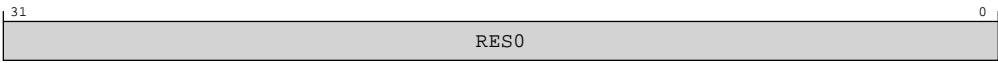


Table B-313: CTIDEVID1 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CTI	0xFC4	CTIDEVID1	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-204: ext\_ctidevid bit assignments

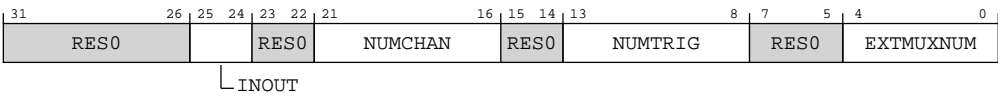


Table B-315: CTIDEVID bit descriptions

Bits	Name	Description	Reset
[31:26]	RES0	Reserved	RES0
[25:24]	INOUT	Input/output options. Indicates presence of the input gate.  <b>0b00</b> ext-CTIGATE does not mask propagation of input events from external channels.  <b>0b01</b> ext-CTIGATE masks propagation of input events from external channels.  All other values are reserved.	0b01
[23:22]	RES0	Reserved	RES0
[21:16]	NUMCHAN	Number of ECT channels implemented.  <b>0b000100</b> 4 channels (0..3) implemented.	0b000100
[15:14]	RES0	Reserved	RES0
[13:8]	NUMTRIG	Number of triggers implemented.  <b>0b001010</b> 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.  <b>0b000000</b> No multiplexors implemented. ext-ASICCTL is unused.	5 {x}

Accessibility

Component	Offset	Instance	Range
CTI	0xFC8	CTIDEVID	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-205: ext\_ctidevtype bit assignments

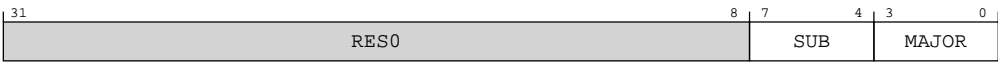


Table B-317: CTIDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SUB	Subtype. <b>0b0001</b> Embedded Cross-Trigger.	0b0001
[3:0]	MAJOR	Major type. <b>0b0100</b> Debug Control.	0b0100

Accessibility

Component	Offset	Instance	Range
CTI	0xFCC	CTIDEVTYPE	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-206: ext\_ctipidr4 bit assignments

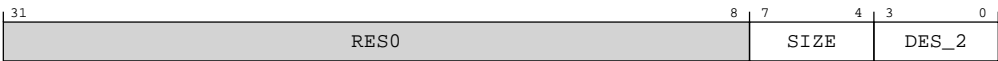




Table B-319: CTIPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count.  0b0000 The component uses a single 4KB block.	Cluster 0b0000  Core See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.
[3:0]	DES_2	JEP106 continuation code.  0b0100 Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	Cluster 0b0100  Core See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.

Accessibility

Component	Offset	Instance	Range
CTI	0xFD0	CTIPIDR4	None

This interface is accessible as follows:

RO

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





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-207: ext\_ctipidr0 bit assignments

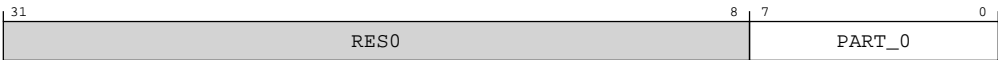


Table B-321: CTIPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0].  0b11101100  DSU-120AE Cross Trigger Interface. Bits [7:0] of part number 0x4EC.	Cluster 0xEC  Core See section CTI register identification values in chapter Debug in your core Technical Reference Manual for this value.

Accessibility

Component	Offset	Instance	Range
CTI	0xFE0	CTIPIDR0	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-208: ext\_ctipidr1 bit assignments



Table B-323: CTIPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0]. <b>0b1011</b> Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	<b>Cluster</b> 0b1011 <b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.
[3:0]	PART_1	Part number bits [11:8]. <b>0b0100</b> DSU-120AE Cross Trigger Interface. Bits [11:8] of part number 0x4EC.	<b>Cluster</b> 0b0100 <b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.

Accessibility

Component	Offset	Instance	Range
CTI	0xFE4	CTIPIDR1	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-209: ext\_ctipidr2 bit assignments

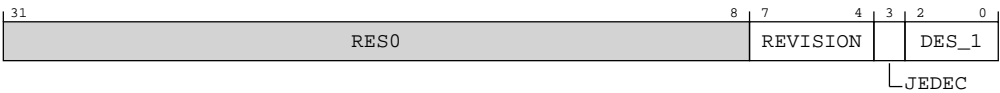


Table B-325: CTIPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVISION	<p>Component major revision.</p> <p><b>0b0000</b> Component major revision 0.</p> <p><b>0b0001</b> Component major revision 1.</p> <p>For DSU-120AE:</p> <ul style="list-style-type: none"> <li>Major revision 0 corresponds to r0p0.</li> <li>Major revision 1 corresponds to r0p1.</li> </ul>	<p><b>Cluster</b> 0b0001</p> <p><b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.</p>
[3]	JEDEC	<p>JEDEC assignee.</p> <p><b>0b1</b> JEDEC-assignee values is used.</p>	<p><b>Cluster</b> 0b1</p> <p><b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.</p>
[2:0]	DES_1	<p>JEP106 identification code bits [6:4].</p> <p><b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.</p>	<p><b>Cluster</b> 0b011</p> <p><b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.</p>

## Accessibility

Component	Offset	Instance	Range
CTI	0xFE8	CTIPIDR2	None

This interface is accessible as follows:

RO

### 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-210: ext\_ctipidr3 bit assignments



Table B-327: CTIPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVAND	Component minor revision. <b>0b0000</b> Component minor revision 0.	<b>Cluster</b> 0b0000 <b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.
[3:0]	CMOD	Customer Modified. <b>0b0000</b> The component is not modified from the original design.	<b>Cluster</b> 0b0000 <b>Core</b> See section <i>CTI register identification values</i> in chapter <i>Debug</i> in your core <i>Technical Reference Manual</i> for this value.

Accessibility

Component	Offset	Instance	Range
CTI	0xFEC	CTIPIDR3	None

This interface is accessible as follows:

RO

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

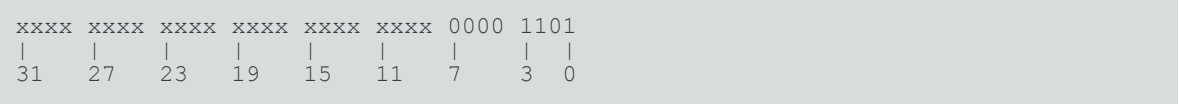
Register offset

0xFF0

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-211: ext\_cticidr0 bit assignments



Table B-329: CTICIDR0 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	Instance	Range
CTI	0xFF0	CTICIDR0	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-212: ext\_cticidr1 bit assignments





Table B-331: 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

Accessibility

Component	Offset	Instance	Range
CTI	0xFF4	CTICIDR1	None

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-213: ext\_cticidr2 bit assignments

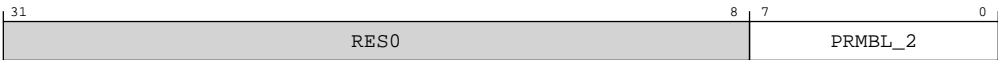


Table B-333: CTICIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	Preamble.  0b00000101 CoreSight component identification preamble.	0x05

Accessibility

Component	Offset	Instance	Range
CTI	0xFF8	CTICIDR2	None

This interface is accessible as follows:

RO

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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-214: ext\_cticidr3 bit assignments



Table B-335: CTICIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_3	Preamble. <b>0b10110001</b> CoreSight component identification preamble.	0xB1

Accessibility

Component	Offset	Instance	Range
CTI	0xFFC	CTICIDR3	None

This interface is accessible as follows:

RO

B.2.2 External cluster ROM registers summary

The cluster ROM table registers are only accessible using memory-mapped accesses over the debug APB interface.

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**.
- The number of registers that contain valid entries depends on the number of cores configured for the cluster.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table B-337: CLUSTERROM registers summary**

Offset	Name	Reset	Width	Description
0x000	CLUSTERROM_ROMENTRY0	See individual bit resets.	32-bit	Cluster ROM table entry 0
0x004	CLUSTERROM_ROMENTRY1	See individual bit resets.	32-bit	Cluster ROM table entry 1
0x008	CLUSTERROM_ROMENTRY2	See individual bit resets.	32-bit	Cluster ROM table entry 2
0x00C	CLUSTERROM_ROMENTRY3	See individual bit resets.	32-bit	Cluster ROM table entry 3
0x010	CLUSTERROM_ROMENTRY4	See individual bit resets.	32-bit	Cluster ROM table entry 4
0x014	CLUSTERROM_ROMENTRY5	See individual bit resets.	32-bit	Cluster ROM table entry 5
0x018	CLUSTERROM_ROMENTRY6	See individual bit resets.	32-bit	Cluster ROM table entry 6
0x01C	CLUSTERROM_ROMENTRY7	See individual bit resets.	32-bit	Cluster ROM table entry 7
0x020	CLUSTERROM_ROMENTRY8	See individual bit resets.	32-bit	Cluster ROM table entry 8
0x024	CLUSTERROM_ROMENTRY9	See individual bit resets.	32-bit	Cluster ROM table entry 9
0x028	CLUSTERROM_ROMENTRY10	See individual bit resets.	32-bit	Cluster ROM table entry 10
0x02C	CLUSTERROM_ROMENTRY11	See individual bit resets.	32-bit	Cluster ROM table entry 11
0x030	CLUSTERROM_ROMENTRY12	See individual bit resets.	32-bit	Cluster ROM table entry 12
0x034	CLUSTERROM_ROMENTRY13	See individual bit resets.	32-bit	Cluster ROM table entry 13
0x038	CLUSTERROM_ROMENTRY14	See individual bit resets.	32-bit	Cluster ROM table entry 14
0x03C	CLUSTERROM_ROMENTRY15	See individual bit resets.	32-bit	Cluster ROM table entry 15
0xA00	CLUSTERROM_DBGPCR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 0
0xA04	CLUSTERROM_DBGPCR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 1
0xA08	CLUSTERROM_DBGPCR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 2
0xA0C	CLUSTERROM_DBGPCR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 3
0xA10	CLUSTERROM_DBGPCR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 4
0xA14	CLUSTERROM_DBGPCR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 5
0xA18	CLUSTERROM_DBGPCR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 6
0xA1C	CLUSTERROM_DBGPCR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 7
0xA20	CLUSTERROM_DBGPCR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 8
0xA24	CLUSTERROM_DBGPCR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 9
0xA28	CLUSTERROM_DBGPCR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 10
0xA2C	CLUSTERROM_DBGPCR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 11
0xA30	CLUSTERROM_DBGPCR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 12
0xA34	CLUSTERROM_DBGPCR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register 13

Offset	Name	Reset	Width	Description
0xA80	CLUSTERROM_DBGPSR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 0
0xA84	CLUSTERROM_DBGPSR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 1
0xA88	CLUSTERROM_DBGPSR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 2
0xA8C	CLUSTERROM_DBGPSR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 3
0xA90	CLUSTERROM_DBGPSR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 4
0xA94	CLUSTERROM_DBGPSR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 5
0xA98	CLUSTERROM_DBGPSR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 6
0xA9C	CLUSTERROM_DBGPSR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 7
0xAA0	CLUSTERROM_DBGPSR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 8
0xAA4	CLUSTERROM_DBGPSR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 9
0xAA8	CLUSTERROM_DBGPSR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 10
0xAAC	CLUSTERROM_DBGPSR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 11
0xAB0	CLUSTERROM_DBGPSR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 12
0xAB4	CLUSTERROM_DBGPSR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register 13
0xC00	CLUSTERROM_PRIDR0	See individual bit resets.	32-bit	Cluster ROM table Power Request ID Register 0
0xFB8	CLUSTERROM_AUTHSTATUS	See individual bit resets.	32-bit	Cluster ROM table Authentication Status Register
0xFBC	CLUSTERROM_DEVARCH	See individual bit resets.	32-bit	Cluster ROM table Device Architecture Register
0xFC8	CLUSTERROM_DEVID	See individual bit resets.	32-bit	Cluster ROM table Device Configuration Register
0xFCC	CLUSTERROM_DEVTYPE	See individual bit resets.	32-bit	Cluster ROM table Device Type Register
0xFD0	CLUSTERROM_PIDR4	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 4
0xFE0	CLUSTERROM_PIDR0	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 0
0xFE4	CLUSTERROM_PIDR1	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 1
0xFE8	CLUSTERROM_PIDR2	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 2
0xFEC	CLUSTERROM_PIDR3	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 3
0xFF0	CLUSTERROM_CIDR0	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 0
0xFF4	CLUSTERROM_CIDR1	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 1
0xFF8	CLUSTERROM_CIDR2	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 2
0xFFC	CLUSTERROM_CIDR3	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 3

### 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-215: ext\_clusterrom\_romentry0 bit assignments

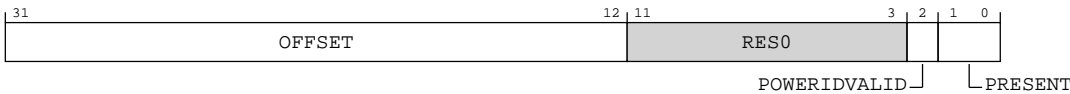


Table B-338: CLUSTERROM\_ROMENTRY0 bit descriptions

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).  <b>0b000000000000000010000</b> Cluster PMU table at address 0x2_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.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

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

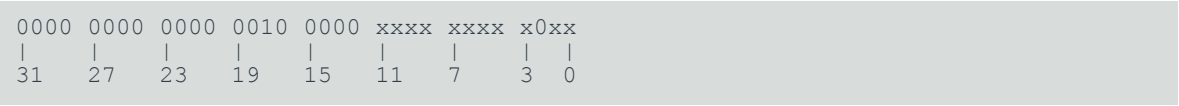
Register offset

0x004

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-216: ext\_clusterrom\_romentry1 bit assignments

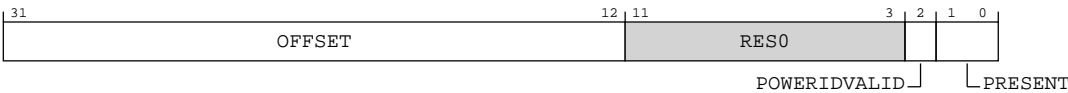


Table B-339: CLUSTERROM\_ROMENTRY1 bit descriptions

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).  <b>0b0000000000000001000000</b> Cluster ELA at address 0x3_0000 in the Cluster Debug APB address map.	0x00020
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b10</b> ELA is not present so the ROM entry is not present.  <b>0b11</b> ELA is present so the ROM entry is present.	xx

Accessibility

This interface is accessible as follows:

RO

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	0111	0000	xxxx	xxxx	xxxx
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.



Bit descriptions

Figure B-217: ext\_clusterrom\_romentry2 bit assignments

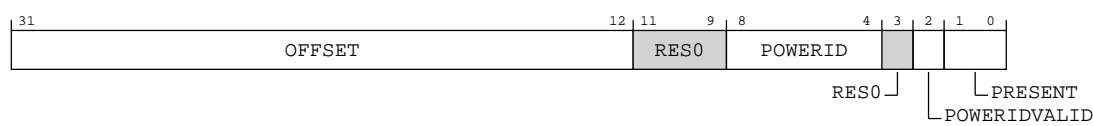


Table B-340: CLUSTERROM\_ROMENTRY2 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0000000000000001110000</b></p> <p>Core 0 ROM table at address 0x8_0000 in the Cluster Debug APB address map.</p>	0x00070
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	x

Bits	Name	Description	Reset
[1:0]	PRESENT	<p>Indicates whether an entry is present at this location in the ROM Table.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.</p> <p><b>0b01</b> Reserved.</p> <p><b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.</p> <p><b>0b11</b> The ROM entry is present.</p>	xx

### Accessibility

This interface is accessible as follows:

RO

### 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

### Bit descriptions

When NUM\_CORES >= 2

Figure B-218: ext\_clusterrom\_romentry3 bit assignments

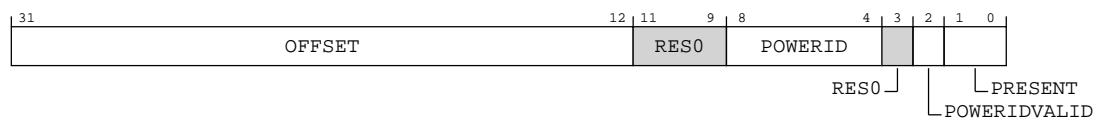


Table B-341: CLUSTERROM\_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0000000000000011110000</b></p> <p>Core 1 ROM table at address 0x10_0000 in the Cluster Debug APB address map.</p>	0x000F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 2

Figure B-219: ext\_clusterrom\_romentry3 bit assignments

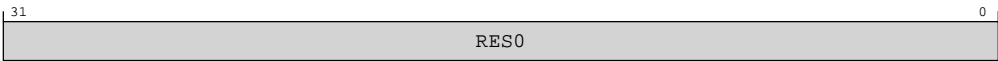


Table B-342: CLUSTERROM\_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 3

Figure B-220: ext\_clusterrom\_romentry4 bit assignments

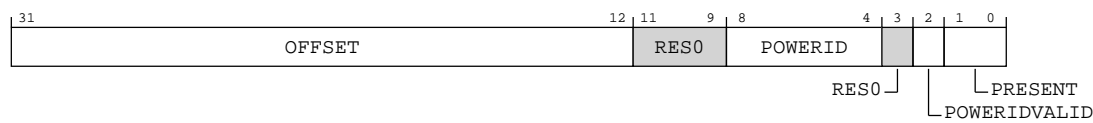


Table B-343: CLUSTERROM\_ROMENTRY4 bit descriptions

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.  <b>0b000000000000101110000</b> Core 2 ROM table at address 0x18_0000 in the Cluster Debug APB address map.	0x00170
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 3

Figure B-221: ext\_clusterrom\_romentry4 bit assignments

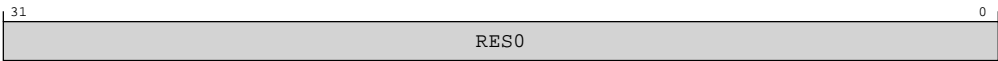


Table B-344: CLUSTERROM\_ROMENTRY4 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 4

Figure B-222: ext\_clusterrom\_romentry5 bit assignments

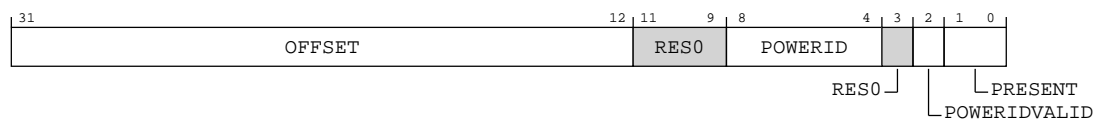


Table B-345: CLUSTERROM\_ROMENTRY5 bit descriptions

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.  <b>0b000000000000111110000</b> Core 3 ROM table at address 0x20_0000 in the Cluster Debug APB address map.	0x001F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 4

Figure B-223: ext\_clusterrom\_romentry5 bit assignments

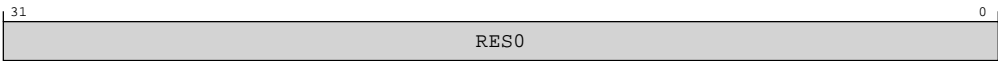


Table B-346: CLUSTERROM\_ROMENTRY5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 5

Figure B-224: ext\_clusterrom\_romentry6 bit assignments

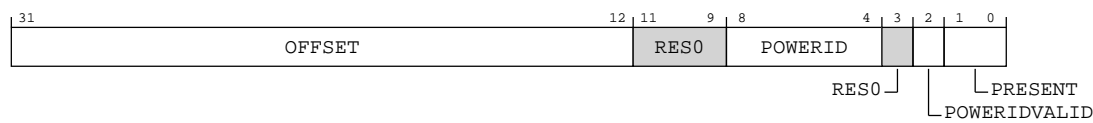


Table B-347: CLUSTERROM\_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000000001001110000</b></p> <p>Core 4 ROM table at address 0x28_0000 in the Cluster Debug APB address map.</p>	0x00270
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 5

Figure B-225: ext\_clusterrom\_romentry6 bit assignments

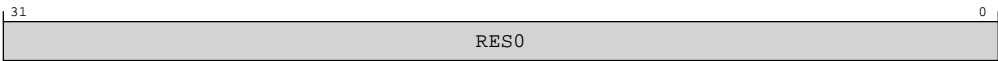


Table B-348: CLUSTERROM\_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 6

Figure B-226: ext\_clusterrom\_romentry7 bit assignments

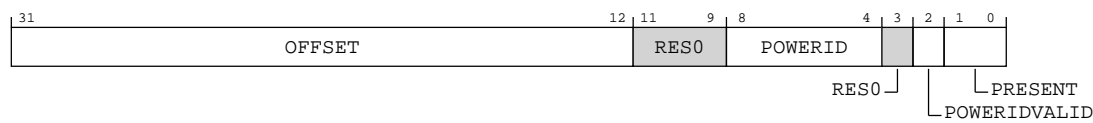


Table B-349: CLUSTERROM\_ROMENTRY7 bit descriptions

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.  <b>0b00000000001011110000</b> Core 5 ROM table at address 0x30_0000 in the Cluster Debug APB address map.	0x002F0
[11:9]	RES0	Reserved	RES0



Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 6

Figure B-227: ext\_clusterrom\_romentry7 bit assignments

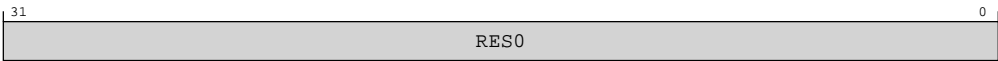


Table B-350: CLUSTERROM\_ROMENTRY7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 7

Figure B-228: ext\_clusterrom\_romentry8 bit assignments

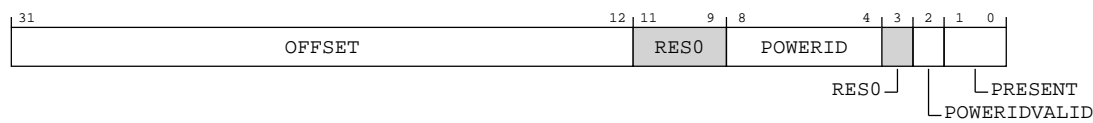


Table B-351: CLUSTERROM\_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000000001101110000</b></p> <p>Core 6 ROM table at address 0x38_0000 in the Cluster Debug APB address map.</p>	0x00370
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 7

