arm

Arm® C1-DynamIQ™ Shared Unit

Revision r0p2

Technical Reference Manual

Non-Confidential

Issue 05

107804_0002_05_en

Arm® C1-DynamIQ™ Shared Unit Technical Reference Manual

This document (107804_0002_05_en) was issued on 2025-09-10. There might be a later issue at https://developer.arm.com/documentation/107804

The product revision is r0p2.

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 an Arm core.

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.

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

1. The C1-DynamIQ™ Shared Unit

The C1-DynamIQ™ Shared Unit (C1-DSU) provides a shared L3 memory system, snoop control and filtering, and other control logic. The C1-DSU is used to support a cluster of A-class architecture cores and associated C1-Scalable Matrix Extension 2 units. The cluster is called the C1-DSU cluster. All the external interfaces to the System on Chip (SoC) are provided through the C1-DSU.

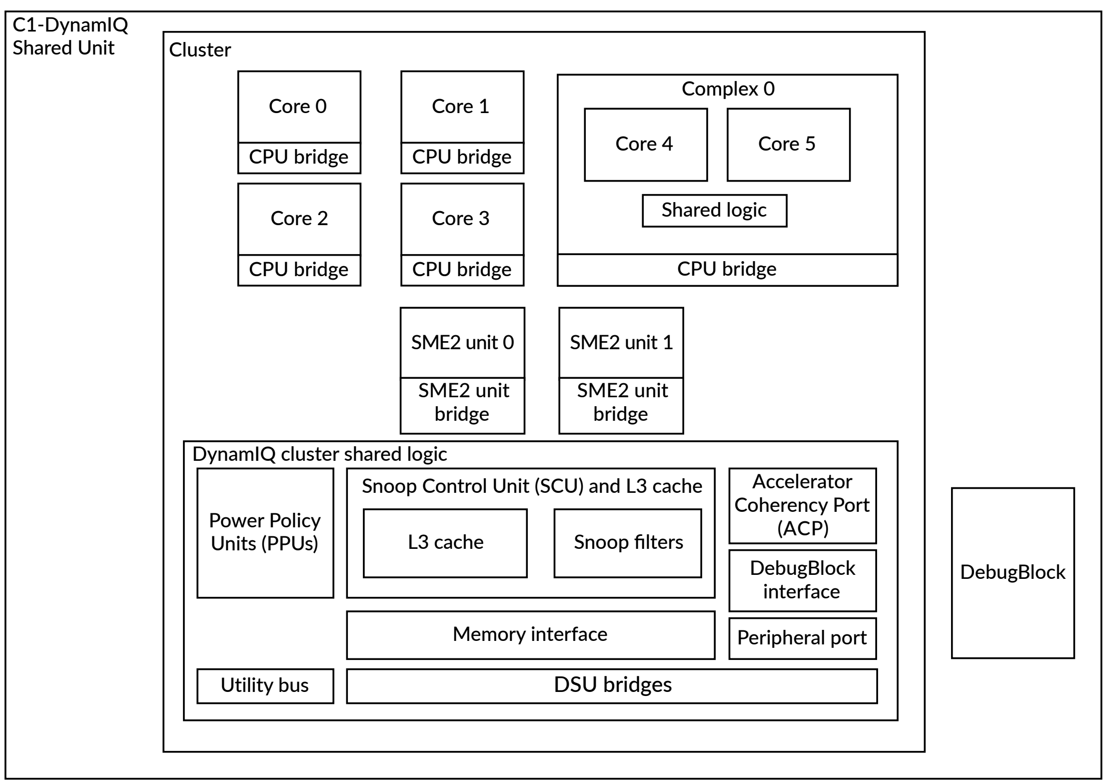
Note

In this book, the following naming conventions apply:

- The C1-DynamIQ™ Shared Unit is referred to as C1-DSU.
- The C1-DSU cluster is referred to as a cluster in cases where distinguishing between the C1-DSU cluster and C1-DSU is not important to the context.
- The C1-Scalable Matrix Extension 2 is referred to as SME2 unit.

The following figure shows an example of a C1-DSU-based cluster.

Figure 1-1: C1-DSU cluster



Note

Some types of core can only be instantiated as a single instance in the cluster in a special type of configuration called Direct connect. In this configuration, no SME2 units are supported in the cluster. For more information on Direct connect, see 1.3.2 L3 memory system variants on page 23.

A C1-DSU cluster consists of between one and 14 cores, with up to three 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 C1-DSU cluster also supports complexes where typically two cores are linked together and share logic. Examples of shared logic include a floating-point unit and an L2 cache. For more information on complexes, see 1.3.1 What is a complex? on page 22.

Note

The C1-DSU cluster supports a type of Streaming Mode Compute Unit (SMCU) as defined in Arm® Architecture Reference Manual for A-profile architecture called a C1-Scalable Matrix Extension 2 unit. In this book this is referred to by its generic name SME2 unit. Optionally, the C1-DSU supports up to 2 of these units instantiated in the cluster. These units act as a shared resource for all the cores in cluster for accelerating specific matrix operations. Each unit implements the Scalable Matrix Extension (SME) architecture, the Scalable Matrix Extension 2 (SME 2) architecture, and the Arm®v9.3-A architecture.

For further information on the SME2 unitsee the following:

- For features and behavior of the SME2 unit, see the *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*.
- For SME2 unit macrocell implementation, see *Arm® C1-Scalable Matrix Extension 2 Configuration and Integration Manual*.

Some cores might require that the SME2 unit is present when configuring the core and cluster. To check this requirement, see the *DSU dependent features* section in your licensed core *Technical Reference Manual*.

All cores in the C1-DSU cluster, including those in complexes, are coherently connected to an L3 memory system that includes an L3 cache and a Snoop Control Unit (SCU). The SCU maintains coherency between caches in the cores and the L3 cache, and includes a snoop filter to optimize coherency maintenance operations. The shared L3 cache simplifies process migration between the cores. The SCU also maintains full coherency between the cores and the SME2 units.

The C1-DSU 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 C1-DSU cluster supports many mechanisms to reduce static and dynamic power dissipation. For example, placing the cores and L3 cache into retention and powering down parts of the L3 cache.

All the external interfaces including those to the cores are provided through the C1-DSU 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 requester interface. The Accelerator Coherency Port (ACP) provides coherent access for non-cached requesters 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 C1-DSU that provides the interface for debug capability. The DebugBlock is instanced as a separate unit for supporting debug over powerdown.

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

Note

For information on the behavior and features of your core, including whether your core is supported in a complex, see the Technical Reference Manual (TRM) for your core.

1.1 C1-DynamIQ™ Shared Unit features

Some features in the C1-DSU are fixed and some features are optional. You can configure optional features in the RTL during build time configuration, to meet your requirements.

Cache features

The C1-DSU 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 C1-DSU 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 C1-DSU has the following cluster features:

- Support for Arm®v9.3-A architecture cores.
- Support for Scalable Matrix Extension (SME) architecture and the Scalable Matrix Extension 2 (SME2) architecture.. The cluster supports up to 2 instances of the SME2 unit.
- Support for Realm Management Extension (RME), in Direct Connect configurations only.
- Support for up to three types of core, and a maximum of 14 cores in the cluster.
- The C1-DSU 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, the cores, and the SME2 units.
- 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 C1-DSU has the following interface features:

- For Direct connect configurations, support for AMBA 5 CHI Issue G.
- 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 PPUs, and other system components.
- Optional peripheral port interface that is implemented as either an AXI 64-bit wide port, AXI 256-bit wide port, or CHI 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.

Note

If RME is supported in the C1-DSU, the bus requester interface uses the CHI Issue G protocol, and the utility bus interface uses the AXI Issue J protocol. For more information on RME support in C1-DSU, see 1.4.1 Realm management extension on page 26.

Debug and trace features

The C1-DSU 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

Note

The ELA-600 is licensed separately.

1.2 C1-DynamIQ™ Shared Unit configuration options

You must configure the C1-DSU RTL for your implementation requirements prior to hardware synthesis at build time configuration. Configuration for the C1-DSU is carried out together with configuration for the cores, and optionally, the SME2 units in your cluster.

Note

For a complete list of the configuration parameters and guidelines, see *RTL configuration process* in *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

Note

The *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual* is a confidential document that is available with the appropriate product license.

The C1-DSU implementation options include:

Number of cores

You can configure the cluster to have between one 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 three different types of cores. See 1.3 Cluster configurations on page 18, for more information on the types of core that are supported.

Number of SME2 units

Optionally, you can configure the cluster to have either one or two SME2 units.

Note

The SME2 unit is not supported when the cluster is in a Direct connect configuration.

Direct connect

You can configure the cluster for Direct connect memory system variant. For more information on Direct connect, see 1.3.2 L3 memory system variants on page 23.

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

Note

Setting the size of 0KB implements the C1-DSU without an L3 cache, see *L3 cache not present* in 1.3.2 L3 memory system variants on page 23.

L3 cache slices

You can configure the C1-DSU 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 129.

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 C1-DSU transport, see *RTL configuration process* in *Arm® C1-DynamIQ™ Shared Unit 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 C1-DSU 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.

SCU cache protection

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

Timing closure

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

ELA

Include support for integrating the CoreSight Embedded Logic Analyzer (ELA)-600 into the C1-DSU.

Note

The ELA-600 is a separately licensable product.

1.3 Cluster configurations

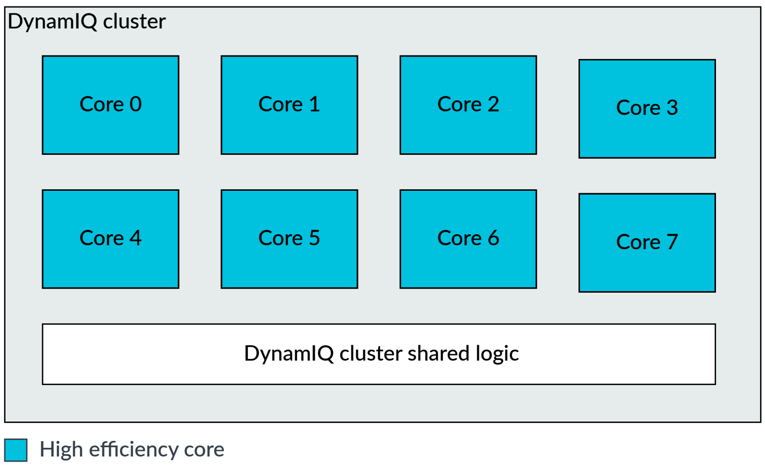
A cluster can be configured with up to three different types of cores in the same cluster. Each core type targeting different power efficiency and performance levels. This arrangement allows for an intermediate core that has an intermediate performance and efficiency level. The cluster also supports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

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

- One or more cores of the same type.
- Various arrangements of two types of cores. For example, one or more cores targeting either a high-performance level or a higher power efficiency level.
- Various arrangements of three 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 1.3.1 What is a complex? on page 22.
- One or more complexes and individual cores.
- At least one core or complex, with one or two SME2 units.

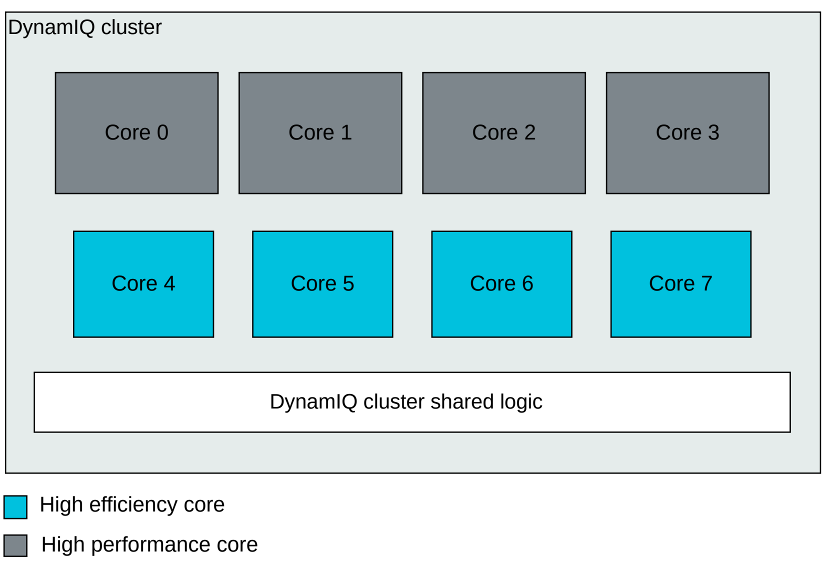
The following figure shows a cluster that is configured with all the same type of core.

Figure 1-2: DynamIQ cluster with one type of core



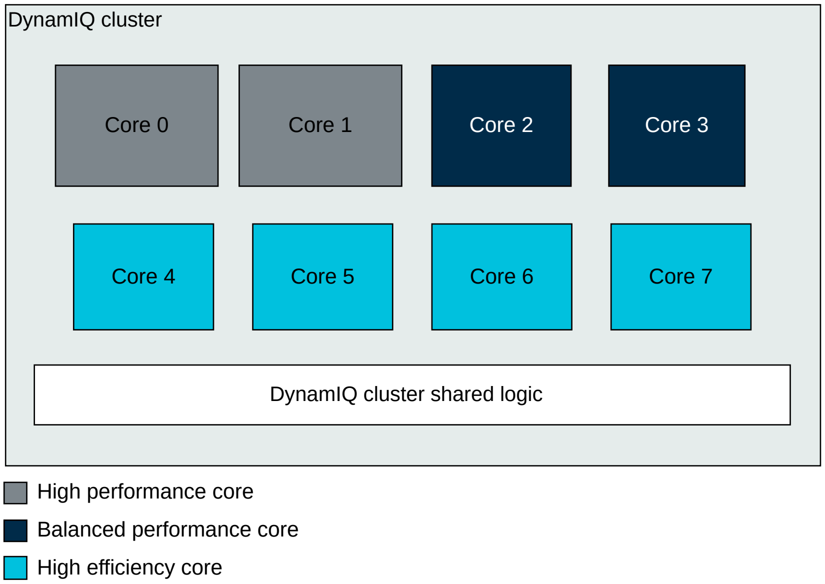
The following figure shows a cluster that is configured with two types of core.

Figure 1-3: DynamIQ cluster with two types of cores



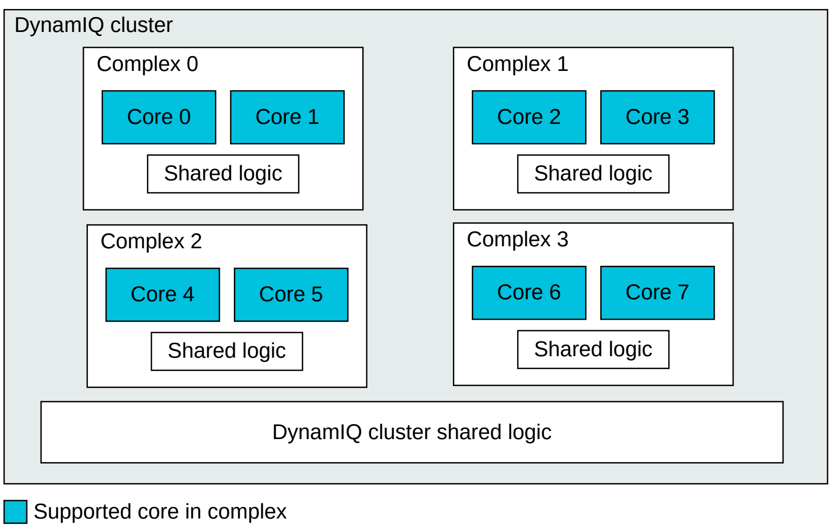
The following figure shows a cluster that is configured with three types of core.

Figure 1-4: DynamIQ cluster with three types of cores



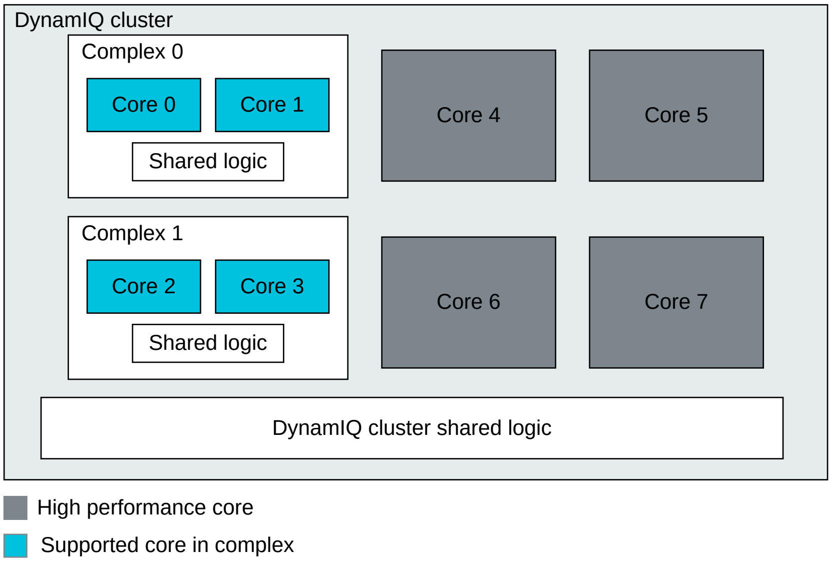
The following figure shows a cluster that is configured with four complexes.

Figure 1-5: DynamIQ cluster with four complexes



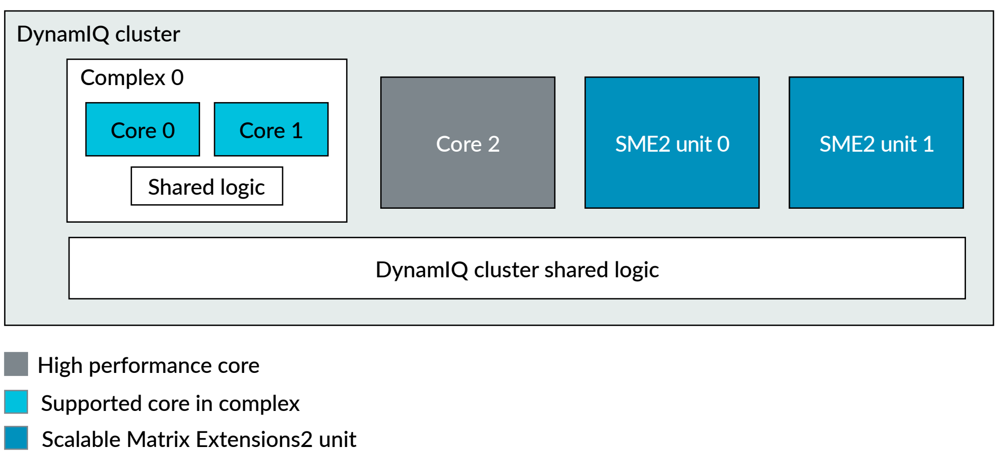
The following figure shows a cluster that is configured with two complexes, and four individual cores.

Figure 1-6: DynamIQ cluster with two complexes and four high-performance cores



The following figure shows a cluster that is configured with one complex, one individual core, and 2 SME2 units.

Figure 1-7: DynamIQ cluster with one complex, one core, and 2 SME2 units



Note

All C1-DSU-compatible cores support use in a multi-core cluster. Any combination of these cores should be configurable in the cluster provided that:

- The total number of cores does not exceed 14.
- There are no more than three different types of core in the cluster.

For any combinations that are specifically not permitted see the *Arm® C1-DynamIQ™ Shared Unit (MP200) Release Note*

Not all cores support use in a multi-core cluster. For example, some types of core might only support a Direct connect configuration, so that only a single core can be instantiated in the cluster. For information about the type of core you have licensed, see your core Technical Reference Manual.

1.3.1 What is a complex?

The C1-DSU 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.

Note

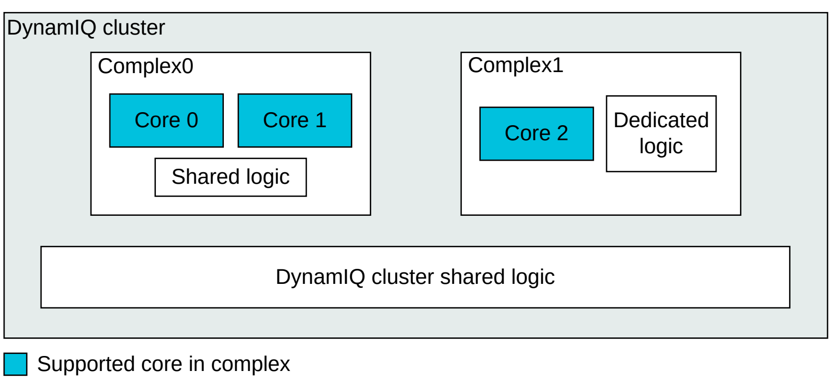
Only certain types of cores which have a merged-core microarchitecture can be used in a complex. To see if your core is supported in a complex and for further details of complexes, see your core Technical Reference Manual (TRM).

The maximum number of cores instantiated in the cluster is 14. This number includes:

- Standalone cores that are not instantiated in a complex.
- 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 1-8: Cluster with a dual-core complex and a single-core complex



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

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

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

Note

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

1.3 Cluster configurations on page 18
1.7 Core, complex, SME2 unit, and processing element numbering on page 28

1.3.2 L3 memory system variants

By default the C1-DSU is implemented with an L3 cache. Depending on your requirements, you can instead implement the C1-DSU without an L3 cache.

Alternatively you can implement the C1-DSU to support a Direct connect connection to your core if your core supports this.

Note

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

There are three possible L3 memory system implementations:

L3 cache present

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

L3 cache not present

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

This variant allows multiple cores in the cluster and manages the coherency between them. It supports other implementation options such as Accelerator Coherency Port (ACP), Peripheral Port, and AXI or CHI 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.

Direct connect

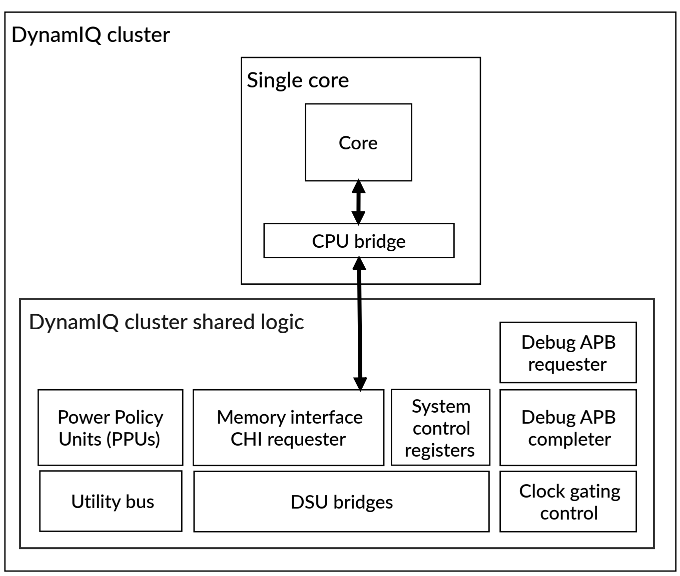
In the Direct connect implementation, the L3 cache, snoop filter, and SCU logic are not present.

This variant is specifically for use with a CHI interconnect. It offers extra area savings and reduced latency when compared to the previous variants. Because there is no L3 cache in the cluster, this variant relies on the system cache in the interconnect for performance. To check if your core supports this variant, see the *C1-DSU dependent features* section in your core TRM.

Because this variant does not include any coherency logic, it is only supported when there is a single core in the cluster. When the C1-DSU is configured for Direct connect, optional interfaces such as ACP or the peripheral port are not supported. See the *RTL configuration process* chapter of the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual* for more information. The requester port must be a single 256-bit CHI interface.

The following diagram shows the C1-DSU implemented with Direct connect.

Figure 1-9: L3 Direct connect implementation



For more information on how to implement the C1-DSU with one of the L3 memory system variants, see the *Configuration Guidelines* chapter in *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

Related information

1.2 C1-DynamIQ Shared Unit configuration options on page 16
1.1 C1-DynamIQ Shared Unit features on page 14

1.4 Supported standards and specifications

The C1-DSU complies with the Arm®v9.3-A architecture and all previous Arm®v8-A architectures up to Arm®v8.8-A.

Note

The C1-DSU 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 C1-DSU complies with the architectures listed in the following table.

Table 1-1: Standards and specifications support in the C1-DSU

Standard of specification	Version	Notes
Arm architecture	Arm®v9.3-A	The C1-DSU supports cores based on the Arm®v9.3-A architecture. These cores also support previous Arm®v8-A architectures up to Arm®v8.8-A, dependent on the core. See the section *Supported standards and specifications* in your core Technical Reference Manual (TRM) for details.
FEAT_RAS, Reliability, Availability, and Serviceability (RAS)	RAS v8.4	The C1-DSU supports RAS features that conform to the v8.4 RAS architecture, see 11. RAS extension support on page 186.
FEAT_RME, Realm Management Extension (RME)	Arm®v9.3-A	The C1-DSU supports RME, see 1.4.1 Realm management extension on page 26.

Standard of specification	Version	Notes
Advanced Microcontroller Bus Architecture (AMBA)	• AMBA 5 CHI Issue E • AMBA 5 CHI Issue G for Direct connect configurations only • AMBA AXI5 Issue H • AMBA AXI5 Issue J, only if RME is supported • AMBA APB5 Issue D If RME is supported, AMBA APB5 Issue E protocol is used.	For more information on what AMBA protocols the C1-DSU interfaces support, see 2.4 Interfaces on page 38.
CoreSight™ architecture	v3.0	For more information on CoreSight™ architecture, see the *Arm® CoreSight™ Architecture Specification v3.0*.
Debug	Arm®v9.3-A	Arm®v9.3-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 *Arm® Architecture Reference Manual for A-profile architecture* for information on this architectural feature.
FEAT_GICv4p1, Generic Interrupt Controller (GIC) architecture CPU interface and Stream Protocol interface.	GICv4.1	The C1-DSU 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 *Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4*.
Performance Monitoring Unit (PMU)	PMUv3	-

Related information

2.2 C1-DSU cluster shared logic components on page 33
2.4 Interfaces on page 38

1.4.1 Realm management extension

The C1-DSU can use the Realm Management Extension (RME) for a core that supports RME, provided that the cluster is configured in Direct connect and the input signal LEGACYTZEN is LOW.

Throughout this book, where reference is made to the phrase, RME supported, for example when referring to what bus protocols are supported, this indicates that:

- The cluster is in Direct connect.
- The core in the cluster supports RME.

Throughout this book, where reference is made to the phrase, RME enabled, for example when referring to register access, this indicates that:

- RME is supported
- The input signal LEGACYTZEN is LOW.

If any of the previous conditions are not satisfied, then RME cannot be used. For example, a core could be configured for Direct connect, and support RME, but if the signal LEGACYTZEN is HIGH, then the C1-DSU reverts to TrustZone security.

For more information on RME, see *Arm® Architecture Reference Manual for A-profile architecture*.

1.5 Test features

The C1-DSU 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 C1-DSU. Additionally, internal Memory Built-In Self Test (MBIST) interfaces are provided to test the L3 cache and other memory arrays of the C1-DSU.

The C1-DSU includes an ATPG test interface that provides signals to control the Design for Test (DFT) features of the C1-DSU, the cores, and the SME2 units 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 C1-DSU 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*. For information about the test signals related to your core, see your core *Configuration and Integration Manual*.

1.6 Design Tasks

Both the C1-DSU and the cores in the cluster are delivered as synthesizable RTL descriptions in Verilog HDL. Before you can use the C1-DSU, SME2 unit, 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 C1-DSU, the SME2 unit, 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 C1-DSU 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 C1-DSU 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 C1-DSU, the SME2 unit, and the cores in the cluster by programming values into registers. The configuration choices affect the behavior of the C1-DSU 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*
- See *Physical implementation* in the *Arm® C1-Scalable Matrix Extension 2 Configuration and Integration Manual*

1.7 Core, complex, SME2 unit, and processing element numbering

A cluster contains one or more cores. The cluster can also contain one or more complexes which can be made up of either a single core or two cores. Additionally the cluster can contain one or two SME2 units. Because certain parts of the design, such as signal names and register bit values, depend on the number of cores, complexes, and SME2 units within the cluster, a numbering system has been created.

Throughout this document, the following numbering is used for cores, Processing Elements (PEs), complexes, and SME2 units:

Core

The total number of cores in the cluster is equal to CN. This number also includes cores instantiated in a complex. For example, CN = 6 for cluster comprised of two dual-core complexes and two standalone cores.

For core instantiation, the instance numbering starts from 0. For individual core instances the term y is used, which ranges from zero to CN-1. For example, when referring to the second instance of a core, y = 1. The term y is called the core instance number.

Complexes

The total number of complexes in the cluster is equal to CX. This number includes both single-core complexes and dual-core complexes. For example, CX = 2 for a cluster comprised of one dual-core complex and one single-core complex.

For complex instantiation, the instance numbering starts from 0. For individual complex instances, the term x is used, which ranges from 0 to CX-1. The term x is called the complex instance number.

Processing element

The Arm architecture allows for cores to be single, or multi-threaded. A processing element performs a thread of execution. A single-threaded core has one processing element and a multi-threaded core has two or more processing elements. The total number of processing elements in the cluster is equal to PE.

Note

In the current C1-DSU, each core only has one PE. Therefore, PE = CN. There are no multithreaded cores in the cluster. Therefore, each core has only one PE. Therefore PE = CN. For example, PE = CN = 5 for a cluster comprised of two dual-core complexes and one standalone core.

SME2 units

The numbering of SME2 unit instances in the cluster ranges from zero to CM, where CM has the value of total number of SME2 units minus 1. Therefore, CM can be either be 0 or 1. For individual SME2 unit instances, the term c is used, which ranges from zero to CM. For example, when referring to the second instance of an SME2 unit, c = 1. The term c is called the SME2 unit instance number.

When considering the programming of the SME2 units through the utility bus, the SME2 unit is addressed as if it were an additional core to the cluster. Therefore, this is also referenced by <y> (which now represents core and SME2 unit instance number), where y is given: * y = CN + 1 for SME2 unit instance 0, and * y = CN + 2 for SME2 unit instance 1.

Note

This is applicable for programming purposes only. For other use cases, the y value represents the usual definition and c is used as the SME2 unit instance number.

For more information on the instance numbering for cores, complexes, and SME2 units in the cluster, see *RTL configuration process* in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

1.8 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 limited access release
r0p1	Support for optional user bits on the AW and AR channels of the Accelerator Coherency Port
r0p2	Maintenance release

Changes in functionality that have an impact on the documentation also appear in Revision history on page 1046.

2. Technical overview

A C1-DSU cluster-based system is referred to as C1-DSU in the remainder of the document.

The C1-DSU comprises two top-level modules, these are:

A module to form a C1-DSU cluster

This module includes the cores, complexes, SME2 units, and the C1-DSU cluster shared logic.

A separate module for the DebugBlock

Separating the debug components from the C1-DSU 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 powerdown.

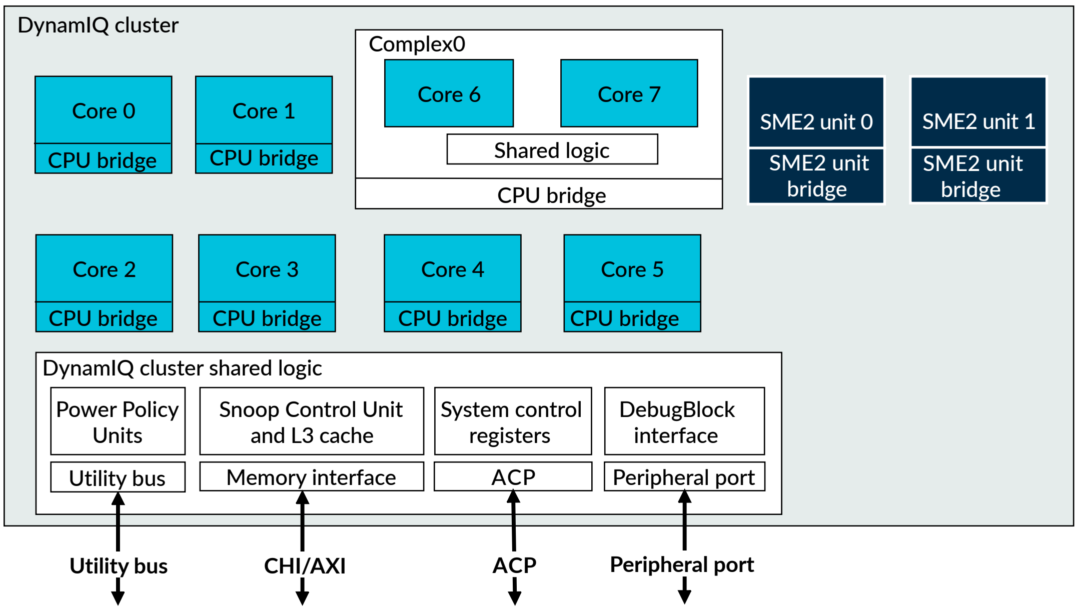
All the main System on Chip (SoC) interfaces appear at the top level of the C1-DSU. The C1-DSU connects the cores, complexes, and SME2 units to an external memory system and the rest of the SoC.

2.1 DynamIQ cluster components

The C1-DSU cluster contains all the cores, complexes, and optionally, the SME2 units together with the C1-DSU cluster shared logic. All the C1-DSU cluster shared logic is automatically connected to the cores, complexes, and SME2 units by the configuration script during build-time configuration.

The following figure shows the main components that make up the C1-DSU cluster within the C1-DSU.

Figure 2-1: C1-DSU cluster components



Cores

There can be up to 14 cores in the C1-DSU cluster, with up to three different types of core. For information on the behavior and features of each core, see the Technical Reference Manual (TRM) of each core.

Complexes

The C1-DSU cluster supports complexes as follows:

- Up to 14 single-core complexes
- Up to seven dual-core complexes

Complexes are made up of specialized cores. To determine if your core is supported in a complex, see the *C1-DSU dependent features* section in your core TRM. For more information about complexes , see 1.3.1 What is a complex? on page 22.

SME2 units

The C1-DSU also supports up to 2 SME2 units. For information on the behavior and features of the SME2 unit, see the *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*.

C1-DSU cluster shared logic

The C1-DSU cluster shared logic forms part of the C1-DSU cluster. See 2.2 C1-DSU cluster shared logic components on page 33.

Related information

1.3 Cluster configurations on page 18
1.3.1 What is a complex? on page 22
2.2 C1-DSU cluster shared logic components on page 33
2.3 DebugBlock components on page 37

2.1.1 Integration of the cores in the cluster

When you implement a C1-DSU cluster, all interfacing between the cores, complexes, and the C1-DSU is implemented automatically. All the external signal inputs and outputs pass through the C1-DSU. The C1-DSU 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 C1-DSU L3 memory system. Where necessary, the C1-DSU implements additional buffering to compensate for different clock rates of the core and C1-DSU L3 memory system.

Each core has an external clock interface, which is routed through the C1-DSU to the respective core.

2.1.2 Integration of the SME2 unit in the cluster

When you implement a C1-DSU, all interfacing between the SME2 units and the DSU logic and between the cores and the SME2 units is implemented automatically. Similar to the cores, the SME2 units can operate at different clock speeds. Each SME2 unit can be clocked at a different frequency if required. The DSU buffers and resynchronizes many signals to allow the SME2 units to be clocked at different speeds.

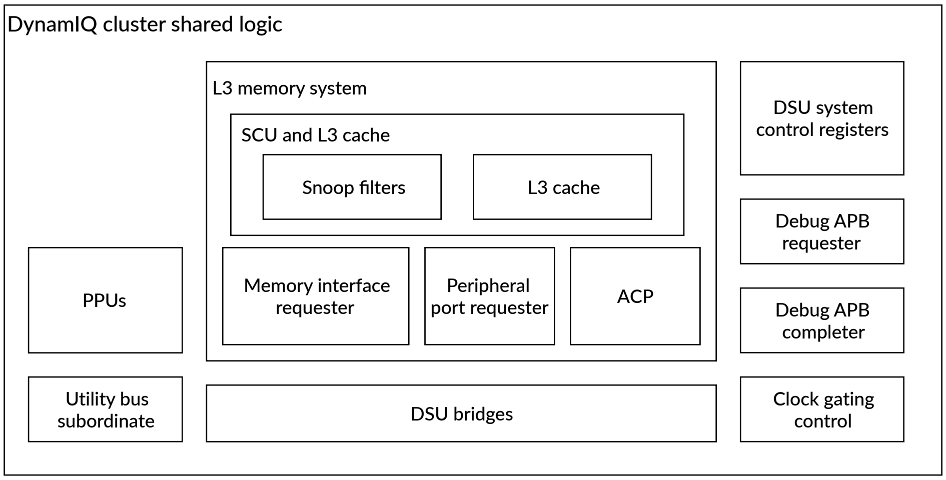
Each SME2 unit instance has an external clock interface, which is routed through the C1-DSU to the respective SME2.

2.2 C1-DSU cluster shared logic components

The C1-DSU cluster shared logic includes the following main components.

The following figure shows the components of the C1-DSU cluster shared logic.

Figure 2-2: C1-DSU cluster shared logic 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 the cores without having to read or write data to the L3 cache. These buffers can also handle direct cache to cache transfers between the L1 caches of the SME2 units. 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, and in the SME2 unit caches. Including the snoop filters means that the SCU does not need to request a look up in the core cache or the SME2 unit cache when it receives a coherent memory request. These snoop filters are accessed by the coherent requests from the other cores, SME2 units, 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 or SME2 unit. The size of the snoop filter is automatically determined from the configured number of cores and SME2 units, and the cache sizes in those cores and SME2 units.

Clock management

Clock gating is supported through Q-Channel requests from an external clock controller to the C1-DSU. 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
- CMEcCLK where c is the SME2 unit number
- GICCLK
- PCLK
- PERIPHCLK

- PPUCLK
- SCLK

L3 memory interfaces

Main memory requester

The main memory requester provides an interface between the C1-DSU 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. When the core is connected in Direct connect to the C1-DSU, the memory interface connection uses CHI Issue G instead of CHI Issue E. For details, see 7. CHI requester interface on page 131. 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 AXI5 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 the L3 caches. Up to two ACP interfaces can be configured, with each interface configured to be either 128-bit wide or 256-bit wide.

Peripheral port

The peripheral port is an optional requester interface and provides accesses to tightly coupled accelerators. 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
- Max Power Mitigation Mechanism (MPMM) registers in the cores and SME2 units
- 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

Size	Associativity
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 C1-DSU 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. You can program and access the PPUs from your System Control Processor (SCP) using the utility bus.

C1-DSU system control registers

The C1-DSU 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 C1-DSU
- L3 cache hit and miss count information

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

Debug and trace components

Each core includes an Embedded Trace Extension (ETE) to allow program tracing while debugging. The SME2 unit does not contain debug logic. The SME2 unit supports debug through the core it is connected to. For more information, see chapter 11, Debug, of the *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*.

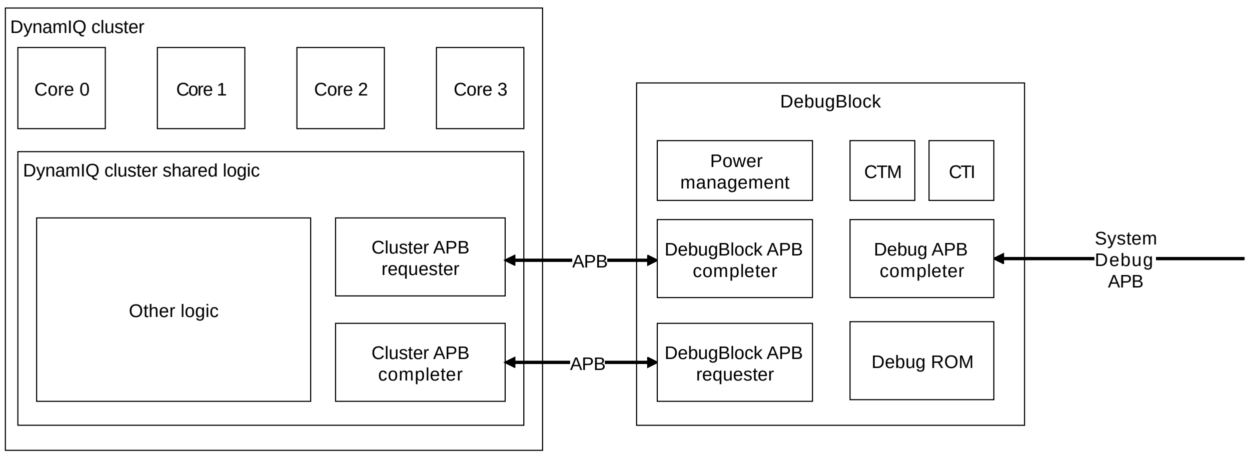
Trigger events from the cores are combined and output to the DebugBlock. Trigger events to the cores, and Debug register accesses, are received in the DebugBlock.

2.3 DebugBlock components

The DebugBlock is a dedicated debug component for the C1-DSU but is instanced 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 APB (requester) to DebugBlock APB (completer)

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

DebugBlock APB (requester) to cluster APB (completer)

Trigger events to the cores are transferred as APB writes to the C1-DSU. Register accesses from the system debug APB are transferred to the C1-DSU.

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. One additional CTI, referred to as the cluster CTI, is allocated to the cluster Performance Monitoring Unit (PMU) and the cluster Embedded Logic Analyzer (ELA) when present.

Note

- The cluster CTI is not present in a Direct connect configuration.
- There is a CTI for the SME2 unit, which is only used to drive triggers to the SME2 ELA. This CTI is always present, even if the ELA for the SME2 is not included.

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.

2.4 Interfaces

The C1-DSU manages all the external interfaces to the System on Chip (SoC) including those from the cores, SME2 units, and complexes in the cluster.

C1-DSU interfaces

The following figure shows the major external interfaces of the C1-DSU cluster.

The following table describes the external interfaces of the C1-DSU cluster.

Table 2-2: C1-DSU cluster interfaces

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

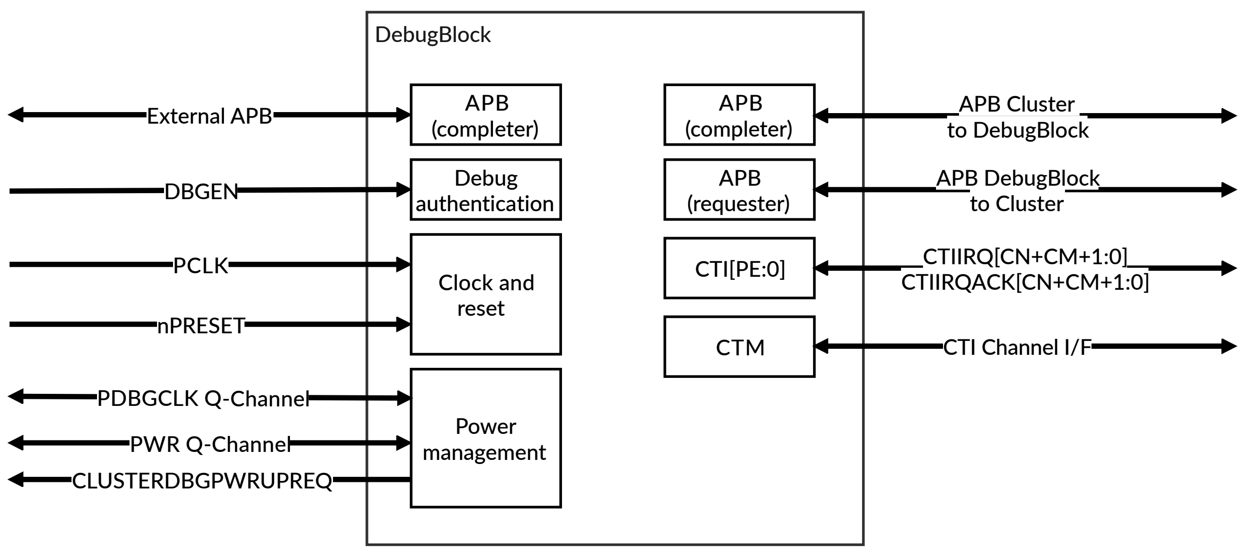
Purpose	Protocol	Notes
Memory	The bus requester interface uses either of the following protocols: • AMBA 5 CHI Issue E • AMBA 5 CHI Issue G. This protocol is only used when there is support for the Realm Management Extension (RME). For more information about support for RME, see 1.4.1 Realm management extension on page 26. • AMBA AXI5 Issue H is the protocol used when RME is not supported. • AMBA AXI5 Issue J is the protocol used when RME is supported.	Requester interface to main memory. You can configure the C1-DSU with either: • 1, 2, 3, or 4 CHI bus requester ports; or • 1, 2, 3, or 4 AXI bus manager ports For more information, see 1.2 C1-DynamIQ Shared Unit configuration options on page 16
Accelerator Coherency Port (optional)	AMBA ACE5-Lite	Subordinate interface allowing an external manager to make coherent requests to cacheable memory. You can configure the C1-DSU 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	The utility bus uses either of the following protocols: • If RME is not supported, AMBA AXI5 Issue H • If RME is supported, AMBA AXI5 Issue J	Memory-mapped port for accessing the following: • Power Policy Units (PPUs) • Activity monitors • Max Power Mitigation Mechanism (MPMM) registers • RAS registers • Memory System Resource Partitioning and Monitoring (MPAM) registers
Cluster to DebugBlock	The cluster to DebugBlock connection uses either of the following protocols: • If RME is not supported, AMBA APB5 Issue D • If RME is supported, AMBA APB5 Issue E	APB interface from the cluster (requester) to the DebugBlock (completer)
DebugBlock to cluster	The DebugBlock to cluster connection uses either of the following protocols: • If RME is not supported, AMBA APB5 Issue D • If RME is supported, AMBA APB5 Issue E	APB interface from the DebugBlock (requester) to the cluster (completer)
Power state control	P-Channel	P-Channels for C1-DSU and core power management

Purpose	Protocol	Notes
Clock state control	Q-Channel	Q-Channels for clock gating control
Wait For Event (WFE) event signaling	-	Signals for Wait For Event (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 Automatic Test Pattern Generation (ATPG) scan-path testing.
Memory Built-In Self Test (MBIST)	Arm MBIST	Internal interface that supports the manufacturing test of the L3 cache and Snoop Control Unit (SCU) memories embedded in the C1-DSU. Each core has its own internal MBIST interface.

DebugBlock interfaces

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

Figure 2-4: DebugBlock interfaces



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

Table 2-3: DebugBlock interfaces

Purpose	Protocol	Notes
External debug	The external debug interface uses either of the following protocols: • If RME is not supported, AMBA APB5 Issue D • If RME is supported, AMBA APB5 Issue E	Completer interface to external debug component, for example a Debug Access Port (DAP). It allows access to Debug registers and resources.

Purpose	Protocol	Notes
Cluster to DebugBlock	The cluster to DebugBlock connection uses either of the following protocols: • If RME is not supported, AMBA APB5 Issue D • If RME is supported, AMBA APB5 Issue E	APB interface from the cluster (requester) to the DebugBlock (completer).
DebugBlock to cluster	The DebugBlock to cluster connection uses either of the following protocols: • If RME is not supported, AMBA APB5 Issue D • If RME is supported, AMBA APB5 Issue E	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 43
7. CHI requester interface on page 131
9. ACP interface on page 160
8. AXI manager interface on page 148
10. AXI or CHI peripheral port on page 170
14. Debug on page 206

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 C1-DSU. 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. PBHA 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.

Note

For information on PBHA signals, see *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

Related information

1.2 C1-DynamIQ Shared Unit configuration options on page 16

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.

3. Clocks and resets

The C1-DSU has separate clock signals for each of the standalone cores (those cores not in a complex), SME2 units, and for each complex. There are also clocks for the internal logic, and some of the external interfaces of the C1-DSU.

The C1-DSU 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, and SME2 units can also occur due to power state changes driven from the Power Policy Units (PPUs).

3.1 Clocks

The C1-DSU has a separate clock signal for each standalone core, complex, or SME2 unit. There are also separate clocks for the internal logic, and some of the external interfaces.

The following table describes the clock signals of the C1-DSU.

Table 3-1: C1-DSU clock signals

Signal	Description
COREyCLK	The clocks for each of the cores in the cluster that are not part of a complex. These signals clock all core logic, including L1 and L2 caches. y is the core instance number, for example, CORE0CLK is the clock for core 0.
CMEcCLK	The clocks for each SME2 unit in the cluster, up to 2 SME2 units in a C1-DSU cluster. These signals clock all SME2 logic. c is the SME2 instance number, for example, CME0CLK.
COMPLEXxCLK	The clocks for each complex in the cluster. Each clock is connected to all cores in the respective complex. x is the complex instance number, for example, COMPLEX0CLK is the clock for complex 0.
SCLK	This clock is used for the Snoop Control Unit (SCU), L3 memory system, and all the external interfaces, including AXI, CHI, and Accelerator Coherency Port (ACP). It is also used for cores and complexes that are configured to run synchronously with the C1-DSU.
PCLK (DebugBlock)	The clock for the DebugBlock **Note:** The DebugBlock and the C1-DSU 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 C1-DSU cluster, ensure that you place an asynchronous bridge between the two clock domains.
PCLK (C1-DSU cluster)	The clock for the debug interface in the C1-DSU cluster **Note:** The DebugBlock and the C1-DSU 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 C1-DSU cluster, ensure that you place an asynchronous bridge between the two clock domains.
ATCLK	The clock for the ATB trace bus output from the C1-DSU.

Signal	Description
GICCLK	The clock for the Generic Interrupt Controller (GIC) AXI-Stream interface between the C1-DSU and an external GIC.
PERIPHCLK	The clock for the peripheral logic inside the C1-DSU such as clock and power management logic and timers.
PPUCLK	The clock for the Power Policy Units (PPUs). The PPUs reside in their own clock domain, see 3.2 Clock domains on page 45.

For more information on core and complex clock signaling, see *Functional integration* in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

All clocks can be driven fully asynchronously to each other. The C1-DSU contains all the necessary synchronizing logic for crossing between clock domains. There are no clock dividers and no latches in the design. The entire design is rising-edge triggered.

Note

- You can configure the cores to run synchronously to the L3 memory system, on a per-core basis at the build time configuration stage. If this option is chosen, the corresponding COREyCLK signals and COMPLEXxCLK signals (if applicable) are not present and the synchronous cores are run with SCLK.
- You can configure the SME2 to run synchronously to the L3 memory system, on a per-SME2 unit basis at the build time configuration stage. If this option is chosen, the corresponding CMEcCLK signal is not present, and the corresponding SME2 unit is run with SCLK. For information on how to configure the cores to run synchronously with the L3 memory system, see *YAML configuration parameters* in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*
- 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 C1-DSU is designed with the following expectations to achieve an acceptable performance:

- The SME2 unit clocks are run at a frequency that gives sufficient performance while remaining energy efficient. This is typically a similar frequency to SCLK.
- The COREyCLK or COMPLEXxCLK can be dynamically scaled to match the performance requirements of that core.
- 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

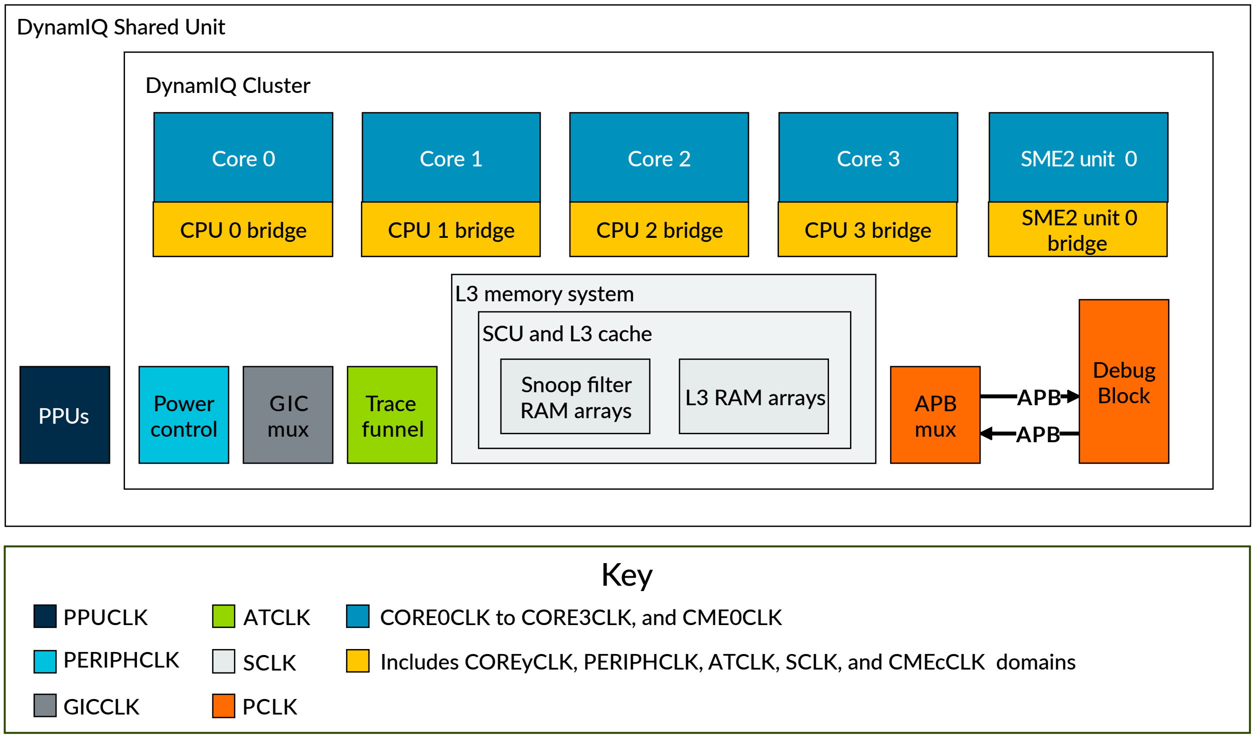
3.2 Clock domains on page 45
1.2 C1-DynamIQ Shared Unit configuration options on page 16

3.2 Clock domains

The C1-DSU has multiple clock domains. Each core or complex can be implemented in a separate clock domain.

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

Figure 3-1: C1-DSU 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. Similarly for the SME2 units, the SME2 unit bridge contains asynchronous bridges for all crossings between the unit and cluster clock domains. The unit bridge is split, with one half of each bridge being in the unit clock domain and the other half being in the cluster domain. At the cluster level, there is the Snoop Control Unit (SCU) bridge which contains crossings between the cluster clock domains as required.

Note

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

3.3 Resets

The C1-DSU has three external reset signals, a cluster-wide Cold reset, a cluster-wide MBIST reset, and a reset for the DebugBlock. Other resets, such as Warm resets to the cores are controlled, inside the cluster, by the Power Policy Units (PPUs).

The following table describes the C1-DSU reset signals.

Table 3-2: DSU reset signals

Signal	Description
nRESET	A C1-DSU cluster reset. nRESET is a single cluster-wide Cold reset. nRESET resets the PPU for the cluster, cores, and SME2 units, which in turn resets the C1-DSU cluster shared logic, cores, and SME2 units.
nMBISTRESET	A C1-DSU cluster reset. nMBISTRESET is a single cluster-wide reset signal that resets all necessary logic in the cores, SME2 and cluster for Memory Built-In Self Test (MBIST) testing.
nPRESET	The DebugBlock reset. A reset for all resettable registers in the DebugBlock.

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

All reset inputs can be asserted (HIGH to LOW) and deasserted (LOW to HIGH) asynchronously. Reset synchronization logic inside the C1-DSU 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. Similarly, the SME2 unit 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.

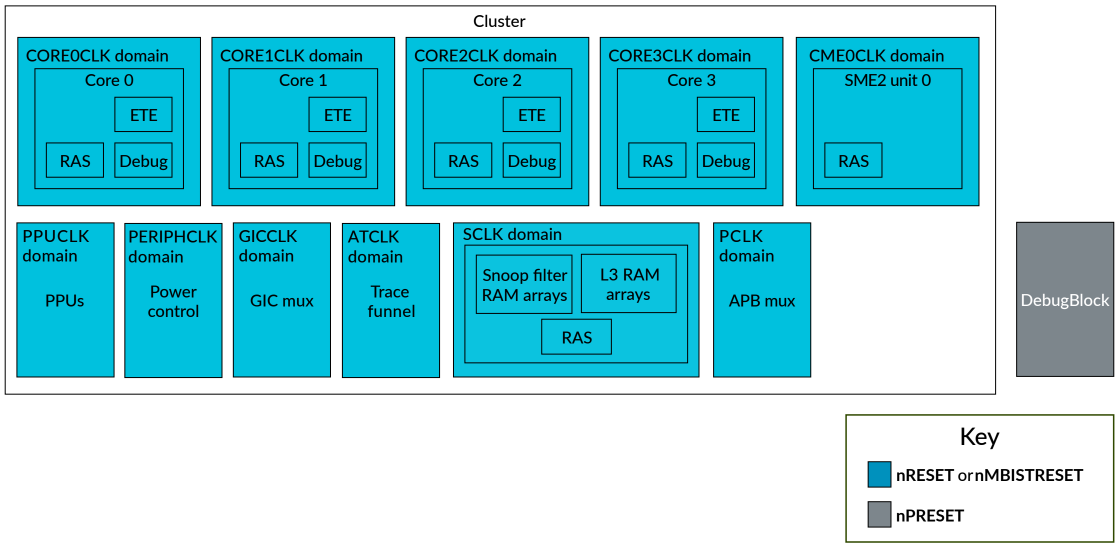
Resetting individual cores and SME2 units, and other cluster logic can be performed by programming the appropriate integrated PPU, see 5. Power and reset control with Power Policy Units on page 88. When the cluster internal resets are asserted, the PPUs drive the reset output signals that correspond to the power domain logic that is being reset.

Note

You can use the C1-DSU 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: C1-DSU pin-controlled reset domains



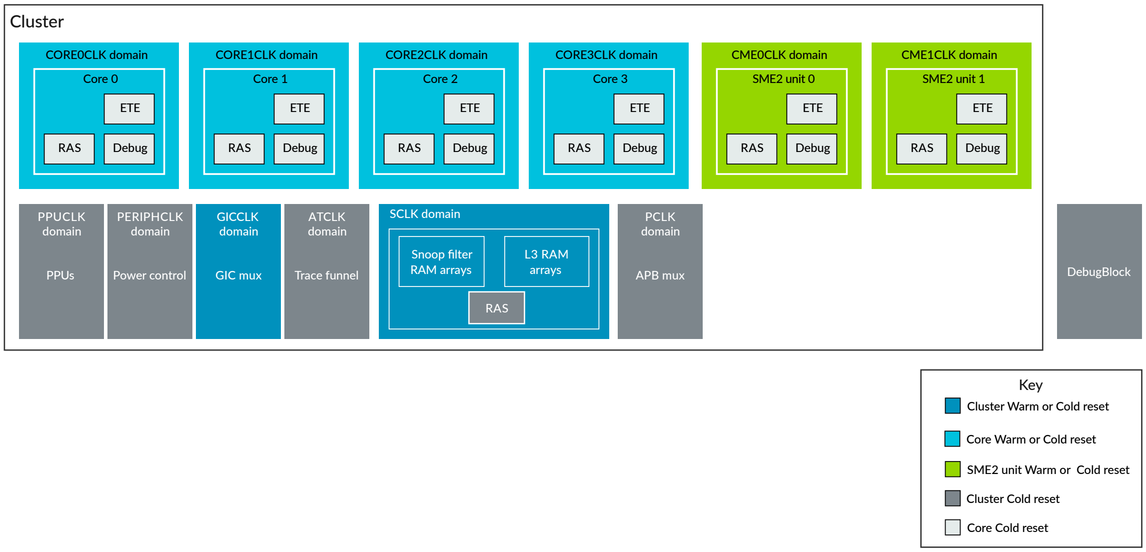
3.4 Resetting with Power Policy Units

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. Similarly, each of the PPUs for the SME2 units control their respective power features. 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 88.

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

Figure 3-3: C1-DSU 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, SME2, and core logic. For more details on this process, see 5.2.2 nRESET sequence on page 92.

4. Power management

This chapter describes the power domains, power modes, and operating modes for the C1-DSU and for the cores, complexes and SME2 units. 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 C1-DSU.

Note

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 88.

4.1 Power management in the C1-DSU

The C1-DSU 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 C1-DSU and cores in the cluster include:

- Internal core clock gating where different internal parts of the core are clock idle
- Per-core Dynamic Voltage and Frequency Scaling (DVFS)
- Powerdown of components of the cluster which can include:
  - Cores
  - SME2 units
  - All the L3 cache or parts of the L3 cache. See 4.4.1 L3 cache RAM powerdown on page 58 and 4.4.2 L3 cache slice powerdown on page 62.
- 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

Note

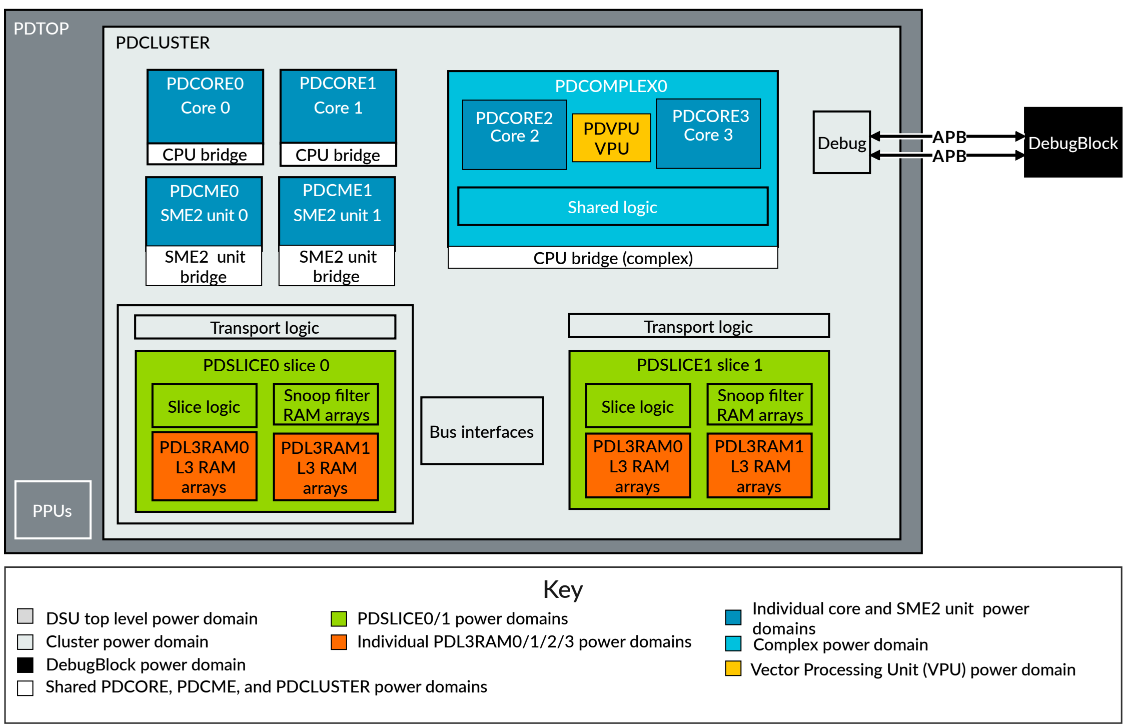
- The C1-DSU power domain architecture, power modes, and operational modes, are based on the Arm Power Control System Architecture, see *Arm® Power Control System Architecture*.
- This chapter does not describe how to use the PPUs for the cluster, cores, or SME2 units. See instead 5. Power and reset control with Power Policy Units on page 88.

4.2 C1-DSU supported power domains

The C1-DSU 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, complexes, and SME2 units 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 C1-DSU-based cluster.

Figure 4-1: C1-DSU power domains



Note

The logic for each CPU bridge is split across the core, SME2 and cluster power domains.

The cluster comprises the following power domains:

PDTOP

The top-level power domain (PDTOP) is typically placed in the same power domain as the other system components, for example, external bus infrastructure. The only cluster logic in this domain is the Power Policy Units (PPUs). This domain must be relatively always on compared to the other power domains. This is because the PPUs need to be able to power down the other domains including PDCLUSTER while remaining active. Therefore, the PDTOP power domain must be powered up before any of the other power domains are powered up, and it must only be powered down after the other power domains have been powered down.

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

PDCLUSTER

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

PDCME <CMEN>

Optionally, you can include up to two SME2 units in the cluster. Each SME2 unit resides in its own separate power domain.

For each instantiated SME2 unit, the SME2 unit bridge has logic both in the PDCME power domain and in the PDCLUSTER power domain.

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: C1-DSU power domains on page 51. If the Vector Processing Unit (VPU) is included, this resides in its own separate PDVPU power domain within PDCOMPLEX. Both PDCORE and PDVPU are gated power domains that can support retention.

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

For more information on the power domains of a complex, see 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, in a C1-DSU configured with four L3 cache slices, cache slices 2-3 are always powered down together.

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 C1-DSU cluster and each of the cores, complexes, and SME2 units 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 C1-DSU cluster.

Table 4-1: C1-DSU 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 C1-DSU 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 C1-DSU cluster including the L3 logic, the cores, complexes, and SME2 units 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.

Power mode	Short name	Description
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 C1-DSU cluster shared logic apart from the PPUs.
Debug recovery mode	DBG_RECOV	Debug recovery mode is used for applying a Warm reset to the cluster, while preserving memory and Reliability, Availability, and Serviceability (RAS) registers for debug purposes. Both L3 cache and RAS state are preserved when transitioning from DBG_RECOV mode to ON mode. Debug recovery mode is typically used in debugging a watchdog timeout.

4.3.1 On mode (ON)

In the On mode, the C1-DSU 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 C1-DSU cluster shared logic is powered down, including the snoop filters and L3 cache RAMs. The PDCLUSTER domain is inoperable and all state is lost.

In the Off mode, power is removed from PDCLUSTER power domain. However, the PDTOP power domain is still powered up including all the Power Policy Units (PPUs).

The C1-DSU 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 C1-DSU 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.2.6 IMP_CLUSTERPWRCTLR_EL1, Cluster Power Control Register on page 318.

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 88.

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 C1-DSU 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 88.

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 FULL_RET 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 C1-DSU 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 C1-DSU remains coherent when in Memory retention mode. Any snoop arriving is stalled while the C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU cluster shared logic remains powered. This means the L3 cache RAMs are in retention but the snoop filter RAMs and the rest of the C1-DSU 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 C1-DSU cluster shared logic in the cluster.

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.
- 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, SME2 unit, 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.

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 C1-DSU cluster shared logic including the L3 cache RAMs is powered up.

The C1-DSU 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.

Important information when using the Debug recovery mode

- Debug recovery mode is strictly for debug purposes. It must not be used for functional purposes, because correct operation of the cluster is not guaranteed when entering this mode.
- Debug recovery mode can occur at any time with no guarantee of the state of the cluster. A request of this type is accepted immediately, therefore its effects on the core, cluster, SME2 unit, 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 the 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.

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 C1-DSU 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 power-down 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 workload 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 59
- 4.4.1.2 Setting CLUSTERPWRCTLR_EL1.PRTNRQ power portion control on page 60
- 4.4.1.3 L3 RAM power control using PPU static transactions on page 60

Note

For information on operating modes, see 4.5 Cluster operating modes on page 66.

4.4.1.1 Setting automatic L3 cache power portion control

The C1-DSU 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

Note

This register is only for use in conjunction with automated slice powerdown. See 4.4.2.2 Automated slice powerdown on page 64.

See 4.4.1.4 Calculating values for threshold registers on page 61 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 C1-DSU 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 the IMP_CLUSTERL3HIT_EL1 and IMP_CLUSTERL3MISS_EL1 registers are compared against the values programmed in the threshold registers:

- IMP_CLUSTERL3DNTH0_EL1, CLUSTERL3DNTH0
- IMP_CLUSTERL3DNTH1_EL1, CLUSTERL3DNTH1
- IMP_CLUSTERL3UPTH0_EL1, CLUSTERL3UPTH0
- IMP_CLUSTERL3UPTH1_EL1, CLUSTERL3UPTH1
- IMP_CLUSTERL3UPTH1_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 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 C1-DSU 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

The following sequence 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.2.6 IMP_CLUSTERPWRCTLR_EL1, Cluster Power Control Register on page 318 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 59 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 caches of the powered up cores. 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 additional 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 down. 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 up require a snoop filter size greater than provided in the current slice power mode then 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 power mode for the automated slice powerdown calculation, even if the core is in OFF mode.
- 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 up is greater than the HSLCCNT or OSLCCNT fields, after masking with the HSLCMASK or OSLCMASK fields, then 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, and you want to ensure that:
    - If any high performance core is powered up, the cluster is placed into ALL SLICE mode.
    - If no high performance cores are powered up, the cluster is placed into HALF SLICE mode.

- If only one or two high efficiency cores are powered up, the cluster is placed into ONE SLICE mode.

Then you can configure this example using the following 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.

Note

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 operating system 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 down 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 down 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

The C1-DSU has a quick nap mode, which is enabled by default if the RAMs, that are implemented, support quick nap. There is no need for additional software control.

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.

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 C1-DSU, 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 C1-DSU 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 Direct connect configurations, there are no operating modes.
- 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 up and active in four different slice configurations.

Table 4-4: C1-DSU 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.

Note

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.).

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: C1-DSU 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

Note

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 122.

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

Table 4-6: C1-DSU 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

Cluster power mode	ON, OFF_EMU			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 C1-DSU 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.

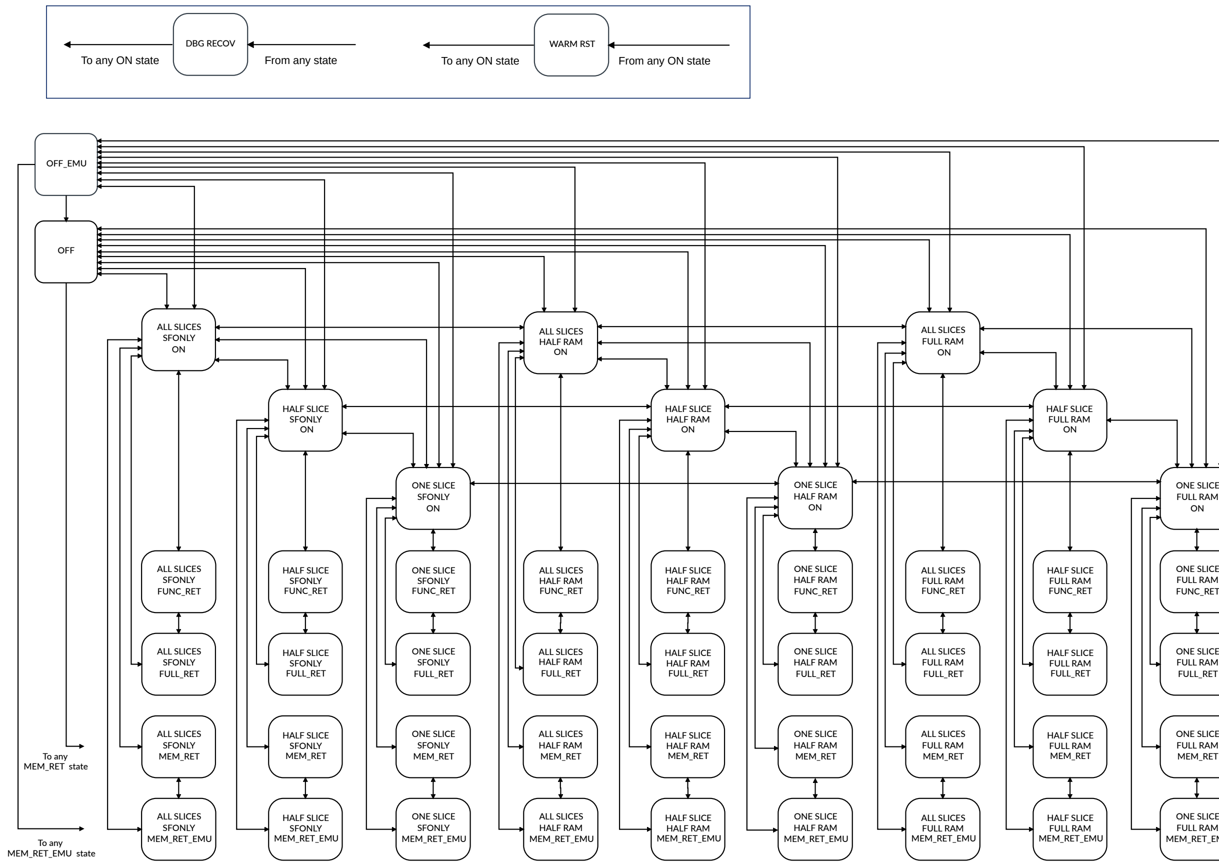
The cluster PPU controls transitions between the cluster PPU modes. Therefore, a System Control Processor (SCP) can program the PPU to go to any allowed PPU mode, and the PPU automatically makes the necessary transitions to reach the requested PPU mode.

The following figure shows the supported PPU mode transitions for the C1-DSU cluster.

Note

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

Figure 4-2: C1-DSU cluster PPU mode transitions

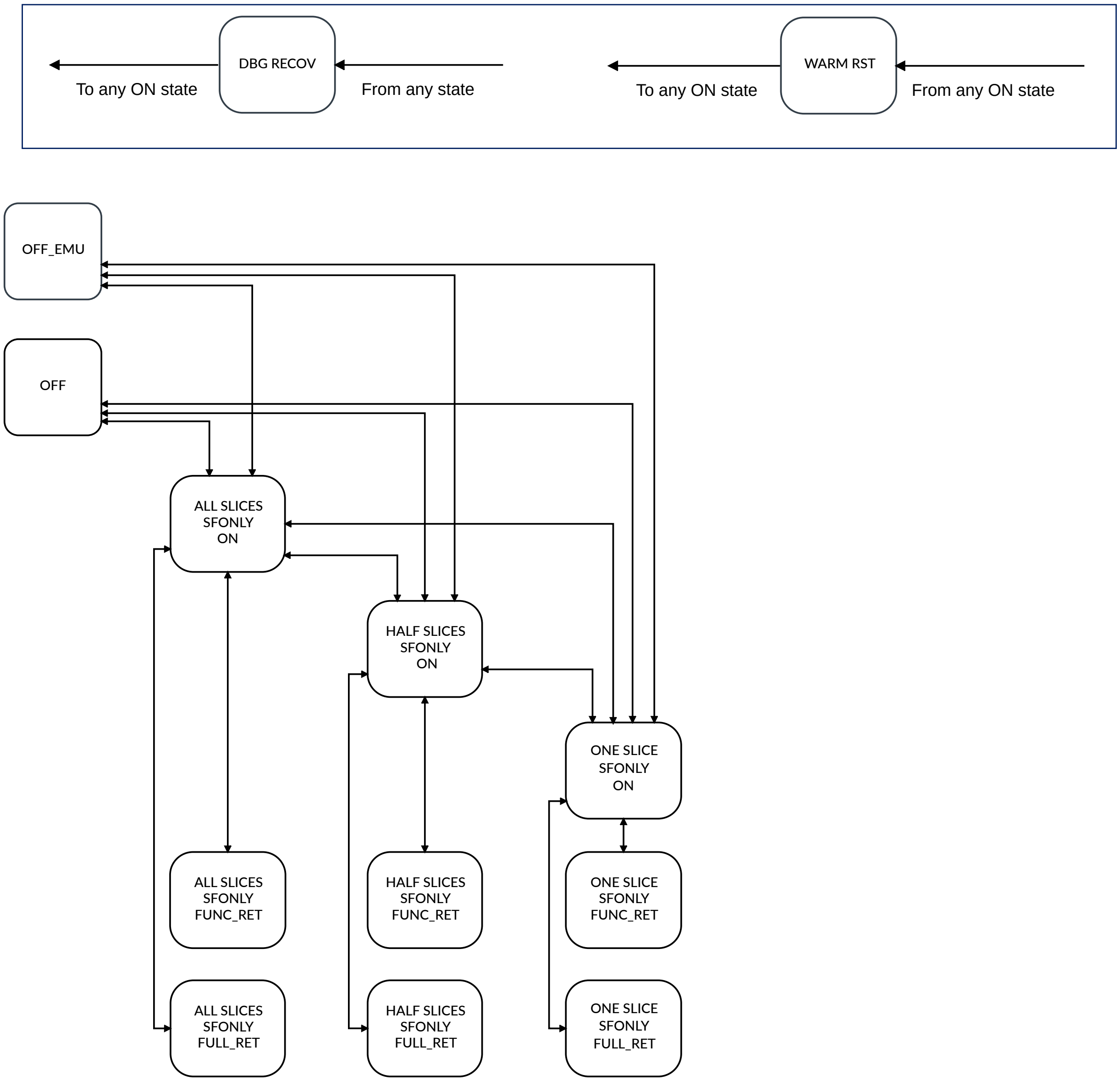


The following figure shows the supported PPU mode transitions for the C1-DSU cluster where L3 cache is not implemented.

Note

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

Figure 4-3: C1-DSU cluster PPU mode transitions, no L3 cache

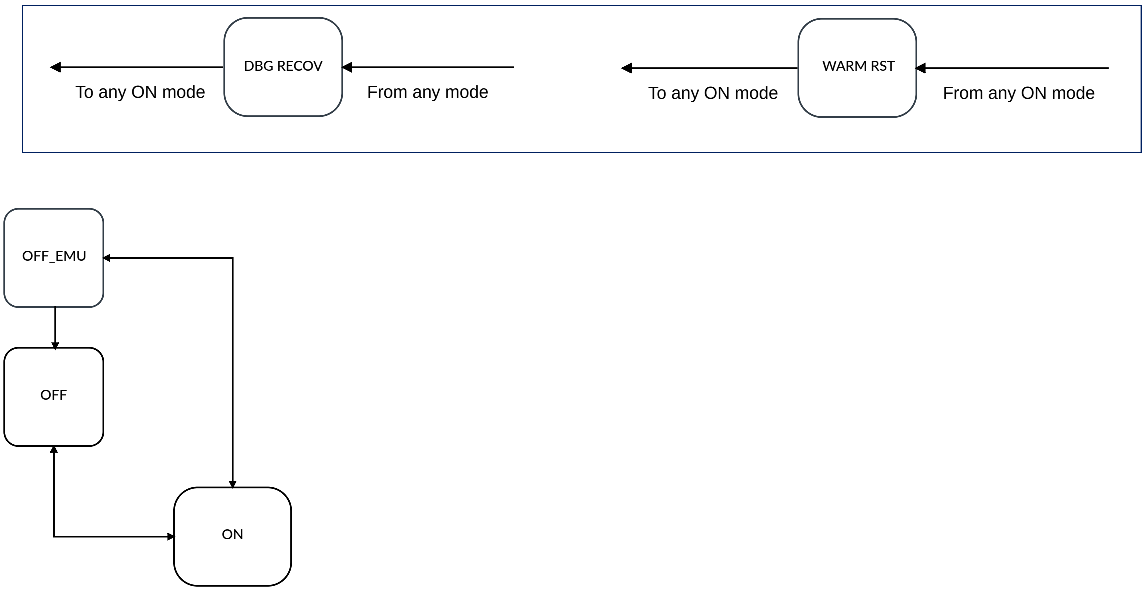


The following figure shows the supported PPU mode transitions for C1-DSU cluster when Direct connect is implemented.

Note

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

Figure 4-4: C1-DSU cluster PPU mode transitions, Direct connect



ALL SLICE FULL RAM ON

In this PPU mode, all the C1-DSU 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 C1-DSU cluster shared logic, including snoop filter RAMs, is powered up but half or all the L3 cache RAMs remain powered down. If the C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU 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 5.2 Power policy unit operation on page 90.

Table 4-7: Cluster domain PPU mode transition behavior

Start PPU mode	End PPU mode	C1-DSU 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.

Start PPU mode	End PPU mode	C1-DSU behavior
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® C1-DynamIQ™ Shared Unit 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 C1-DSU 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 C1-DSU 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 C1-DSU 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 Vector Processing Unit (VPU), if present, is OFF.
Full retention	FULL_RET	The core is in retention state. In this mode, only power that is required to retain register and RAM state is available. The core is non-operational. If the core supports functional retention and functional retention is enabled, then the core must be in Functional retention mode before it enters this mode.
Off	OFF	The core is powered down, either by using internal power switches or externally by the voltage regulator.
Emulated off	OFF_EMU	On mode, with Warm reset asserted. Debug state is retained and accessible.
Debug recovery	DBG_RECOV	The RAM and logic are powered up. This mode is for applying a Warm reset to the cluster, while preserving memory and RAS registers for debug purposes. Both cache and Reliability, Availability, and Serviceability (RAS) state are preserved when transitioning from DBG_RECOV to ON. **Caution:** This mode must not be used during normal system operation.
Warm reset	WARM_RST	This Warm reset mode is used to reset the core. For more information about what is reset, see your core Technical Reference Manual (TRM).

4.8.1 Core PPU mode transitions

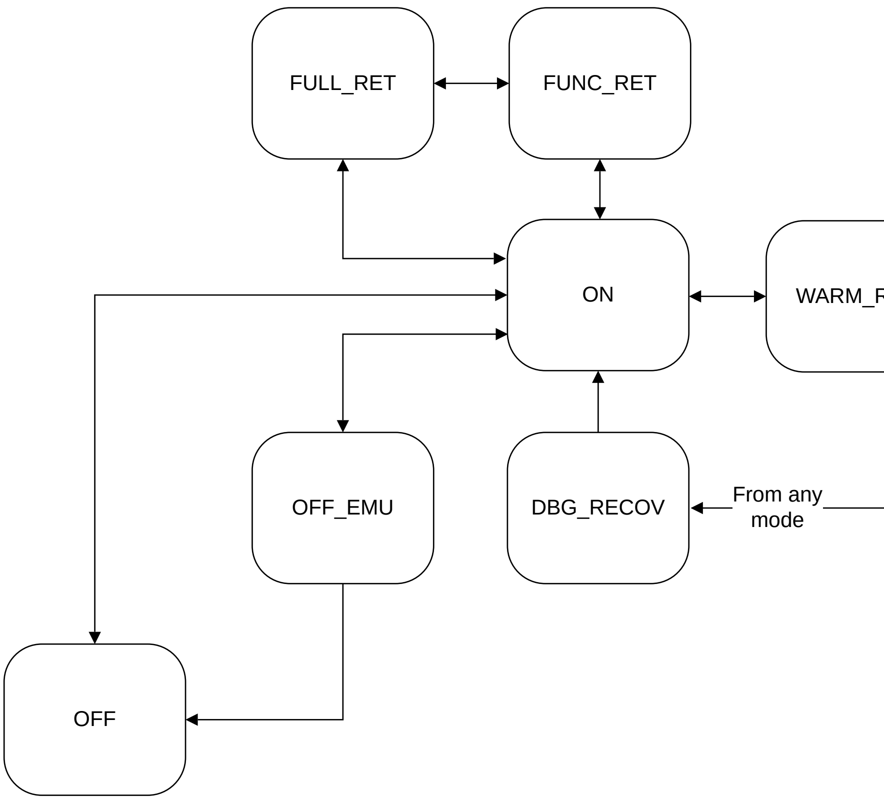
Each core supports a set of Power Policy Unit (PPU) mode transitions. These transitions are controlled by their respective core PPU. Therefore, a System Control Processor (SCP) can program a core PPU to go to any allowed PPU mode, and the PPU automatically makes the necessary transitions to reach the 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 4-5: Permitted core PPU mode transitions



On mode (ON)

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

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

Off mode (OFF)

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

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

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

Functional retention (FUNC_RET) mode

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

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

Full retention mode (FULL_RET)

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

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

Debug recovery mode (DBG_RECOV)

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

Note

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

Emulated off mode (OFF_EMU)

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

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

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

Requested PPU mode		PCSM channel		
Core0	Core1	Core0	Core1	Shared logic
Functional retention	On	ON	ON	ON
Functional retention	Functional retention	ON	ON	FUNC_RET
Functional retention	Full retention	ON	FULL_RET	FUNC_RET
Functional retention	Debug recovery	ON	ON	ON
Functional retention	Emulated off	ON	ON	ON
Functional retention	Off	ON	OFF	FUNC_RET
Full retention	On	FULL_RET	ON	ON
Full retention	Functional retention	FULL_RET	ON	FUNC_RET
Full retention	Full retention	FULL_RET	FULL_RET	FULL_RET
Full retention	Debug recovery	FULL_RET	ON	ON
Full retention	Emulated off	FULL_RET	ON	ON
Full retention	Off	FULL_RET	OFF	FULL_RET
Debug recovery	On	ON	ON	ON
Debug recovery	Functional retention	ON	ON	ON
Debug recovery	Full retention	ON	FULL_RET	ON
Debug recovery	Debug recovery	ON	ON	ON
Debug recovery	Emulated off	ON	ON	ON
Debug recovery	Off	ON	OFF	ON
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: • Makes a request from ON mode to OFF mode. • Makes a request from ON mode to OFF_EMU mode. • Requests a reset using the RMR.RR register bit field.	Core B must be in the ON power mode before core A can transition.	Core B automatically indicates that it must transition from FULL_RET or FUNC_RET mode to ON mode. The core PPU must request this transition for core B before core A transition can proceed.

The DEVPACTIVE* inputs to the PPUs indicate that the core B must transition to ON mode, and so if the PPUs are in dynamic mode then this is handled automatically. If the PPUs are in static mode then the component programming the PPUs must ensure that this transition can happen. However, Arm recommends that FUNC_RET and FULL_RET modes are not used when the PPUs are in static mode, see 5.11 Core Full retention mode and static mode restrictions on page 118.

4.10 SME2 unit power modes

The SME2 unit power domain has a defined set of power modes and corresponding legal transitions between these modes. The power mode of each SME2 unit can be independent of other SME2 units in a cluster.

The Power Policy Unit (PPU) of an SME2 unit manages at the cluster level the transitions between the power modes for that unit.

The following table shows the supported SME2 unit power modes.

Caution

Power modes that are not shown in the following table are not supported and must not occur. Deviating from the legal power modes can lead to **UNPREDICTABLE** results.

Table 4-11: SME2 unit power modes

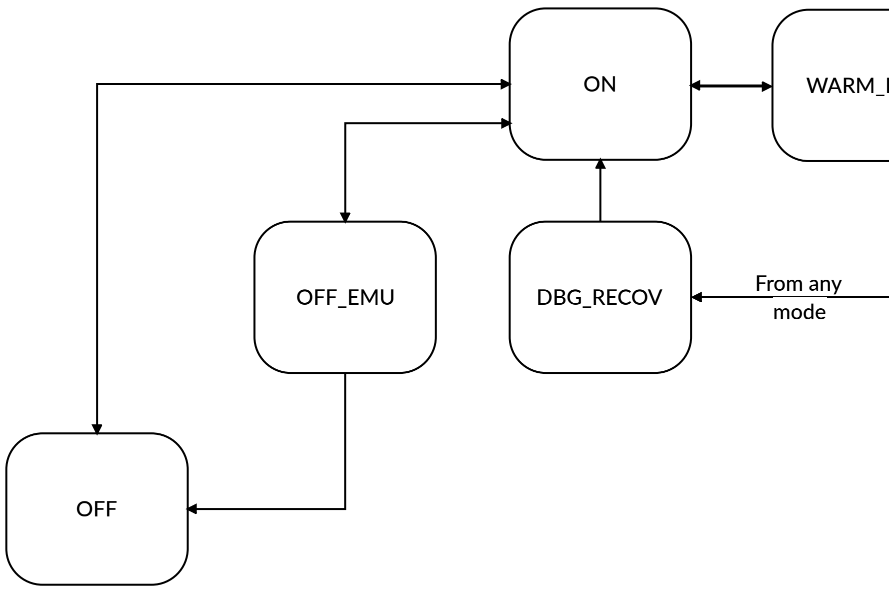
Power mode	Short name	Power state
On	ON	The SME2 unit is powered up and active. **Note:** For general use, Arm® recommends using dynamic policy for the SME2 unit PPU. This gives the most automation and quickest response times to requested power mode changes.
Off	OFF	The SME2 unit is powered down. **Caution:** The core stalls and does not execute any further instructions if all of the following are true: • The SME2 unit PPU is programmed in static mode, or dynamic mode with the PPU_PWPR.LOCK_EN bit set. • The SME2 unit is in Off power mode • A core is in On power mode and tries to execute Streaming SVE (SSVE) instructions • The system does not react to the requested power state and does not program the PPU to allow the SME2 unit to power on.
Emulated Off	OFF_EMU	Emulated off mode permits you to debug the powerup and powerdown cycle without changing the software. In this mode, the SME2 unit proceeds through all the powerdown steps, except: • The clock is not gated and power is not removed when the SME2 unit is powered down. • Only a Warm reset is asserted.
Debug recovery	DBG_RECOV	The RAM and logic are powered up. This mode is for applying a Warm reset to the C1-DSU cluster, while preserving memory and Reliability, Availability, and Serviceability (RAS) registers for debug purposes. Both cache and RAS state are preserved when transitioning from DBG_RECOV to ON. **Caution:** This mode must not be used during normal system operation.
Warm reset	WARM_RST	A Warm reset resets all logic except for the RAS registers. **Caution:** This mode must not be used during normal system operation.

The following figure shows the supported modes for the SME2 unit power domain and the legal transitions between them.

Note

The automatic transition between On and Off power modes is possible only if the SME2 unit PPU is in dynamic mode. This means that when the SME2 unit PPU is in dynamic mode, no software support is required for the transition between On and Off power modes.

Figure 4-6: SME2 unit power mode transitions



For further information, see the *Architectural clock gating modes* in the *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*

4.10.1 On mode

In the On power mode, the SME2 unit is on and fully operational.

Arm® strongly recommends that the SME2 unit PPU is only programmed in dynamic mode, and this must be done before a core executes the Streaming SVE (SSVE) instructions.

When the SME2 unit PPU is programmed in dynamic mode, the SME2 unit turns on automatically as soon as one of the following occurs:

- A core sends instructions to the SME2 unit.
- A debug access to the SME2 unit through the APB interface.

When a transition to the On mode is completed, the L1 data cache is accessible and coherent. No software support is required to turn on the SME2 unit when the SME2 unit PPU is in dynamic mode.

4.10.2 Off mode

In the Off power mode, power is removed completely from the unit.

All the following conditions should be met for the SME2 unit to turn off automatically:

- The SME2 unit Power Policy Unit (PPU) is programmed in dynamic mode
- All cores connected to the SME2 unit have PSTATE.SM and PSTATE.ZA set to 0
- No interrupt is pending from the Reliability, Availability, and Serviceability (RAS) node

Note

If the RAS interrupts are enabled and if RAS errors occur which cause RAS interrupts while the unit is powering down, then the RAS interrupts prevent the unit from powering down to prevent loss of information.

Once these conditions are reached, then if the SME2 unit is idle for a sufficient period then it can start turning off.

Caution

If the SME2 unit PPU is programmed in static mode, or dynamic mode with the PPU_PWPR.LOCK_EN bit set, a core tries to execute Streaming SVE (SSVE) instructions when the unit is Off, the core stalls and does not execute further instructions until the system responds and programs the PPU to allow a transition to ON.

In Off mode, all SME2 unit logic and RAMs are off. The values of **IMPLEMENTATION DEFINED** System registers, RAS, Maximum Power Mitigation Mechanism (MPMM), and Activity Monitoring Unit (AMU) registers are preserved and their values are restored when the SME2 unit is powered up. The L1 cache is disabled, cleaned, and invalidated. The unit is also removed from coherency automatically on transition to Off mode.

An attempted access to the **IMPLEMENTATION DEFINED** System registers through an MSR or MRS instruction or a utility bus access make the SME2 unit turn on so the transfer can succeed.

4.10.3 Emulated off mode

In Emulated off mode, all functional interfaces behave as if the SME2 unit were in Off mode.

4.10.4 Debug recovery mode

Debug recovery mode supports debug of external watchdog-triggered reset events, such as watchdog timeout.

By default, the SME2 unit invalidates its cache when it transitions from Off to On mode. Using Debug recovery mode allows the L1 data cache content that was present before the reset to be observable after the reset. In this mode, the content of the L1 data cache is retained and is not altered on the transition back to the On mode.

In addition to preserving the cache content, Debug recovery supports preserving the Reliability, Availability, and Serviceability (RAS) state. A transition to Debug recovery mode is made from any state, which puts the unit into a Warm reset state. There is no external mechanism to apply a Warm reset mode other than programming the C1-DynamIQ™ Shared Unit (DSU) Power Policy Units (PPUs).

Caution

Debug recovery is strictly for debug purposes. It must not be used for functional purposes, because correct operation of the L1 data cache is not guaranteed when entering this mode.

Debug recovery mode can occur at any time with no guarantee of the state of the unit. A request of this type is accepted immediately, therefore its effects on the unit, the C1-DSU cluster, or the wider system are **UNPREDICTABLE**, and a wider system reset might be required. In particular, any outstanding memory system transactions at the time of the reset might complete after the reset. The unit is not expecting these transactions to complete after a reset, and might cause a system deadlock.

If the system sends a snoop to the C1-DSU cluster during Debug recovery mode, depending on the cluster state:

- The snoop might get a response and disturb the contents of the L1 data cache.
- The snoop might not get a response and cause a system deadlock.

4.10.5 Warm reset mode

A Warm reset resets all state except for the Reliability, Availability, and Serviceability (RAS) registers.

Caution

Warm reset mode is strictly for debug purposes. It must not be used for functional purposes, because correct operation of the L1 data cache is not guaranteed when entering this mode.

Warm reset mode can occur at any time with no guarantee of the state of the unit. A request of this type is accepted immediately, therefore its effects on the unit, the C1-DSU cluster, or the wider system are **UNPREDICTABLE**, and a wider system reset might be required. In particular, any outstanding memory system transactions at the time of the reset might complete after the reset. The unit is not expecting these transactions to complete after a reset, and might cause a system deadlock.

If the system sends a snoop to the C1-DSU cluster during Debug recovery mode, depending on the cluster state:

- The snoop might get a response and disturb the contents of the L1 data cache.

- The snoop might not get a response and cause a system deadlock.

4.11 Maximum Power Mitigation Mechanism

The C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU 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.

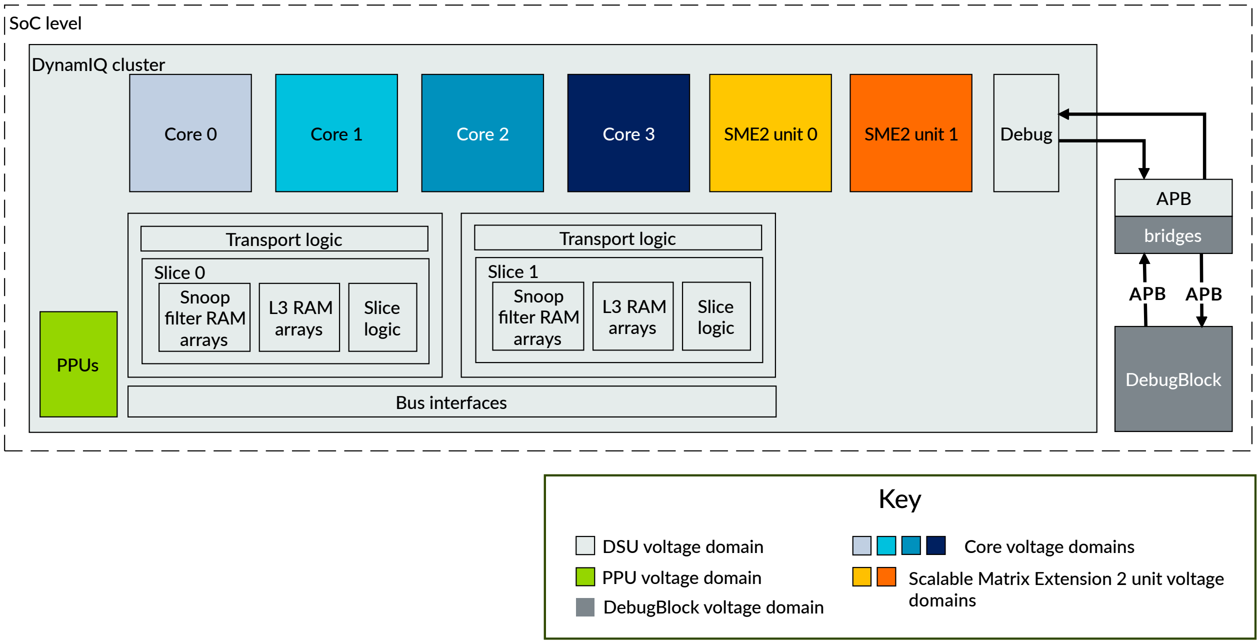
For more information on MPMM, see the *Maximum Power Mitigation Mechanism* section of the *System design* chapter in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

4.12 C1-DSU voltage domains

The C1-DSU supports each core and SME2 unit in the C1-DSU cluster being implemented in a separate voltage domain. There is also a separate voltage domain for the C1-DSU cluster itself.

The following figure shows the voltage domains in the cluster.

Figure 4-7: C1-DSU voltage domains



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

Note

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

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

Optionally, each instance of the SME2 unit can be placed in its own voltage domain like the cores. The boundary of the SME2 unit voltage domain is within the SME2 unit hierarchy itself. For an SME2 unit with an asynchronous bridge, part of the bridge is in the SME2 unit voltage domain and part is in the cluster voltage domain.

The C1-DSU 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 using using the Power Policy Units (PPUs) for the C1-DSU cluster, cores, and complexes, and the SME2 units.

5.1 The Power Policy Unit

Power mode control for the C1-DSU is provided by the Power Policy Units (PPUs) that are integrated into the cluster. These PPUs control all the PPU modes for all components in the cluster.

A PPU is a standard component for abstracting software-controlled power domain policy to low-level hardware control signaling. There is one PPU for controlling the C1-DSU cluster power domain (PDCLUSTER). Also, each core has a PPU for controlling the L1PD core power domain for the core (for the core pipeline and L1 caches), and a separate PPU for controlling the L2PD power domain (for the level 2 cache). When the cores are included in a complex each core L1PD domain is controlled by a separate PPU.

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

Note

- The PPUs for the cluster, the SME2 units, and all the core PPUs are provided as part of the C1-DSU.
- The implementation process automatically creates all the PPUs for the cluster, the cores, and the SME2 units, and connects these into the C1-DSU cluster. Each PPU has a set of memory-mapped control registers which is accessed using the utility bus.

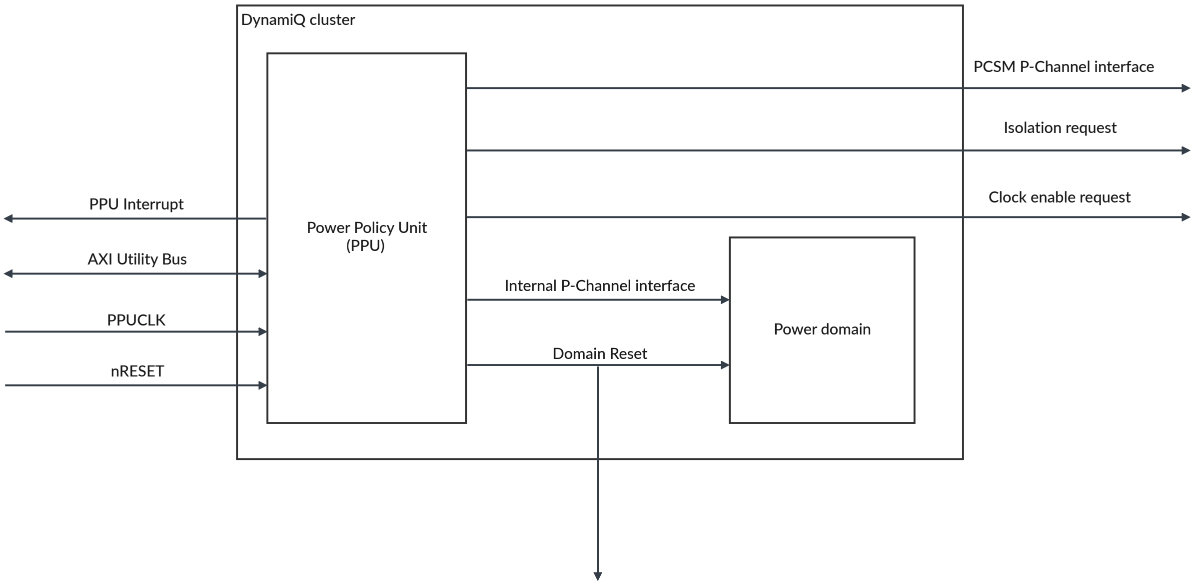
The PPUs:

- Abstract away the underlying mechanics of power state machine control of the C1-DSU. 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 C1-DSU PPU interfaces. All interfaces are external to the C1-DSU 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: C1-DSU 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 core and SME2 unit power domains.

Note

Some of the device control interface signals are exported outside of the C1-DSU 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:

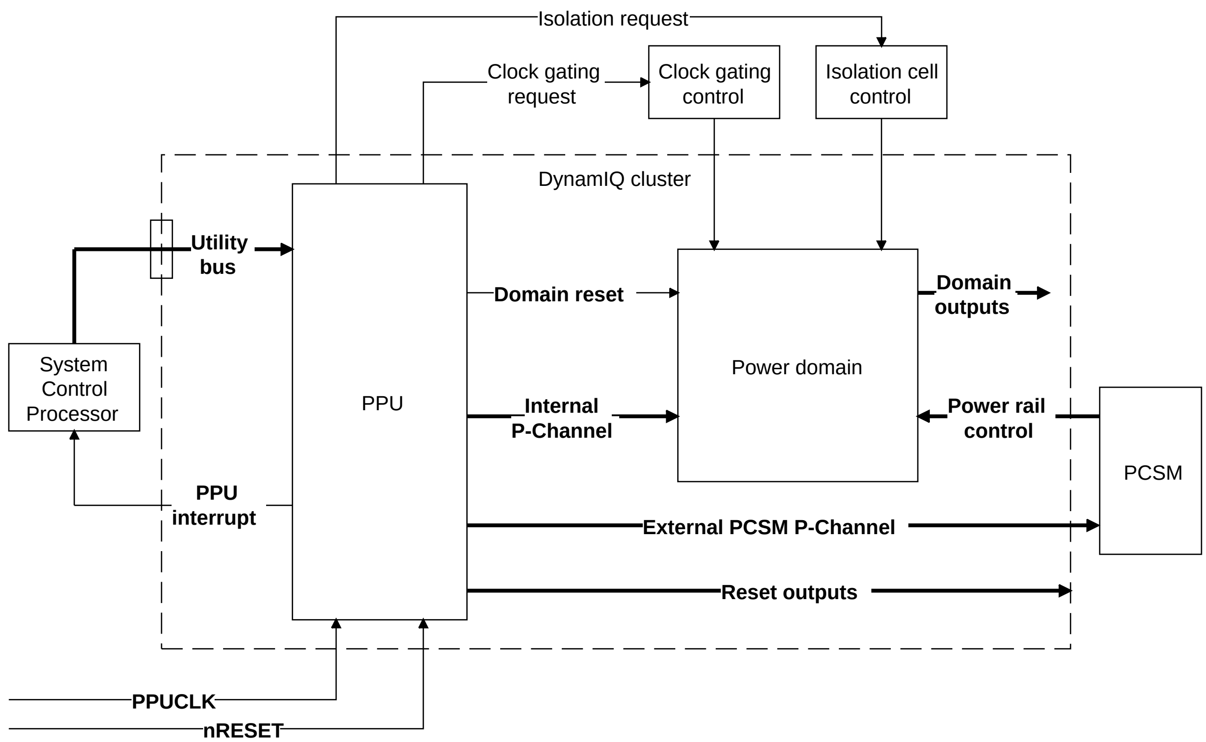
- The device interface, which consists of a P-Channel interface, see *AMBA® Low Power Interface Specification*.
- The device control interface, which includes clock enables, resets, and isolation control.

PCSM interface

The Power Control State Machine (PCSM) interface is an external interface for controlling low-level technology-specific power switch and retention controls. You must connect a PCSM to each interface as part of the C1-DSU implementation. There are separate PSCM interfaces for each core instantiated in the cluster, and a separate PCSM for the C1-DSU cluster itself. There are also separate PCSM interfaces for each SME2 unit instance.

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

Figure 5-2: C1-DSU PPU connections to a power-gated domain



All the PPUs contained within the C1-DSU are pre-built and are based on configurations of the CoreLink PCK-600 Power Policy Units and comply with the PPU architecture specification version 1.1, see *Arm® Power Policy Unit Architecture Specification*.

5.2 Power policy unit operation

The Power Policy Unit (PPU) supports all the C1-DSU 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, SME2 unit or cluster is only made once the core, SME2 unit or C1-DSU 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.

Note

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

Each PPU contains a state machine representation of its supported PPU mode transitions. For example, the cluster PPU has the PPU mode transitions for the cluster, see 4.7 Cluster PPU mode transitions on page 69. 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 SME2 units, the interrupt output signals are CMEPPUIRQ[<c>], where c is the SME2 unit instance number.

For the C1-DSU, a PPU is programmed by the System Control Processor (SCP) through the C1-DSU utility bus. The SCP programs the PPU mode or range of PPU modes that it wants the C1-DSU cluster to enter based on the current system requirements.

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

The PPU and the power domain communicate through the device P-Channel. This device P-Channel is internal to the C1-DSU. 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

5.2.1 Implicit resets from power modes

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

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

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

5.2.2 nRESET sequence

Asserting nRESET causes all the cluster, core and SME2 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, SME2, and cluster logic. Each PPU is responsible for resetting its associated core, SME2 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, cores and SME2 units are also reset.
2. nRESET is deasserted:
    - The PPUs are now active and can start logical operation.
    - The cluster, cores and SME2 units 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. Depending on how the core PPUs are configured at build time to either reset to static mode or dynamic mode, using the build time configuration parameter `PPU_RST_STATE`, determines how the reset power mode is programmed:
    - If `PPU_RST_STATE = FALSE`, the PPUs reset to static mode, then software programs the PPUs to enter the desired mode, typically ON mode.
    - If `PPU_RST_STATE = TRUE`, the PPUs reset to dynamic mode, then an automatic wakeup event is generated for all the cores in the cluster. The PPUs automatically request the cluster and core power domains to enter the ON power mode.
4. The system continues.

5.2.3 Initial cluster operating mode

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

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

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

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.

5.3 Utility bus accesses

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

Accesses to the 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, and each of the SME2 units are grouped on separate 64KB page boundaries allowing access control to be enforced by a Memory Management Unit (MMU).

You can only access PPUs with one of the following Security states:

- Root state only If the cluster is enabled for Realm Management Extension (RME). For more information about when RME is enabled, see 1.4.1 Realm management extension on page 26.
- Secure state if RME is not enabled.

Accesses to these registers with the Non-secure bit set or Realm bit set are treated as **RAZ/WI**.

5.4 Cluster PPU mode control

The Power Policy Units (PPUs), that are integrated into the cluster, control all the PPU modes for all components in the cluster. There is one PPU for the C1-DSU 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.

5.4.1 External cluster PPU registers

The Power Policy Unit (PPU) registers for the C1-DSU 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If Realm Management Extension (RME) is enabled, you must access the cluster system control registers from Root state. If RME is not enabled, you must access the cluster system control registers from the Secure state. For RME to be enabled, the cluster must be in Direct connect configuration and the

LEGACYTZEN input signal is LOW, see 1.4.1 Realm management extension on page 26.

- 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.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 5-1: Cluster registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0000	CLUSTERIDR	See individual bit resets.	64-bit	Cluster Main Revision Register	Yes
0x0008	CLUSTERREVIDR	See individual bit resets.	64-bit	Cluster ECO ID Register	Yes
0x0010	CLUSTERPWRCTLR	See individual bit resets.	64-bit	Cluster Power Control Register	No
0x0028	CLUSTERL3DNTH0	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register	No
0x0030	CLUSTERL3DNTH1	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register	No
0x0038	CLUSTERL3UPTH0	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register	No
0x0040	CLUSTERL3UPTH1	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register	No
0x0048	CLUSTERBUSQOS	See individual bit resets.	64-bit	Cluster Bus QoS Control Register	No
0x0050	CLUSTERCFR	See individual bit resets.	64-bit	Cluster Configuration Register	Yes
0x0058	CLUSTERACTLR	See individual bit resets.	64-bit	Cluster Auxiliary Control Register	Yes
0x0060	CLUSTERECTLR	See individual bit resets.	64-bit	Cluster Extended Control Register	No
0x0068	CLUSTERCFR2	See individual bit resets.	64-bit	Cluster Configuration Register 2	No
0x0080	CLUSTERPPMCR	See individual bit resets.	64-bit	Cluster PPM Control Register	No
0x0088	CLUSTERMPMMCR	See individual bit resets.	64-bit	Cluster MPMM Control Register	No
0x0090	CLUSTERL3UPTH2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register	No

5.4.2 Encodings for cluster power and operating modes

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

The following table shows the power mode encodings for the C1-DSU cluster.

Note

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 C1-DSU 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 C1-DSU cluster operating mode encodings for CLUSTERPCSMPSTATE[7:4] and CLUSTERPPUHWSTAT[31:16].

Note

In the following table:

- 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 C1-DSU might generate PCSM transition requests between equivalent encodings.

Table 5-4: Operating mode enumeration for the C1-DSU cluster

Operating mode name	PPU_PWPR.OP_POLICY	PCSMPSTATE[7:4]	PPUHWSTAT[36:16]
ONE SLICE, SFONLY	0x0	0x0	0x01
ONE SLICE, HALF RAM	0x1	0x1	0x02

Operating mode name	PPU_PWPR.OP_POLICY	PCSMPSTATE[7:4]	PPUHWSTAT[36:16]
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

Note

In Direct connect configurations, where there is no Snoop Control Unit (SCU), none of the operating modes are supported that are listed in tables Table 5-3: Operating mode encodings for PPU_PWPR.OP_POLICY bit field on page 96 and Table 5-4: Operating mode enumeration for the C1-DSU cluster on page 96. For this configuration, the operating mode must be programmed to `0x0`.

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

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

<table>
<tr><th>Operating mode name</th><th>PPU_PWPR.OP_POLICY</th><th>Default configuration (with L3 cache and SCU)</th><th>Direct connect (No L3 cache and no SCU)</th><th>No L3 cache present (SCU present)</th></tr>
<tr><td>ONE SLICE, SFONLY</td><td>0x0</td><td rowspan="9">Supported</td><td rowspan="9">Not supported</td><td>Supported</td></tr>
<tr><td>ONE SLICE, HALF RAM</td><td>0x1</td><td>Not supported</td></tr>
<tr><td>ONE SLICE, FULL RAM</td><td>0x3</td><td>Not supported</td></tr>
<tr><td>HALF SLICES, SFONLY</td><td>0x8</td><td>Supported</td></tr>
<tr><td>HALF SLICES, HALF RAM</td><td>0x9</td><td>Not supported</td></tr>
<tr><td>HALF SLICES, FULL RAM</td><td>0xB</td><td>Not supported</td></tr>
<tr><td>ALL SLICES, SFONLY</td><td>0x4</td><td>Supported</td></tr>
<tr><td>ALL SLICES, HALF RAM</td><td>0x5</td><td>Not supported</td></tr>
<tr><td>ALL SLICES, FULL RAM</td><td>0x7</td><td>Not supported</td></tr>
</table>

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.

5.5 Core power mode control

There are separate Power Policy Units (PPUs) for each of the cores in the C1-DSU cluster.

A component such as a System Control Processor (SCP) can program each of the core PPUs using AXI transactions to the utility bus to set the appropriate power policy. The core PPU controls the low-level details of powering up, powering down, and resetting domains as necessary to implement the requested policy. The hardware performs any actions to reach the requested power mode, such as gating clocks, flushing caches, or disabling coherency. The power mode of each core can be changed independently of other cores in the cluster. There is no restriction on the order that cores are powered on or off, with respect to the other cores.

5.5.1 External core and SME2 unit PPU registers

The Power Policy Unit (PPU) for each core or SME2 unit in the C1-DSU 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 or SME2 unit in the C1-DSU. For more information about a register, click on the register name in the table.

Note

- This set of registers is a generic set that applies to both the core PPUs and the SME2 unit PPUs. Therefore, the values listed for the retention power states only apply when programming the core PPUs, because the SME2 units do not support retention states.
- If Realm Management Extension (RME) is enabled, you must access the cluster system control registers from Root state. If RME is not enabled, you must access the cluster system control registers from the Secure state. For RME to be enabled, the cluster must be in Direct connect configuration and the LEGACYTZEN input signal is LOW, see 1.4.1 Realm management extension on page 26.
- The core or SME2 unit 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 and SME2 unit PPU registers are based on a typical multi-core cluster configuration, but these values might vary for different cluster configurations.
- If the C1-DSU is configured for Direct connect, all these registers are present.
- The base address for the core and SME2 unit PPU registers is `0x<y>80000`, where y is the core and SME2 unit programming instance number. For example, for core 0 the PPU base address is `0x080000` and for core 1 the PPU base

address is `0x180000`. For the SME2 units , y starts counting from the number of cores in the cluster. For example, for a 4 core cluster with 2 SME2 units, the base address for the first unit would be `0x480000` and the base address for the second unit would be `0x580000`.

- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 5-6: Core Power Policy Unit registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	See individual bit resets.	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	See individual bit resets.	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	See individual bit resets.	32-bit	Power Status Register	Yes
0x010	PPU_DISR	See individual bit resets.	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	See individual bit resets.	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	See individual bit resets.	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	See individual bit resets.	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	See individual bit resets.	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	See individual bit resets.	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	See individual bit resets.	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	See individual bit resets.	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	See individual bit resets.	32-bit	Additional Interrupt Status Register	Yes
0x040	PPU_IESR	See individual bit resets.	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register	Yes
0x054	PPU_FULRR	See individual bit resets.	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	See individual bit resets.	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	See individual bit resets.	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	See individual bit resets.	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	See individual bit resets.	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0	Yes
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1	Yes
0xFE8	PPU_PIDR2	See individual bit resets.	32-bit	PPU Peripheral Identification Register 2	Yes
0xFEC	PPU_PIDR3	See individual bit resets.	32-bit	PPU Peripheral Identification Register 3	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFF0	PPU_CIDR0	See individual bit resets.	32-bit	PPU Component Identification Register 0	Yes
0xFF4	PPU_CIDR1	See individual bit resets.	32-bit	PPU Component Identification Register 1	Yes
0xFF8	PPU_CIDR2	See individual bit resets.	32-bit	PPU Component Identification Register 2	Yes
0xFFC	PPU_CIDR3	See individual bit resets.	32-bit	PPU Component Identification Register 3	Yes

5.5.2 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 C1-DSU 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>PCSMSTATE[15:4] value is `0x000`.

For information on the core and SME2 unit PPU registers, see B.1.4 External cluster PPU registers summary on page 530 and B.1.7 External core and SME2 unit PPU registers summary on page 654.

Related information

1.7 Core, complex, SME2 unit, and processing element numbering on page 28

5.6 SME2 unit power mode control

There are separate Power Policy Units (PPUs) for each of the SME2 units in the C1-DSU cluster.

Similar to the PPUs for the core, the System Control Processor (SCP) can program each of the SME2 unit PPUs using AXI transactions to the utility bus to set the appropriate power policy. The SME2 unit PPU controls the low-level details of powering up, powering down, and resetting domains as necessary to implement the requested policy. There is no restriction on the order that the SME2 units are powered on or off, with respect to other SME2 units.

From a programming perspective, the PPU registers for the SME2 units are identical to the core PPU registers group. Therefore, the core PPU registers become a generic set of registers that apply to the cores and SME2 units. However, unlike the cores, the SME2 units do not have retention power modes. When trying to program a PPU mode that is not supported, the PPU denies the request.

When considering the programming of the SME2 units through the utility bus, the SME2 unit is addressed as if it were an additional core to the cluster. Therefore, this is also referenced by <y> (which now becomes the core and SME2 unit programming instance number), where y is given by: * y = CN for SME2 unit instance 0 and * y = CN + 1 for SME2 unit instance 1.

Note

This is only applicable in the case of programming purposes only. For other use cases, the y takes on its usual definition and c is used as the SME2 unit instance number. For more details, see 1.7 Core, complex, SME2 unit, and processing element numbering on page 28.

5.6.1 Encodings for SME2 unit power modes

The SME2 unit Power Policy Unit (PPU) register bitfield PPU_PWPR.PWR_POLICY encodes the supported power modes for the SME2 units.

The following table shows the encodings for the SME2 unit power modes.

Note

In the following table:

- PCSMPSTATE[3:0] refers to CME<CMEN>PCSMPSTATE[3:0], where CMEN is the SME2 unit instance number
- PPUHWSTAT[15:0] refers to CME<CMEN>PPUHWSTAT[15:0], where CMEN is the SME2 unit instance number

---

Table 5-8: Power mode enumeration for the SME2 units in the C1-DSU cluster

Power mode	PPU_PWPR.PWR_POLICY	PCSMPSTATE[3:0]	PPUHWSTAT[15:0]
OFF	0x0	0x0	0x0001
OFF_EMU	0x1	0x8	0x0002

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

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

For information on the core and SME2 unit PPU registers, see B.1.4 External cluster PPU registers summary on page 530 and B.1.7 External core and SME2 unit PPU registers summary on page 654.

Related information

1.7 Core, complex, SME2 unit, and processing element numbering on page 28

5.7 Programming sequences for the cluster, core, and SME2 unit

Example Power Policy Unit (PPU) programming sequences are provided for both the cluster, cores, and the SME2 unit. 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.

5.7.1 Programming sequence to bring the cluster, cores, and SME2 unit from Off to On mode

Use the following steps, to program the Power Policy Unit (PPU) for the C1-DSU cluster, each of the cores, and each SME2 unit 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 5.7.3 Programming sequence for an interrupt controller to control transitions between On and Off mode on page 104.

Note

- Steps 2 and 5 are only required if you need to know when the power transition has completed. Otherwise they can be omitted.
- This example programs the cluster power mode before the core and the SME2 unit power modes. It is possible to program the power modes for the core and SME2 unit 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. Similarly, the SME2 unit power mode does not transition to the On power mode until the cluster has

reached the On power mode. However, it is possible to transition the SME2 unit power mode to On before transitioning the core power mode.

- In this task, <y> is the core and SME2 unit programming instance number. See 1.7 Core, complex, SME2 unit, and processing element numbering on page 28 for a description of <y>.

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. For each SME2 unit instance <y>, write to the SME2 unit PPU_PWPR register, address `0x<y>80000`, value `0x00000100`. This sets the dynamic power mode policy, with a minimum power mode of Off. It is possible to program the PPU for the SME2 unit for static mode by setting this value to `0x00000008`, but this should only be used for test purposes.
4. For each SME2 unit instance <y>, write to the SME2 unit PPU_PWPR register, address `0x<y>80000`, value `0x00000100`. This sets the dynamic power mode policy, with a minimum power mode of Off. It is possible to program the PPU for the SME2 unit for static mode by setting this value to `0x00000008`, but this should only be used for test purposes.
5. 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.
6. 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.

5.7.2 Programming sequence to bring the cluster, core, and SME2 unit from On to Off mode

Use the following steps, to program the Power Policy Unit (PPU) for the C1-DSU cluster, each of the cores, and each SME2 unit to request a change of static PPU mode 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 and SME2 unit programming instance number. See 1.7 Core, complex, SME2 unit, and processing element numbering on page 28 for a description of <y>.
- Steps 3, 5 and 7 are only required if you must know when the power transition has completed. Otherwise they can be omitted.

Caution

There are wake-up events for the Wait For Interrupt (WFI) executed on the cores that can cause the powerdown process to be abandoned. See your core Technical Reference Manual to determine how the core software powerdown sequence must be written so that it operates correctly if the powerdown sequence is abandoned.

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. For each SME2 unit instance <y>, write to the SME2 unit PPU_PWPR register, address `0x<y>80000`, value `0x00000000`. This sets the static power mode policy to OFF."
5. For each SME2 unit instance <y>, poll the SME2 unit PPU_PWSR register, address `0x<y>80008`, until the value read matches the value written to the PPU_PWPR register.
6. Write to the cluster PPU_PWPR register, address `0x030000`, value `0x00000000`.
   This sets the static power mode policy to OFF.
7. Poll the cluster PPU_PWSR register, address `0x030008`, until the value read matches the value written to the PPU_PWPR register.

5.7.3 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 C1-DSU cluster and each of the cores and SME2 units to power up the cluster, cores, and SME2 units when the signal COREWAKEREQUEST[<y>] is asserted, and to power down automatically when software has finished running on the cores.

About this task

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

Note

- In this task, <y> is the core and SME2 unit instance number. See 5.6 SME2 unit power mode control on page 100 for details on y.

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 SME2 unit PPU_PWPR register for the SME2 unit at programming instance <y>, address `0x<y>80000`, value `0x00000100`. This sets the dynamic power mode policy, with a minimum power mode of OFF.
3. 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.
4. To power up core <y> or power down core <y>, see steps in the following table.
    - 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.
- When the SME2 unit is in dynamic power mode, the SME2 unit will power up and power down in response to requests for SME2 unit activity from the core software.

5.8 Explicit reset of cluster, cores, SME2 units, and debug recovery mode

There are several reset scenarios for part, or all, of the C1-DSU cluster, cores and the SME2 units.

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

Note

- In two of the reset scenarios (5.8.4 Warm reset of the cluster, excluding the PPUs on page 108and 5.8.6 Reset of the cluster, excluding the PPUs, retaining the cache contents for debug on page 111) four different methods are given for performing the same reset. Arm strongly recommends the methods of either programming the CLUSTERCTL_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 CLUSTERCTL_CLUSTERRECOV register, or a signal reset using the input/output domain P-Channel signals, Arm strongly recommends that the PPUs 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 PPUs and associated logic prevents unsupported transactions from occurring.

The scenarios for resetting all or part of the C1-DSU cluster are:

- 5.8.1 Powerup (Cold) reset on page 106
- 5.8.2 Core software initiated Warm reset of an individual core on page 106
- 5.8.3 Core software initiated Cold or Warm reset of the cluster, excluding the PPUs on page 107
- 5.8.4 Warm reset of the cluster, excluding the PPUs on page 108
- 5.8.5 Cold reset of the whole cluster, including the PPUs, retaining cache contents for debug on page 110
- 5.8.6 Reset of the cluster, excluding the PPUs, retaining the cache contents for debug on page 111

5.8.1 Powerup (Cold) reset

This reset must be done the first time that the cluster is powered up. It resets parts of the C1-DSU including the Power Policy Units (PPUs).

Procedure

1. Assert the nRESET signal for a minimum of three PPUCLK cycles.
2. Deassert the nRESET signal.
3. Program the PPU for the cluster to On mode, see 5.7.1 Programming sequence to bring the cluster, cores, and SME2 unit from Off to On mode on page 102.
4. Program the PPU for each required core to On mode, see 5.7.1 Programming sequence to bring the cluster, cores, and SME2 unit from Off to On mode on page 102.
5. Program the PPU for each required SME2 unit to On mode. See 5.7.1 Programming sequence to bring the cluster, cores, and SME2 unit from Off to On mode on page 102.

5.8.2 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.

5.8.3 Core software initiated Cold or Warm reset of the cluster, excluding the PPUs

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

About this task

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

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.

5.8.4 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.

Note

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

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 CLUSTERCTL_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.

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 PPUs 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 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

---

1. Write to the cluster register CLUSTERCTL_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 DEVPACTIVE* 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

---

1. Write to the cluster register CLUSTERCTL_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 CLUSTERCTL_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 CLUSTERCTL_CLUSTERRECOV register, offset address `0x050050`, value `0x0000_0000_0000_0000`.

When the cluster and cores are placed in 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.

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 PPUs 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 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® C1-DynamIQ™ Shared Unit 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 PPUs are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPUs is required.

5.8.5 Cold reset of the whole cluster, including the PPUs, retaining cache contents for debug

To provide a Cold reset to the whole cluster, including the Power Policy Units (PPUs) 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 *pilatus.yaml configuration parameters* in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.
- Because this method resets the PPUs, 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.

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 three 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.

5.8.6 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 Pollicy Units (PPUs) and retaining the cache contents for debug, use one of the following four methods.

Note

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

  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.

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.

Note

- For the reset sequence to take effect automatically, ensure steps 1 and 2 are followed. This sets the cluster and core PPUs 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 PPUs 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 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

---

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 PPUs are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPUs 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:

Note

- For the reset sequence to take effect automatically, ensure steps 1 and 2 are followed. This sets the cluster and core PPUs 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 PPUs 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 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

---

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 PPUs are in dynamic mode, as recommended, then the cluster operating mode is also preserved and so no programming of the PPUs is required.

Method 4: Reset of the cluster using INPUTDOMAINPACTIVE and OUTPUTDOMAINPACTIVE 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:

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 DEVPACTIVE* 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

---

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 INPUTDOMAINPACTIVE[10] signal or the OUTPUTDOMAINPACTIVE[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.

> **Note**
> Optionally, keeping the INPUTDOMAINPACTIVE[10] or OUTPUTDOMAINPACTIVE[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 INPUTDOMAINPACTIVE[10] signal or OUTPUTDOMAINPACTIVE[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.

5.9 Power mode dependencies between the core or SME2, and the cluster

There are some dependencies between the Power Policy Unit (PPU) modes of core and SME2 unit, and the PPU modes of C1-DSU 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 SME2 unit PPU modes.

Table 5-10: PPU mode dependencies for SME2 unit

Current SME2 unit PPU mode	Requested SME2 unit PPU mode	Cluster dependency	Effect on SME2 unit
OFF or OFF_EMU	ON	The SME2 unit can only transition to ON once the cluster is in ON, cluster FULL_RET or FUNC_RET.	The SME2 unit 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 SME2 unit request can be accepted.	The SME2 unit 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 SME2 unit request can be accepted.	The SME2 unit request stalls until the cluster has transitioned from DBG_RECOV to ON.

The following table describes dependencies on the requested C1-DSU cluster PPU modes.

Table 5-11: 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 and SME2 units are OFF	Cluster PPU mode request is denied
ON	OFF/OFF_EMU/ MEM_RET/ MEM_RET_EMU	If the Accelerator Coherency Port (ACP) interface is present and SYSCOREQS is asserted	Cluster PPU mode request is denied
ON	MEM_RET_EMU or OFF_EMU	Not all cores and SME2 units are in OFF or OFF_EMU	Cluster PPU mode request is denied
ON	OFF	If a core or SME2 unit 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 and SME2 units not in OFF, OFF_EMU, WARM_RST, or DBG_RECOV	Cluster PPU mode request is denied
DBG_RECOV	ON	Cores and SME2 units 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 80.

5.10 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 and 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 powers 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, core, or SME2 unit 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, SME2 unit, or cluster are asserted.

   The RAS interrupt signals are n<type>ERRIRQ, n<type>FAULTIRQ, and n<type>CTITIRQ, where type can be CLUSTER, CORE, COMPLEX, or SME2 unit. For example:

   - nCLUSTERFAULTIRQ
   - nCOREFAULTIRQ[CN:0]
   - nCOMPLEXFAULTIRQ[CX:0]
   - nCMEFAULTIRQ[CM:0]
   - nCMEERRIRQ[CM:0]
2. If the Power Policy Unit (PPU) is currently transitioning to an OFF or MEM_RET power modes, then these requests to 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 1.7 Core, complex, SME2 unit, and processing element numbering on page 28

5.11 Core Full retention mode and static mode restrictions

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

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

Note

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

5.12 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 C1-DSU cluster share the L3 cache. The L3 cache is not supported in a Direct connect cluster configuration.

The shared L3 cache of the C1-DSU (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 by the Memory System Resource Partitioning and Monitoring (MPAM) architecture extension. A process is an instance of a computer program. 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 C1-DSU automatically performs cache cleaning, eliminating the need for software-controlled cache cleaning.

6.1 L3 cache allocation policy

The C1-DSU 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 C1-DSU 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.

Consider the following scenario:

1. An initial request from core 0 allocates data in the L1 or L2 caches but not in the L3 cache.
2. 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.
3. 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.
4. 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 120

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 129
1.2 C1-DynamIQ Shared Unit configuration options on page 16

6.3 Memory System Resource Partitioning and Monitoring control

The C1-DSU 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 C1-DSU 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* (IHI 0099).

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 C1-DSU 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. If this transaction requires a Secure state PARTID, then only the lower three bits of the PARTID are used.

> **Note**
>
> While a 6 bit PARTID field is stored in the L3 cache and output to the system, the L3 cache partitioning and the bandwidth partitioning might support all 6 bits or only 3 bits, depending on the configuration of the C1-DSU.

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 C1-DSU 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 C1-DSU 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 C1-DSU 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 C1-DSU, see 6.3 Memory System Resource Partitioning and Monitoring control on page 121.

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 C1-DSU 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 58) then the number of cache ways in each L3 cache partition portion are halved. 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 3 and 8 to 11) 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 58
6. L3 cache on page 120

6.5 Bandwidth partitioning

The C1-DSU 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 C1-DSU 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® Architecture Reference Manual for A-profile architecture*.

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.

Note

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 C1-DSU, the interconnect should stop returning link-layer credits on the CHI REQ channel. The C1-DSU 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 C1-DSU 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 423.

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.

Note

- If cores share a complex, then a stash request targeting the L2 cache is allocated into the shared L2 cache of this complex.
- In Direct connect, stashes are only supported if the core supports them and stashes always target the L2 cache of the core.

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 242.

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

Related information

6. L3 cache on page 120
16.2 PMU events on page 242

6.7 L3 cache data RAM latency

The C1-DSU L3 data RAM interface can be implemented with a configurable latency on the input and output paths.

The following options are available:

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

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

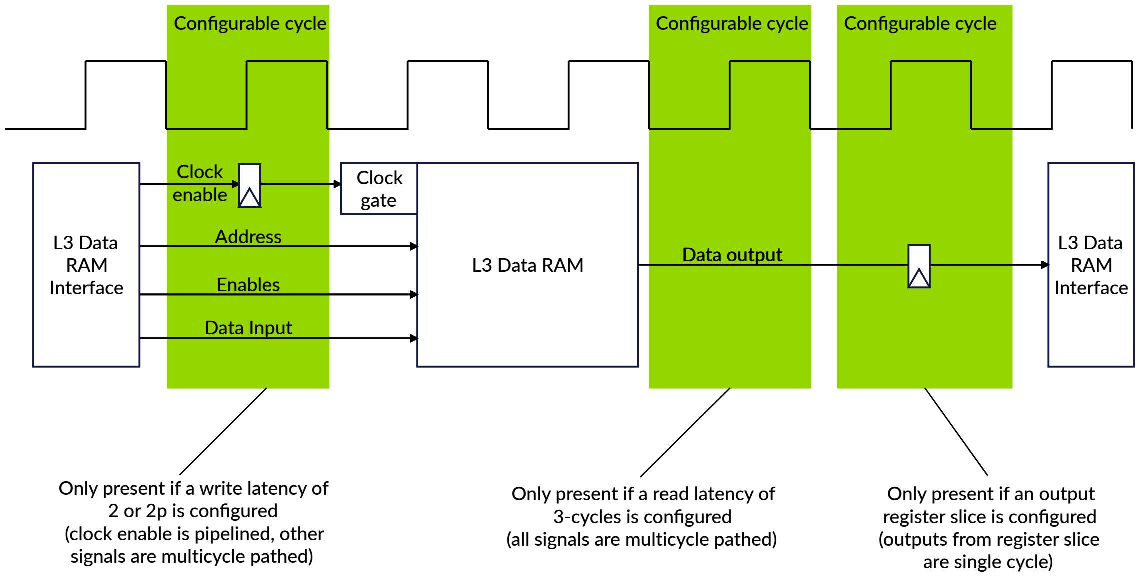
- An optional register slice on the output of the L3 data RAMs.

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

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

The following 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.

Note

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

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

Table 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

L3_DATA_WR_LATENCY	L3_DATA_RD_LATENCY	L3_DATA_RD_SLICE	L3 data RAM access cycles	L3 lookup bandwidth
2-cycles (including 2p)	3-cycles	Yes	5	Access every 3-clock cycles

Related information

1.2 C1-DynamIQ Shared Unit configuration options on page 16
6. L3 cache on page 120

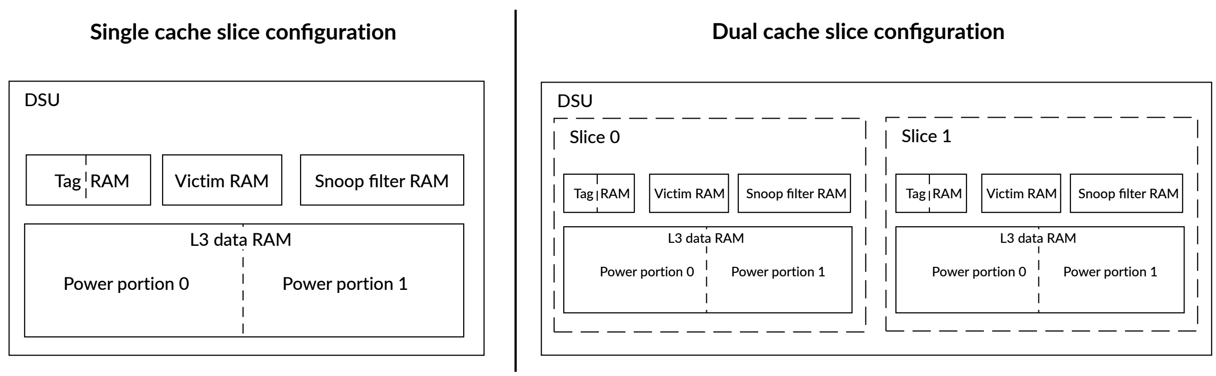
6.8 Cache slices and power portions

The L3 cache of the C1-DSU 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 portions.

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 121
6.8.1 Cache slice and requester port selection on page 129
4.4.2 L3 cache slice powerdown on page 62

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 C1-DSU to have either one, two, three, or four bus requester interface ports that use the AMBA 5 CHI Issue E protocol.

Note

If Realm Management Extension (RME) is not supported, all L3 memory system variants for the C1-DSU support the CHI Issue E protocol. If RME is supported, then the cluster is in Direct connect, and the bus interface uses the CHI Issue G protocol.

7.1 Multiple CHI bus requester port configurations

You can configure the C1-DSU 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 C1-DSU 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 C1-DSU 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual* on how to configure the RTL for the C1-DSU.

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 132.
   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	4
2	8
3	6
4	16

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 134.
   Once the mapping has been set up, the C1-DSU 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 135.

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 2-bit, 3-bit, or 4-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 sixteen groups.

Hashing for four, eight, or sixteen 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.
- REQUESTERINTERLEAVE3 is the configurable mask value set by REQUESTERINTERLEAVE3 input signal.
- ADDRESS is the PA of the transaction.
- PA_W is the physical address width.