Figure B-229: ext\_clusterrom\_romentry8 bit assignments

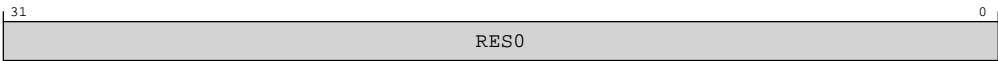


Table B-352: CLUSTERROM\_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 8

Figure B-230: ext\_clusterrom\_romentry9 bit assignments

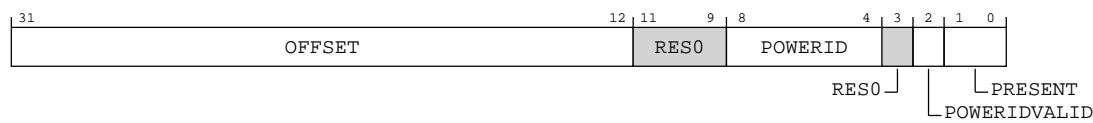


Table B-353: CLUSTERROM\_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000000001111110000</b></p> <p>Core 7 ROM table at address 0x40_0000 in the Cluster Debug APB address map.</p>	0x003F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 8

Figure B-231: ext\_clusterrom\_romentry9 bit assignments

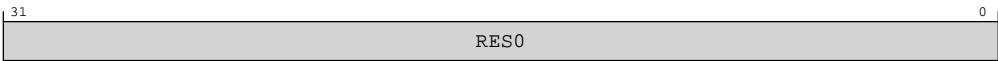


Table B-354: CLUSTERROM\_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 9

Figure B-232: ext\_clusterrom\_romentry10 bit assignments

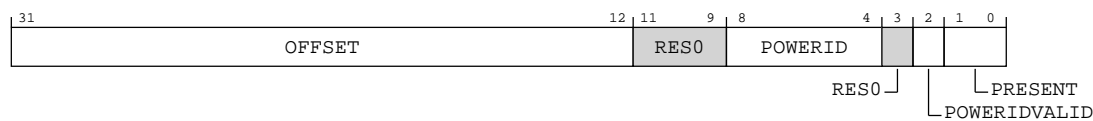


Table B-355: CLUSTERROM\_ROMENTRY10 bit descriptions

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.  <b>0b00000000010001110000</b> Core 8 ROM table at address 0x48_0000 in the Cluster Debug APB address map.	0x00470
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 9

Figure B-233: ext\_clusterrom\_romentry10 bit assignments

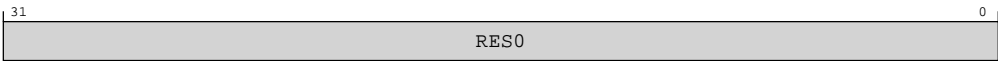


Table B-356: CLUSTERROM\_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 10

Figure B-234: ext\_clusterrom\_romentry11 bit assignments

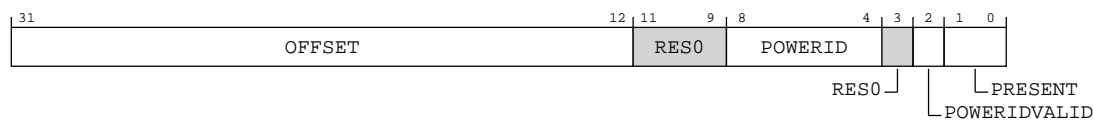


Table B-357: CLUSTERROM\_ROMENTRY11 bit descriptions

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.  <b>0b00000000010011110000</b> Core 9 ROM table at address 0x50_0000 in the Cluster Debug APB address map.	0x004F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 10

Figure B-235: ext\_clusterrom\_romentry11 bit assignments

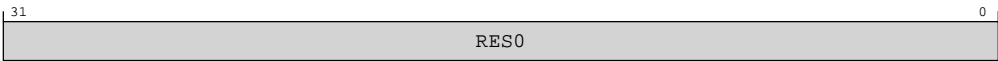


Table B-358: CLUSTERROM\_ROMENTRY11 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 11

Figure B-236: ext\_clusterrom\_romentry12 bit assignments

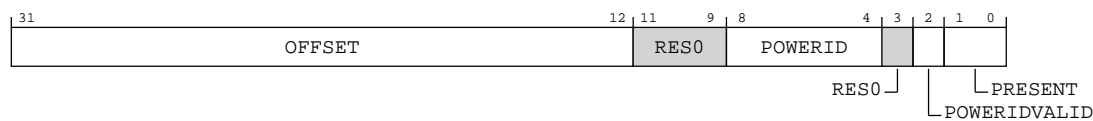


Table B-359: CLUSTERROM\_ROMENTRY12 bit descriptions

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.  <b>0b00000000010101110000</b> Core 10 ROM table at address 0x58_0000 in the Cluster Debug APB address map.	0x00570
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 11

Figure B-237: ext\_clusterrom\_romentry12 bit assignments

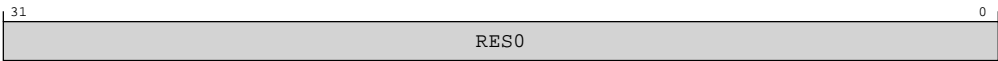


Table B-360: CLUSTERROM\_ROMENTRY12 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 12

Figure B-238: ext\_clusterrom\_romentry13 bit assignments

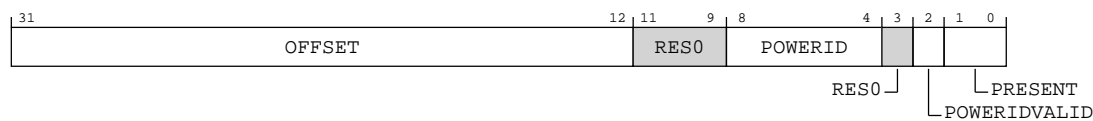


Table B-361: CLUSTERROM\_ROMENTRY13 bit descriptions

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.  <b>0b00000000010111110000</b> Core 11 ROM table at address 0x60_0000 in the Cluster Debug APB address map.	0x005F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 12

Figure B-239: ext\_clusterrom\_romentry13 bit assignments

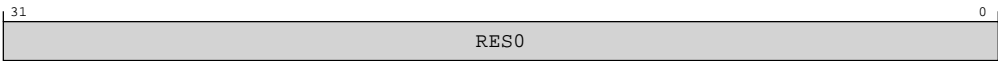


Table B-362: CLUSTERROM\_ROMENTRY13 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.15 CLUSTERROM\_ROMENTRY14, Cluster ROM table entry 14

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x038

Access type  
RO

Bit descriptions  
When NUM\_CORES >= 13

Figure B-240: ext\_clusterrom\_romentry14 bit assignments

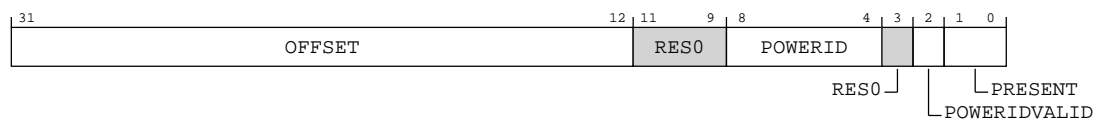


Table B-363: CLUSTERROM\_ROMENTRY14 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b00000000011001110000</b></p> <p>Core 12 ROM table at address 0x68_0000 in the Cluster Debug APB address map.</p>	0x00670
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 13

Figure B-241: ext\_clusterrom\_romentry14 bit assignments

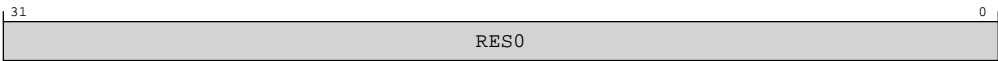


Table B-364: CLUSTERROM\_ROMENTRY14 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.16 CLUSTERROM\_ROMENTRY15, Cluster ROM table entry 15

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x03C

Access type  
RO

Bit descriptions  
When NUM\_CORES >= 14

Figure B-242: ext\_clusterrom\_romentry15 bit assignments

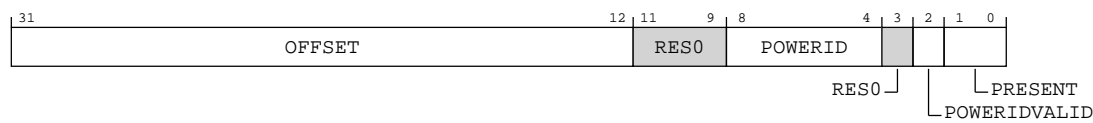


Table B-365: CLUSTERROM\_ROMENTRY15 bit descriptions

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.  <b>0b00000000011011110000</b> Core 13 ROM table at address 0x70_0000 in the Cluster Debug APB address map.	0x006F0
[11:9]	RES0	Reserved	RES0



Bits	Name	Description	Reset
[8:4]	POWERID	<p>The power domain ID of the component. This field is only valid if the POWERIDVALID field is 0b1.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b000000</b> PDCOMPLEX0 power domain.</p> <p><b>0b000001</b> PDCOMPLEX1 power domain.</p> <p><b>0b000010</b> PDCOMPLEX2 power domain.</p> <p><b>0b000011</b> PDCOMPLEX3 power domain.</p> <p><b>0b000100</b> PDCOMPLEX4 power domain.</p> <p><b>0b000101</b> PDCOMPLEX5 power domain.</p> <p><b>0b000110</b> PDCOMPLEX6 power domain.</p> <p><b>0b000111</b> PDCOMPLEX7 power domain.</p> <p><b>0b001000</b> PDCOMPLEX8 power domain.</p> <p><b>0b001001</b> PDCOMPLEX9 power domain.</p> <p><b>0b001010</b> PDCOMPLEX10 power domain.</p> <p><b>0b001011</b> PDCOMPLEX11 power domain.</p> <p><b>0b001100</b> PDCOMPLEX12 power domain.</p> <p><b>0b001101</b> PDCOMPLEX13 power domain.</p>	5 {x}
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	<p>Indicates if the Power domain ID field contains a Power domain ID.</p> <p>The value of this field depends on the cluster configuration.</p> <p><b>0b0</b> A power domain ID is not provided.</p> <p><b>0b1</b> The POWERID field provides a power domain ID.</p>	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.  <b>0b00</b> The ROM entry is not present and this is the final entry in the ROM table.  <b>0b01</b> Reserved.  <b>0b10</b> The ROM entry is not present and this is not the final entry in the ROM table.  <b>0b11</b> The ROM entry is present.	xx

When NUM\_CORES < 14

Figure B-243: ext\_clusterrom\_romentry15 bit assignments

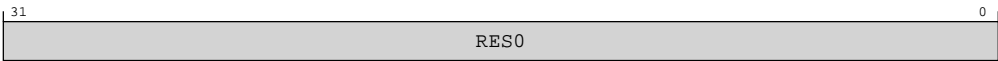


Table B-366: CLUSTERROM\_ROMENTRY15 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.17 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-244: ext\_clusterrom\_dbgpcr0 bit assignments



Table B-367: CLUSTERROM\_DBGPCR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

Accessibility

This interface is accessible as follows:

RW

B.2.2.18 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

Bit descriptions

When NUM\_CORES >= 2

Figure B-245: ext\_clusterrom\_dbgpcr1 bit assignments

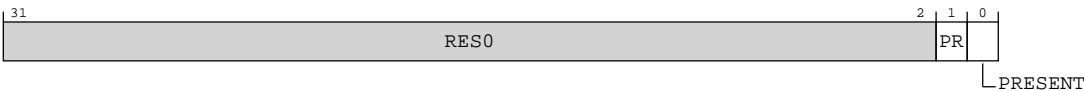


Table B-368: CLUSTERROM\_DBGPCR1 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 2

Figure B-246: ext\_clusterrom\_dbgpcr1 bit assignments

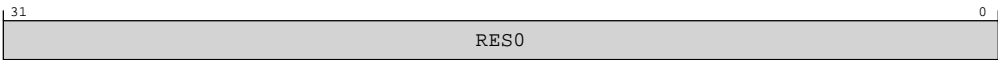


Table B-369: CLUSTERROM\_DBGPCR1 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.19 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

Bit descriptions

When NUM\_CORES >= 3

Figure B-247: ext\_clusterrom\_dbgpcr2 bit assignments



Table B-370: CLUSTERROM\_DBGPCR2 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 3

Figure B-248: ext\_clusterrom\_dbgpcr2 bit assignments

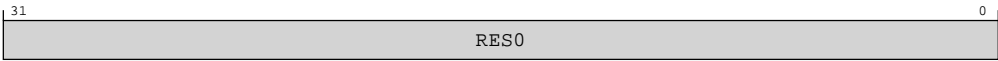


Table B-371: CLUSTERROM\_DBGPCR2 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.20 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

Bit descriptions  
When NUM\_CORES >= 4

Figure B-249: ext\_clusterrom\_dbgpcr3 bit assignments

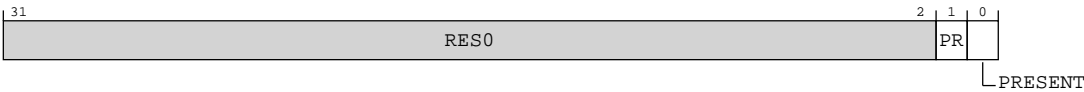


Table B-372: CLUSTERROM\_DBGPCR3 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 4

Figure B-250: ext\_clusterrom\_dbgpcr3 bit assignments

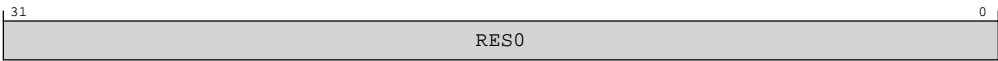


Table B-373: CLUSTERROM\_DBGPCR3 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility  
This interface is accessible as follows:  
  
RW

B.2.2.21 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

Bit descriptions

When NUM\_CORES >= 5

Figure B-251: ext\_clusterrom\_dbgpcr4 bit assignments

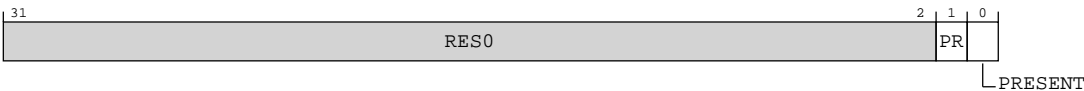


Table B-374: CLUSTERROM\_DBGPCR4 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 5



Figure B-252: ext\_clusterrom\_dbgpcr4 bit assignments

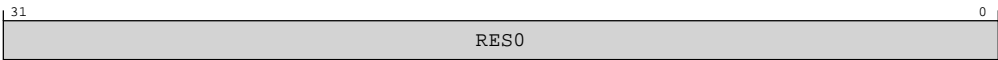


Table B-375: CLUSTERROM\_DBGPCR4 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.22 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

Bit descriptions

When NUM\_CORES >= 6

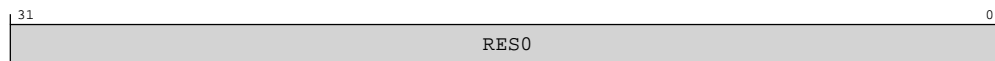
Figure B-253: ext\_clusterrom\_dbgpcr5 bit assignments



**Table B-376: CLUSTERROM\_DBGPCR5 bit descriptions**

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 6

**Figure B-254: ext\_clusterrom\_dbgpcr5 bit assignments****Table B-377: CLUSTERROM\_DBGPCR5 bit descriptions**

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

### Accessibility

This interface is accessible as follows:

RW

### B.2.2.23 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

Bit descriptions  
When NUM\_CORES >= 7

Figure B-255: ext\_clusterrom\_dbgpcr6 bit assignments

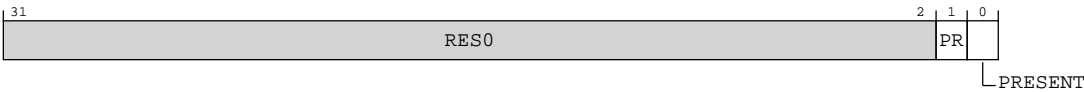


Table B-378: CLUSTERROM\_DBGPCR6 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 7

Figure B-256: ext\_clusterrom\_dbgpcr6 bit assignments

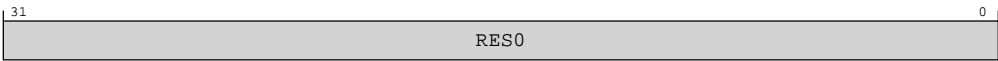


Table B-379: CLUSTERROM\_DBGPCR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility  
This interface is accessible as follows:  
  
RW

B.2.2.24 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

Bit descriptions

When NUM\_CORES >= 8

Figure B-257: ext\_clusterrom\_dbgpcr7 bit assignments

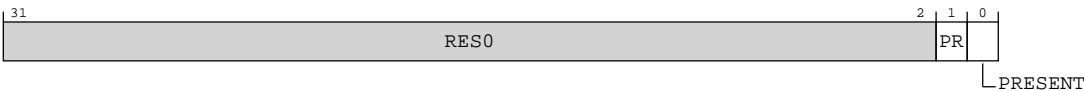


Table B-380: CLUSTERROM\_DBGPCR7 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 8

Figure B-258: ext\_clusterrom\_dbgpcr7 bit assignments

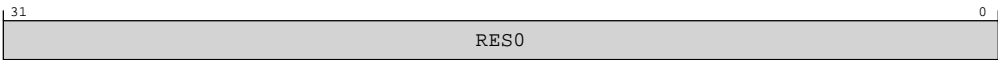


Table B-381: CLUSTERROM\_DBGPCR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.25 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

Bit descriptions

When NUM\_CORES >= 9

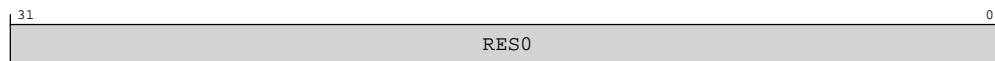
Figure B-259: ext\_clusterrom\_dbgpcr8 bit assignments



**Table B-382: CLUSTERROM\_DBGPCR8 bit descriptions**

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 9

**Figure B-260: ext\_clusterrom\_dbgpcr8 bit assignments****Table B-383: CLUSTERROM\_DBGPCR8 bit descriptions**

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

### Accessibility

This interface is accessible as follows:

RW

### B.2.2.26 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

Bit descriptions  
When NUM\_CORES >= 10

Figure B-261: ext\_clusterrom\_dbgpcr9 bit assignments

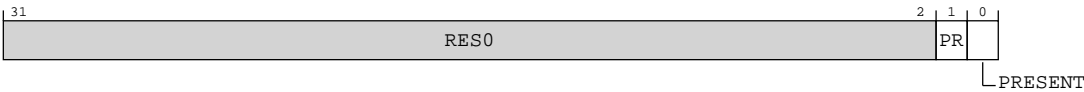


Table B-384: CLUSTERROM\_DBGPCR9 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  0b0 Power is not requested for target core or complex power domain.  0b1 Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  0b1 Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 10

Figure B-262: ext\_clusterrom\_dbgpcr9 bit assignments

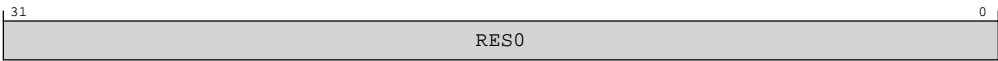


Table B-385: CLUSTERROM\_DBGPCR9 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility  
This interface is accessible as follows:  
  
RW

B.2.2.27 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

Bit descriptions

When NUM\_CORES >= 11

Figure B-263: ext\_clusterrom\_dbgpcr10 bit assignments

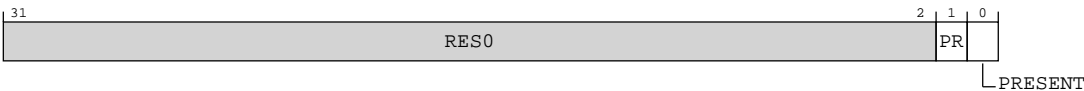


Table B-386: CLUSTERROM\_DBGPCR10 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  0b0 Power is not requested for target core or complex power domain.  0b1 Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  0b1 Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 11



Figure B-264: ext\_clusterrom\_dbgpcr10 bit assignments

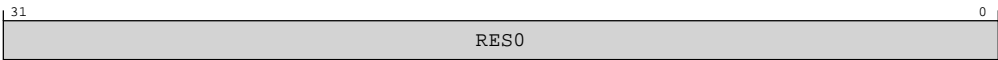


Table B-387: CLUSTERROM\_DBGPCR10 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.28 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

Bit descriptions

When NUM\_CORES >= 12

Figure B-265: ext\_clusterrom\_dbgpcr11 bit assignments



Table B-388: CLUSTERROM\_DBGPCR11 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 12

Figure B-266: ext\_clusterrom\_dbgpcr11 bit assignments

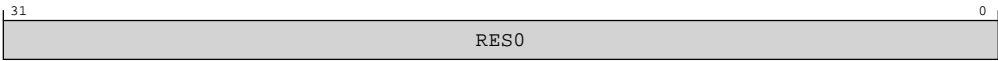


Table B-389: CLUSTERROM\_DBGPCR11 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.29 CLUSTERROM\_DBGPCR12, Cluster ROM table Debug Power Control Register 12

Controls power requests for PDCOMPLEX12.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA30

Access type

RO

Bit descriptions

When NUM\_CORES >= 13

Figure B-267: ext\_clusterrom\_dbgpcr12 bit assignments

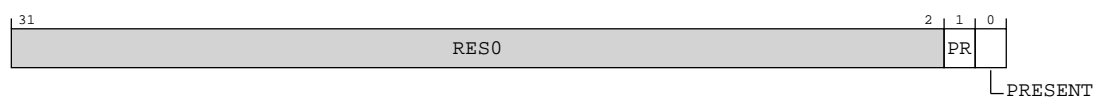


Table B-390: CLUSTERROM\_DBGPCR12 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 13

Figure B-268: ext\_clusterrom\_dbgpcr12 bit assignments

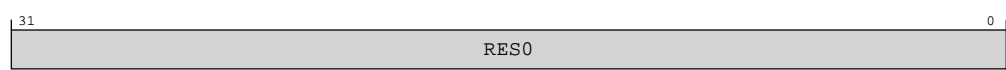


Table B-391: CLUSTERROM\_DBGPCR12 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.30 CLUSTERROM\_DBGPCR13, Cluster ROM table Debug Power Control Register 13

Controls power requests for PDCOMPLEX13.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA34

Access type

RO

Bit descriptions

When NUM\_CORES >= 14

Figure B-269: ext\_clusterrom\_dbgpcr13 bit assignments

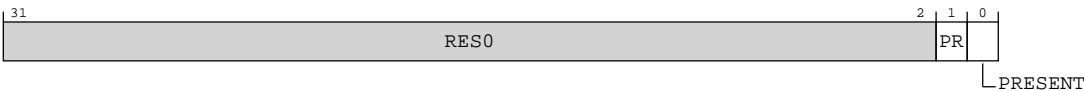


Table B-392: CLUSTERROM\_DBGPCR13 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  <b>0b0</b> Power is not requested for target core or complex power domain.  <b>0b1</b> Power is requested for target core or complex power domain.	x
[0]	PRESENT	Power request implemented.  <b>0b1</b> Power request for target core or complex power domain is included in the PPU power control.	x

When NUM\_CORES < 14

Figure B-270: ext\_clusterrom\_dbgpcr13 bit assignments

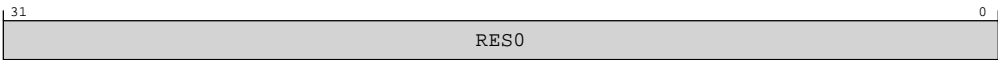


Table B-393: CLUSTERROM\_DBGPCR13 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RW