Hashing for four address target groups

The hash is:

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

Hashing for eight address target groups

The hash is:

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

Hashing for sixteen address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE0[PA_W-1:6])
ADDRESS TARGET GROUP bit[1] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE1[PA_W-1:6])
ADDRESS TARGET GROUP bit[2] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE2[PA_W-1:6])
ADDRESS TARGET GROUP bit[3] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE3[PA_W-1:6])
```

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).

Note

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® C1-DynamIQ™ Shared Unit 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[NP-1:0], where NP 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 main requester port number.

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

Number of bus requester ports (NP)	REQUESTERDISABLE[NP-1:0]	Address target group mapping
1	-	All traffic to MP 0
2	`0b00`	Traffic for groups 0, 2, 4, 6 to MP 0 Traffic for groups 1, 3, 5, 7 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, 8, 12 to MP 0 Traffic for groups 1, 5, 9, 13 to MP 1 Traffic for groups 2, 6, 10, 14 to MP 2 Traffic for groups 3, 7, 11, 15 to MP 3
	`0b1100`	Traffic for groups 0, 2, 4, 6, 8, 10, 12, 14 to MP 0. Traffic for groups 1, 3, 5, 7, 9, 11, 13, 15 to MP 1.

Number of bus requester ports (NP)	REQUESTERDISABLE[NP-1:0]	Address target group mapping
	`0b0011`	Traffic for groups 0, 2, 4, 6, 8, 10, 12, 14 to MP 2. Traffic for groups 1, 3, 5, 7, 9, 11, 13, 15 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

7.2.3 CHI id bit setting

The allocation of address target groups numbers is also used to set the id, by the C1-DSU, 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 = 00	TgtID = 01	TgtID=10	TtgID=11
4	Group 0	Group 1	Group 2	Group 3
6	Groups 0, 1, 2	Groups 3, 4, 5	Not applicable	Not applicable
8	Groups 0, 1	Groups 2, 3	Groups 4, 5	Groups 6, 7
16	Groups 0, 1, 2, 3	Groups 4, 5, 6, 7	Groups 8, 9, 10, 11	Groups 12, 13, 14, 15

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 134 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.
Device non-reorderable transactions	These are sent to either: • The bus requester port, which is assigned to address target group 0. • All bus requester interfaces.
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: • Either sent to the bus requester port assigned to address target group 0; or • Not used on this interface and these transactions are managed by the peripheral port.

Note

- In the preceeding table, you can find the bus requester port that corresponds to the target group given from the lookup table, see Table 7-2: Mapping between address target groups and bus requester ports on page 134.
- By default, transactions described in the preceeding 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 132.

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:

`0b00`
All Device non-reorderable transactions are sent to address target group 0.

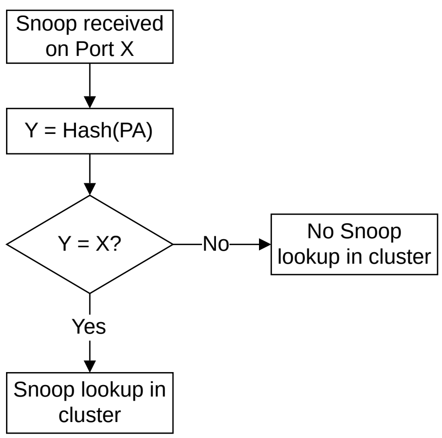
`0b01`
Device non-reorderable transactions are sent to any requester interface port that is based on the same address interleaving as for non-Device transactions.

`0b10`	Reserved
`0b11`	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 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

DVM messages do not have a Physical Address (PA) that is associated with them, and therefore cannot be routed to the appropriate requester interface port. To avoid processing

DVM messages multiple times, the C1-DSU only processes them when sent to the port used for outgoing DVM messages. Any DVM messages sent to the other bus requester ports are responded to but have no effect inside the cluster. Therefore, for best performance, Arm® recommends that your system is configured to only send DVM messages to the bus requester port that is used for outgoing DVM messages.

Outgoing DVM messages are always sent on the bus requester port that is allocated to address target group 0, unless the DEFAULTP signal is asserted. If DEFAULTP signal is asserted, all outgoing DVM messages are sent on the peripheral port.

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 C1-DSU supports, or requires the interconnect and system to support.

Table 7-5: CHI interconnect properties for the C1-DSU

CHI property	Supported by the C1-DSU	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 C1-DSU supports this feature if required by your interconnect.
Direct_Cache_Transfer	Yes	OPTIONAL. The C1-DSU supports this feature if required by your interconnect.
Data_Poison	For non-Direct connect configurations, yes if cache protection is enabled. Cache protection is enabled by setting the configuration parameter `SCU_CACHE_PROTECTION`. For Direct connect configurations, as there is no L3 cache, the Data_Poison property is enabled depending on if your core has cache protection enabled. See the *RTL configuration process* chapter of the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual* for more information.	Yes if cache protection is enabled.
Data_Check	No	No
CCF_Wrap_Order	No. The C1-DSU sends data packets in any order.	No
Barrier_Transactions	No	No. The C1-DSU 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 C1-DSU does not use these transaction types.

CHI property	Supported by the C1-DSU	Interconnect support required
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 C1-DSU.

Table 7-6: CHI requester interface values for the C1-DSU

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

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

Note

The C1-DSU 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 C1-DSU, 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

Note

- 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 C1-DSU 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. **Note:** The source of the transaction is encoded in the LPID field.
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	-

Note

- For the write issuing and read issuing capabilities, the total issuing capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. For multiple-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.

  The peripheral port can support up to 128 transactions, or the total number of outstanding transactions 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 C1-DSU 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 C1-DSU.

Table 7-11: CHI channel properties

Property	Value	Comment
Snoop acceptance capability	Configuration dependent	The total snoop acceptance capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. Each 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.
DVM acceptance capability	Four per requester port	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. The interconnect must be configured to never send more than four DVM messages to a CHI requester interface port, otherwise the system might deadlock.
Snoop latency	Hit and miss latencies depend on configuration	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. Snoops that hit in the L1 or L2 caches of a core have a higher latency. This latency depends on the type of core, and whether the hit is in the L1 or L2 cache. Typically the rate sustained is at least half that for the L3 cache bandwidth. Latencies can be higher if hazards occur or if there are not enough buffers to absorb requests.
	Miss	Dependent on build-time configuration
	DVM	Dependent on build-time configuration
Snoop filter	Supported	The cluster supports an external snoop filter in an interconnect. It indicates when clean lines are evicted from the cluster by sending Evict transactions on the CHI write channel. However there are some cases that can prevent an Evict transaction from being sent. Therefore you must ensure that you build any external snoop filter to handle a capacity overflow. When exceeding capacity, the snoop filter should send a back-invalidation to the cluster. Examples of case where evicts are not produced include: • Linefills that take External aborts. • Store exclusives that fail. • Mis-matched aliases.

Property	Value	Comment
Supported transactions	-	The C1-DSU supports all transaction types produced by the CHI protocol.

Non-Direct connect credit issuing

In non-Direct connect configurations, the C1-DSU issues 4 credits on the CHI receiver (RX) channels.

The latency for the C1-DSU to issue a credit after receiving a flit depends on the build-time configuration parameter `UNREGISTERED_CHI_REQUESTERS`:

- `TRUE`, same cycle
- `FALSE`, 2 cycles later

For the CHI transmitter (TX) channels, the latency for the C1-DSU to issue a flit after receiving a credit depends on the build-time configuration parameter `UNREGISTERED_CHI_REQUESTERS`:

- `TRUE`, 1 cycle later
- `FALSE`, 2 cycles later

Note

- The C1-DSU supports the CHI architectural maximum of 15 credits on its CHI transmit (TX) interfaces.
- For more information on the build-time configuration parameter `UNREGISTERED_CHI_REQUESTERS`, see *pilatus.yaml configuration parameters* in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

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 list shows CHI LPID[4:0] bitfield encodings:

Note

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

0x0- 0xD
: Core instance number

0xE
: Accelerator Coherency Port (ACP) 0

0xF
: Cache Write-Back

0x1A- 0x1B
: SME2 unit 0

0x1C- 0x1D
: SME2 unit 1

0x1E
: ACP 1

The following table shows the CHI read and write transaction types supported by the CHI-configured requester port on the C1-DSU.

Caution

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

Table 7-12: CHI read and write transactions supported by CHI-configured requester port, not applicable to Direct connect configurations

Transaction	Operation	Produced by C1-DSU
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 BROADCASTPERSIST input signal is HIGH.	Yes
CleanUnique	Not used	No
DVMOp	Branch predictor maintenance instructions, and Translation Lookaside Buffer (TLB) and instruction cache maintenance instructions when enabled by the BROADCASTTLBIINNER, BROADCASTTLBIOUTER, and BROADCASTICINVAL input signals	Yes
Evict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes

Transaction	Operation	Produced by C1-DSU
MakeInvalid	Not used	No
MakeReadUnique	Store instructions when the line is already cached in a Shared state inside the cluster. This includes store exclusive instructions, which set Excl HIGH.	Yes
MakeUnique	Store instructions of a full cache line of data that miss in the caches.	Yes
PCrdReturn	Not used	No
PrefetchTgt	Hardware prefetch hint to the memory controller	Yes
ReadClean	Reading Memory Tagging Extension (MTE) tags for a Cacheable shareable line that is already cached in the cluster without tags.	Yes
ReadNoSnp	Non-cacheable loads or instruction fetches, or cache linefills of Non-shareable cache lines into L1 or L2 caches.	Yes
ReadNoSnpSep	Not used	No
ReadNotSharedDirty	Cache data linefills started by a load instruction, or cache linefills started by an instruction fetch	Yes
ReadOnce	Cacheable shareable instruction fetches that are not allocating into a coherent cache	Yes
ReadOnceCleanInvalid	Not used	No
ReadOnceMakeInvalid	Not used	No
ReadPreferUnique	Speculative store to Cacheable shareable memory or, if Excl is HIGH, a load exclusive instruction.	Yes
ReadShared	Not used	No
ReadUnique	Cache data linefills started by a store instruction	Yes
ReqLCrdReturn	Link credit return	Yes
StashOnceSepShared	Cache prefetch when the L3 cache is not present or powered down. Configured by CLUSTERECTLR_EL1.	Not generated
StashOnceSepUnique	Cache prefetch when the L3 cache is not present or powered down. Configured by CLUSTERECTLR_EL1.	Not generated
StashOnceShared	Not used	No
StashOnceUnique	Not used	No
WriteBackFull	Evictions of dirty cacheable shareable lines from the cluster	Yes
WriteBackFullCMO	Cache maintenance instruction evicting a dirty shareable cache line	Yes
WriteBackPtl	Not used	No
WriteCleanFull	Evictions of dirty lines from the L3 cache, when the line is still present in an L1 or L2 cache.	Yes
WriteCleanFullCMO	Cache maintenance instruction cleaning a dirty shareable cache line	Yes
WriteEvictFull	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
WriteEvictOrEvict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1 register.	Yes
WriteNoSnpFull	Non-cacheable store instructions. Evictions of Non-shareable cache lines	Yes
WriteNoSnpFullCMO	Cache maintenance instruction evicting a dirty Non-shareable cache line	Yes
WriteNoSnpPtl	Non-cacheable store instructions	Yes
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

Transaction	Operation	Produced by C1-DSU
WriteUniqueFullStash	Not used	No
WriteUniquePtl	Generated as a result of Accelerator Coherency Port (ACP) WriteUniquePtl transactions when not allocating to the L3 cache	Yes
WriteUniquePtlCMO	Not used	No
WriteUniquePtlStash	Not used	No
WriteUniqueZero	Write of zeroes to a Shareable cache line using the `DC ZVA` instruction	Yes

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

Table 7-13: 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.
	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 C1-DSU never sends SnpRespDataPtl, NCBWrDataCompAck, 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 should use the recommended encodings in the table *Suggested DataSource value encodings* provided in the *AMBA® 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 C1-DSU 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 C1-DSU 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 C1-DSU to have either one, two, three, or four AXI manager interface ports.

8.1 Multiple AXI bus manager port configurations

You can configure the C1-DSU 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 C1-DSU 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 C1-DSU 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® C1-DynamIQ™ Shared Unit Configuration and Integration Manual* on how to configure the RTL for the C1-DSU.
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 149.

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	4
2	8
3	6
4	16

3. 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 151.
   Once the mapping has been set up, the C1-DSU 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 152.

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 2-bit, 3-bit, or 4-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 sixteen groups.

Hashing for four, eight, or sixteen 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.
- REQUESTERINTERLEAVE3 is the configurable mask value set by REQUESTERINTERLEAVE3 input signal.
- ADDRESS is the PA of the transaction.
- PA_W is the physical address width.

Hashing for four address target groups

The hash is:

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

Hashing for eight address target groups

The hash is:

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

Hashing for sixteen address target groups

The hash is:

```
ADDRESS TARGET GROUP bit[0] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE0[PA_W-1:6])
ADDRESS TARGET GROUP bit[1] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE1[PA_W-1:6])
ADDRESS TARGET GROUP bit[2] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE2[PA_W-1:6])
ADDRESS TARGET GROUP bit[3] = ^(ADDRESS[PA_W-1:6]
 & REQUESTERINTERLEAVE3[PA_W-1:6])
```

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).

Note

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® C1-DynamIQ™ Shared Unit 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[NP-1:0], where NP 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 main manager port number.

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

Number of bus requester ports (NP)	REQUESTERDISABLE[NP-1:0]	Address target group mapping
1	-	All traffic to MP 0
2	`0b00`	Traffic for groups 0, 2, 4, 6 to MP 0 Traffic for groups 1, 3, 5, 7 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, 8, 12 to MP 0 Traffic for groups 1, 5, 9, 13 to MP 1 Traffic for groups 2, 6, 10, 14 to MP 2 Traffic for groups 3, 7, 11, 15 to MP 3
	`0b1100`	Traffic for groups 0, 2, 4, 6, 8, 10, 12, 14 to MP 0. Traffic for groups 1, 3, 5, 7, 9, 11, 13, 15 to MP 1.

Number of bus requester ports (NP)	REQUESTERDISABLE[NP-1:0]	Address target group mapping
	`0b0011`	Traffic for groups 0, 2, 4, 6, 8, 10, 12, 14 to MP 2. Traffic for groups 1, 3, 5, 7, 9, 11, 13, 15 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 C1-DSU, in the transaction address.

The following table shows how the two least significant bits of the AXI read and write address IDs (AxID[1:0]) are 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	AxID[1:0] = 00	AxID[1:0] = 01	AxID[1:0] = 10	AxID[1:0] = 11
4	Group 0	Group 1	Group 2	Group 3
6	Groups 0, 1, 2	Groups 3, 4, 5	Not applicable	Not applicable
8	Groups 0, 1	Groups 2, 3	Groups 4, 5	Groups 6, 7
16	Groups 0, 1, 2, 3	Groups 4, 5, 6, 7	Groups 8, 9, 10, 11	Groups 12, 13, 14, 15

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 requester ports on page 151 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.

Transaction type	Routed to
Device non-reorderable transactions	These transactions are sent to either: • The bus manager port which is assigned to address target group 0. • All bus manager interfaces.
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 lookup table, see Table 7-2: Mapping between address target groups and bus requester ports on page 134.
- 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 149.

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:

`0b00` All Device non-reorderable transactions are sent to address target group 0.

`0b01` Device non-reorderable transactions are sent to any manager interface port that is based on the same address interleaving as for non-Device transactions.

`0b10` Reserved

`0b11` 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 Read Receipt 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 C1-DSU 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 C1-DSU

AXI property	Supported by the C1-DSU	Interconnect or system support required
Atomic_Transactions	Yes	Optional, to send atomics to the interconnect set the BROADCASTATOMIC signal HIGH.
Barrier_Transactions	No	No
Cache_Stash_Transactions	No	No
Check_Type	No	No
Coherency_Connection_Signals	No	No
DeAllocation_Transactions	No	No
DVM_Message_Support	No	No
DVM_v8	No	No
DVM_v8.1	No	No
DVM_v8.4	No	No
Exclusive_Accesses	Yes	Yes
Loopback_Signals	No	No
Max_Transaction_Bytes	64	-
MPAM_Support	MPAM_6_1 supported by C1-DSU	Optional
MTE support	Yes	MTE support is dependent on the BROADCASTMTE signal: • If set HIGH, Standard MTE support is provided. • If set LOW, no MTE support is provided.
Multi_Copy_Atomicity	Yes	Yes
NSAccess_Identifiers	No	No
Ordered_Write_Observation	No	No
Partial_Read_Data	No	No
Persist_CMO	No	No
Poison	No	No
QoS_Accept	No	No
Read_Data_Reordering	No	No
Read_Interleaving_Disabled	No	No
Regular_Transaction_Only	Yes	No
Shareable_Transactions	No	No
Trace_Signals	No	No

AXI property	Supported by the C1-DSU	Interconnect or system support required
Unique_ID_Support	Yes	No
Untranslated_Transactions	No	No
Wakeup_Signals	Yes	Yes
WriteCMO_Transactions	No	No
WriteEvict_Transaction	No	No

8.5 AXI configurations

The AXI bus managers can be configured to support the AXI4 protocol or to broadcast Cache Maintenance Operations (CMOs).

AXI4 compatibility

The AXI manager interface of the C1-DSU 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.

Cache maintenance operations for AXI

Cache Maintenance Operations (CMOs) can be sent out on the AXI bus managers to downstream caches by setting the BROADCASTCACHEMAINT signal HIGH.

The following table shows the settings of BROADCASTCACHEMAINTdepending on your system level cache requirements.

Note

- 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.

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 C1-DSU 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 76 outstanding non-reorderable Device read transactions can be issued.
Write ID capability	Configuration dependent	Only Device memory types with nGnRnE or nGnRE can have more than one outstanding transaction with the same AXI ID. All other memory types use a unique AXI ID for every outstanding transaction.
Read ID capability	Configuration dependent	Only Device memory types with nGnRnE or nGnRE can have more than one outstanding transaction with the same AXI ID. All other memory types use a unique AXI ID for every outstanding transaction.
AWID width	10 bits	-
ARID width	10 bits	-

Note

For the read issuing and write issuing capabilities, the total issuing capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter. If there is only one 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 C1-DSU 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 C1-DSU does not generate any FIXED bursts and a burst does not cross a cache line boundary.

The C1-DSU 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 C1-DSU 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.

Note

For interoperability reasons, Arm recommends that system components fully comply with the AXI specification and do not rely on the C1-DSU behavior described here.

8.11 Barriers

The C1-DSU 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 interface

The Accelerator Coherency Port (ACP) is an optional AXI5 subordinate interface that provides coherent transaction support between the C1-DSU 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 interface allows an external manager to access memory through the main memory interface of the C1-DSU. 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 C1-DSU 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 C1-DSU, see the *RTL configuration process* chapter in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

9.1 ACP interface properties

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

Note

See *AMBA® AXI Protocol Specification* for more information on the AXI properties that are supported for ACP.

The following table shows the AXI properties for the ACP interface that are supported by the C1-DSU.

Table 9-1: AXI properties supported by the ACP interface for the C1-DSU

ACP property	Supported by the C1-DSU
Atomic_Transactions	Yes

ACP property	Supported by the C1-DSU
Cache_Stash_Transactions	Yes
Data_Check	No
DeAllocation_Transactions	No
DVM_Message_Support	Receiver
DVM_v8	Yes
DVM_v8.1	Yes
DVM_v8.4	Yes
Exclusive_Accesses	No
Loopback_Signals	No
Low_Power_Signals	Yes
Max_Transaction_Bytes	64
MPAM_Support	MPAM_9_1 supported by C1-DSU **Note:** Although this interface has 9 to 1 bits, only 6 to 1 bits are supported.
MTE_Support	MTE support is dependent on the BROADCASTMTE signal: • If set HIGH, Basic MTE support is provided. • If set LOW, no MTE support is provided.
Multi_Copy_Atomicity	Yes. System support is required.
NSAccess_Identifiers	No
Ordered_Write_Observation	No
Persist_CMO	No
PBHA_Support	Yes
Poison	No
Port_Type	Accelerator
Prefetch_Transactions	No
QoS_Accept	No
Regular_Transactions_Only	Only regular transactions are supported.
RME_Support	No
Shareable_Transactions	Yes
Trace_Signals	No
Untranslated_Transactions	No
Write_Plus_CMO	No
WriteEvict_Transaction	No
WriteZero_Transaction	No

Related information

9.2 ACP AXI protocol subset on page 162
9.3 ACP transactions on page 162

9.2 ACP AXI protocol subset

The Accelerator Coherency Port (ACP) is an AXI interface that includes support for Cacheable, Non-cacheable, and Device memory accesses.

The AXI properties for the ACP interface are listed in 9.1 ACP interface properties on page 160.These properties mean that the ACP has the following behavior:

- 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.
- The ACP does not support the Quality of Service (QoS) signals ARQOS and AWQOS.

The ACP interface does not support the following features as defined in the AMBA AXI5 protocol specification:

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

Note

For information on how to connect the ACP interface to your system, see *Functional Integration* in the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

Related information

9.1 ACP interface properties on page 160
9.3 ACP transactions on page 162
9.4 ACP performance on page 166

9.3 ACP transactions

The Accelerator Coherency Port (ACP) supports Cacheable, Non-cacheable, Device, and Atomic memory accesses.

The following table lists the AXI 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 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)
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).
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] = `0b00000`)	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] = `0b00000`)	32-byte boundary (AWADDR[4:0] = `0b00000`)	Any combination of bytes is valid.
16-byte	`0x4` (16-bytes)	`0x0` (1-beat)	INCR or WRAP	16-byte boundary (AWADDR[3:0] = `0b0000`)	16-byte boundary (AWADDR[3:0] = `0b0000`)	Any combination of bytes is valid. This includes no bytes, which mimics a `PLDW` instruction (read-unique preload).
8-byte	`0x3` (8-bytes)	`0x0` (1-beat)	INCR or WRAP	8-byte boundary (AWADDR[2:0] = `0b000`)	8-byte boundary (AWADDR[2:0] = `0b000`)	Any combination of bytes is valid.
4-byte	`0x2` (4-bytes)	`0x0` (1-beat)	INCR or WRAP	4-byte boundary (AWADDR[1:0] = `0b00`)	4-byte boundary (AWADDR[1:0] = `0b00`)	Any combination of bytes is valid.
2-byte	`0x1` (2-bytes)	`0x0` (1-beat)	INCR or WRAP	2-byte boundary (AWADDR[0] = `0b0`)	2-byte boundary (AWADDR[0] = `0b0`)	Any combination of bytes is valid.
1-byte	`0x0` (1-byte)	`0x0` (1-beat)	INCR or WRAP	Any	Any	Any combination of bytes is valid.

Stash requests can target the L2 cache of a selected core by asserting signal AWSTASHLPIDENS and indicating the selected core instance number on AWSTASHLPIDS[3:0]. See 1.7 Core, complex, SME2 unit, and processing element numbering on page 28 for a description of the core instance number.

Note

The C1-DSU generates an SLVERR in response to any of the following conditions:

- AxDOMAIN is `0b11` (System domain access) when AxCACHE is `0bxx11` (Write-Back cacheable).
- For 128-bit wide data mode, AxLEN is a value other than `0b00000011`, `0b00000001`, or `0b00000000`.
- For 256-bit wide data mode, AxLEN is a value other than `0b00000001` or `0b00000000`.
- For 128-bit wide data mode, an SLVERR is produced if either:
  - AxLEN is 00000001 and AxADDR[4:0] is a value other than `0b00000`.
  - AxLEN is 00000011 and AxADDR[5:0] is a value other than `0b000000`.
- For 256-bit wide data mode, AxADDR[5:0] is a value other than `0b000000` when AxLEN is not `0b00000000`.
- AWSNOOP is any transaction other than WriteNoSnoop, WriteUniquePtl, WriteUniqueFull, WriteUniquePtlStash, WriteUniqueFullStash, StashOnceShared, StashOnceUnique, AtomicLoad, AtomicStore, AtomicSwap, or AtomicCompare.

- ARSNOOP is any transaction other than ReadNoSnoop, ReadOnce, or DVMComplete.
- AxBURST is a value other than `0b01` or `0b10`. Only incremental or wrap bursts are supported.

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

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 interface properties on page 160
9.2 ACP AXI protocol subset on page 162
9.3 ACP transactions on page 162

9.5 DVM snoop transaction support

The Accelerator Coherency Port (ACP) of the C1-DSU 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 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 168.

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 C1-DSU 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 C1-DSU has acknowledged the request.
   When both the SYSCOREQS and SYSCOACKS signals are HIGH, the C1-DSU 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 C1-DSU 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.

9.6 ACP user bits

The Accelerator Coherency Port (ACP) supports optional user bits on the AW and AR channels.

If an ACP transaction is sent to a CHI bus requester port or a CHI-configured peripheral port then the C1-DSU propagates the user bits to the CHI port. The width of the user bits is configurable between 0-32 bits.

The ACP user bits are not stored in the L3 cache, so they are not correct for L3 evictions. The user bits are valid on the CHI port if the Logical processor ID (LPID) field indicates the source is the ACP. This is only guaranteed to happen for ACP transactions that miss in the L3 cache and the caches of the cores.

For more information on configuring the width of the user bits, see the build-time configuration parameter `ACP_USER_WIDTH` described in the section *pilatus.yaml configuration parameters* of the *Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual*.

10. AXI or CHI 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® C1-DynamIQ™ Shared Unit 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 C1-DSU 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:

ASTART0MP[PA-1:20]
: This bus defines the start address for the first address range, which is inclusive. PA is the largest physical address size of any connected core.

AEND0MP[PA-1:20]
: This bus defines the end address for the first address range, which is exclusive. PA is the largest physical address size of any connected core.

Therefore, the address range is defined as:

```
ASTART0MP[PA-1:20] <= peripheral port address range < AEND0MP[PA-1:20]
```

The first address range is configurable to granularity of 1MB. The values for the ASTART0MP[PA-1:20] and AEND0MP[PA-1:20] signals are only captured at reset. The first address range is also captured into registers IMP_CLUSTERPPSTART_EL1 and IMP_CLUSTERPPEND_EL1 and can be changed at runtime using software.

You can define the start address and end address of the remaining address ranges using the following bus signals:

ASTART<n>MP[PA-1:30]
: This bus defines the start address for the corresponding address range n, where n = 1, 2, or 3 and PA is the largest physical address size of any connected core. The start address is inclusive.

AEND<n>MP[PA-1:30]
: This bus defines the end address for the corresponding address range n, where n = 1, 2, or 3, and PA is the largest physical address size of any connected core. The end address is exclusive.

Therefore, the address range is defined as:

```
ASTART<n>MP[PA-1:20] <= peripheral port address range < AEND<n>MP[PA-1:20]
```

Note

- If an address range is not used, you must set the start address to the end address. For example, tie ASTART1MP and AEND1MP LOW.

- If DEFAULTP signal is LOW, and both ASTART0MP and AEND0MP 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 172 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 135 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 C1-DSU 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 C1-DynamIQ™ Shared Unit cluster.
- If the old address range is marked as Cacheable, then clean and invalidate all the addresses in that address range. This ensures any cached data is written back on the same interface that it originally came from. This must include a Data Synchronization Barrier (DSB) at the end to ensure the clean and invalidate has completed.

Caution

Failure to perform this invalidation could cause system deadlocks if data remains in the L3 cache for the old address range.

Procedure

1. Reprogram the IMP_CLUSTERPPSTART_EL1 and IMP_CLUSTERPPEND_EL1 registers as appropriate.
2. Execute a `DSB` and `ISB` instructions to ensure the register updates have completed.
3. You can now map the memory as required or enable the L1 and L2 caches in the cores.

Caution

During steps 1 and 2, transactions to the address range being changed might go to either the old port or the new port. Therefore, transactions to this address range might not occur in the expected order.

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 C1-DSU 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 C1-DSU	Interconnect or system support required
Atomic_Transactions	Yes	Optional, to send atomics to the interconnect set the BROADCASTATOMICMP signal HIGH.
Barrier_Transactions	No	No
Cache_Stash_Transactions	No	No
Check_Type	No	No
Coherency_Connection_Signals	No	No
DeAllocation_Transactions	No	No
DVM_v8	No	No
DVM_v8.1	No	No
Loopback_Signals	No	No

AXI property	Supported by the C1-DSU	Interconnect or system support required
MPAM_Support	Yes, the C1-DSU cluster supports MPAM_6_1.	Optional
MTE_Support	No	Not applicable
Multi_Copy_Atomicity	Yes	Yes
NSAccess_Identifiers	No	No
Ordered_Write_Observation	No	No
Partial_Read_Data	No	No
Persist_CMO	No	No
Poison	No	No
QoS_Accept	No	No
Read_Data_Reordering	No	No
Read_Interleaving_Disabled	No	No
Trace_Signals	No	No
Unique_ID_Support	Yes	No
Untranslated_Transactions	No	No
Wakeup_Signals	Yes	Yes
WriteCMO_Transactions	No	No
WriteEvict_Transaction	No	No

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 C1-DSU 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 C1-DSU	Interconnect or system support required
Atomic_Transactions	Yes	Optional, to send atomics to the interconnect set the BROADCASTATOMICMP signal HIGH.
Barrier_Transactions	No	No
Cache_Stash_Transactions	No	No
Check_Type	No	No
Coherency_Connection_Signals	No	No
DeAllocation_Transactions	No	No
DVM_v8	No	No
DVM_v8.1	No	No
Loopback_Signals	No	No

AXI property	Supported by the C1-DSU	Interconnect or system support required
MPAM_Support	Yes, the C1-DSU cluster supports MPAM_6_1.	Optional
MTE_Support	Yes	MTE support is dependent on the BROADCASTMTE signal: • If set HIGH, Standard MTE support is provided. • If set LOW, no MTE support is provided.
Multi_Copy_Atomicity	Yes	Yes
NSAccess_Identifiers	No	No
Ordered_Write_Observation	No	No
Partial_Read_Data	No	No
Persist_CMO	No	No
Poison	No	No
QoS_Accept	No	No
Read_Data_Reordering	No	No
Read_Interleaving_Disabled	No	No
Trace_Signals	No	No
Unique_ID_Support	Yes	No
Untranslated_Transactions	No	No
Wakeup_Signals	Yes	Yes
WriteCMO_Transactions	No	No
WriteEvict_Transaction	No	No

10.5 AXI 64-bit peripheral port transactions

The AXI 64-bit configured peripheral port of the C1-DSU 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 C1-DSU does not generate any FIXED bursts, and a burst does not cross a cache line boundary.

The C1-DSU generates only a subset of all possible AXI transactions on the manager interface.

For Normal Non-cacheable or Device accesses, for a 64-bit AXI configured peripheral port, the following burst types are supported:

- INCR N (N:2,4,8) 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 C1-DSU 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.

Transaction	Operation
AtomicLoad	-
AtomicStore	-
AtomicSwap	-
AtomicCompare	-

The cache linefill fetch length is always 64 bytes. The C1-DSU does not generate any FIXED bursts and a burst does not cross a cache line boundary.

The C1-DSU 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 C1-DSU 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. **Note:** The source of the transaction is encoded in the LPID field.
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	-

Note

- For the write issuing and read issuing capabilities, the total issuing capability of the cluster is the value of the `NUM_LTDBS` configuration parameter multiplied by the `NUM_L3_SLICES` parameter.

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

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 C1-DSU 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 C1-DSU cluster	Interconnect support required
Atomic_Transactions	The C1-DSU cluster supports this property if BROADCASTATOMIC is HIGH.	Yes
Cache_Stash_Transactions	Yes	Yes
Direct_Memory_Transfer	Yes	Yes
Direct_Cache_Transfer	Yes	Yes
Data_Poison	The C1-DSU cluster supports this property if cache protection is enabled.	Yes
Data_Check	No	No
CCF_Wrap_Order	No	No
Enhanced_Features	The C1-DSU cluster supports data return from SC state. The C1-DSU cluster does not support input/output deallocation transactions, for example ReadOnceMakeInvalid (ROMI) and ReadOnceCleanInvalid (ROCI). The C1-DSU cluster supports ReadNotSharedDirty transactions and requires interconnect support. If BROADCASTPERSIST is HIGH, the C1-DSU cluster supports CleanSharedPersist transactions and requires interconnect support.	Yes, if the cluster supports ReadNotSharedDirty transactions or if the BROADCASTPERSIST signal is set to HIGH.

The following table shows the different width values that the CHI-configured peripheral port supports.

Table 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 C1-DynamIQ™ Shared Unit (C1-DSU) 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 C1-DSU.

Table 10-8: CHI read and write transactions supported by C1-DSU

Transaction	Operation	Produced by C1-DSU
AtomicCompare	Atomic instruction that is not allocating inside the cluster	Yes
AtomicLoad	Atomic instruction that is not allocating inside the cluster	Yes
AtomicStore	Atomic instruction that is not allocating inside the cluster	Yes
AtomicSwap	Atomic instruction that is not allocating inside the cluster	Yes
CleanInvalid	Cache maintenance instructions	Yes
CleanShared	Cache maintenance instructions	Yes
CleanSharedPersist	Not used. CleanSharedPersistSep is used instead.	No
CleanSharedPersistSep	Cache maintenance instructions. The Data Cache Clean to the Point of Persistence (DC CVAP) cache maintenance instruction generates this transaction when the BROADCASTPERSISTMP input signal is HIGH.	Yes
CleanUnique	Not used	No
DVMOp	Translation Lookaside Buffer (TLB) and instruction cache maintenance instructions when enabled by the BROADCASTTLBIINNER, BROADCASTTLBIOUTER, and BROADCASTICINVAL input signals.	Yes
Evict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
MakeInvalid	Not used	No
MakeReadUnique	Store instructions when the line is already cached in a Shared state inside the cluster. This includes store exclusive instructions, which set Excl HIGH.	Yes
MakeUnique	Store instructions of a full cache line of data that miss in the caches	Yes
PCrdReturn	Not used	No
PrefetchTgt	Hardware prefetch hint to the memory controller	Yes
ReadClean	Reading Memory Tagging Extension (MTE) tags for a Cacheable shareable line that is already cached in the cluster without tags	Yes
ReadNoSnp	Non-cacheable loads or instruction fetches, or cache linefills of Non-shareable cache lines into L1 or L2 caches	Yes
ReadNoSnpSep	Not used	No
ReadNotSharedDirty	Cache data linefills started by a load instruction, or cache linefills started by an instruction fetch	Yes
ReadOnce	Cacheable shareable instruction fetches that are not allocating into a coherent cache	Yes
ReadOnceCleanInvalid	Not used	No
ReadOnceMakeInvalid	Not used	No
ReadPreferUnique	Speculative store to Cacheable shareable memory or, if Excl is HIGH, a load exclusive instruction	Yes
ReadShared	Not used	No
ReadUnique	Cache data linefills started by a store instruction	Yes

Transaction	Operation	Produced by C1-DSU
ReqLCrdReturn	Link credit return	Yes
StashOnceSepShared	Cache prefetch when the L3 cache is not present or powered down and configured by the CLUSTERECTLR_EL1	No
StashOnceSepUnique	Cache prefetch when the L3 cache is not present or powered down and configured by the CLUSTERECTLR_EL1	No
StashOnceShared	Not used	No
StashOnceUnique	Not used	No
WriteBackFull	Evictions of dirty cacheable shareable lines from the cluster	Yes
WriteBackFullCMO	Cache maintenance instruction evicting a dirty shareable cache line	Yes
WriteBackPtl	Not used	No
WriteCleanFull	Evictions of dirty lines from the L3 cache, when the line is still present in an L1 or L2 cache	Yes
WriteCleanFullCMO	Cache maintenance instruction cleaning a dirty shareable cache line	Yes
WriteEvictFull	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
WriteEvictOrEvict	Evictions of clean lines, when configured in the CLUSTERECTLR_EL1	Yes
WriteNoSnpFull	Non-cacheable store instructions. Evictions of Non-shareable cache lines.	Yes
WriteNoSnpFullCMO	Cache maintenance instruction evicting a dirty Non-shareable cache line	Yes
WriteNoSnpPtl	Non-cacheable store instructions	Yes
WriteNoSnpPtlCMO	Not used	No
WriteNoSnpZero	Write of zeroes to Non-cacheable or Non-shareable memory using the DC ZVA instruction	Yes
WriteUniqueFull	Cacheable writes of a full cache line not allocating into L1, L2, or L3 caches, for example streaming writes	Yes
WriteUniqueFullCMO	Not used	No
WriteUniqueFullStash	Not used	No
WriteUniquePtl	Generated as a result of Accelerator Coherency Port (ACP) WriteUniquePtl transactions when not allocating to the L3 cache	Yes
WriteUniquePtlCMO	Not used	No
WriteUniquePtlStash	Not used	No
WriteUniqueZero	Write of zeroes to a Shareable cache line using the DC ZVA instruction	Yes

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 cores and SME2 units configured in your cluster.

64-bit AXI peripheral port read and write capabilities

The maximum read or write issuing capability is 4 x (CN + CM + 3), where CN and CM are defined as follows:

- CN is the total number of cores in the cluster, including those in complexes, and can range from 1 to 14.
- CM is the total number of SME2 units in the cluster, can range from 0 to 2.

Therefore, the maximum issuing capability for a cluster of 14 cores and 2 SME2 units is 76 outstanding reads or writes.

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	4 x (CN + CM + 3)	**Note:** The DSU does not limit the amount of transactions from each source, provided the total of all transactions does not exceed the maximum value for the configuration.
Read issuing capability	4 x (CN + CM + 3)	
Write ID capability	Configuration dependent	All transactions from a given source use the same AXI ID.
Read ID capability	Configuration dependent	All transactions from a given source 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	`0xdd`	The encodings of dd are: **`0x00 - 0x0D`** Core n, where n ranges 0 to 13. **`0x0E`** L3 eviction **`0x0F`** Accelerator Coherency Port (ACP) 0 **`0x10`** ACP 1 **`0x11`** SME2 unit 0 **`0x12`** SME2 unit 1

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 155 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:

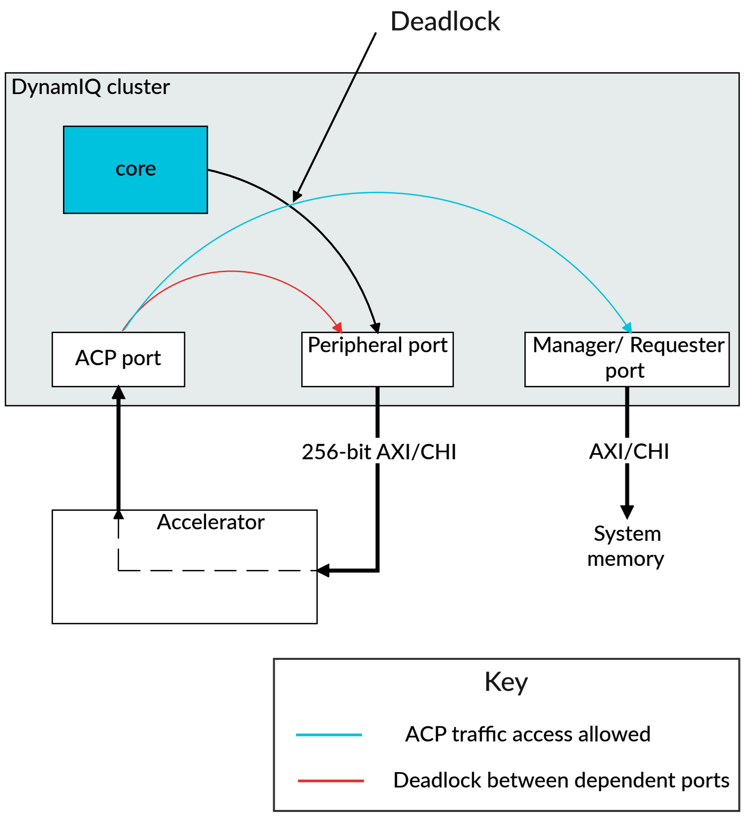
- For connecting to a tightly-coupled accelerator.
- For use as a more general system port.

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 C1-DSU is connected to a tightly-coupled accelerator.

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



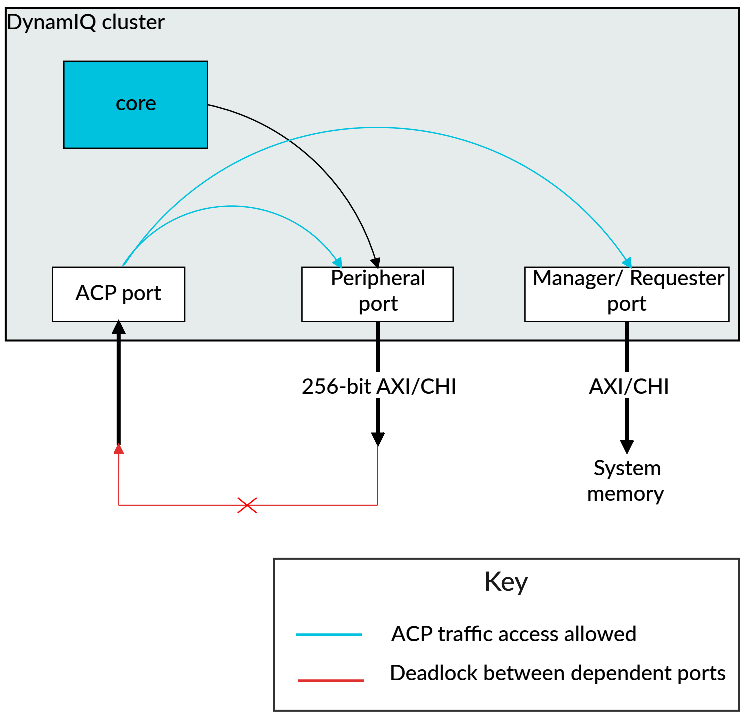
When the peripheral port is connected to a tightly-coupled accelerator, the accelerator might have an internal dependency, this is shown by the dashed line in the preceding figure. This means, a read or write to the registers of the accelerator through the peripheral port cannot complete until an outstanding 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 ARPROTP0[0] and AWPROTP0[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 C1-DSU 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 C1-DSU 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 C1-DSU. 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®v9.0-A Architecture Reference Manual Armv9, for Armv9-A architecture profile*.

11.1 Cache protection behavior

The configuration of the Reliability, Availability, Serviceability (RAS) Extension that is implemented in the C1-DSU includes Error Correcting Code (ECC) cache protection. In this case, the C1-DSU protects against errors that result in a RAM bitcell holding the incorrect value.

The RAMs in the C1-DSU support Single Error Correct Double Error Detect (SECDED). SECDED allows detection and correction of any 1-bit error, and detection of any 2-bit error in all protected RAMs. When the datum and code bits are all-zero, or all-one, the interpretation is that an error has occurred that the Error Correcting Code (ECC) scheme cannot correct. However, it might be corrected by other means, such as refetching cached data.

The following table describes the protection type is applied to each RAM. The C1-DSU 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, SECDED	9 ECC bits per 132 bits

RAM	Protection	Description
L3 cache tag	ECC, SECDED	The number of ECC bits for each 58-bits depend on the size of entry as follows: • If the tag entry is 58 bits wide, there are 8 ECC bits. • If the tag entry is 57 bits wide or less, there are 7 ECC bits.
L3 cache victim	None	-
Long-Term Data Buffer (LTDB) RAMs	ECC, SECDED	9 ECC bits per 145 bits
Snoop filter RAMs	ECC, SECDED	7 ECC bits per 48 bits

Error correction

If there are multiple single bit errors in different RAMs, or within different protection granules within the same RAM, then the C1-DSU remains functionally correct.

If there is a double bit error in a single RAM within the same protection granule, the C1-DSU 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 C1-DSU might or might not detect the errors. The cache protection feature of the C1-DSU 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 C1-DSU 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 C1-DSU 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.2.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register on page 313 or B.1.1.11 CLUSTERECTLR, Cluster Extended Control Register on page 413, 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 C1-DSU 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 C1-DSU 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 C1-DSU does not poison the data transaction of a Distributed Virtual Message (DVM) operation it initiates because the data payload of a DVM operation is not derived from cached data.

If the C1-DSU receives a poisoned DVM operation from a core, it consumes the error and reports it as an uncorrectable error (`producer error`). If the C1-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 C1-DSU 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 C1-DSU detects a fault, it raises a Fault Handling Interrupt (FHI) exception through the fault signals. FHIs are reflected in the Error Data Record Registers that are updated in the node that detects the errors.

Fault handling interrupt

When ERR0CTLR.FI is set, all Deferred errors and Uncorrected errors that the C1-DSU detects generate an FHI through the nCLUSTERFAULTIRQ signal.

When ERR0CTLR.CFI or any other CE-counter overflow bits are set, then all detected Corrected errors also cause an FHI to be generated.

Error recovery interrupt

When ERR0CTLR.UI is set, all Uncorrected errors that are detected and not deferred generate an error recovery interrupt through the nCLUSTERERRIRQ signal.

Critical error interrupt

When ERR0CTLR.CI is set, all critical errors that the C1-DSU detects generate a critical error interrupt on the nCLUSTERCRITIRQ signal.

Clearing reported faults

The signals nCLUSTERFAULTIRQ, nCLUSTERERRIRQ, and nCLUSTERCRITIRQ remain asserted until software clears them by writing to the ERR0STATUS register.

11.4 Error detection and reporting

When the C1-DSU consumes an error, it raises an Error Recovery Interrupt (ERI).

Error detection and reporting registers

The following registers are provided:

- The cluster Error Record Feature Register, CLUSTERRAS_ERR0FR. This is a read-only register that specifies various error record settings.

- The cluster Error Record Control Register, CLUSTERRAS_ERR0CTLR.
- The cluster Error Record Miscellaneous Register 0-3, CLUSTERRAS_ERR0MISC0-3. These registers record details of the error location and counts.
- The cluster Pseudo-fault Generation Feature register, CLUSTERRAS_ERR0PFGF. Read-only register.
- The cluster Error Record Primary Status Register, CLUSTERRAS_ERR0STATUS.

The cluster Reliability, Availability, and Serviceability (RAS) registers are accessible either from memory-mapped accesses on the utility bus or from System register accesses from the cores.

Error types

The following describes the different types of errors that can occur in the C1-DSU and their effects:

- Corrected errors.
- Uncorrectable errors in the L3 data RAMs when read by a core can cause a precise or imprecise Data Abort or Prefetch Abort, depending on the implementation of the core.
- Uncorrectable errors in the L3 data RAMs in a line when this line is being evicted from a cache cause the data to be poisoned. The eviction might be because of a natural eviction, a linefill from a higher level of cache, a cache maintenance operation, or a snoop. If the poisoned line is evicted from the cluster for any reason and the interconnect does not support data poisoning, then the nCLUSTERERRIRQ signal is asserted.
- Uncorrectable errors in the L3 tag RAMs or Snoop Control Unit (SCU) filter RAMs cause the nCLUSTERERRIRQ signal to be asserted.

Note

Arm recommends that the ERRIRQ signals are connected to the interrupt controller, so that an interrupt or system error is generated when the signals are asserted.

The fault and error interrupt pins can be cleared by writing to the CLUSTERRAS_ERR0STATUS register.

When a dirty cache line with an error on the data RAMs is evicted from the cluster, the write on the requester interface still takes place. However, if the error is uncorrectable then:

- If the C1-DSU is configured with an AXI manager-port, the uncorrected data is written and the error is reported in the RAS registers.
- If the C1-DSU 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 C1-DSU 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 242

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 C1-DSU detects an Error Correcting Code (ECC) error in the L3 data RAM, the C1-DSU 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 C1-DSU detects an error in the L3 data RAM, the C1-DSU 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 C1-DSU reports a second error in the Error Record registers. Therefore, an ECC error in the L3 Data RAM is reported as if two errors occurred.

An error on a single read of the L3 data RAM results in the following error record contents, assuming the Error Record was initially empty:

- A 1-bit error increments the ERR0MISC0.CECO due to the reporting of a second Correctable Error. The contents of the ERR0STATUS accurately shows that the error came from the L3 Data RAM. For example, ERR0STATUS.SERR=6, ERR0STATUS.V=1 and ERR0STATUS.CE=1.

- A 2-bit error might be Deferred or Uncontainable depending on whether the target of the data supports poison. This is determined during the LTDB RAM read. The L3 data RAM always generates a Deferred Error, if there is a 2-bit error.

  Depending on if the error is Deferred or Uncontainable, the Error Record is updated as follows:

  - For a Deferred error, the contents of the Error Record accurately shows the error that came from the L3 data RAM. For example, ERR0STATUS.V=1, ERR0STATUS.DE=1 and ERR0STATUS.SERR=6. However, the extra error from the LTDB RAM also sets ERR0STATUS.OF=1.
  - For an Uncontainable error, the contents of the Error Record shows the LTDB RAM error. However, it does not provide details of the original L3 data RAM error. For example, ERR0STATUS.V=1, ERR0STATUS.UE=1, ERR0STATUS.SERR=2. The extra error also means that ERR0STATUS.OF=1. Also even though L3 data RAM poisoned the data, ERR0STATUS.PN=0.

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 C1-DSU 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.

Note

The error injection mechanism only injects pseudo fault reports into the error reporting registers for the purposes of testing error handling and error identification software in real systems. It does not inject actual errors into the hardware.

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 and 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 powers 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, core, or SME2 unit 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, SME2 unit, or cluster are asserted.

   The RAS interrupt signals are n<type>ERRIRQ, n<type>FAULTIRQ, and n<type>CTITIRQ, where type can be CLUSTER, CORE, COMPLEX, or SME2 unit. For example:

   - nCLUSTERFAULTIRQ
   - nCOREFAULTIRQ[CN:0]
   - nCOMPLEXFAULTIRQ[CX:0]
   - nCMEFAULTIRQ[CM:0]
   - nCMEERRIRQ[CM:0]

2. If the Power Policy Unit (PPU) is currently transitioning to an OFF or MEM_RET power modes, then these requests to 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 1.7 Core, complex, SME2 unit, and processing element numbering on page 28

11.7 Cluster RAS registers

The cluster Reliability, Availability, and Serviceability (RAS) registers are treated as a separate node in the memory-mapped view. The cluster RAS registers are accessible either from memory-mapped accesses on the utility bus or from System register accesses from the cores. You must access the cluster RAS registers from the Secure address space.

Note

- The cluster RAS registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
- Any address that is not documented is treated as **RAZ/WI**.
- The cluster RAS registers are not present if Realm Management Extension (RME) is enabled. This is because when RME is enabled, the cluster is in Direct connect and the cluster RAS registers are not present for Direct connect configurations.

11.7.1 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**. Therefore, if the C1-DSU is enabled for Realm Management Extension (RME), none of these registers are present.

- 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	Present in Direct connect
ERXFR_EL1	3	0	C5	C4	0	See individual bit resets.	64-bit	Selected Error Record Feature Register	No
ERXCTLR_EL1	3	0	C5	C4	1	See individual bit resets.	64-bit	Selected Error Record Control Register	No
ERXSTATUS_EL1	3	0	C5	C4	2	See individual bit resets.	64-bit	Selected Error Record Primary Status Register	No
ERXPFGF_EL1	3	0	C5	C4	4	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Feature Register	No
ERXPFGCTL_EL1	3	0	C5	C4	5	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Control Register	No
ERXPFGCDN_EL1	3	0	C5	C4	6	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Countdown Register	No
ERXMISC0_EL1	3	0	C5	C5	0	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 0	No
ERXMISC1_EL1	3	0	C5	C5	1	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 1	No
ERXMISC2_EL1	3	0	C5	C5	2	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 2	No
ERXMISC3_EL1	3	0	C5	C5	3	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 3	No

11.7.2 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If Realm Management Extension (RME) is enabled, meaning that the cluster is in Direct connect, these registers are not present. For more information on enabling RME, see 1.4.1 Realm management extension on page 26.
- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- 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	Present in Direct connect
0x0	CLUSTERRAS_ERR0FR	See individual bit resets.	64-bit	Error Record <n> Feature Register	No
0x8	CLUSTERRAS_ERR0CTLR	See individual bit resets.	64-bit	Error Record <n> Control Register	No
0x010	CLUSTERRAS_ERR0STATUS	See individual bit resets.	64-bit	Error Record <n> Primary Status Register	No
0x018	CLUSTERRAS_ERR0ADDR	See individual bit resets.	64-bit	Error Record Address Register	No
0x20	CLUSTERRAS_ERR0MISC0	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 0	No
0x28	CLUSTERRAS_ERR0MISC1	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 1	No
0x30	CLUSTERRAS_ERR0MISC2	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 2	No
0x38	CLUSTERRAS_ERR0MISC3	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 3	No
0x800	CLUSTERRAS_ERR0PFGF	See individual bit resets.	64-bit	Error Record <n> Pseudo-fault Generation Feature Register	No
0x808	CLUSTERRAS_ERR0PFGCTL	See individual bit resets.	64-bit	Error Record <n> Pseudo-fault Generation Control Register	No
0x810	CLUSTERRAS_ERR0PFGCDN	See individual bit resets.	64-bit	Error Record <n> Pseudo-fault Generation Countdown Register	No
0xE00	CLUSTERRAS_ERRGSR	See individual bit resets.	64-bit	Error Group Status Register	No
0xE10	CLUSTERRAS_ERRIIDR	See individual bit resets.	32-bit	Implementation Identification Register	No
0xFA8	CLUSTERRAS_ERRDEVAFF	See individual bit resets.	64-bit	Device Affinity Register	No
0xFBC	CLUSTERRAS_ERRDEVARCH	See individual bit resets.	32-bit	Device Architecture Register	No
0xFC8	CLUSTERRAS_ERRDEVID	See individual bit resets.	32-bit	Device Configuration Register	No
0xFD0	CLUSTERRAS_ERRPIDR4	See individual bit resets.	32-bit	Peripheral Identification Register 4	No
0xFD4	CLUSTERRAS_ERRPIDR5	See individual bit resets.	32-bit	Peripheral Identification Register 5	No
0xFD8	CLUSTERRAS_ERRPIDR6	See individual bit resets.	32-bit	Peripheral Identification Register 6	No
0xFDC	CLUSTERRAS_ERRPIDR7	See individual bit resets.	32-bit	Peripheral Identification Register 7	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFE0	CLUSTERRAS_ERRPIDR0	See individual bit resets.	32-bit	Peripheral Identification Register 0	No
0xFE4	CLUSTERRAS_ERRPIDR1	See individual bit resets.	32-bit	Peripheral Identification Register 1	No
0xFE8	CLUSTERRAS_ERRPIDR2	See individual bit resets.	32-bit	Peripheral Identification Register 2	No
0xFEC	CLUSTERRAS_ERRPIDR3	See individual bit resets.	32-bit	Peripheral Identification Register 3	No
0xFF0	CLUSTERRAS_ERRCIDR0	See individual bit resets.	32-bit	Component Identification Register 0	No
0xFF4	CLUSTERRAS_ERRCIDR1	See individual bit resets.	32-bit	Component Identification Register 1	No
0xFF8	CLUSTERRAS_ERRCIDR2	See individual bit resets.	32-bit	Component Identification Register 2	No
0xFFC	CLUSTERRAS_ERRCIDR3	See individual bit resets.	32-bit	Component Identification Register 3	No

12. Utility bus

The utility bus provides access to control registers for various system components in the C1-DSU, the cores, and the SME2 unit within the C1-DSU 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, each of the cores, and each SME2 unit
- Cluster control registers, including the L3 cache power-related monitors
- Reliability, Availability, and Serviceability (RAS) registers for the cluster, cores, and SME2 units
- Memory Partitioning and Monitoring (MPAM) registers for the cluster
- Activity Monitor Unit (AMU) registers in the cores and cluster
- Max Power Mitigation Mechanism (MPMM) registers in the cores and cluster

Note

Information about the PPU registers for the cores and the SME2 units 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). For information on all other SME2 unit registers accessed from the utility bus, see *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*.

12.1 Utility bus interface properties

The C1-DSU utility bus is implemented as a 64-bit AMBA AXI5 subordinate port. AMBA defines a set of properties for AXI for which the utility bus only supports a subset of these.

The following table shows which AXI properties the utility bus supports.

Table 12-1: AXI interconnect properties for the utility bus

AXI property	Supported by the utility bus	Interconnect or system support required
Atomic_Transactions	No	No
Barrier_Transactions	No	No
Cache_Stash_Transactions	No	No
Check_Type	No	No
CMO_On_Read	No	No
CMO_On_Write	No	No
Coherency_Connection_Signals	No	No
DeAllocation_Transactions	No	No
DVM_Message_Support	No	No
DVM_v8	No	No

AXI property	Supported by the utility bus	Interconnect or system support required
DVM_v8.1	No	No
DVM_v8.4	No	No
Exclusive_Accesses	No	No
Loopback_Signals	No	No
Max_Transaction_Bytes	64	-
MPAM_Support	No	No
MTE_Support	No	No
Multi_Copy_Atomicity	Yes	No
NSAccess_Identifiers	No	No
Ordered_Write_Observation	No	No
Persist_CMO	No	No
Poison	No	No
Prefetch_Transactions	No	No
QoS_Accept	No	No
Read_Interleaving_Disabled	Yes	No
Shareable_Transactions	No	No
Trace_Signals	No	No
Unique_ID_Support	No	No
Untranslated_Transactions	No	No
Wakeup_Signals	Yes	Yes
Write_Plus_CMO	No	No
WriteEvict_Transaction	No	No
WriteZero_Transaction	No	No

12.2 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 12-4: Utility bus base addresses for system component

registers on page 203 . 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 12-2: 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.

12.2.1 Core access to system component registers

Some of the system component registers are only available through memory-mapped accesses on the utility bus. For these registers, there is no direct access to the registers from the cores. If you require memory-mapped access from the cores, Arm® recommends allowing your interconnect to provide a loopback address mapping for the cores to access the utility bus through your interconnect.

The following table shows which system components are directly accessible from the cores using System register access instructions. Note that all the registers are accessible through the utility bus.

Table 12-3: System component registers accessible from cores

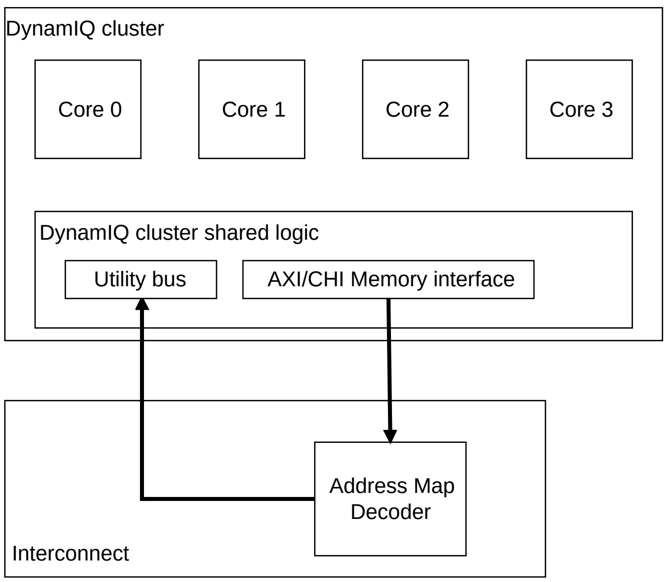
Registers	Directly accessible from cores
Cluster power control	Yes
Cluster Memory System Resource Partitioning and Monitoring (MPAM)	No
Cluster Reliability, Availability, and Serviceability (RAS)	Yes
Cluster Power Policy Unit (PPU)	No
Cluster Activity Monitor Unit (AMU)	No
Core PPU	No
SME2 unit PPU	No
CLUSTERCTL	No

Note

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



12.2.2 Cluster, core, and SME2 unit PPU register access

The Power Policy Unit (PPU) registers for the cluster and each core and SME2 unit are still accessible when the cluster is powered down.

If a core is powered down, then any access to a core register (not including the PPU registers) is treated as **RAZ/WI**.

If the cluster is powered down, then any access to a cluster register (not including the PPU registers) is treated as **RAZ/WI**.

If an SME2 unit is powered down but the cluster is powered up (ON power mode), any access to the SME2 unit registers proceeds normally without waking up the SME2 unit. If the cluster is powered down then any access to an SME2 register (not including the PPU registers) is treated as **RAZ/WI**.

Note

- The PPUs for the cluster, each of the cores, and each SME2 unit are still accessible when the cluster is powered down.
- The PPU registers for an core or an SME2 unit are still accessible when that core or SME2 unit is powered down.

12.3 Base addresses for system components

Each set of System registers is grouped on separate 64KB page boundaries allowing access to be enforced by a Memory Management Unit (MMU).

The following table shows the base addresses for each set of system component registers and what Security state they should be accessed from.

Note

- The base address for each set of registers for the core and SME2 unit Power Policy Units (PPUs) depends on the core and SME2 unit programming instance number <y>, ranging from 0 to CN + CM + 1, where CN is the number of cores in the cluster -1 and CM is the number of SME2 units in the cluster - 1. For a definition of y, CN, and CM, see 1.7 Core, complex, SME2 unit, and processing element numbering on page 28.
- Each SME2 unit is addressed as if it were an additional core in the cluster. Therefore, y = CN + 1 for the first SME2 unit instance and y = CN + 2 for the second SME2 unit instance.
- 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<y>90000` - `0x<y>F0000`, see your core Technical Reference Manual (TRM).
- For base addresses of SME2 unit registers mapped from `0x<y>90000` - `0x<y>F0000`, see the *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*.
- 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.
- Realm Management Extension (RME) is supported if the core supports RME and the cluster is in Direct connect. For more information on RME, see 1.4.1 Realm management extension on page 26.

Table 12-4: Utility bus base addresses for system component registers

Base address, y is the core and SME2 unit programming instance number	Registers	Security state if Realm Management Extension (RME) is supported, Direct connect configurations only	Security state if RME is not present	Memory map
0x000000	Cluster control	Secure state if the LEGACYTZEN signal is HIGH, otherwise Root state	Secure state	B.1.1 External cluster system control registers summary on page 391
0x010000	Cluster MPAM	Not present	Any state	B.1.2 External MPAM registers summary on page 423
0x020000	Cluster RAS	Not present	Secure state	B.1.3 External cluster RAS registers summary on page 458
0x030000	Cluster PPU	Secure state if the LEGACYTZEN signal is HIGH, otherwise Root state.	Secure state	B.1.4 External cluster PPU registers summary on page 530
0x040000	Activity Monitors	Secure state if the LEGACYTZEN signal is HIGH, otherwise Root state	Secure state	B.1.5 External cluster AMU registers summary on page 588
0x050000	CLUSTERCTL	Secure state if the LEGACYTZEN signal is HIGH, otherwise Root state	Secure state	B.1.6 External CLUSTERCTL registers summary on page 630
0x060000 - 0x070000	Reserved for future cluster registers	-	-	-
0x<y>80000	Core <y> PPU	Secure state if the LEGACYTZEN signal is HIGH, otherwise Root state	Secure state	B.1.7 External core and SME2 unit PPU registers summary on page 654
0x<y>90000 - 0x<y>F0000	Core <y> registers	See your core TRM	See your core TRM	See your core TRM
0x<y>80000	SME2 unit <y> PPU	Not present	Secure state	B.1.7 External core and SME2 unit PPU registers summary on page 654
0x<y>90000 - 0x<y>F0000	SME2 unit <y> registers	Not present	See *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*	See *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*

13. System control registers

The system control registers control and provide status information for the functions that the C1-DSU 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- Any AArch64 Generic System Control registers that are not present in Direct connect are treated as **RAZ/WI**.
- 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	Present in Direct connect
IMP_CLUSTERCFR_EL1	3	0	C15	C3	0	See individual bit resets.	64-bit	Cluster Configuration Register	Yes
IMP_CLUSTERIDR_EL1	3	0	C15	C3	1	See individual bit resets.	64-bit	Cluster Main Revision Register	Yes
IMP_CLUSTERREVIDR_EL1	3	0	C15	C3	2	See individual bit resets.	64-bit	Cluster ECO ID Register	Yes
IMP_CLUSTERACTLR_EL1	3	0	C15	C3	3	See individual bit resets.	64-bit	Cluster Auxiliary Control Register	Yes
IMP_CLUSTERECTLR_EL1	3	0	C15	C3	4	See individual bit resets.	64-bit	Cluster Extended Control Register	No
IMP_CLUSTERPWRCTLR_EL1	3	0	C15	C3	5	See individual bit resets.	64-bit	Cluster Power Control Register	No
IMP_CLUSTERPWRDN_EL1	3	0	C15	C3	6	See individual bit resets.	64-bit	Cluster Power Down Register	No
IMP_CLUSTERPWRSTAT_EL1	3	0	C15	C3	7	See individual bit resets.	64-bit	Cluster Power Status Register	No
IMP_CLUSTERL3DNTH0_EL1	3	0	C15	C4	0	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register	No
IMP_CLUSTERL3DNTH1_EL1	3	0	C15	C4	1	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register	No
IMP_CLUSTERL3UPTH0_EL1	3	0	C15	C4	2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register	No

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERL3UPTH1_EL1	3	0	C15	C4	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register	No
IMP_CLUSTERBUSQOS_EL1	3	0	C15	C4	4	See individual bit resets.	64-bit	Cluster Bus QoS Control Register	No
IMP_CLUSTERL3HIT_EL1	3	0	C15	C4	5	See individual bit resets.	64-bit	Cluster L3 Hit Counter Register	No
IMP_CLUSTERL3MISS_EL1	3	0	C15	C4	6	See individual bit resets.	64-bit	Cluster L3 Miss Counter Register	No
IMP_CLUSTERPPSTART_EL1	3	0	C15	C9	0	See individual bit resets.	64-bit	Cluster Peripheral Port Start Address Register	No
IMP_CLUSTERPPEND_EL1	3	0	C15	C9	1	See individual bit resets.	64-bit	Cluster Peripheral Port End Address Register	No
IMP_CLUSTERCFR2_EL1	3	0	C15	C9	2	See individual bit resets.	64-bit	Cluster Configuration Register 2	No
IMP_CLUSTERL3UPTH2_EL1	3	0	C15	C9	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register	No
IMP_CLUSTERCDBG_EL3	3	6	C15	C4	7	See individual bit resets.	64-bit	Cluster Cache Debug Register	No
IMP_CLUSTERPMMDCR_EL3	3	6	C15	C6	3	See individual bit resets.	64-bit	Monitor Debug Configuration Register (EL3)	No

14. Debug

The C1-DSU 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 C1-DSU. 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:

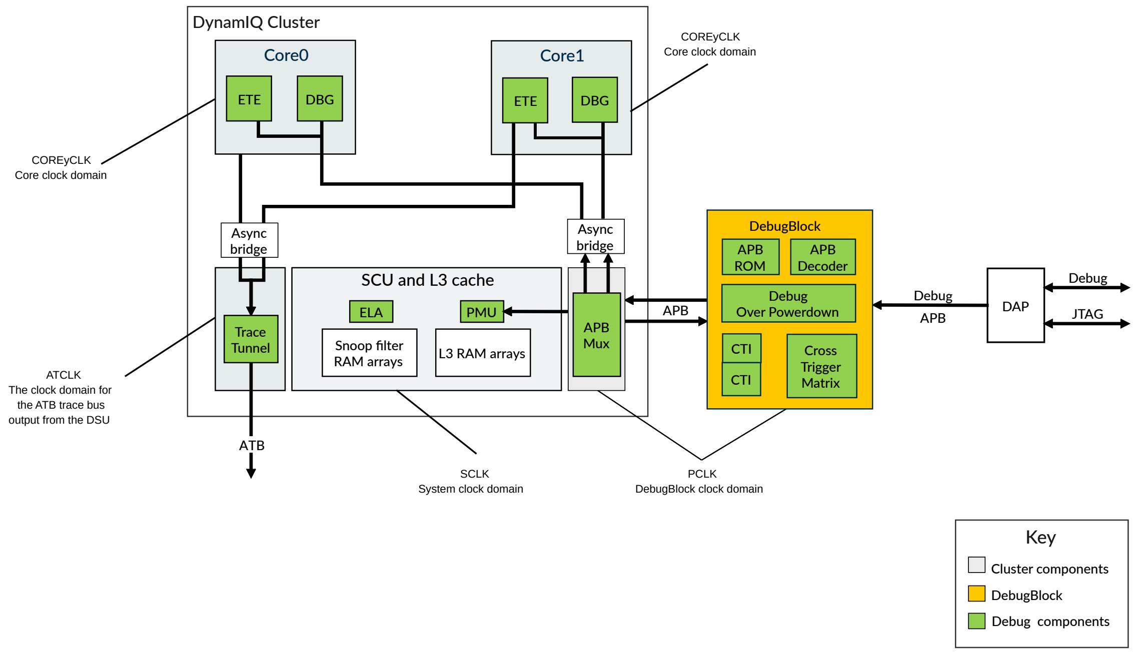
- The DebugBlock is placed in a separate power domain, to ensure that it is possible to maintain the connection to a debugger while the cores and cluster are powered down.
- The DebugBlock is physically placed with the other CoreSight logic in the SoC, rather than close to the cluster.

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 C1-DSU implements the following CoreSight debug components:

- Per-core Embedded Trace Extension (ETE). Although the ETE is supplied with the core, the C1-DSU 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 C1-DSU 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 C1-DSU with the physical location to read, and the C1-DSU 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 C1-DSU 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 C1-DSU cache debug information is provided through the C1-DSU 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 C1-DSU 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 whether 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	The L3 cache Set locations in each cache slice are all power-of-2 in size and therefore can be identified using contiguous index locations. The Set index values for slice 0 start from value zero in this field, followed by the index locations for slice 1, and then sequentially up to the total number of cache slices. The total index width varies depending on the size of the RAM being accessed. The cache slice identification number, slice ID, forms the upper used bits of the cache location encoding in this field. For details on tag index widths, see Table 14-3: Tag index width for L3 RAM accesses on page 210. For details on slice ID widths see Table 14-2: Slice ID width on page 209. 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.
[5:3]	CHUNK	Select of 64-bit data chunk to read from 512-bit data RAM cache line. Only used when accessing data RAM data. **0b000** Data[63:0] **0b001** Data[127:64] **0b010** Data[191:128] **0b011** Data[255:192] **0b100** Data[319:256] **0b101** Data[383:320] **0b110** Data[447:384] **0b111** Data[511:448]
[2:0]	RAM	RAM to be accessed. All other values are reserved. **0b001** Snoop Filter RAM **0b010** Tag RAM **0b011** Data RAM - accessing cacheline data **0b111** Data RAM - accessing cacheline Memory Tagging Extension (MTE) tags

The following table shows how to determine the slice ID width from the number of cache slices configured:

Table 14-2: Slice ID width

Number of cache slices	Slice ID width
1	0
2	1

Number of cache slices	Slice ID width
4	2
8	3

For L3 RAM accesses, the following table shows how to determine the tag index width from the cache size per slice:

Table 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 Snoop Filter (SF) RAM accesses, the following table shows how to determine the snoop filter index widths from the cache size per slice:

Note

For certain sizes of L3 Tag and snoop filter RAM, the index width is 12 bits or more. However, these RAMs have only an 11-bit address space and there are multiple RAM banks. The Least Significant Bit (LSBs) of the `SLCID_IDX` field determine which RAM bank is accessed.

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, 5120KB	13

The following table describes how to interpret RAM data read back from the C1-DSU CLUSTERCDBG register, for a snoop filter access.

Table 14-5: CLUSTERCDBG bit descriptions for a snoop filter RAM access

Bits	Width (bits)	Description
[63:56]	8	Snoop filter Error Correcting Code (ECC)
[55:MAX_CMPXS+40]	16 - MAX_CMPXS	RAZ
[MAX_CMPXS+39:40]	MAX_CMPXS	One bit per standalone core or per complex. When a bit is 1 it identifies a core or complex where the cache line is allocated. When a bit is 0 it indicates the cache line is not allocated in this core or complex.

Bits	Width (bits)	Description
[39:38]	2	This field has the following values: **0b00** Cache line is invalid **0b01** Cluster received a shared copy of the cache line. The cores know it is shared. **0b10** Cluster received a unique copy of the cache line and has given a unique copy to a core or complex (the core thinks it is unique). **0b11** Cluster received a unique copy of the cache line and has given a shared copy to one or more complexes (the core thinks it is shared).
[37:26]	12	RAZ
[25]	1	NS (Non-Secure). This bit has the following values: **0** The cache line is Secure **1** The cache line is Non-secure
[24:0]	25	Physical address tag. The encoding of these bits depends on IMP_CLUSTERCFR_EL1.SFIDX as follows: **0x9** `{PA[39:15]}` **0xA** `{PA[39:16},1'b0}` **0xB** `{PA[39:17},2'b00}` **0xC** `{PA[39:18},3'b000}`

The following table describes how to interpret RAM data read back from the C1-DSU 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
[57]	1	Memory System Resource Partitioning and Monitoring (MPAM) - PMG bit. If MPAM values are stored in the cache, then this bit saves the MPAM PMG value.
[56:51]	6	MPAM - PartID. If MPAM values are stored in the cache, then these bit saves the MPAM PMG value. bits save the MPAM PARTID value.

Bits	Width (bits)	Description
[50]	1	MPAM - NS. If MPAM values are stored in the cache, then this bit saves the MPAM PMG value. bit indicates if it is a Non-secure state PARTID or a Secure state PARTID.
[49:46]	4	Page-Based Hardware Attribute (PBHA). If the PBHA bits are stored in the cache, then these bits report the PBHA values for this cache line.
[45:44]	2	MTE state. If MTE values are stored in the cache, then these bits save the MTE values for this line. The possible values are: **0b00** MTE tag is Invalid **0b01** MTE tag is Clean **0b10** MTE tag is Dirty (the tag value for the cache line has been modified using an instruction such as `STG`.
[43]	1	OA (Outer Allocation). The possible values are: **0** This hints that the system should not allocate the cache line. **1** This hints that the system should allocate the cache line.
[42]	1	PF (PreFetch). The possible values are: **0** This hints that the cache line is not considered to be a prefetch (the cache line has been used). **1** This hints that the cache line is an unused prefetch.
[41]	1	CP (CPU Presence). The possible values are: **0** Indicates that there is no snoop filter entry for this cache line. **1** Indicates a snoop filter entry for this cache line.

Bits	Width (bits)	Description
[40:39]	2	Tag RAM State. The possible values are: **0b00** Cache line entry Invalid **0b01** UniqueDirty. The cluster has a unique, modified (dirty) copy of the cache line. **0b10** SharedClean. The cluster has a shared copy of the cache line that is coherent with the external memory location. **0b11** UniqueClean. The cluster has a unique copy of the cache line.
[38:27]	12	RAZ
[26]	12	NS (Non-Secure). The possible values are: **0** The cache line is Secure. **0** The cache line is Non-secure.
[25:0]	26	Physical Address Tag Encoding varies depending on the number of L3 sets divided by the number of L3 cache slices. The number of L3 sets can be found by writing `0x4` to CSSELR_EL1, and then calculating CCSIDR_EL1.NumSets + 1. The number of L3 cache slices can be found from the IMP_CLUSTERCFR_EL1.L3SLC. For (L3 sets/L3 cache slices), the possible values are: **256** {PA[39:14]} **512** {PA[39:15],1'b0} **1024** {PA[39:16],2'b00} **2048** {PA[39:17],3'b000} Where PA is the Physical Address width.

The following table describes how to interpret the data read back from the C1-DSU 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 C1-DSU CLUSTERCDBG register, for a Data RAM tag value access:

Table 14-8: CLUSTERCDBG bit descriptions for a L3 data RAM MTE tag value and L3 tag RAM and L3 data RAM ECC access

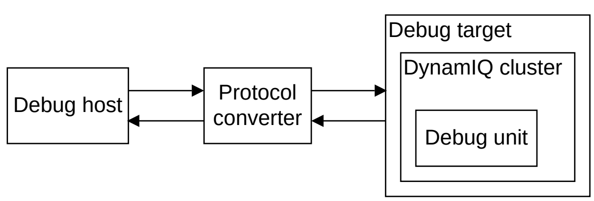
Bits	Width (bits)	Description
[63:60]	-	RAZ
[59:52]	8	L3 tag RAM Error Correcting Code (ECC)
[51:16]	36	L3 data RAM ECC
[15:12]	4	MTE tag for selected cache line bits [511:384]
[11:8]	4	MTE tag for selected cache line bits [383:256]
[7:4]	4	MTE tag for selected cache line bits [255:128]
[3:0]	4	MTE tag for selected cache line bits [127:0]

14.2 Supported debug methods

The C1-DSU 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 C1-DSU based devices, the mechanism used to access the debug unit is based on the CoreSight architecture. The C1-DSU 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 C1-DSU.

Debug unit

Helps debugging software that is running on the core:

- C1-DSU 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 C1-DSU 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

1.7 Core, complex, SME2 unit, and processing element numbering on page 28

14.4 Cluster and core Debug power control

The cores included in the C1-DSU, 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 the cores, the SME2 units, 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_DBGPCR0) 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_DBGPCR0-CLUSTERROM_DBGPCR15).

Note

- 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.2.3.19 DBROM_DBGPCR0, DebugBlock ROM table Debug Power Control Register 0 on page 946
B.2.3.20 DBROM_DBGPSR0, DebugBlock ROM table Debug Power Status Register 0 on page 948
B.2.2.19 CLUSTERROM_DBGPCR0, Cluster ROM table Debug Power Control Register on page 862
B.2.2.35 CLUSTERROM_DBGPSR0, Cluster ROM table Debug Power Status Register on page 882
B.2.1.31 CTIDEVCTL, CTI Device Control register on page 780
B.2.3 External debug ROM registers summary on page 920
B.2.2 External cluster ROM registers summary on page 808

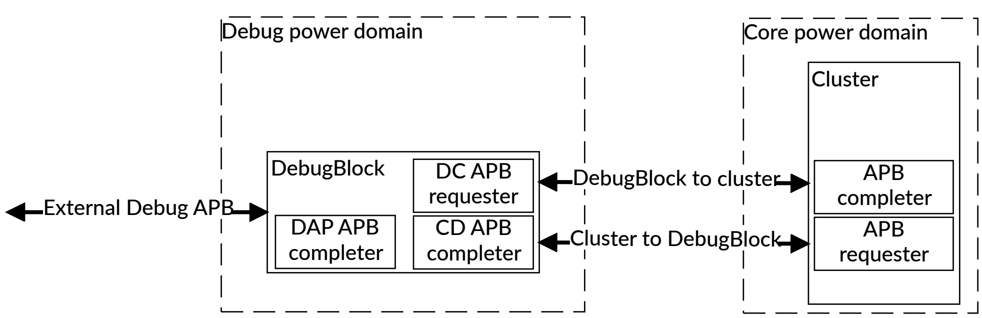
14.5 DebugBlock overview

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 C1-DSU also allows powering down the DebugBlock when debug is not in process.

The following diagram shows how the DebugBlock is connected to the cluster.

Figure 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®v9.0-A Architecture Reference Manual Armv9, for Armv9-A architecture profile*.

Accesses through the DAP APB interface to Debug domain registers are handled internally by the DebugBlock.

Accesses through the DAP APB interface to core power domain registers are passed on to the cluster through the DC APB interface.

CTI trigger events

Trigger events are transferred between the DebugBlock and cluster through the CD APB and DC APB interfaces.

Input trigger events

Input trigger events are sent from the cluster to the CTIs through the CD APB as write transactions.

Output trigger events

Output trigger events are sent from the CTIs to the cluster through the DC APB as write transactions.

DebugBlock power states

The DebugBlock supports two power modes: ON and OFF. These power modes are controlled using the power Q-Channel interface. When the DebugBlock is in the OFF power mode, any uncompleted transactions on the external Debug APB interface to complete with an SLVERR.

Related information

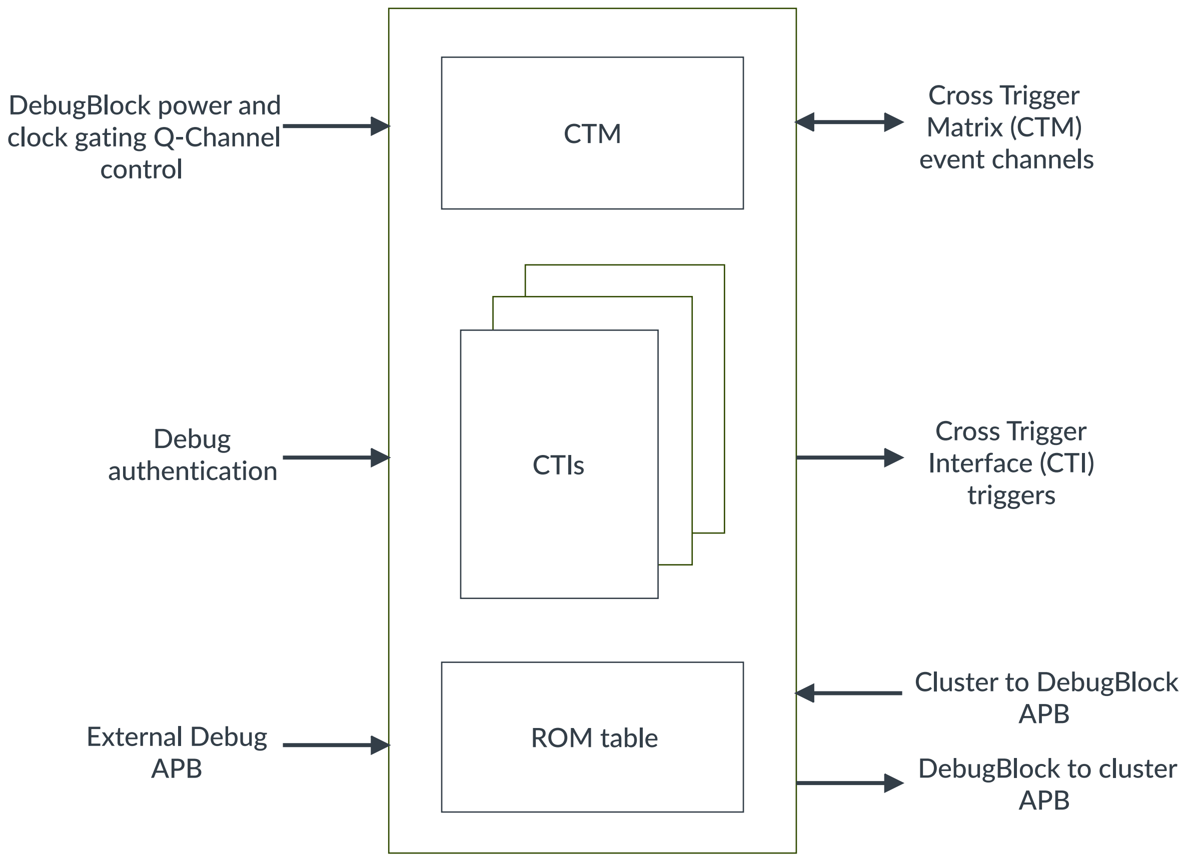
14. Debug on page 206
14.6 DebugBlock subcomponents on page 218
14.7 Embedded Cross Trigger overview on page 220

14.6 DebugBlock subcomponents

The DebugBlock component consists of various subcomponents that facilitate the debugging of the C1-DSU cluster while the cores, complexes, and cluster are powered down.

The following figure shows the DebugBlock.

Figure 14-4: DebugBlock block diagram



Note

The CTIs shown in the diagram include the CTI attached to each core, the CTI attached to each SME2 unit, 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 227 for more information on ROM tables.

Related information

14.7 Embedded Cross Trigger overview on page 220

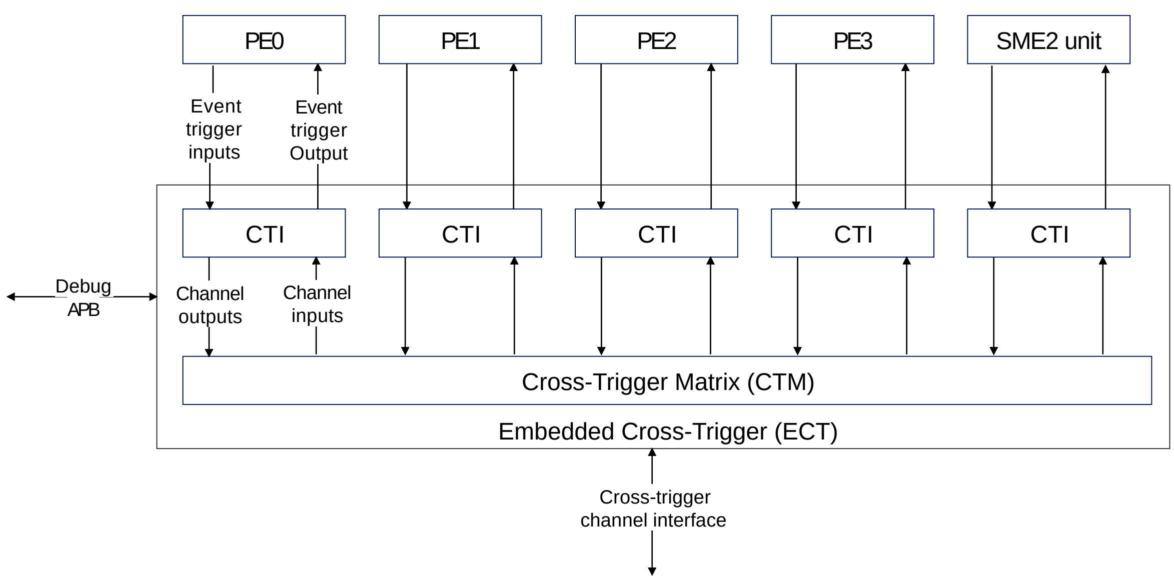
14.7 Embedded Cross Trigger overview

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 and SME2 unit in the cluster. There is also a cluster CTI, which is present in all configurations except Direct connect. The CTIs are interconnected through a Cross Trigger Matrix (CTM) to send debug and trace events between PEs and SME2 units.

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 221
14.6 DebugBlock subcomponents on page 218

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.
1	Restart request	CTI	PE	Pulse	Request the PE to exit Debug state.
2	Generic CTI interrupt	CTI	GIC	Pulse	This trigger event must be sent to the Generic Interrupt Controller (GIC) for the PE.
3	Reserved	-	-	-	Reserved
4-7	ETE trace external input	CTI	ETE	Pulse	This trigger event is sent to the Embedded Trace Extension (ETE) trace in the PE.
8-9	ELA input	CTI	ELA	Pulse	This trigger event is sent to the ELA CTTRIGIN[1:0] attached to the PE.

Allocation of cluster CTI trigger inputs

The following table shows how events are mapped onto the cluster CTI input triggers.

Table 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 Generic Interrupt Controller (GIC).
3-7	Reserved	-	-	-	Reserved
8-9	Cluster ELA input	Cluster CTI	Cluster ELA	Pulse	CTI output trigger events that are mapped onto the trigger events in the cluster ELA CTTRIGIN[1:0].

Related information

14.7 Embedded Cross Trigger overview on page 220

14.8 External cluster, core, and SME2 CTI registers

The cluster Cross Trigger Interface (CTI) registers, core CTI registers, and SME2 unit CTI registers are only accessible using memory-mapped accesses over the Debug APB interface.

The summary table provides an overview of all the cluster, core, and SME2 unit CTI registers. For more information about a register, click on the register name in the table.

Note

- Registers that differ in descriptions and values, between the cluster, core, and SME2 unit, are indicated in the Identical core CTI and SME2 unit CTI column. These registers are the CTIPIDR0-4 registers, and the CTIDEVAFF0-1 registers.
- The CTI registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the C1-DSU is configured for Direct connect all these registers are present.
- If the C1-DSU is enabled for Realm Management Extension (RME) all these registers are present.
- The cluster CTI part number is `0x4F0`.
- The core and SME2 unit CTI part numbers are `0x4ED` and the CTIDEVAFF0-1 registers can be used to identify which core or SME2 unit the CTI is associated with.
- 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	Present in Direct connect	Identical core and SME2 unit CTI
0x000	CTICONTROL	See individual bit resets.	32-bit	CTI Control register	Yes, for core and SME2 unit CTI only	Yes
0x010	CTIINTACK	See individual bit resets.	32-bit	CTI Output Trigger Acknowledge register	Yes, for core and SME2 unit CTI only	Yes
0x014	CTIAPPSET	See individual bit resets.	32-bit	CTI Application Trigger Set register	Yes, for core and SME2 unit CTI only	Yes
0x018	CTIAPPCLEAR	See individual bit resets.	32-bit	CTI Application Trigger Clear register	Yes, for core and SME2 unit CTI only	Yes
0x01C	CTIAPPPULSE	See individual bit resets.	32-bit	CTI Application Pulse register	Yes, for core and SME2 unit CTI only	Yes
0x20	CTIINEN0	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x24	CTIINEN1	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x28	CTIINEN2	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x2C	CTIINEN3	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core and SME2 unit CTI
0x30	CTIINEN4	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x34	CTIINEN5	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x38	CTIINEN6	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x3C	CTIINEN7	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x40	CTIINEN8	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x44	CTIINEN9	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xA0	CTIOUTEN0	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xA4	CTIOUTEN1	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xA8	CTIOUTEN2	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xAC	CTIOUTEN3	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xB0	CTIOUTEN4	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xB4	CTIOUTEN5	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xB8	CTIOUTEN6	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xBC	CTIOUTEN7	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xC0	CTIOUTEN8	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xC4	CTIOUTEN9	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x130	CTITRIGINSTATUS	See individual bit resets.	32-bit	CTI Trigger In Status register	Yes, for core and SME2 unit CTI only	Yes
0x134	CTITRIGOUTSTATUS	See individual bit resets.	32-bit	CTI Trigger Out Status register	Yes, for core and SME2 unit CTI only	Yes
0x138	CTICHINSTATUS	See individual bit resets.	32-bit	CTI Channel In Status register	Yes, for core and SME2 unit CTI only	Yes
0x13C	CTICHOUTSTATUS	See individual bit resets.	32-bit	CTI Channel Out Status register	Yes, for core and SME2 unit CTI only	Yes
0x140	CTIGATE	See individual bit resets.	32-bit	CTI Channel Gate Enable register	Yes, for core and SME2 unit CTI only	Yes
0x150	CTIDEVCTL	See individual bit resets.	32-bit	CTI Device Control register	Yes, for core and SME2 unit CTI only	Yes
0xFA0	CTICLAIMSET	See individual bit resets.	32-bit	CTI CLAIM Tag Set register	Yes, for core and SME2 unit CTI only	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core and SME2 unit CTI
0xFA4	CTICLAIMCLR	See individual bit resets.	32-bit	CTI CLAIM Tag Clear register	Yes, for core and SME2 unit CTI only	Yes
0xFA8	CTIDEVAFF0	See individual bit resets.	32-bit	CTI Device Affinity register 0	Yes, for core and SME2 unit CTI only	No, see individual register
0xFAC	CTIDEVAFF1	See individual bit resets.	32-bit	CTI Device Affinity register 1	Yes, for core and SME2 unit CTI only	No, see individual register
0xFB8	CTIAUTHSTATUS	See individual bit resets.	32-bit	CTI Authentication Status register	Yes, for core and SME2 unit CTI only	Yes
0xFBC	CTIDEVARCH	See individual bit resets.	32-bit	CTI Device Architecture register	Yes, for core and SME2 unit CTI only	Yes
0xFC0	CTIDEVID2	See individual bit resets.	32-bit	CTI Device ID register 2	Yes, for core and SME2 unit CTI only	Yes
0xFC4	CTIDEVID1	See individual bit resets.	32-bit	CTI Device ID register 1	Yes, for core and SME2 unit CTI only	Yes
0xFC8	CTIDEVID	See individual bit resets.	32-bit	CTI Device ID register 0	Yes, for core and SME2 unit CTI only	Yes
0xFCC	CTIDEVTYPE	See individual bit resets.	32-bit	CTI Device Type register	Yes, for core and SME2 unit CTI only	Yes
0xFD0	CTIPIDR4	See individual bit resets.	32-bit	CTI Peripheral Identification Register 4	Yes, for core and SME2 unit CTI only	Yes
0xFE0	CTIPIDR0	See individual bit resets.	32-bit	CTI Peripheral Identification Register 0	Yes, for core and SME2 unit CTI only	No, see individual register
0xFE4	CTIPIDR1	See individual bit resets.	32-bit	CTI Peripheral Identification Register 1	Yes, for core and SME2 unit CTI only	Yes
0xFE8	CTIPIDR2	See individual bit resets.	32-bit	CTI Peripheral Identification Register 2	Yes, for core and SME2 unit CTI only	Yes
0xFEC	CTIPIDR3	See individual bit resets.	32-bit	CTI Peripheral Identification Register 3	Yes, for core and SME2 unit CTI only	Yes
0xFF0	CTICIDR0	See individual bit resets.	32-bit	CTI Component Identification Register 0	Yes, for core and SME2 unit CTI only	Yes
0xFF4	CTICIDR1	See individual bit resets.	32-bit	CTI Component Identification Register 1	Yes, for core and SME2 unit CTI only	Yes
0xFF8	CTICIDR2	See individual bit resets.	32-bit	CTI Component Identification Register 2	Yes, for core and SME2 unit CTI only	Yes
0xFFC	CTICIDR3	See individual bit resets.	32-bit	CTI Component Identification Register 3	Yes, for core and SME2 unit CTI only	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.

Note

Optionally, the cores and cluster can also include instances of the ELA-600, if this IP has been licensed.

The ELA-600 instances are always configured to generate Advanced Trace Bus (ATB) trace. The trace from the Embedded Logic Analyzer (ELA) instances is funneled to the same ATB trace interface as the ETE trace.

14.10 CoreSight component identification

The following table lists the CoreSight ID values for the components present within the C1-DSU.

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	0x04002BB4ED	0xB105900D	0x00000000	0x47700AF7	r0p2
Cluster ROM	0x04002BB4EE	0xB105900D	0x00000000	0x47700AF7	r0p2
Cluster CTI	0x04002BB4EE	0xB105900D	0x00000014	0x47711A14	r0p2
Cluster PMU	0x04002BB4EE	0xB105900D	0x00000016	0x47702A16	r0p2

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 C1-DSU 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, complex or SME2 unit.

All the ROM tables comply with the *Arm® CoreSight™ Architecture Specification v3.0*. The ROM tables for the C1-DSU 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, complex, or SME2 unit in the cluster.

The debug components in the C1-DSU include components for each core in the cluster, for example a Cross Trigger Inteface (CTI) for each core in the cluster.

Note

For a cluster comprised of complexes or cores , the numbering follows the core instance numbering, see 1.7 Core, complex, SME2 unit, and processing element numbering on page 28.

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 C1-DSU.

Table 15-1: Types of components listed in the ROM tables for the C1-DSU

ROM table	ROM table located in	Components
DebugBlock	DebugBlock	• Cluster CTI • CTI for each core or SME2 unit. • Power control and status registers for the cluster • Peripheral and component identification registers • ROM table entry point for the Cluster ROM table
Cluster	DebugBlock	• Cluster Performance Monitoring Unit (PMU) • Cluster Embedded Logic Analyzer (ELA) • ROM table entry points for each standalone core, complex, or SME2 unit. • Power control and status registers for each standalone core, complex, or SME2 unit in the cluster. • Peripheral and component identification registers
Standalone core	Core	See the Technical Reference Manual (TRM) for your core.
Complex	Complex	See the TRM for your core.
SME2 unit	SME2 unit	See the *Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual*.

15.1 Debug system address map

The debug system address map for the C1-DSU 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 <n>, complex <n>, or SME2 unit ROM tables:

Where n is the core instance number. These entries point to the ROM tables for a core, complex or SME2 unit. 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.

The address space for the SME2 unit is treated like another core, sequential to the cores that are present. For example, a cluster comprising of five stand-alone cores and two SME2 units, then:

- SME2 unit instance 0 occupies the address space of what would have been the sixth core.
- SME2 unit instance 1 occupies the address space of what would have been the seventh core.

Cluster ELA

These entries are only present if the Embedded Logic Analyzer (ELA) is included in the cluster.

Core or SME2 unit ELAs

These entries are only present if the ELA is included in the core, complex, or SME2 unit. 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

For a core or SME2 unit, their components are only present, when they are included in the cluster.

Note

The SME2 unit, has no PMU, ETE, or Debug units.

Debug APB system address map

The following table shows the debug system address map for the C1-DSU 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`
Core 0 CTI	`0xF0000`
Complex, SME2 unit, or core 1 ROM Table	`0x100000`
Core 1 Debug	`0x110000`
Core 1 PMU	`0x120000`
Core 1 ETE	`0x130000`
Core 1 or SME2 unit ELA	`0x140000`
Core 1 or SME2 unit CTI	`0x170000`
Complex, SME2 unit, or core 2 ROM Table	`0x180000`
Core 2 Debug	`0x190000`
Core 2 PMU	`0x1A0000`
Core 2 ETE	`0x1B0000`
Core 2 or SME2 unit ELA	`0x1C0000`
Core 2 or SME2 unit CTI	`0x1F0000`
Complex, SME2 unit, or core 3 ROM Table	`0x200000`
Core 3 Debug	`0x210000`
Core 3 PMU	`0x220000`
Core 3 ETE	`0x230000`
Core 3 or SME2 unit ELA	`0x240000`
Core 3 or SME2 unit CTI	`0x270000`
Complex, SME2 unit, or core 4 ROM Table	`0x280000`
Core 4 Debug	`0x290000`
Core 4 PMU	`0x2A0000`
Core 4 ETE	`0x2B0000`
Core 4 or SME2 unit ELA	`0x2C0000`
Core 4 or SME2 unit CTI	`0x2F0000`
Complex, SME2 unit, or core 5 ROM Table	`0x300000`

Debug component (if present)	Debug APB address offset
Core 5 Debug	0x310000
Core 5 PMU	0x320000
Core 5 ETE	0x330000
Core 5 or SME2 unit ELA	0x340000
Core 5 or SME2 unit CTI	0x370000
Complex, SME2 unit, or core 6 ROM Table	0x380000
Core 6 Debug	0x390000
Core 6 PMU	0x3A0000
Core 6 ETE	0x3B0000
Core 6 or SME2 unit ELA	0x3C0000
Core 6 or SME2 unit CTI	0x3F0000
Complex, SME2 unit, or core 7 ROM Table	0x400000
Core 7 Debug	0x410000
Core 7 PMU	0x420000
Core 7 ETE	0x430000
Core 7 or SME2 unit ELA	0x440000
Core 7 or SME2 unit CTI	0x470000
Complex, SME2 unit, or core 8 ROM Table	0x480000
Core 8 Debug	0x490000
Core 8 PMU	0x4A0000
Core 8 ETE	0x4B0000
Core 8 or SME2 unit ELA	0x4C0000
Core 8 or SME2 unit CTI	0x4F0000
Complex, SME2 unit, or core 9 ROM Table	0x500000
Core 9 Debug	0x510000
Core 9 PMU	0x520000
Core 9 ETE	0x530000
Core 9 or SME2 unit ELA	0x540000
Core 9 or SME2 unit CTI	0x570000
Complex, SME2 unit, or core 10 ROM Table	0x580000
Core 10 Debug	0x590000
Core 10 PMU	0x5A0000
Core 10 ETE	0x5B0000
Core 10 or SME2 unit ELA	0x5C0000
Core 10 or SME2 unit CTI	0x5F0000
Complex, SME2 unit, or core 11 ROM Table	0x600000
Core 11 Debug	0x610000
Core 11 PMU	0x620000
Core 11 ETE	0x630000
Core 11 or SME2 unit ELA	0x640000

Debug component (if present)	Debug APB address offset
Core 11 or SME2 unit CTI	`0x670000`
Complex, SME2 unit, or core 12 ROM Table	`0x680000`
Core 12 Debug	`0x690000`
Core 12 PMU	`0x6A0000`
Core 12 ETE	`0x6B0000`
Core 12 or SME2 unit ELA	`0x6C0000`
Core 12 or SME2 unit CTI	`0x6F0000`
Complex, SME2 unit, or core 13 ROM Table	`0x700000`
Core 13 Debug	`0x710000`
Core 13 PMU	`0x720000`
Core 13 ETE	`0x730000`
Core 13 or SME2 unit ELA	`0x740000`
Core 13 or SME2 unit CTI	`0x770000`
SME2 unit ROM Table	`0x780000`
SME2 unit ELA	`0x7C0000`
SME2 unit CTI	`0x7F0000`
SME2 unit ROM Table	`0x800000`
SME2 unit ELA	`0x840000`
SME2 unit CTI	`0x870000`

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 or SME2 unit.
- 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 234.

The ROMENTRY entry values depend on the number and type of cores implemented. The register formats are described in the *Arm® CoreSight™ Architecture Specification v3.0*.

Note

The DebugBlock ROM table part number is `0x4ED`.

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 or SME2 unit
0x0010	ROMENTRY4	CTI for core 2 or SME2 unit
0x0014	ROMENTRY5	CTI for core 3 or SME2 unit
0x0018	ROMENTRY6	CTI for core 4 or SME2 unit
0x001C	ROMENTRY7	CTI for core 5 or SME2 unit
0x0020	ROMENTRY8	CTI for core 6 or SME2 unit
0x0024	ROMENTRY9	CTI for core 7 or SME2 unit
0x0028	ROMENTRY10	CTI for core 8 or SME2 unit
0x002C	ROMENTRY11	CTI for core 9 or SME2 unit
0x0030	ROMENTRY12	CTI for core 10 or SME2 unit
0x0034	ROMENTRY13	CTI for core 11 or SME2 unit
0x0038	ROMENTRY14	CTI for core 12 or SME2 unit
0x003C	ROMENTRY15	CTI for core 13 or SME2 unit
0x0040	ROMENTRY16	CTI for SME2 unit
0x0044	ROMENTRY17	CTI for SME2 unit
0x0048-0x09FC	-	Reserved
0x0A00	DBGPCR0	Debug Power Control Register
0x0A04-0x0A7C	-	Reserved
0x0A80	DBGPSR0	Debug Power Status Register
0x0A84-0x0AFC	-	Reserved
0x0B00	SYSPCR0	System Power Control Register
0x0B04-0x0B7C	-	Reserved
0x0B80	SYSPSR0	System Power Status Register
0x0B84-0x0BFC	-	Reserved
0x0C00	PRIDR0	Power Reset Identification Register
0x0C04-0x0FB4	-	Reserved
0x0FB8	AUTHSTATUS	Authentication Status Register
0x0FBC	DEVARCH	Device Architecture Register
0x0FC0-0x0FC4	-	Reserved
0x0FC8	DEVID	Device ID Register
0x0FCC	DEVTYPE	Device Type Register
0x0FD0	PIDR4	Peripheral Identification Register 4
0x0FD4-0FDC	-	Reserved
0x0FE0	PIDR0	Peripheral Identification Register 0
0x0FE4	PIDR1	Peripheral Identification Register 1
0x0FE8	PIDR2	Peripheral Identification Register 2

Offset	Name	Description
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, complex, or SME2 unit.
- Power control registers for each standalone core, complex, or SME2 unit. These power control registers allow powerup requests to be made for each standalone core, complex, or SME2 unit, see 15.4 ROM table power request registers for cluster and cores on page 234.

The ROMENTRY entry values depend on, the number and type of cores implemented, and the number of SME2 units 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.
- The cluster ROM table part number is 0x4EE.
- If the cluster is configured with any SME2 units, these are listed sequentially after the last core or complex.
  For example, a cluster comprising of five stand-alone cores and two SME2 units, then:
  - SME2 unit instance 0 occupies the address space of what would have been the sixth core.
  - SME2 unit instance 1 occupies the address space of what would have been the seventh core.

Table 15-4: ROM table registers

Offset	Name	Description
0x0000	ROMENTRY0	Cluster PMU

Offset	Name	Description
0x0004	ROMENTRY1	Cluster ELA
0x0008	ROMENTRY2	Core 0 ROM table
0x000C	ROMENTRY3	Core 1 or SME2 unit ROM table
0x0010	ROMENTRY4	Core 2 or SME2 unit ROM table
0x0014	ROMENTRY5	Core 3 or SME2 unit ROM table
0x0018	ROMENTRY6	Core 4 or SME2 unit ROM table
0x001C	ROMENTRY7	Core 5 or SME2 unit ROM table
0x0020	ROMENTRY8	Core 6 or SME2 unit ROM table
0x0024	ROMENTRY9	Core 7 or SME2 unit ROM table
0x0028	ROMENTRY10	Core 8 or SME2 unit ROM table
0x002C	ROMENTRY11	Core 9 or SME2 unit ROM table
0x0030	ROMENTRY12	Core 10 or SME2 unit ROM table
0x0034	ROMENTRY13	Core 11 or SME2 unit ROM table
0x0038	ROMENTRY14	Core 12 or SME2 unit ROM table
0x003C	ROMENTRY15	Core 13 or SME2 unit ROM table
0x0040	ROMENTRY16	SME2 unit ROM table
0x0044	ROMENTRY17	SME2 unit ROM table
0x0048-0x09FC	-	Reserved
0x0A00-0x0A34	DBGPCR<n>	Debug Power Control Register for core <n> or SME2 unit
0x0A38-0x0A7C	-	Reserved
0x0A80-0x0AB4	DBGPSR<n>	Debug Power Status Register for core <n> or SME2 unit
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
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, complexes, or SME2 units 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, if your cluster contains SME2 units, then there are DBGPCR registers present for each SME2 unit, which can be used by the debugger to request a powerup for the corresponding SME2 unit. These registers are located sequentially after the last core/complex DBGPCR register. For example, for a cluster that contains two standalone cores and two SME2 units, then:

- Register DBGPCR2 at address `0xA02` can be used to make a powerup request for SME2 unit instance 0.
- Register DBGPCR3 at address `0xA03` can be used to make a powerup request for SME2 unit instance 1.

Your debugger can use the power control register, DBGPCR0, located in the DebugBlock ROM table, to make a request to powerup the C1-DSU cluster.

The corresponding core, SME2 unit, 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**.
- If the C1-DSU is configured for Direct connect, all these registers are present.
- If the C1-DSU is enabled for Realm Management Extension (RME) all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.
- The cluster ROM table registers part number is `0x4F0`.

Table 15-5: CLUSTERROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	CLUSTERROM_ROMENTRY0	See individual bit resets.	32-bit	Cluster ROM table entry 0	Yes
0x004	CLUSTERROM_ROMENTRY1	See individual bit resets.	32-bit	Cluster ROM table entry 1	Yes
0x8	CLUSTERROM_ROMENTRY2	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0xC	CLUSTERROM_ROMENTRY3	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x10	CLUSTERROM_ROMENTRY4	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x14	CLUSTERROM_ROMENTRY5	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x18	CLUSTERROM_ROMENTRY6	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x1C	CLUSTERROM_ROMENTRY7	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x20	CLUSTERROM_ROMENTRY8	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x24	CLUSTERROM_ROMENTRY9	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x28	CLUSTERROM_ROMENTRY10	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x2C	CLUSTERROM_ROMENTRY11	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x30	CLUSTERROM_ROMENTRY12	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x34	CLUSTERROM_ROMENTRY13	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x38	CLUSTERROM_ROMENTRY14	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x3C	CLUSTERROM_ROMENTRY15	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x40	CLUSTERROM_ROMENTRY16	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x44	CLUSTERROM_ROMENTRY17	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0xA00	CLUSTERROM_DBGPCR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA04	CLUSTERROM_DBGPCR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA08	CLUSTERROM_DBGPCR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA0C	CLUSTERROM_DBGPCR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xA10	CLUSTERROM_DBGPCR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA14	CLUSTERROM_DBGPCR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA18	CLUSTERROM_DBGPCR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA1C	CLUSTERROM_DBGPCR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA20	CLUSTERROM_DBGPCR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA24	CLUSTERROM_DBGPCR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA28	CLUSTERROM_DBGPCR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA2C	CLUSTERROM_DBGPCR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA30	CLUSTERROM_DBGPCR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA34	CLUSTERROM_DBGPCR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA38	CLUSTERROM_DBGPCR14	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA3C	CLUSTERROM_DBGPCR15	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA80	CLUSTERROM_DBGPSR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA84	CLUSTERROM_DBGPSR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA88	CLUSTERROM_DBGPSR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA8C	CLUSTERROM_DBGPSR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA90	CLUSTERROM_DBGPSR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA94	CLUSTERROM_DBGPSR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA98	CLUSTERROM_DBGPSR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA9C	CLUSTERROM_DBGPSR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAA0	CLUSTERROM_DBGPSR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAA4	CLUSTERROM_DBGPSR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAA8	CLUSTERROM_DBGPSR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xAAC	CLUSTERROM_DBGPSR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAB0	CLUSTERROM_DBGPSR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAB4	CLUSTERROM_DBGPSR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAB8	CLUSTERROM_DBGPSR14	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xABC	CLUSTERROM_DBGPSR15	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xC00	CLUSTERROM_PRIDR0	See individual bit resets.	32-bit	Cluster ROM table Power Request ID Register 0	Yes
0xFB8	CLUSTERROM_AUTHSTATUS	See individual bit resets.	32-bit	Cluster ROM table Authentication Status Register	Yes
0xFBC	CLUSTERROM_DEVARCH	See individual bit resets.	32-bit	Cluster ROM table Device Architecture Register	Yes
0xFC8	CLUSTERROM_DEVID	See individual bit resets.	32-bit	Cluster ROM table Device Configuration Register	Yes
0xFCC	CLUSTERROM_DEVTYPE	See individual bit resets.	32-bit	Cluster ROM table Device Type Register	Yes
0xFD0	CLUSTERROM_PIDR4	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 4	Yes
0xFE0	CLUSTERROM_PIDR0	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 0	Yes
0xFE4	CLUSTERROM_PIDR1	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 1	Yes
0xFE8	CLUSTERROM_PIDR2	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 2	Yes
0xFEC	CLUSTERROM_PIDR3	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 3	Yes
0xFF0	CLUSTERROM_CIDR0	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 0	Yes
0xFF4	CLUSTERROM_CIDR1	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 1	Yes
0xFF8	CLUSTERROM_CIDR2	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 2	Yes
0xFFC	CLUSTERROM_CIDR3	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 3	Yes

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, implemented with the following C1-DSU implementation parameters:
  - DIRECT_CONNECT is set to FALSE.
  - NUM_CORES 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**.
- If the C1-DSU is configured for Direct connect, all these registers are present.
- If the C1-DSU is enabled for Realm Management Extension (RME) all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 15-6: DBROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	DBROM_ROMENTRY0	See individual bit resets.	32-bit	DebugBlock ROM table Entry 0	Yes
0x4	DBROM_ROMENTRY1	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x8	DBROM_ROMENTRY2	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0xC	DBROM_ROMENTRY3	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x10	DBROM_ROMENTRY4	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x14	DBROM_ROMENTRY5	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x18	DBROM_ROMENTRY6	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x1C	DBROM_ROMENTRY7	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x20	DBROM_ROMENTRY8	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x24	DBROM_ROMENTRY9	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x28	DBROM_ROMENTRY10	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x2C	DBROM_ROMENTRY11	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x30	DBROM_ROMENTRY12	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x34	DBROM_ROMENTRY13	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x38	DBROM_ROMENTRY14	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x3C	DBROM_ROMENTRY15	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x40	DBROM_ROMENTRY16	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x44	DBROM_ROMENTRY17	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0xA00	DBROM_DBGPCR0	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Control Register 0	Yes
0xA80	DBROM_DBGPSR0	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Status Register 0	Yes
0xC00	DBROM_PRIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Power Request ID Register 0	Yes
0xFB8	DBROM_AUTHSTATUS	See individual bit resets.	32-bit	DebugBlock ROM table Authentication Status Register	Yes
0xFBC	DBROM_DEVARCH	See individual bit resets.	32-bit	DebugBlock ROM table Device Architecture Register	Yes
0xFC8	DBROM_DEVID	See individual bit resets.	32-bit	DebugBlock ROM table Device Configuration Register	Yes
0xFCC	DBROM_DEVTYPE	See individual bit resets.	32-bit	DebugBlock ROM table Device Type Register	Yes
0xFD0	DBROM_PIDR4	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 4	Yes
0xFE0	DBROM_PIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 0	Yes
0xFE4	DBROM_PIDR1	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 1	Yes
0xFE8	DBROM_PIDR2	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 2	Yes
0xFEC	DBROM_PIDR3	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 3	Yes
0xFF0	DBROM_CIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 0	Yes
0xFF4	DBROM_CIDR1	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 1	Yes
0xFF8	DBROM_CIDR2	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 2	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFFC	DBROM_CIDR3	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 3	Yes

16. Performance Monitors Extension support

The C1-DSU 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.

Note

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 C1-DSU supports access to the performance monitor registers from the internal System register interface. The C1-DSU 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 Snoop Control Unit (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 (Error Correcting Code (ECC) or parity) in the protected RAMs.
`0x001D`	BUS_CYCLES	AXI or CHI bus cycle counter.
`0x0029`	L3D_CACHE_ALLOCATE	Level 3 unified cache allocation without refill counter. Counts every full cache line write into the L3 cache which does not cause a linefill.
`0x002A`	L3D_CACHE_REFILL	Level 3 unified cache refill counter Counts every Cacheable read transaction issued to the interconnect. This event counts the sum of L3D_CACHE_REFILL_RD and L3D_CACHE_REFILL_WR.
`0x002B`	L3D_CACHE	Level 3 unified cache access counter Counts every Cacheable read or write transaction issued to the Snoop Control Unit (SCU). This event counts the sum of L3D_CACHE_RD and L3D_CACHE_WR.
`0x002C`	L3D_CACHE_WB	Level 3 unified cache write-back counter Counts every write-back from the L3 cache.
`0x0060`	BUS_ACCESS_RD	Bus access, read counter Counts every beat of data transferred over the read data channel between the SCU and the interconnect. **Note:** If the cluster generates a CHI MakeReadUnique transaction for a shared line upgrade, it is unknown at the time of counting if this results in a data transfer or not. Therefore, the counter assumes the data will not be transferred.

PMU event number	Event mnemonic	Event description
0x0061	BUS_ACCESS_WR	Bus access, write counter Counts every beat of data transferred over the write data channel between the SCU and the interconnect. **Note:** If the cluster generates a CHI WriteEvictOrEvict transaction for a clean eviction, it is unknown at the time of counting if this results in a data transfer or not. Therefore, the counter assumes the data will be transferred.
0x0062	BUS_ACCESS_SHARED	Bus access, shared counter Counts every beat of shared data transferred over the data channels between the SCU and the interconnect.
0x0063	BUS_ACCESS_NOT_SHARED	Bus access, not shared counter Counts every beat of not shared data transferred over the write data channel between the SCU and the interconnect.
0x0064	BUS_ACCESS_NORMAL	Bus access, normal counter Counts every beat of normal data transferred over the write data channel between the SCU and the interconnect.
0x0065	BUS_ACCESS_PERIPH	Bus access, periph counter Counts every beat of Device data transferred over the write data channel between the SCU and the interconnect.
0x00A0	L3D_CACHE_RD	Level 3 unified cache access, read counter Counts every Cacheable shareable read transaction that is issued to the SCU. Prefetches and stashes are not counted.
0x00A1	L3D_CACHE_WR	Level 3 unified cache access, write counter Counts every Cacheable write transaction issued to the SCU.
0x00A2	L3D_CACHE_REFILL_RD	Level 3 unified cache refill, read counter Counts every Cacheable read transaction issued to the interconnect caused by a Cacheable shareable read transaction. Prefetches and stashes are not counted.
0x00A3	L3D_CACHE_REFILL_WR	Level 3 unified cache refill, write counter Counts every Cacheable read transaction issued to the interconnect caused by a write transaction.
0x0119	ACP_ACCESS	Accelerator Coherency Port (ACP) access counter 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. This event counts the sum of ACP_ACCESS_RD and ACP_ACCESS_WR.
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: • The stash hit in the cluster or; • The stash is not performed due to the L3 cache being off.
`0x0510`	SCU_STASH_ICN_ACCESS	Stash access, ICN counter Counts every stash transaction originating from the interconnect.

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: • The stash hit in the cluster or; • The stash was not performed due to L3 cache being off.
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_TAG_SF_ECC	TAG/SF ECC errors counter ECC errors detected on a tag RAM or snoop filter RAM accesses that cause a way to be avoided but are not corrected or reported in the Reliability, Availability, and Serviceability (RAS) registers.

16.3 PMU interrupt

The C1-DSU 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**.
- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- If the C1-DSU is enabled for Realm Management Extension (RME), none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- The part number is `0x4EE`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table 16-2: performance-monitors registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0	CLUSTERPMU_PMEVCNTR0	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x8	CLUSTERPMU_PMEVCNTR1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x10	CLUSTERPMU_PMEVCNTR2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x18	CLUSTERPMU_PMEVCNTR3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x20	CLUSTERPMU_PMEVCNTR4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x28	CLUSTERPMU_PMEVCNTR5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x400	CLUSTERPMU_PMEVTYPER0	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x404	CLUSTERPMU_PMEVTYPER1	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x408	CLUSTERPMU_PMEVTYPER2	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x40C	CLUSTERPMU_PMEVTYPER3	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x410	CLUSTERPMU_PMEVTYPER4	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x414	CLUSTERPMU_PMEVTYPER5	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x600	CLUSTERPMU_PMEVCNTSR0	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x608	CLUSTERPMU_PMEVCNTSR1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x610	CLUSTERPMU_PMEVCNTSR2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x618	CLUSTERPMU_PMEVCNTSR3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x620	CLUSTERPMU_PMEVCNTSR4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x628	CLUSTERPMU_PMEVCNTSR5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x638	CLUSTERPMU_PMSSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Status register	No
0x640	CLUSTERPMU_PMOVSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Status Snapshot register	No
0xC00	CLUSTERPMU_PMCNTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Set register	No
0xC20	CLUSTERPMU_PMCNTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Clear register	No
0xC40	CLUSTERPMU_PMINTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Set register	No
0xC60	CLUSTERPMU_PMINTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Clear register	No
0xC80	CLUSTERPMU_PMOVSCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Clear register	No
0xCC0	CLUSTERPMU_PMOVSSET	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Set register	No
0xE00	CLUSTERPMU_PMCFGR	See individual bit resets.	32-bit	Cluster Performance Monitors Configuration Register	No
0xE04	CLUSTERPMU_PMCR	See individual bit resets.	32-bit	Cluster Performance Monitors Control Register	No
0xE08	CLUSTERPMU_PMIIDR	See individual bit resets.	32-bit	Cluster Performance Monitors Implementation Identification register	No
0xE20	CLUSTERPMU_PMCEID0	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 0	No
0xE24	CLUSTERPMU_PMCEID1	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 1	No
0xE28	CLUSTERPMU_PMCEID2	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 2	No
0xE2C	CLUSTERPMU_PMCEID3	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 3	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xE30	CLUSTERPMU_PMSSCR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Capture register	No
0xE38	CLUSTERPMU_PMSSRR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Reset register	No
0xFA8	CLUSTERPMU_PMDEVAFF0	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 0	No
0xFAC	CLUSTERPMU_PMDEVAFF1	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 1	No
0xFB8	CLUSTERPMU_PMAUTHSTATUS	See individual bit resets.	32-bit	Cluster Performance Monitors Authentication Status register	No
0xFBC	CLUSTERPMU_PMDEVARCH	See individual bit resets.	32-bit	Cluster Performance Monitors Device Architecture register	No
0xFC8	CLUSTERPMU_PMDEVID	See individual bit resets.	32-bit	Cluster Performance Monitors Device ID register	No
0xFCC	CLUSTERPMU_PMDEVTYPE	See individual bit resets.	32-bit	Cluster Performance Monitors Device Type register	No
0xFD0	CLUSTERPMU_PMPIDR4	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 4	No
0xFE0	CLUSTERPMU_PMPIDR0	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 0	No
0xFE4	CLUSTERPMU_PMPIDR1	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 1	No
0xFE8	CLUSTERPMU_PMPIDR2	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 2	No
0xFEC	CLUSTERPMU_PMPIDR3	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 3	No
0xFF0	CLUSTERPMU_PMCIDR0	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 0	No
0xFF4	CLUSTERPMU_PMCIDR1	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 1	No
0xFF8	CLUSTERPMU_PMCIDR2	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 2	No
0xFFC	CLUSTERPMU_PMCIDR3	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 3	No

17. Activity Monitors Extension support

The C1-DSU 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 C1-DSU 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 C1-DSU 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`. If the cluster has Realm Management Extension (RME) enabled, these registers are accessed from Root state, otherwise these 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 C1-DSU 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 253.
- 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 C1-DSU. 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.
- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers is treated as **RAZ/WI**.
- Any address that is not documented is treated as **RAZ/WI**.
- The part number is `0x04EE`.
- 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	Present in Direct connect
0x0	CLUSTERAMU_AMEVCNTR0 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x4	CLUSTERAMU_AMEVCNTR0 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x8	CLUSTERAMU_AMEVCNTR1 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0xC	CLUSTERAMU_AMEVCNTR1 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x10	CLUSTERAMU_AMEVCNTR2 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x14	CLUSTERAMU_AMEVCNTR2 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x18	CLUSTERAMU_AMEVCNTR3 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x1C	CLUSTERAMU_AMEVCNTR3 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x20	CLUSTERAMU_AMEVCNTR4 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x24	CLUSTERAMU_AMEVCNTR4 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x400	CLUSTERAMU_AMEVTYPER0	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x404	CLUSTERAMU_AMEVTYPER1	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x408	CLUSTERAMU_AMEVTYPER2	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x40C	CLUSTERAMU_AMEVTYPER3	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x410	CLUSTERAMU_AMEVTYPER4	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0xC00	CLUSTERAMU_AMCNTENSET	See individual bit resets.	32-bit	Activity Monitors Count Enable Set Register 0	No
0xC20	CLUSTERAMU_AMCNTENCLR	See individual bit resets.	32-bit	Activity Monitors Count Enable Clear Register 0	No
0xE00	CLUSTERAMU_AMCFGR	See individual bit resets.	32-bit	Activity Monitors Configuration Register	No
0xE04	CLUSTERAMU_AMCR	See individual bit resets.	32-bit	Activity Monitors Control Register	No
0xE08	CLUSTERAMU_AMIIDR	See individual bit resets.	32-bit	Activity Monitors Implementation Identification Register	No
0xFA8	CLUSTERAMU_AMDEVAFF	See individual bit resets.	64-bit	Activity Monitors Device Affinity Register 0	No
0xFBC	CLUSTERAMU_AMDEVARCH	See individual bit resets.	32-bit	Activity Monitors Device Architecture Register	No
0xFC8	CLUSTERAMU_AMDEVID	See individual bit resets.	32-bit	Cluster Activity Monitors Device ID register	No
0xFCC	CLUSTERAMU_AMDEVTYPE	See individual bit resets.	32-bit	Activity Monitors Device Type Register	No
0xFD0	CLUSTERAMU_AMPIDR4	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 4	No
0xFE0	CLUSTERAMU_AMPIDR0	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 0	No
0xFE4	CLUSTERAMU_AMPIDR1	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 1	No
0xFE8	CLUSTERAMU_AMPIDR2	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 2	No
0xFEC	CLUSTERAMU_AMPIDR3	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 3	No
0xFF0	CLUSTERAMU_AMCIDR0	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 0	No
0xFF4	CLUSTERAMU_AMCIDR1	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 1	No
0xFF8	CLUSTERAMU_AMCIDR2	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 2	No
0xFFC	CLUSTERAMU_AMCIDR3	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 3	No

17.4 Activity monitors events

Activity monitors events in the C1-DSU 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 **Note:** 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 **Note:** 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.
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.

Note

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 C1-DSU.

A.1 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- If the C1-DSU is enabled for Realm Management Extension (RME), none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table A-1: Performance Monitors registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERPMCR_EL1	3	0	C15	C5	0	See individual bit resets.	64-bit	Performance Monitors Control Register	No
IMP_CLUSTERPMCNTENSET_EL1	3	0	C15	C5	1	See individual bit resets.	64-bit	Performance Monitors Count Enable Set Register	No
IMP_CLUSTERPMCNTENCLR_EL1	3	0	C15	C5	2	See individual bit resets.	64-bit	Performance Monitors Count Enable Clear Register	No
IMP_CLUSTERPMOVSSET_EL1	3	0	C15	C5	3	See individual bit resets.	64-bit	Performance Monitors Overflow Flag Status Set Register	No
IMP_CLUSTERPMOVSCLR_EL1	3	0	C15	C5	4	See individual bit resets.	64-bit	Performance Monitors Overflow Flag Status Clear Register	No
IMP_CLUSTERPMSELR_EL1	3	0	C15	C5	5	See individual bit resets.	64-bit	Performance Monitors Event Counter Selection Register	No
IMP_CLUSTERPMINTENSET_EL1	3	0	C15	C5	6	See individual bit resets.	64-bit	Performance Monitors Interrupt Enable Set Register	No

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERPMINTENCLR_EL1	3	0	C15	C5	7	See individual bit resets.	64-bit	Performance Monitors Interrupt Enable Clear Register	No
IMP_CLUSTERRSVD_6_0_EL1	3	0	C15	C6	0	See individual bit resets.	64-bit	Reserved	No
IMP_CLUSTERPMXEVTYPER_EL1	3	0	C15	C6	1	See individual bit resets.	64-bit	Performance Monitors Selected Event Type Register	No
IMP_CLUSTERPMXEVCNTR_EL1	3	0	C15	C6	2	See individual bit resets.	64-bit	Performance Monitors Selected Event Count Register	No
IMP_CLUSTERPMCEID0_EL1	3	0	C15	C6	4	See individual bit resets.	64-bit	Performance Monitors Common Event Identification Register 0	No
IMP_CLUSTERPMCEID1_EL1	3	0	C15	C6	5	See individual bit resets.	64-bit	Performance Monitors Common Event Identification Register 1	No

A.1.1 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 xx0x xxxx xxx0
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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_clusterpmcr_el1 bit assignments

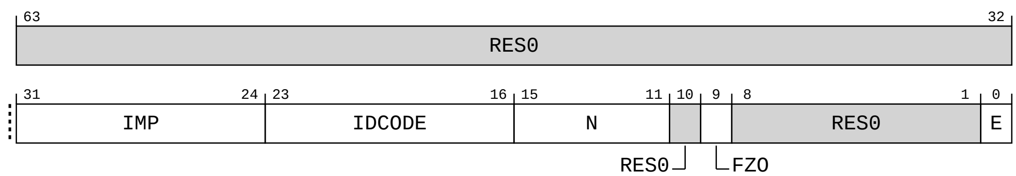


Table A-2: IMP_CLUSTERPMCR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31:24]	IMP	Implementer code. This field is RO with a value of 0 indicating that IMP_CLUSTERPMCR_EL1.IDCODE is **RES0** and software must use the AArch64-MIDR_EL1 to identify the PE. **`0b00000000`** Software must use the AArch64-MIDR_EL1 to identify the PE.	`8{x}`
[23:16]	IDCODE	Identification code. This field is RO with a value of `0x00`. **`0b00000000`** Software must use the AArch64-MIDR_EL1 to identify the PE.	`8{x}`
[15:11]	N	A Read Only field that indicates the number of event counters implemented. **`0b00110`** Indicates that 6 event counters are implemented. Access to this field is: RO	`5{x}`
[10]	**RES0**	Reserved	**RES0**
[9]	FZO	Freeze on overflow. **`0b0`** Freeze on overflow disabled. **`0b1`** Freeze on overflow enabled.	`0b0`
[8:1]	**RES0**	Reserved	**RES0**
[0]	E	Enable. **`0b0`** All event counters in the range [0..(PMN-1)] are disabled. **`0b1`** 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
        X[t, 64] = IMP_CLUSTERPMCR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMCR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMCR_EL1;
```

MSR S3_0_C15_C5_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCR_EL1 = X[t, 64];
```

A.1.2 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     3  0
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-2: AArch64_imp_clusterpmcntenset_el1 bit assignments

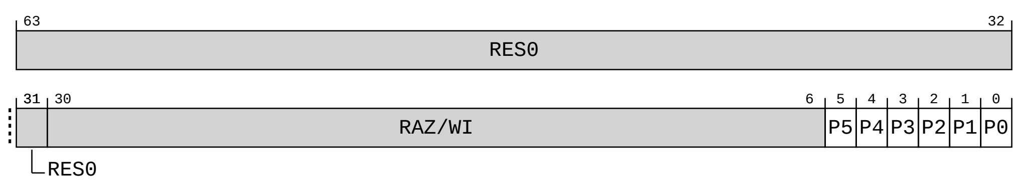


Table A-5: IMP_CLUSTERPMCNTENSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[4]	P4	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[3]	P3	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[2]	P2	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[1]	P1	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[0]	P0	Event counter enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter is enabled. When written, enables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x

Access

MRS <Xt>, S3_0_C15_C5_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b001

MSR S3_0_C15_C5_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b001

Accessibility

MRS <Xt>, S3_0_C15_C5_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMCNTENSET_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMCNTENSET_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMCNTENSET_EL1;
```

MSR S3_0_C15_C5_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENSET_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENSET_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCNTENSET_EL1 = X[t, 64];
```

A.1.3 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     3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-3: AArch64_imp_clusterpmcntenclr_el1 bit assignments

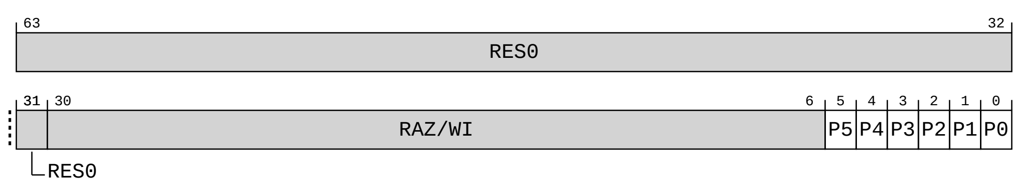


Table A-8: IMP_CLUSTERPMCNTENCLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[4]	P4	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[3]	P3	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[2]	P2	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[1]	P1	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x
[0]	P0	Event counter disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is disabled. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is enabled. When written, disables AArch64-IMP_CLUSTERPMXEVCNTR_EL1.	x

Access

MRS <Xt>, S3_0_C15_C5_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b010

MSR S3_0_C15_C5_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b010

Accessibility

MRS <Xt>, S3_0_C15_C5_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMCNTENCLR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMCNTENCLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMCNTENCLR_EL1;
```

MSR S3_0_C15_C5_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCNTENCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCNTENCLR_EL1 = X[t, 64];
```

A.1.4 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-4: AArch64_imp_clusterpmovsset_el1 bit assignments

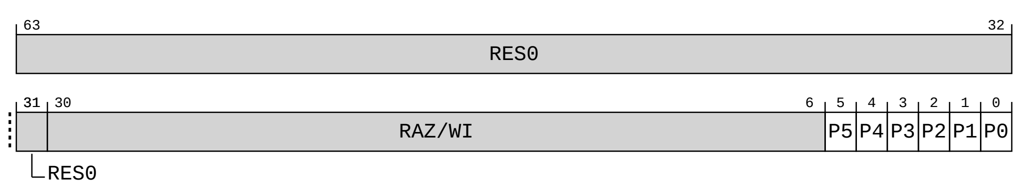


Table A-11: IMP_CLUSTERPMOVSSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[4]	P4	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[3]	P3	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[2]	P2	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[1]	P1	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x
[0]	P0	Event counter overflow set bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, sets the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 1.	x

Access

MRS <Xt>, S3_0_C15_C5_3

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b011

MSR S3_0_C15_C5_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b011

Accessibility

MRS <Xt>, S3_0_C15_C5_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMOVSSET_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMOVSSET_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMOVSSET_EL1;
```

MSR S3_0_C15_C5_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSSET_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSSET_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMOVSSET_EL1 = X[t, 64];
```

A.1.5 IMP_CLUSTERPMOVSCLR_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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-5: AArch64_imp_clusterpmovsclr_el1 bit assignments

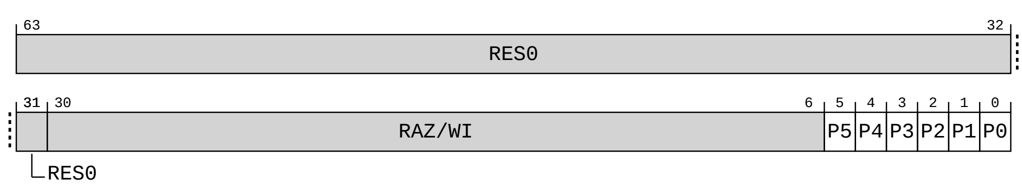


Table A-14: IMP_CLUSTERPMOVSCLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[4]	P4	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[3]	P3	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[2]	P2	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x
[1]	P1	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **0b0** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	x

Bits	Name	Description	Reset
[0]	P0	Event counter overflow clear bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. If N is less than 31, then bits [30:N] are **RAZ/WI**. N is the value in AArch64-IMP_CLUSTERPMCR_EL1.N. **`0b0`** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has not overflowed since this bit was last cleared. When written, has no effect. **`0b1`** When read, means that AArch64-IMP_CLUSTERPMXEVCNTR_EL1 has overflowed since this bit was last cleared. When written, clears the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 overflow bit to 0.	`x`

Access

MRS <Xt>, S3_0_C15_C5_4

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b100

MSR S3_0_C15_C5_4, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b100

Accessibility

MRS <Xt>, S3_0_C15_C5_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMOVSCLR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMOVSCLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMOVSCLR_EL1;
```

MSR S3_0_C15_C5_4, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
```

```
        IMP_CLUSTERPMOVSCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMOVSCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMOVSCLR_EL1 = X[t, 64];
```

A.1.6 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_CLUSTERPMXEVTYPER_EL1 to determine the event that increments a selected event counter, and the modes and states in which the selected counter increments.

It is also used in conjunction with AArch64-IMP_CLUSTERPMXEVCNTR_EL1, to determine the value of a selected event counter.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-6: AArch64_imp_clusterpmselr_el1 bit assignments

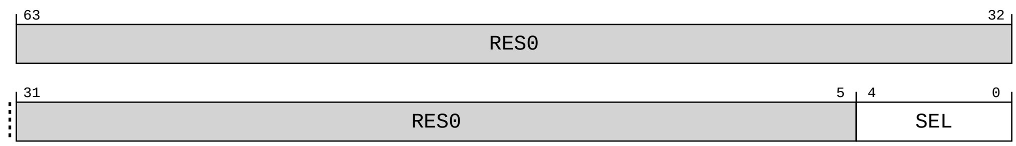


Table A-17: IMP_CLUSTERPMSELR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:5]	**RES0**	Reserved	**RES0**
[4:0]	SEL	Selects event counter, AArch64-IMP_CLUSTERPMXEVCNTR_EL1, where n is the value held in this field. This value identifies which event counter is accessed when a subsequent access to AArch64-IMP_CLUSTERPMXEVTYPER_EL1 or AArch64-IMP_CLUSTERPMXEVCNTR_EL1 occurs. This field can take any value from 0 (`0b00000`) to (PMCR.N)-1. • A read or write of AArch64-IMP_CLUSTERPMXEVTYPER_EL1 is treated as **RAZ/WI**. • A read or write of AArch64-IMP_CLUSTERPMXEVCNTR_EL1 is treated as **RAZ/WI**. If this field is set to a value greater than or equal to the number of counters accessible at the current Exception level, but not equal to 31: • Direct reads of this field are treated as **RAZ/WI**. • The results of access to AArch64-IMP_CLUSTERPMXEVTYPER_EL1 or AArch64-IMP_CLUSTERPMXEVCNTR_EL1 are treated as **RAZ/WI**.	`5{x}`

Access

MRS <Xt>, S3_0_C15_C5_5

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0101`	`0b101`

MSR S3_0_C15_C5_5, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0101`	`0b101`

Accessibility

MRS <Xt>, S3_0_C15_C5_5

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMSELR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMSELR_EL1;
```

```
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMSELR_EL1;
```

MSR S3_0_C15_C5_5, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMSELR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMSELR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMSELR_EL1 = X[t, 64];
```

A.1.7 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

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-7: AArch64_imp_clusterpmintenset_el1 bit assignments

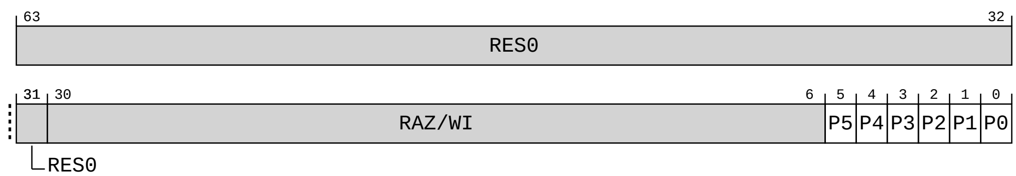


Table A-20: IMP_CLUSTERPMINTENSET_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	**RES0**	Reserved	**RES0**
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[4]	P4	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[3]	P3	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x

Bits	Name	Description	Reset
[2]	P2	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[1]	P1	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x
[0]	P0	Event counter overflow interrupt request enable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **0b0** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, enables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	x

Access

MRS <Xt>, S3_0_C15_C5_6

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b110

MSR S3_0_C15_C5_6, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0101	0b110

Accessibility

MRS <Xt>, S3_0_C15_C5_6

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMINTENSET_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMINTENSET_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMINTENSET_EL1;
```

MSR S3_0_C15_C5_6, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENSET_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENSET_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMINTENSET_EL1 = X[t, 64];
```

A.1.8 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-8: AArch64_imp_clusterpmintenclr_el1 bit assignments

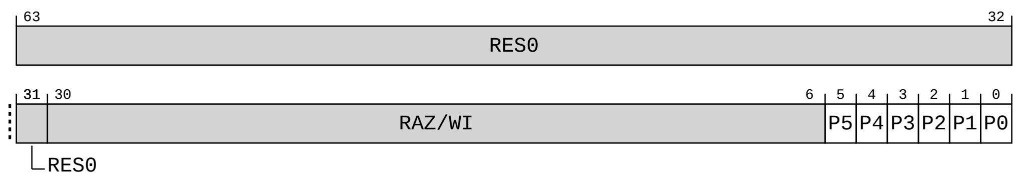


Table A-23: IMP_CLUSTERPMINTENCLR_EL1 bit descriptions

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_CLUSTERPMXEVCNTR_EL1.
		`0b0`
		When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect.
		`0b1`
		When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.
			x

[4]	P4	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.
		`0b0`
		When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect.
		`0b1`
		When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.
			x

[3]	P3	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1.
		`0b0`
		When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect.
		`0b1`
		When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.
			x

Bits	Name	Description	Reset
[2]	P2	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **`0b0`** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	`x`
[1]	P1	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **`0b0`** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	`x`
[0]	P0	Event counter overflow interrupt request disable bit for AArch64-IMP_CLUSTERPMXEVCNTR_EL1. **`0b0`** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 event counter interrupt request is enabled. When written, disables the AArch64-IMP_CLUSTERPMXEVCNTR_EL1 interrupt request.	`x`

Access

MRS <Xt>, S3_0_C15_C5_7

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0101`	`0b111`

MSR S3_0_C15_C5_7, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0101`	`0b111`

Accessibility

MRS <Xt>, S3_0_C15_C5_7

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMINTENCLR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMINTENCLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMINTENCLR_EL1;
```

MSR S3_0_C15_C5_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMINTENCLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMINTENCLR_EL1 = X[t, 64];
```

A.1.9 IMP_CLUSTERRSVD_6_0_EL1, Reserved

Reserved register.

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-9: AArch64_imp_clusterrsvd_6_0_el1 bit assignments

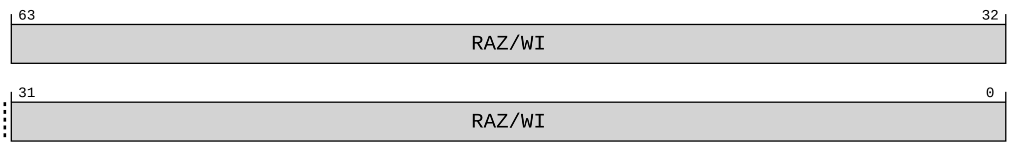


Table A-26: IMP_CLUSTERRSVD_6_0_EL1 bit descriptions

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;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERRSVD_6_0_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERRSVD_6_0_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERRSVD_6_0_EL1;
```