B.2.2.31 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

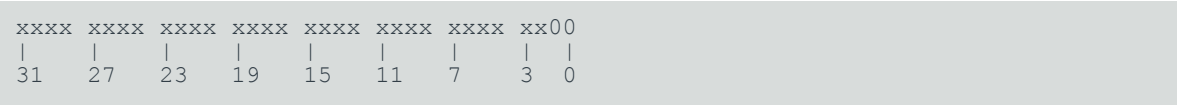
Register offset

0xA80

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-271: ext\_clusterrom\_dbgpsr0 bit assignments

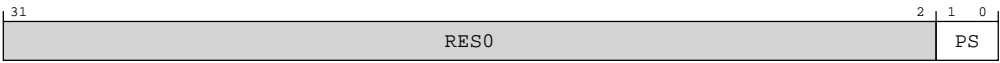


Table B-394: CLUSTERROM\_DBGPSR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

Accessibility

This interface is accessible as follows:

RO

B.2.2.32 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

Bit descriptions  
When NUM\_CORES >= 2

Figure B-272: ext\_clusterrom\_dbgpsr1 bit assignments

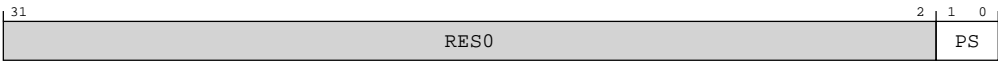


Table B-395: CLUSTERROM\_DBGPSR1 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  0b00 Target core or complex debug power domain might not be powered.  0b01 Target core or complex debug power domain is powered.  0b10 Reserved.  0b11 Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 2

Figure B-273: ext\_clusterrom\_dbgpsr1 bit assignments

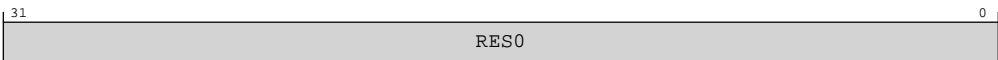


Table B-396: CLUSTERROM\_DBGPSR1 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility  
This interface is accessible as follows:  
  
RO

B.2.2.33 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

Bit descriptions

When NUM\_CORES >= 3

Figure B-274: ext\_clusterrom\_dbgpsr2 bit assignments

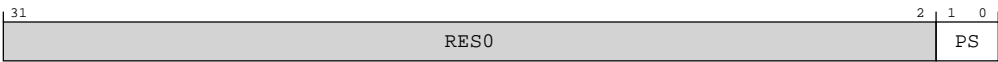


Table B-397: CLUSTERROM\_DBGPSR2 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  0b00 Target core or complex debug power domain might not be powered.  0b01 Target core or complex debug power domain is powered.  0b10 Reserved.  0b11 Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 3



Figure B-275: ext\_clusterrom\_dbgpsr2 bit assignments

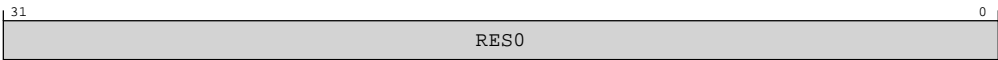


Table B-398: CLUSTERROM\_DBGPSR2 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.34 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

Bit descriptions

When NUM\_CORES >= 4

Figure B-276: ext\_clusterrom\_dbgpsr3 bit assignments

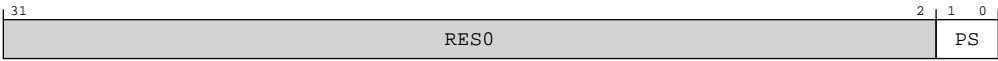


Table B-399: CLUSTERROM\_DBGPSR3 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 4

Figure B-277: ext\_clusterrom\_dbgpsr3 bit assignments

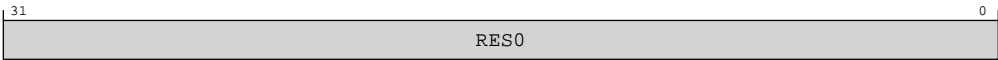


Table B-400: CLUSTERROM\_DBGPSR3 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.35 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

Bit descriptions  
When NUM\_CORES >= 5

Figure B-278: ext\_clusterrom\_dbgpsr4 bit assignments

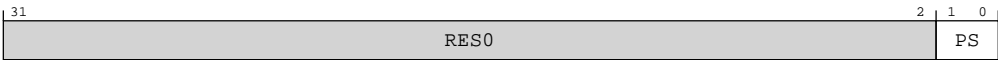


Table B-401: CLUSTERROM\_DBGPSR4 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  0b00 Target core or complex debug power domain might not be powered.  0b01 Target core or complex debug power domain is powered.  0b10 Reserved.  0b11 Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 5

Figure B-279: ext\_clusterrom\_dbgpsr4 bit assignments

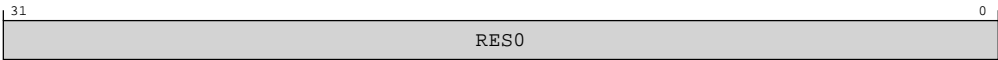


Table B-402: CLUSTERROM\_DBGPSR4 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility  
This interface is accessible as follows:  
  
RO

B.2.2.36 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

Bit descriptions

When NUM\_CORES >= 6

Figure B-280: ext\_clusterrom\_dbgpsr5 bit assignments

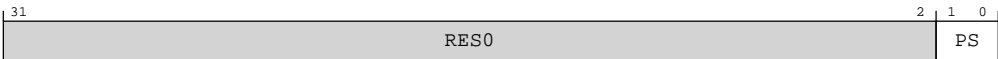


Table B-403: CLUSTERROM\_DBGPSR5 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 6

Figure B-281: ext\_clusterrom\_dbgpsr5 bit assignments

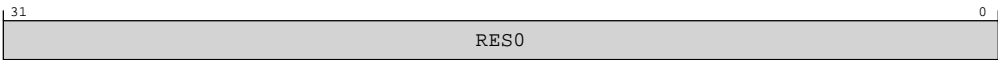


Table B-404: CLUSTERROM\_DBGPSR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.37 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

Bit descriptions

When NUM\_CORES >= 7

Figure B-282: ext\_clusterrom\_dbgpsr6 bit assignments

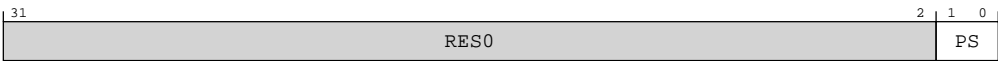


Table B-405: CLUSTERROM\_DBGPSR6 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 7

Figure B-283: ext\_clusterrom\_dbgpsr6 bit assignments

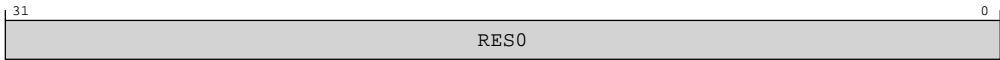


Table B-406: CLUSTERROM\_DBGPSR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.38 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

Bit descriptions

When NUM\_CORES >= 8

Figure B-284: ext\_clusterrom\_dbgpsr7 bit assignments

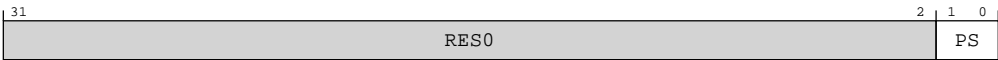


Table B-407: CLUSTERROM\_DBGPSR7 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 8

Figure B-285: ext\_clusterrom\_dbgpsr7 bit assignments

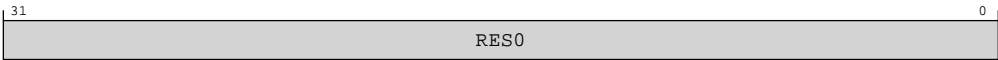


Table B-408: CLUSTERROM\_DBGPSR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.39 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

Bit descriptions

When NUM\_CORES >= 9

Figure B-286: ext\_clusterrom\_dbgpsr8 bit assignments

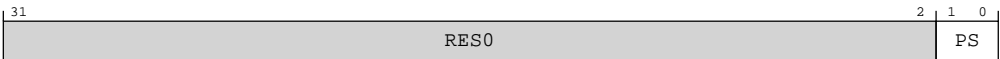


Table B-409: CLUSTERROM\_DBGPSR8 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 9



Figure B-287: ext\_clusterrom\_dbgpsr8 bit assignments

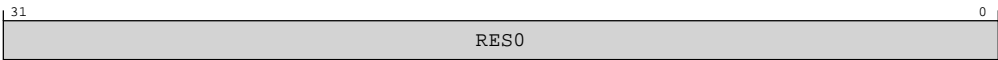


Table B-410: CLUSTERROM\_DBGPSR8 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.40 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

Bit descriptions

When NUM\_CORES >= 10

Figure B-288: ext\_clusterrom\_dbgpsr9 bit assignments

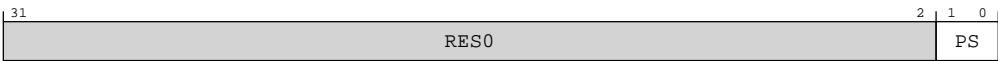


Table B-411: CLUSTERROM\_DBGPSR9 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 10

Figure B-289: ext\_clusterrom\_dbgpsr9 bit assignments

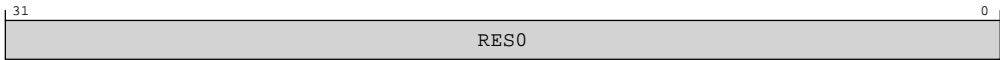


Table B-412: CLUSTERROM\_DBGPSR9 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.41 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

Bit descriptions

When NUM\_CORES >= 11

Figure B-290: ext\_clusterrom\_dbgpsr10 bit assignments

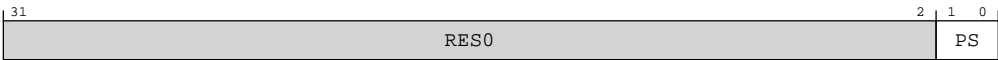


Table B-413: CLUSTERROM\_DBGPSR10 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  0b00 Target core or complex debug power domain might not be powered.  0b01 Target core or complex debug power domain is powered.  0b10 Reserved.  0b11 Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 11

Figure B-291: ext\_clusterrom\_dbgpsr10 bit assignments

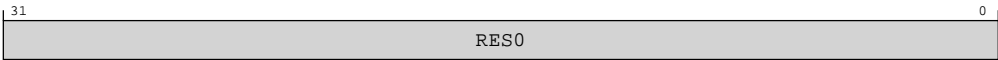


Table B-414: CLUSTERROM\_DBGPSR10 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.42 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

Bit descriptions

When NUM\_CORES >= 12

Figure B-292: ext\_clusterrom\_dbgpsr11 bit assignments

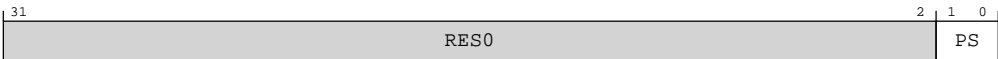


Table B-415: CLUSTERROM\_DBGPSR11 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 12

Figure B-293: ext\_clusterrom\_dbgpsr11 bit assignments

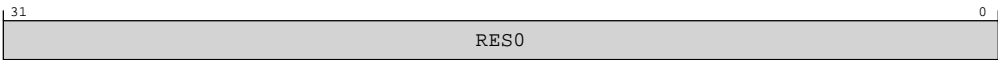


Table B-416: CLUSTERROM\_DBGPSR11 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.43 CLUSTERROM\_DBGPSR12, Cluster ROM table Debug Power Status Register 12

Indicates the power status for PDCOMPLEX12.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAB0

Access type

RO

Bit descriptions

When NUM\_CORES >= 13

Figure B-294: ext\_clusterrom\_dbgpsr12 bit assignments

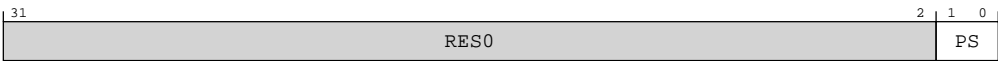


Table B-417: CLUSTERROM\_DBGPSR12 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 13

Figure B-295: ext\_clusterrom\_dbgpsr12 bit assignments

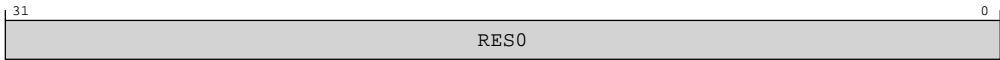


Table B-418: CLUSTERROM\_DBGPSR12 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.44 CLUSTERROM\_DBGPSR13, Cluster ROM table Debug Power Status Register 13

Indicates the power status for PDCOMPLEX13.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAB4

Access type

RO

Bit descriptions

When NUM\_CORES >= 14

Figure B-296: ext\_clusterrom\_dbgpsr13 bit assignments

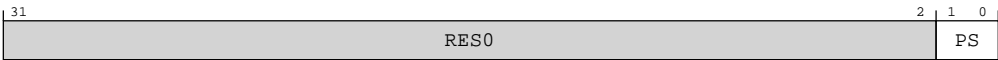


Table B-419: CLUSTERROM\_DBGPSR13 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  <b>0b00</b> Target core or complex debug power domain might not be powered.  <b>0b01</b> Target core or complex debug power domain is powered.  <b>0b10</b> Reserved.  <b>0b11</b> Target core or complex debug power domain is powered and must remain powered until DBGPCR0.PR is set to 0.	0b00

When NUM\_CORES < 14

Figure B-297: ext\_clusterrom\_dbgpsr13 bit assignments

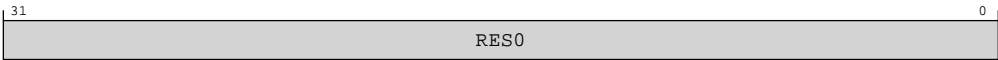


Table B-420: CLUSTERROM\_DBGPSR13 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.2.45 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-298: ext\_clusterrom\_pridr0 bit assignments

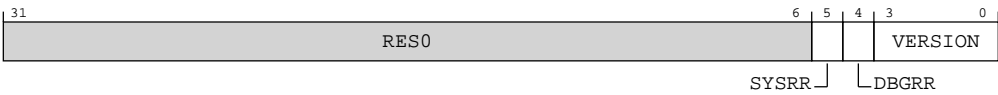


Table B-421: CLUSTERROM\_PRIDR0 bit descriptions

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



Bits	Name	Description	Reset
[4]	DBGRR	Debug reset request functionality present.  <b>0b0</b> The debug reset request functionality is not implemented.	0b0
[3:0]	VERSION	Version of the power request functionality.  <b>0b0001</b> The power request functionality version 0, and the per-core controls for power requests (e.g. ext-CLUSTERROM_DBGPCRO and ext-CLUSTERROM_DBGPSRO), are implemented.	0b0001

Accessibility

This interface is accessible as follows:

RO

B.2.2.46 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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-299: ext\_clusterrom\_authstatus bit assignments

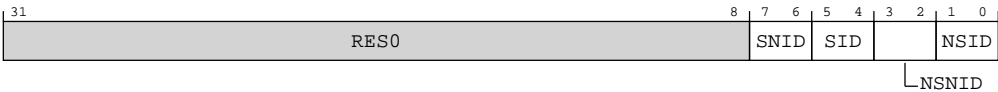


Table B-422: CLUSTERROM\_AUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:6]	SNID	Secure Non-invasive Debug.  <b>0b00</b> Debug level is not supported.  ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled().  This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug.  <b>0b00</b> Debug level is not supported.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug.  <b>0b11</b> Supported and enabled.	0b11
[1:0]	NSID	Non-secure Invasive Debug.  <b>0b00</b> Debug level is not supported.	0b00

Accessibility

This interface is accessible as follows:

RO

B.2.2.47 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-300: ext\_clusterrom\_devarch bit assignments

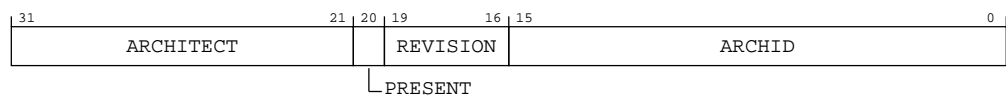


Table B-423: CLUSTERROM\_DEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. <b>0b01000111011</b> JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. <b>0b1</b> DEVARCH information present.	0b1
[19:16]	REVISION	Revision. <b>0b0000</b> Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. <b>0b0000101011110111</b> ROM Table v0. The debug tool must inspect ext-CLUSTERROM_DEVTYPE and ext-CLUSTERROM_DEVID to determine further information about the ROM Table.	0x0AF7

Accessibility

This interface is accessible as follows:

RO

B.2.2.48 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-301: ext\_clusterrom\_devid bit assignments

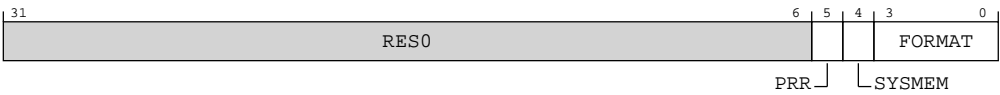


Table B-424: CLUSTERROM\_DEVID bit descriptions

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]	SYMMEM	System memory present.  0b0 System memory is not present on the bus.	0b0

Bits	Name	Description	Reset
[3:0]	FORMAT	ROM format.  <b>0b0000</b> 32-bit format 0.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.2.49 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-302: ext\_clusterrom\_devtype bit assignments

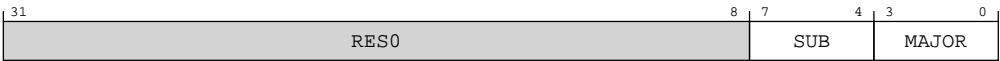


Table B-425: CLUSTERROM\_DEVTYPE bit descriptions

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

Accessibility

This interface is accessible as follows:

RO

B.2.2.50 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





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-303: ext\_clusterrom\_pidr4 bit assignments

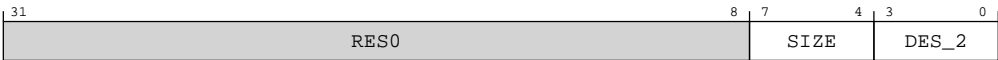


Table B-426: CLUSTERROM\_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. <b>0b0000</b> The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. <b>0b0100</b> Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.2.2.51 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	1100
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-304: ext\_clusterrom\_pidr0 bit assignments



Table B-427: CLUSTERROM\_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. <b>0b11101100</b> DSU-120AE Cluster ROM table. Bits [7:0] of part number 0x4EC.	0xEC

Accessibility

This interface is accessible as follows:

RO

B.2.2.52 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

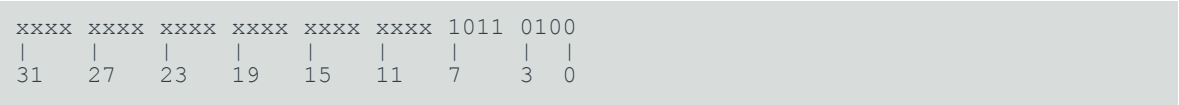
Register offset

0xFE4

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-305: ext\_clusterrom\_pidr1 bit assignments

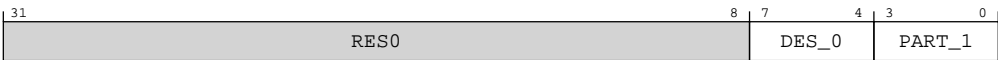


Table B-428: CLUSTERROM\_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0]. <b>0b1011</b> Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. <b>0b0100</b> DSU-120AE Cluster ROM table. Bits [11:8] of part number 0x4EC.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.2.2.53 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

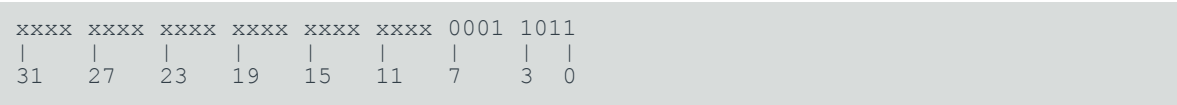
Register offset

0xFE8

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-306: ext\_clusterrom\_pidr2 bit assignments

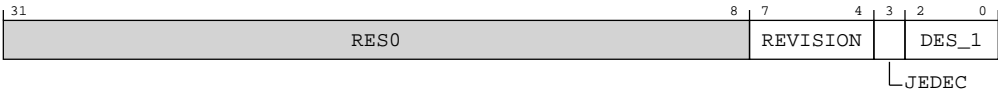


Table B-429: CLUSTERROM\_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVISION	Component major revision.  <b>0b0000</b> Component major revision 0.  <b>0b0001</b> Component major revision 1.  For DSU-120AE: <ul style="list-style-type: none"><li>Major revision 0 corresponds to r0p0.</li><li>Major revision 1 corresponds to r0p1.</li></ul>	0b0001
[3]	JEDEC	JEDEC assignee.  <b>0b1</b> JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4].  <b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

This interface is accessible as follows:

RO

B.2.2.54 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



312723191511730



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-307: ext\_clusterrom\_pidr3 bit assignments

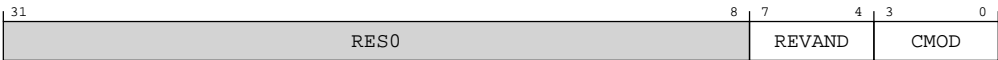


Table B-430: CLUSTERROM\_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVAND	Component minor revision. <b>0b0000</b> Component minor revision 0.	0b0000
[3:0]	CMOD	Customer Modified. <b>0b0000</b> The component is not modified from the original design.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.2.55 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-308: ext\_clusterrom\_cidr0 bit assignments



Table B-431: CLUSTERROM\_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	CoreSight component identification preamble. <b>0b00001101</b> CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

B.2.2.56 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-309: ext\_clusterrom\_cidr1 bit assignments

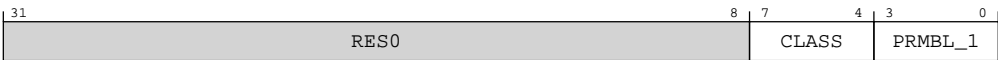


Table B-432: CLUSTERROM\_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	CLASS	CoreSight component class. <b>0b1001</b> CoreSight component.	0b1001
[3:0]	PRMBL_1	CoreSight component identification preamble. <b>0b0000</b> CoreSight component identification preamble.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.2.57 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-310: ext\_clusterrom\_cidr2 bit assignments

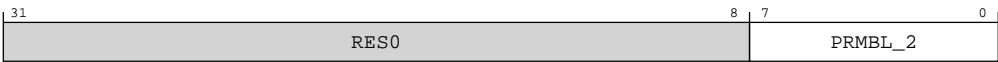


Table B-433: CLUSTERROM\_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	CoreSight component identification preamble. <b>0b00000101</b> CoreSight component identification preamble.	0x05

Accessibility

This interface is accessible as follows:

RO

B.2.2.58 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-311: ext\_clusterrom\_cidr3 bit assignments



Table B-434: 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



## Accessibility

This interface is accessible as follows:

RO

### B.2.3 External debug ROM registers summary

The debug ROM table registers are only accessible using memory-mapped accesses over the debug APB interface.