MSR S3_0_C15_C6_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
```

```
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERRSVD_6_0_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERRSVD_6_0_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERRSVD_6_0_EL1 = X[t, 64];
```

A.1.10 IMP_CLUSTERPMXEVTYPER_EL1, Performance Monitors Selected Event Type Register

When AArch64-IMP_CLUSTERPMSELR_EL1.SEL selects an event counter, this accesses a AArch64-IMP_CLUSTERPMXEVTYPER_EL1 register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Performance Monitors registers

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-10: AArch64_imp_clusterpmxevtyper_el1 bit assignments

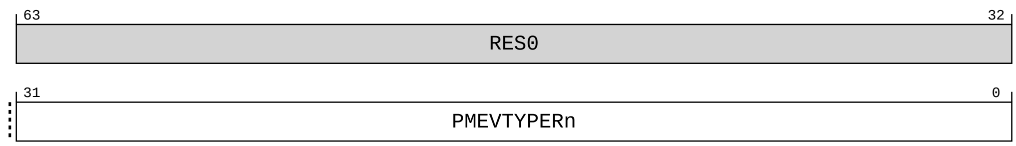


Table A-29: IMP_CLUSTERPMXEVTYPER_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31:0]	PMEVTYPERn	This register accesses AArch64-IMP_CLUSTERPMXEVTYPER_EL1 where n is the value in AArch64-IMP_CLUSTERPMSELR_EL1.SEL.	`32{x}`

Access

MRS <Xt>, S3_0_C15_C6_1

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0110`	`0b001`

MSR S3_0_C15_C6_1, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0110`	`0b001`

Accessibility

MRS <Xt>, S3_0_C15_C6_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMXEVTYPER_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMXEVTYPER_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMXEVTYPER_EL1;
```

MSR S3_0_C15_C6_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
```

```
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVTYPER_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVTYPER_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMXEVTYPER_EL1 = X[t, 64];
```

A.1.11 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-11: AArch64_imp_clusterpmxevcntr_el1 bit assignments

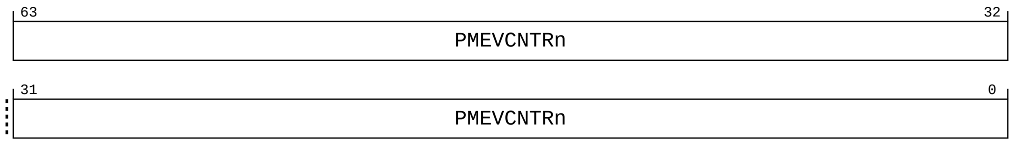


Table A-32: IMP_CLUSTERPMXEVCNTR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTRn	Value of the selected event counter, AArch64-IMP_CLUSTERPMXEVCNTR_EL1, where n is the value stored in AArch64-IMP_CLUSTERPMSELR_EL1.SEL.	`64{x}`

Access

MRS <Xt>, S3_0_C15_C6_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b010

MSR S3_0_C15_C6_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b010

Accessibility

MRS <Xt>, S3_0_C15_C6_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMXEVCNTR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMXEVCNTR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMXEVCNTR_EL1;
```

MSR S3_0_C15_C6_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
```

```
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVCNTR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMXEVCNTR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMXEVCNTR_EL1 = X[t, 64];
```

A.1.12 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-12: AArch64_imp_clusterpmceid0_el1 bit assignments

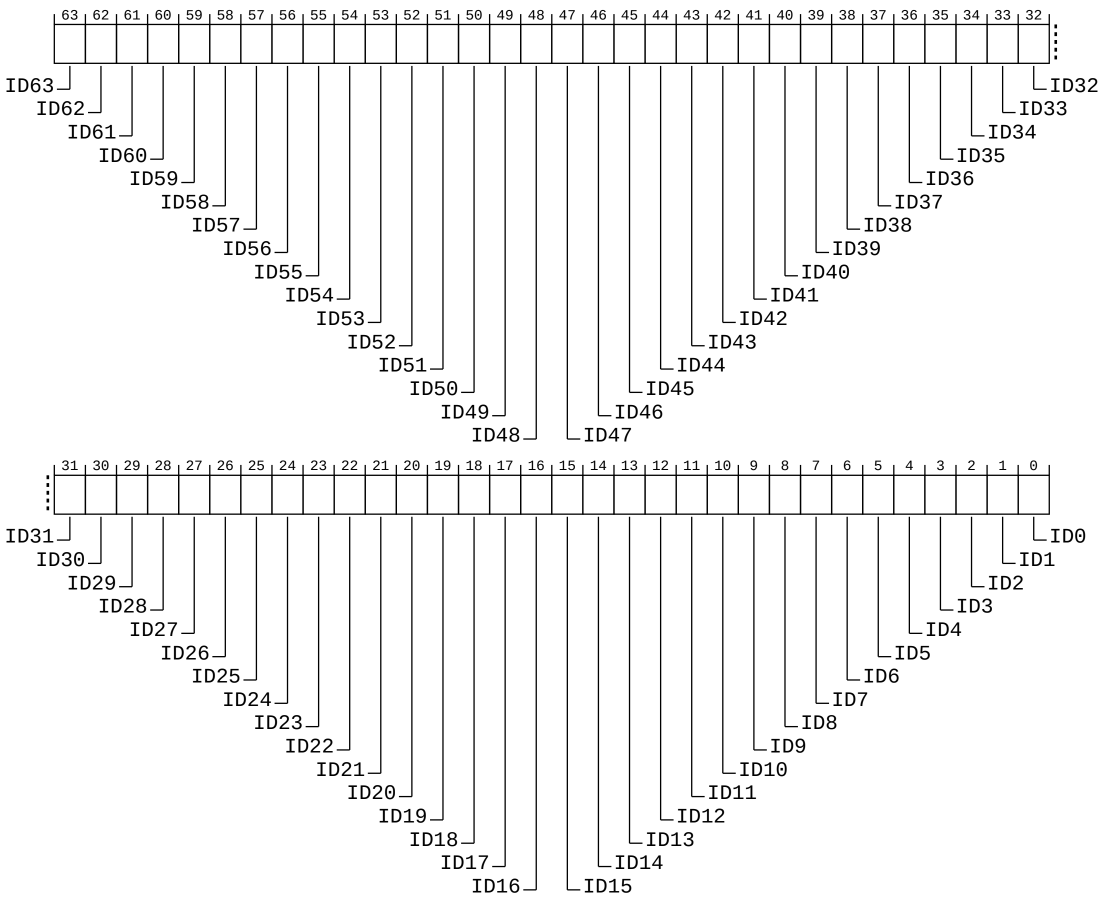


Table A-35: IMP_CLUSTERPMCEID0_EL1 bit descriptions

Bits	Name	Description	Reset
[63]	ID63	Common event `0x003F` implemented. **0b0** Event 0x003F not implemented.	0b0
[62]	ID62	Common event `0x003E` implemented. **0b0** Event 0x003E not implemented.	0b0
[61]	ID61	Common event `0x003D` implemented. **0b0** Event 0x003D not implemented.	0b0
[60]	ID60	Common event `0x003C` implemented. **0b0** Event 0x003C not implemented.	0b0

Bits	Name	Description	Reset
[59]	ID59	Common event `0x003B` implemented. **`0b0`** Event 0x003B not implemented.	`0b0`
[58]	ID58	Common event `0x003A` implemented. **`0b0`** Event 0x003A not implemented.	`0b0`
[57]	ID57	Common event `0x0039` implemented. **`0b0`** Event 0x0039 not implemented.	`0b0`
[56]	ID56	Common event `0x0038` implemented. **`0b0`** Event 0x0038 not implemented.	`0b0`
[55]	ID55	Common event `0x0037` implemented. **`0b0`** Event 0x0037 not implemented.	`0b0`
[54]	ID54	Common event `0x0036` implemented. **`0b0`** Event 0x0036 not implemented.	`0b0`
[53]	ID53	Common event `0x0035` implemented. **`0b0`** Event 0x0035 not implemented.	`0b0`
[52]	ID52	Common event `0x0034` implemented. **`0b0`** Event 0x0034 not implemented.	`0b0`
[51]	ID51	Common event `0x0033` implemented. **`0b0`** Event 0x0033 not implemented.	`0b0`
[50]	ID50	Common event `0x0032` implemented. **`0b0`** Event 0x0032 not implemented.	`0b0`
[49]	ID49	Common event `0x0031` implemented. **`0b0`** Event 0x0031 not implemented.	`0b0`
[48]	ID48	Common event `0x0030` implemented. **`0b0`** Event 0x0030 not implemented.	`0b0`
[47]	ID47	Common event `0x002F` implemented. **`0b0`** Event 0x002F not implemented.	`0b0`
[46]	ID46	Common event `0x002E` implemented. **`0b0`** Event 0x002E not implemented.	`0b0`

Bits	Name	Description	Reset
[45]	ID45	Common event `0x002D` implemented. **`0b0`** Event 0x002D not implemented.	`0b0`
[44]	ID44	Common event `0x002C` implemented. **`0b1`** L3D_CACHE_WB event implemented.	`0b1`
[43]	ID43	Common event `0x002B` implemented. **`0b1`** L3D_CACHE event implemented.	`0b1`
[42]	ID42	Common event `0x002A` implemented. **`0b1`** L3D_CACHE_REFILL event implemented.	`0b1`
[41]	ID41	Common event `0x0029` implemented. **`0b1`** L3D_CACHE_ALLOCATE event implemented.	`0b1`
[40]	ID40	Common event `0x0028` implemented. **`0b0`** Event 0x0028 not implemented.	`0b0`
[39]	ID39	Common event `0x0027` implemented. **`0b0`** Event 0x0027 not implemented.	`0b0`
[38]	ID38	Common event `0x0026` implemented. **`0b0`** Event 0x0026 not implemented.	`0b0`
[37]	ID37	Common event `0x0025` implemented. **`0b0`** Event 0x0025 not implemented.	`0b0`
[36]	ID36	Common event `0x0024` implemented. **`0b0`** Event 0x0024 not implemented.	`0b0`
[35]	ID35	Common event `0x0023` implemented. **`0b0`** Event 0x0023 not implemented.	`0b0`
[34]	ID34	Common event `0x0022` implemented. **`0b0`** Event 0x0022 not implemented.	`0b0`
[33]	ID33	Common event `0x0021` implemented. **`0b0`** Event 0x0021 not implemented.	`0b0`
[32]	ID32	Common event `0x0020` implemented. **`0b0`** Event 0x0020 not implemented.	`0b0`

Bits	Name	Description	Reset
[31]	ID31	Common event `0x001F` implemented. **0b0** Event 0x001F not implemented.	0b0
[30]	ID30	Common event `0x001E` implemented. **0b0** CHAIN event implemented.	0b0
[29]	ID29	Common event `0x001D` implemented. **0b1** BUS_CYCLES event implemented.	0b1
[28]	ID28	Common event `0x001C` implemented. **0b0** Event 0x001C not implemented.	0b0
[27]	ID27	Common event `0x001B` implemented. **0b0** Event 0x001B not implemented.	0b0
[26]	ID26	Common event `0x001A` implemented. **0b1** MEMORY_ERROR event implemented.	0b1
[25]	ID25	Common event `0x0019` implemented. **0b1** BUS_ACCESS event implemented.	0b1
[24]	ID24	Common event `0x0018` implemented. **0b0** Event 0x0018 not implemented.	0b0
[23]	ID23	Common event `0x0017` implemented. **0b0** Event 0x0017 not implemented.	0b0
[22]	ID22	Common event `0x0016` implemented. **0b0** Event 0x0016 not implemented.	0b0
[21]	ID21	Common event `0x0015` implemented. **0b0** Event 0x0015 not implemented.	0b0
[20]	ID20	Common event `0x0014` implemented. **0b0** Event 0x0014 not implemented.	0b0
[19]	ID19	Common event `0x0013` implemented. **0b0** Event 0x0013 not implemented.	0b0
[18]	ID18	Common event `0x0012` implemented. **0b0** Event 0x0012 not implemented.	0b0

Bits	Name	Description	Reset
[17]	ID17	Common event `0x0011` implemented. **`0b1`** CYCLES event implemented.	`0b1`
[16]	ID16	Common event `0x0010` implemented. **`0b0`** Event 0x0010 not implemented.	`0b0`
[15]	ID15	Common event `0x000F` implemented. **`0b0`** Event 0x000F not implemented.	`0b0`
[14]	ID14	Common event `0x000E` implemented. **`0b0`** Event 0x000E not implemented.	`0b0`
[13]	ID13	Common event `0x000D` implemented. **`0b0`** Event 0x000D not implemented.	`0b0`
[12]	ID12	Common event `0x000C` implemented. **`0b0`** Event 0x000C not implemented.	`0b0`
[11]	ID11	Common event `0x000B` implemented. **`0b0`** Event 0x000B not implemented.	`0b0`
[10]	ID10	Common event `0x000A` implemented. **`0b0`** Event 0x000A not implemented.	`0b0`
[9]	ID9	Common event `0x0009` implemented. **`0b0`** Event 0x0009 not implemented.	`0b0`
[8]	ID8	Common event `0x0008` implemented. **`0b0`** Event 0x0008 not implemented.	`0b0`
[7]	ID7	Common event `0x0007` implemented. **`0b0`** Event 0x0007 not implemented.	`0b0`
[6]	ID6	Common event `0x0006` implemented. **`0b0`** Event 0x0006 not implemented.	`0b0`
[5]	ID5	Common event `0x0005` implemented. **`0b0`** Event 0x0005 not implemented.	`0b0`
[4]	ID4	Common event `0x0004` implemented. **`0b0`** Event 0x0004 not implemented.	`0b0`

Bits	Name	Description	Reset
[3]	ID3	Common event `0x0003` implemented. **`0b0`** Event 0x0003 not implemented.	`0b0`
[2]	ID2	Common event `0x0002` implemented. **`0b0`** Event 0x0002 not implemented.	`0b0`
[1]	ID1	Common event `0x0001` implemented. **`0b0`** Event 0x0001 not implemented.	`0b0`
[0]	ID0	Common event `0x0000` implemented. **`0b0`** Event 0x0000 not implemented.	`0b0`

Access

MRS <Xt>, S3_0_C15_C6_4

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b100

MSR S3_0_C15_C6_4, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b100

Accessibility

MRS <Xt>, S3_0_C15_C6_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMCEID0_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMCEID0_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMCEID0_EL1;
```

MSR S3_0_C15_C6_4, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID0_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if ACTLR_EL3.CLUSTERPMUEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPMCEID0_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMCEID0_EL1 = X[t, 64];
```

A.1.13 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-13: AArch64_imp_clusterpmceid1_el1 bit assignments

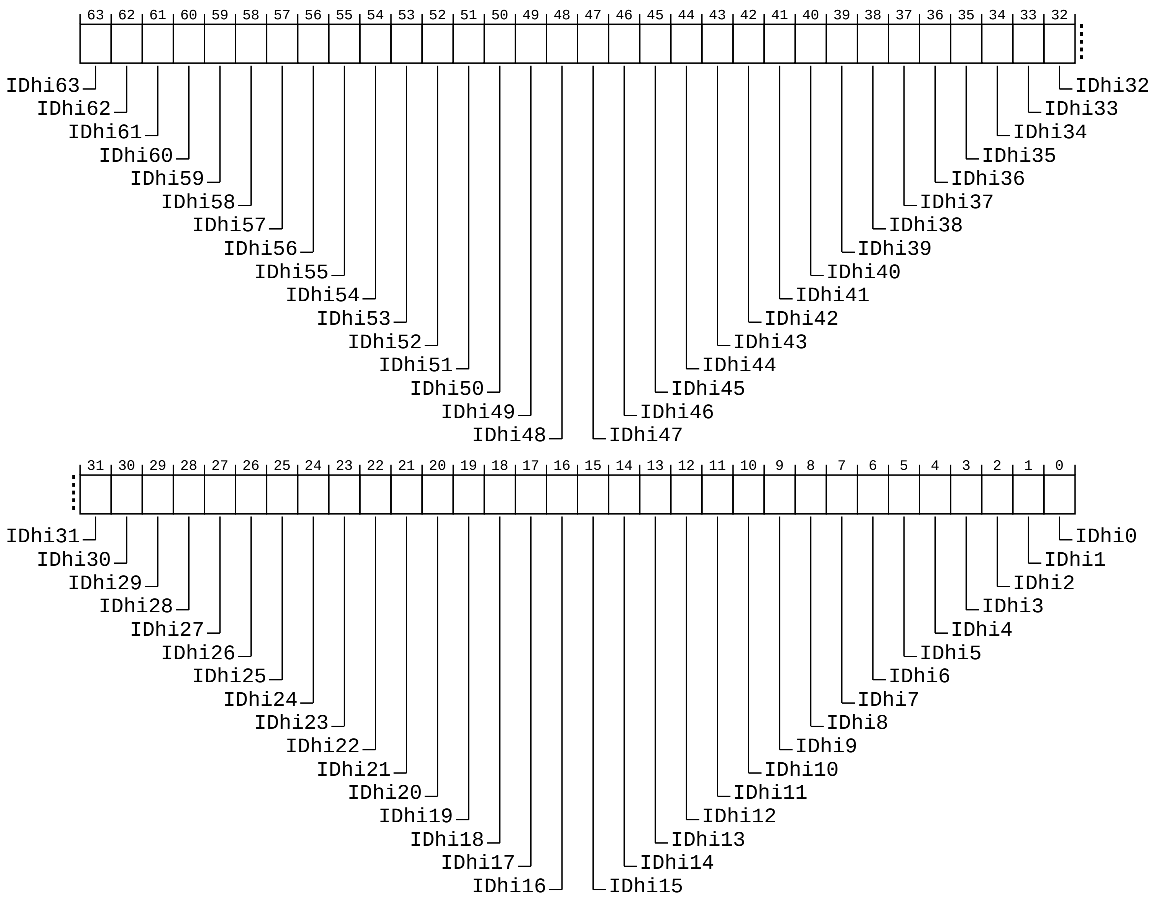


Table A-38: IMP_CLUSTERPMCEID1_EL1 bit descriptions

Bits	Name	Description	Reset
[63]	IDhi63	Common event `0x403F` implemented. **0b0** Event 0x403F not implemented.	0b0
[62]	IDhi62	Common event `0x403E` implemented. **0b0** Event 0x403E not implemented.	0b0
[61]	IDhi61	Common event `0x403D` implemented. **0b0** Event 0x403D not implemented.	0b0
[60]	IDhi60	Common event `0x403C` implemented. **0b0** Event 0x403C not implemented.	0b0

Bits	Name	Description	Reset
[59]	IDhi59	Common event `0x403B` implemented. **0b0** Event 0x403B not implemented.	`0b0`
[58]	IDhi58	Common event `0x403A` implemented. **0b0** Event 0x403A not implemented.	`0b0`
[57]	IDhi57	Common event `0x4039` implemented. **0b0** Event 0x4039 not implemented.	`0b0`
[56]	IDhi56	Common event `0x4038` implemented. **0b0** Event 0x4038 not implemented.	`0b0`
[55]	IDhi55	Common event `0x4037` implemented. **0b0** Event 0x4037 not implemented.	`0b0`
[54]	IDhi54	Common event `0x4036` implemented. **0b0** Event 0x4036 not implemented.	`0b0`
[53]	IDhi53	Common event `0x4035` implemented. **0b0** Event 0x4035 not implemented.	`0b0`
[52]	IDhi52	Common event `0x4034` implemented. **0b0** Event 0x4034 not implemented.	`0b0`
[51]	IDhi51	Common event `0x4033` implemented. **0b0** Event 0x4033 not implemented.	`0b0`
[50]	IDhi50	Common event `0x4032` implemented. **0b0** Event 0x4032 not implemented.	`0b0`
[49]	IDhi49	Common event `0x4031` implemented. **0b0** Event 0x4031 not implemented.	`0b0`
[48]	IDhi48	Common event `0x4030` implemented. **0b0** Event 0x4030 not implemented.	`0b0`
[47]	IDhi47	Common event `0x402F` implemented. **0b0** Event 0x402F not implemented.	`0b0`
[46]	IDhi46	Common event `0x402E` implemented. **0b0** Event 0x402E not implemented.	`0b0`

Bits	Name	Description	Reset
[45]	IDhi45	Common event `0x402D` implemented. **0b0** Event 0x402D not implemented.	`0b0`
[44]	IDhi44	Common event `0x402C` implemented. **0b0** Event 0x402C not implemented.	`0b0`
[43]	IDhi43	Common event `0x402B` implemented. **0b0** Event 0x402B not implemented.	`0b0`
[42]	IDhi42	Common event `0x402A` implemented. **0b0** Event 0x402A not implemented.	`0b0`
[41]	IDhi41	Common event `0x4029` implemented. **0b0** Event 0x4029 not implemented.	`0b0`
[40]	IDhi40	Common event `0x4028` implemented. **0b0** Event 0x4028 not implemented.	`0b0`
[39]	IDhi39	Common event `0x4027` implemented. **0b0** Event 0x4027 not implemented.	`0b0`
[38]	IDhi38	Common event `0x4026` implemented. **0b0** Event 0x4026 not implemented.	`0b0`
[37]	IDhi37	Common event `0x4025` implemented. **0b0** Event 0x4025 not implemented.	`0b0`
[36]	IDhi36	Common event `0x4024` implemented. **0b0** Event 0x4024 not implemented.	`0b0`
[35]	IDhi35	Common event `0x4023` implemented. **0b0** Event 0x4023 not implemented.	`0b0`
[34]	IDhi34	Common event `0x4022` implemented. **0b0** Event 0x4022 not implemented.	`0b0`
[33]	IDhi33	Common event `0x4021` implemented. **0b0** Event 0x4021 not implemented.	`0b0`
[32]	IDhi32	Common event `0x4020` implemented. **0b0** Event 0x4020 not implemented.	`0b0`

Bits	Name	Description	Reset
[31]	IDhi31	Common event `0x401F` implemented. **`0b0`** Event 0x401F not implemented.	`0b0`
[30]	IDhi30	Common event `0x401E` implemented. **`0b0`** Event 0x401E not implemented.	`0b0`
[29]	IDhi29	Common event `0x401D` implemented. **`0b0`** Event 0x401D not implemented.	`0b0`
[28]	IDhi28	Common event `0x401C` implemented. **`0b0`** Event 0x401C not implemented.	`0b0`
[27]	IDhi27	Common event `0x401B` implemented. **`0b0`** Event 0x401B not implemented.	`0b0`
[26]	IDhi26	Common event `0x401A` implemented. **`0b0`** Event 0x401A not implemented.	`0b0`
[25]	IDhi25	Common event `0x4019` implemented. **`0b0`** Event 0x4019 not implemented.	`0b0`
[24]	IDhi24	Common event `0x4018` implemented. **`0b0`** Event 0x4018 not implemented.	`0b0`
[23]	IDhi23	Common event `0x4017` implemented. **`0b0`** Event 0x4017 not implemented.	`0b0`
[22]	IDhi22	Common event `0x4016` implemented. **`0b0`** Event 0x4016 not implemented.	`0b0`
[21]	IDhi21	Common event `0x4015` implemented. **`0b0`** Event 0x4015 not implemented.	`0b0`
[20]	IDhi20	Common event `0x4014` implemented. **`0b0`** Event 0x4014 not implemented.	`0b0`
[19]	IDhi19	Common event `0x4013` implemented. **`0b0`** Event 0x4013 not implemented.	`0b0`
[18]	IDhi18	Common event `0x4012` implemented. **`0b0`** Event 0x4012 not implemented.	`0b0`

Bits	Name	Description	Reset
[17]	IDhi17	Common event `0x4011` implemented. **0b0** Event 0x4011 not implemented.	`0b0`
[16]	IDhi16	Common event `0x4010` implemented. **0b0** Event 0x4010 not implemented.	`0b0`
[15]	IDhi15	Common event `0x400F` implemented. **0b0** Event 0x400F not implemented.	`0b0`
[14]	IDhi14	Common event `0x400E` implemented. **0b0** Event 0x400E not implemented.	`0b0`
[13]	IDhi13	Common event `0x400D` implemented. **0b0** Event 0x400D not implemented.	`0b0`
[12]	IDhi12	Common event `0x400C` implemented. **0b0** Event 0x400C not implemented.	`0b0`
[11]	IDhi11	Common event `0x400B` implemented. **0b0** Event 0x400B not implemented.	`0b0`
[10]	IDhi10	Common event `0x400A` implemented. **0b0** Event 0x400A not implemented.	`0b0`
[9]	IDhi9	Common event `0x4009` implemented. **0b0** Event 0x4009 not implemented.	`0b0`
[8]	IDhi8	Common event `0x4008` implemented. **0b0** Event 0x4008 not implemented.	`0b0`
[7]	IDhi7	Common event `0x4007` implemented. **0b0** Event 0x4007 not implemented.	`0b0`
[6]	IDhi6	Common event `0x4006` implemented. **0b0** Event 0x4006 not implemented.	`0b0`
[5]	IDhi5	Common event `0x4005` implemented. **0b0** Event 0x4005 not implemented.	`0b0`
[4]	IDhi4	Common event `0x4004` implemented. **0b0** Event 0x4004 not implemented.	`0b0`

Bits	Name	Description	Reset
[3]	IDhi3	Common event `0x4003` implemented. **0b0** Event 0x4003 not implemented.	0b0
[2]	IDhi2	Common event `0x4002` implemented. **0b0** Event 0x4002 not implemented.	0b0
[1]	IDhi1	Common event `0x4001` implemented. **0b0** Event 0x4001 not implemented.	0b0
[0]	IDhi0	Common event `0x4000` implemented. **0b0** Event 0x4000 not implemented.	0b0

Access

MRS <Xt>, S3_0_C15_C6_5

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b101

MSR S3_0_C15_C6_5, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0110	0b101

Accessibility

MRS <Xt>, S3_0_C15_C6_5

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPMCEID1_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPMCEID1_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMCEID1_EL1;
```

MSR S3_0_C15_C6_5, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.CLUSTERPMUEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.CLUSTERPMUEN == '0' then
```

```
            AArch64.SystemAccessTrap(EL2, 0x18);
        elsif ACTLR_EL3.CLUSTERPMUEN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCEID1_EL1 = X[t, 64];
    elsif PSTATE.EL == EL2 then
        if ACTLR_EL3.CLUSTERPMUEN == '0' then
            if Halted() && EDSCR.SDD == '1' then
                UNDEFINED;
            else
                AArch64.SystemAccessTrap(EL3, 0x18);
        else
            IMP_CLUSTERPMCEID1_EL1 = X[t, 64];
    elsif PSTATE.EL == EL3 then
        IMP_CLUSTERPMCEID1_EL1 = X[t, 64];
```

A.2 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- Any AArch64 Generic System Control registers that are not present in Direct connect are treated as **RAZ/WI**.
- For registers with a listed reset value refer to the individual field resets documented on the register description pages.

Table A-41: Generic System Control registers summary

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERCFR_EL1	3	0	C15	C3	0	See individual bit resets.	64-bit	Cluster Configuration Register	Yes
IMP_CLUSTERIDR_EL1	3	0	C15	C3	1	See individual bit resets.	64-bit	Cluster Main Revision Register	Yes
IMP_CLUSTERREVIDR_EL1	3	0	C15	C3	2	See individual bit resets.	64-bit	Cluster ECO ID Register	Yes
IMP_CLUSTERACTLR_EL1	3	0	C15	C3	3	See individual bit resets.	64-bit	Cluster Auxiliary Control Register	Yes
IMP_CLUSTERECTLR_EL1	3	0	C15	C3	4	See individual bit resets.	64-bit	Cluster Extended Control Register	No
IMP_CLUSTERPWRCTLR_EL1	3	0	C15	C3	5	See individual bit resets.	64-bit	Cluster Power Control Register	No
IMP_CLUSTERPWRDN_EL1	3	0	C15	C3	6	See individual bit resets.	64-bit	Cluster Power Down Register	No

Name	Op0	Op1	CRn	CRm	Op2	Reset	Width	Description	Present in Direct connect
IMP_CLUSTERPWRSTAT_EL1	3	0	C15	C3	7	See individual bit resets.	64-bit	Cluster Power Status Register	No
IMP_CLUSTERL3DNTH0_EL1	3	0	C15	C4	0	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register	No
IMP_CLUSTERL3DNTH1_EL1	3	0	C15	C4	1	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register	No
IMP_CLUSTERL3UPTH0_EL1	3	0	C15	C4	2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register	No
IMP_CLUSTERL3UPTH1_EL1	3	0	C15	C4	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register	No
IMP_CLUSTERBUSQOS_EL1	3	0	C15	C4	4	See individual bit resets.	64-bit	Cluster Bus QoS Control Register	No
IMP_CLUSTERL3HIT_EL1	3	0	C15	C4	5	See individual bit resets.	64-bit	Cluster L3 Hit Counter Register	No
IMP_CLUSTERL3MISS_EL1	3	0	C15	C4	6	See individual bit resets.	64-bit	Cluster L3 Miss Counter Register	No
IMP_CLUSTERPPSTART_EL1	3	0	C15	C9	0	See individual bit resets.	64-bit	Cluster Peripheral Port Start Address Register	No
IMP_CLUSTERPPEND_EL1	3	0	C15	C9	1	See individual bit resets.	64-bit	Cluster Peripheral Port End Address Register	No
IMP_CLUSTERCFR2_EL1	3	0	C15	C9	2	See individual bit resets.	64-bit	Cluster Configuration Register 2	No
IMP_CLUSTERL3UPTH2_EL1	3	0	C15	C9	3	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register	No
IMP_CLUSTERCDBG_EL3	3	6	C15	C4	7	See individual bit resets.	64-bit	Cluster Cache Debug Register	No
IMP_CLUSTERPMMDCR_EL3	3	6	C15	C6	3	See individual bit resets.	64-bit	Monitor Debug Configuration Register (EL3)	No

A.2.1 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 405 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 0xxx xxxx xxx0 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-14: AArch64_imp_clustercfr_el1 bit assignments

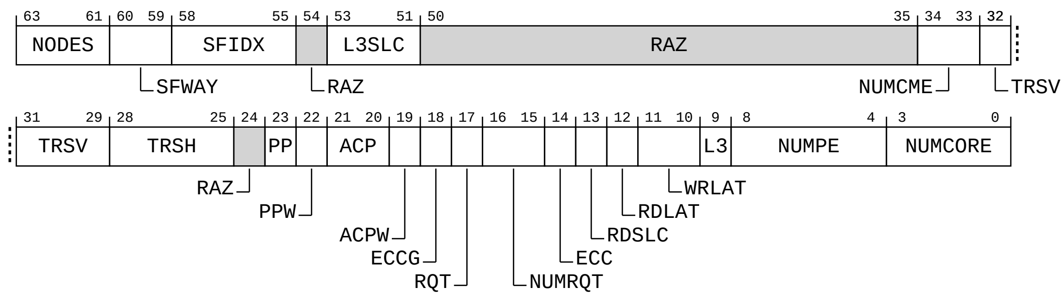


Table A-42: IMP_CLUSTERCFR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:61]	NODES	Number of transport nodes. **0b001** One node. **0b010** Two nodes. **0b011** Three nodes. **0b100** Four nodes. **0b101** Six nodes.	xxx

Bits	Name	Description	Reset
[60:59]	SFWAY	Number of Snoop Filter ways. **0b00** 4 ways **0b01** 6 ways **0b10** 8 ways **0b11** 12 ways	xx
[58:55]	SFIDX	Log2 of the number of snoop filter indexes.	xxxx
[54]	RAZ	Reserved	RAZ
[53:51]	L3SLC	Number of L3 cache slices. **0b000** Eight L3 cache slices. **0b001** One L3 cache slice. **0b010** Two L3 cache slices. **0b100** Four L3 cache slices.	xxx
[50:35]	RAZ	Reserved	RAZ
[34:33]	NUMCME	Number of SME2 units configured **0b00** No SME2 units present **0b01** One SME2 unit present **0b10** Two SME2 units present	xx

Bits	Name	Description	Reset
[32:29]	TRSV	Transport register slices, vertical. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx
[28:25]	TRSH	Transport register slices, horizontal. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx
[24]	RAZ	Reserved	RAZ

Bits	Name	Description	Reset
[23]	PP	Peripheral port presence. **0b0** No peripheral port present **0b1** Peripheral port present	x
[22]	PPW	Peripheral port width. **0b0** 64 bit data width **0b1** 256 bit data width	x
[21:20]	ACP	ACP interface presence. **0b00** No ACP interface present **0b01** One ACP interface present **0b10** Two ACP interface present	xx
[19]	ACPW	ACP interface width. **0b0** 128 bit data width **0b1** 256 bit data width	x
[18]	ECCG	L3 DataRAM ECC Granule. **0b0** 128 bit data ECC **0b1** 256 bit data ECC	x
[17]	RQT	Requester bus interface type. **0b0** AXI interface **0b1** CHI interface	x
[16:15]	NUMRQT	Number of Requester interfaces. **0b00** One requester **0b01** Two requesters **0b10** Three requesters **0b11** Four requesters	xx

Bits	Name	Description	Reset
[14]	ECC	SCU-L3 ECC configuration. **0b0** SCU-L3 is configured with no ECC **0b1** SCU-L3 is configured with ECC	x
[13]	RDSLC	L3 data RAM read register slice. **0b0** No register slice present **0b1** Register slice present	x
[12]	RDLAT	L3 Data RAM read latency. **0b0** Two cycle output delay from L3 data RAMs **0b1** Three cycle output delay from L3 data RAMs	x
[11:10]	WRLAT	L3 Data RAM write latency. **0b00** One cycle input delay from L3 data RAMs **0b01** Two cycle input delay from L3 data RAMs **0b10** Two cycle input delay plus a one cycle hold	xx
[9]	L3	L3 cache presence. **0b0** No L3 cache present **0b1** L3 cache present	x
[8:4]	NUMPE	Number of PEs present in the cluster. For single threaded cores, this number will be the same as bits [3:0]; for multi-threaded cores it will be larger.	5{x}

Bits	Name	Description	Reset
[3:0]	NUMCORE	Number of cores present in the cluster. **0b0000** One core **0b0001** Two cores **0b0010** Three cores **0b0011** Four cores **0b0100** Five cores **0b0101** Six cores **0b0110** Seven cores **0b0111** Eight cores **0b1000** Nine core **0b1001** Ten cores **0b1010** Eleven cores **0b1011** Twelve cores **0b1100** Thirteen cores **0b1101** Fourteen cores	xxxx

Access

MRS <Xt>, S3_0_C15_C3_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b000

MSR S3_0_C15_C3_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b000

Accessibility

MRS <Xt>, S3_0_C15_C3_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERCFR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERCFR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERCFR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERCFR_EL1 = X[t, 64];
```

A.2.2 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 392 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 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-15: AArch64_imp_clusteridr_el1 bit assignments

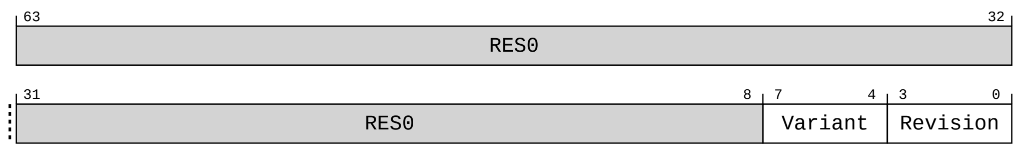


Table A-45: IMP_CLUSTERIDR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:8]	RES0	Reserved	RES0
[7:4]	Variant	Indicates the variant of the DSU. This is the major revision number x in the rx part of the rxpy description of the product revision status. **0b0000** Cluster major revision 0.	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. **0b0010** Cluster minor revision 2.	0b0010

Access

MRS <Xt>, S3_0_C15_C3_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b001

MSR S3_0_C15_C3_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b001

Accessibility

MRS <Xt>, S3_0_C15_C3_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERIDR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERIDR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERIDR_EL1;
```

MSR S3_0_C15_C3_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERIDR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERIDR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERIDR_EL1 = X[t, 64];
```

A.2.3 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 393 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-16: AArch64_imp_clusterrevidr_el1 bit assignments

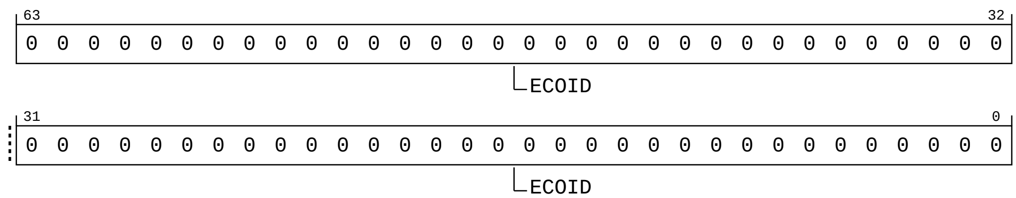


Table A-48: IMP_CLUSTERREVIDR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	ECOID	Contains ECO information. Refer to the errata documentation for any bit allocations. **0b0000000000000000000000000000000000000000000000000000000000000000** Customer ECO ID	0x0000000000000000

Access

MRS <Xt>, S3_0_C15_C3_2

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b010

MSR S3_0_C15_C3_2, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b010

Accessibility

MRS <Xt>, S3_0_C15_C3_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERREVIDR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERREVIDR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERREVIDR_EL1;
```

MSR S3_0_C15_C3_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
```

```
        IMP_CLUSTERREVIDR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERREVIDR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERREVIDR_EL1 = X[t, 64];
```

A.2.4 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 411 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-17: AArch64_imp_clusteractlr_el1 bit assignments

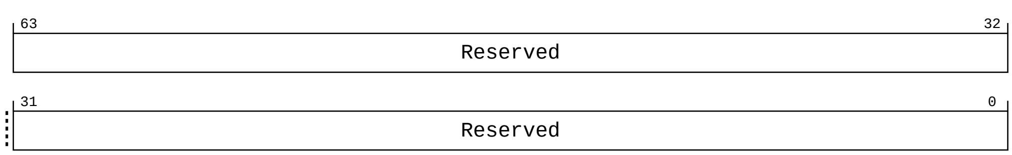


Table A-51: IMP_CLUSTERACTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	Reserved	Reserved for Arm internal use	64{x}

Access

MRS <Xt>, S3_0_C15_C3_3

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b011

MSR S3_0_C15_C3_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b011

Accessibility

MRS <Xt>, S3_0_C15_C3_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERACTLR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERACTLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERACTLR_EL1;
```

MSR S3_0_C15_C3_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ACTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ACTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ACTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERACTLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ACTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ACTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
```

```
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERACTLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERACTLR_EL1 = X[t, 64];
```

A.2.5 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 413 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 1010 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-18: AArch64_imp_clusterectlr_el1 bit assignments

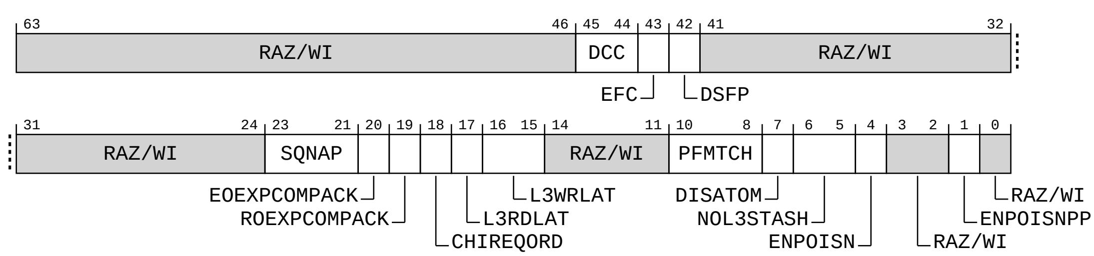


Table A-54: IMP_CLUSTERECTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:46]	**RAZ/WI**	Reserved	**RAZ/WI**
[45:44]	DCC	Downstream cache control. Controls whether evictions of clean cachelines send data on the CHI interface. Set this based on whether there is a cache on the path to memory. This bit is **RES0** in direct connect configuration. **`0b00`** Disables sending data when clean cachelines are evicted. **`0b01`** Enables sending WriteEvictFull transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **`0b10`** Enables sending WriteEvictOrEvict transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **`0b11`** Enables sending WriteEvictOrEvict transactions when Unique Clean or Shared Clean cachelines are evicted. This is the reset value.	`0b11`
[43]	EFC	Eviction flush control. Controls whether hardware cache flushes and DC CISW instructions send data when evicting clean cachelines on the CHI interface. This bit is **RES0** in direct connect configuration. **`0b0`** Disables sending data when hardware cache flushes or DC CISW instructions evict a clean cacheline. Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP). This is the reset value. **`0b1`** Sending of data when hardware cache flushes or DC CISW instructions evict clean cachelines is controlled by Downstream Cache Control (DCC). Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP).	`0b0`
[42]	DSFP	Downstream snoop filter present. Enables sending Evict transactions on the CHI interface when clean cachelines are evicted without data. Enable this if there is at least one snoop filter in the path to memory. This bit is **RES0** in direct connect configuration. **`0b0`** Disables sending Evict transactions when clean cachelines are evicted without data. **`0b1`** Enables sending of Evict transactions when clean cachelines are evicted without data. This is the reset value.	`0b1`
[41:24]	**RAZ/WI**	Reserved	**RAZ/WI**

Bits	Name	Description	Reset
[23:21]	SQNAP	L3 data RAM quick nap sub-enable time. Controls the number of SCLK cycles to wait after an L3 data RAM access before setting the corresponding quick nap pin for the sub-bank. If this sub-bank of the RAM has been idle for this number of cycles, then it is put into quick nap, even if the entire slice timeout controlled by CLUSTERPWRCTLR.QNAP has not become idle yet. This bit is **RES0** in direct connect configuration. **0b000** Per-pin quick nap disabled **0b001** Enabled with 8 cycle timeout **0b010** Enabled with 12 cycle timeout **0b011** Enabled with 16 cycle timeout **0b100** Enabled with 24 cycle timeout **0b101** Enabled with 32 cycle timeout **0b110** Enabled with 64 cycle timeout **0b111** Enabled with 128 cycle timeout	0b101
[20]	EOEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Endpoint Order transactions to the system. This bit is **RES0** in direct connect configuration. **0b0** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=1	0b0
[19]	ROEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Request Order transactions to the system. This bit is **RES0** in direct connect configuration. **0b0** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=1	0b0
[18]	CHIREQORD	Allow Request Order on CHI ports. Enables the use of Request Order when sending Non-snoopable CHI transactions to the system for Dev-R and Normal NC memory. This bit is **RES0** in direct connect configuration. **0b0** Disables sending Request Order on the CHI interface to the system. Will send No Order instead. **0b1** Enables sending Request Order on the CHI interface to the system for Non-snoopable transactions.	0b0

Bits	Name	Description	Reset
[17]	L3RDLAT	L3 data RAM read (output) latency. This bit is **RES0** in direct connect configuration. **0b0** The L3 data RAM output latency is 2 cycles. **0b1** The L3 data RAM output latency is 3 cycles.	`x` [1]
[16:15]	L3WRLAT	L3 data RAM write (input) latency. This bit is **RES0** in direct connect configuration. **0b00** The L3 data RAM input latency is 1 cycle with an additional hold cycle. **0b01** The L3 data RAM input latency is 2 cycles without an additional hold cycle. **0b10** The L3 data RAM input latency is 2 cycles with an additional hold cycle. This is only usable if the L3 data RAM output latency is 3 cycles.	`xx` [2]
[14:11]	**RAZ/WI**	Reserved	**RAZ/WI**
[10:8]	PFMTCH	Prefetch matching delay. Controls the amount of tie a prefetch waits for a possible match with a later read. Encoded as powers of 2, from 1-128. This bit is **RES0** in direct connect configuration. **0b000** Wait for 1 cycle. **0b001** Wait for 2 cycles. **0b010** Wait for 4 cycles. **0b011** Wait for 8 cycles. **0b100** Wait for 16 cycles. **0b101** Wait for 32 cycles. **0b110** Wait for 64 cycles. **0b111** Wait for 128 cycles.	`0b101`
[7]	DISATOM	Disable cacheable shareable atomics being sent to the interconnect. This bit is **RES0** in direct connect configuration. **0b0** Cacheable shareable atomics will be sent to the interconnect if the BROADCASTATOMIC pin is set. **0b1** Cacheable shareable atomics will be handled inside the cluster.	`0b0`

[1] This field resets to the value of the L3_DATA_RD_LATENCY configuration parameter.
[2] This field resets to the value of the L3_DATA_WR_LATENCY configuration parameter.

Bits	Name	Description	Reset
[6:5]	NOL3STASH	CPU StashOnce request behaviour when L3 is not present or powered down. This bit is **RES0** in direct connect configuration. **`0b00`** Stashes are sent out to the interconnect, if supported. **`0b01`** Normal read request sent to interconnect. **`0b10`** StashOnce has no effect.	`0b10`
[4]	ENPOISN	Interconnect data poisoning support for the CHI Requester(s). This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES1** in direct connect configuration. **`0b0`** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **`0b1`** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	`0b1`
[3:2]	**RAZ/WI**	Reserved	**RAZ/WI**
[1]	ENPOISNPP	Interconnect data poisoning support for the CHI Peripheral Port. This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES1** in direct connect configuration. **`0b0`** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **`0b1`** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	`0b1`
[0]	**RAZ/WI**	Reserved	**RAZ/WI**

Access

MRS <Xt>, S3_0_C15_C3_4

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0011`	`0b100`

MSR S3_0_C15_C3_4, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b0011`	`0b100`

Accessibility

MRS <Xt>, S3_0_C15_C3_4

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
```

```
        X[t, 64] = IMP_CLUSTERECTLR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERECTLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERECTLR_EL1 = X[t, 64];
```

A.2.6 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 1011 0100 0000 0001 0111 0000

Bit descriptions

Figure A-19: AArch64_imp_clusterpwrctlr_el1 bit assignments

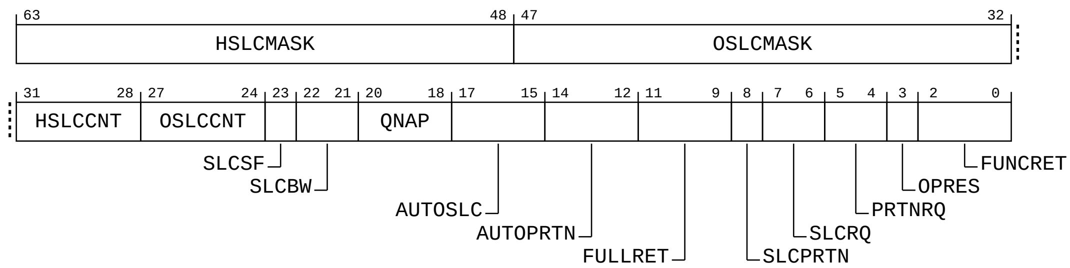


Table A-57: IMP_CLUSTERPWRCTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:48]	HSLCMASK	Half slice core mask. This contains one bit per core or SME2 unit, and if that bit is set then the core/SME2 unit is included in the count that is compared with the HSLCCNT field. Bits above the number of cores plus SME2 units implemented are **RAZ/WI**.	`0x0000`
[47:32]	OSLCMASK	One slice core mask. This contains one bit per core or SME2 unit, and if that bit is set then the core/SME2 unit is included in the count that is compared with the OSLCCNT field. Bits above the number of cores plus SME2 units implemented are **RAZ/WI**.	`0x0000`
[31:28]	HSLCCNT	Half slice core count. When AUTOSLC is non-zero, if the number of cores or SME2 units 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/SME2 unit 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 or SME2 units 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/SME2 unit that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this count.	`0b0000`
[23]	SLCSF	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 and SME2 units then this will prevent the slice powerdown. Note that a core/SME2 unit that is off but has the IMP_CLUSTERPWRDN_EL1.SHORTSLP bit set will be treated as if it was on for this calculation. **`0b0`** Ignore the snoop filter in AUTOSLC decisions **`0b1`** AUTOSLC will ensure enough slices are powered on for the currently powered on cores and SME2s units	`0b1`

Bits	Name	Description	Reset
[22:21]	SLCBW	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. **0b00** Ignore the slice bandwidth in AUTOSLC decisions **0b01** AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 3% of the AUTOSLC time period **0b10** AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 6% of the AUTOSLC time period **0b11** AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 12% of the AUTOSLC time period	0b01
[20:18]	QNAP	L3 data RAM quick nap enable time. Controls the number of SCLK cycles to wait after each L3 slice is empty before setting all the L3 data RAM quick nap pins for that slice. Note that sub-banks of the RAM can enter quick nap before this point based on the CLUSTERECTLR.SQNAP field. **0b000** Quick nap disabled **0b001** Enabled with 8 cycle timeout **0b010** Enabled with 12 cycle timeout **0b011** Enabled with 16 cycle timeout **0b100** Enabled with 24 cycle timeout **0b101** Enabled with 32 cycle timeout **0b110** Enabled with 64 cycle timeout **0b111** Enabled with 128 cycle timeout	0b101

Bits	Name	Description	Reset
[17:15]	AUTOSLC	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. **0b000** Disabled **0b001** 524,288 architectural timer ticks, time period of 524us **0b010** 1,048,576 architectural timer ticks, time period of 1ms **0b011** 2,097,152 architectural timer ticks, time period of 2.1ms **0b100** 4,194,304 architectural timer ticks, time period of 4.2ms **0b101** 8,388,608 architectural timer ticks, time period of 8.4ms **0b110** 16,777,216 architectural timer ticks, time period of 16.8ms **0b111** 33,554,432 architectural timer ticks, time period of 33.6ms	0b000
[14:12]	AUTOPRTN	Enable automatic RAM power down and configure evaluation time period. Note that a shorter time period allows better responsiveness to changing workloads, however if it is too short then the cost of frequent resizing can be too high. **0b000** Disabled **0b001** 524,288 architectural timer ticks, time period of 524us **0b010** 1,048,576 architectural timer ticks, time period of 1ms **0b011** 2,097,152 architectural timer ticks, time period of 2.1ms **0b100** 4,194,304 architectural timer ticks, time period of 4.2ms **0b101** 8,388,608 architectural timer ticks, time period of 8.4ms **0b110** 16,777,216 architectural timer ticks, time period of 16.8ms **0b111** 33,554,432 architectural timer ticks, time period of 33.6ms	0b000

Bits	Name	Description	Reset
[11:9]	FULLRET	Enable the FULL_RET slice powerdown mode and time period. Note that while this would typically be a longer period than the FUNCRET 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. **0b000** Disabled **0b001** 128 architectural timer ticks, time period of 128ns **0b010** 512 architectural timer ticks, time period of 512ns **0b011** 2,048 architectural timer ticks, time period of 2us **0b100** 4,096 architectural timer ticks, time period of 4.1us **0b101** 8,192 architectural timer ticks, time period of 8.2us **0b110** 16,384 architectural timer ticks, time period of 16.4us **0b111** 32,768 architectural timer ticks, time period of 32.8us	`0b000`
[8]	SLCPRTN	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 **0b0** AUTOSLC decisions should ignore the AUTOPRTN status **0b1** AUTOSLC decisions should include the AUTOPRTN status	`0b1`
[7:6]	SLCRQ	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. **0b00** Request that one L3 cache slice is powered on **0b01** Request that all L3 cache slices are powered on **0b10** Request that half the L3 cache slices are powered on	`0b01`
[5:4]	PRTNRQ	Cache portion power request. These bits are passed to the PPU as an advisory request for which portions to power. Note that these bits are only used when AUTOPRTN bits are 3'b000. **0b00** Request that none of the L3 cache portions in each slice is powered on **0b01** Request that half of the L3 cache portions in each slice are powered on **0b11** Request that both of the L3 cache portions in each slice are powered on	`0b11`

Bits	Name	Description	Reset
[3]	OPRES	Restore previous operating mode after cluster power off **0b0** Operating mode defaults to ALL RAM ALL SLICE when powering on the cluster **0b1** Enable restoring of previous operating mode	0b0
[2:0]	FUNCRET	L3 Data RAM retention control. **0b000** Disable the retention circuit. **0b001** 128 architectural timer ticks, time period of 128ns minimum delay before retention **0b010** 512 architectural timer ticks, time period of 512ns minimum delay before retention **0b011** 2,048 architectural timer ticks, time period of 2us minimum delay before retention **0b100** 4,096 architectural timer ticks, time period of 4.1us minimum delay before retention **0b101** 8,192 architectural timer ticks, time period of 8.2us minimum delay before retention **0b110** 16,384 architectural timer ticks, time period of 16.4us minimum delay before retention **0b111** 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
        X[t, 64] = IMP_CLUSTERPWRCTLR_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPWRCTLR_EL1;
```

```
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPWRCTLR_EL1 = X[t, 64];
```

A.2.7 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-20: AArch64_imp_clusterpwrdn_el1 bit assignments

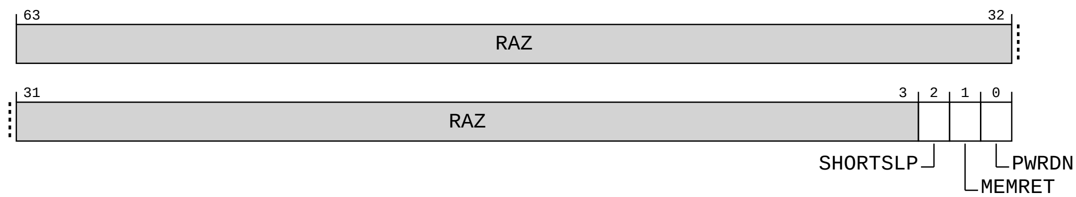


Table A-60: IMP_CLUSTERPWRDN_EL1 bit descriptions

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 will prevent the cluster from entering OFF or OFF_EMU.	0b0
[0]	PWRDN	Indicate to the PPU that cluster power is required even when all cores are powered down. This will prevent the cluster from entering OFF, OFF_EMU, MEM_RET, or MEM_RET_EMU.	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
        X[t, 64] = IMP_CLUSTERPWRDN_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPWRDN_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPWRDN_EL1 = X[t, 64];
```

A.2.8 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-21: AArch64_imp_clusterpwrstat_el1 bit assignments

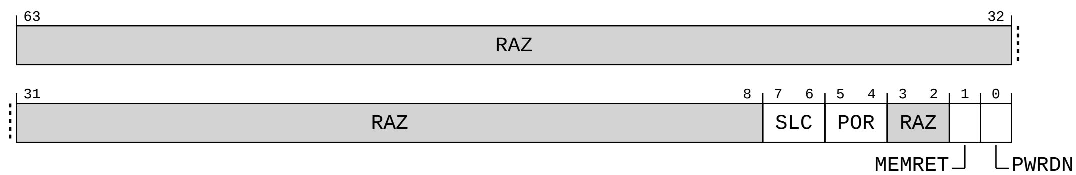


Table A-63: IMP_CLUSTERPWRSTAT_EL1 bit descriptions

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, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0011	0b111

Accessibility

MRS <Xt>, S3_0_C15_C3_7

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
```

```
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPWRSTAT_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPWRSTAT_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPWRSTAT_EL1;
```

MSR S3_0_C15_C3_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.PWREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRSTAT_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.PWREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.PWREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPWRSTAT_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPWRSTAT_EL1 = X[t, 64];
```

A.2.9 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 399 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-22: AArch64_imp_clusterl3dnth0_el1 bit assignments

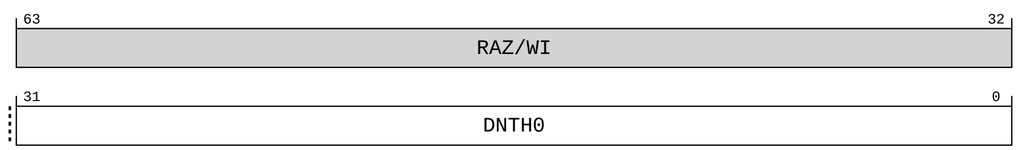


Table A-66: IMP_CLUSTERL3DNTH0_EL1 bit descriptions

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
        X[t, 64] = IMP_CLUSTERL3DNTH0_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3DNTH0_EL1;
elsif PSTATE.EL == EL3 then
```

```
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3DNTH0_EL1 = X[t, 64];
```

A.2.10 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 400 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-23: AArch64_imp_clusterl3dnth1_el1 bit assignments

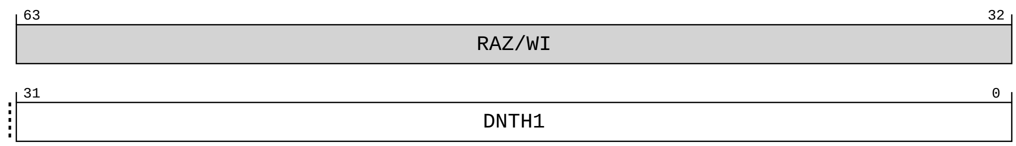


Table A-69: IMP_CLUSTERL3DNTH1_EL1 bit descriptions

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
        X[t, 64] = IMP_CLUSTERL3DNTH1_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3DNTH1_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3DNTH1_EL1 = X[t, 64];
```

A.2.11 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 402 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-24: AArch64_imp_clusterl3upth0_el1 bit assignments

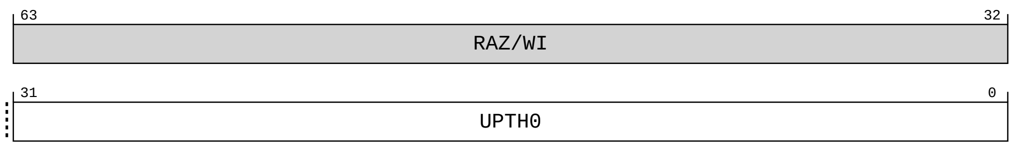


Table A-72: IMP_CLUSTERL3UPTH0_EL1 bit descriptions

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
        X[t, 64] = IMP_CLUSTERL3UPTH0_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3UPTH0_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3UPTH0_EL1 = X[t, 64];
```

A.2.12 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 403 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-25: AArch64_imp_clusterl3upth1_el1 bit assignments

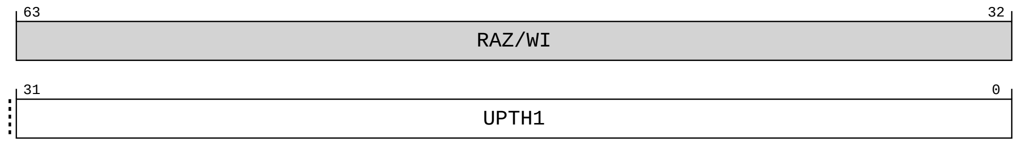


Table A-75: IMP_CLUSTERL3UPTH1_EL1 bit descriptions

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
        X[t, 64] = IMP_CLUSTERL3UPTH1_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3UPTH1_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3UPTH1_EL1 = X[t, 64];
```

A.2.13 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 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 1110 1110 1110 1110 1110 1110

Bit descriptions

Figure A-26: AArch64_imp_clusterbusqos_el1 bit assignments

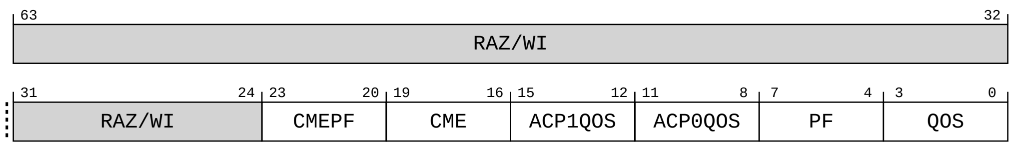


Table A-78: IMP_CLUSTERBUSQOS_EL1 bit descriptions

Bits	Name	Description	Reset
[63:24]	**RAZ/WI**	Reserved	**RAZ/WI**
[23:20]	CMEPF	Value driven on the CHI bus QoS field for prefetches and non-latency critical accesses from any SME2 unit.	0b1110
[19:16]	CME	Value driven on the CHI bus QoS field for latency critical demand accesses from any SME2 unit.	0b1110
[15:12]	ACP1QOS	Value driven on the CHI bus QoS field for ACP 1 accesses.	0b1110
[11:8]	ACP0QOS	Value driven on the CHI bus QoS field for ACP 0 accesses.	0b1110
[7:4]	PF	Value driven on the CHI bus QoS field for prefetches and non-latency critical accesses. This includes all L3 evictions, including those caused by an SME2 unit or ACP transaction allocating to the L3 cache.	0b1110
[3:0]	QOS	Value driven on the CHI bus QoS field for latency critical 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
        X[t, 64] = IMP_CLUSTERBUSQOS_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERBUSQOS_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERBUSQOS_EL1 = X[t, 64];
```

A.2.14 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-27: AArch64_imp_clusterl3hit_el1 bit assignments

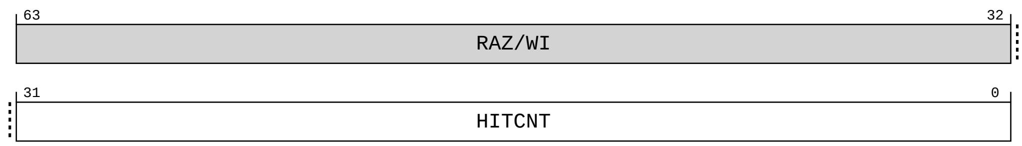


Table A-81: IMP_CLUSTERL3HIT_EL1 bit descriptions

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
        X[t, 64] = IMP_CLUSTERL3HIT_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3HIT_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3HIT_EL1 = X[t, 64];
```

A.2.15 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-28: AArch64_imp_clusterl3miss_el1 bit assignments

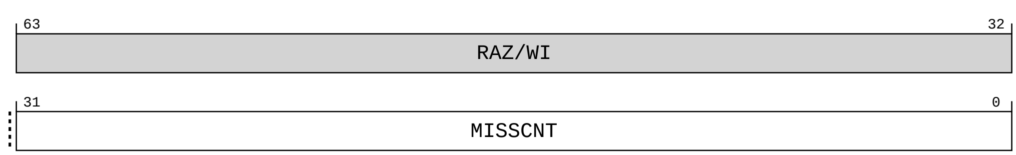


Table A-84: IMP_CLUSTERL3MISS_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:0]	MISSCNT	Count of number of L3 misses, for use in portion control calculations.	0x00000000

Access

MRS <Xt>, S3_0_C15_C4_6

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b110

MSR S3_0_C15_C4_6, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b0100	0b110

Accessibility

MRS <Xt>, S3_0_C15_C4_6

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERL3MISS_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3MISS_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3MISS_EL1 = X[t, 64];
```

A.2.16 IMP_CLUSTERPPSTART_EL1, Cluster Peripheral Port Start Address Register

Determines the start address for the peripheral port address range.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

```
0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0000 0000
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-29: AArch64_imp_clusterppstart_el1 bit assignments

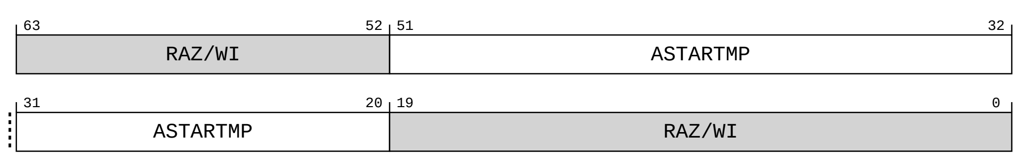


Table A-87: IMP_CLUSTERPPSTART_EL1 bit descriptions

Bits	Name	Description	Reset
[63:52]	RAZ/WI	Reserved	RAZ/WI
[51:20]	ASTARTMP	Start address for peripheral port address range.	32{x}
[19:0]	RAZ/WI	Reserved	RAZ/WI

Access

MRS <Xt>, S3_0_C15_C9_0

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b000

MSR S3_0_C15_C9_0, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b000

Accessibility

MRS <Xt>, S3_0_C15_C9_0

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPPSTART_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPPSTART_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPPSTART_EL1;
```

MSR S3_0_C15_C9_0, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPSTART_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ECTLREN == '0' then
```

```
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPSTART_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPPSTART_EL1 = X[t, 64];
```

A.2.17 IMP_CLUSTERPPEND_EL1, Cluster Peripheral Port End Address Register

Determines the end address for the peripheral port address range.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

```
0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000 0000 0000 0000
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |
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-30: AArch64_imp_clusterppend_el1 bit assignments

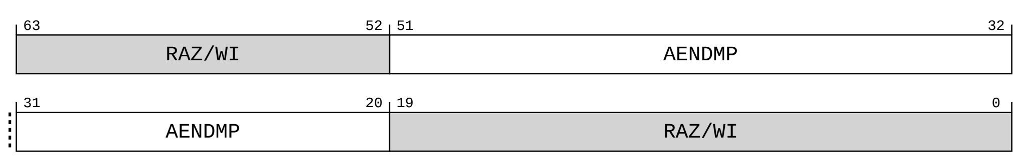


Table A-90: IMP_CLUSTERPPEND_EL1 bit descriptions

Bits	Name	Description	Reset
[63:52]	RAZ/WI	Reserved	RAZ/WI
[51:20]	AENDMP	End address for peripheral port address range. If the end address is the same as the start address then no accesses will be sent to the peripheral port. The end address is non-inclusive. The defined range is from the start address to the end address but excluding the byte at the end address.	32{x}
[19:0]	RAZ/WI	Reserved	RAZ/WI

Access

MRS <Xt>, S3_0_C15_C9_1

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b001

MSR S3_0_C15_C9_1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b1111	0b1001	0b001

Accessibility

MRS <Xt>, S3_0_C15_C9_1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERPPEND_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERPPEND_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPPEND_EL1;
```

MSR S3_0_C15_C9_1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && ACTLR_EL2.ECTLREN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPEND_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
```

```
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && ACTLR_EL3.ECTLREN == '0' then
        UNDEFINED;
    elsif ACTLR_EL3.ECTLREN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        IMP_CLUSTERPPEND_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPPEND_EL1 = X[t, 64];
```

A.2.18 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 417 bits [63:0].