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 debug ROM table register values are based on a cluster, where DSU-120AE NUM\_CORES parameter is set to 14.
- 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**.
- The number of registers that contain valid entries depends on the number of cores configured for the cluster.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

**Table B-435: DBROM registers summary**

Offset	Name	Reset	Width	Description
0x000	<a href="#">DBROM_ROMENTRY0</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 0
0x004	<a href="#">DBROM_ROMENTRY1</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 1
0x008	<a href="#">DBROM_ROMENTRY2</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 2
0x00C	<a href="#">DBROM_ROMENTRY3</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 3
0x010	<a href="#">DBROM_ROMENTRY4</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 4
0x014	<a href="#">DBROM_ROMENTRY5</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 5
0x018	<a href="#">DBROM_ROMENTRY6</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 6
0x01C	<a href="#">DBROM_ROMENTRY7</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 7
0x020	<a href="#">DBROM_ROMENTRY8</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 8
0x024	<a href="#">DBROM_ROMENTRY9</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 9
0x028	<a href="#">DBROM_ROMENTRY10</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 10
0x02C	<a href="#">DBROM_ROMENTRY11</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 11
0x030	<a href="#">DBROM_ROMENTRY12</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 12
0x034	<a href="#">DBROM_ROMENTRY13</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 13
0x038	<a href="#">DBROM_ROMENTRY14</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 14
0x03C	<a href="#">DBROM_ROMENTRY15</a>	See individual bit resets.	32-bit	DebugBlock ROM table Entry 15

Offset	Name	Reset	Width	Description
0xA00	<a href="#">DBROM_DBGPCRO</a>	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Control Register 0
0xA80	<a href="#">DBROM_DBGPSRO</a>	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Status Register 0
0xC00	<a href="#">DBROM_PRIDRO</a>	See individual bit resets.	32-bit	DebugBlock ROM table Power Request ID Register 0
0xFB8	<a href="#">DBROM_AUTHSTATUS</a>	See individual bit resets.	32-bit	DebugBlock ROM table Authentication Status Register
0xFBC	<a href="#">DBROM_DEVARCH</a>	See individual bit resets.	32-bit	DebugBlock ROM table Device Architecture Register
0xFC8	<a href="#">DBROM_DEVID</a>	See individual bit resets.	32-bit	DebugBlock ROM table Device Configuration Register
0xFCC	<a href="#">DBROM_DEVTYPE</a>	See individual bit resets.	32-bit	DebugBlock ROM table Device Type Register
0xFD0	<a href="#">DBROM_PIDR4</a>	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 4
0xFE0	<a href="#">DBROM_PIDR0</a>	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 0
0xFE4	<a href="#">DBROM_PIDR1</a>	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 1
0xFE8	<a href="#">DBROM_PIDR2</a>	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 2
0xFEC	<a href="#">DBROM_PIDR3</a>	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 3
0xFF0	<a href="#">DBROM_CIDR0</a>	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 0
0xFF4	<a href="#">DBROM_CIDR1</a>	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 1
0xFF8	<a href="#">DBROM_CIDR2</a>	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 2
0xFFC	<a href="#">DBROM_CIDR3</a>	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 3

### 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
|    |    |    |    |    |    |    |
31   27   23   19   15   11   7     3   0

```



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-312: ext\_dbrom\_romentry0 bit assignments

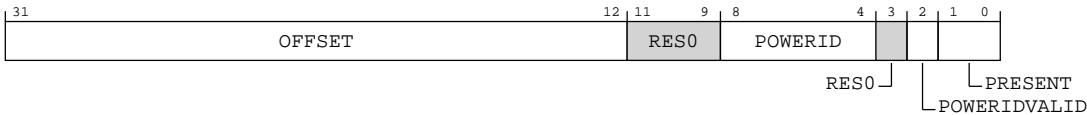


Table B-436: DBROM\_ROMENTRY0 bit descriptions

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).  <b>0b000000000000011000000</b> Cluster ROM table at address 0xC_0000.	0x000C0
[11:9]	RES0	Reserved	RES0
[8:4]	POWERID	The power domain ID of the component.  <b>0b00000</b> Cluster power domain.	0b00000
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b1</b> 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.  <b>0b11</b> The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-313: ext\_dbrom\_romentry1 bit assignments

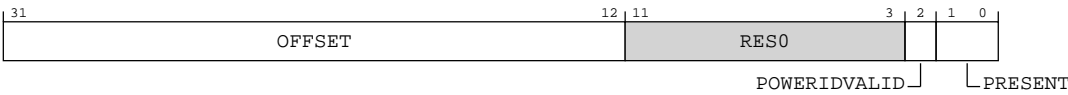


Table B-437: DBROM\_ROMENTRY1 bit descriptions

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).  <b>0b000000000000011110000</b> Cluster CTI at address 0xF_0000.	0x00F0
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

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	0000	1111	0000	xxxx	xxxx	x011
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-314: ext\_dbrom\_romentry2 bit assignments

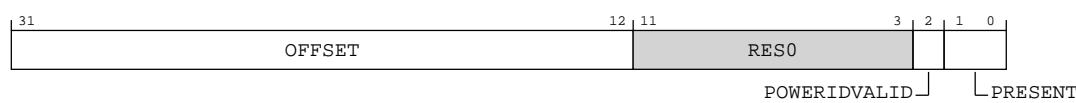


Table B-438: DBROM\_ROMENTRY2 bit descriptions

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).  <b>0b000000000000011110000</b> Core 0 CTI at address 0xF_0000.	0x000F0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 2

Figure B-315: ext\_dbrom\_romentry3 bit assignments

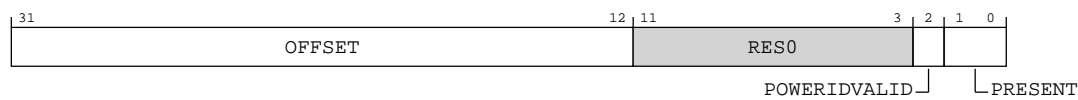


Table B-439: DBROM\_ROMENTRY3 bit descriptions

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).  <b>0b000000000000101110000</b> Core 1 CTI at address 0x17_0000.	0x00170
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 2

Figure B-316: ext\_dbrom\_romentry3 bit assignments



Table B-440: DBROM\_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility  
This interface is accessible as follows:  
  
RO

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

Bit descriptions

When NUM\_CORES >= 3

Figure B-317: ext\_dbrom\_romentry4 bit assignments

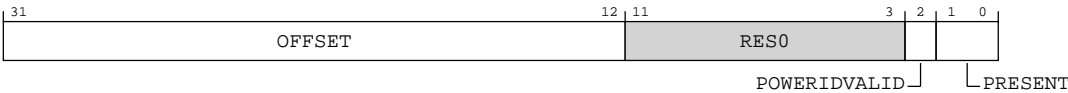


Table B-441: DBROM\_ROMENTRY4 bit descriptions

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).  <b>0b000000000000111110000</b> Core 2 CTI at address 0x1F_0000.	0x001F0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 3



Figure B-318: ext\_dbrom\_romentry4 bit assignments

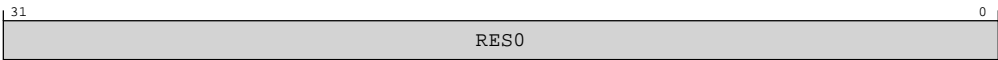


Table B-442: DBROM\_ROMENTRY4 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

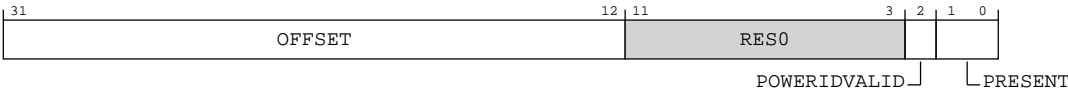
Access type

RO

Bit descriptions

When NUM\_CORES >= 4

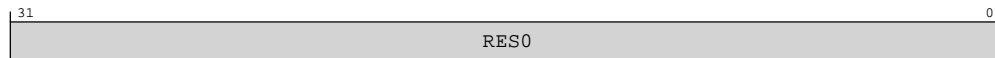
Figure B-319: ext\_dbrom\_romentry5 bit assignments



**Table B-443: 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).  <b>0b00000000001001110000</b> Core 3 CTI at address 0x27_0000.	0x00270
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 4

**Figure B-320: ext\_dbrom\_romentry5 bit assignments****Table B-444: DBROM\_ROMENTRY5 bit descriptions**

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

### Accessibility

This interface is accessible as follows:

RO

### 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

Bit descriptions

When NUM\_CORES >= 5

Figure B-321: ext\_dbrom\_romentry6 bit assignments



Table B-445: 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).  <b>0b00000000001011110000</b> Core 4 CTI at address 0x2F_0000.	0x002F0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 5

Figure B-322: ext\_dbrom\_romentry6 bit assignments

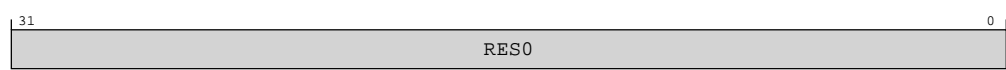


Table B-446: DBROM\_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions

When NUM\_CORES >= 6

Figure B-323: ext\_dbrom\_romentry7 bit assignments

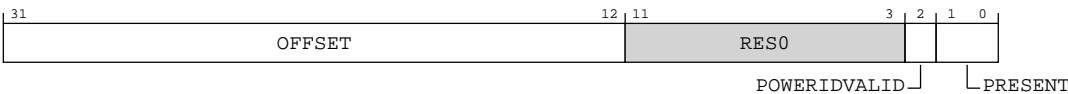


Table B-447: DBROM\_ROMENTRY7 bit descriptions

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).  <b>0b000000000001101110000</b> Core 5 CTI at address 0x37_0000.	0x00370
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 6

Figure B-324: ext\_dbrom\_romentry7 bit assignments



Table B-448: DBROM\_ROMENTRY7 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions

When NUM\_CORES >= 7

Figure B-325: ext\_dbrom\_romentry8 bit assignments

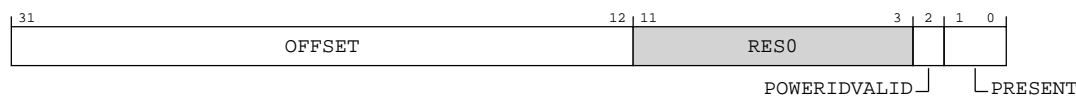


Table B-449: DBROM\_ROMENTRY8 bit descriptions

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).  <b>0b00000000001111110000</b> Core 6 CTI at address 0x3F_0000.	0x003F0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 7

Figure B-326: ext\_dbrom\_romentry8 bit assignments



Table B-450: DBROM\_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions  
When NUM\_CORES >= 8

Figure B-327: ext\_dbrom\_romentry9 bit assignments

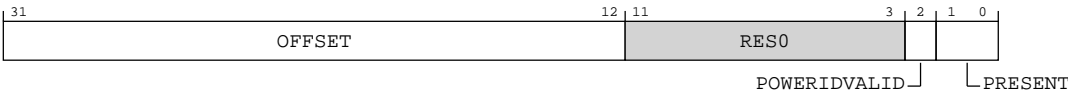
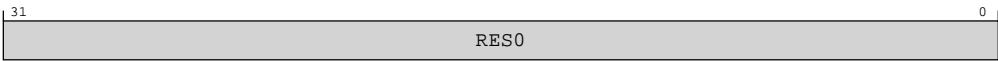


Table B-451: DBROM\_ROMENTRY9 bit descriptions

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).  <b>0b00000000010001110000</b> Core 7 CTI at address 0x47_0000.	0x00470
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 8

Figure B-328: ext\_dbrom\_romentry9 bit assignments



### Table B-452: DBROM\_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

## Accessibility

This interface is accessible as follows:

RO

#### 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

## Bit descriptions

When NUM\_CORES  $\geq 9$

**Figure B-329: ext\_dbrom\_romentry10 bit assignments**



### Table B-453: DBROM\_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	<p>The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation:</p> <p>Component Address = ROM Table Base Address + (OFFSET &lt;&lt; 12).</p> <p><b>0b00000000010011110000</b></p> <p>Core 8 CTI at address 0x4F_0000.</p>	0x004F0



Bits	Name	Description	Reset
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 9

Figure B-330: ext\_dbrom\_romentry10 bit assignments

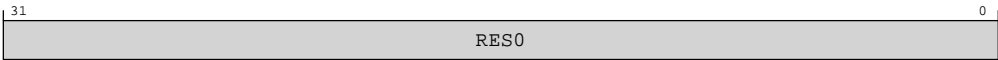


Table B-454: DBROM\_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions

When NUM\_CORES >= 10

Figure B-331: ext\_dbrom\_romentry11 bit assignments

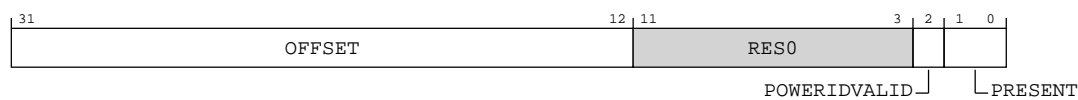


Table B-455: DBROM\_ROMENTRY11 bit descriptions

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).  <b>0b00000000010101110000</b> Core 9 CTI at address 0x57_0000.	0x00570
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 10

Figure B-332: ext\_dbrom\_romentry11 bit assignments

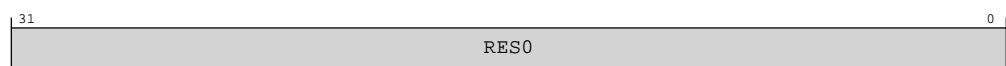


Table B-456: DBROM\_ROMENTRY11 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

Bit descriptions

When NUM\_CORES >= 11

Figure B-333: ext\_dbrom\_romentry12 bit assignments

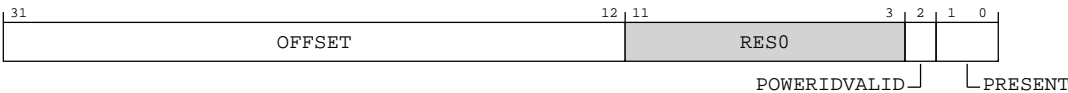


Table B-457: DBROM\_ROMENTRY12 bit descriptions

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).  <b>0b00000000010111110000</b> Core 10 CTI at address 0x5F_0000.	0x005F0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 11

Figure B-334: ext\_dbrom\_romentry12 bit assignments

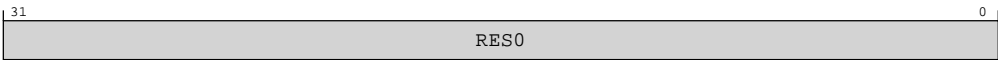


Table B-458: DBROM\_ROMENTRY12 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

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

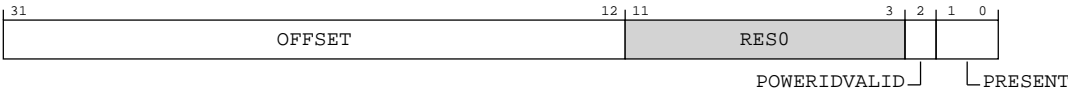
Access type

RO

Bit descriptions

When NUM\_CORES >= 12

Figure B-335: ext\_dbrom\_romentry13 bit assignments



**Table B-459: 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).  <b>0b00000000011001110000</b> Core 11 CTI at address 0x67_0000.	0x00670
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 12

**Figure B-336: ext\_dbrom\_romentry13 bit assignments**



**Table B-460: DBROM\_ROMENTRY13 bit descriptions**

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

### Accessibility

This interface is accessible as follows:

RO

### B.2.3.15 DBROM\_ROMENTRY14, DebugBlock ROM table Entry 14

Provides the address offset for one CoreSight component.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32

Component  
DBROM

Register offset  
0x038

Access type  
RO

Bit descriptions

When NUM\_CORES >= 13

Figure B-337: ext\_dbrom\_romentry14 bit assignments

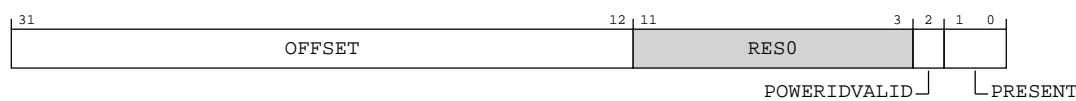


Table B-461: DBROM\_ROMENTRY14 bit descriptions

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).  <b>0b00000000011011110000</b> Core 12 CTI at address 0x6F_0000.	0x006F0
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 13

Figure B-338: ext\_dbrom\_romentry14 bit assignments

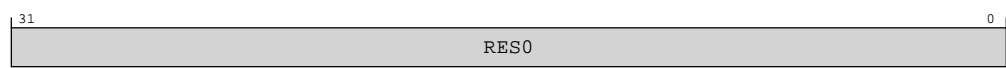


Table B-462: DBROM\_ROMENTRY14 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.3.16 DBROM\_ROMENTRY15, DebugBlock ROM table Entry 15

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x03C

Access type

RO

Bit descriptions

When NUM\_CORES >= 14

Figure B-339: ext\_dbrom\_romentry15 bit assignments

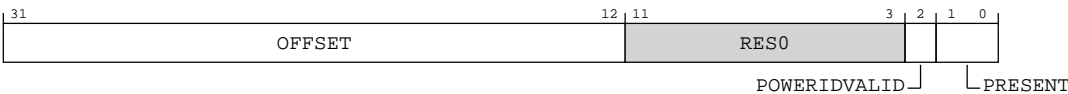


Table B-463: DBROM\_ROMENTRY15 bit descriptions

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).  <b>0b00000000011101110000</b> Core 13 CTI at address 0x77_0000.	0x00770
[11:3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID.  <b>0b0</b> A power domain ID is not provided.	0b0

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table.  <b>0b11</b> The ROM Entry is present.	0b11

When NUM\_CORES < 14

Figure B-340: ext\_dbrom\_romentry15 bit assignments



Table B-464: DBROM\_ROMENTRY15 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.2.3.17 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

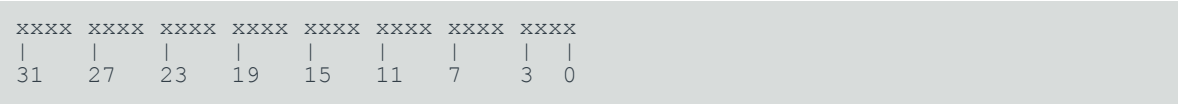
Register offset

0xA00

Access type

RO

Reset value







Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-341: ext\_dbrom\_dbgpcr0 bit assignments

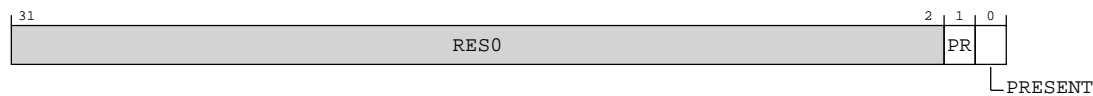


Table B-465: DBROM\_DBGPCR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request.  0b0 Power is not requested for Cluster power domain.  0b1 Power is requested for Cluster power domain.	x
[0]	PRESENT	Power request implemented.  0b1 Power request for PDCLUSTER is implemented.	x

Accessibility

This interface is accessible as follows:

RW

B.2.3.18 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-342: ext\_dbrom\_dbgpsr0 bit assignments

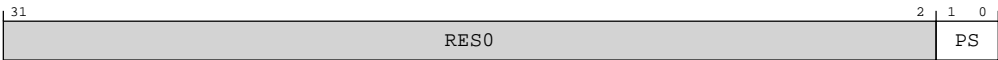


Table B-466: DBROM\_DBGPSR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status.  0b00 Cluster power domain might not be powered.  0b01 Cluster power domain is powered.	0b00

Accessibility

This interface is accessible as follows:

RO

B.2.3.19 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

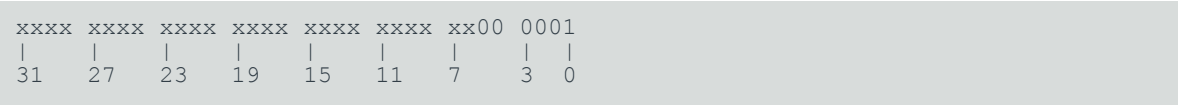
Register offset

0xC00

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-343: ext\_dbrom\_pridr0 bit assignments



Table B-467: DBROM\_PRIDR0 bit descriptions

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5]	SYSRR	System reset request functionality present. <b>0b0</b> The system reset request functionality is not implemented.	0b0
[4]	DBGRR	Debug reset request functionality present. <b>0b0</b> The debug reset request functionality is not implemented.	0b0
[3:0]	VERSION	Version of the power request functionality. <b>0b0001</b> The power request functionality version 0, and the ext-DBROM_DBGPCR0, ext-DBROM_DBGPSR0, which provide controls for power requests, are implemented.	0b0001

Accessibility

This interface is accessible as follows:

RO

B.2.3.20 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

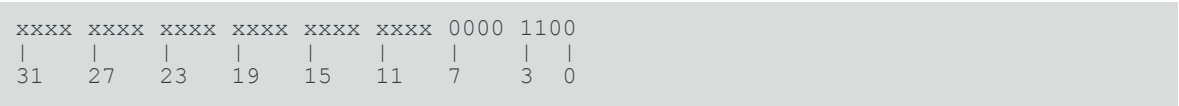
Register offset

0xFB8

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-344: ext\_dbrom\_authstatus bit assignments



Table B-468: DBROM\_AUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:6]	SNID	Secure Non-invasive Debug. <b>0b00</b> Debug level is not supported.  ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled().  This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug. <b>0b00</b> Debug level is not supported.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug. <b>0b11</b> Supported and enabled.	0b11
[1:0]	NSID	Non-secure Invasive Debug. <b>0b00</b> Debug level is not supported.	0b00

## Accessibility

This interface is accessible as follows:

RO

### B.2.3.21 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-345: ext\_dbrom\_devarch bit assignments

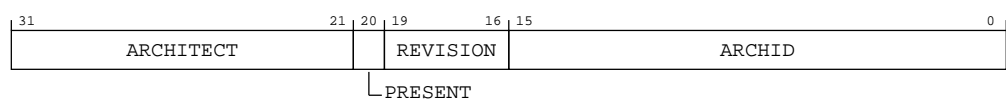


Table B-469: DBROM\_DEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. <b>0b01000111011</b> JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. <b>0b1</b> DEVARCH information present.	0b1
[19:16]	REVISION	Revision. <b>0b0000</b> Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. <b>0b0000101011110111</b> ROM Table v0. The debug tool must inspect ext-DBROM_DEVTYPE and ext-DBROM_DEVID to determine further information about the ROM Table.	0x0AF7

Accessibility

This interface is accessible as follows:

RO

B.2.3.22 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

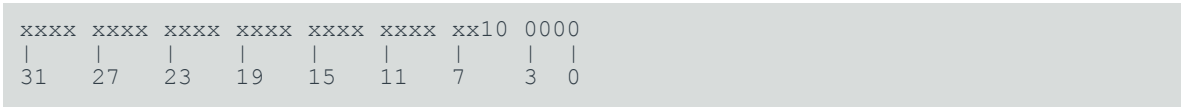
DBROM

Register offset

0xFC8

Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-346: ext\_dbrom\_devid bit assignments



Table B-470: DBROM\_DEVID bit descriptions

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5]	PRR	Power Request functionality included. <b>0b1</b> Power Request functionality included. ext-DBROM_PRIDR0 is implemented.	0b1
[4]	SYSMEM	System memory present. <b>0b0</b> System memory is not present on the bus.	0b0
[3:0]	FORMAT	ROM format. <b>0b0000</b> 32-bit format 0.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.3.23 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-347: ext\_dbrom\_devtype bit assignments



Table B-471: DBROM\_DEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SUB	Sub number <b>0b0000</b> Other, undefined.	0b0000
[3:0]	MAJOR	Major number <b>0b0000</b> Miscellaneous.	0b0000



Accessibility

This interface is accessible as follows:

RO

B.2.3.24 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-348: ext\_dbrom\_pidr4 bit assignments

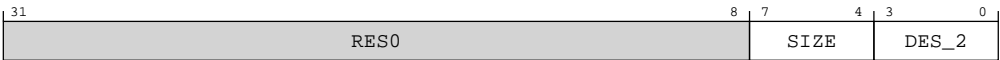


Table B-472: DBROM\_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	SIZE	4KB count.  <b>0b0000</b> The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code.  <b>0b0100</b> Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.2.3.25 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	1011
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-349: ext\_dbrom\_pidr0 bit assignments

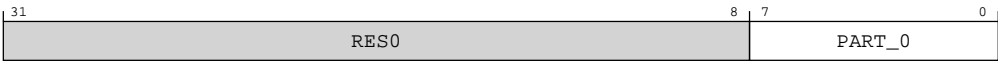


Table B-473: DBROM\_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0].  0b11101011 DSU-120AE DebugBlock ROM table. Bits [7:0] of part number 0x4EB.	0xEB

Accessibility

This interface is accessible as follows:

RO

B.2.3.26 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-350: ext\_dbrom\_pidr1 bit assignments

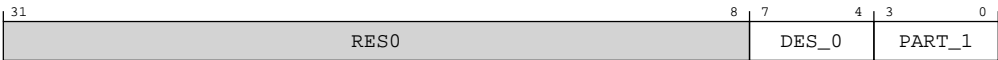


Table B-474: DBROM\_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0].  <b>0b1011</b> Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8].  <b>0b0100</b> DSU-120AE DebugBlock ROM table. Bits [11:8] of part number 0x4EB.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.2.3.27 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-351: ext\_dbrom\_pidr2 bit assignments



Table B-475: DBROM\_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVISION	Component major revision.  <b>0b0000</b> Component major revision 0.  <b>0b0001</b> Component major revision 1.  For DSU-120AE: <ul style="list-style-type: none"><li>Major revision 0 corresponds to r0p0.</li><li>Major revision 1 corresponds to r0p1.</li></ul>	0b0001
[3]	JEDEC	JEDEC assignee.  <b>0b1</b> JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4].  <b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

This interface is accessible as follows:

RO

B.2.3.28 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

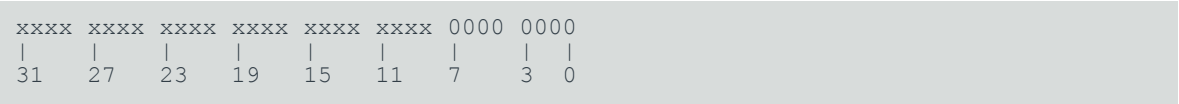
Register offset

0xFEC

Access type

RO

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-352: ext\_dbrom\_pidr3 bit assignments

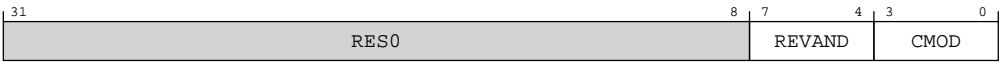


Table B-476: DBROM\_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVAND	Component minor revision.  0b0000 Component minor revision 0.	0b0000

Bits	Name	Description	Reset
[3:0]	CMOD	Customer Modified.  <b>0b0000</b> The component is not modified from the original design.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.3.29 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
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-353: ext\_dbrom\_cidr0 bit assignments

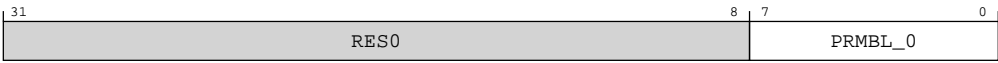


Table B-477: DBROM\_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	CoreSight component identification preamble. <b>0b00001101</b> CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

B.2.3.30 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







Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-354: ext\_dbrom\_cidr1 bit assignments

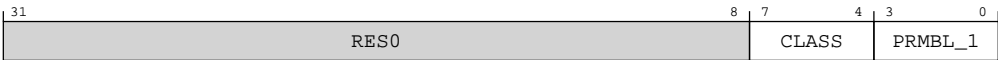


Table B-478: DBROM\_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	CLASS	CoreSight component class. <b>0b1001</b> CoreSight component.	0b1001
[3:0]	PRMBL_1	CoreSight component identification preamble. <b>0b0000</b> CoreSight component identification preamble.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.3.31 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

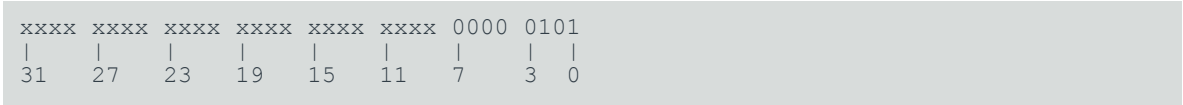
DBROM

Register offset

0xFF8

Access type  
RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-355: ext\_dbrom\_cidr2 bit assignments



Table B-479: DBROM\_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	CoreSight component identification preamble. <b>0b000000101</b> CoreSight component identification preamble.	0x05

Accessibility

This interface is accessible as follows:

RO

B.2.3.32 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

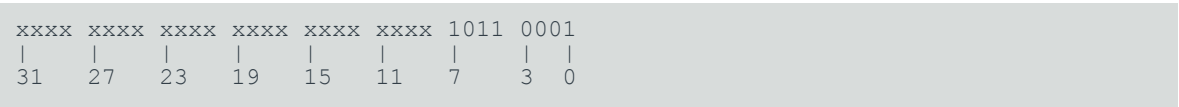
Register offset

0xFFC

Access type

RO

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-356: ext\_dbrom\_cidr3 bit assignments

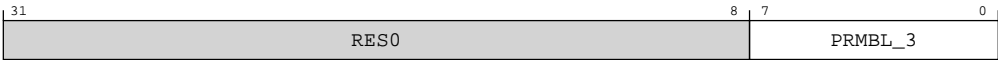


Table B-480: DBROM\_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_3	CoreSight component identification preamble. <b>0b10110001</b> CoreSight component identification preamble.	0xB1

Accessibility

This interface is accessible as follows:

RO

B.2.4 External cluster PMU registers summary

The cluster *Performance Monitoring Unit* (PMU) registers are accessible either from memory-mapped accesses over the debug APB interface or from System register accesses from the cores.

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**.
- 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-481: CLUSTERPMU registers summary**

Offset	Name	Reset	Width	Description
0x0	CLUSTERPMU_PMEVCNTR0	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x8	CLUSTERPMU_PMEVCNTR1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x10	CLUSTERPMU_PMEVCNTR2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x18	CLUSTERPMU_PMEVCNTR3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x20	CLUSTERPMU_PMEVCNTR4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x28	CLUSTERPMU_PMEVCNTR5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers
0x400	CLUSTERPMU_PMEVTYPE0	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x404	CLUSTERPMU_PMEVTYPE1	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x408	CLUSTERPMU_PMEVTYPE2	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x40C	CLUSTERPMU_PMEVTYPE3	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x410	CLUSTERPMU_PMEVTYPE4	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x414	CLUSTERPMU_PMEVTYPE5	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers
0x600	CLUSTERPMU_PMEVCNTRS0	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x608	CLUSTERPMU_PMEVCNTRS1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x610	CLUSTERPMU_PMEVCNTRS2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x618	CLUSTERPMU_PMEVCNTRS3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x620	CLUSTERPMU_PMEVCNTRS4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers
0x628	CLUSTERPMU_PMEVCNTRS5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers

Offset	Name	Reset	Width	Description
0x638	CLUSTERPMU_PMSSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Status register
0x640	CLUSTERPMU_PMOVSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Status Snapshot register
0xC00	CLUSTERPMU_PMCNTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Set register
0xC20	CLUSTERPMU_PMCNTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Clear register
0xC40	CLUSTERPMU_PMINTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Set register
0xC60	CLUSTERPMU_PMINTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Clear register