Attributes

Width

64

Functional group

Generic System Control

Access type

See bit descriptions

Reset value

```
0000 0000 0000 0000 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-31: AArch64_imp_clustercfr2_el1 bit assignments

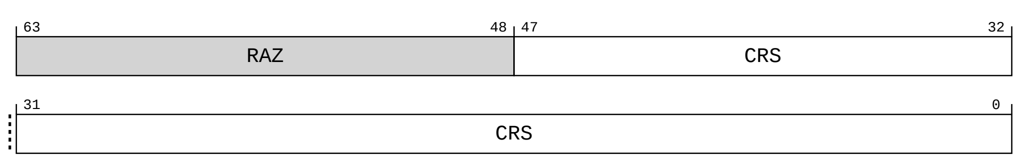


Table A-93: IMP_CLUSTERCFR2_EL1 bit descriptions

Bits	Name	Description	Reset
[63:48]	RAZ	Reserved	RAZ

[47:0]	CRS	Core and SME2 unit register slices : Each three bits represents a core or SME2 unit, with bits [2:0] for core 0 up to bits [41:39] for core 13. The SME2 unit bits follow immediately after the highest numbered core, e.g. bits [14:12] for the first SME2 unit when 4 cores are configured.
		`0b000000000000000000000000000000000000000000000000`
		No register slices
		`0b000000000000000000000000000000000000000000000001`
		One register slice
		`0b000000000000000000000000000000000000000000000010`
		Two register slices
		`0b000000000000000000000000000000000000000000000011`
		Three register slices
		`0b000000000000000000000000000000000000000000000100`
		Four register slices
		`0b000000000000000000000000000000000000000000000101`
		Five register slices
		`0b000000000000000000000000000000000000000000000110`
		Six register slices
		`0b000000000000000000000000000000000000000000000111`
		Seven register slices
			`48{x}`

Access

MRS <Xt>, S3_0_C15_C9_2

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b1001`	`0b010`

MSR S3_0_C15_C9_2, <Xt>

op0	op1	CRn	CRm	op2
`0b11`	`0b000`	`0b1111`	`0b1001`	`0b010`

Accessibility

MRS <Xt>, S3_0_C15_C9_2

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERCFR2_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERCFR2_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERCFR2_EL1;
```

MSR S3_0_C15_C9_2, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        IMP_CLUSTERCFR2_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    IMP_CLUSTERCFR2_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERCFR2_EL1 = X[t, 64];
```

A.2.19 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 421 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-32: AArch64_imp_clusterl3upth2_el1 bit assignments

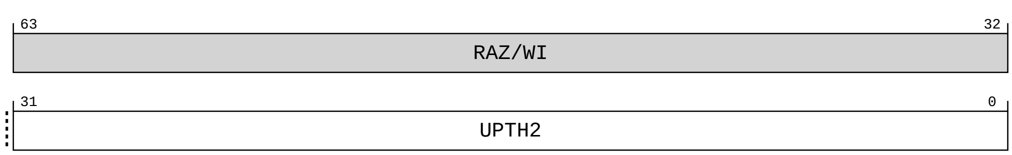


Table A-96: IMP_CLUSTERL3UPTH2_EL1 bit descriptions

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;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        X[t, 64] = IMP_CLUSTERL3UPTH2_EL1;
elsif PSTATE.EL == EL2 then
    X[t, 64] = IMP_CLUSTERL3UPTH2_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = 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);
    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_CLUSTERL3UPTH2_EL1 = X[t, 64];
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_CLUSTERL3UPTH2_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERL3UPTH2_EL1 = X[t, 64];
```

A.2.20 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-33: AArch64_imp_clustercdbg_el3 bit assignments

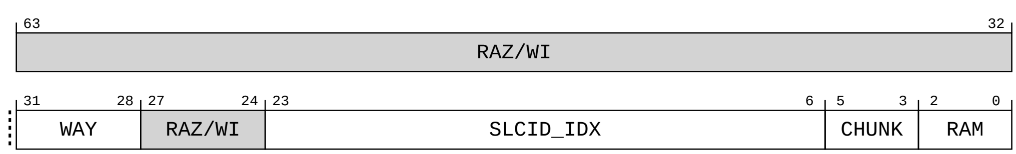


Table A-99: IMP_CLUSTERCDBG_EL3 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[31:28]	WAY	Way of RAM being accessed.	0b0000
[27:24]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[23:6]	SLCID_IDX	The L3 cache Set locations in each cache slice are all power-of-2 in size and therefore can be identified using contiguous index locations. The Set index values for slice 0 start from value zero in this field, followed by the index locations for slice 1, then slice 2, and so on. The total index width varies depending on the size of the RAM being accessed. The cache slice identification number, Slice ID, forms the upper used bits of the cache location encoding in this field. For a Tag RAM or Data RAM access this field will encode as {'0, SLICE_ID_W, TagRAM_IDX_W} For a Snoop Filter RAM access this field will encode as {'0, SLICE_ID_W, SFRAM_IDX_W}.	`0b000000000000000000`
[5:3]	CHUNK	Select of 64-bit data chunk to read from 512-bit Data RAM cache line. Only used when accessing Data RAM data. **`0b000`** Data[63:0] **`0b001`** Data[127:64] **`0b010`** Data[191:128] **`0b011`** Data[255:192] **`0b100`** Data[319:256] **`0b101`** Data[383:320] **`0b110`** Data[447:384] **`0b111`** Data[511:448]	`0b000`
[2:0]	RAM	RAM to be accessed. All other values are reserved. **`0b001`** Snoop Filter RAM, including Snoop Filter ECC **`0b010`** L3 Tag RAM **`0b011`** L3 Data RAM - accessing cacheline data **`0b111`** L3 Data RAM - accessing cacheline MTE tags, and ECC of both L3 Tag and L3 Data RAMs	`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
    X[t, 64] = IMP_CLUSTERCDBG_EL3;
```

MSR S3_6_C15_C4_7, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERCDBG_EL3 = X[t, 64];
```

A.2.21 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

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |     |  |
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-34: AArch64_imp_clusterpmmdcr_el3 bit assignments

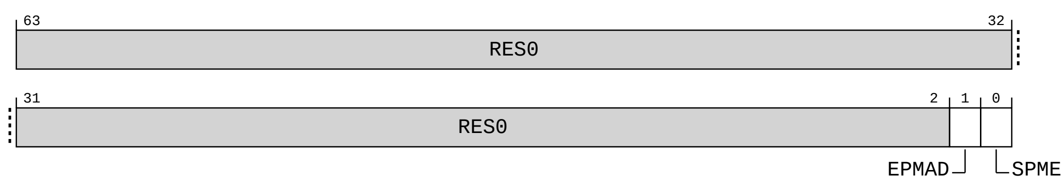


Table A-102: IMP_CLUSTERPMMDCR_EL3 bit descriptions

Bits	Name	Description	Reset
[63:2]	**RES0**	Reserved	**RES0**
[1]	EPMAD	External Performance Monitors Non-secure Access Disable. Controls Non-secure access to Performance Monitor registers by an external debugger. **`0b0`** Non-secure access to Performance Monitor registers from external debugger is permitted. **`0b1`** Non-secure access to Performance Monitor registers from external debugger is not permitted.	`0b0`
[0]	SPME	Secure Performance Monitors enable. This allows event counting in Secure state. **`0b0`** Event counting prohibited in Secure state. **`0b1`** Event counting in Secure state not affected by this bit.	`0b0`

Access

MRS <Xt>, S3_6_C15_C6_3

op0	op1	CRn	CRm	op2
`0b11`	`0b110`	`0b1111`	`0b0110`	`0b011`

MSR S3_6_C15_C6_3, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b110	0b1111	0b0110	0b011

Accessibility

MRS <Xt>, S3_6_C15_C6_3

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    X[t, 64] = IMP_CLUSTERPMMDCR_EL3;
```

MSR S3_6_C15_C6_3, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if EL2Enabled() && HCR_EL2.TIDCP == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    else
        UNDEFINED;
elsif PSTATE.EL == EL2 then
    UNDEFINED;
elsif PSTATE.EL == EL3 then
    IMP_CLUSTERPMMDCR_EL3 = X[t, 64];
```

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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**. Therefore, if the C1-DSU is enabled for Realm Management Extension (RME), none of these registers are present.
- 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	Present in Direct connect
ERXFR_EL1	3	0	C5	C4	0	See individual bit resets.	64-bit	Selected Error Record Feature Register	No
ERXCTLR_EL1	3	0	C5	C4	1	See individual bit resets.	64-bit	Selected Error Record Control Register	No
ERXSTATUS_EL1	3	0	C5	C4	2	See individual bit resets.	64-bit	Selected Error Record Primary Status Register	No
ERXPFGF_EL1	3	0	C5	C4	4	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Feature Register	No
ERXPFGCTL_EL1	3	0	C5	C4	5	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Control Register	No
ERXPFGCDN_EL1	3	0	C5	C4	6	See individual bit resets.	64-bit	Selected Pseudo-fault Generation Countdown Register	No
ERXMISC0_EL1	3	0	C5	C5	0	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 0	No
ERXMISC1_EL1	3	0	C5	C5	1	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 1	No
ERXMISC2_EL1	3	0	C5	C5	2	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 2	No
ERXMISC3_EL1	3	0	C5	C5	3	See individual bit resets.	64-bit	Selected Error Record Miscellaneous Register 3	No

A.3.1 ERXFR_EL1, Selected Error Record Feature Register

Accesses ext-CLUSTERRAS_ERR0FR when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00 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

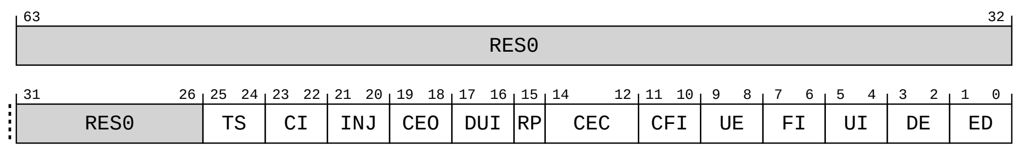


Table A-106: ERXFR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:26]	**RES0**	Reserved	**RES0**
[25:24]	TS	Timestamp Extension. **0b00** 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. **0b10** Critical error interrupt is supported and it can be enabled using associated controls. All other values are reserved.	0b10
[21:20]	INJ	Fault Injection Extension. Indicates whether the RAS Common Fault Injection Model Extension is implemented. **0b00** The node does not implement the RAS Common Fault Injection Model Extension. **0b01** The node implements the RAS Common Fault Injection Model Extension. See ext-CLUSTERRAS_ERR0PFGF 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. **0b00** Counts Corrected errors. Keeps the previous error syndrome. If the counter overflows then CLUSTERRAS_ERR0STATUS.OF is set to 1. All other values are reserved.	0b00

Bits	Name	Description	Reset
[17:16]	DUI	Error recovery interrupt for deferred errors. Indicates whether the node implements a control for enabling error recovery interrupts on deferred errors. **0b00** Does not support feature. ext-CLUSTERRAS_ERR0CTLR.DUI is **RES0**. All other values are reserved.	0b00
[15]	RP	Repeat counter. Indicates whether the node implements a repeat Corrected error counter in CLUSTERRAS_ERR0MISC0 for each error record <m> owned by the node that implements a standard Corrected error counter. **0b1** A first (repeat) counter and a second (other) counter are implemented. The repeat counter is the same size as the primary error counter.	0b1
[14:12]	CEC	Corrected Error Counter. Indicates whether the node implements standard Corrected error counter (CE counter) mechanisms in CLUSTERRAS_ERR0MISC0 for each error record <m> owned by the node that can record countable errors. **0b010** Implements an 8-bit Corrected error counter in CLUSTERRAS_ERR0MISC0[39:32]. All other values are reserved.	0b010
[11:10]	CFI	Fault handling interrupt for corrected errors. Indicates whether the node implements a control for enabling fault handling interrupts on corrected errors. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.CFI. All other values are reserved.	0b10
[9:8]	UE	In-band uncorrected error reporting. Indicates whether the node implements in-band uncorrected error reporting (External aborts), and, if so, whether the node implements controls for enabling and disabling the reporting. **0b01** Feature always enabled. ext-CLUSTERRAS_ERR0CTLR.UE is **RES0**.	0b01
[7:6]	FI	Fault handling interrupt. Indicates whether the node implements a fault handling interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.FI.	0b10
[5:4]	UI	Error recovery interrupt for uncorrected errors. Indicates whether the node implements an error recovery interrupt, and, if so, whether the node implements controls for enabling and disabling the interrupt. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.UI.	0b10

Bits	Name	Description	Reset
[3:2]	DE	Deferred error enable. **0b01** Deferred errors is always enabled.	0b01
[1:0]	ED	Error reporting and logging. Indicates this is the first record owned by the cluster. The cluster implements controls for enabling and disabling error reporting and logging. **0b10** Feature is controllable using ext-CLUSTERRAS_ERR0CTLR.ED. The value 0b11 is reserved.	0b10

Access

MRS <Xt>, ERXFR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b000

Accessibility

MRS <Xt>, ERXFR_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXFR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = 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
        X[t, 64] = ERXFR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXFR_EL1;
```

A.3.2 ERXCTLR_EL1, Selected Error Record Control Register

Accesses ext-CLUSTERRAS_ERR0CTLR when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0 xx00
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-36: AArch64_erxctlr_el1 bit assignments

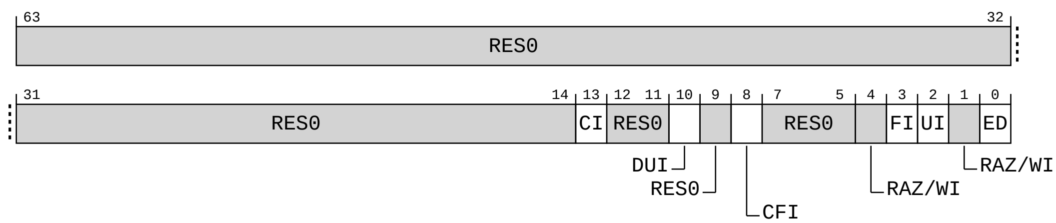


Table A-108: ERXCTLR_EL1 bit descriptions

Bits	Name	Description	Reset
[63:14]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[13]	CI	Critical error interrupt enable. When enabled, the critical error interrupt is generated for a critical error condition. **0b0** Critical error interrupt not generated for critical errors. Critical errors are treated as Uncontained errors. **0b1** Critical error interrupt generated for critical errors.	x
[12:11]	**RES0**	Reserved	**RES0**
[10]	DUI	Error recovery interrupt for deferred errors enable. This control applies to errors arising from both reads and writes. When enabled, an error recovery interrupt is generated for all detected Deferred errors. **0b0** Error recovery interrupt not generated for deferred errors. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error. Access to this field is: RO	x
[9]	**RES0**	Reserved	**RES0**
[8]	CFI	Fault handling interrupt for Corrected errors enable. This control applies to errors arising from both reads and writes. When enabled, the fault handling interrupt is generated when a counter overflows and the overflow bit for the counter is set to 1. For more information, see ext-CLUSTERRAS_ERR<n>MISC0. **0b0** Fault handling interrupt not generated for Corrected errors. **0b1** Fault handling interrupt generated for Corrected errors. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[7:5]	**RES0**	Reserved	**RES0**
[4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	FI	Fault handling interrupt enable. This control applies to errors arising from both reads and writes. When enabled, the fault handling interrupt is generated for all detected Corrected errors, Deferred errors,and Uncorrected errors. **0b0** Fault handling interrupt disabled. **0b1** Fault handling interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x

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. **0b0** Error recovery interrupt disabled. **0b1** Error recovery interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	x
[1]	RAZ/WI	Reserved	RAZ/WI
[0]	ED	Error reporting and logging enable. When disabled, the node behaves as if error detection and correction are disabled, and no errors are recorded or signaled by the node. **0b0** Error reporting disabled. **0b1** Error reporting enabled.	0b0

Access

MRS <Xt>, ERXCTLR_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b001

MSR ERXCTLR_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b001

Accessibility

MRS <Xt>, ERXCTLR_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXCTLR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
```

```
        X[t, 64] = ERXCTLR_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXCTLR_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXCTLR_EL1;
```

MSR ERXCTLR_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXCTLR_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXCTLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXCTLR_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    ERXCTLR_EL1 = X[t, 64];
```

A.3.3 ERXSTATUS_EL1, Selected Error Record Primary Status Register

Accesses ext-CLUSTERRAS_ERR0STATUS when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

AArch64 register ERXSTATUS_EL1 bits [63:0] are architecturally mapped to External System register B.1.3.3 CLUSTERRAS_ERR0STATUS, Error Record <n> Primary Status Register on page 467 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

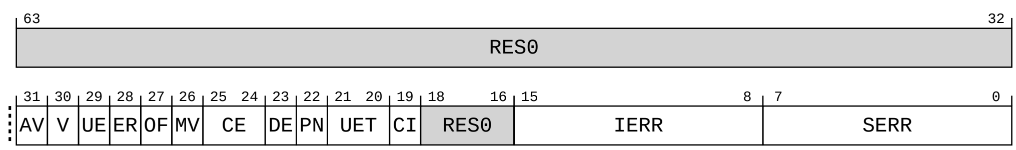


Table A-111: ERXSTATUS_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	AV	Address Valid. **0b0** ext-CLUSTERRAS_ERR0ADDR not valid. This bit is unimplemented and treated as **RAZ/WI**.	0b0
[30]	V	Status Register Valid. **0b0** CLUSTERRAS_ERR0STATUS not valid. **0b1** CLUSTERRAS_ERR0STATUS valid. At least one error has been recorded. This bit is read/write-one-to-clear.	0b0

Bits	Name	Description	Reset
[29]	UE	Uncorrected error. **0b0** No errors have been detected, or all detected errors have been either corrected or deferred. **0b1** At least one detected error was not corrected and not deferred. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write one to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	`0b0`
[28]	ER	Error Reported. **0b0** No in-band error (External abort) reported. This bit is unimplemented and treated as **RAZ/WI**.	`0b0`
[27]	OF	Overflow. Indicates that multiple errors have been detected. This bit is set to 1 when one of the following occurs: • A Corrected error is counted and the counter overflows. • CLUSTERRAS_ERR0STATUS.V was previously set to 1 and a type of error other than a Corrected error is recorded. Otherwise, this bit is unchanged when an error is recorded. A direct write that modifies the counter overflow flag indirectly might set this bit to an **UNKNOWN** value. A direct write to this bit that clears this bit to zero might indirectly set the counter overflow flag to an **UNKNOWN** value. **0b0** Since this bit was last cleared to zero, no error syndrome has been discarded and, if a Corrected error counter is implemented, it has not overflowed. **0b1** Since this bit was last cleared to zero, at least one error syndrome has been discarded or, if a Corrected error counter is implemented, it might have overflowed. If this bit is nonzero, then software must write 1 to this bit, to clear this bit to zero, when clearing CLUSTERRAS_ERR0STATUS.V to 0. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	`0b0`

Bits	Name	Description	Reset
[26]	MV	Miscellaneous Registers (CLUSTERRAS_ERR0MISC0) Valid. **0b0** CLUSTERRAS_ERR0MISC0 is not valid. **0b1** The contents of CLUSTERRAS_ERR0MISC0 contains additional information for an error recorded by this record. Only CLUSTERRAS_ERR0MISC0 is implemented. CLUSTERRAS_ERR0MISC1,2,3 are treated as **RAZ/WI**. This bit is read/write-one-to-clear.	0b0
[25:24]	CE	Corrected Error. **0b00** No errors were corrected. **0b10** At least one error was corrected. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this field is nonzero, then software must write ones to this field to clear this field to zero. If CLUSTERRAS_ERR0STATUS.V is set to 0, this field is not valid and reads **UNKNOWN**. This field is read/write-one-to-clear. Writing a value other than all-zeros or all-ones sets this field to an **UNKNOWN** value.	0b00
[23]	DE	Deferred Error. **0b0** No errors were deferred. **0b1** At least one error was not corrected and deferred. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. If CLUSTERRAS_ERR0STATUS.V is set to 0, this bit is not valid and reads **UNKNOWN**. This bit is read/write-one-to-clear.	0b0

Bits	Name	Description	Reset
[22]	PN	Poison. **0b0** Uncorrected error or Deferred error recorded because a corrupt value was detected. **0b1** Uncorrected error or Deferred error recorded because a poison value was detected. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if any of the following are true: • CLUSTERRAS_ERR0STATUS.V is set to 0. • CLUSTERRAS_ERR0STATUS.{DE, UE} are both set to 0. This bit is read/write-one-to-clear.	0b0
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. **0b00** Uncorrected error, Uncontainable error (UC). This field is not implemented and is treated as **RAZ/WI**.	0b00
[19]	CI	Critical error. Indicates whether a critical error condition has been recorded. **0b0** No critical error condition recorded. **0b1** Critical error condition recorded. When clearing CLUSTERRAS_ERR0STATUS.V to 0, if this bit is nonzero, then software must write 1 to this bit to clear this bit to zero. This bit is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0. This bit is read/write-one-to-clear.	0b0
[18:16]	**RES0**	Reserved	**RES0**
[15:8]	IERR	**IMPLEMENTATION DEFINED** Extended error code. Used with any primary error code SERR value. Additional information is placed in the CLUSTERRAS_ERR0MISC0 register. **0b00000000** If SERR == 0x7, indicates a Tag RAM error. Not used with other SERR values. **0b00000010** If SERR == 0x7, indicates a Snoop Filter RAM error. Not used with other SERR values. This field is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0.	0x00

Bits	Name	Description	Reset
[7:0]	SERR	Primary error code. Indicates the type of Primary error. **0b00000000** No error. **0b00000001** IMPLEMENTATION DEFINED error. **0b00000010** Data value from (non-associative) internal memory. For example, ECC from on-chip SRAM or buffer. **0b00000011** IMPLEMENTATION DEFINED pin. For example, nSEI pin. **0b00000100** Assertion failure. For example, consistency failure. **0b00000101** Error detected on internal data path. For example, parity on ALU result. **0b00000110** Data value from associative memory. For example, ECC error on cache data. **0b00000111** Address/control value from associative memory. For example, ECC error on cache tag. **0b00001000** Data value from a TLB. For example, ECC error on TLB data. **0b00001001** Address/control value from a TLB. For example, ECC error on TLB tag.	0x00

Bits	Name	Description	Reset
[7:0] continued	SERR	**0b00001010** Data value from producer. For example, parity error on write data bus. **0b00001011** Address/control value from producer. For example, parity error on address bus. **0b00001100** Data value from (non-associative) external memory. For example, ECC error in SDRAM. **0b00001101** Illegal address (software fault). For example, access to unpopulated memory. **0b00001110** Illegal access (software fault). For example, byte write to word register. **0b00001111** Illegal state (software fault). For example, device not ready. **0b00010000** Internal data register. For example, parity on a SIMD&FP register. For a PE, all general-purpose, stack pointer, SIMD&FP, and SVE registers are data registers. **0b00010001** Internal control register. For example, Parity on a System register. For a PE, all registers other than general-purpose, stack pointer, SIMD&FP, and SVE registers are control registers. **0b00010010** Error response from subordinate. For example, error response from cache write-back. **0b00010011** External timeout. For example, timeout on interaction with another node.	0x00
[7:0] continued	SERR	**0b00010100** Internal timeout. For example, timeout on interface within the node. **0b00010101** Deferred error from subordinate not supported at requester. For example, poisoned data received from a subordinate by a requester that cannot defer the error further. All other values are reserved. This field is not valid and reads **UNKNOWN** if CLUSTERRAS_ERR0STATUS.V is set to 0.	0x00

Access

MRS <Xt>, ERXSTATUS_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b010

MSR ERXSTATUS_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b010

Accessibility

MRS <Xt>, ERXSTATUS_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXSTATUS_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXSTATUS_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXSTATUS_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXSTATUS_EL1;
```

MSR ERXSTATUS_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXSTATUS_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXSTATUS_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXSTATUS_EL1 = X[t, 64];
```

```
elsif PSTATE.EL == EL3 then
    ERXSTATUS_EL1 = X[t, 64];
```

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, Error Record <n> Pseudo-fault Generation Feature Register on page 493 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 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure A-38: AArch64_erxpfgf_el1 bit assignments

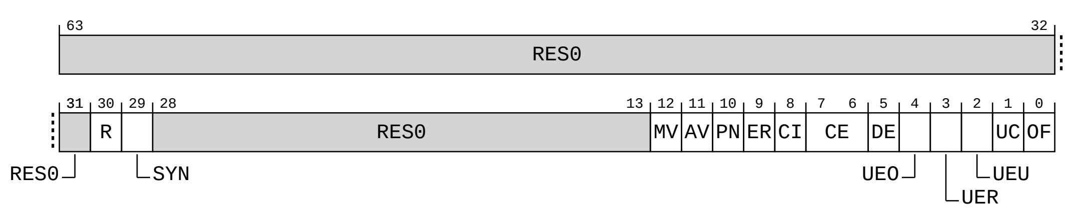


Table A-114: ERXPFGF_EL1 bit descriptions

Bits	Name	Description	Reset
[63:31]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[30]	R	Restartable. Support for Error Generation Counter restart mode. **0b1** Feature controllable.	0b1
[29]	SYN	Syndrome. Fault syndrome injection. **0b1** When an injected error is recorded, the node does not update the ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} fields. ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} are writable when ext-CLUSTERRAS_ERR0STATUS.V == 0. **Note:** Software can write intended values into the ext-CLUSTERRAS_ERR0STATUS.{IERR, SERR} fields when setting up a fault injection event.	0b1
[28:13]	**RES0**	Reserved	**RES0**
[12]	MV	Miscellaneous syndrome. Additional syndrome injection. Defines whether software can control all or part of the syndrome recorded in the CLUSTERRAS_ERR0MISC0 register when an injected error is recorded. CLUSTERRAS_ERR0MISC1-3 registers are reserved and unused for this purpose. **0b1** When an injected error is recorded, the node does not update all the syndrome fields in CLUSTERRAS_ERR0MISC0. The node records syndrome in CLUSTERRAS_ERR0MISC0 OFO, CECO, OFR, CECR, WAY, INDX, LVL, and IND fields and sets ext-CLUSTERRAS_ERR0STATUS.MV to 1. CLUSTERRAS_ERR0PGFCTL.MV is **RAO**. **Note:** Software can write intended values into the CLUSTERRAS_ERR0MISC0 register when setting up a fault injection event.	0b1
[11]	AV	Address syndrome. Address syndrome injection. Always **RAZ/WI**. **0b0** The node does not support ext-CLUSTERRAS_ERR0ADDR and does not set ext-CLUSTERRAS_ERR0STATUS.AV.	0b0
[10]	PN	Poison flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.PN status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.PN is set to ext-CLUSTERRAS_ERR0PFGCTL.PN.	0b1
[9]	ER	Error Reported flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.ER status flag. **0b0** When an injected error is recorded, the node does not set ext-CLUSTERRAS_ERR0STATUS.ER. This bit reads-as-zero.	0b0

Bits	Name	Description	Reset
[8]	CI	Critical Error flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.CI status flag. **`0b1`** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.CI is set to ext-CLUSTERRAS_ERR0PFGCTL.CI.	`0b1`
[7:6]	CE	Corrected Error generation. Describes the types of Corrected Error that the fault generation feature of the node can generate. **`0b01`** The fault generation feature of the node allows generation of a non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERR0STATUS.CE == 0b10.	`0b01`
[5]	DE	Deferred Error generation. Describes whether the fault generation feature of the node can generate this type of error. **`0b1`** The fault generation feature of the node allows generation of this type of error.	`0b1`
[4]	UEO	Latent or Restartable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **`0b0`** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	`0b0`
[3]	UER	Signaled or Recoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **`0b0`** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	`0b0`
[2]	UEU	Unrecoverable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **`0b0`** The fault generation feature of the node cannot generate this type of error. This bit reads-as-zero.	`0b0`
[1]	UC	Uncontainable Error generation. Describes whether the fault generation feature of the node can generate this type of error. **`0b1`** The fault generation feature of the node allows generation of this type of error.	`0b1`
[0]	OF	Overflow flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR0STATUS.OF status flag. **`0b1`** When an injected error is recorded, ext-CLUSTERRAS_ERR0STATUS.OF is set to ext-CLUSTERRAS_ERR0PFGCTL.OF.	`0b1`

Access

MRS <Xt>, ERXPFGF_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b100

Accessibility

MRS <Xt>, ERXPFGF_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXPFGF_EL1 == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXPFGF_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXPFGF_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXPFGF_EL1;
```

A.3.5 ERXPFGCTL_EL1, Selected Pseudo-fault Generation Control Register

Accesses the ext-CLUSTERRAS_ERR0PFGCTL register when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Configurations

AArch64 register ERXPFGCTL_EL1 bits [63:0] are architecturally mapped to External System register B.1.3.10 CLUSTERRAS_ERR0PFGCTL, Error Record <n> Pseudo-fault Generation Control Register on page 498 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
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-39: AArch64_erxpfgctl_el1 bit assignments

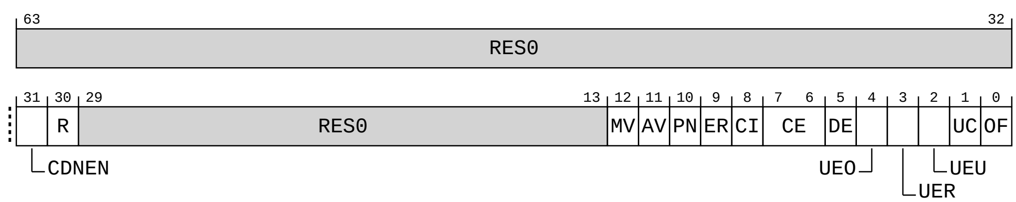


Table A-116: ERXPFGCTL_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	CDNEN	Countdown Enable. Controls transfers from the value that is held in the ext-CLUSTERRAS_ERR0PFGCDN into the Error Generation Counter, and enables this counter. **`0b0`** The Error Generation Counter is disabled. **`0b1`** The Error Generation Counter is enabled. On a write of 1 to this bit, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.	`0b0`
[30]	R	Restart. Controls whether, on reaching zero, the Error Generation Counter restarts from the ext-CLUSTERRAS_ERR0PFGCDN value, or stops. **`0b0`** On reaching 0, the Error Generation Counter stops. **`0b1`** On reaching 0, the Error Generation Counter is set to ext-CLUSTERRAS_ERR0PFGCDN.CDN.	`x`
[29:13]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset

[12]	MV	Miscellaneous syndrome. The value that is written to ext-CLUSTERRAS_ERR0STATUS.MV when an injected error is recorded.
		`0b0`
		ext-CLUSTERRAS_ERR0STATUS.MV is set to 0 when an injected error is recorded.
		`0b1`
		ext-CLUSTERRAS_ERR0STATUS.MV is set to 1 when an injected error is recorded.
			`x`

[11]	AV	Address syndrome. The value that is written to ext-CLUSTERRAS_ERR0STATUS.AV when an injected error is recorded. `0b0` ext-CLUSTERRAS_ERR0STATUS.AV is set to 0 when an injected error is recorded. This bit is **RES0**. Access to this field is: **RES0**	`x`

[10]	PN	Poison flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.PN when an injected error is recorded.
		`0b0`
		ext-CLUSTERRAS_ERR0STATUS.PN is set to 0 when an injected error is recorded.
		`0b1`
		ext-CLUSTERRAS_ERR0STATUS.PN is set to 1 when an injected error is recorded.
			`x`

[9]	ER	`0b0` ext-CLUSTERRAS_ERR0STATUS.ER is set to 0 when an injected error is recorded. This bit is **RES0**. Access to this field is: **RES0**	`x`

[8]	CI	Critical Error flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.CI when an injected error is recorded.
		`0b0`
		ext-CLUSTERRAS_ERR0STATUS.CI is set to 0 when an injected error is recorded.
		`0b1`
		ext-CLUSTERRAS_ERR0STATUS.CI is set to 1 when an injected error is recorded.
			`x`

[7:6]	CE	Corrected Error generation enable. Controls the type of Corrected Error condition that might be generated.
		`0b00`
		No error of this type is generated.
		`0b01`
		A non-specific Corrected Error, that is, a Corrected Error that is recorded as ext-CLUSTERRAS_ERR0STATUS.CE == 0b10, might be generated when the Error Generation Counter decrements to zero.
		The set of permitted values for this field is defined by ext-CLUSTERRAS_ERR0PFGF.CE.
			`xx`

[5]	DE	Deferred Error generation enable. Controls whether this type of error condition might be generated.
		`0b0`
		No error of this type is generated.
		`0b1`
		An error of this type might be generated when the Error Generation Counter decrements to zero.
			`x`

Bits	Name	Description	Reset
[4]	UEO	Latent or Restartable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[3]	UER	Signaled or Recoverable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[2]	UEU	Unrecoverable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. This bit is **RES0**. Access to this field is: **RES0**	x
[1]	UC	Uncontainable Error generation enable. Controls whether this type of error condition might be generated. **0b0** No error of this type is generated. **0b1** An error of this type might be generated when the Error Generation Counter decrements to zero.	x
[0]	OF	Overflow flag. The value that is written to ext-CLUSTERRAS_ERR0STATUS.OF when an injected error is recorded. **0b0** ext-CLUSTERRAS_ERR0STATUS.OF is set to 0 when an injected error is recorded. **0b1** ext-CLUSTERRAS_ERR0STATUS.OF is set to 1 when an injected error is recorded.	x

Access

MRS <Xt>, ERXPFGCTL_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b101

MSR ERXPFGCTL_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b101

Accessibility

MRS <Xt>, ERXPFGCTL_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXPFGCTL_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXPFGCTL_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXPFGCTL_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXPFGCTL_EL1;
```

MSR ERXPFGCTL_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXPFGCTL_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCTL_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCTL_EL1 = X[t, 64];
```

```
elsif PSTATE.EL == EL3 then
    ERXPFGCTL_EL1 = X[t, 64];
```

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, Error Record <n> Pseudo-fault Generation Countdown Register on page 504 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 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-40: AArch64_erxpfgcdn_el1 bit assignments

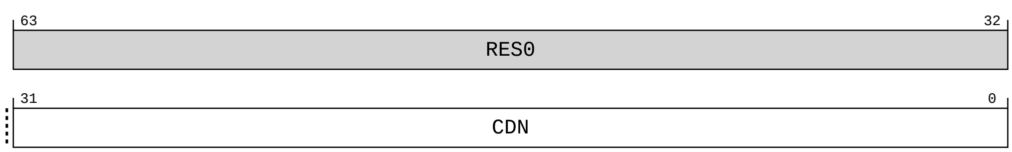


Table A-119: ERXPFGCDN_EL1 bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[31:0]	CDN	Countdown value. This field is copied to Error Generation Counter when either: • Software writes ext-CLUSTERRAS_ERR0PFGCTL.CDNEN with 1. • The Error Generation Counter decrements to zero and ext-CLUSTERRAS_ERR0PFGCTL.R == 1. While ext-CLUSTERRAS_ERR0PFGCTL.CDNEN == 1 and the Error Generation Counter is nonzero, the counter decrements by 1 for each cycle. When the counter reaches 0, one of the errors enabled in the ext-CLUSTERRAS_ERR0PFGCTL register is generated. **Note:** The current Error Generation Counter value is not visible to software.	32{x}

Access

MRS <Xt>, ERXPFGCDN_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b110

MSR ERXPFGCDN_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0100	0b110

Accessibility

MRS <Xt>, ERXPFGCDN_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXPFGCDN_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXPFGCDN_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
```

```
    else
        X[t, 64] = ERXPFGCDN_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXPFGCDN_EL1;
```

MSR ERXPFGCDN_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.FIEN == '0' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXPFGCDN_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCDN_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.FIEN == '0' then
        UNDEFINED;
    elsif SCR_EL3.FIEN == '0' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXPFGCDN_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    ERXPFGCDN_EL1 = X[t, 64];
```

A.3.7 ERXMISC0_EL1, Selected Error Record Miscellaneous Register 0

Accesses ext-CLUSTERRAS_ERR0MISC0 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Miscellaneous error syndrome register. The Miscellaneous error syndrome register contains:

- 2 architecturally-defined Corrected error counters with sticky overflow bits,
- Information to identify the FRU in which the error was detected, including Index, Way, Level, Instruction vs. Data fields.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

```
0000 0000 0000 0000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |     |  |
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-41: AArch64_erxmisc0_el1 bit assignments

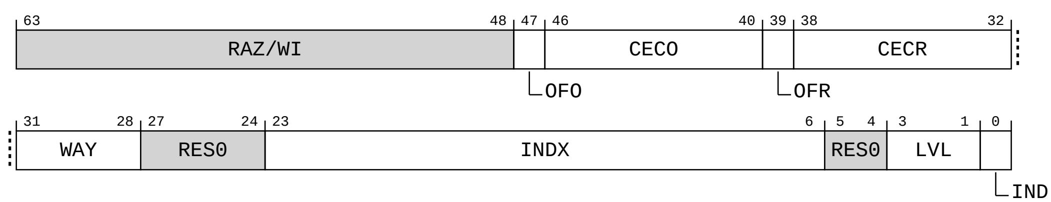


Table A-122: ERXMISC0_EL1 bit descriptions

Bits	Name	Description	Reset
[63:48]	RAZ/WI	Reserved	RAZ/WI
[47]	OFO	Sticky overflow bit for Other errors. Set to 1 when the Corrected error count Other (CECO) field is incremented and wraps through zero. **0b0** Other counter has not overflowed. **0b1** Other counter has overflowed. A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an UNKNOWN value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an UNKNOWN value.	x
[46:40]	CECO	Corrected error count for Other errors. The Other error counter increments for all Corrected errors that are not counted by the CECR Repeat error counter due to the syndrome of the new error mismatching against the recorded syndrome of the first Repeat error. Refer to the CECR Repeat error description for fields used to match syndrome. At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.	7{x}

Bits	Name	Description	Reset
[39]	OFR	Sticky overflow bit for Repeat errors. Set to 1 when the Corrected error count Repeat (CECR) field is incremented and wraps through zero. **0b0** Repeat counter has not overflowed. **0b1** Repeat counter has overflowed. A direct write that modifies this bit might indirectly set ext-CLUSTERRAS_ERR0STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR0STATUS.OF that clears it to zero might indirectly set this bit to an **UNKNOWN** value.	x
[38:32]	CECR	Corrected error count for Repeat errors. The Repeat error counter increments for the first Corrected error and records the syndrome for the error in the fields described below. It also increments for each subsequent Corrected error with a syndrome matching the first error's recorded syndrome, otherwise the error causes an increment to the CECO Other counter. The syndrome is recorded in the following fields: • ext-CLUSTERRAS_ERR0STATUS.IERR • ext-CLUSTERRAS_ERR0STATUS.SERR • ext-CLUSTERRAS_ERR0MISC0.INDX • ext-CLUSTERRAS_ERR0MISC0.WAY The syndrome is matched on a new Corrected error if all of the following are true: • ext-CLUSTERRAS_ERR0STATUS.MV bit is set, • ext-CLUSTERRAS_ERR0STATUS.IERR matches the new error, • ext-CLUSTERRAS_ERR0STATUS.SERR matches the new error, • ext-CLUSTERRAS_ERR0MISC0.INDX matches the new error, • ext-CLUSTERRAS_ERR0MISC0.WAY matches the new error. CLUSTERRAS_ERR0STATUS.MV indicates the validity of the INDX and WAY fields of the CLUSTERRAS_ERR0MISC0 register At most 1 error can be counted per clock cycle even if there are multiple Corrected errors and/or sources.	7{x}
[31:28]	WAY	L3 Cache Way that contained the error.	xxxx
[27:24]	**RES0**	Reserved	**RES0**
[23:6]	INDX	L3 Cache Index that contained the error.	18{x}
[5:4]	**RES0**	Reserved	**RES0**
[3:1]	LVL	L3 Cache Level that contained the error. Always `0x2`. **0b010** Level 3 cache.	xxx
[0]	IND	L3 Cache instruction vs. data cache that contained the error. Always data (`0x0`). **0b0** Data cache error.	x

Access

MRS <Xt>, ERXMISC0_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b000

MSR ERXMISC0_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b000

Accessibility

MRS <Xt>, ERXMISC0_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXMISC0_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXMISC0_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXMISC0_EL1;
```

MSR ERXMISC0_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
```

```
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC0_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC0_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    ERXMISC0_EL1 = X[t, 64];
```

A.3.8 ERXMISC1_EL1, Selected Error Record Miscellaneous Register 1

Accesses ext-CLUSTERRAS_ERR0MISC1 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-42: AArch64_erxmisc1_el1 bit assignments

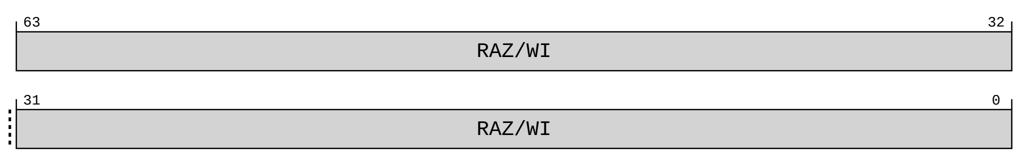


Table A-125: ERXMISC1_EL1 bit descriptions

Bits	Name	Description	Reset
[63:0]	RAZ/WI	Reserved	RAZ/WI

Access

MRS <Xt>, ERXMISC1_EL1

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b001

MSR ERXMISC1_EL1, <Xt>

op0	op1	CRn	CRm	op2
0b11	0b000	0b0101	0b0101	0b001

Accessibility

MRS <Xt>, ERXMISC1_EL1

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGRTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXMISC1_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXMISC1_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXMISC1_EL1;
```

MSR ERXMISC1_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
```

```
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC1_EL1 = X[t, 64];
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, 64];
elsif PSTATE.EL == EL3 then
    ERXMISC1_EL1 = X[t, 64];
```

A.3.9 ERXMISC2_EL1, Selected Error Record Miscellaneous Register 2

Accesses ext-CLUSTERRAS_ERR0MISC2 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-43: AArch64_erxmisc2_el1 bit assignments

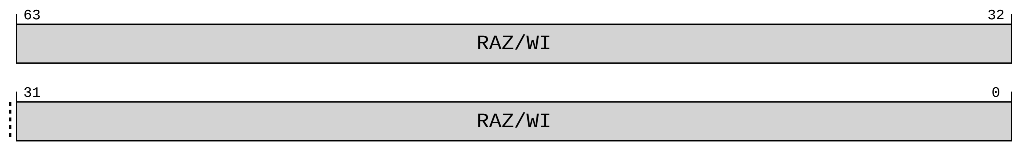


Table A-128: ERXMISC2_EL1 bit descriptions

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;
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
        X[t, 64] = ERXMISC2_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
```

```
    else
        X[t, 64] = ERXMISC2_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXMISC2_EL1;
```

MSR ERXMISC2_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC2_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC2_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    ERXMISC2_EL1 = X[t, 64];
```

A.3.10 ERXMISC3_EL1, Selected Error Record Miscellaneous Register 3

Accesses ext-CLUSTERRAS_ERR0MISC3 when the value in AArch64-ERRSELR_EL1.SEL is set to 0.

Unimplemented error syndrome register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Functional group

RAS registers

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure A-44: AArch64_erxmisc3_el1 bit assignments

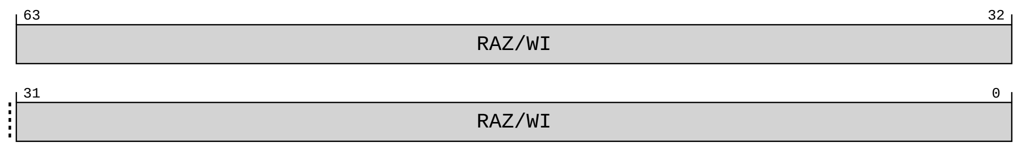


Table A-131: ERXMISC3_EL1 bit descriptions

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
        X[t, 64] = ERXMISC3_EL1;
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
```

```
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        X[t, 64] = ERXMISC3_EL1;
elsif PSTATE.EL == EL3 then
    X[t, 64] = ERXMISC3_EL1;
```

MSR ERXMISC3_EL1, <Xt>

```
if PSTATE.EL == EL0 then
    UNDEFINED;
elsif PSTATE.EL == EL1 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif EL2Enabled() && HCR_EL2.TERR == '1' then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif EL2Enabled() && SCR_EL3.FGTEn == '1' && HFGWTR_EL2.ERXMISCn_EL1 == '1'
 then
        AArch64.SystemAccessTrap(EL2, 0x18);
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC3_EL1 = X[t, 64];
elsif PSTATE.EL == EL2 then
    if Halted() && EDSCR.SDD == '1' && boolean IMPLEMENTATION_DEFINED "EL3 trap
 priority when SDD == '1'" && SCR_EL3.TERR == '1' then
        UNDEFINED;
    elsif SCR_EL3.TERR == '1' then
        if Halted() && EDSCR.SDD == '1' then
            UNDEFINED;
        else
            AArch64.SystemAccessTrap(EL3, 0x18);
    else
        ERXMISC3_EL1 = X[t, 64];
elsif PSTATE.EL == EL3 then
    ERXMISC3_EL1 = X[t, 64];
```

Appendix B External registers

This appendix contains the descriptions for all the external (memory-mapped) registers in the C1-DSU.

B.1 Registers accessed over the utility bus

This section contains the descriptions for all the external registers in the C1-DSU 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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If Realm Management Extension (RME) is enabled, only those registers that are available in Direct connect can be accessed. You must access those cluster system control registers from Root state. If RME is not enabled, you must access the cluster system control registers from the Secure state. For RME to be enabled, the cluster must be in Direct connect configuration and the LEGACYTZEN input signal is LOW, see 1.4.1 Realm management extension on page 26.
- The cluster power control registers are treated as **RAZ/WI** if either:
  - The register is marked as Reserved.
  - The register is accessed in the wrong Security state.
  - The register is not present in Direct connect.
- Any address that is not documented is treated as **RAZ/WI**.
- The base address for the cluster power control registers is `0x000000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-1: Cluster registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0000	CLUSTERIDR	See individual bit resets.	64-bit	Cluster Main Revision Register	Yes
0x0008	CLUSTERREVIDR	See individual bit resets.	64-bit	Cluster ECO ID Register	Yes
0x0010	CLUSTERPWRCTLR	See individual bit resets.	64-bit	Cluster Power Control Register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0028	CLUSTERL3DNTH0	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold0 Register	No
0x0030	CLUSTERL3DNTH1	See individual bit resets.	64-bit	Cluster L3 Downsize Threshold1 Register	No
0x0038	CLUSTERL3UPTH0	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold0 Register	No
0x0040	CLUSTERL3UPTH1	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold1 Register	No
0x0048	CLUSTERBUSQOS	See individual bit resets.	64-bit	Cluster Bus QoS Control Register	No
0x0050	CLUSTERCFR	See individual bit resets.	64-bit	Cluster Configuration Register	Yes
0x0058	CLUSTERACTLR	See individual bit resets.	64-bit	Cluster Auxiliary Control Register	Yes
0x0060	CLUSTERECTLR	See individual bit resets.	64-bit	Cluster Extended Control Register	No
0x0068	CLUSTERCFR2	See individual bit resets.	64-bit	Cluster Configuration Register 2	No
0x0080	CLUSTERPPMCR	See individual bit resets.	64-bit	Cluster PPM Control Register	No
0x0088	CLUSTERMPMMCR	See individual bit resets.	64-bit	Cluster MPMM Control Register	No
0x0090	CLUSTERL3UPTH2	See individual bit resets.	64-bit	Cluster L3 Upsize Threshold2 Register	No

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.2.2 IMP_CLUSTERIDR_EL1, Cluster Main Revision Register on page 307 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 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-1: ext_clusteridr bit assignments

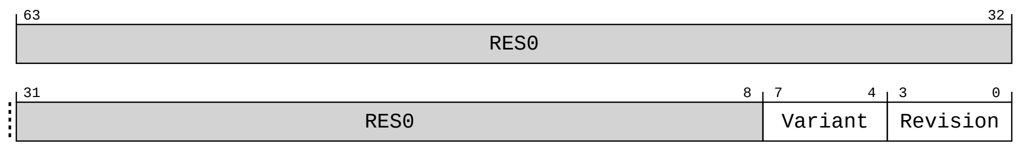


Table B-2: CLUSTERIDR bit descriptions

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. **0b0010** Cluster minor revision 2.	0b0010

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.2.3 IMP_CLUSTERREVIDR_EL1, Cluster ECO ID Register on page 309 bits [63:0].

Attributes

Width

64

Component

Cluster

Register offset

0x0008

Access type

RO

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-2: ext_clusterrevidr bit assignments

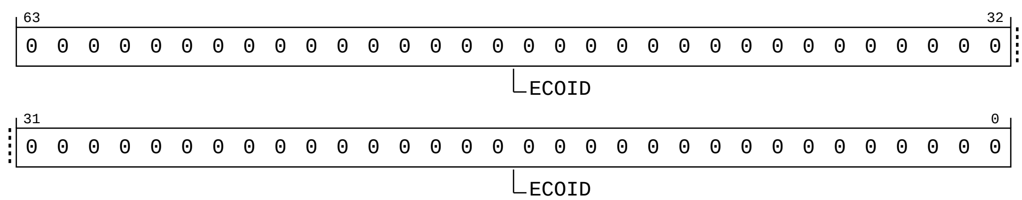


Table B-4: CLUSTERREVIDR bit descriptions

Bits	Name	Description	Reset
[63:0]	ECOID	Contains ECO information. Refer to the errata documentation for any bit allocations. `0b0000000000000000000000000000000000000000000000000000000000000000` Customer ECO ID	`0x0000000000000000`

Accessibility

Component	Offset	Instance	Range
Cluster	0x0008	CLUSTERREVIDR	None

This interface is accessible as follows:

RO

B.1.1.3 CLUSTERPWRCTLR, Cluster Power Control Register

This register controls power features of the cluster.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

Cluster

Register offset

0x0010

Access type

RW

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1011 0100 0000 0001 0111 0000

Bit descriptions

Figure B-3: ext_clusterpwrctlr bit assignments

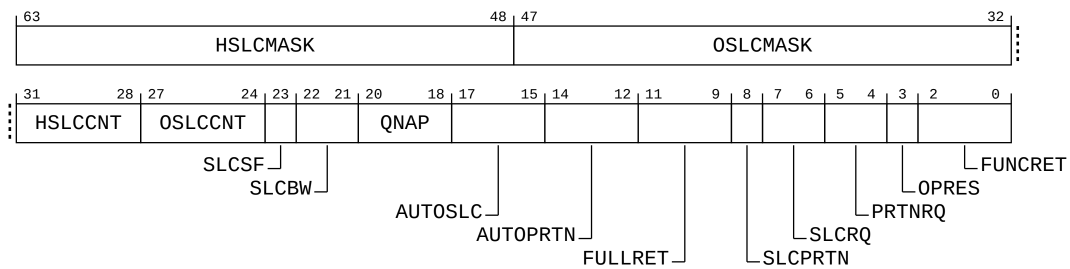


Table B-6: CLUSTERPWRCTLR bit descriptions

Bits	Name	Description	Reset
[63:48]	HSLCMASK	Half slice core mask. This contains one bit per core or SME2 unit, and if that bit is set then the core/SME2 unit is included in the count that is compared with the HSLCCNT field. Bits above the number of cores plus SME2 units implemented are **RAZ/WI**.	0x0000
[47:32]	OSLCMASK	One slice core mask. This contains one bit per core or SME2 unit, and if that bit is set then the core/SME2 unit is included in the count that is compared with the OSLCCNT field. Bits above the number of cores plus SME2 units implemented are **RAZ/WI**.	0x0000
[31:28]	HSLCCNT	Half slice core count. When AUTOSLC is non-zero, if the number of cores or SME2 units 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/SME2 unit 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 or SME2 units 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/SME2 unit 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	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. **`0b0`** Ignore the snoop filter in AUTOSLC decisions **`0b1`** AUTOSLC will ensure enough slices are powered on for the currently powered on cores	`0b1`
[22:21]	SLCBW	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. **`0b00`** Ignore the slice bandwidth in AUTOSLC decisions **`0b01`** AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 3% of the AUTOSLC time period **`0b10`** AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 6% of the AUTOSLC time period **`0b11`** AUTOSLC decisions are affected by high slice bandwidth that lasts for at least 12% of the AUTOSLC time period	`0b01`
[20:18]	QNAP	L3 data RAM quick nap enable time. Controls the number of SCLK cycles to wait after each L3 slice is empty before setting all the L3 data RAM quick nap pins for that slice. Note that sub-banks of the RAM can enter quick nap before this point based on the CLUSTERECTLR.SQNAP field. **`0b000`** Quick nap disabled **`0b001`** Enabled with 8 cycle timeout **`0b010`** Enabled with 12 cycle timeout **`0b011`** Enabled with 16 cycle timeout **`0b100`** Enabled with 24 cycle timeout **`0b101`** Enabled with 32 cycle timeout **`0b110`** Enabled with 64 cycle timeout **`0b111`** Enabled with 128 cycle timeout	`0b101`

Bits	Name	Description	Reset
[17:15]	AUTOSLC	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. **`0b000`** Disabled **`0b001`** 524,288 architectural timer ticks, time period of 524us **`0b010`** 1,048,576 architectural timer ticks, time period of 1ms **`0b011`** 2,097,152 architectural timer ticks, time period of 2.1ms **`0b100`** 4,194,304 architectural timer ticks, time period of 4.2ms **`0b101`** 8,388,608 architectural timer ticks, time period of 8.4ms **`0b110`** 16,777,216 architectural timer ticks, time period of 16.8ms **`0b111`** 33,554,432 architectural timer ticks, time period of 33.6ms	`0b000`
[14:12]	AUTOPRTN	Enable automatic RAM power down and configure evaluation time period. Note that a shorter time period allows better responsiveness to changing workloads, however if it is too short then the cost of frequent resizing can be too high. **`0b000`** Disabled **`0b001`** 524,288 architectural timer ticks, time period of 524us **`0b010`** 1,048,576 architectural timer ticks, time period of 1ms **`0b011`** 2,097,152 architectural timer ticks, time period of 2.1ms **`0b100`** 4,194,304 architectural timer ticks, time period of 4.2ms **`0b101`** 8,388,608 architectural timer ticks, time period of 8.4ms **`0b110`** 16,777,216 architectural timer ticks, time period of 16.8ms **`0b111`** 33,554,432 architectural timer ticks, time period of 33.6ms	`0b000`

Bits	Name	Description	Reset
[11:9]	FULLRET	Enable the FULL_RET slice powerdown mode and time period. Note that while this would typically be a longer period than the FUNCRET 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. **0b000** Disabled **0b001** 128 architectural timer ticks, time period of 128ns **0b010** 512 architectural timer ticks, time period of 512ns **0b011** 2,048 architectural timer ticks, time period of 2us **0b100** 4,096 architectural timer ticks, time period of 4.1us **0b101** 8,192 architectural timer ticks, time period of 8.2us **0b110** 16,384 architectural timer ticks, time period of 16.4us **0b111** 32,768 architectural timer ticks, time period of 32.8us	`0b000`
[8]	SLCPRTN	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 **0b0** AUTOSLC decisions should ignore the AUTOPRTN status **0b1** AUTOSLC decisions should include the AUTOPRTN status	`0b1`
[7:6]	SLCRQ	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. **0b00** Request that one L3 cache slice is powered on **0b01** Request that all L3 cache slices are powered on **0b10** Request that half the L3 cache slices are powered on	`0b01`
[5:4]	PRTNRQ	Cache portion power request. These bits are passed to the PPU as an advisory request for which portions to power. Note that these bits are only used when AUTOPRTN bits are 3'b000. **0b00** Request that none of the L3 cache portions in each slice is powered on **0b01** Request that half of the L3 cache portions in each slice are powered on **0b11** Request that both of the L3 cache portions in each slice are powered on	`0b11`

Bits	Name	Description	Reset
[3]	OPRES	Restore previous operating mode after cluster power off **0b0** Operating mode defaults to ALL RAM ALL SLICE when powering on the cluster **0b1** Enable restoring of previous operating mode	`0b0`
[2:0]	FUNCRET	L3 Data RAM retention control. **0b000** Disable the retention circuit. **0b001** 128 architectural timer ticks, time period of 128ns minimum delay before retention **0b010** 512 architectural timer ticks, time period of 512ns minimum delay before retention **0b011** 2,048 architectural timer ticks, time period of 2us minimum delay before retention **0b100** 4,096 architectural timer ticks, time period of 4.1us minimum delay before retention **0b101** 8,192 architectural timer ticks, time period of 8.2us minimum delay before retention **0b110** 16,384 architectural timer ticks, time period of 16.4us minimum delay before retention **0b111** 32,768 architectural timer ticks, time period of 32.8us minimum delay before retention	`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.2.9 IMP_CLUSTERL3DNTH0_EL1, Cluster L3 Downsize Threshold0 Register on page 328 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

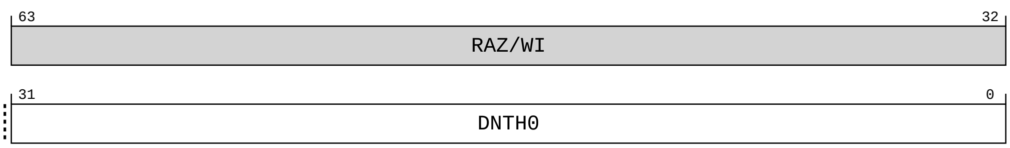


Table B-8: CLUSTERL3DNTH0 bit descriptions

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

Bit descriptions

Figure B-5: ext_clusterl3dnth1 bit assignments

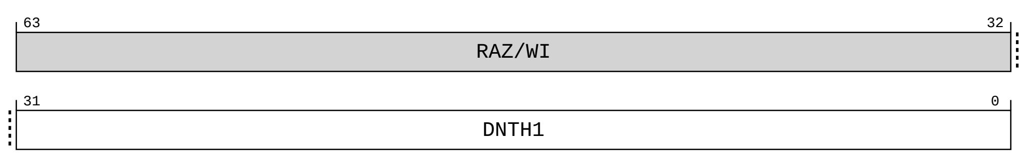


Table B-10: CLUSTERL3DNTH1 bit descriptions

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

Bit descriptions

Figure B-6: ext_clusterl3upth0 bit assignments

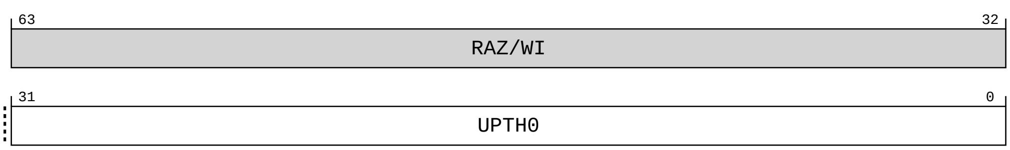


Table B-12: CLUSTERL3UPTH0 bit descriptions

Bits	Name	Description	Reset
[63:32]	RAZ/WI	Reserved	RAZ/WI
[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

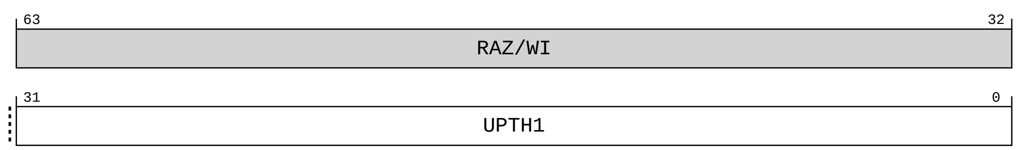


Table B-14: CLUSTERL3UPTH1 bit descriptions

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.2.13 IMP_CLUSTERBUSQOS_EL1, Cluster Bus QoS Control Register on page 336 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 1110 1110 1110 1110 1110 1110

Bit descriptions

Figure B-8: ext_clusterbusqos bit assignments

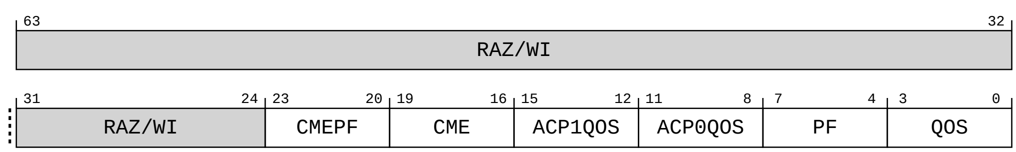


Table B-16: CLUSTERBUSQOS bit descriptions

Bits	Name	Description	Reset
[63:24]	RAZ/WI	Reserved	RAZ/WI
[23:20]	CMEPF	Value driven on the CHI bus QoS field for prefetches and non-latency critical accesses from any SME2 unit.	0b1110
[19:16]	CME	Value driven on the CHI bus QoS field for latency critical demand accesses from any SME2 unit.	0b1110
[15:12]	ACP1QOS	Value driven on the CHI bus QoS field for ACP 1 accesses.	0b1110
[11:8]	ACP0QOS	Value driven on the CHI bus QoS field for ACP 0 accesses.	0b1110

Bits	Name	Description	Reset
[7:4]	PF	Value driven on the CHI bus QoS field for prefetches and non-latency critical accesses. This includes all L3 evictions, including those caused by an SME2 unit or ACP transaction allocating to the L3 cache.	`0b1110`
[3:0]	QOS	Value driven on the CHI bus QoS field for latency critical 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.2.1 IMP_CLUSTERCFR_EL1, Cluster Configuration Register on page 300 bits [63:0].

Attributes

Width

64

Component

Cluster

Register offset

0x0050

Access type

RO

Reset value

```
xxxx xxxx x0xx x000 0000 0000 0000 0xxx xxxx xxx0 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-9: ext_clustercfr bit assignments

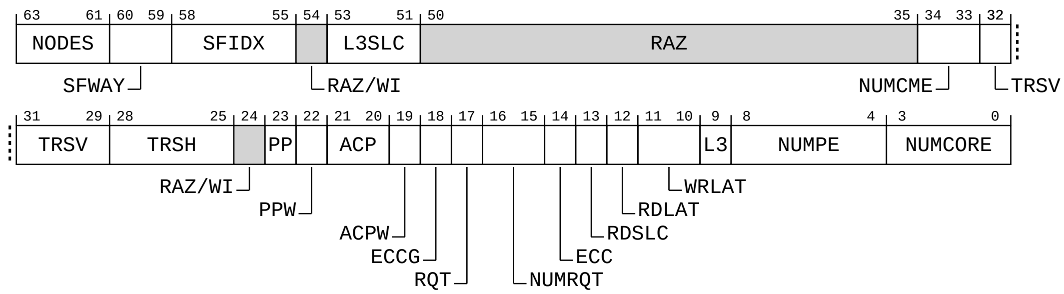


Table B-18: CLUSTERCFR bit descriptions

Bits	Name	Description	Reset
[63:61]	NODES	Number of transport nodes. **0b001** One node. **0b010** Two nodes. **0b011** Three nodes. **0b100** Four nodes. **0b101** Six nodes.	xxx
[60:59]	SFWAY	Number of Snoop Filter ways. **0b00** 4 ways **0b01** 6 ways **0b10** 8 ways **0b11** 12 ways	xx
[58:55]	SFIDX	Log2 of the number of snoop filter indexes.	xxxx
[54]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[53:51]	L3SLC	Number of L3 cache slices. **0b000** Eight L3 cache slices. **0b001** One L3 cache slice. **0b010** Two L3 cache slices. **0b100** Four L3 cache slices.	xxx
[50:35]	RAZ	Reserved	RAZ
[34:33]	NUMCME	Number of SME2 units configured **0b00** No SME2 units present **0b01** One SME2 unit present **0b10** Two SME2 units present	xx
[32:29]	TRSV	Transport register slices, vertical. **0b0000** No register slices **0b0001** One register slice **0b0010** Two register slices **0b0011** Three register slices **0b0100** Four register slices **0b0101** Five register slices **0b0110** Six register slices **0b0111** Seven register slices **0b1000** Eight register slices	xxxx

Bits	Name	Description	Reset
[28:25]	TRSH	Transport register slices, horizontal. `0b0000` No register slices `0b0001` One register slice `0b0010` Two register slices `0b0011` Three register slices `0b0100` Four register slices `0b0101` Five register slices `0b0110` Six register slices `0b0111` Seven register slices `0b1000` Eight register slices	`xxxx`
[24]	RAZ/WI	Reserved	RAZ/WI
[23]	PP	Peripheral port presence. `0b0` No peripheral port present `0b1` Peripheral port present	`x`
[22]	PPW	Peripheral port width. `0b0` 64 bit data width `0b1` 256 bit data width	`x`
[21:20]	ACP	ACP interface presence. `0b00` No ACP interface present `0b01` One ACP interface present `0b10` Two ACP interface present	`xx`

Bits	Name	Description	Reset

[19]	ACPW	ACP interface width.
		`0b0`
		128 bit data width
		`0b1`
		256 bit data width
			x

[18]	ECCG	L3 DataRAM ECC Granule.
		`0b0`
		128 bit data ECC
		`0b1`
		256 bit data ECC
			x

[17]	RQT	Requester bus interface type.
		`0b0`
		AXI interface
		`0b1`
		CHI interface
			x

[16:15]	NUMRQT	Number of Requester interfaces.
		`0b00`
		One requester
		`0b01`
		Two requesters
		`0b10`
		Three requesters
		`0b11`
		Four requesters
			xx

[14]	ECC	SCU-L3 ECC configuration.
		`0b0`
		SCU-L3 is configured with no ECC
		`0b1`
		SCU-L3 is configured with ECC
			x

[13]	RDSLC	L3 data RAM read register slice.
		`0b0`
		No register slice present
		`0b1`
		Register slice present
			x

[12]	RDLAT	L3 Data RAM read latency.
		`0b0`
		Two cycle output delay from L3 data RAMs
		`0b1`
		Three cycle output delay from L3 data RAMs
			x

Bits	Name	Description	Reset
[11:10]	WRLAT	L3 Data RAM write latency. **0b00** One cycle input delay from L3 data RAMs **0b01** Two cycle input delay from L3 data RAMs **0b10** Two cycle input delay plus a one cycle hold	xx
[9]	L3	L3 cache presence. **0b0** No L3 cache present **0b1** L3 cache present	x
[8:4]	NUMPE	Number of PEs present in the cluster. For single threaded cores, this number will be the same as bits [3:0]; for multi-threaded cores it will be larger.	5{x}

Bits	Name	Description	Reset
[3:0]	NUMCORE	Number of cores present in the cluster. **0b0000** One core **0b0001** Two cores **0b0010** Three cores **0b0011** Four cores **0b0100** Five cores **0b0101** Six cores **0b0110** Seven cores **0b0111** Eight cores **0b1000** Nine core **0b1001** Ten cores **0b1010** Eleven cores **0b1011** Twelve cores **0b1100** Thirteen cores **0b1101** 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.2.4 IMP_CLUSTERACTLR_EL1, Cluster Auxiliary Control Register on page 311 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
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-10: ext_clusteractlr bit assignments

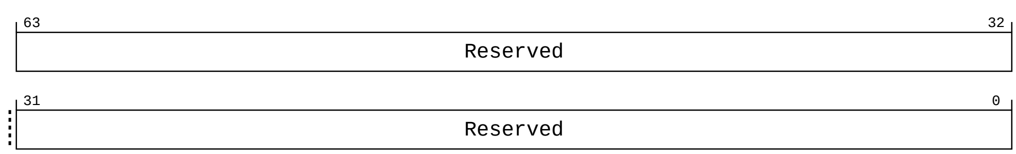


Table B-20: CLUSTERACTLR bit descriptions

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.2.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register on page 313 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 1010 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

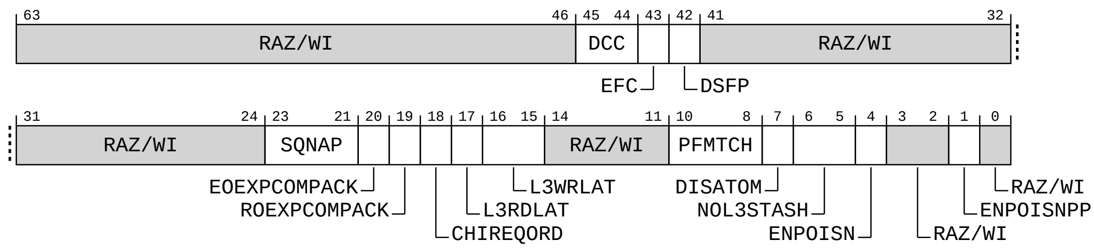


Table B-22: CLUSTERECTLR bit descriptions

Bits	Name	Description	Reset
[63:46]	RAZ/WI	Reserved	RAZ/WI
[45:44]	DCC	Downstream cache control. Controls whether evictions of clean cachelines send data on the CHI interface. Set this based on whether there is a cache on the path to memory. This bit is **RES0** in direct connect configuration. **0b00** Disables sending data when clean cachelines are evicted. **0b01** Enables sending WriteEvictFull transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **0b10** Enables sending WriteEvictOrEvict transactions when Unique Clean cachelines are evicted. Shared Clean cacheline evictions do not send data. **0b11** Enables sending WriteEvictOrEvict transactions when Unique Clean or Shared Clean cachelines are evicted. This is the reset value.	0b11
[43]	EFC	Eviction flush control. Controls whether hardware cache flushes and DC CISW instructions send data when evicting clean cachelines on the CHI interface. This bit is **RES0** in direct connect configuration. **0b0** Disables sending data when hardware cache flushes or DC CISW instructions evict a clean cacheline. Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP). This is the reset value. **0b1** Sending of data when hardware cache flushes or DC CISW instructions evict clean cachelines is controlled by Downstream Cache Control (DCC). Sending of Evict transactions is controlled by Downstream Snoop Filter Present (DSFP).	0b0

Bits	Name	Description	Reset
[42]	DSFP	Downstream snoop filter present. Enables sending Evict transactions on the CHI interface when clean cachelines are evicted without data. Enable this if there is at least one snoop filter in the path to memory. This bit is **RES0** in direct connect configuration. **0b0** Disables sending Evict transactions when clean cachelines are evicted without data. **0b1** Enables sending of Evict transactions when clean cachelines are evicted without data. This is the reset value.	`0b1`
[41:24]	**RAZ/WI**	Reserved	**RAZ/WI**
[23:21]	SQNAP	L3 data RAM quick nap sub-enable time. Controls the number of SCLK cycles to wait after an L3 data RAM access before setting the corresponding quick nap pin for the sub-bank. If this sub-bank of the RAM has been idle for this number of cycles, then it is put into quick nap, even if the entire slice timeout controlled by CLUSTERPWRCTLR.QNAP has not become idle yet. This bit is **RES0** in direct connect configuration. **0b000** Per-pin quick nap disabled **0b001** Enabled with 8 cycle timeout **0b010** Enabled with 12 cycle timeout **0b011** Enabled with 16 cycle timeout **0b100** Enabled with 24 cycle timeout **0b101** Enabled with 32 cycle timeout **0b110** Enabled with 64 cycle timeout **0b111** Enabled with 128 cycle timeout	`0b101`
[20]	EOEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Endpoint Order transactions to the system. This bit is **RES0** in direct connect configuration. **0b0** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Endpoint Order will have ExpCompAck=1	`0b0`
[19]	ROEXPCOMPACK	Controls the CHI ExpCompAck field when sending CHI ReadNoSnp Request Order transactions to the system. This bit is **RES0** in direct connect configuration. **0b0** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=0 **0b1** CHI ReadNoSnp transactions sent with Request Order will have ExpCompAck=1	`0b0`

Bits	Name	Description	Reset
[18]	CHIREQORD	Allow Request Order on CHI ports. Enables the use of Request Order when sending Non-snoopable CHI transactions to the system for Dev-R and Normal NC memory. This bit is **RES0** in direct connect configuration. **`0b0`** Disables sending Request Order on the CHI interface to the system. Will send No Order instead. **`0b1`** Enables sending Request Order on the CHI interface to the system for Non-snoopable transactions.	`0b0`
[17]	L3RDLAT	L3 data RAM read (output) latency. This bit is **RES0** in direct connect configuration. **`0b0`** The L3 data RAM output latency is 2 cycles. **`0b1`** The L3 data RAM output latency is 3 cycles.	`x` [3]
[16:15]	L3WRLAT	L3 data RAM write (input) latency. This bit is **RES0** in direct connect configuration. **`0b00`** The L3 data RAM input latency is 1 cycle with an additional hold cycle. **`0b01`** The L3 data RAM input latency is 2 cycles without an additional hold cycle. **`0b10`** The L3 data RAM input latency is 2 cycles with an additional hold cycle. This is only usable if the L3 data RAM output latency is 3 cycles.	`xx` [4]
[14:11]	**RAZ/WI**	Reserved	**RAZ/WI**
[10:8]	PFMTCH	Prefetch matching delay. Controls the amount of tie a prefetch waits for a possible match with a later read. Encoded as powers of 2, from 1-128. This bit is **RES0** in direct connect configuration. **`0b000`** Wait for 1 cycle. **`0b001`** Wait for 2 cycles. **`0b010`** Wait for 4 cycles. **`0b011`** Wait for 8 cycles. **`0b100`** Wait for 16 cycles. **`0b101`** Wait for 32 cycles. **`0b110`** Wait for 64 cycles. **`0b111`** Wait for 128 cycles.	`0b101`

[3] This field resets to the value of the L3_DATA_RD_LATENCY configuration parameter.
[4] This field resets to the value of the L3_DATA_WR_LATENCY configuration parameter.

Bits	Name	Description	Reset
[7]	DISATOM	Disable cacheable shareable atomics being sent to the interconnect. This bit is **RES0** in direct connect configuration. **0b0** Cacheable shareable atomics will be sent to the interconnect if the BROADCASTATOMIC pin is set. **0b1** Cacheable shareable atomics will be handled inside the cluster.	0b0
[6:5]	NOL3STASH	CPU StashOnce request behaviour when L3 is not present or powered down. This bit is **RES0** in direct connect configuration. **0b00** Stashes are sent out to the interconnect, if supported. **0b01** Normal read request sent to interconnect. **0b10** StashOnce has no effect.	0b10
[4]	ENPOISN	Interconnect data poisoning support for the CHI Requester(s). This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES1** in direct connect configuration. **0b0** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **0b1** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	0b1
[3:2]	**RAZ/WI**	Reserved	**RAZ/WI**
[1]	ENPOISNPP	Interconnect data poisoning support for the CHI Peripheral Port. This bit is ignored for AXI configurations, which never support poisoning. This bit is **RES1** in direct connect configuration. **0b0** Interconnect does not support data poisoning, so nCLUSTERERRIREQ will be asserted when poisoned data is evicted from the cluster or returned on a snoop. **0b1** Interconnect supports data poisoning, so no error recovery interrupt will be generated when poisoned data is evicted from the cluster or returned on a snoop.	0b1
[0]	**RAZ/WI**	Reserved	**RAZ/WI**

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.2.18 IMP_CLUSTERCFR2_EL1, Cluster Configuration Register 2 on page 346 bits [63:0].

Attributes

Width

64

Component

Cluster

Register offset

0x0068

Access type

RO

Reset value

```
0000 0000 0000 0000 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-12: ext_clustercfr2 bit assignments

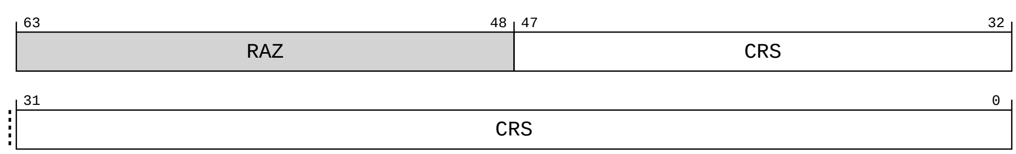


Table B-24: CLUSTERCFR2 bit descriptions

Bits	Name	Description	Reset
[63:48]	RAZ	Reserved	RAZ

Bits	Name	Description	Reset

[47:0]	CRS	Core and SME2 unit register slices: Each three bits represents a core or SME2 unit, with bits [2:0] for core 0 up to bits [41:39] for core 13. The SME2 unit bits follow immediately after the highest numbered core, e.g. bits [14:12] for the first SME2 unit when 4 cores are configured.
		`0b000000000000000000000000000000000000000000000000`
		No register slices
		`0b000000000000000000000000000000000000000000000001`
		One register slice
		`0b000000000000000000000000000000000000000000000010`
		Two register slices
		`0b000000000000000000000000000000000000000000000011`
		Three register slices
		`0b000000000000000000000000000000000000000000000100`
		Four register slices
		`0b000000000000000000000000000000000000000000000101`
		Five register slices
		`0b000000000000000000000000000000000000000000000110`
		Six register slices
		`0b000000000000000000000000000000000000000000000111`
		Seven register slices
			`48{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

RW

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

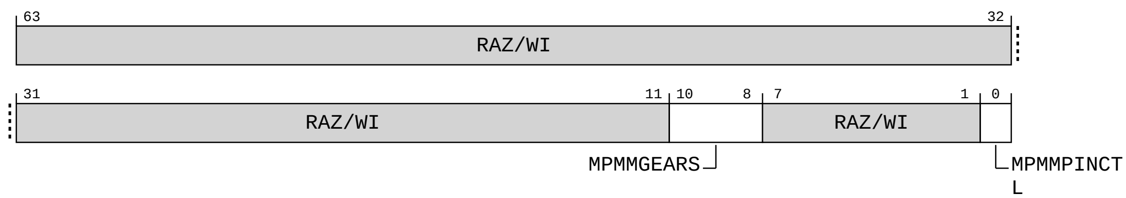


Table B-26: CLUSTERPPMCR bit descriptions

Bits	Name	Description	Reset
[63:11]	**RAZ/WI**	Reserved	**RAZ/WI**
[10:8]	MPMMGEARS	Number of MPMM gears implemented. **0b100** Four MPMM gears implemented	0b100
[7:1]	**RAZ/WI**	Reserved	**RAZ/WI**
[0]	MPMMPINCTL	MPMM pin control enable. **0b0** MPMM controlled by the CLUSTERMPMMCR register, the external pins are ignored **0b1** 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_clustermpmmcr bit assignments

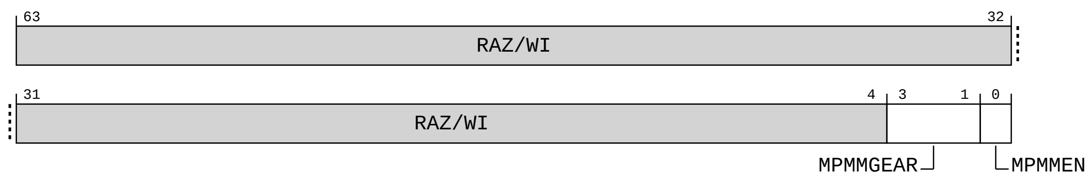


Table B-28: CLUSTERMPMMCR bit descriptions

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

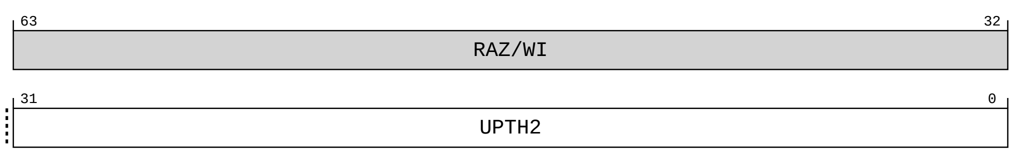


Table B-30: CLUSTERL3UPTH2 bit descriptions

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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If Realm Management Extension (RME) is enabled, meaning that the cluster is in Direct connect, these registers are not present. For more information on enabling RME, see 1.4.1 Realm management extension on page 26.
- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- 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	Present in Direct connect
0x0000	MPAMF_IDR	See individual bit resets.	64-bit	MPAM Features Identification Register	No
0x0000	MPAMF_IDR	See individual bit resets.	64-bit	MPAM Features Identification Register	No
0x0008	MPAMF_SIDR	See individual bit resets.	32-bit	MPAM Features Secure Identification Register	No
0x0018	MPAMF_IIDR	See individual bit resets.	32-bit	MPAM Implementation Identification Register	No
0x0020	MPAMF_AIDR	See individual bit resets.	32-bit	MPAM Architecture Identification Register	No
0x0030	MPAMF_CPOR_IDR	See individual bit resets.	32-bit	MPAM Features Cache Portion Partitioning ID register	No
0x0030	MPAMF_CPOR_IDR	See individual bit resets.	32-bit	MPAM Features Cache Portion Partitioning ID register	No
0x0040	MPAMF_MBW_IDR	See individual bit resets.	32-bit	MPAM Memory Bandwidth Partitioning Identification Register	No
0x0040	MPAMF_MBW_IDR	See individual bit resets.	32-bit	MPAM Memory Bandwidth Partitioning Identification Register	No
0x00F0	MPAMF_ECR	See individual bit resets.	32-bit	MPAM Error Control Register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x00F0	MPAMF_ECR	See individual bit resets.	32-bit	MPAM Error Control Register	No
0x00F8	MPAMF_ESR	See individual bit resets.	32-bit	MPAM Error Status Register	No
0x00F8	MPAMF_ESR	See individual bit resets.	32-bit	MPAM Error Status Register	No
0x0100	MPAMCFG_PART_SEL	See individual bit resets.	32-bit	MPAM Partition Configuration Selection Register	No
0x0100	MPAMCFG_PART_SEL	See individual bit resets.	32-bit	MPAM Partition Configuration Selection Register	No
0x0500	MPAMCFG_MBW_PROP_ns	See individual bit resets.	32-bit	MPAM Memory Bandwidth Proportional Stride Partition Configuration for Non-Secure PARTIDs	No
0x0500	MPAMCFG_MBW_PROP_s	See individual bit resets.	32-bit	MPAM Memory Bandwidth Proportional Stride Partition Configuration for Secure PARTIDs	No
0x1000	MPAMCFG_CPBM_ns	See individual bit resets.	32-bit	MPAM Cache Portion Bitmap Partition Configuration Register for Non-secure PARTIDs	No
0x1000	MPAMCFG_CPBM_s	See individual bit resets.	32-bit	MPAM Cache Portion Bitmap Partition Configuration Register for Secure PARTIDs	No

B.1.2.1 MPAMF_IDR, MPAM Features Identification Register

Indicates which memory partitioning and monitoring features are present on this MSC.

MPAMF_IDR_s indicates the MPAM features accessed from the Secure MPAM feature page.

MPAMF_IDR_ns indicates the MPAM features accessed from the Non-secure MPAM feature page.

When MPAMF_IDR.HAS_RIS is 1, some fields in this register give information for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS. The description of every field that is affected by ext-MPAMCFG_PART_SEL.RIS has that information within the field description.

Configurations

MPAMF_IDR is 64-bit register when MPAM v0.1 or v1.1 is implemented.

Otherwise, MPAMF_IDR is a 32-bit register.

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

64

Component

MPAM

Register offsets (2)

0x0000,0x0000

Access type

Read

R

Write

RESERVED

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 B-16: ext_mpamf_idr bit assignments

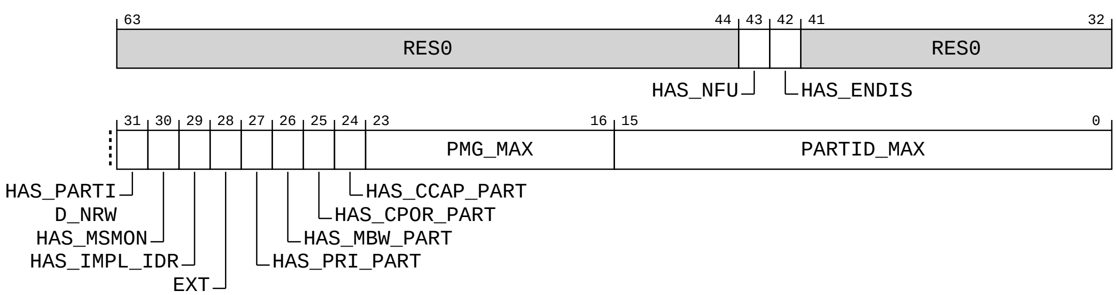


Table B-33: MPAMF_IDR bit descriptions

Bits	Name	Description	Reset
[63:44]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[43]	HAS_NFU	Has No Future Use field in ext-MPAMCFG_DIS. Indicates that ext-MPAMCFG_DIS.NFU is implemented. **0b0** ext-MPAMCFG_DIS.NFU is not implemented. A PARTID disabled through access to MPAMCFG_DIS must preserve the control settings of the disabled PARTID. **0b1** Implements ext-MPAMCFG_DIS.NFU. A PARTID disabled with NFU as 1 may have its control settings forgotten. If ext-MPAMF_IDR.HAS_ENDIS is `0b0`, this field must also be `0b0`. This field must be the same in each instance of this register and for any value in ext-MPAMCFG_PART_SEL.RIS.	x
[42]	HAS_ENDIS	Has PARTID enable and disable. Indicates that this MSC supports PARTID disable and enable via ext-MPAMCFG_DIS, ext-MPAMCFG_EN and ext-MPAMCFG_EN_FLAGS registers. **0b0** Does not support PARTID enable and disable functionality, and ext-MPAMCFG_EN, ext-MPAMCFG_DIS and MPAMCFG_EN_FLAGS registers are not implemented. **0b1** Supports PARTID enable and disable through the ext-MPAMCFG_EN, ext-MPAMCFG_DIS and MPAMCFG_EN_FLAGS registers. All three registers must be implemented when this field is 1, ext-MPAMCFG_EN, ext-MPAMCFG_DIS, and ext-MPAMCFG_EN_FLAGS. This field must be the same in each instance of this register and for any value in ext-MPAMCFG_PART_SEL.RIS.	x
[41:32]	**RES0**	Reserved	**RES0**
[31]	HAS_PARTID_NRW	Has PARTID narrowing. **0b0** Does not have ext-MPAMF_PARTID_NRW_IDR, ext-MPAMCFG_INTPARTID, or intPARTID mapping support. **0b1** Supports the ext-MPAMF_PARTID_NRW_IDR, ext-MPAMCFG_INTPARTID registers.	x
[30]	HAS_MSMON	Has resource Monitors. Indicates whether this MSC has MPAM resource monitors. **0b0** Does not support MPAM resource monitoring by groups or ext-MPAMF_MSMON_IDR. **0b1** Supports resource monitoring by matching a combination of PARTID and PMG. See ext-MPAMF_MSMON_IDR.	x
[29]	HAS_IMPL_IDR	Has ext-MPAMF_IMPL_IDR. Indicates whether this MSC has the IMPLEMENTATION SPECIFIC MPAM features register, ext-MPAMF_IMPL_IDR. **0b0** Does not have ext-MPAMF_IMPL_IDR. **0b1** Has ext-MPAMF_IMPL_IDR.	x

Bits	Name	Description	Reset
[28]	EXT	Extended MPAMF_IDR. `0b0` MPAMF_IDR has no defined bits in [63:32]. The register is effectively 32 bits.	x

[27]	HAS_PRI_PART	Has Priority Partitioning. Indicates that MPAM priority partitioning is implemented and ext-MPAMF_PRI_IDR exists.
		`0b0`
		Does not support priority partitioning or have ext-MPAMF_PRI_IDR.
		`0b1`
		Has priority partitioning and ext-MPAMF_PRI_IDR.
		If RIS is implemented, this field indicates the presence of priority partitioning resource controls as described in ext-MPAMF_PRI_IDR for the selected resource instance.
			x

[26]	HAS_MBW_PART	Has Memory Bandwidth Partitioning. Indicates whether this MSC implements MPAM memory bandwidth partitioning and ext-MPAMF_MBW_IDR.
		`0b0`
		Does not support memory bandwidth partitioning or have ext-MPAMF_MBW_IDR register.
		`0b1`
		Has ext-MPAMF_MBW_IDR register.
		If RIS is implemented, this field indicates the presence of memory bandwidth partitioning resource controls as described in ext-MPAMF_MBW_IDR for the selected resource instance.
			x

[25]	HAS_CPOR_PART	Has Cache Portion Partitioning. Indicates whether this MSC implements MPAM cache portion partitioning and ext-MPAMF_CPOR_IDR.
		`0b0`
		Does not support cache portion partitioning or have ext-MPAMF_CPOR_IDR or ext-MPAMCFG_CPBM<n> registers.
		`0b1`
		Has ext-MPAMF_CPOR_IDR and ext-MPAMCFG_CPBM<n> registers.
		If RIS is implemented, this field indicates the presence of cache portion partitioning resource controls as described in ext-MPAMF_CPOR_IDR for the selected resource instance.
			x

[24]	HAS_CCAP_PART	Has Cache Capacity Partitioning. Indicates whether this MSC implements MPAM cache capacity partitioning and the ext-MPAMF_CCAP_IDR and ext-MPAMCFG_CMAX registers.
		`0b0`
		Does not support cache capacity partitioning or have ext-MPAMF_CCAP_IDR and ext-MPAMCFG_CMAX registers.
		`0b1`
		Has ext-MPAMF_CCAP_IDR and ext-MPAMCFG_CMAX registers.
		If RIS is implemented, this field indicates the presence of cache capacity partitioning resource controls as described in ext-MPAMF_CPOR_IDR for the selected resource instance.
			x

Bits	Name	Description	Reset
[23:16]	PMG_MAX	Maximum supported value of PMG. The value of this field is permitted to vary between the instances of ext-MPAMF_IDR, each reporting the maximum supported PMG value in the PARTID space associated with that instance. In MPAMF_IDR_s, this field is permitted to report the maximum PMG value for the Non-secure PARTID space or for the Secure PARTID space. The maximum PMG value for the Secure PARTID space can be read from ext-MPAMF_SIDR.PMG_MAX.	8{x}
[15:0]	PARTID_MAX	Maximum supported value of PARTID. The value of this field is permitted to vary between the instances of ext-MPAMF_IDR, each reporting the maximum supported PARTID value in the PARTID space associated with that instance. In MPAMF_IDR_s, this field is permitted to report the maximum PARTID value for the Non-secure PARTID space or for the Secure PARTID space. The maximum PARTID value for the Secure PARTID space can be read from ext-MPAMF_SIDR.PARTID_MAX.	16{x}

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_IDR is read-only.

MPAMF_IDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_IDR is permitted to have the same contents when read from the Secure or Non-secure MPAM feature pages unless the register contents are different for the different versions, when MPAMF_IDR_s is permitted to have either the same or different contents to MPAMF_IDR_ns.

There must be separate registers in the Secure (MPAMF_IDR_s) and Non-secure (MPAMF_IDR_ns) MPAM feature pages.

When ext-MPAMF_IDR.HAS_RIS is 1, MPAMF_IDR shows the configuration of MSC MPAM for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS. Fields that mention RIS in their field descriptions have values that track the implemented properties of the resource instance. Fields that do not mention RIS are constant across all resource instances.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_IDR is read-only.

MPAMF_IDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_IDR is permitted to have the same contents when read from the Secure or Non-secure MPAM feature pages unless the register contents are different for the different versions, when MPAMF_IDR_s is permitted to have either the same or different contents to MPAMF_IDR_ns.

There must be separate registers in the Secure (MPAMF_IDR_s) and Non-secure (MPAMF_IDR_ns) MPAM feature pages.

When ext-MPAMF_IDR.HAS_RIS is 1, MPAMF_IDR shows the configuration of MSC MPAM for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS. Fields that mention RIS in their field descriptions have values that track the implemented properties of the resource instance. Fields that do not mention RIS are constant across all resource instances.

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

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offset

0x0008

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx 0000 0001 0000 0000 0000 0111
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-17: ext_mpamf_sidr bit assignments

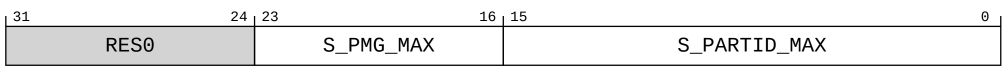


Table B-36: MPAMF_SIDR bit descriptions

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. **0b0000000000000111** Supports 8 Secure PARTIDs.	0x0007

Access

This register is only within the Secure MPAM feature page memory frame.

MPAMF_SIDR is read-only.

MPAMF_SIDR must only be readable from the Secure MPAM feature page. If the system or the MSC does not support the Secure address map, this register must not be accessible.

Accessibility

This register is only within the Secure MPAM feature page memory frame.

MPAMF_SIDR is read-only.

MPAMF_SIDR must only be readable from the Secure MPAM feature page. If the system or the MSC does not support the Secure address map, this register must not be accessible.

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

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offset

0x0018

Access type

Read

R

Write

RESERVED

Reset value

0100 1110 1110 0000 0010 0100 0011 1011

Bit descriptions

Figure B-18: ext_mpamf_iidr bit assignments

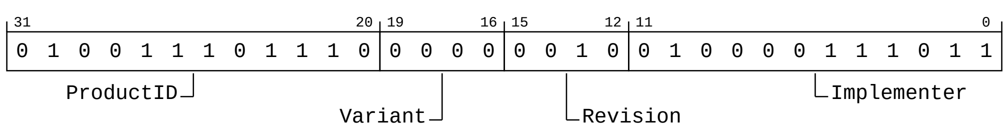


Table B-38: MPAMF_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	The MSC implementer as identified in the MPAMF_IIDR.Implementer field must assure each product has a unique ProductID from any other with the same Implementer value. **`0b010011101110`** DSU-C1 Cluster MPAM.	`0x4EE`

Bits	Name	Description	Reset
[19:16]	Variant	This field distinguishes product variants or major revisions of the product. **Note:** Implementations of ProductID with differing software interfaces are expected to have different values in the MPAMF_IIDR.Variant field. **`0b0000`** Product Variant 0.	`0b0000`
[15:12]	Revision	This field distinguishes minor revisions of the product. **Note:** This field is intended to differentiate product revisions that are minor changes and are largely software compatible with previous revisions. **`0b0010`** Product major revision 2.	`0b0010`
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the MPAM MSC. [11:8] must contain the JEP106 continuation code of the implementer. [7] must always be 0. [6:0] must contain the JEP106 identity code of the implementer. For an Arm implementation, bits[11:0] are `0x43B`. **`0b010000111011`** Arm implementation.	`0x43B`

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_IIDR is read-only.

MPAMF_IIDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_IIDR must have the same contents in the Secure and Non-secure MPAM feature pages.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_IIDR is read-only.

MPAMF_IIDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_IIDR must have the same contents in the Secure and Non-secure MPAM feature pages.

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

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offset

0x0020

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0001 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-19: ext_mpamf_aidr bit assignments

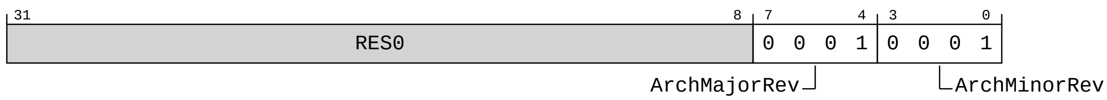


Table B-40: MPAMF_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	ArchMajorRev	Major revision of the MPAM architecture implemented by the MSC. **0b0001** This table shows the only valid combinations of MPAM version numbers in an MSC. FORCE_NS functionality is only available in MPAM v0.1. Table B-41: ArchMajorRev description on page 434 Use of MPAMv0.1 in MSCs is restricted to limited circumstances. The MSC must be able to initiate requests in the Secure address space which have MPAM PARTID forced to the Non-secure space with that forcing not controllable or observable by the software that configures the device for Secure requests. Please contact Arm before setting MPAMF_AIDR to report MPAMv0.1.	0b0001
[3:0]	ArchMinorRev	Minor revision of the MPAM architecture implemented by the MSC. **0b0001** See the table in the description of the ArchMajorRev field in this register.	0b0001

Table B-41: ArchMajorRev description

ArchMajorRev	ArchMinorRev	MPAMv	Available
0	0		None.
0	1	v0.1	MPAMv1.0 + MPAMv1.1 + FORCE_NS
1	0	v1.0	MPAMv1.0
1	1	v1.1	MPAMv1.0 + MPAMv1.1 - FORCE_NS

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_AIDR is read-only.

MPAMF_AIDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_AIDR must have the same contents in the Secure and Non-secure MPAM feature pages.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_AIDR is read-only.

MPAMF_AIDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_AIDR must have the same contents in the Secure and Non-secure MPAM feature pages.

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<n>.

MPAMF_CPOR_IDR_s indicates the number of bits in the Secure instance of ext-MPAMCFG_CPBM<n>.

MPAMF_CPOR_IDR_ns indicates the number of bits in the Non-secure instance of ext-MPAMCFG_CPBM<n>.

When ext-MPAMF_IDR.HAS_RIS is 1, some fields in this register give information for the resource instance selector, ext-MPAMCFG_PART_SEL.RIS. The description of every field that is affected by ext-MPAMCFG_PART_SEL.RIS has information within the field description.

Configurations

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0030,0x0030

Access type

Read

R

Write

RESERVED

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-20: ext_mpamf_cpor_idr bit assignments

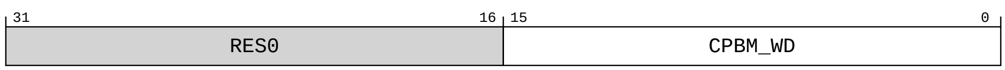


Table B-43: MPAMF_CPOR_IDR bit descriptions

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<n>. This field must contain a value from 1 to 32768, inclusive. Values greater than 32 require a group of 32-bit registers to access the CPBM, up to 1024 if CPBM_WD is the largest value. If RIS is implemented, this field indicates the number bits in the cache portion bitmap for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	16{x}

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_CPOR_IDR is read-only.

MPAMF_CPOR_IDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_CPOR_IDR is permitted to have the same contents when read from the Secure or Non-secure MPAM feature pages unless the register contents are different for the different versions, when MPAMF_CPOR_IDR_s is permitted to have either the same or different contents to MPAMF_CPOR_IDR_ns.

There must be separate registers in the Secure (MPAMF_CPOR_IDR_s) and Non-secure (MPAMF_CPOR_IDR_ns) MPAM feature pages.

When ext-MPAMF_IDR.HAS_RIS is 1, MPAMF_CPOR_IDR shows the configuration of cache portion partitioning for the cache resource instance selected by ext-MPAMCFG_PART_SEL.RIS. Fields that mention RIS in their field descriptions have values that track the implemented properties of the resource instance. Fields that do not mention RIS are constant across all resource instances.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_CPOR_IDR is read-only.

MPAMF_CPOR_IDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_CPOR_IDR is permitted to have the same contents when read from the Secure or Non-secure MPAM feature pages unless the register contents are different for the different versions, when MPAMF_CPOR_IDR_s is permitted to have either the same or different contents to MPAMF_CPOR_IDR_ns.

There must be separate registers in the Secure (MPAMF_CPOR_IDR_s) and Non-secure (MPAMF_CPOR_IDR_ns) MPAM feature pages.

When ext-MPAMF_IDR.HAS_RIS is 1, MPAMF_CPOR_IDR shows the configuration of cache portion partitioning for the cache resource instance selected by ext-MPAMCFG_PART_SEL.RIS. Fields that mention RIS in their field descriptions have values that track the implemented properties of the resource instance. Fields that do not mention RIS are constant across all resource instances.

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.

When ext-MPAMF_IDR.HAS_RIS is 1, some fields in this register give information for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS. The description of every field that is affected by ext-MPAMCFG_PART_SEL.RIS has that information within the field description.

Configurations

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x0040,0x0040

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx xx1x xxxx xx00 0110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-21: ext_mpamf_mbw_idr bit assignments

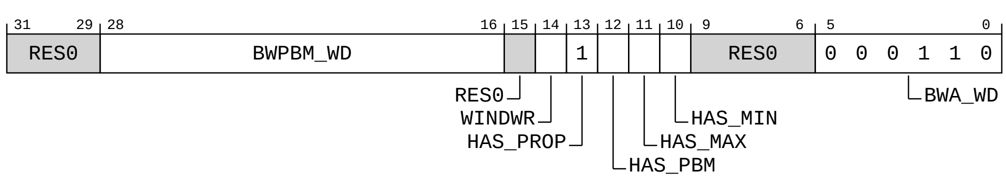


Table B-46: MPAMF_MBW_IDR bit descriptions

Bits	Name	Description	Reset
[31:29]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[28:16]	BWPBM_WD	Bandwidth portion bitmap width. The number of bandwidth portion bits in the ext-MPAMCFG_MBW_PBM<n> register array. If MPAMF_MBW_IDR.HAS_PBM is 1, this field must contain a value from 1 to 4096, inclusive. Values greater than 32 require a group of 32-bit registers to access the BWPBM, up to 128 if BWPBM_WD is the largest value. If MPAMF_MBW_IDR.HAS_PBM is 0, this field must be ignored by software. If RIS is implemented, this field indicates the width of the memory bandwidth portion bitmap partitioning control for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	`13{x}`
[15]	RES0	Reserved	RES0
[14]	WINDWR	Indicates the bandwidth accounting period register is writable. **`0b0`** The bandwidth accounting period is readable from ext-MPAMCFG_MBW_WINWD which might be fixed or vary due to clock rate reconfiguration of the memory channel or memory controller. **`0b1`** The bandwidth accounting width is readable and writable per partition in ext-MPAMCFG_MBW_WINWD.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[13]	HAS_PROP	Indicates that this MSC implements proportional stride bandwidth partitioning and the ext-MPAMCFG_MBW_PROP register can be accessed. **`0b1`** The proportional stride memory bandwidth partitioning scheme is supported and the ext-MPAMCFG_MBW_PROP register can be accessed. If RIS is implemented, this field indicates the presence of the memory bandwidth proportional stride partitioning control for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	`0b1`
[12]	HAS_PBM	Indicates that bandwidth portion partitioning is implemented and the ext-MPAMCFG_MBW_PBM<n> register array can be accessed. **`0b0`** There is no memory bandwidth portion control and the ext-MPAMCFG_MBW_PBM<n> is RES0. **`0b1`** The memory bandwidth portion allocation scheme exists and the ext-MPAMCFG_MBW_PBM<n> register can be accessed. If RIS is implemented, this field indicates the presence of the memory bandwidth portion partitioning control for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	The reset values can be the following: `0b0`, `0b1`, respective to the value.

Bits	Name	Description	Reset
[11]	HAS_MAX	Indicates that this MSC implements maximum bandwidth partitioning and the ext-MPAMCFG_MBW_MAX register can be accessed. **`0b0`** There is no maximum memory bandwidth control and the ext-MPAMCFG_MBW_MAX register is **RES0**. **`0b1`** The maximum memory bandwidth allocation scheme is supported and the ext-MPAMCFG_MBW_MAX register can be accessed. If RIS is implemented, this field indicates the presence of the maximum bandwidth partitioning control for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[10]	HAS_MIN	Indicates that this MSC implements minimum bandwidth partitioning and the ext-MPAMCFG_MBW_MIN register can be accessed. **`0b0`** There is no minimum memory bandwidth control and the ext-MPAMCFG_MBW_MIN register is **RES0**. **`0b1`** The minimum memory bandwidth allocation scheme is supported and the ext-MPAMCFG_MBW_MIN register can be accessed. If RIS is implemented, this field indicates the presence of the minimum bandwidth partitioning control for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[9:6]	**RES0**	Reserved	**RES0**
[5:0]	BWA_WD	Number of implemented bits in the bandwidth allocation fields: MIN, MAX, and STRIDE. See ext-MPAMCFG_MBW_MIN, ext-MPAMCFG_MBW_MAX, and ext-MPAMCFG_MBW_PROP. In any of these bandwidth allocation fields exist, this field must have a value from 1 to 16, inclusive. Otherwise, it is permitted to be 0. **`0b000110`** If RIS is implemented, this field indicates the number of implemented bits in the bandwidth allocation control fields for the resource instance selected by ext-MPAMCFG_PART_SEL.RIS.	`0b000110`

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_MBW_IDR is read-only.

MPAMF_MBW_IDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_MBW_IDR is permitted to have the same contents when read from the Secure or Non-secure MPAM feature pages unless the register contents are different for the different versions, when MPAMF_MBW_IDR_s is permitted to have either the same or different contents to MPAMF_MBW_IDR_ns.

There must be separate registers in the Secure (MPAMF_MBW_IDR_s) and Non-secure (MPAMF_MBW_IDR_ns) MPAM feature pages.

When ext-MPAMF_IDR.HAS_RIS is 1, MPAMF_MBW_IDR shows the configuration of memory bandwidth partitioning for the bandwidth resource instance selected by ext-MPAMCFG_PART_SEL.RIS. Fields that mention RIS in their field descriptions have values that track the implemented properties of the resource instance. Fields that do not mention RIS are constant across all resource instances.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_MBW_IDR is read-only.

MPAMF_MBW_IDR must be readable from the Non-secure and Secure MPAM feature pages.

MPAMF_MBW_IDR is permitted to have the same contents when read from the Secure or Non-secure MPAM feature pages unless the register contents are different for the different versions, when MPAMF_MBW_IDR_s is permitted to have either the same or different contents to MPAMF_MBW_IDR_ns.

There must be separate registers in the Secure (MPAMF_MBW_IDR_s) and Non-secure (MPAMF_MBW_IDR_ns) MPAM feature pages.

When ext-MPAMF_IDR.HAS_RIS is 1, MPAMF_MBW_IDR shows the configuration of memory bandwidth partitioning for the bandwidth resource instance selected by ext-MPAMCFG_PART_SEL.RIS. Fields that mention RIS in their field descriptions have values that track the implemented properties of the resource instance. Fields that do not mention RIS are constant across all resource instances.

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

If an MSC cannot encounter any of the error conditions listed in 'Errors in MSCs' in Arm® Architecture Reference Manual Supplement, Memory System Resource Partitioning and Monitoring (MPAM), for Armv8-A (ARM DDI 0598), both the ext-MPAMF_ESR and MPAMF_ECR must be RAZ/WI.

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x00F0,0x00F0

Access type

Read

R

Write

W

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-22: ext_mpamf_ecr bit assignments

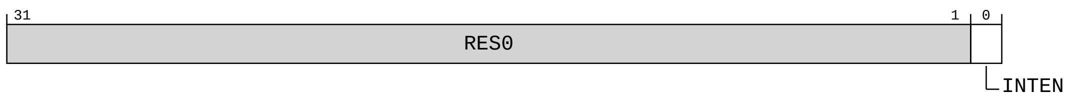


Table B-49: MPAMF_ECR bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	INTEN	Interrupt Enable. **`0b0`** MPAM error interrupts are not signaled. **`0b1`** MPAM error interrupts are signaled.	`x`

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_ECR_s must only be accessible from the Secure MPAM feature page. MPAMF_ECR_ns must only be accessible from the Non-secure MPAM feature page.

MPAMF_ECR_s and MPAMF_ECR_ns must be separate registers. The Secure instance (MPAMF_ECR_s) accesses the error interrupt controls used for Secure PARTIDs, and the Non-secure instance (MPAMF_ECR_ns) accesses the error interrupt controls used for Non-secure PARTIDs.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_ECR_s must only be accessible from the Secure MPAM feature page. MPAMF_ECR_ns must only be accessible from the Non-secure MPAM feature page.

MPAMF_ECR_s and MPAMF_ECR_ns must be separate registers. The Secure instance (MPAMF_ECR_s) accesses the error interrupt controls used for Secure PARTIDs, and the Non-secure instance (MPAMF_ECR_ns) accesses the error interrupt controls used for Non-secure PARTIDs.

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

MPAMF_ESR is 64-bit register when MPAM v0.1 or v1.1 is implemented and MPAMF_IDR.HAS_EXTD_ESR == 1.

Otherwise, MPAMF_ESR is a 32-bit register.

If an MSC cannot encounter any of the error conditions listed in 'Errors in MSCs' in Arm® Architecture Reference Manual Supplement, Memory System Resource Partitioning and Monitoring (MPAM), for Armv8-A (ARM DDI 0598), both the MPAMF_ESR and ext-MPAMF_ECR must be RAZ/WI.

The power and reset domain of each MSC component is specific to that component.

Attributes

Width

32

Component

MPAM

Register offsets (2)

0x00F8,0x00F8

Access type

Read

R

Write

W

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-23: ext_mpamf_esr bit assignments

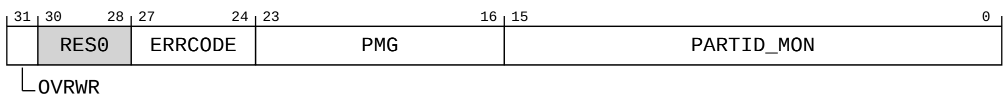


Table B-52: MPAMF_ESR bit descriptions

Bits	Name	Description	Reset
[31]	OVRWR	Overwritten. If 0 and ERRCODE == `0b0000`, no errors have occurred. If 0 and ERRCODE is nonzero, a single error has occurred and is recorded in this register. If 1 and ERRCODE is nonzero, multiple errors have occurred and this register records the most recent error. The state where this bit is 1 and ERRCODE is 0 must not be produced by hardware and is only reached when software writes this combination into this register.	`x`
[30:28]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[27:24]	ERRCODE	Error code. **0b0000** No error. **0b0001** PARTID_SEL_Range. **0b0010** Req_PARTID_Range. **0b0011** MSMONCFG_ID_RANGE. **0b0100** Req_PMG_Range. **0b0101** Monitor_Range. **0b0110** intPARTID_Range. **0b0111** Unexpected_INTERNAL. **0b1000** Reserved. **0b1001** Reserved. **0b1010** Reserved. **0b1011** Reserved. **0b1100** Reserved. **0b1101** Reserved. **0b1110** Reserved. **0b1111** Reserved.	xxxx
[23:16]	PMG	Program monitoring group. Set to the PMG on an error that captures PMG. Otherwise, set to 0x00 on an error that does not capture PMG.	8{x}
[15:0]	PARTID_MON	PARTID or monitor. Set to the PARTID on an error that captures PARTID. Set to the monitor index on an error that captures MON. On an error that captures neither PARTID nor MON, this field is set to 0x0000.	16{x}

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_ESR_s must only be accessible from the Secure MPAM feature page. MPAMF_ESR_ns must only be accessible from the Non-secure MPAM feature page.

MPAMF_ESR_s and MPAMF_ESR_ns must be separate registers. The Secure instance (MPAMF_ESR_s) accesses the error status used for Secure PARTIDs, and the Non-secure instance (MPAMF_ESR_ns) accesses the error status used for Non-secure PARTIDs.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMF_ESR_s must only be accessible from the Secure MPAM feature page. MPAMF_ESR_ns must only be accessible from the Non-secure MPAM feature page.

MPAMF_ESR_s and MPAMF_ESR_ns must be separate registers. The Secure instance (MPAMF_ESR_s) accesses the error status used for Secure PARTIDs, and the Non-secure instance (MPAMF_ESR_ns) accesses the error status used for Non-secure PARTIDs.

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

The power and reset domain of each MSC component is specific to that component.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-24: ext_mpamcfg_part_sel bit assignments

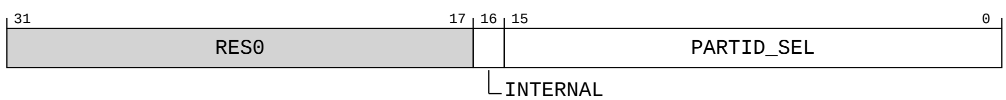


Table B-55: MPAMCFG_PART_SEL bit descriptions

Bits	Name	Description	Reset
[31:17]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[16]	INTERNAL	Internal PARTID. If ext-MPAMF_IDR.HAS_PARTID_NRW == `0b1`: **`0b0`** PARTID_SEL is interpreted as a request PARTID and ignored except for use with ext-MPAMCFG_INTPARTID register access. **`0b1`** PARTID_SEL is interpreted as an internal PARTID and used for access to MPAMCFG control settings except for ext-MPAMCFG_INTPARTID. If PARTID narrowing is implemented as indicated by ext-MPAMF_IDR.HAS_PARTID_NRW = 1, when accessing other MPAMCFG registers the value of the MPAMCFG_PART_SEL.INTERNAL bit is checked for these conditions: • When the ext-MPAMCFG_INTPARTID register is read or written, if the value of MPAMCFG_PART_SEL.INTERNAL is not 0, an Unexpected_INTERNAL error is set in ext-MPAMF_ESR. • When an MPAMCFG register other than ext-MPAMCFG_INTPARTID is read or written, if the value of MPAMCFG_PART_SEL.INTERNAL is not 1, ext-MPAMF_ESR is set to indicate an intPARTID_Range error. In either error case listed here, the value returned by a read operation is **UNPREDICTABLE**, and the control settings are not affected by a write. Access to this field is: **RAZ/WI**	`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}`

Access

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMCFG_PART_SEL_s must only be accessible from the Secure MPAM feature page. MPAMCFG_PART_SEL_ns must only be accessible from the Non-secure MPAM feature page.

MPAMCFG_PART_SEL_s and MPAMCFG_PART_SEL_ns must be separate registers. The Secure instance (MPAMCFG_PART_SEL_s) accesses the PARTID selector used for Secure PARTIDs, and the Non-secure instance (MPAMCFG_PART_SEL_ns) accesses the PARTID selector used for Non-secure PARTIDs.

Accessibility

This register is within the MPAM feature page memory frames. In a system that supports Secure and Non-secure memory maps, there must be both Secure and Non-secure MPAM feature pages.

MPAMCFG_PART_SEL_s must only be accessible from the Secure MPAM feature page. MPAMCFG_PART_SEL_ns must only be accessible from the Non-secure MPAM feature page.

MPAMCFG_PART_SEL_s and MPAMCFG_PART_SEL_ns must be separate registers. The Secure instance (MPAMCFG_PART_SEL_s) accesses the PARTID selector used for Secure PARTIDs, and the Non-secure instance (MPAMCFG_PART_SEL_ns) accesses the PARTID selector used for Non-secure PARTIDs.

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

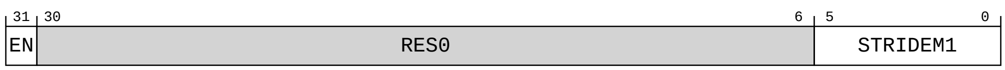


Table B-58: MPAMCFG_MBW_PROP_ns bit descriptions

Bits	Name	Description	Reset
[31]	EN	Enable proportional stride bandwidth partitioning. **0b0** The selected partition is not regulated by proportional stride bandwidth partitioning. **0b1** 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

```
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-26: ext_mpamcfg_mbw_prop_s bit assignments

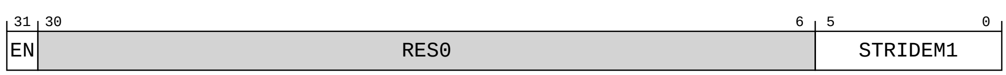


Table B-60: MPAMCFG_MBW_PROP_s bit descriptions

Bits	Name	Description	Reset
[31]	EN	Enable proportional stride bandwidth partitioning. **0b0** The selected partition is not regulated by proportional stride bandwidth partitioning. **0b1** 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

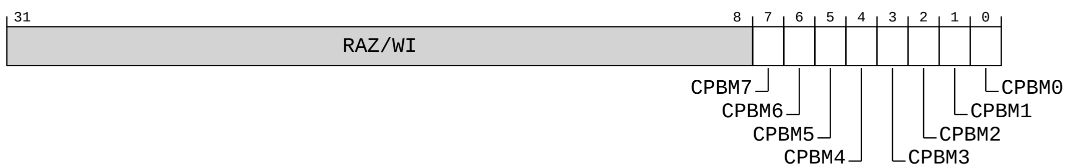


Table B-62: MPAMCFG_CPBM_ns bit descriptions

Bits	Name	Description	Reset
[31:8]	RAZ/WI	Reserved	RAZ/WI
[7]	CPBM7	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[6]	CPBM6	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[5]	CPBM5	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

Bits	Name	Description	Reset
[4]	CPBM4	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[3]	CPBM3	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[2]	CPBM2	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[1]	CPBM1	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[0]	CPBM0	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

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

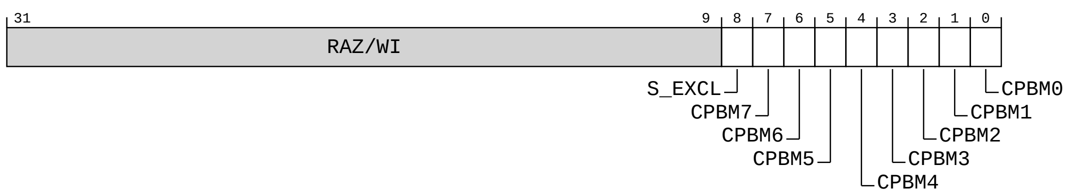


Table B-64: MPAMCFG_CPBM_s bit descriptions

Bits	Name	Description	Reset
[31:9]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[8]	S_EXCL	Exclusive Secure CPBM enable. If set, all portions enabled in the Secure MPAMCFG_CPBM_s register will prevent corresponding portions enabled in the MPAMCFG_CPBM_ns register from taking effect. **0b0** Each set MPAMCFG_CPBM_s bit has no effect on the corresponding MPAMCFG_CPBM_ns bit. **0b1** Each set MPAMCFG_CPBM_s bit masks the corresponding MPAMCFG_CPBM_ns bit from taking effect.	0b0
[7]	CPBM7	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[6]	CPBM6	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[5]	CPBM5	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[4]	CPBM4	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[3]	CPBM3	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

Bits	Name	Description	Reset
[2]	CPBM2	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[1]	CPBM1	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1
[0]	CPBM0	Each bit, CPBM<n>, grants permission to the PARTID to allocate cache lines within cache portion n. **0b0** The PARTID is not permitted to allocate into cache portion n. **0b1** The PARTID is permitted to allocate within cache portion n. The number of bits in the cache portion partitioning bit map of this component is given in ext-MPAMF_CPOR_IDR.CPBM_WD.	0b1

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 C1-DSU. For more information about a register, click on the register name in the table.

Note

- If Realm Management Extension (RME) is enabled, meaning that the cluster is in Direct connect, these registers are not present. For more information on enabling RME, see 1.4.1 Realm management extension on page 26.

- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- 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-66: CLUSTERRAS registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0	CLUSTERRAS_ERR0FR	See individual bit resets.	64-bit	Error Record <n> Feature Register	No
0x8	CLUSTERRAS_ERR0CTLR	See individual bit resets.	64-bit	Error Record <n> Control Register	No
0x010	CLUSTERRAS_ERR0STATUS	See individual bit resets.	64-bit	Error Record <n> Primary Status Register	No
0x018	CLUSTERRAS_ERR0ADDR	See individual bit resets.	64-bit	Error Record Address Register	No
0x20	CLUSTERRAS_ERR0MISC0	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 0	No
0x28	CLUSTERRAS_ERR0MISC1	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 1	No
0x30	CLUSTERRAS_ERR0MISC2	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 2	No
0x38	CLUSTERRAS_ERR0MISC3	See individual bit resets.	64-bit	Error Record <n> Miscellaneous Register 3	No
0x800	CLUSTERRAS_ERR0PFGF	See individual bit resets.	64-bit	Error Record <n> Pseudo-fault Generation Feature Register	No
0x808	CLUSTERRAS_ERR0PFGCTL	See individual bit resets.	64-bit	Error Record <n> Pseudo-fault Generation Control Register	No
0x810	CLUSTERRAS_ERR0PFGCDN	See individual bit resets.	64-bit	Error Record <n> Pseudo-fault Generation Countdown Register	No
0xE00	CLUSTERRAS_ERRGSR	See individual bit resets.	64-bit	Error Group Status Register	No
0xE10	CLUSTERRAS_ERRIIDR	See individual bit resets.	32-bit	Implementation Identification Register	No
0xFA8	CLUSTERRAS_ERRDEVAFF	See individual bit resets.	64-bit	Device Affinity Register	No
0xFBC	CLUSTERRAS_ERRDEVARCH	See individual bit resets.	32-bit	Device Architecture Register	No
0xFC8	CLUSTERRAS_ERRDEVID	See individual bit resets.	32-bit	Device Configuration Register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFD0	CLUSTERRAS_ERRPIDR4	See individual bit resets.	32-bit	Peripheral Identification Register 4	No
0xFD4	CLUSTERRAS_ERRPIDR5	See individual bit resets.	32-bit	Peripheral Identification Register 5	No
0xFD8	CLUSTERRAS_ERRPIDR6	See individual bit resets.	32-bit	Peripheral Identification Register 6	No
0xFDC	CLUSTERRAS_ERRPIDR7	See individual bit resets.	32-bit	Peripheral Identification Register 7	No
0xFE0	CLUSTERRAS_ERRPIDR0	See individual bit resets.	32-bit	Peripheral Identification Register 0	No
0xFE4	CLUSTERRAS_ERRPIDR1	See individual bit resets.	32-bit	Peripheral Identification Register 1	No
0xFE8	CLUSTERRAS_ERRPIDR2	See individual bit resets.	32-bit	Peripheral Identification Register 2	No
0xFEC	CLUSTERRAS_ERRPIDR3	See individual bit resets.	32-bit	Peripheral Identification Register 3	No
0xFF0	CLUSTERRAS_ERRCIDR0	See individual bit resets.	32-bit	Component Identification Register 0	No
0xFF4	CLUSTERRAS_ERRCIDR1	See individual bit resets.	32-bit	Component Identification Register 1	No
0xFF8	CLUSTERRAS_ERRCIDR2	See individual bit resets.	32-bit	Component Identification Register 2	No
0xFFC	CLUSTERRAS_ERRCIDR3	See individual bit resets.	32-bit	Component Identification Register 3	No

B.1.3.1 CLUSTERRAS_ERR0FR, Error Record <n> Feature Register

Defines whether error record <n> is the first record owned by a node:

- If error record <n> is the first error record owned by a node, then ERR<n>FR.ED is not 0b00.
- If error record <n> is not the first error record owned by a node, then ERR<n>FR.ED is 0b00.

If error record <n> is the first record owned by the node, defines which of the common architecturally-defined features are implemented by the node and, of the implemented features, which are software programmable.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0xxx xx00 1001 xx00 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 B-29: ext_clusterras_err0fr bit assignments

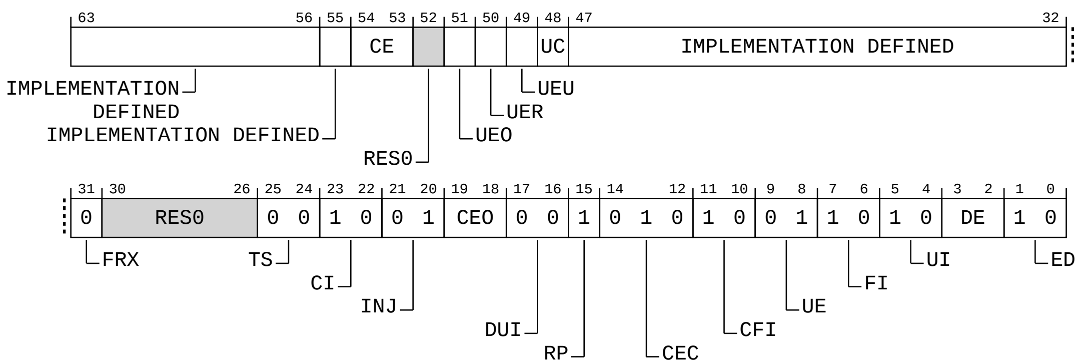


Table B-67: CLUSTERRAS_ERR0FR bit descriptions

Bits	Name	Description	Reset
[63:56]	None	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	8{x}
[55]	None	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	x
[54:53]	CE	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	xx
[52]	**RES0**	Reserved	**RES0**
[51]	UEO	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	x
[50]	UER	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	x
[49]	UEU	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	x
[48]	UC	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	x
[47:32]	None	Reserved for identifying **IMPLEMENTATION DEFINED** controls.	16{x}

Bits	Name	Description	Reset
[31]	FRX	Feature Register extension. Defines whether ERR<n>FR[63:48] are architecturally defined. Defined values are: **0b0** ERR<n>FR[63:48] are IMPLEMENTATION DEFINED.	`0b0`
[30:26]	RES0	Reserved	RES0
[25:24]	TS	Timestamp Extension. Indicates whether, for each error record <m> owned by this node, ERR<m>MISC3 is used as the timestamp register, and, if it is, the timebase used by the timestamp. **0b00** 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 by the node. **0b10** Critical error interrupt is supported and controllable using ext-CLUSTERRAS_ERR<n>CTLR.CI. All other values are reserved.	`0b10`
[21:20]	INJ	Fault Injection Extension. Indicates whether the Common Fault Injection Model Extension is implemented by the node. **0b01** Supports the Common Fault Injection Model Extension. See ext-CLUSTERRAS_ERR<n>PFGF for more information. All other values are reserved.	`0b01`
[19:18]	CEO	Corrected Error overwrite. Indicates the behavior of the node when a second or subsequent Corrected error is recorded and a first Corrected error has previously been recorded by an error record <m> owned by the node. **0b00** Keeps the previous error syndrome. **0b01** If ERR<m>STATUS.OF is 1 before the Corrected error is counted, then the error record keeps the previous syndrome. Otherwise the previous syndrome is overwritten. All other values are reserved. The second or subsequent Corrected error is counted by the Corrected error counter, regardless of the value of this field. If counting the error causes unsigned overflow of the counter, then ERR<m>STATUS.OF is set to 1. This means that, if no other error is subsequently recorded that overwrites the syndrome: • If ERR<n>FR.CEO is `0b00`, the error record holds the syndrome for the first recorded Corrected error. • If ERR<n>FR.CEO is `0b01`, the error record holds the syndrome for the most recently recorded Corrected error before the counter overflows.	The reset values can be the following: `0b00`, `0b01`, respective to the value.

Bits	Name	Description	Reset
[17:16]	DUI	Error recovery interrupt for deferred errors control. Indicates whether the enabling and disabling of error recovery interrupts on deferred errors is supported by the node. **0b00** Does not support the enabling and disabling of error recovery interrupts on deferred errors. ext-CLUSTERRAS_ERR<n>CTLR.DUI is **RES0**. All other values are reserved.	0b00
[15]	RP	Repeat counter. Indicates whether the node implements a second Corrected error counter in ERR<m>MISC0 for each error record <m> owned by the node that can record countable errors. **0b1** Implements a first (repeat) counter and a second (other) counter in ERR<m>MISC0 for each error record <m> owned by the node that can record countable errors. The repeat counter is the same size as the primary error counter.	0b1
[14:12]	CEC	Corrected Error Counter. Indicates whether the node implements the standard format Corrected error counter mechanisms in ERR<m>MISC0 for each error record <m> owned by the node that can record countable errors. **0b010** Implements an 8-bit Corrected error counter in ERR<m>MISC0[39:32] for each error record <m> owned by the node that can record countable errors. All other values are reserved. **Note:** Implementations might include other error counter models, or might include the standard format model and not indicate this in ERR<n>FR.	0b010
[11:10]	CFI	Fault handling interrupt for corrected errors control. Indicates whether the enabling and disabling of fault handling interrupts on corrected errors is supported by the node. **0b10** Enabling and disabling of fault handling interrupts on corrected errors is supported and controllable using ext-CLUSTERRAS_ERR<n>CTLR.CFI. All other values are reserved.	0b10
[9:8]	UE	In-band error reponse (External Abort). Indicates whether the in-band error response and associated controls are implemented by the node. **0b01** In-band error response is supported and always enabled. ext-CLUSTERRAS_ERR<n>CTLR.UE is **RES0**. It is **IMPLEMENTATION DEFINED** whether an uncorrected error that is deferred and recorded as Deferred error, but is not deferred to the Requester, will signal an in-band error response to the Requester.	0b01
[7:6]	FI	Fault handling interrupt. Indicates whether the fault handling interrupt and associated controls are implemented by the node. **0b10** Fault handling interrupt is supported and controllable using ext-CLUSTERRAS_ERR<n>CTLR.FI.	0b10

Bits	Name	Description	Reset
[5:4]	UI	Error recovery interrupt for uncorrected errors. Indicates whether the error handling interrupt and associated controls are implemented by the node. **0b10** Error handling interrupt is supported and controllable using ext-CLUSTERRAS_ERR<n>CTLR.UI.	0b10
[3:2]	DE	This is the before description. This is the after description	0b01
[1:0]	ED	Error reporting and logging. Indicates error record <n> is a normal record and the first record owned the node, and whether the node implements the controls for enabling and disabling error reporting and logging. Defined values are: **0b10** Error reporting and logging is controllable using ext-CLUSTERRAS_ERR<n>CTLR.ED. All other values are reserved.	0b10

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x0	ERR0FR	None

This interface is accessible as follows:

RO

B.1.3.2 CLUSTERRAS_ERR0CTLR, Error Record <n> Control Register

The error control register contains enable bits for the node that writes to this record:

- Enabling error detection and correction.
- Enabling the critical error, error recovery, and fault handling interrupts.
- Enabling in-band error response for uncorrected errors.

For each bit, if the node does not support the feature, then the bit is **RES0**. The definition of each record is IMPLEMENTATION DEFINED.

Configurations

ext-CLUSTERRAS_ERR<n>FR describes the features implemented by the node.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x8

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx0x xxx0 xxxx 00x0
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-30: ext_clusterras_err0ctlr bit assignments

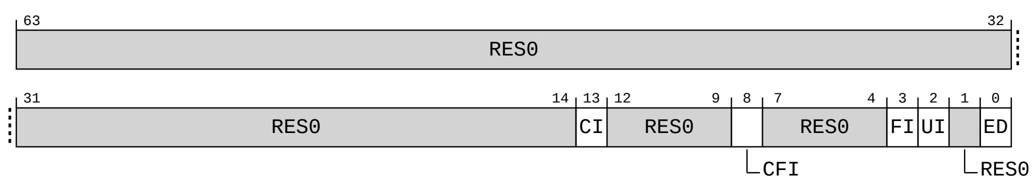


Table B-69: CLUSTERRAS_ERR0CTLR bit descriptions

Bits	Name	Description	Reset
[63:14]	RES0	Reserved	RES0
[13]	CI	Critical error interrupt enable. When enabled, the critical error interrupt is generated for a critical error condition. **0b0** Critical error interrupt not generated for critical errors. Critical errors are treated as Uncontained errors. **0b1** Critical error interrupt generated for critical errors.	0b0
[12:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8]	CFI	Fault handling interrupt for corrected error events enable. When ext-CLUSTERRAS_ERR<n>FR.CFI == `0b10`, this control applies to errors on both reads and writes. When enabled, the fault handling interrupt is generated for all corrected error events. **`0b0`** Fault handling interrupt not generated for corrected error events. **`0b1`** Fault handling interrupt generated for corrected error events. If the node implements a corrected error counter or counters, then a corrected error event is defined as follows: • A corrected error event occurs when a counter overflows and sets a counter overflow flag to 1. • It is **UNPREDICTABLE** whether a corrected error event occurs when a software write sets a counter overflow flag to 1. • It is **UNPREDICTABLE** whether a corrected error event occurs when a counter overflows and the overflow flag was previously set to 1. Otherwise, a corrected error event occurs when the error record records an error as a Corrected error. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	`0b0`
[7:4]	**RES0**	Reserved	**RES0**
[3]	FI	Fault handling interrupt enable. When ext-CLUSTERRAS_ERR<n>FR.FI == `0b10`, this control applies to errors on both reads and writes. When enabled: • The fault handling interrupt is generated for all errors recorded as either Deferred error or Uncorrected error. • If the fault handling interrupt control for corrected error events, ERR<n>CTLR.CFI, is not implemented, then the fault handling interrupt is generated for all corrected error events. **`0b0`** Fault handling interrupt disabled. **`0b1`** Fault handling interrupt enabled. See ERR<n>CTLR.CFI for more information on corrected error events. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	`0b0`

Bits	Name	Description	Reset
[2]	UI	Uncorrected error recovery interrupt enable. When ext-CLUSTERRAS_ERR<n>FR.UI == `0b10`, this control applies to errors arising from both reads and writes. When enabled, the error recovery interrupt is generated for all errors recorded as Uncorrected error. **0b0** Error recovery interrupt disabled. **0b1** Error recovery interrupt enabled. The interrupt is generated even if the error syndrome is discarded because the error record already records a higher priority error.	`0b0`
[1]	**RES0**	Reserved	**RES0**
[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. Arm recommends that, when disabled, correct error detection and correction codes are written for writes, unless disabled by an **IMPLEMENTATION DEFINED** control for error injection. **0b0** Error reporting disabled. **0b1** Error reporting enabled. It is IMPLEMENTATION DEFINED whether the node fully disables error detection and correction when reporting is disabled. That is, even with error reporting disabled, the node might continue to silently correct errors. Uncorrected errors might result in corrupt data being silently propagated by the node. **Note:** If this node requires initialization after Cold reset to prevent signaling false errors, then Arm recommends this field is set to 0 on Cold reset, meaning errors are not reported from Cold reset. This allows boot software to initialize a node without signaling errors. Software can enable error reporting after the node is initialized. Otherwise, the Cold reset value is IMPLEMENTATION DEFINED. If the Cold reset value is 1, the reset values of other controls in this register are also IMPLEMENTATION DEFINED and should not be **UNKNOWN**.	`0b0`

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x8	ERR0CTLR	None

This interface is accessible as follows:

RW

B.1.3.3 CLUSTERRAS_ERR0STATUS, Error Record <n> Primary Status Register

Contains status information for error record <n>, 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 reported 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:

- ERR<n>STATUS.{AV, V, MV} are valid bits that define whether error record <n> registers are valid.
- ERR<n>STATUS.{UE, OF, CE, DE, UET} encode the types of error or errors recorded.
- ERR<n>STATUS.{CI, ER, PN, IERR, SERR} are syndrome fields.

Configurations

ERR<q>FR describes the features implemented by the node that owns error record <n>. <q> is the index of the first error record owned by the same node as error record <n>. If the node owns a single record then q = n.

For IMPLEMENTATION DEFINED fields in ERR<n>STATUS, writing zero returns the error record to an initial quiescent state.

In particular, if any IMPLEMENTATION DEFINED syndrome fields might generate a Fault Handling or Error Recovery Interrupt request, writing zero is sufficient to deactivate the Interrupt request.

Fields that are read-only, nonzero, and ignore writes are compliant with this requirement.

Note

Arm recommends that any IMPLEMENTATION DEFINED syndrome field that can generate a Fault Handling, Error Recovery, Critical, or IMPLEMENTATION DEFINED, interrupt request is disabled at Cold reset and is enabled by software writing an IMPLEMENTATION DEFINED nonzero value to an IMPLEMENTATION DEFINED field in ERR<q>CTLR.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x010

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 00xx x0xx 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-31: ext_clusterras_err0status bit assignments

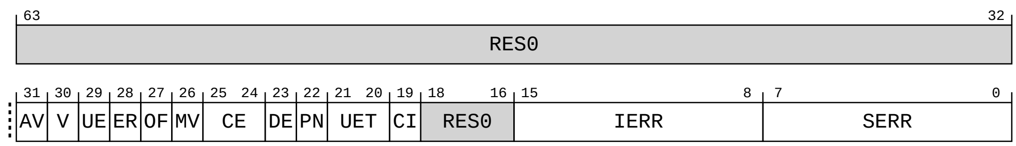


Table B-71: CLUSTERRAS_ERR0STATUS bit descriptions

Bits	Name	Description	Reset
[63:32]	RES0	Reserved	RES0
[31]	AV	Address Valid. **0b0** ext-CLUSTERRAS_ERR<n>ADDR not valid. **0b1** ext-CLUSTERRAS_ERR<n>ADDR contains an address associated with the highest priority error recorded by this record. Access to this field is: W1C	0b0
[30]	V	Status Register Valid. **0b0** ERR<n>STATUS not valid. **0b1** ERR<n>STATUS valid. At least one error has been recorded. Access to this field is: W1C	0b0

Bits	Name	Description	Reset
[29]	UE	Uncorrected Error. **0b0** No errors have been detected, or all detected errors have been either corrected or deferred. **0b1** At least one detected error was not corrected and not deferred. When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write 1 to this field to clear this field to zero. **When ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: W1C	x

Bits	Name	Description	Reset
[28]	ER	**When in-band error responses can be returned for a Deferred error** Error Reported. **`0b0`** No in-band error response (External Abort) signaled to the Requester making the access or other transaction. **`0b1`** An in-band error response was signaled by the component to the Requester making the access or other transaction. This can be because any of the following are true: • The ERR<q>CTLR.UE field, or applicable one of the ERR<q>CTLR.{WUE, RUE} fields, is implemented and was 1 when an error was detected and not corrected. • The ERR<q>CTLR.{WUE, RUE, UE} fields are not implemented and the component always reports errors. **Note:** An in-band error response signaled by the component might be masked and not generate any exception. It is IMPLEMENTATION DEFINED whether an uncorrected error that is deferred and recorded as a Deferred error, but is not deferred to the Requester, can signal an in-band error response to the Requester, causing this field to be set to 1. **When in-band error responses are never be returned for a Deferred error** Error Reported. **`0b0`** No in-band error response (External Abort) signaled to the Requester making the access or other transaction. **`0b1`** An in-band error response was signaled by the component to the Requester making the access or other transaction. This can be because any of the following are true: • The ERR<q>CTLR.UE field, or applicable one of the ERR<q>CTLR.{WUE, RUE} fields, is implemented and was 1 when an error was detected and not corrected. • The ERR<q>CTLR.{WUE, RUE, UE} fields are not implemented and the component always reports errors. **Note:** An in-band error response signaled by the component might be masked and not generate any exception. It is IMPLEMENTATION DEFINED whether an uncorrected error that is deferred and recorded as a Deferred error, but is not deferred to the Requester, can signal an in-band error response to the Requester, causing this field to be set to 1. **Otherwise** RES0	`x`

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>Reset</th></tr>
<tr><td>[27]</td><td>OF</td><td>Overflow.

Indicates that multiple errors have been detected. This field is set to 1 when one of the following occurs:
• A Corrected error counter is implemented, an error is counted, and the counter overflows.
• ERR<n>STATUS.V was previously 1, a Corrected error counter is not implemented, and a Corrected error is recorded.
• ERR<n>STATUS.V was previously 1, and a type of error other than a Corrected error is recorded.

Otherwise, this field is unchanged when an error is recorded.

If a Corrected error counter is implemented, then:
• A direct write that modifies the counter overflow flag indirectly might set this field to an UNKNOWN value.
• A direct write to this field that clears this field to zero might indirectly set the counter overflow flag to an UNKNOWN value.

<code>0b0</code>
Since this field was last cleared to zero, no error syndrome has been discarded and, if a Corrected error counter is implemented, it has not overflowed.
<code>0b1</code>
Since this field 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.

When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write 1 to this field to clear this field to zero.

When ext-CLUSTERRAS_ERR<n>STATUS.V == '0'
Access to this field is: UNKNOWN/WI
Otherwise
Access to this field is: W1C</td><td><code>x</code></td></tr>
<tr><td>[26]</td><td>MV</td><td>Miscellaneous Registers Valid.
<code>0b0</code>
ERR<n>MISC<m> not valid.
<code>0b1</code>
The contents of the ERR<n>MISC<m> registers contain additional information for an error recorded by this record.

Note:
If the ERR<n>MISC<m> registers can contain additional information for a previously recorded error, then the contents must be self-describing to software or a user. For example, certain fields might relate only to Corrected errors, and other fields only to the most recent error that was not discarded.

Access to this field is: W1C</td><td><code>0b0</code></td></tr>
</table>

Bits	Name	Description	Reset
[25:24]	CE	Corrected Error. **0b00** No errors were corrected. **0b01** At least one transient error was corrected. **0b10** At least one error was corrected. **0b11** At least one persistent error was corrected. The mechanism by which a component or node detects whether a Corrected error is transient or persistent is IMPLEMENTATION DEFINED. If no such mechanism is implemented, then the node sets this field to `0b10` when a corrected error is recorded. When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write ones to this field to clear this field to zero. **When ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: W1C	`xx`
[23]	DE	Deferred Error. **0b0** No errors were deferred. **0b1** At least one error was not corrected and deferred. Support for deferring errors is IMPLEMENTATION DEFINED. When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write 1 to this field to clear this field to zero. **When ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: W1C	`x`

Bits	Name	Description	Reset
[22]	PN	Poison. **0b0** Uncorrected error or Deferred error recorded because a corrupt value was detected, for example, by an error detection code (EDC), or Corrected error recorded. **0b1** Uncorrected error or Deferred error recorded because a poison value was detected. When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write 1 to this field to clear this field to zero. **When ext-CLUSTERRAS_ERR<n>STATUS.V == '0' || ext-CLUSTERRAS_ERR<n>STATUS.[DE,UE] == '00'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: W1C	x
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. **0b00** Uncorrected error, Uncontainable error (UC). **0b01** Uncorrected error, Unrecoverable error (UEU). **0b10** Uncorrected error, Latent or Restartable error (UEO). **0b11** Uncorrected error, Signaled or Recoverable error (UER). UER can mean either Signaled or Recoverable error, and UEO can mean either Latent or Restartable error. When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write ones to this field to clear this field to zero. **When ext-CLUSTERRAS_ERR<n>STATUS.V == '0' || ext-CLUSTERRAS_ERR<n>STATUS.UE == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: W1C	xx
[19]	CI	Critical Error. Indicates whether a critical error condition has been recorded. **0b0** No critical error condition. **0b1** Critical error condition. When clearing ERR<n>STATUS.V to 0, if this field is nonzero, then Arm recommends that software write 1 to this field to clear this field to zero. **When ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: W1C	x

Bits	Name	Description	Reset
[18:16]	RES0	Reserved	RES0
[15:8]	IERR	IMPLEMENTATION DEFINED error code. Used with any primary error code ERR<n>STATUS.SERR value. Further IMPLEMENTATION DEFINED information can be placed in the ERR<n>MISC<m> registers. The implemented set of valid values that this field can take is IMPLEMENTATION DEFINED. If any value not in this set is written to this register, then the value read back from this field is UNKNOWN. **Note:** This means that one or more bits of this field might be implemented as fixed read-as-zero or read-as-one values. **When Text("the Common Fault Injection Model Extension is not implemented by the node that owns this error record") && ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **When ext-CLUSTERRAS_ERR<q>PFGF.SYN == '0' && ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: RW	8{x}

[7:0]	SERR	Architecturally-defined primary error code. The primary error code might be used by a fault handling agent to triage an error without requiring device-specific code. For example, to count and threshold corrected errors in software, or generate a short log entry.
		`0b00000000`
		No error.
		`0b00000001`
		IMPLEMENTATION DEFINED error.
		`0b00000010`
		Data value from (non-associative) internal memory. For example, ECC from on-chip SRAM or buffer.
		`0b00000011`
		IMPLEMENTATION DEFINED pin. For example, nSEI pin.
		`0b00000100`
		Assertion failure. For example, consistency failure.
		`0b00000101`
		Error detected on internal data path. For example, parity on ALU result.
		`0b00000110`
		Data value from associative memory. For example, ECC error on cache data.
		`0b00000111`
		Address/control value from associative memory. For example, ECC error on cache tag.
		`0b00001000`
		Data value from a TLB. For example, ECC error on TLB data.
		`0b00001001`
		Address/control value from a TLB. For example, ECC error on TLB tag.
			8{x}

Bits	Name	Description	Reset

[7:0] continued	SERR	`0b00001010`
		Data value from producer. For example, parity error on write data bus.
		`0b00001011`
		Address/control value from producer. For example, parity error on address bus.
		`0b00001100`
		Data value from (non-associative) external memory. For example, ECC error in SDRAM.
		`0b00001101`
		Illegal address (software fault). For example, access to unpopulated memory.
		`0b00001110`
		Illegal access (software fault). For example, byte write to word register.
		`0b00001111`
		Illegal state (software fault). For example, device not ready.
		`0b00010000`
		Internal data register. For example, parity on a SIMD&FP register. For a PE, all general-purpose, stack pointer, SIMD&FP, and SVE registers are data registers.
		`0b00010001`
		Internal control register. For example, Parity on a System register. For a PE, all registers other than general-purpose, stack pointer, SIMD&FP, and SVE registers are control registers.
		`0b00010010`
		Error response from Completer of access. For example, error response from cache write-back.
		`0b00010011`
		External timeout. For example, timeout on interaction with another component.
			`8{x}`

Bits	Name	Description	Reset
[7:0] continued	SERR	**0b00010100** Internal timeout. For example, timeout on interface within the component. **0b00010101** Deferred error from Completer not supported at Requester. For example, poisoned data received from the Completer of an access by a Requester that cannot defer the error further. **0b00010110** Deferred error from Requester not supported at Completer. For example, poisoned data received from the Requester of an access by a Completer that cannot defer the error further. **0b00010111** Deferred error from Completer passed through. For example, poisoned data received from the Completer of an access and returned to the Requester. **0b00011000** Deferred error from Requester passed through. For example, poisoned data received from the Requester of an access and deferred to the Completer. **0b00011001** Error recorded by PCIe error logs. Indicates that the component has recorded an error in a PCIe error log. This might be the PCIe device status register, AER, DVSEC, or other mechanisms defined by PCIe. **0b00011010** Other internal error. For example, parity error on internal state of the component that is not covered by another primary error code. All other values are reserved. The implemented set of valid values that this field can take is IMPLEMENTATION DEFINED. If any value not in this set is written to this register, then the value read back from this field is UNKNOWN. **Note:** This means that one or more bits of this field might be implemented as fixed read-as-zero or read-as-one values. **When Text("the Common Fault Injection Model Extension is not implemented by the node that owns this error record") && ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **When ext-CLUSTERRAS_ERR<q>PFGF.SYN == '0' && ext-CLUSTERRAS_ERR<n>STATUS.V == '0'** Access to this field is: UNKNOWN/WI **Otherwise** Access to this field is: RW	8{x}

Access

ERR<n>STATUS.{AV, V, UE, ER, OF, MV, CE, DE, PN, UET, CI} are write-one-to-clear (W1C) fields, meaning writes of zero are ignored, and a write of one or all-ones to the field clears the field to zero. ERR<n>STATUS.{IERR, SERR} are read/write (RW) fields, although the set of implemented valid values is **IMPLEMENTATION DEFINED**. See also ext-CLUSTERRAS_ERR<n>PFGF.SYN.

After reading ERR<n>STATUS, software must clear the valid fields in the register to allow new errors to be recorded. However, between reading the register and clearing the valid fields, a new error might have overwritten the register. To prevent this error being lost by software, the register prevents updates to fields that might have been updated by a new error.

When RAS System Architecture v1.0 is implemented:

- Writes to ERR<n>STATUS.{UE, DE, CE} are ignored if ERR<n>STATUS.OF is 1 and is not being cleared to 0.
- Writes to ERR<n>STATUS.V are ignored if any of ERR<n>STATUS.{UE, DE, CE} are nonzero and are not being cleared to zero.
- Writes to ERR<n>STATUS.{AV, MV} and the ERR<n>STATUS.{ER, PN, UET, IERR, SERR} syndrome fields are ignored if the highest priority nonzero error status field is not being cleared to zero. The error status fields in priority order from highest to lowest, are ERR<n>STATUS.UE, ERR<n>STATUS.DE, and ERR<n>STATUS.CE.

When RAS System Architecture v1.1 is implemented, a write to the register is ignored if all of:

- Any of ERR<n>STATUS.{V, UE, OF, CE, DE} are nonzero before the write.
- The write does not clear the nonzero ERR<n>STATUS.{V, UE, OF, CE, DE} fields to zero by writing ones to the applicable field or fields.

Some of the fields in ERR<n>STATUS are also defined as **UNKNOWN** where certain combinations of ERR<n>STATUS.{V, DE, UE} are zero. The rules for writes to ERR<n>STATUS allow a node to implement such a field as a fixed read-only value.

For example, when RAS System Architecture v1.1 is implemented, a write to ERR<n>STATUS when ERR<n>STATUS.V is 1 results in either ERR<n>STATUS.V field being cleared to zero, or ERR<n>STATUS.V not changing. Since all fields in ERR<n>STATUS, other than ERR<n>STATUS.{AV, V, MV}, usually read as **UNKNOWN** values when ERR<n>STATUS.V is zero, this means those fields can be implemented as read-only if applicable.

To ensure correct and portable operation, when software is clearing the valid fields in the register to allow new errors to be recorded, Arm recommends that software performs the following sequence of operations in order:

- Read ERR<n>STATUS and determine which fields need to be cleared to zero.
- In a single write to ERR<n>STATUS:
    - Write ones to all the W1C fields that are nonzero in the read value.
    - Write zero to all the W1C fields that are zero in the read value.
    - Write zero to all the RW fields.
- Read back ERR<n>STATUS after the write to confirm no new fault has been recorded.

Otherwise, these fields might not have the correct value when a new fault is recorded.

Accessibility

ERR<n>STATUS.{AV, V, UE, ER, OF, MV, CE, DE, PN, UET, CI} are write-one-to-clear (W1C) fields, meaning writes of zero are ignored, and a write of one or all-ones to the field clears the field to zero. ERR<n>STATUS.{IERR, SERR} are read/write (RW) fields, although the set of implemented valid values is IMPLEMENTATION DEFINED. See also ext-CLUSTERRAS_ERR<n>PFGF.SYN.

After reading ERR<n>STATUS, software must clear the valid fields in the register to allow new errors to be recorded. However, between reading the register and clearing the valid fields, a new error might have overwritten the register. To prevent this error being lost by software, the register prevents updates to fields that might have been updated by a new error.

When RAS System Architecture v1.0 is implemented:

- Writes to ERR<n>STATUS.{UE, DE, CE} are ignored if ERR<n>STATUS.OF is 1 and is not being cleared to 0.
- Writes to ERR<n>STATUS.V are ignored if any of ERR<n>STATUS.{UE, DE, CE} are nonzero and are not being cleared to zero.
- Writes to ERR<n>STATUS.{AV, MV} and the ERR<n>STATUS.{ER, PN, UET, IERR, SERR} syndrome fields are ignored if the highest priority nonzero error status field is not being cleared to zero. The error status fields in priority order from highest to lowest, are ERR<n>STATUS.UE, ERR<n>STATUS.DE, and ERR<n>STATUS.CE.

When RAS System Architecture v1.1 is implemented, a write to the register is ignored if all of:

- Any of ERR<n>STATUS.{V, UE, OF, CE, DE} are nonzero before the write.
- The write does not clear the nonzero ERR<n>STATUS.{V, UE, OF, CE, DE} fields to zero by writing ones to the applicable field or fields.

Some of the fields in ERR<n>STATUS are also defined as UNKNOWN where certain combinations of ERR<n>STATUS.{V, DE, UE} are zero. The rules for writes to ERR<n>STATUS allow a node to implement such a field as a fixed read-only value.

For example, when RAS System Architecture v1.1 is implemented, a write to ERR<n>STATUS when ERR<n>STATUS.V is 1 results in either ERR<n>STATUS.V field being cleared to zero, or ERR<n>STATUS.V not changing. Since all fields in ERR<n>STATUS, other than ERR<n>STATUS.{AV, V, MV}, usually read as UNKNOWN values when ERR<n>STATUS.V is zero, this means those fields can be implemented as read-only if applicable.

To ensure correct and portable operation, when software is clearing the valid fields in the register to allow new errors to be recorded, Arm recommends that software performs the following sequence of operations in order:

- Read ERR<n>STATUS and determine which fields need to be cleared to zero.
- In a single write to ERR<n>STATUS:
    - Write ones to all the W1C fields that are nonzero in the read value.
    - Write zero to all the W1C fields that are zero in the read value.
    - Write zero to all the RW fields.
- Read back ERR<n>STATUS after the write to confirm no new fault has been recorded.

Otherwise, these fields might not have the correct value when a new fault is recorded.

Component	Offset	Instance	Range
CLUSTERRAS	0x010	ERR0STATUS	None

This interface is accessible as follows:

When ext-CLUSTERRAS_ERR<n>STATUS.V != '0' && Text("ERR<n>STATUS.V is not being cleared to 0b0 in the same write")

RO

When ext-CLUSTERRAS_ERR<n>STATUS.UE != '0' && Text("ERR<n>STATUS.UE is not being cleared to 0b0 in the same write")

RO

When ext-CLUSTERRAS_ERR<n>STATUS.OF != '0' && Text("ERR<n>STATUS.OF is not being cleared to 0b0 in the same write")

RO

When ext-CLUSTERRAS_ERR<n>STATUS.CE != '00' && Text("ERR<n>STATUS.CE is not being cleared to 0b00 in the same write")

RO

When ext-CLUSTERRAS_ERR<n>STATUS.DE != '0' && Text("ERR<n>STATUS.DE is not being cleared to 0b0 in the same write")

RO

Otherwise

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