0xC80	CLUSTERPMU_PMOVSLR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Clear register
0xCC0	CLUSTERPMU_PMOVSET	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Set register
0xE00	CLUSTERPMU_PMCFGR	See individual bit resets.	32-bit	Cluster Performance Monitors Configuration Register
0xE04	CLUSTERPMU_PMCR	See individual bit resets.	32-bit	Cluster Performance Monitors Control Register
0xE08	CLUSTERPMU_PMIIDR	See individual bit resets.	32-bit	Cluster Performance Monitors Implementation Identification register
0xE20	CLUSTERPMU_PMCEID0	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 0
0xE24	CLUSTERPMU_PMCEID1	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 1
0xE28	CLUSTERPMU_PMCEID2	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 2
0xE2C	CLUSTERPMU_PMCEID3	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 3
0xE30	CLUSTERPMU_PMSSCR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Capture register
0xE38	CLUSTERPMU_PMSRR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Reset register
0xFA8	CLUSTERPMU_PMDEVAFF0	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 0
0xFAC	CLUSTERPMU_PMDEVAFF1	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 1
0xFB8	CLUSTERPMU_PMAUTHSTATUS	See individual bit resets.	32-bit	Cluster Performance Monitors Authentication Status register
0xFBC	CLUSTERPMU_PMDEVARCH	See individual bit resets.	32-bit	Cluster Performance Monitors Device Architecture register
0xFC8	CLUSTERPMU_PMDEVID	See individual bit resets.	32-bit	Cluster Performance Monitors Device ID register
0xFCC	CLUSTERPMU_PMDEVTYPE	See individual bit resets.	32-bit	Cluster Performance Monitors Device Type register

Offset	Name	Reset	Width	Description
0xFD0	<a href="#">CLUSTERPMU_PMPIDR4</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 4
0xFE0	<a href="#">CLUSTERPMU_PMPIDR0</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 0
0xFE4	<a href="#">CLUSTERPMU_PMPIDR1</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 1
0xFE8	<a href="#">CLUSTERPMU_PMPIDR2</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 2
0xFEC	<a href="#">CLUSTERPMU_PMPIDR3</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 3
0xFF0	<a href="#">CLUSTERPMU_PMCIDR0</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 0
0xFF4	<a href="#">CLUSTERPMU_PMCIDR1</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 1
0xFF8	<a href="#">CLUSTERPMU_PMCIDR2</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 2
0xFFC	<a href="#">CLUSTERPMU_PMCIDR3</a>	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 3

### B.2.4.1 CLUSTERPMU\_PMEVCNTR0, Cluster Performance Monitors Event Count Registers

Holds event counter 0, which counts events.

#### Configurations

External register CLUSTERPMU\_PMEVCNTR0 bits [63:0] are architecturally mapped to AArch64 System register [A.2.11 IMP\\_CLUSTERPMXEVCNTR\\_EL1, Performance Monitors Selected Event Count Register](#) on page 343 bits [63:0].

#### 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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-357: ext\_clusterpmu\_pmevcntr0 bit assignments

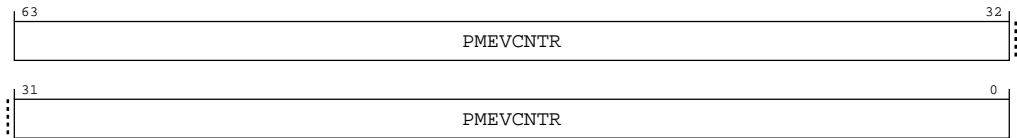


Table B-482: CLUSTERPMU\_PMEVCNTR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 0.	64 { x }

Accessibility

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

External register CLUSTERPMU\_PMEVCNTR1 bits [63:0] are architecturally mapped to AArch64 System register [A.2.11 IMP\\_CLUSTERPMXEVCNTR\\_EL1, Performance Monitors Selected Event Count Register](#) on page 343 bits [63:0].

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	
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3	0



Where the reset reads xxxx, see individual bits.

**Bit descriptions**

**Figure B-358: ext\_clusterpmu\_pmevcntr1 bit assignments**

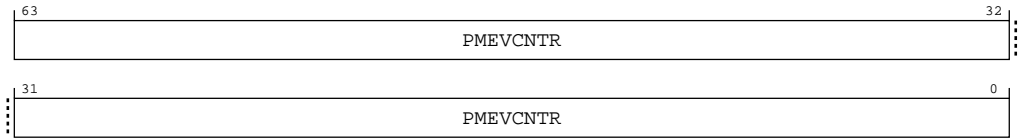


Table B-483: CLUSTERPMU\_PMEVCNTR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 1.	64 { x }

**Accessibility**

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

External register CLUSTERPMU\_PMEVCNTR2 bits [63:0] are architecturally mapped to AArch64 System register [A.2.11 IMP\\_CLUSTERPMXEVCNTR\\_EL1, Performance Monitors Selected Event Count Register](#) on page 343 bits [63:0].

Attributes

Width

64

Component

CLUSTERPMU

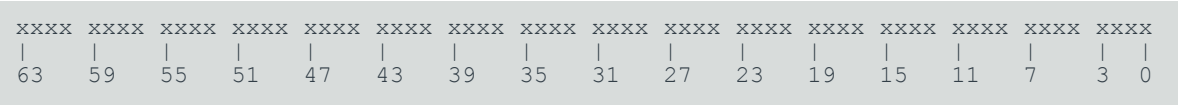
Register offset

0x10

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-359: ext\_clusterpmu\_pmevcntr2 bit assignments

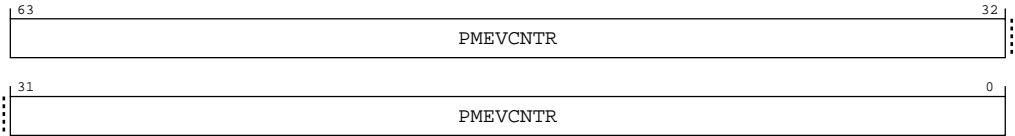


Table B-484: CLUSTERPMU\_PMEVCNTR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 2.	64 {x}

Accessibility

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

External register CLUSTERPMU\_PMEVCNTR3 bits [63:0] are architecturally mapped to AArch64 System register [A.2.11 IMP\\_CLUSTERPMXEVCNTR\\_EL1, Performance Monitors Selected Event Count Register](#) on page 343 bits [63:0].

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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-360: ext\_clusterpmu\_pmevcntr3 bit assignments

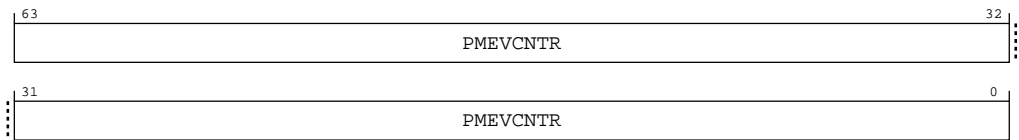


Table B-485: CLUSTERPMU\_PMEVCNTR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 3.	64 {x}

Accessibility

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

External register CLUSTERPMU\_PMEVCNTR4 bits [63:0] are architecturally mapped to AArch64 System register [A.2.11 IMP\\_CLUSTERPMXEVCNTR\\_EL1, Performance Monitors Selected Event Count Register](#) on page 343 bits [63:0].

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
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-361: ext\_clusterpmu\_pmevcntr4 bit assignments

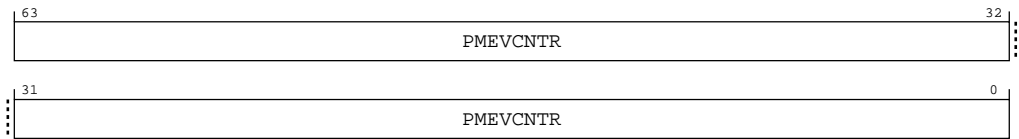


Table B-486: CLUSTERPMU\_PMEVCNTR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 4.	64 { x }

Accessibility

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

External register CLUSTERPMU\_PMEVCNTR5 bits [63:0] are architecturally mapped to AArch64 System register [A.2.11 IMP\\_CLUSTERPMXEVCNTR\\_EL1, Performance Monitors Selected Event Count Register](#) on page 343 bits [63:0].

Attributes

Width

64

Component

CLUSTERPMU

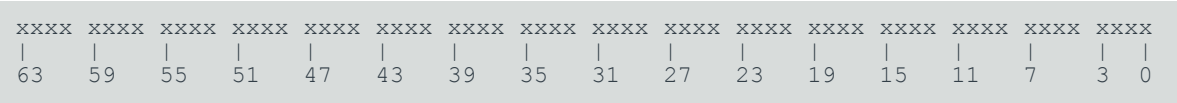
Register offset

0x28

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-362: ext\_clusterpmu\_pmevcntr5 bit assignments

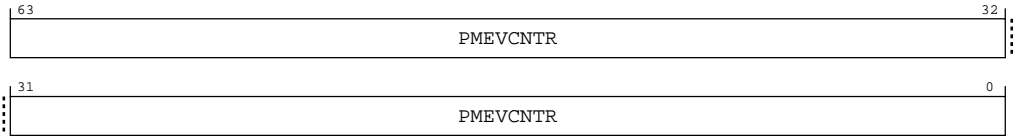


Table B-487: CLUSTERPMU\_PMEVCNTR5 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 5.	64 {x}

Accessibility

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\_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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-363: ext\_clusterpmu\_pmevtyper0 bit assignments

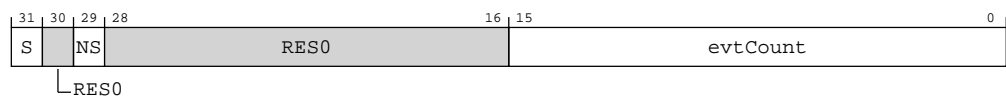


Table B-488: CLUSTERPMU\_PMEVTYPE0 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions.  <b>0b0</b> Count Secure events.  <b>0b1</b> 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

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\_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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-364: ext\_clusterpmu\_pmevtyper1 bit assignments

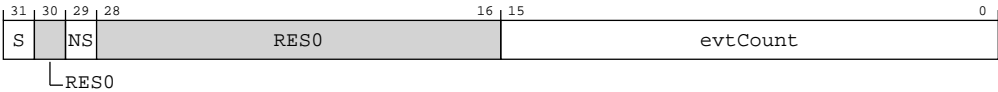


Table B-489: CLUSTERPMU\_PMEVTYPER1 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions.  <b>0b0</b> Count Secure events.  <b>0b1</b> Do not count Secure events.	x
[30]	RES0	Reserved	RES0



Bits	Name	Description	Reset
[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

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\_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

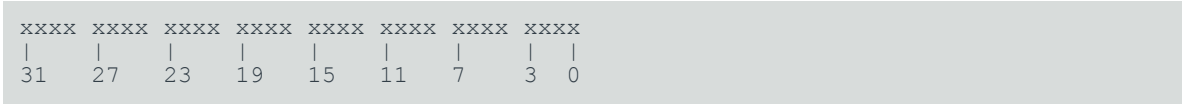
#### Register offset

0x408

#### Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-365: ext\_clusterpmu\_pmevtyper2 bit assignments

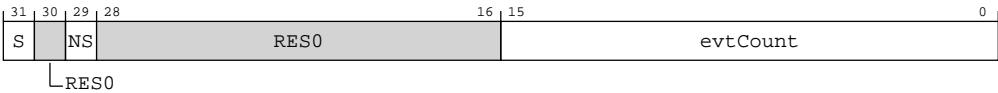


Table B-490: CLUSTERPMU\_PMEVTYPER2 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions.  <b>0b0</b> Count Secure events.  <b>0b1</b> 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

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\_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

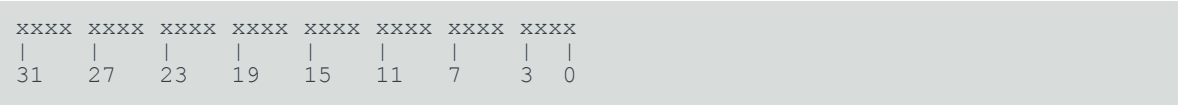
Register offset

0x40C

Access type

See bit descriptions

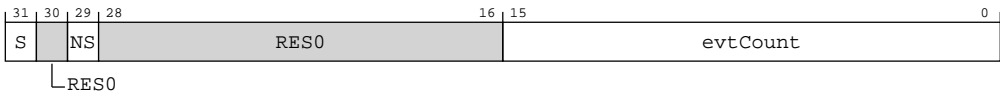
Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-366: ext\_clusterpmu\_pmevtyper3 bit assignments



**Table B-491: CLUSTERPMU\_PMEVTYPER3 bit descriptions**

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions.  <b>0b0</b> Count Secure events.  <b>0b1</b> 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

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\_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

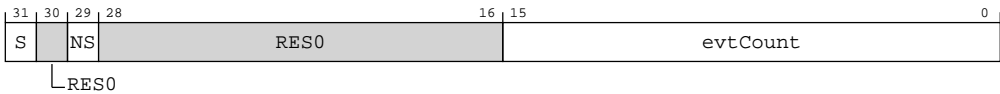
**Reset value**



Where the reset reads xxxx, see individual bits.

**Bit descriptions**

**Figure B-367: ext\_clusterpmu\_pmevtyper4 bit assignments**



**Table B-492: CLUSTERPMU\_PMEVTYPER4 bit descriptions**

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions.  <b>0b0</b> Count Secure events.  <b>0b1</b> 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

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\_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**

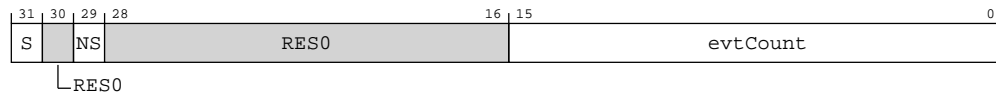
xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-368: ext\_clusterpmu\_pmevtyper5 bit assignments**



**Table B-493: CLUSTERPMU\_PMEVTYPER5 bit descriptions**

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions.  <b>0b0</b> Count Secure events.  <b>0b1</b> 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

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\_PMEVCNTR0, 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

0x600

Access type

See bit descriptions

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-369: ext\_clusterpmu\_pmevcntr0 bit assignments

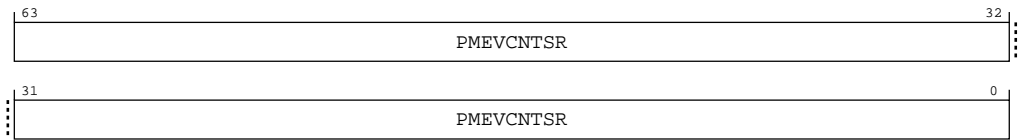


Table B-494: CLUSTERPMU\_PMEVCNTR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 0.	64 { x }

Accessibility

This interface is accessible as follows:



**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess() && SoftwareLockStatus()**

RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess() && !SoftwareLockStatus()**

RO

**Otherwise**

ERROR

B.2.4.14 CLUSTERPMU\_PMEVCNTR1, 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

0x608

Access type

See bit descriptions

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-370: ext\_clusterpmu\_pmevcntrs1 bit assignments

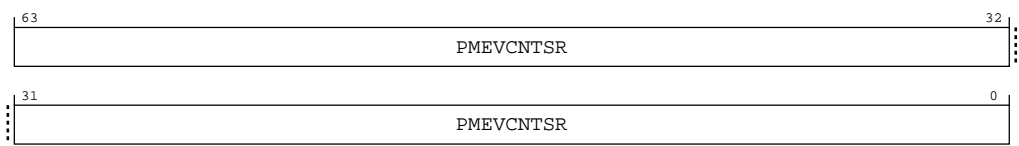


Table B-495: CLUSTERPMU\_PMEVCNTR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 1.	64 { x }

Accessibility

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

RO

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

RO

Otherwise

ERROR

B.2.4.15 CLUSTERPMU\_PMEVCNTR2, 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

0x610

Access type

See bit descriptions

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3
															0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-371: ext\_clusterpmu\_pmevcntrs2 bit assignments

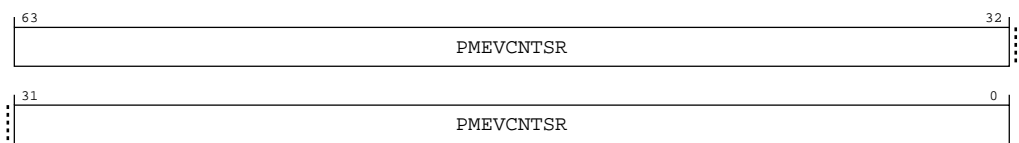


Table B-496: CLUSTERPMU\_PMEVCNTR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 2.	64 { x }

Accessibility

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\_PMEVCNTR3, 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

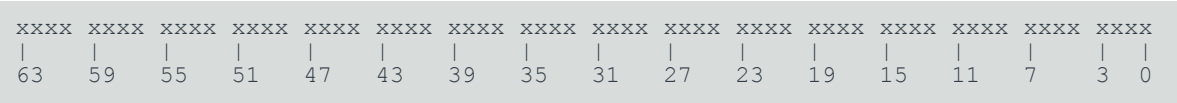
Register offset

0x618

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-372: ext\_clusterpmu\_pmevcntr3 bit assignments

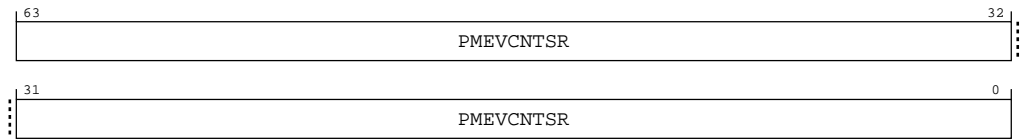


Table B-497: CLUSTERPMU\_PMEVCNTR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 3.	64 { x }

Accessibility

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\_PMEVCNTR4, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 4, which counts events.

**Configurations**  
This register is available in all configurations.

**Attributes**

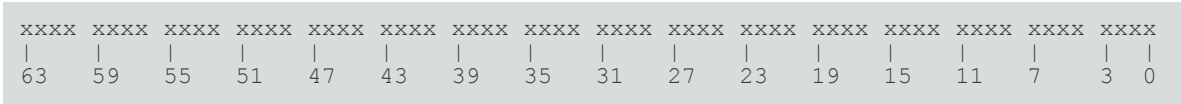
**Width**  
64

**Component**  
CLUSTERPMU

**Register offset**  
0x620

**Access type**  
See bit descriptions

**Reset value**



Where the reset reads xxxx, see individual bits.

**Bit descriptions**

Figure B-373: ext\_clusterpmu\_pmevcntr4 bit assignments

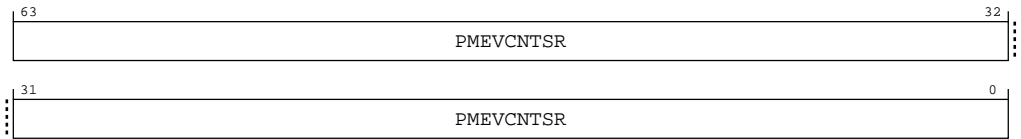


Table B-498: CLUSTERPMU\_PMEVCNTR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR4	Event counter 4.	64 { x }

Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()**  
RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()**  
RO

**Otherwise**  
ERROR

B.2.4.18 CLUSTERPMU\_PMEVCNTR5, 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**  
0x628

**Access type**  
See bit descriptions

Reset value

xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
63	59	55	51	47	43	39	35	31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-374: ext\_clusterpmu\_pmevcntsr5 bit assignments

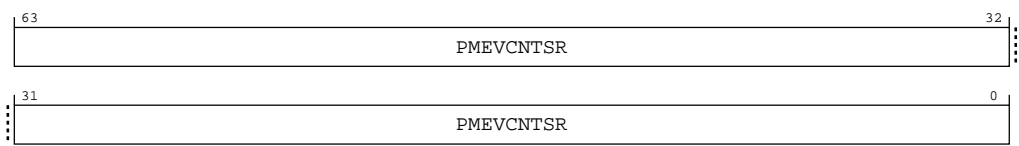


Table B-499: CLUSTERPMU\_PMEVCNTR5 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR5	Event counter 5.	64 { x }

Accessibility

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\_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

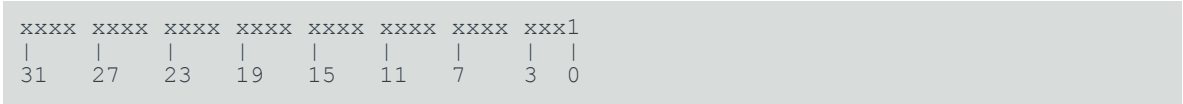
Register offset

0x638

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-375: ext\_clusterpmu\_pmsssr bit assignments

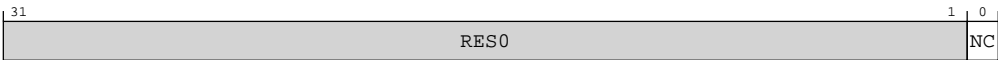


Table B-500: CLUSTERPMU\_PMSSSR bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0
[0]	NC	No capture. Indicates whether the PMU counters have been captured.  <b>0b0</b> PMU counters captured.  <b>0b1</b> PMU counters not captured.	0b1

Accessibility

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\_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\_PMOVSSSET and ext-CLUSTERPMU\_PMOVSCLR.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32

##### Component

CLUSTERPMU

##### Register offset

0x640

##### Access type

See bit descriptions

##### Reset value



Note

Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure B-376: ext\_clusterpmu\_pmovssr bit assignments

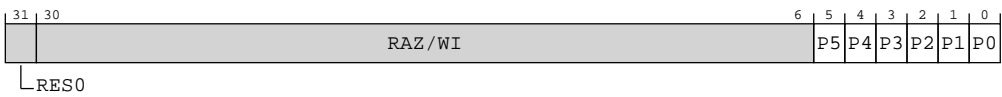


Table B-501: CLUSTERPMU\_PMOVSSR bit descriptions

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	<p>Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x
[4]	P4	<p>Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x
[3]	P3	<p>Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x
[2]	P2	<p>Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x
[1]	P1	<p>Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x

Bits	Name	Description	Reset
[0]	P0	<p>Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are RAZ/WI.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x

### Accessibility

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\_PMCNTENSET, Cluster Performance Monitors Count Enable Set register

Enables any implemented event counters AArch64-IMP\_CLUSTERPMEVCNTR<n>.

### Configurations

External register CLUSTERPMU\_PMCNTENSET bits [31:0] are architecturally mapped to AArch64 System register [A.2.2 IMP\\_CLUSTERPMCNTENSET\\_EL1, Performance Monitors Count Enable Set Register](#) on page 319 bits [31:0].

### Attributes

#### Width

32

#### Component

CLUSTERPMU

#### Register offset

0xC00

#### Access type

See bit descriptions

#### Reset value

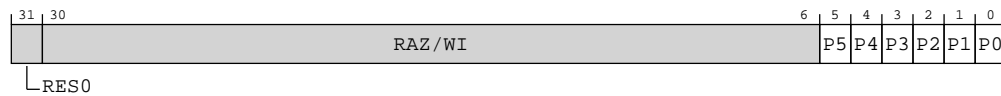
x000 0000 0000 0000 0000 0000 00xx xxxx



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-377: ext\_clusterpmu\_pmcntenset bit assignments**



**Table B-502: CLUSTERPMU\_PMCNTENSET bit descriptions**

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[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 <b>RAZ/WI</b> . <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. <b>0b1</b> 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 <b>RAZ/WI</b> . <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. <b>0b1</b> 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 <b>RAZ/WI</b> . <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. <b>0b1</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x

Bits	Name	Description	Reset
[2]	P2	<p>Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x
[1]	P1	<p>Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x
[0]	P0	<p>Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x

## Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()**

RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()**

RW

**Otherwise**

ERROR

B.2.4.22 CLUSTERPMU\_PMCNTENCLR, Cluster Performance Monitors Count Enable Clear register

Disables any implemented event counters AArch64-IMP\_CLUSTERPMEVCNTR<n>.

Configurations

External register CLUSTERPMU\_PMCNTENCLR bits [31:0] are architecturally mapped to AArch64 System register [A.2.3 IMP\\_CLUSTERPMCNTENCLR\\_EL1, Performance Monitors Count Enable Clear Register](#) on page 322 bits [31:0].

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC20

Access type

See bit descriptions

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-378: ext\_clusterpmu\_pmcntenclr bit assignments

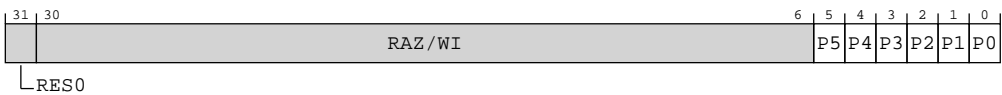


Table B-503: CLUSTERPMU\_PMCNTENCLR bit descriptions

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	<p>Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x
[4]	P4	<p>Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x
[3]	P3	<p>Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x
[2]	P2	<p>Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x
[1]	P1	<p>Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x

Bits	Name	Description	Reset
[0]	P0	<p>Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p>	x

## Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()**

RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()**

RW

**Otherwise**

ERROR

### B.2.4.23 CLUSTERPMU\_PMINTENSET, Cluster Performance Monitors Interrupt Enable Set register

Enables the generation of interrupt requests on overflows from the event counters ext-CLUSTERPMU\_PMEVCNTR<n>.

## Configurations

External register CLUSTERPMU\_PMINTENSET bits [31:0] are architecturally mapped to AArch64 System register [A.2.7 IMP\\_CLUSTERPMINTENSET\\_EL1, Performance Monitors Interrupt Enable Set Register](#) on page 334 bits [31:0].

## Attributes

### Width

32

### Component

CLUSTERPMU

### Register offset

0xC40

### Access type

See bit descriptions



## Reset value

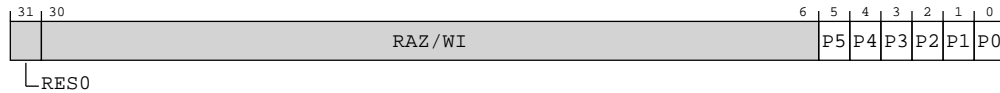
```
x000 0000 0000 0000 0000 0000 00xx xxxx
|    |    |    |    |    |    |    |
31   27   23   19   15   11   7     3   0
```



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-379: ext\_clusterpmu\_pmintenset bit assignments**



**Table B-504: CLUSTERPMU\_PMINTENSET bit descriptions**

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	<p>Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[4]	P4	<p>Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x

Bits	Name	Description	Reset
[3]	P3	<p>Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[2]	P2	<p>Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[1]	P1	<p>Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[0]	P0	<p>Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x

## Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()**

RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()**

RW

Otherwise  
ERROR

B.2.4.24 CLUSTERPMU\_PMINTENCLR, Cluster Performance Monitors Interrupt Enable Clear register

Disables the generation of interrupt requests on overflows from the event counters ext-CLUSTERPMU\_PMEVCNTR<n>.

**Configurations**  
External register CLUSTERPMU\_PMINTENCLR bits [31:0] are architecturally mapped to AArch64 System register [A.2.8 IMP\\_CLUSTERPMINTENCLR\\_EL1, Performance Monitors Interrupt Enable Clear Register](#) on page 337 bits [31:0].

**Attributes**  
**Width**  
32  
**Component**  
CLUSTERPMU  
**Register offset**  
0xC60  
**Access type**  
See bit descriptions  
**Reset value**

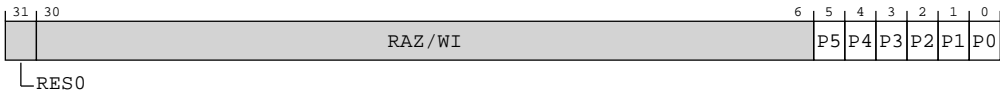
x000	0000	0000	0000	0000	0000	00xx	xxxx
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-380: ext\_clusterpmu\_pmintenclr bit assignments



**Table B-505: CLUSTERPMU\_PMINTENCLR bit descriptions**

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	<p>Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[4]	P4	<p>Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[3]	P3	<p>Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[2]	P2	<p>Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x

Bits	Name	Description	Reset
[1]	P1	<p>Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x
[0]	P0	<p>Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;.</p> <p>If ext-CLUSTERPMU_PMCFCGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is disabled. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; interrupt request.</p>	x

## Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess() && SoftwareLockStatus()**

RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess() && !SoftwareLockStatus()**

RW

**Otherwise**

ERROR

## B.2.4.25 CLUSTERPMU\_PMOVSLR, Cluster Performance Monitors Overflow Flag Status Clear register

Contains the state of the overflow bit for each of the implemented event counters AArch64-PMEVCNTR<n>. Writing to this register clears these bits.

## Configurations

External register CLUSTERPMU\_PMOVSLR bits [31:0] are architecturally mapped to AArch64 System register [A.2.5 IMP\\_CLUSTERPMOVSLR\\_EL1, Performance Monitors Overflow Flag Status Clear Register](#) on page 328 bits [31:0].

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xC80

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-381: ext\_clusterpmu\_pmovsclr bit assignments

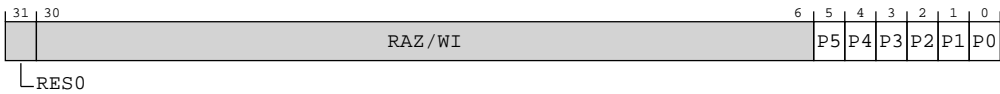


Table B-506: CLUSTERPMU\_PMOVSLR bit descriptions

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[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 <b>RAZ/WI</b> .  <b>0b0</b>  When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b>  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
[4]	P4	<p>Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 0.</p>	x
[3]	P3	<p>Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 0.</p>	x
[2]	P2	<p>Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 0.</p>	x
[1]	P1	<p>Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 0.</p>	x
[0]	P0	<p>Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 0.</p>	x

## Accessibility

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\_PMOVSET, Cluster Performance Monitors Overflow Flag Status Set register

Sets the state of the overflow bit for each of the implemented event counters AArch64-PMVCNTR<n>.

Configurations

External register CLUSTERPMU\_PMOVSET bits [31:0] are architecturally mapped to AArch64 System register [A.2.4 IMP\\_CLUSTERPMOVSET\\_EL1, Performance Monitors Overflow Flag Status Set Register](#) on page 325 bits [31:0].

Attributes

Width

32

Component

CLUSTERPMU

Register offset

0xCC0

Access type

See bit descriptions

Reset value

x000	0000	0000	0000	0000	0000	00xx	xxxx
31	27	23	19	15	11	7	3 0

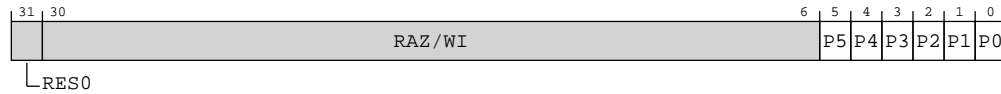


Where the reset reads xxxx, see individual bits.



## Bit descriptions

**Figure B-382: ext\_clusterpmu\_pmovsset bit assignments**



**Table B-507: CLUSTERPMU\_PMOVSSET bit descriptions**

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	P5	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFCGR.N] are <b>RAZ/WI</b> .  <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> 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_PMCFCGR.N] are <b>RAZ/WI</b> .  <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> 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_PMCFCGR.N] are <b>RAZ/WI</b> .  <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> 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_PMCFCGR.N] are <b>RAZ/WI</b> .  <b>0b0</b> When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.  <b>0b1</b> 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
[1]	P1	<p>Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x
[0]	P0	<p>Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR&lt;n&gt;. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are <b>RAZ/WI</b>.</p> <p><b>0b0</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has not overflowed since this bit was last cleared. When written, has no effect.</p> <p><b>0b1</b></p> <p>When read, means that ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR&lt;n&gt; overflow bit to 1.</p>	x

## Accessibility

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\_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

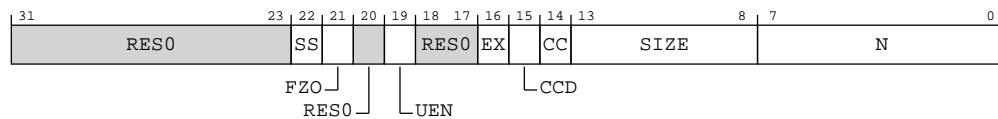
xxxx	xxxx	x11x	0xx0	0011	1111	0000	0101
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-383: ext\_clusterpmu\_pmcfr bit assignments**



**Table B-508: CLUSTERPMU\_PMCFR bit descriptions**

Bits	Name	Description	Reset
[31:23]	RES0	Reserved	RES0
[22]	SS	Snapshot supported. <b>0b1</b> Snapshot supported.	0b1
[21]	FZO	Freeze on overflow supported. <b>0b1</b> Freeze on overflow supported.	0b1
[20]	RES0	Reserved	RES0
[19]	UEN	User-mode Enable Register supported. <b>0b0</b> User-mode Enable Register not supported.	0b0
[18:17]	RES0	Reserved	RES0
[16]	EX	Export supported. <b>0b0</b> ext-CLUSTERPMU_PMC.R.X is RES0.	0b0
[15]	CCD	Cycle counter has prescale. <b>0b0</b> ext-CLUSTERPMU_PMC.R.D is RES0.	0b0

Bits	Name	Description	Reset
[14]	CC	Dedicated cycle counter.  <b>0b0</b> Dedicated cycle counter ext-CLUSTERPMU_PMCCNTR is not supported.	0b0
[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.  <b>0b111111</b> The largest counter is 64-bits. Counters are at doubleword-aligned addresses.	0b111111
[7:0]	N	Number of counters implemented.  <b>0b00000101</b> Six event counters implemented.	0x05

### Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess()**

RO

**Otherwise**

ERROR

### B.2.4.28 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\_PMCFCGR and the ID registers.

External register CLUSTERPMU\_PMCR bits [7:0] are architecturally mapped to AArch64 System register [A.2.1 IMP\\_CLUSTERPMCR\\_EL1, Performance Monitors Control Register](#) on page 316 bits [7:0].

### Attributes

#### Width

32

#### Component

CLUSTERPMU

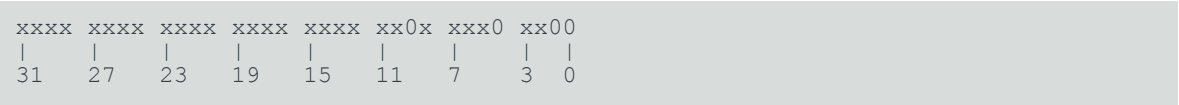
#### Register offset

0xE04

Access type

inconsistent

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-384: ext\_clusterpmu\_pmcr bit assignments

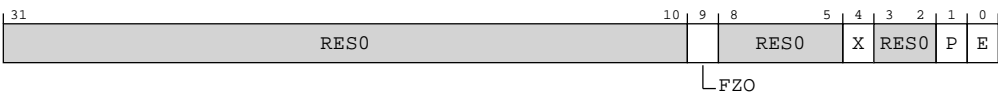


Table B-509: CLUSTERPMU\_PMCR bit descriptions

Bits	Name	Description	Reset
[31:10]	RES0	Reserved	RES0
[9]	FZO	Freeze on overflow. <b>0b0</b> Freeze on overflow disabled. <b>0b1</b> Freeze on overflow enabled.	0b0
[8:5]	RES0	Reserved	RES0
[4]	X	This field enables the exporting of events over an event bus to another device. <b>0b0</b> Cluster PMU events are not exported externally.	0b0
[3:2]	RES0	Reserved	RES0
[1]	P	Event counter reset. This bit is WO. <b>0b0</b> No action. <b>0b1</b> Reset all event counters to zero.  This bit is always <b>RAZ</b> .  <b>Note:</b> Resetting the event counters does not change the event counter overflow bits.	0b0

Bits	Name	Description	Reset
[0]	E	<p>Enable.</p> <p><b>0b0</b> All event counters are disabled.</p> <p><b>0b1</b> All event counters can be enabled by ext-CLUSTERPMU_PMCNTENSET.</p> <p>This bit is RW.</p>	0b0

### Accessibility

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\_PMIIDR, Cluster Performance Monitors Implementation Identification register

Defines the implemented of the component..

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32

#### Component

CLUSTERPMU

#### Register offset

0xE08

#### Access type

See bit descriptions

#### Reset value

0100 1110 1100 0000 0001 0100 0011 1011

Bit descriptions

Figure B-385: ext\_clusterpmu\_pmiidr bit assignments

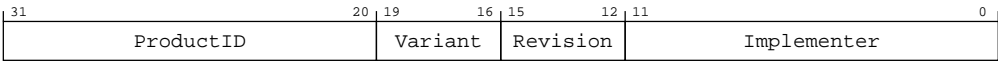


Table B-510: CLUSTERPMU\_PMIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Value identifying the PMU Component. <b>0b010011101100</b> DSU-120AE Cluster PMU.	0x4EC
[19:16]	Variant	Value used to distinguish product variants, or major revisions of the product. <b>0b0000</b> Product variant 0.	0b0000
[15:12]	Revision	Value used to distinguish minor revisions of the product. <b>0b0000</b> Product revision 0. <b>0b0001</b> Product revision 1.	0b0001
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the PMU Component. For an Arm implementation, bits[11:0] are 0x43B. <b>0b010000111011</b> Arm implementation.	0x43B

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.30 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

External register CLUSTERPMU\_PMCEID0 bits [31:0] are architecturally mapped to AArch64 System register [A.2.12 IMP\\_CLUSTERPMCEID0\\_EL1, Performance Monitors Common Event Identification Register 0](#) on page 345 bits [31:0].

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-386: ext\_clusterpmu\_pmceid0 bit assignments

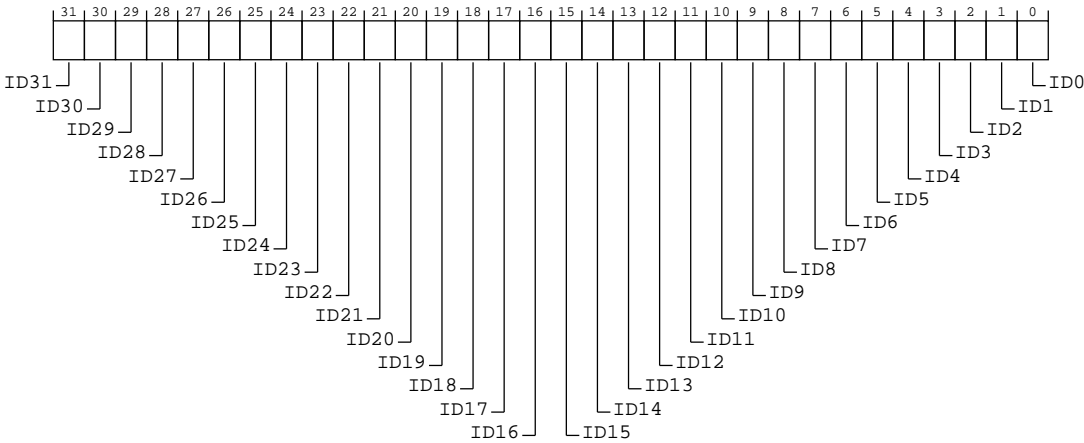


Table B-511: CLUSTERPMU\_PMCEID0 bit descriptions

Bits	Name	Description	Reset
[31]	ID31	Common event 0x001F implemented. <b>0b0</b> Event 0x001F not implemented.	0b0
[30]	ID30	Common event 0x001E implemented. <b>0b0</b> CHAIN event not implemented.	0b0



Bits	Name	Description	Reset
[29]	ID29	Common event 0x001D implemented. <b>0b1</b> BUS_CYCLES event implemented.	0b1
[28]	ID28	Common event 0x001C implemented. <b>0b0</b> Event 0x001C not implemented.	0b0
[27]	ID27	Common event 0x001B implemented. <b>0b0</b> Event 0x001B not implemented.	0b0
[26]	ID26	Common event 0x001A implemented. <b>0b1</b> MEMORY_ERROR event implemented.	0b1
[25]	ID25	Common event 0x0019 implemented. <b>0b1</b> BUS_ACCESS event implemented.	0b1
[24]	ID24	Common event 0x0018 implemented. <b>0b0</b> Event 0x0018 not implemented.	0b0
[23]	ID23	Common event 0x0017 implemented. <b>0b0</b> Event 0x0017 not implemented.	0b0
[22]	ID22	Common event 0x0016 implemented. <b>0b0</b> Event 0x0016 not implemented.	0b0
[21]	ID21	Common event 0x0015 implemented. <b>0b0</b> Event 0x0015 not implemented.	0b0
[20]	ID20	Common event 0x0014 implemented. <b>0b0</b> Event 0x0014 not implemented.	0b0
[19]	ID19	Common event 0x0013 implemented. <b>0b0</b> Event 0x0013 not implemented.	0b0
[18]	ID18	Common event 0x0012 implemented. <b>0b0</b> Event 0x0012 not implemented.	0b0
[17]	ID17	Common event 0x0011 implemented. <b>0b1</b> CYCLES event implemented.	0b1
[16]	ID16	Common event 0x0010 implemented. <b>0b0</b> Event 0x0010 not implemented.	0b0

Bits	Name	Description	Reset
[15]	ID15	Common event 0x000F implemented. <b>0b0</b> Event 0x000F not implemented.	0b0
[14]	ID14	Common event 0x000E implemented. <b>0b0</b> Event 0x000E not implemented.	0b0
[13]	ID13	Common event 0x000D implemented. <b>0b0</b> Event 0x000D not implemented.	0b0
[12]	ID12	Common event 0x000C implemented. <b>0b0</b> Event 0x000C not implemented.	0b0
[11]	ID11	Common event 0x000B implemented. <b>0b0</b> Event 0x000B not implemented.	0b0
[10]	ID10	Common event 0x000A implemented. <b>0b0</b> Event 0x000A not implemented.	0b0
[9]	ID9	Common event 0x0009 implemented. <b>0b0</b> Event 0x0009 not implemented.	0b0
[8]	ID8	Common event 0x0008 implemented. <b>0b0</b> Event 0x0008 not implemented.	0b0
[7]	ID7	Common event 0x0007 implemented. <b>0b0</b> Event 0x0007 not implemented.	0b0
[6]	ID6	Common event 0x0006 implemented. <b>0b0</b> Event 0x0006 not implemented.	0b0
[5]	ID5	Common event 0x0005 implemented. <b>0b0</b> Event 0x0005 not implemented.	0b0
[4]	ID4	Common event 0x0004 implemented. <b>0b0</b> Event 0x0004 not implemented.	0b0
[3]	ID3	Common event 0x0003 implemented. <b>0b0</b> Event 0x0003 not implemented.	0b0
[2]	ID2	Common event 0x0002 implemented. <b>0b0</b> Event 0x0002 not implemented.	0b0

Bits	Name	Description	Reset
[1]	ID1	Common event 0x0001 implemented. <b>0b0</b> Event 0x0001 not implemented.	0b0
[0]	ID0	Common event 0x0000 implemented. <b>0b0</b> Event 0x0000 not implemented.	0b0

### Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess()**

RO

**Otherwise**

ERROR

### B.2.4.31 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

External register CLUSTERPMU\_PMCEID1 bits [31:0] are architecturally mapped to AArch64 System register [A.2.13 IMP\\_CLUSTERPMCEID1\\_EL1, Performance Monitors Common Event Identification Register 1](#) on page 353 bits [31:0].

### Attributes

#### Width

32

#### Component

CLUSTERPMU

#### Register offset

0xE24

#### Access type

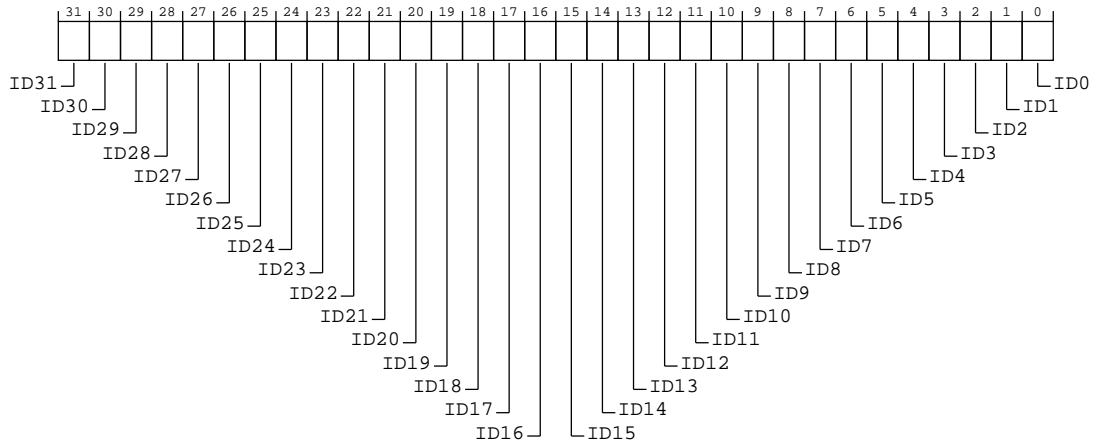
See bit descriptions

#### Reset value

0000 0000 0000 0000 0001 1110 0000 0000

## Bit descriptions

**Figure B-387: ext\_clusterpmu\_pmceid1 bit assignments**



**Table B-512: CLUSTERPMU\_PMCEID1 bit descriptions**

Bits	Name	Description	Reset
[31]	ID31	Common event 0x003F implemented. <b>0b0</b> Event 0x003F not implemented.	0b0
[30]	ID30	Common event 0x003E implemented. <b>0b0</b> Event 0x003E not implemented.	0b0
[29]	ID29	Common event 0x003D implemented. <b>0b0</b> Event 0x003D not implemented.	0b0
[28]	ID28	Common event 0x003C implemented. <b>0b0</b> Event 0x003C not implemented.	0b0
[27]	ID27	Common event 0x003B implemented. <b>0b0</b> Event 0x003B not implemented.	0b0
[26]	ID26	Common event 0x003A implemented. <b>0b0</b> Event 0x003A not implemented.	0b0
[25]	ID25	Common event 0x0039 implemented. <b>0b0</b> Event 0x0039 not implemented.	0b0
[24]	ID24	Common event 0x0038 implemented. <b>0b0</b> Event 0x0038 not implemented.	0b0

Bits	Name	Description	Reset
[23]	ID23	Common event 0x0037 implemented. <b>0b0</b> Event 0x0037 not implemented.	0b0
[22]	ID22	Common event 0x0036 implemented. <b>0b0</b> Event 0x0036 not implemented.	0b0
[21]	ID21	Common event 0x0035 implemented. <b>0b0</b> Event 0x0035 not implemented.	0b0
[20]	ID20	Common event 0x0034 implemented. <b>0b0</b> Event 0x0034 not implemented.	0b0
[19]	ID19	Common event 0x0033 implemented. <b>0b0</b> Event 0x0033 not implemented.	0b0
[18]	ID18	Common event 0x0032 implemented. <b>0b0</b> Event 0x0032 not implemented.	0b0
[17]	ID17	Common event 0x0031 implemented. <b>0b0</b> Event 0x0031 not implemented.	0b0
[16]	ID16	Common event 0x0030 implemented. <b>0b0</b> Event 0x0030 not implemented.	0b0
[15]	ID15	Common event 0x002F implemented. <b>0b0</b> Event 0x002F not implemented.	0b0
[14]	ID14	Common event 0x002E implemented. <b>0b0</b> Event 0x002E not implemented.	0b0
[13]	ID13	Common event 0x002D implemented. <b>0b0</b> Event 0x002D not implemented.	0b0
[12]	ID12	Common event 0x002C implemented. <b>0b1</b> L3D_CACHE_WB event implemented.	0b1
[11]	ID11	Common event 0x002B implemented. <b>0b1</b> L3D_CACHE event implemented.	0b1
[10]	ID10	Common event 0x002A implemented. <b>0b1</b> L3D_CACHE_REFILL event implemented.	0b1

Bits	Name	Description	Reset
[9]	ID9	Common event 0x0029 implemented. <b>0b1</b> L3D_CACHE_ALLOCATE event implemented.	0b1
[8]	ID8	Common event 0x0028 implemented. <b>0b0</b> Event 0x0028 not implemented.	0b0
[7]	ID7	Common event 0x0027 implemented. <b>0b0</b> Event 0x0027 not implemented.	0b0
[6]	ID6	Common event 0x0026 implemented. <b>0b0</b> Event 0x0026 not implemented.	0b0
[5]	ID5	Common event 0x0025 implemented. <b>0b0</b> Event 0x0025 not implemented.	0b0
[4]	ID4	Common event 0x0024 implemented. <b>0b0</b> Event 0x0024 not implemented.	0b0
[3]	ID3	Common event 0x0023 implemented. <b>0b0</b> Event 0x0023 not implemented.	0b0
[2]	ID2	Common event 0x0022 implemented. <b>0b0</b> Event 0x0022 not implemented.	0b0
[1]	ID1	Common event 0x0021 implemented. <b>0b0</b> Event 0x0021 not implemented.	0b0
[0]	ID0	Common event 0x0020 implemented. <b>0b0</b> Event 0x0020 not implemented.	0b0

## Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess()**

RO

**Otherwise**

ERROR

## B.2.4.32 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

External register CLUSTERPMU\_PMCEID2 bits [31:0] are architecturally mapped to AArch64 System register [A.2.12 IMP\\_CLUSTERPMCEID0\\_EL1, Performance Monitors Common Event Identification Register 0](#) on page 345 bits [63:32].

### Attributes

#### Width

32

#### Component

CLUSTERPMU

#### Register offset

0xE28

#### Access type

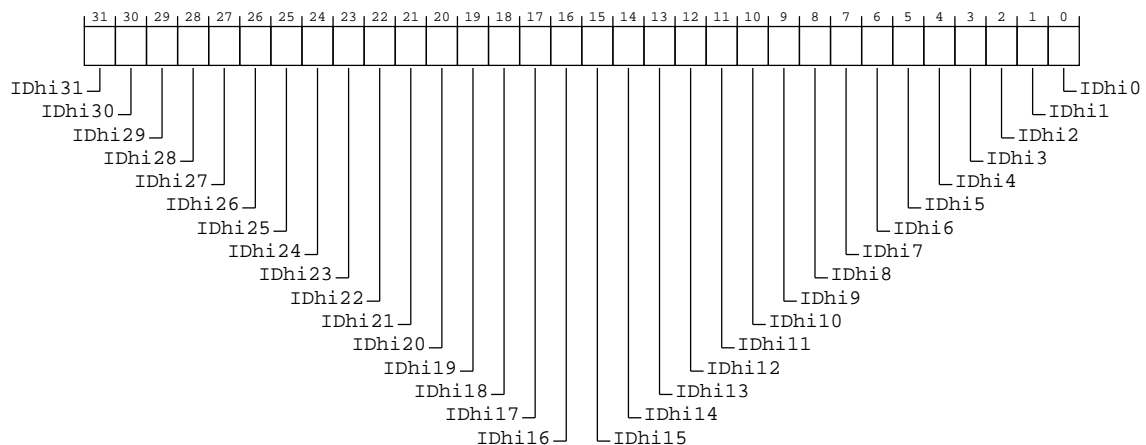
See bit descriptions

#### Reset value

0000 0000 0000 0000 0000 0000 0000 0000

### Bit descriptions

**Figure B-388: ext\_clusterpmu\_pmceid2 bit assignments**



**Table B-513: CLUSTERPMU\_PMCEID2 bit descriptions**

Bits	Name	Description	Reset
[31]	IDhi31	Common event 0x401F implemented. <b>0b0</b> Event 0x401F not implemented.	0b0
[30]	IDhi30	Common event 0x401E implemented. <b>0b0</b> Event 0x401E not implemented.	0b0
[29]	IDhi29	Common event 0x401D implemented. <b>0b0</b> Event 0x401D not implemented.	0b0
[28]	IDhi28	Common event 0x401C implemented. <b>0b0</b> Event 0x401C not implemented.	0b0
[27]	IDhi27	Common event 0x401B implemented. <b>0b0</b> Event 0x401B not implemented.	0b0
[26]	IDhi26	Common event 0x401A implemented. <b>0b0</b> Event 0x401A not implemented.	0b0
[25]	IDhi25	Common event 0x4019 implemented. <b>0b0</b> Event 0x4019 not implemented.	0b0
[24]	IDhi24	Common event 0x4018 implemented. <b>0b0</b> Event 0x4018 not implemented.	0b0
[23]	IDhi23	Common event 0x4017 implemented. <b>0b0</b> Event 0x4017 not implemented.	0b0
[22]	IDhi22	Common event 0x4016 implemented. <b>0b0</b> Event 0x4016 not implemented.	0b0
[21]	IDhi21	Common event 0x4015 implemented. <b>0b0</b> Event 0x4015 not implemented.	0b0
[20]	IDhi20	Common event 0x4014 implemented. <b>0b0</b> Event 0x4014 not implemented.	0b0
[19]	IDhi19	Common event 0x4013 implemented. <b>0b0</b> Event 0x4013 not implemented.	0b0



Bits	Name	Description	Reset
[18]	IDhi18	Common event 0x4012 implemented. <b>0b0</b> Event 0x4012 not implemented.	0b0
[17]	IDhi17	Common event 0x4011 implemented. <b>0b0</b> Event 0x4011 not implemented.	0b0
[16]	IDhi16	Common event 0x4010 implemented. <b>0b0</b> Event 0x4010 not implemented.	0b0
[15]	IDhi15	Common event 0x400F implemented. <b>0b0</b> Event 0x400F not implemented.	0b0
[14]	IDhi14	Common event 0x400E implemented. <b>0b0</b> Event 0x400E not implemented.	0b0
[13]	IDhi13	Common event 0x400D implemented. <b>0b0</b> Event 0x400D not implemented.	0b0
[12]	IDhi12	Common event 0x400C implemented. <b>0b0</b> Event 0x400C not implemented.	0b0
[11]	IDhi11	Common event 0x400B implemented. <b>0b0</b> Event 0x400B not implemented.	0b0
[10]	IDhi10	Common event 0x400A implemented. <b>0b0</b> Event 0x400A not implemented.	0b0
[9]	IDhi9	Common event 0x4009 implemented. <b>0b0</b> Event 0x4009 not implemented.	0b0
[8]	IDhi8	Common event 0x4008 implemented. <b>0b0</b> Event 0x4008 not implemented.	0b0
[7]	IDhi7	Common event 0x4007 implemented. <b>0b0</b> Event 0x4007 not implemented.	0b0
[6]	IDhi6	Common event 0x4006 implemented. <b>0b0</b> Event 0x4006 not implemented.	0b0
[5]	IDhi5	Common event 0x4005 implemented. <b>0b0</b> Event 0x4005 not implemented.	0b0

Bits	Name	Description	Reset
[4]	IDhi4	Common event 0x4004 implemented. <b>0b0</b> Event 0x4004 not implemented.	0b0
[3]	IDhi3	Common event 0x4003 implemented. <b>0b0</b> Event 0x4003 not implemented.	0b0
[2]	IDhi2	Common event 0x4002 implemented. <b>0b0</b> Event 0x4002 not implemented.	0b0
[1]	IDhi1	Common event 0x4001 implemented. <b>0b0</b> Event 0x4001 not implemented.	0b0
[0]	IDhi0	Common event 0x4000 implemented. <b>0b0</b> Event 0x4000 not implemented.	0b0

### Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess()**

RO

**Otherwise**

ERROR

### B.2.4.33 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

External register CLUSTERPMU\_PMCEID3 bits [31:0] are architecturally mapped to AArch64 System register [A.2.13 IMP\\_CLUSTERPMCEID1\\_EL1, Performance Monitors Common Event Identification Register 1](#) on page 353 bits [63:32].

### Attributes

#### Width

32

## Component

CLUSTERPMU

## Register offset

0xE2C

## Access type

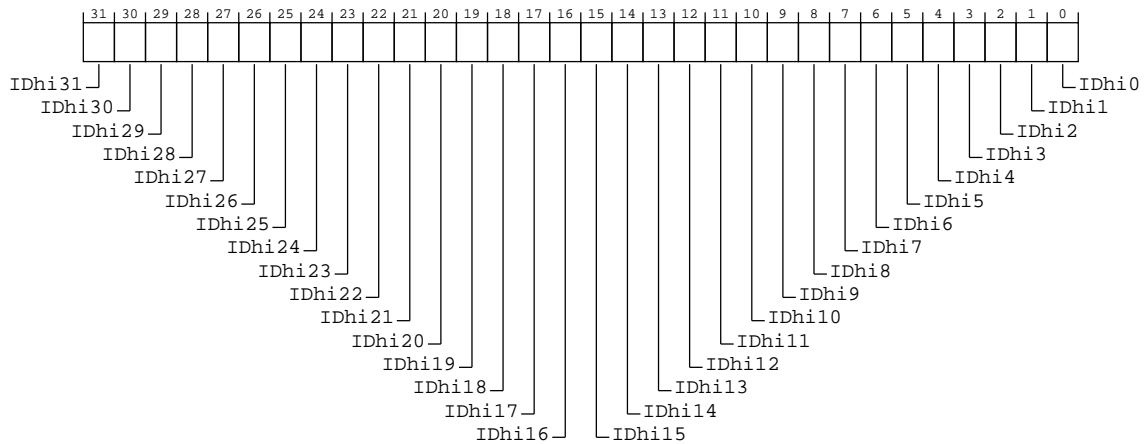
See bit descriptions

## Reset value

0000 0000 0000 0000 0000 0000 0000 0000

## Bit descriptions

**Figure B-389: ext\_clusterpmu\_pmceid3 bit assignments**



**Table B-514: CLUSTERPMU\_PMCEID3 bit descriptions**

Bits	Name	Description	Reset
[31]	IDhi31	Common event 0x403F implemented. <b>0b0</b> Event 0x403F not implemented.	0b0
[30]	IDhi30	Common event 0x403E implemented. <b>0b0</b> Event 0x403E not implemented.	0b0
[29]	IDhi29	Common event 0x403D implemented. <b>0b0</b> Event 0x403D not implemented.	0b0
[28]	IDhi28	Common event 0x403C implemented. <b>0b0</b> Event 0x403C not implemented.	0b0
[27]	IDhi27	Common event 0x403B implemented. <b>0b0</b> Event 0x403B not implemented.	0b0

Bits	Name	Description	Reset
[26]	IDhi26	Common event 0x403A implemented. <b>0b0</b> Event 0x403A not implemented.	0b0
[25]	IDhi25	Common event 0x4039 implemented. <b>0b0</b> Event 0x4039 not implemented.	0b0
[24]	IDhi24	Common event 0x4038 implemented. <b>0b0</b> Event 0x4038 not implemented.	0b0
[23]	IDhi23	Common event 0x4037 implemented. <b>0b0</b> Event 0x4037 not implemented.	0b0
[22]	IDhi22	Common event 0x4036 implemented. <b>0b0</b> Event 0x4036 not implemented.	0b0
[21]	IDhi21	Common event 0x4035 implemented. <b>0b0</b> Event 0x4035 not implemented.	0b0
[20]	IDhi20	Common event 0x4034 implemented. <b>0b0</b> Event 0x4034 not implemented.	0b0
[19]	IDhi19	Common event 0x4033 implemented. <b>0b0</b> Event 0x4033 not implemented.	0b0
[18]	IDhi18	Common event 0x4032 implemented. <b>0b0</b> Event 0x4032 not implemented.	0b0
[17]	IDhi17	Common event 0x4031 implemented. <b>0b0</b> Event 0x4031 not implemented.	0b0
[16]	IDhi16	Common event 0x4030 implemented. <b>0b0</b> Event 0x4030 not implemented.	0b0
[15]	IDhi15	Common event 0x402F implemented. <b>0b0</b> Event 0x402F not implemented.	0b0
[14]	IDhi14	Common event 0x402E implemented. <b>0b0</b> Event 0x402E not implemented.	0b0
[13]	IDhi13	Common event 0x402D implemented. <b>0b0</b> Event 0x402D not implemented.	0b0

Bits	Name	Description	Reset
[12]	IDhi12	Common event 0x402C implemented. <b>0b0</b> Event 0x402C not implemented.	0b0
[11]	IDhi11	Common event 0x402B implemented. <b>0b0</b> Event 0x402B not implemented.	0b0
[10]	IDhi10	Common event 0x402A implemented. <b>0b0</b> Event 0x402A not implemented.	0b0
[9]	IDhi9	Common event 0x4029 implemented. <b>0b0</b> Event 0x4029 not implemented.	0b0
[8]	IDhi8	Common event 0x4028 implemented. <b>0b0</b> Event 0x4028 not implemented.	0b0
[7]	IDhi7	Common event 0x4027 implemented. <b>0b0</b> Event 0x4027 not implemented.	0b0
[6]	IDhi6	Common event 0x4026 implemented. <b>0b0</b> Event 0x4026 not implemented.	0b0
[5]	IDhi5	Common event 0x4025 implemented. <b>0b0</b> Event 0x4025 not implemented.	0b0
[4]	IDhi4	Common event 0x4024 implemented. <b>0b0</b> Event 0x4024 not implemented.	0b0
[3]	IDhi3	Common event 0x4023 implemented. <b>0b0</b> Event 0x4023 not implemented.	0b0
[2]	IDhi2	Common event 0x4022 implemented. <b>0b0</b> Event 0x4022 not implemented.	0b0
[1]	IDhi1	Common event 0x4021 implemented. <b>0b0</b> Event 0x4021 not implemented.	0b0
[0]	IDhi0	Common event 0x4020 implemented. <b>0b0</b> Event 0x4020 not implemented.	0b0

## Accessibility

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() &&  
AllowExternalPMUAccess()

RO

Otherwise

ERROR

B.2.4.34 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

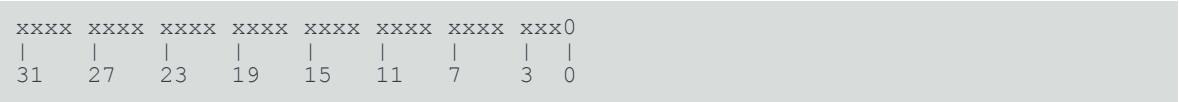
Register offset

0xE30

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-390: ext\_clusterpmu\_pmsscr bit assignments

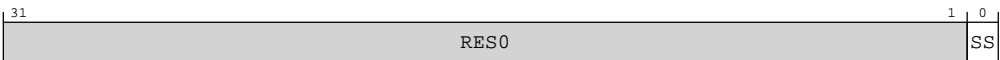


Table B-515: CLUSTERPMU\_PMSSCR bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0
[0]	SS	Capture now. The possible values for writing to this bit are:  <b>0b0</b> Ignored.  <b>0b1</b> Initiate a capture immediately.	0b0

Accessibility

This interface is accessible as follows:

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()

WI

When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()

WO

Otherwise

ERROR

B.2.4.35 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

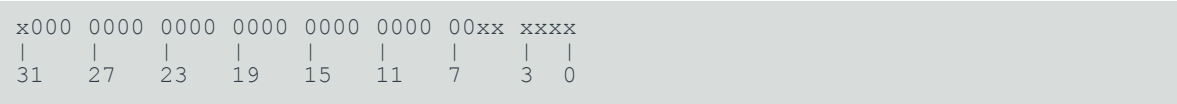
Register offset

0xE38

Access type

See bit descriptions

Reset value



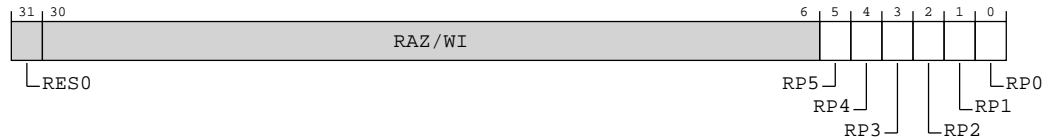


Note

Where the reset reads xxxx, see individual bits.

## Bit descriptions

**Figure B-391: ext\_clusterpmu\_pmssrr bit assignments**



**Table B-516: CLUSTERPMU\_PMSSRR bit descriptions**

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/ WI	Reserved	RAZ/ WI
[5]	RP5	Reset performance counter. For each bit [x], if $x \geq \text{ext-CLUSTERPMU\_PMCR.N}$ , the number of implemented counters, then RP[x] is <b>RAZ/WI</b> . Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSRR[x] are to be reset after a capture.  <b>0b0</b> Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSRR[x] on capture.  <b>0b1</b> Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSRR[x] on capture.	x
[4]	RP4	Reset performance counter. For each bit [x], if $x \geq \text{ext-CLUSTERPMU\_PMCR.N}$ , the number of implemented counters, then RP[x] is <b>RAZ/WI</b> . Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSRR[x] are to be reset after a capture.  <b>0b0</b> Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSRR[x] on capture.  <b>0b1</b> Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSRR[x] on capture.	x
[3]	RP3	Reset performance counter. For each bit [x], if $x \geq \text{ext-CLUSTERPMU\_PMCR.N}$ , the number of implemented counters, then RP[x] is <b>RAZ/WI</b> . Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSRR[x] are to be reset after a capture.  <b>0b0</b> Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSRR[x] on capture.  <b>0b1</b> Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSRR[x] on capture.	x



Bits	Name	Description	Reset
[2]	RP2	Reset performance counter. For each bit [x], if $x \geq \text{ext-CLUSTERPMU\_PMCR.N}$ , the number of implemented counters, then RP[x] is <b>RAZ/WI</b> . Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSER[x] are to be reset after a capture.  <b>0b0</b> Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSER[x] on capture.  <b>0b1</b> Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSER[x] on capture.	x
[1]	RP1	Reset performance counter. For each bit [x], if $x \geq \text{ext-CLUSTERPMU\_PMCR.N}$ , the number of implemented counters, then RP[x] is <b>RAZ/WI</b> . Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSER[x] are to be reset after a capture.  <b>0b0</b> Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSER[x] on capture.  <b>0b1</b> Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSER[x] on capture.	x
[0]	RPO	Reset performance counter. For each bit [x], if $x \geq \text{ext-CLUSTERPMU\_PMCR.N}$ , the number of implemented counters, then RP[x] is <b>RAZ/WI</b> . Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSER[x] are to be reset after a capture.  <b>0b0</b> Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSER[x] on capture.  <b>0b1</b> Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSER[x] on capture.	x

## Accessibility

This interface is accessible as follows:

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && SoftwareLockStatus()**

RO

**When IsCorePowered() && !DoubleLockStatus() && !OSLockStatus() && AllowExternalPMUAccess() && !SoftwareLockStatus()**

RW

**Otherwise**

ERROR

## B.2.4.36 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



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-392: ext\_clusterpmu\_pmdevaff0 bit assignments

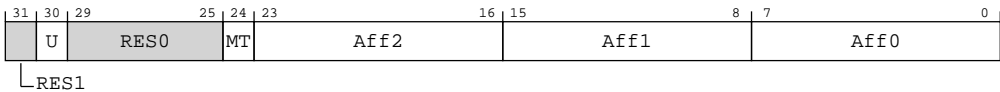


Table B-517: CLUSTERPMU\_PMDEVAFF0 bit descriptions

Bits	Name	Description	Reset
[31]	RES1	Reserved	RES1
[30]	U	Uniprocessor/Multiprocessor system. <b>0b0</b> 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. <b>0b0</b> 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. <b>0b10000000</b> Affinity with all cores in cluster.	0x80

Bits	Name	Description	Reset
[7:0]	Aff0	Affinity level 0.  0b00000000 Affinity with all core threads in cluster.	0x00

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.37 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

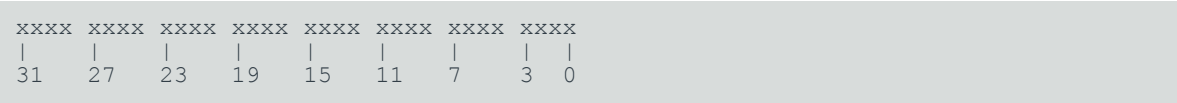
Register offset

0xFAC

Access type

See bit descriptions

Reset value



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-393: ext\_clusterpmu\_pmdevaff1 bit assignments

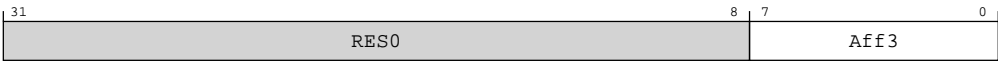


Table B-518: CLUSTERPMU\_PMDEVAFF1 bit descriptions

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

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.38 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

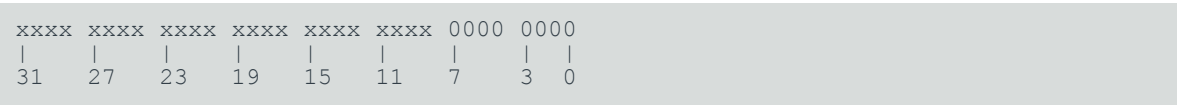
Register offset

0xFB8

Access type

See bit descriptions

Reset value





Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-394: ext\_clusterpmu\_pmauthstatus bit assignments

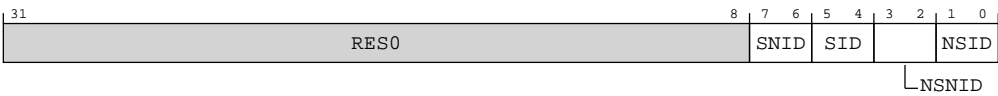


Table B-519: CLUSTERPMU\_PMAUTHSTATUS bit descriptions

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

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.39 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-395: ext\_clusterpmu\_pmdevarch bit assignments

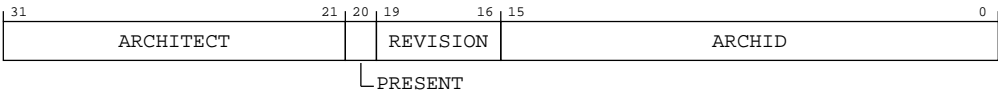


Table B-520: CLUSTERPMU\_PMDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. <b>0b01000111011</b> JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	0b01000111011
[20]	PRESENT	Present. <b>0b1</b> DEVARCH information present.	0b1
[19:16]	REVISION	Revision. <b>0b0000</b> Revision 0.	0b0000
[15:0]	ARCHID	Architecture ID. <b>0b0010101000010110</b> Processor Performance Monitor (PMU) architecture PMUv3.	0x2A16

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.40 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

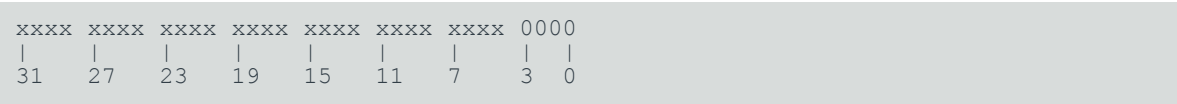
Register offset

0xFC8

Access type

See bit descriptions

Reset value



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-396: ext\_clusterpmu\_pmdevid bit assignments



Table B-521: 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

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.41 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.



Bit descriptions

Figure B-397: ext\_clusterpmu\_pmdevtype bit assignments

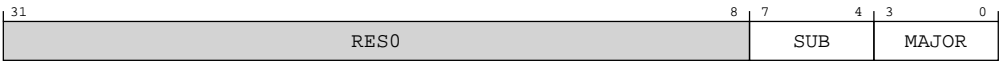


Table B-522: CLUSTERPMU\_PMDEVTYPE bit descriptions

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

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.42 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-398: ext\_clusterpmu\_pmpidr4 bit assignments

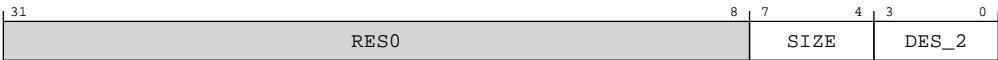


Table B-523: CLUSTERPMU\_PMPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. <b>0b0000</b> The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. <b>0b0100</b> Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.43 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-399: ext\_clusterpmu\_pmpidr0 bit assignments

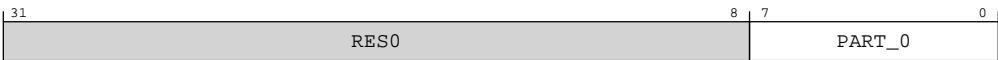


Table B-524: CLUSTERPMU\_PMPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. <b>0b11101100</b> DSU-120AE Cluster PMU. Bits [7:0] of part number 0x4EC.	0xEC

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.44 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-400: ext\_clusterpmu\_pmpidr1 bit assignments

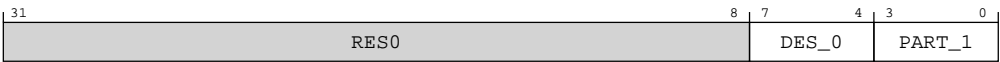


Table B-525: CLUSTERPMU\_PMPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0]. <b>0b1011</b> Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011

Bits	Name	Description	Reset
[3:0]	PART_1	Part number bits [11:8].  <b>0b0100</b> DSU-120AE Cluster PMU. Bits [11:8] of part number 0x4EC.	0b0100

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.45 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	0001	1011
31	27	23	19	15	11	7	3 0



Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-401: ext\_clusterpmu\_pmpidr2 bit assignments

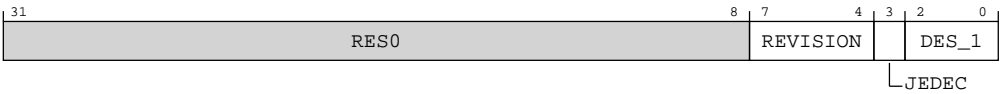


Table B-526: CLUSTERPMU\_PMPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVISION	Component major revision.  <b>0b0000</b> Component major revision 0.  <b>0b0001</b> Component major revision 1.  <b>0b0010</b> Component major revision 2.  <b>0b0011</b> Component major revision 3.  For DSU-120AE: <ul style="list-style-type: none"><li>Major revision 0 corresponds to r0p0.</li><li>Major revision 1 corresponds to r0p1.</li></ul>	0b0001
[3]	JEDEC	JEDEC assignee.  <b>0b1</b> JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4].  <b>0b011</b> Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.46 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



Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-402: ext\_clusterpmu\_pmpidr3 bit assignments

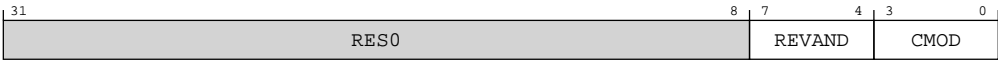


Table B-527: CLUSTERPMU\_PMPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVAND	Component minor revision. <b>0b0000</b> Component minor revision 0.	0b0000

Bits	Name	Description	Reset
[3:0]	CMOD	Customer Modified.  <b>0b0000</b> The component is not modified from the original design.	0b0000

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.47 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
31	27	23	19	15	11	7	3 0



Where the reset reads xxxx, see individual bits.



Bit descriptions

Figure B-403: ext\_clusterpmu\_pmcidr0 bit assignments

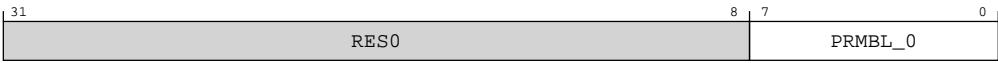


Table B-528: CLUSTERPMU\_PMCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	Preamble.  0b00001101 CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.48 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

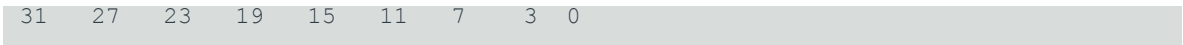
0xFF4

Access type

See bit descriptions

Reset value





Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-404: ext\_clusterpmu\_pmcidr1 bit assignments

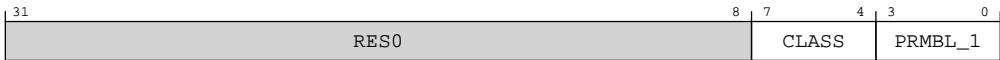


Table B-529: CLUSTERPMU\_PMCIDR1 bit descriptions

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

This interface is accessible as follows:

When IsCorePowered()

RO

Otherwise

ERROR

B.2.4.49 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

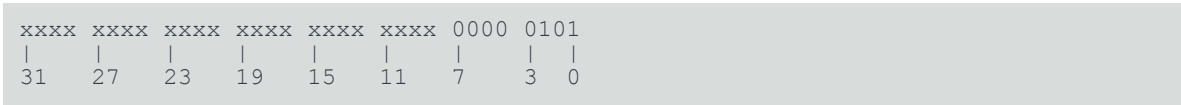
32

**Component**  
CLUSTERPMU

**Register offset**  
0xFF8

**Access type**  
See bit descriptions

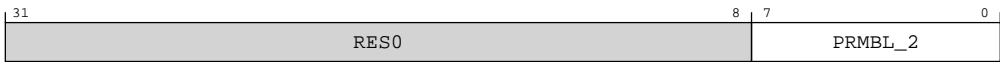
**Reset value**



Where the reset reads xxxx, see individual bits.

**Bit descriptions**

**Figure B-405: ext\_clusterpmu\_pmcidr2 bit assignments**



**Table B-530: 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**  
This interface is accessible as follows:

**When IsCorePowered()**  
RO

**Otherwise**  
ERROR

### B.2.4.50 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



Where the reset reads xxxx, see individual bits.

#### Bit descriptions

Figure B-406: ext\_clusterpmu\_pmcidr3 bit assignments

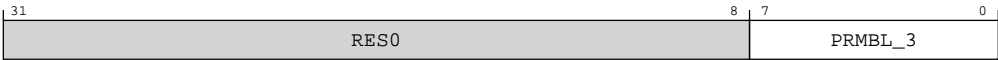


Table B-531: CLUSTERPMU\_PMCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_3	Preamble. <b>0b10110001</b> CoreSight component identification preamble.	0xB1

#### Accessibility

This interface is accessible as follows:

## When IsCorePowered()

RO

## Otherwise

ERROR

# Proprietary Notice

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

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

The content of this document is informational only. Any solutions presented herein are subject to changing conditions, information, scope, and data. This document was produced using reasonable efforts based on information available as of the date of issue of this document. The scope of information in this document may exceed that which Arm is required to provide, and such additional information is merely intended to further assist the recipient and does not represent Arm's view of the scope of its obligations. You acknowledge and agree that you possess the necessary expertise in system security and functional safety and that you shall be solely responsible for compliance with all legal, regulatory, safety and security related requirements concerning your products, notwithstanding any information or support that may be provided by Arm herein. In addition, you are responsible for any applications which are used in conjunction with any Arm technology described in this document, and to minimize risks, adequate design and operating safeguards should be provided for by you.

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

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

Reference by Arm to any third party's products or services within this document is not an express or implied approval or endorsement of the use thereof.

This document consists solely of commercial items. You shall be responsible for ensuring that any permitted use, duplication, or disclosure of this document complies fully with any relevant

export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of this document shall prevail.

The validity, construction and performance of this notice shall be governed by English Law.

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

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

PRE-1121-V1.0

# Product and document information

Read the information in these sections to understand the release status of the product and documentation, and the conventions used in the Arm documents.

## Product status

All products and Services provided by Arm require deliverables to be prepared and made available at different levels of completeness. The information in this document indicates the appropriate level of completeness for the associated deliverables.

### Product completeness status

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

### Product revision status

This product is rOp1, which indicates the revision status of the product described in this manual, where:

- r (value)**Identifies the major revision of the product, for example, r1.
- p (value)**Identifies the minor revision or modification status of the product, for example, p2.

## Revision history

These sections can help you understand how the document has changed over time.

### Document release information

The Document history table gives the issue number and the released date for each released issue of this document.

#### Document history

Issue	Date	Confidentiality	Change
0001-03	13 December 2024	Non-Confidential	First release for rOp1
0000-02	13 March 2024	Non-Confidential	Second early access release for rOp0
0000-01	16 November 2023	Confidential	First early access release for rOp0

The Change history tables describe the technical changes between released issues of this document in reverse order.



**Table 2: Issue 0000-01**

Change	Location
First early access release for r0p0	-

**Table 3: Differences between issue 0000-01 and issue 0000-02**

Change	Location
Second early access release for r0p0	-
Editorial changes	Throughout the document
Updated product name	Throughout the document

**Table 4: Differences between issue 0000-02 and issue 0001-03**

Change	Location
First release for r0p1	-
Editorial changes	Throughout the document

## 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](https://developer.arm.com/glossary).

### Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
<i>italic</i>	Citations.
<b>bold</b>	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 <u>underline</u>	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:  <pre>MRC p15, 0, &lt;Rd&gt;, &lt;CRn&gt;, &lt;CRm&gt;, &lt;Opcode_2&gt;</pre>
<b>SMALL CAPITALS</b>	Terms that have specific technical meanings as defined in the <i>Arm® Glossary</i> . For example, <b>IMPLEMENTATION DEFINED</b> , <b>IMPLEMENTATION SPECIFIC</b> , <b>UNKNOWN</b> , and <b>UNPREDICTABLE</b> .



We recommend the following. If you do not follow these recommendations your system might not work.

---



Your system requires the following. If you do not follow these requirements your system will not work.

---



You are at risk of causing permanent damage to your system or your equipment, or of harming yourself.

---



This information is important and needs your attention.

---



This information might help you perform a task in an easier, better, or faster way.

---



This information reminds you of something important relating to the current content.

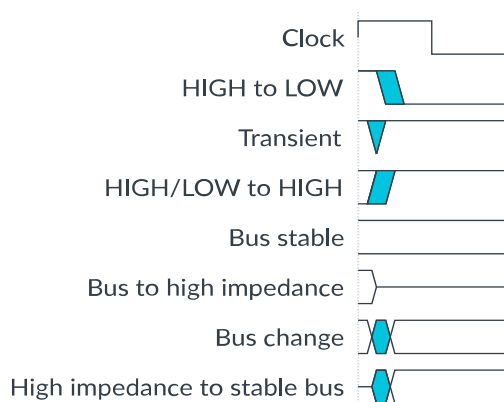
---

## 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: 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.

# Useful resources

This document contains information that is specific to this product. See the following resources for other useful information.

Access to Arm documents depends on their confidentiality:

- Non-Confidential documents are available at [developer.arm.com/documentation](https://developer.arm.com/documentation). Each document link in the following tables goes to the online version of the document.
- Confidential documents are available to licensees only through the product package.

Arm product resources	Document ID	Confidentiality
<a href="#">Arm® Corelink® PCK-600 Power Control Kit Technical Reference Manual</a>	101150	Non-Confidential
<a href="#">Arm® CoreSight™ DAP-Lite2 Technical Reference Manual</a>	100572	Non-Confidential
<a href="#">Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual</a>	100806	Non-Confidential
<a href="#">Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual</a>	101088	Non-Confidential
<a href="#">Arm® DynamIQ™ Shared Unit-120AE Configuration and Integration Manual</a>	107720	Confidential

Arm architecture and specifications	Document ID	Confidentiality
<a href="#">AMBA® 5 CHI Architecture Specification</a>	IHI 0050	Non-Confidential
<a href="#">AMBA® APB Protocol Specification</a>	IHI 0024	Non-Confidential
<a href="#">Arm® Architecture Reference Manual for A-profile architecture</a>	DDI 0487	Non-Confidential
<a href="#">Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification</a>	IHI 0099	Non-Confidential
<a href="#">AMBA® ATB Protocol Specification</a>	IHI 0032	Non-Confidential
<a href="#">AMBA® AXI Protocol Specification</a>	IHI 0022	Non-Confidential
<a href="#">Arm® CoreSight™ Architecture Specification v3.0</a>	IHI 0029	Non-Confidential
<a href="#">Arm® Embedded Trace Macrocell Architecture Specification ETMv4.0 to ETM4.6</a>	IHI 0064	Non-Confidential
<a href="#">Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4</a>	IHI 0069	Non-Confidential
<a href="#">AMBA® Low Power Interface Specification</a>	IHI 0068	Non-Confidential
<a href="#">Arm® Power Control System Architecture</a>	DEN 0050	Non-Confidential
<a href="#">Arm® Power Policy Unit Architecture Specification</a>	DEN 0051	Non-Confidential

Non-Arm resources	Document ID	Organization
-	-	-