```
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 B-32: ext_clusterras_err0addr bit assignments

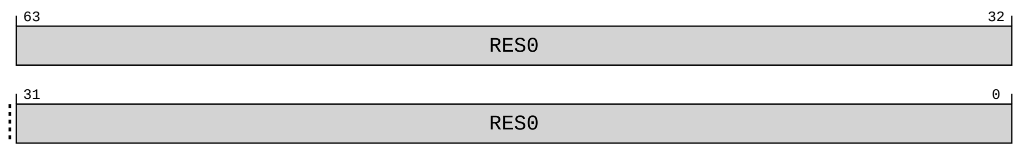


Table B-73: CLUSTERRAS_ERR0ADDR bit descriptions

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_ERR0MISC0, Error Record <n> Miscellaneous Register 0

IMPLEMENTATION DEFINED error syndrome register. The miscellaneous syndrome registers might contain:

- Information to locate where the error was detected.
- If the error was detected within a FRU, the identity of the FRU.
- A Corrected error counter or counters.
- Other state information not present in the corresponding status and address registers.

If the node that owns error record <n> implements a standard format Corrected error counter or counters (ERR<q>FR.CEC != 0b000), then it is IMPLEMENTATION DEFINED whether error record <n> can record countable errors, and:

- If error record <n> records countable errors, then ERR<n>MISC0 implements the standard format Corrected error counter or counters for error record <n>.

- If error record <n> does not record countable errors, then it is recommended that the fields in ERR<n>MISC0 defined for the standard format counter or counters are **RES0**. That is, the fields behave like counters that never count.

Configurations

ERR<n>FR describes the features implemented by the node that owns error record <n>. <q> is the index of the first error record owned by the same node as error record <n>. If the node owns a single record then q = n.

For IMPLEMENTATION DEFINED fields in ERR<n>MISC0, writing zero returns the error record to an initial quiescent state.

In particular, if any IMPLEMENTATION DEFINED syndrome fields might generate a Fault Handling or Error Recovery Interrupt request, writing zero is sufficient to deactivate the Interrupt request.

Fields that are read-only, nonzero, and ignore writes are compliant with this requirement.

Note

Arm recommends that any IMPLEMENTATION DEFINED syndrome field that can generate a Fault Handling, Error Recovery, Critical, or IMPLEMENTATION DEFINED, interrupt request is disabled at Cold reset and is enabled by software writing an IMPLEMENTATION DEFINED nonzero value to an IMPLEMENTATION DEFINED field in ERR<q>CTLR.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x20

Access type

Read

R

Write

W

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000 xxxx 0000 0000 0000 0000 00xx 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-33: ext_clusterras_err0misc0 bit assignments

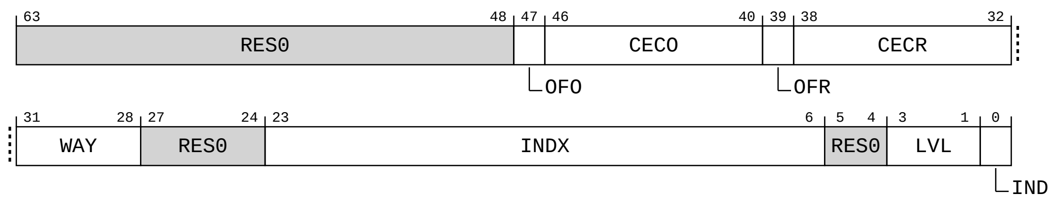


Table B-75: CLUSTERRAS_ERR0MISC0 bit descriptions

Bits	Name	Description	Reset
[63:48]	RES0	Reserved	RES0
[47]	OFO	Sticky overflow bit, other. Set to 1 when ERR<n>MISC0.CECO is incremented and wraps through zero. **0b0** Other counter has not overflowed. **0b1** Other counter has overflowed. A direct write that modifies this field might indirectly set ext-CLUSTERRAS_ERR<n>STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR<n>STATUS.OF that clears it to zero might indirectly set this field to an **UNKNOWN** value.	x
[46:40]	CECO	Corrected error count, other. Incremented for each countable error that is not accounted for by incrementing ERR<n>MISC0.CECR.	7{x}
[39]	OFR	Sticky overflow bit, repeat. Set to 1 when ERR<n>MISC0.CECR is incremented and wraps through zero. **0b0** Repeat counter has not overflowed. **0b1** Repeat counter has overflowed. A direct write that modifies this field might indirectly set ext-CLUSTERRAS_ERR<n>STATUS.OF to an **UNKNOWN** value and a direct write to ext-CLUSTERRAS_ERR<n>STATUS.OF that clears it to zero might indirectly set this field to an **UNKNOWN** value.	x

Bits	Name	Description	Reset
[38:32]	CECR	Corrected error count, repeat. Incremented for the first countable error, which also records other syndrome for the error, and subsequently for each countable error that matches the recorded other syndrome. Corrected errors are countable errors. It is **IMPLEMENTATION DEFINED** and might be UNPREDICTABLE whether Deferred and Uncorrected errors are countable errors. **Note:** For example, the other syndrome might include the set and way information for an error detected in a cache. This might be recorded in the **IMPLEMENTATION DEFINED** ERR<n>MISC<m> fields on a first Corrected error. ERR<n>MISC0.CECR is then incremented for each subsequent Corrected Error in the same set and way.	`7{x}`
[31:28]	WAY	This is the before description. This is the after description	`0b0000`
[27:24]	**RES0**	Reserved	**RES0**
[23:6]	INDX	This is the before description. This is the after description	`0b000000000000000000`
[5:4]	**RES0**	Reserved	**RES0**
[3:1]	LVL	This is the before description. This is the after description	`0b000`
[0]	IND	This is the before description. This is the after description	`0b0`

Access

Reads from ERR<n>MISC0 return an **IMPLEMENTATION DEFINED** value and writes have **IMPLEMENTATION DEFINED** behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<n>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Accessibility

Reads from ERR<n>MISC0 return an IMPLEMENTATION DEFINED value and writes have IMPLEMENTATION DEFINED behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<n>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Component	Offset	Instance	Range
CLUSTERRAS	0x20	ERR0MISC0	None

This interface is accessible as follows:

RW

B.1.3.6 CLUSTERRAS_ERR0MISC1, Error Record <n> Miscellaneous Register 1

IMPLEMENTATION DEFINED error syndrome register. The miscellaneous syndrome registers might contain:

- Information to locate where the error was detected.
- If the error was detected within a FRU, the identity of the FRU.
- A Corrected error counter or counters.
- Other state information not present in the corresponding status and address registers.

Configurations

ERR<q>FR describes the features implemented by the node that owns error record <n>. <q> is the index of the first error record owned by the same node as error record <n>. If the node owns a single record then q = n.

For IMPLEMENTATION DEFINED fields in ERR<n>MISC1, writing zero returns the error record to an initial quiescent state.

In particular, if any IMPLEMENTATION DEFINED syndrome fields might generate a Fault Handling or Error Recovery Interrupt request, writing zero is sufficient to deactivate the Interrupt request.

Fields that are read-only, nonzero, and ignore writes are compliant with this requirement.

Note

Arm recommends that any IMPLEMENTATION DEFINED syndrome field that can generate a Fault Handling, Error Recovery, Critical, or IMPLEMENTATION DEFINED, interrupt request is disabled at Cold reset and is enabled by software writing an IMPLEMENTATION DEFINED nonzero value to an IMPLEMENTATION DEFINED field in ERR<q>CTLR.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x28

Access type

Read

R

Write

W

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 B-34: ext_clusterras_err0misc1 bit assignments

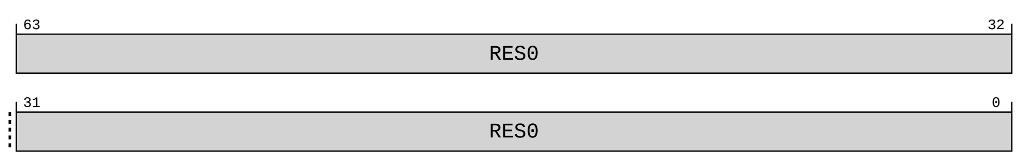


Table B-77: CLUSTERRAS_ERR0MISC1 bit descriptions

Bits	Name	Description	Reset
[63:0]	**RES0**	Reserved	**RES0**

Access

Reads from ERR<n>MISC1 return an **IMPLEMENTATION DEFINED** value and writes have **IMPLEMENTATION DEFINED** behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<q>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Accessibility

Reads from ERR<n>MISC1 return an IMPLEMENTATION DEFINED value and writes have IMPLEMENTATION DEFINED behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<q>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Component	Offset	Instance	Range
CLUSTERRAS	0x28	ERR0MISC1	None

This interface is accessible as follows:

RW

B.1.3.7 CLUSTERRAS_ERR0MISC2, Error Record <n> Miscellaneous Register 2

IMPLEMENTATION DEFINED error syndrome register. The miscellaneous syndrome registers might contain:

- Information to locate where the error was detected.
- If the error was detected within a FRU, the identity of the FRU.
- A Corrected error counter or counters.
- Other state information not present in the corresponding status and address registers.

Configurations

ERR<q>FR describes the features implemented by the node that owns error record <n>. <q> is the index of the first error record owned by the same node as error record <n>. If the node owns a single record then q = n.

For IMPLEMENTATION DEFINED fields in ERR<n>MISC2, writing zero returns the error record to an initial quiescent state.

In particular, if any IMPLEMENTATION DEFINED syndrome fields might generate a Fault Handling or Error Recovery Interrupt request, writing zero is sufficient to deactivate the Interrupt request.

Fields that are read-only, nonzero, and ignore writes are compliant with this requirement.

Arm recommends that if RAS System Architecture v1.1 is not implemented then ERR<n>MISC2 does not require zeroing to return the record to a quiescent state.

Note

Arm recommends that any IMPLEMENTATION DEFINED syndrome field that can generate a Fault Handling, Error Recovery, Critical, or IMPLEMENTATION DEFINED, interrupt request is disabled at Cold reset and is enabled by software writing an IMPLEMENTATION DEFINED nonzero value to an IMPLEMENTATION DEFINED field in ERR<q>CTLR.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x30

Access type

Read

R

Write

W

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 B-35: ext_clusterras_err0misc2 bit assignments

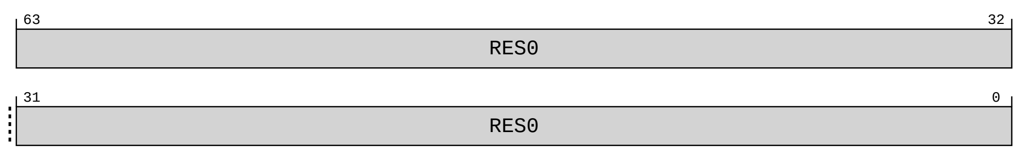


Table B-79: CLUSTERRAS_ERR0MISC2 bit descriptions

Bits	Name	Description	Reset
[63:0]	RES0	Reserved	RES0

Access

Reads from ERR<n>MISC2 return an IMPLEMENTATION DEFINED value and writes have IMPLEMENTATION DEFINED behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<q>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.

- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Accessibility

Reads from ERR<n>MISC2 return an IMPLEMENTATION DEFINED value and writes have IMPLEMENTATION DEFINED behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<q>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Component	Offset	Instance	Range
CLUSTERRAS	0x30	ERR0MISC2	None

This interface is accessible as follows:

RW

B.1.3.8 CLUSTERRAS_ERR0MISC3, Error Record <n> Miscellaneous Register 3

IMPLEMENTATION DEFINED error syndrome register. The miscellaneous syndrome registers might contain:

- Information to locate where the error was detected.
- If the error was detected within a FRU, the identity of the FRU.
- A Corrected error counter or counters.
- Other state information not present in the corresponding status and address registers.

If the node that owns error record n supports the RAS Timestamp Extension (ERR<q>FR.TS != 0b00), then ERR<n>MISC3 contains the timestamp value for error record n when the error was detected. Otherwise the contents of ERR<n>MISC3 are IMPLEMENTATION DEFINED.

Configurations

ERR<q>FR describes the features implemented by the node that owns error record <n>. <q> is the index of the first error record owned by the same node as error record <n>. If the node owns a single record then q = n.

For IMPLEMENTATION DEFINED fields in ERR<n>MISC3, writing zero returns the error record to an initial quiescent state.

In particular, if any IMPLEMENTATION DEFINED syndrome fields might generate a Fault Handling or Error Recovery Interrupt request, writing zero is sufficient to deactivate the Interrupt request.

Fields that are read-only, nonzero, and ignore writes are compliant with this requirement.

Arm recommends that if RAS System Architecture v1.1 is not implemented then ERR<n>MISC3 does not require zeroing to return the record to a quiescent state.

Note

Arm recommends that any IMPLEMENTATION DEFINED syndrome field that can generate a Fault Handling, Error Recovery, Critical, or IMPLEMENTATION DEFINED, interrupt request is disabled at Cold reset and is enabled by software writing an IMPLEMENTATION DEFINED nonzero value to an IMPLEMENTATION DEFINED field in ERR<q>CTLR.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x38

Access type

Read

R

Write

W

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 B-36: ext_clusterras_err0misc3 bit assignments

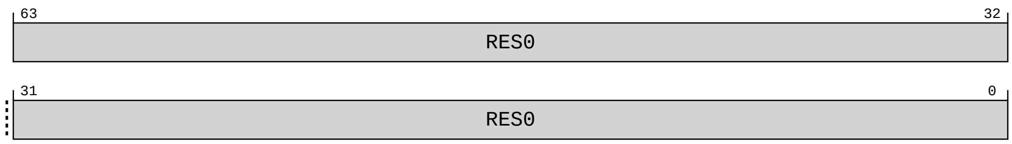


Table B-81: CLUSTERRAS_ERR0MISC3 bit descriptions

Bits	Name	Description	Reset
[63:0]	**RES0**	Reserved	**RES0**

Access

Reads from ERR<n>MISC3 return an **IMPLEMENTATION DEFINED** value and writes have **IMPLEMENTATION DEFINED** behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<q>PFGF.MV is 1, then some parts of this register are read/write when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Accessibility

Reads from ERR<n>MISC3 return an IMPLEMENTATION DEFINED value and writes have IMPLEMENTATION DEFINED behavior.

If the Common Fault Injection Mechanism is implemented by the node that owns this error record, and ERR<q>PFGF.MV is 1, then some parts of this register are read/write when ext-

CLUSTERRAS_ERR<n>STATUS.MV is 0. See ext-CLUSTERRAS_ERR<n>PFGF.MV for more information.

For other parts of this register, or if the Common Fault Injection Mechanism is not implemented, then Arm recommends that:

- Miscellaneous syndrome for multiple errors, such as a corrected error counter, is read/write.
- When ext-CLUSTERRAS_ERR<n>STATUS.MV is 1, the miscellaneous syndrome specific to the most recently recorded error ignores writes.

Note

These recommendations allow a counter to be reset in the presence of a persistent error, while preventing specific information, such as that identifying a FRU, from being lost if an error is detected while the previous error is being logged.

Component	Offset	Instance	Range
CLUSTERRAS	0x38	ERR0MISC3	None

This interface is accessible as follows:

RW

B.1.3.9 CLUSTERRAS_ERR0PFGF, Error Record <n> Pseudo-fault Generation Feature Register

Defines which common architecturally-defined fault generation features are implemented.

Configurations

ext-CLUSTERRAS_ERR<n>FR describes the features implemented by the node.

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 x1x1 011x xx11
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-37: ext_clusterras_err0pfgf bit assignments

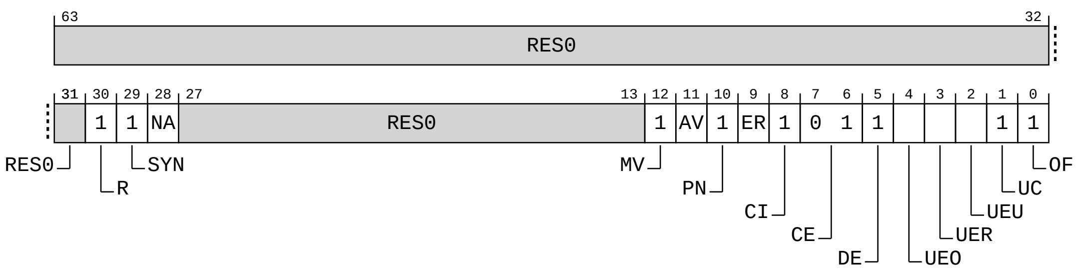


Table B-83: CLUSTERRAS_ERR0PFGF bit descriptions

Bits	Name	Description	Reset
[63:31]	**RES0**	Reserved	**RES0**
[30]	R	Restartable. Support for Error Generation Counter restart mode. **0b1** Error Generation Counter restart mode is implemented and is controlled by ext-CLUSTERRAS_ERR<n>PFGCTL.R. ext-CLUSTERRAS_ERR<n>PFGCTL.R is a read/write field.	`0b1`
[29]	SYN	Syndrome. Fault syndrome injection. **0b1** When an injected error is recorded, the node does not update the ext-CLUSTERRAS_ERR<n>STATUS.{IERR, SERR} fields. ext-CLUSTERRAS_ERR<n>STATUS.{IERR, SERR} are writable when ext-CLUSTERRAS_ERR<n>STATUS.V is 0. **Note:** If ERR<n>PFGF.SYN is 1 then software can write specific values into the ext-CLUSTERRAS_ERR<n>STATUS.{IERR, SERR} fields when setting up a fault injection event. The sets of values that can be written to these fields is **IMPLEMENTATION DEFINED**.	`0b1`
[28]	NA	No access required. Defines whether this component fakes detection of the error on an access to the component or spontaneously in the fault injection state. **0b0** The component fakes detection of the error on an access to the component. **0b1** The component fakes detection of the error spontaneously in the fault injection state.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[27:13]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[12]	MV	Miscellaneous syndrome. Defines whether software can control all or part of the syndrome recorded in the ERR<n>MISC<m> registers when an injected error is recorded. It is **IMPLEMENTATION DEFINED** which ERR<n>MISC<m> syndrome fields, if any, are updated by the node when an injected error is recorded. Some syndrome fields might always be updated by the node when an error, including an injected error, is recorded. For example, a corrected error counter might always be updated when any countable error, including a injected countable error, is recorded. **`0b1`** When an injected error is recorded, the node might update some, but not all ERR<n>MISC<m> syndrome fields: • If any syndrome is recorded by the node in the ERR<n>MISC<m> registers, then ext-CLUSTERRAS_ERR<n>STATUS.MV is set to 1. • Otherwise, ext-CLUSTERRAS_ERR<n>STATUS.MV is set to ext-CLUSTERRAS_ERR<n>PFGCTL.MV. ERR<n>MISC<m> syndrome fields that are not updated by the node are writable when ext-CLUSTERRAS_ERR<n>STATUS.MV is 0. If the node always sets ext-CLUSTERRAS_ERR<n>STATUS.MV to 1 when recording an injected error then ext-CLUSTERRAS_ERR<n>PFGCTL.MV is **RAO/WI**. Otherwise ext-CLUSTERRAS_ERR<n>PFGCTL.MV is a read/write field. If ERR<n>PFGF.MV is 1, software can write specific additional syndrome values into the ERR<n>MISC<m> registers when setting up a fault injection event. The permitted values that can be written to these registers are **IMPLEMENTATION DEFINED**.	`0b1`
[11]	AV	Address syndrome. Defines whether software can control the address recorded in ext-CLUSTERRAS_ERR<n>ADDR when an injected error is recorded. **`0b0`** When an injected error is recorded, the node might record an address in ext-CLUSTERRAS_ERR<n>ADDR. If an address is recorded in ext-CLUSTERRAS_ERR<n>ADDR, then ext-CLUSTERRAS_ERR<n>STATUS.AV is set to 1. Otherwise, ext-CLUSTERRAS_ERR<n>ADDR and ext-CLUSTERRAS_ERR<n>STATUS.AV are unchanged. If the node always records an address and sets ext-CLUSTERRAS_ERR<n>STATUS.AV to 1 when recording an injected error then ext-CLUSTERRAS_ERR<n>PFGCTL.AV might be RAO/WI. Otherwise ext-CLUSTERRAS_ERR<n>PFGCTL.AV is **RES0**. **`0b1`** When an injected error is recorded, the node does not update ext-CLUSTERRAS_ERR<n>ADDR and does one of: • Sets ext-CLUSTERRAS_ERR<n>STATUS.AV to ext-CLUSTERRAS_ERR<n>PFGCTL.AV. ext-CLUSTERRAS_ERR<n>PFGCTL.AV is a read/write field. • Sets ext-CLUSTERRAS_ERR<n>STATUS.AV to 1. ext-CLUSTERRAS_ERR<n>PFGCTL.AV is **RAO/WI**. ext-CLUSTERRAS_ERR<n>ADDR is writable when ext-CLUSTERRAS_ERR<n>STATUS.AV is 0. If ERR<n>PFGF.AV is 1 then software can write a specific address value into ext-CLUSTERRAS_ERR<n>ADDR when setting up a fault injection event.	The reset values can be the following: `0b0`, `0b1`, respective to the value.

Bits	Name	Description	Reset
[10]	PN	Poison flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR<n>STATUS.PN status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR<n>STATUS.PN is set to ext-CLUSTERRAS_ERR<n>PFGCTL.PN. ext-CLUSTERRAS_ERR<n>PFGCTL.PN is a read/write field. This behavior replaces the architecture-defined rules for setting the ext-CLUSTERRAS_ERR<n>STATUS.PN bit.	`0b1`
[9]	ER	Error Reported flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR<n>STATUS.ER status flag. **0b0** When an injected error is recorded, the node sets ext-CLUSTERRAS_ERR<n>STATUS.ER according to the architecture-defined rules for setting the ER field. ext-CLUSTERRAS_ERR<n>PFGCTL.ER is **RES0**. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR<n>STATUS.ER is set to ext-CLUSTERRAS_ERR<n>PFGCTL.ER. This behavior replaces the architecture-defined rules for setting the ER bit. ext-CLUSTERRAS_ERR<n>PFGCTL.ER is a read/write field.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[8]	CI	Critical Error flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR<n>STATUS.CI status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR<n>STATUS.CI is set to ext-CLUSTERRAS_ERR<n>PFGCTL.CI. ext-CLUSTERRAS_ERR<n>PFGCTL.CI is a read/write field. This behavior replaces the architecture-defined rules for setting the ext-CLUSTERRAS_ERR<n>STATUS.CI bit.	`0b1`
[7:6]	CE	Corrected Error generation. Describes the types of Corrected error that the fault generation feature of the node can generate. **0b01** The fault generation feature of the node allows generation of a non-specific Corrected error, that is, a Corrected error that is recorded by setting ext-CLUSTERRAS_ERR<n>STATUS.CE to 0b10. ext-CLUSTERRAS_ERR<n>PFGCTL.CE is a read/write field. The values 0b10 and 0b11 in ext-CLUSTERRAS_ERR<n>PFGCTL.CE are reserved. All other values are reserved. If ext-CLUSTERRAS_ERR<n>FR.FRX is 1 then ext-CLUSTERRAS_ERR<n>FR.CE indicates whether the node supports this type of error.	`0b01`
[5]	DE	Deferred Error generation. Describes whether the fault generation feature of the node can generate Deferred errors. **0b1** The fault generation feature of the node allows generation of Deferred errors. ext-CLUSTERRAS_ERR<n>PFGCTL.DE is a read/write field. If ext-CLUSTERRAS_ERR<n>FR.FRX is 1 then ext-CLUSTERRAS_ERR<n>FR.DE indicates whether the node supports this type of error.	`0b1`

Bits	Name	Description	Reset
[4]	UEO	Latent or Restartable Error generation. Describes whether the fault generation feature of the node can generate Latent or Restartable errors. **0b0** The fault generation feature of the node does not generate Latent or Restartable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UEO is **RES0**. **0b1** The fault generation feature of the node allows generation of Latent or Restartable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UEO is a read/write field. If ext-CLUSTERRAS_ERR<n>FR.FRX is 1 then ext-CLUSTERRAS_ERR<n>FR.UEO indicates whether the node supports this type of error.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[3]	UER	Signaled or Recoverable Error generation. Describes whether the fault generation feature of the node can generate Signaled or Recoverable errors. **0b0** The fault generation feature of the node does not generate Signaled or Recoverable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UER is **RES0**. **0b1** The fault generation feature of the node allows generation of Signaled or Recoverable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UER is a read/write field. If ext-CLUSTERRAS_ERR<n>FR.FRX is 1 then ext-CLUSTERRAS_ERR<n>FR.UER indicates whether the node supports this type of error.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[2]	UEU	Unrecoverable Error generation. Describes whether the fault generation feature of the node can generate Unrecoverable errors. **0b0** The fault generation feature of the node does not generate Unrecoverable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UEU is **RES0**. **0b1** The fault generation feature of the node allows generation of Unrecoverable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UEU is a read/write field. If ext-CLUSTERRAS_ERR<n>FR.FRX is 1 then ext-CLUSTERRAS_ERR<n>FR.UEU indicates whether the node supports this type of error.	The reset values can be the following: `0b0`, `0b1`, respective to the value.
[1]	UC	Uncontainable Error generation. Describes whether the fault generation feature of the node can generate Uncontainable errors. **0b1** The fault generation feature of the node allows generation of Uncontainable errors. ext-CLUSTERRAS_ERR<n>PFGCTL.UC is a read/write field. If ext-CLUSTERRAS_ERR<n>FR.FRX is 1 then ext-CLUSTERRAS_ERR<n>FR.UC indicates whether the node supports this type of error.	`0b1`
[0]	OF	Overflow flag. Describes how the fault generation feature of the node sets the ext-CLUSTERRAS_ERR<n>STATUS.OF status flag. **0b1** When an injected error is recorded, ext-CLUSTERRAS_ERR<n>STATUS.OF is set to ext-CLUSTERRAS_ERR<n>PFGCTL.OF. This behavior replaces the architecture-defined rules for setting the OF bit. ext-CLUSTERRAS_ERR<n>PFGCTL.OF is a read/write field.	`0b1`

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x800	ERR0PFGF	None

This interface is accessible as follows:

RO

B.1.3.10 CLUSTERRAS_ERR0PFGCTL, Error Record <n> Pseudo-fault Generation Control Register

Enables controlled fault generation.

Configurations

ext-CLUSTERRAS_ERR<n>PFGF describes the Common Fault Injection features implemented by the node.

ext-CLUSTERRAS_ERR<n>FR describes the features implemented by the node.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x808

Access type

RW

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

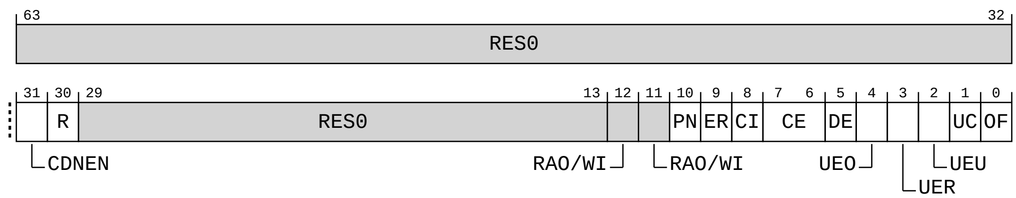


Table B-85: CLUSTERRAS_ERR0PFGCTL bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31]	CDNEN	Countdown Enable. Controls transfers of the value held in ext-CLUSTERRAS_ERR<n>PFGCDN to the Error Generation Counter and enables this counter. **`0b0`** The Error Generation Counter is disabled. **`0b1`** The Error Generation Counter is enabled. On a write of 1 to this field, the Error Generation Counter is set to ext-CLUSTERRAS_ERR<n>PFGCDN.CDN.	`0b0`
[30]	R	**When the node supports this control** Restart. Controls whether the Error Generation Counter restarts or stops counting on reaching zero. **`0b0`** On reaching zero, the Error Generation Counter will stop counting. **`0b1`** On reaching zero, the Error Generation Counter is set to ext-CLUSTERRAS_ERR<n>PFGCDN.CDN. **Otherwise** RES0	`x`
[29:13]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[12]	MV	**When the node supports this control && the node always sets ERR<n>STATUS.MV to 0b1 when an injected error is recorded** **RAO/WI** 8 **When the node supports this control** Miscellaneous syndrome. The value written to ext-CLUSTERRAS_ERR<n>STATUS.MV when an injected error is recorded. **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.MV is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.MV is set to 1 when an injected error is recorded. **When the node always sets ERR<n>STATUS.MV to 0b1 when an injected error is recorded && this field is RAO/WI** **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.MV is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.MV is set to 1 when an injected error is recorded. **Otherwise** RES0	x
[11]	AV	**When the node supports this control && the node always sets ERR<n>STATUS.AV to 0b1 when an injected error is recorded** **RAO/WI** 8 **When the node supports this control** Address syndrome. The value written to ext-CLUSTERRAS_ERR<n>STATUS.AV when an injected error is recorded. **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.AV is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.AV is set to 1 when an injected error is recorded. **When the node always sets ERR<n>STATUS.AV to 0b1 when an injected error is recorded && this field is RAO/WI** **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.AV is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.AV is set to 1 when an injected error is recorded. **Otherwise** RES0	x

Bits	Name	Description	Reset
[10]	PN	**When the node supports this control** Poison flag. The value written to ext-CLUSTERRAS_ERR<n>STATUS.PN when an injected error is recorded. **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.PN is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.PN is set to 1 when an injected error is recorded. **Otherwise** RES0	`x`
[9]	ER	**When the node supports this control** Error Reported flag. The value written to ext-CLUSTERRAS_ERR<n>STATUS.ER when an injected error is recorded. **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.ER is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.ER is set to 1 when an injected error is recorded. **Otherwise** RES0	`x`
[8]	CI	**When the node supports this control** Critical Error flag. The value written to ext-CLUSTERRAS_ERR<n>STATUS.CI when an injected error is recorded. **`0b0`** ext-CLUSTERRAS_ERR<n>STATUS.CI is set to 0 when an injected error is recorded. **`0b1`** ext-CLUSTERRAS_ERR<n>STATUS.CI is set to 1 when an injected error is recorded. **Otherwise** RES0	`x`

Bits	Name	Description	Reset
[7:6]	CE	**When the node supports this control** Corrected Error generation enable. Controls the type of injected Corrected error generated by the fault injection feature of the node. **`0b00`** An injected Corrected error will not be generated by the fault injection feature of the node. **`0b01`** An injected non-specific Corrected error is generated in the fault injection state. ext-CLUSTERRAS_ERR<n>STATUS.CE is set to 0b10 when the injected error is recorded. **`0b10`** An injected transient Corrected error is generated in the fault injection state. ext-CLUSTERRAS_ERR<n>STATUS.CE is set to 0b01 when the injected error is recorded. **`0b11`** An injected persistent Corrected error is generated in the fault injection state. ext-CLUSTERRAS_ERR<n>STATUS.CE is set to 0b11 when the injected error is recorded. The set of permitted values for this field is defined by ext-CLUSTERRAS_ERR<n>PFGF.CE. The node enters the fault injection state when the Error Generation Counter decrements to zero. It is IMPLEMENTATION DEFINED whether the injected error is generated when the error is generated on an access to the component in the fault injection state and the data is not consumed. **Otherwise** RES0	`xx`
[5]	DE	**When the node supports this control** Deferred Error generation enable. Controls whether an injected Deferred error is generated by the fault injection feature of the node. **`0b0`** An injected Deferred error will not be generated by the fault generation feature of the node. **`0b1`** An injected Deferred error is generated in the fault injection state. The node enters the fault injection state when the Error Generation Counter decrements to zero. It is IMPLEMENTATION DEFINED whether the injected error is generated when the error is generated on an access to the component in the fault injection state and the data is not consumed. **Otherwise** RES0	`x`

Bits	Name	Description	Reset
[4]	UEO	**When the node supports this control** Latent or Restartable Error generation enable. Controls whether an injected Latent or Restartable error is generated by the fault injection feature of the node. **`0b0`** An injected Latent or Restartable error will not be generated by the fault generation feature of the node. **`0b1`** An injected Latent or Restartable error is generated in the fault injection state. The node enters the fault injection state when the Error Generation Counter decrements to zero. It is IMPLEMENTATION DEFINED whether the injected error is generated when the error is generated on an access to the component in the fault injection state and the data is not consumed. **Otherwise** RES0	x
[3]	UER	**When the node supports this control** Signaled or Recoverable Error generation enable. Controls whether an injected Signaled or Recoverable error is generated by the fault injection feature of the node. **`0b0`** An injected Signaled or Recoverable error will not be generated by the fault generation feature of the node. **`0b1`** An injected Signaled or Recoverable error is generated in the fault injection state. The node enters the fault injection state when the Error Generation Counter decrements to zero. It is IMPLEMENTATION DEFINED whether the injected error is generated when the error is generated on an access to the component in the fault injection state and the data is not consumed. **Otherwise** RES0	x
[2]	UEU	**When the node supports this control** Unrecoverable Error generation enable. Controls whether an injected Unrecoverable error is generated by the fault injection feature of the node. **`0b0`** An injected Unrecoverable error will not be generated by the fault generation feature of the node. **`0b1`** An injected Unrecoverable error is generated in the fault injection state. The node enters the fault injection state when the Error Generation Counter decrements to zero. It is IMPLEMENTATION DEFINED whether the injected error is generated when the error is generated on an access to the component in the fault injection state and the data is not consumed. **Otherwise** RES0	x

Bits	Name	Description	Reset

[1]	UC	**When the node supports this control**
		Uncontainable Error generation enable. Controls whether an injected Uncontainable error is generated by the fault injection feature of the node.
		`0b0`
		An injected Uncontainable error will not be generated by the fault generation feature of the node.
		`0b1`
		An injected Uncontainable error is generated in the fault injection state.
		The node enters the fault injection state when the Error Generation Counter decrements to zero. It is IMPLEMENTATION DEFINED whether the injected error is generated when the error is generated on an access to the component in the fault injection state and the data is not consumed.
		Otherwise
		RES0
			x

[0]	OF	**When the node supports this control**
		Overflow flag. The value written to ext-CLUSTERRAS_ERR<n>STATUS.OF when an injected error is recorded.
		`0b0`
		ext-CLUSTERRAS_ERR<n>STATUS.OF is set to 0 when an injected error is recorded.
		`0b1`
		ext-CLUSTERRAS_ERR<n>STATUS.OF is set to 1 when an injected error is recorded.
		Otherwise
		RES0
			x

Accessibility

Component	Offset	Instance	Range
CLUSTERRAS	0x808	ERR0PFGCTL	None

This interface is accessible as follows:

RW

B.1.3.11 CLUSTERRAS_ERR0PFGCDN, Error Record <n> Pseudo-fault Generation Countdown Register

Generates one of the errors enabled in the corresponding ext-CLUSTERRAS_ERR<n>PFGCTL register.

Configurations

ext-CLUSTERRAS_ERR<n>FR describes the features implemented by the node.

Attributes

Width

64

Component

CLUSTERRAS

Register offset

0x810

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |
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-39: ext_clusterras_err0pfgcdn bit assignments

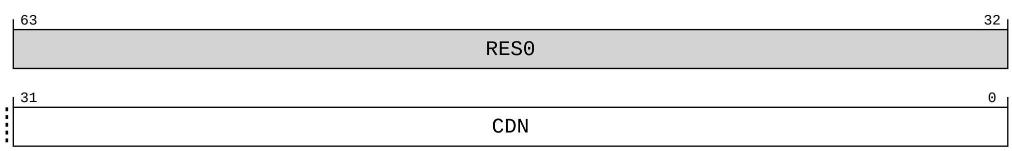


Table B-87: CLUSTERRAS_ERR0PFGCDN bit descriptions

Bits	Name	Description	Reset
[63:32]	**RES0**	Reserved	**RES0**
[31:0]	CDN	Countdown value. This field is copied to Error Generation Counter when either: • Software writes 1 to ext-CLUSTERRAS_ERR<n>PFGCTL.CDNEN. • The Error Generation Counter decrements to zero and ext-CLUSTERRAS_ERR<n>PFGCTL.R is 1. While ext-CLUSTERRAS_ERR<n>PFGCTL.CDNEN is 1 and the Error Generation Counter is nonzero, the counter decrements by 1 for each cycle at an **IMPLEMENTATION DEFINED** clock rate. When the counter reaches zero, one of the errors enabled in the ext-CLUSTERRAS_ERR<n>PFGCTL register is generated. **Note:** The current Error Generation Counter value is not visible to software.	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

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-40: ext_clusterras_errgsr bit assignments

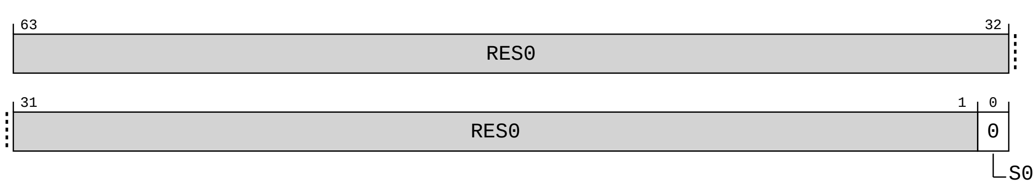


Table B-89: CLUSTERRAS_ERRGSR bit descriptions

Bits	Name	Description	Reset
[63:1]	**RES0**	Reserved	**RES0**
[0]	S0	The status for Error Record 0. A read-only copy of CLUSTERRAS_ERR0STATUS.V. **0b0** No error.	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 component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xE10

Access type

RO

Reset value

0100 1110 1110 0000 0010 0100 0011 1011

Bit descriptions

Figure B-41: ext_clusterras_erriidr bit assignments

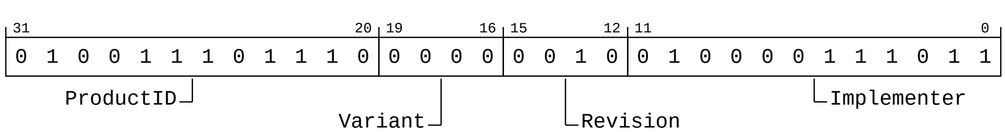


Table B-91: CLUSTERRAS_ERRIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Part number, bits [11:0]. The part number is selected by the designer of the component. **`0b010011101110`** DSU-C1 Cluster RAS. If ext-CLUSTERRAS_ERRPIDR0 and ext-CLUSTERRAS_ERRPIDR1 are implemented, ext-CLUSTERRAS_ERRPIDR0.PART_0 matches bits [7:0] of ext-CLUSTERRAS_ERRIIDR.ProductID and ext-CLUSTERRAS_ERRPIDR1.PART_1 matches bits [11:8] of ext-CLUSTERRAS_ERRIIDR.ProductID.	`0x4EE`
[19:16]	Variant	Component major revision. This field distinguishes product variants or major revisions of the product. **`0b0000`** Product Variant 0. If ext-CLUSTERRAS_ERRPIDR2 is implemented, ext-CLUSTERRAS_ERRPIDR2.REVISION matches ext-CLUSTERRAS_ERRIIDR.Variant.	`0b0000`
[15:12]	Revision	Component minor revision. This field distinguishes minor revisions of the product. **`0b0010`** Product major revision 2. If ext-CLUSTERRAS_ERRPIDR3 is implemented, ext-CLUSTERRAS_ERRPIDR3.REVAND matches ext-CLUSTERRAS_ERRIIDR.Revision.	`0b0010`
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the RAS component. For an Arm implementation, this field has the value `0x43B`. **`0b010000111011`** Arm implementation. Bits [11:8] contain the JEP106 continuation code of the implementer, and bits [6:0] contain the JEP106 identity code of the implementer. If ext-CLUSTERRAS_ERRPIDR4 is implemented, ext-CLUSTERRAS_ERRPIDR2 is implemented, and ext-CLUSTERRAS_ERRPIDR1 is implemented, ext-CLUSTERRAS_ERRPIDR4.DES_2 matches bits [11:8] of ext-CLUSTERRAS_ERRIIDR.Implementer, ext-CLUSTERRAS_ERRPIDR2.DES_1 matches bits [6:4] of ext-CLUSTERRAS_ERRIIDR.Implementer, and ext-CLUSTERRAS_ERRPIDR1.DES_0 matches bits [3:0] of ext-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

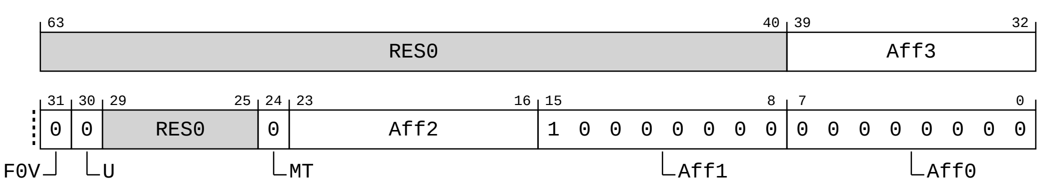


Table B-93: CLUSTERRAS_ERRDEVAFF bit descriptions

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}

Bits	Name	Description	Reset
[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**
[24]	MT	Multithreaded. The AArch64-MPIDR_EL1.MT bit viewed from the highest Exception level of the associated PE. **0b0** Performance of PEs at the lowest affinity level is largely independent. If ERRDEVAFF.Aff0 is not valid, this bit is not valid and reads as **UNKNOWN**.	0b0
[23:16]	Aff2	Affinity level 2. This field is the AArch64-MPIDR_EL1.Aff2 field viewed from the highest Exception level of the associated PE or PEs.	8{x}
[15:8]	Aff1	Affinity level 1. **0b10000000** ERRDEVAFF.Aff2 is valid, and the PE affinity level is 2.	0x80
[7:0]	Aff0	Affinity level 0. **0b00000000** ERRDEVAFF.Aff1 is valid, and the PE affinity level is 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

ERRDEVARCH is implemented only as part of a memory-mapped group of error records.

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

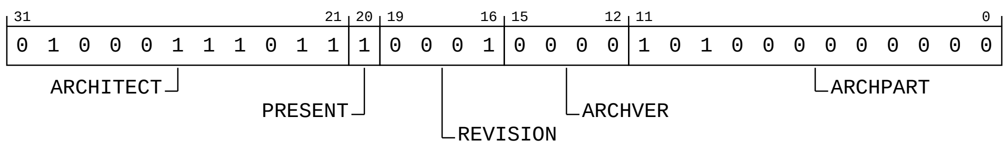


Table B-95: CLUSTERRAS_ERRDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. Defines the architect of the component. Bits [31:28] are the JEP106 continuation code (JEP106 bank ID, minus 1) and bits [27:21] are the JEP106 ID code. **`0b01000111011`** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	`0b01000111011`
[20]	PRESENT	DEVARCH present. Defines that ERRDEVARCH register is present. Defined values are: **`0b1`** Device Architecture information present. This field reads as 1.	`0b1`
[19:16]	REVISION	Revision. Defines the architecture revision of the component. Defined values are: **`0b0001`** RAS System Architecture, error record group v1.1. As 0b0000 and also: • Simplifies ext-CLUSTERRAS_ERR<n>STATUS. • Adds support for additional ERR<n>MISC<m> registers. • Adds support for the optional RAS Timestamp Extension. • Adds support for the optional Common Fault Injection Model Extension. All other values are reserved.	`0b0001`

Bits	Name	Description	Reset
[15:12]	ARCHVER	Architecture Version. Defines the architecture version of the component. Defined values are: **0b0000** RAS System Architecture, error record group v1. All other values are reserved. ERRDEVARCH.ARCHVER and ERRDEVARCH.ARCHPART are also defined as a single field, ERRDEVARCH.ARCHID, so that ERRDEVARCH.ARCHVER is ERRDEVARCH.ARCHID[15:12].	0b0000
[11:0]	ARCHPART	Architecture Part. Defines the architecture of the component. Defined values are: **0b101000000000** RAS System Architecture, error record group. ERRDEVARCH.ARCHVER and ERRDEVARCH.ARCHPART are also defined as a single field, ERRDEVARCH.ARCHID, so that ERRDEVARCH.ARCHPART is ERRDEVARCH.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

ERRDEVID is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFC8

Access type

RO

Reset value

```
xxxx xxxx xx0x 0000 0000 0000 0000 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-44: ext_clusterras_errdevid bit assignments

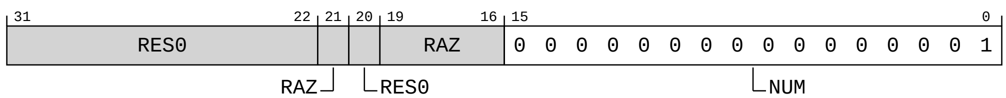


Table B-97: CLUSTERRAS_ERRDEVID bit descriptions

Bits	Name	Description	Reset
[31:22]	**RES0**	Reserved	**RES0**
[21]	**RAZ**	Reserved	**RAZ**
[20]	**RES0**	Reserved	**RES0**
[19:16]	**RAZ**	Reserved	**RAZ**
[15:0]	NUM	Highest numbered index of the error records in this group, plus one. Each implemented record is owned by a node. A node might own multiple records. **0b0000000000000001** One record implemented in this group. This manual describes a group of error records accessed via a standard 4KB memory-mapped peripheral. For a 4KB peripheral, up to 24 error records can be accessed if the Common Fault Injection Model is implemented, and up to 56 otherwise.	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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRPIDR4 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFD0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-45: ext_clusterras_errpidr4 bit assignments

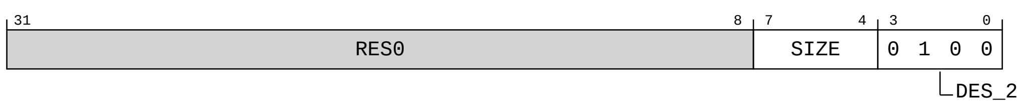


Table B-99: CLUSTERRAS_ERRPIDR4 bit descriptions

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. A value of `0b0000` means one of the following is true: • The component uses a single 4KB block. • The component uses an IMPLEMENTATION DEFINED number of 4KB blocks. Any other value means the component occupies $2^{\text{ERRPIDR4.SIZE}}$ 4KB blocks. Using this field to indicate the size of the component is deprecated. This field might not correctly indicate the size of the component. Arm recommends that software determine the size of the component from the Unique Component Identifier fields, and other IMPLEMENTATION DEFINED registers in the component.	xxxx

Bits	Name	Description	Reset
[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. To obtain a number, or to see the assignment of these codes, contact JEDEC http://www.jedec.org. **`0b0100`** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B. **Note:** For a component designed by Arm Limited, the JEP106 bank is 5, meaning this field has the value `0x4`.	`0b0100`

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

```
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-46: ext_clusterras_errpidr5 bit assignments

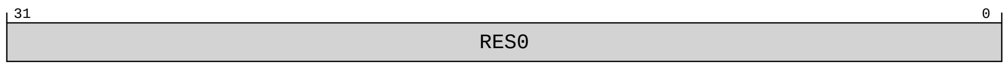


Table B-101: CLUSTERRAS_ERRPIDR5 bit descriptions

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

```
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-47: ext_clusterras_errpidr6 bit assignments

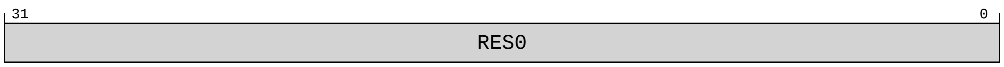


Table B-103: CLUSTERRAS_ERRPIDR6 bit descriptions

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

```
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-48: ext_clusterras_errpidr7 bit assignments

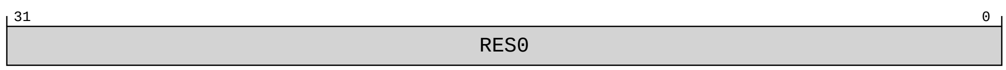


Table B-105: CLUSTERRAS_ERRPIDR7 bit descriptions

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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRPIDR0 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFE0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1110 1110
|    |    |    |    |    |    |    |   |
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

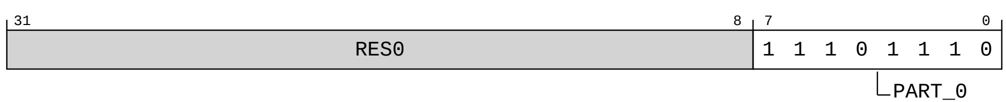


Table B-107: CLUSTERRAS_ERRPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number, bits [7:0]. The part number is selected by the designer of the component. The designer chooses whether to use a 12-bit or a 16-bit part number: • If a 12-bit part number is used, then it is stored in ext-CLUSTERRAS_ERRPIDR1.PART_1 and ERRPIDR0.PART_0. There are 8 bits, ext-CLUSTERRAS_ERRPIDR2.REVISION and ext-CLUSTERRAS_ERRPIDR3.REVAND, available to define the revision of the component. • If a 16-bit part number is used, then it is stored in ext-CLUSTERRAS_ERRPIDR2.PART_2, ext-CLUSTERRAS_ERRPIDR1.PART_1 and ERRPIDR0.PART_0. There are 4 bits, ext-CLUSTERRAS_ERRPIDR3.REVISION, available to define the revision of the component. **0b11101110** DSU-C1 DSU Cluster RAS. Bits [7:0] of part number 0x4EE.	0xEE

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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRPIDR1 is implemented only as part of a memory-mapped group of error records.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-50: ext_clusterras_errpidr1 bit assignments

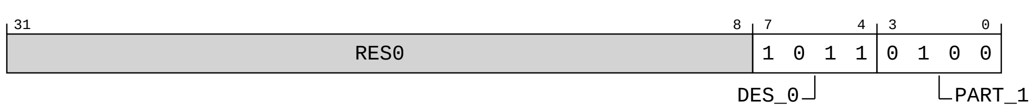


Table B-109: CLUSTERRAS_ERRPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	DES_0	Designer, JEP106 identification code, bits [3:0]. ERRPIDR1.DES_0 and ext-CLUSTERRAS_ERRPIDR2.DES_1 together form the JEDEC-assigned JEP106 identification code for the designer of the component. The parity bit in the JEP106 identification code is not included. The code identifies the designer of the component, which might not be not the same as the implementer of the device containing the component. To obtain a number, or to see the assignment of these codes, contact JEDEC http://www.jedec.org. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B. **Note:** For a component designed by Arm Limited, the JEP106 identification code is `0x3B`.	`0b1011`
[3:0]	PART_1	Part number, bits [11:8]. The part number is selected by the designer of the component. The designer chooses whether to use a 12-bit or a 16-bit part number: • If a 12-bit part number is used, then it is stored in ERRPIDR1.PART_1 and ext-CLUSTERRAS_ERRPIDR0.PART_0. There are 8 bits, ext-CLUSTERRAS_ERRPIDR2.REVISION and ext-CLUSTERRAS_ERRPIDR3.REVAND, available to define the revision of the component. • If a 16-bit part number is used, then it is stored in ext-CLUSTERRAS_ERRPIDR2.PART_2, ERRPIDR1.PART_1 and ext-CLUSTERRAS_ERRPIDR0.PART_0. There are 4 bits, ext-CLUSTERRAS_ERRPIDR3.REVISION, available to define the revision of the component. **0b0100** DSU-C1 DSU Cluster RAS. Bits [11:8] of part number 0x4EE.	`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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRPIDR2 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFE8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 1011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-51: ext_clusterras_errpidr2 bit assignments

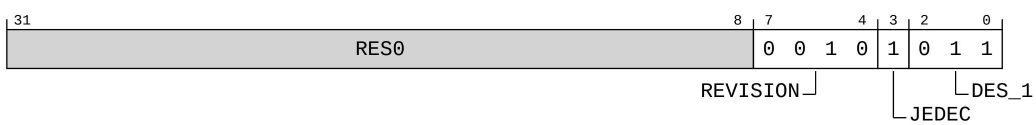


Table B-111: CLUSTERRAS_ERRPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. ERRPIDR2.REVISION and ext-CLUSTERRAS_ERRPIDR3.REVAND together form the revision number of the component, with ERRPIDR2.REVISION being the most significant part and ext-CLUSTERRAS_ERRPIDR3.REVAND the least significant part. When a component is changed, ERRPIDR2.REVISION or ext-CLUSTERRAS_ERRPIDR3.REVAND are increased to ensure that software can differentiate the different revisions of the component. ext-CLUSTERRAS_ERRPIDR3.REVAND should be set to `0b0000` when ERRPIDR2.REVISION is increased. **`0b0010`** Component major revision 2.	`0b0010`
[3]	JEDEC	JEDEC-assigned JEP106 implementer code is used. **`0b1`**	`0b1`

Bits	Name	Description	Reset
[2:0]	DES_1	Designer, JEP106 identification code, bits [6:4]. ext-CLUSTERRAS_ERRPIDR1.DES_0 and ERRPIDR2.DES_1 together form the JEDEC-assigned JEP106 identification code for the designer of the component. The parity bit in the JEP106 identification code is not included. The code identifies the designer of the component, which might not be not the same as the implementer of the device containing the component. To obtain a number, or to see the assignment of these codes, contact JEDEC http://www.jedec.org. **`0b011`** Arm Limited. Bits [7:4] of JEP106 identification code 0x3B. **Note:** 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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRPIDR3 is implemented only as part of a memory-mapped group of error records.

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

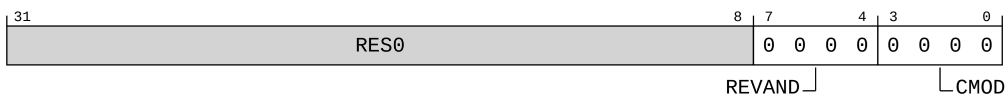


Table B-113: CLUSTERRAS_ERRPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Component minor revision. ext-CLUSTERRAS_ERRPIDR2.REVISION and ERRPIDR3.REVAND together form the revision number of the component, with ext-CLUSTERRAS_ERRPIDR2.REVISION being the most significant part and ERRPIDR3.REVAND the least significant part. When a component is changed, ext-CLUSTERRAS_ERRPIDR2.REVISION or ERRPIDR3.REVAND are increased to ensure that software can differentiate the different revisions of the component. ERRPIDR3.REVAND should be set to `0b0000` when ext-CLUSTERRAS_ERRPIDR2.REVISION is increased. **`0b0000`** Component minor revision 0.	`0b0000`
[3:0]	CMOD	Customer Modified. Indicates the component has been modified. A value of `0b0000` means the component is not modified from the original design. Any other value means the component has been modified in an **IMPLEMENTATION DEFINED** way. **`0b0000`** The component is not modified from the original design. For any two components with the same Unique Component Identifier: • If ERRPIDR3.CMOD is zero in both components, then the components are identical. • If ERRPIDR3.CMOD has the same nonzero value in both components, then this does not necessarily mean that they have the same modifications. • If ERRPIDR3.CMOD is nonzero in either component, the two components might not be identical despite having the same Unique Component Identifier.	`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 about the component.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRCIDR0 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

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-53: ext_clusterras_errcidr0 bit assignments

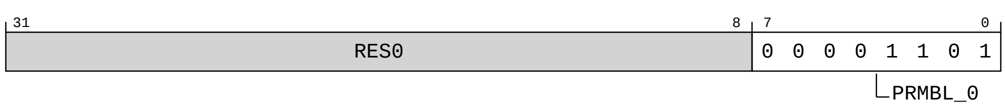


Table B-115: CLUSTERRAS_ERRCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:0]	PRMBL_0	Component identification preamble, segment 0. **0b00001101**	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 about the component.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRCIDR1 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFF4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-54: ext_clusterras_errcidr1 bit assignments

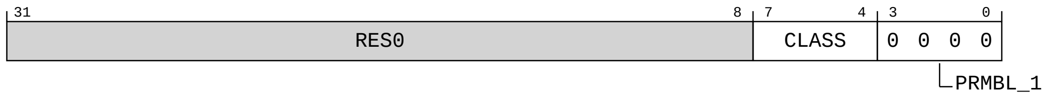


Table B-117: CLUSTERRAS_ERRCIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	Component class. `0b1111` Generic peripheral with **IMPLEMENTATION DEFINED** register layout. Other values are defined by the CoreSight Architecture. This field reads as `0xF`.	`xxxx`
[3:0]	PRMBL_1	Component identification preamble, segment 1. `0b0000`	`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 about the component.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRCIDR2 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

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-55: ext_clusterras_errcidr2 bit assignments

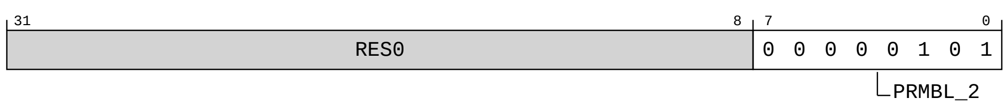


Table B-119: CLUSTERRAS_ERRCIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	Component identification preamble, segment 2. `0b00000101`	`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 about the component.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

ERRCIDR3 is implemented only as part of a memory-mapped group of error records.

Attributes

Width

32

Component

CLUSTERRAS

Register offset

0xFFC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-56: ext_clusterras_errcidr3 bit assignments

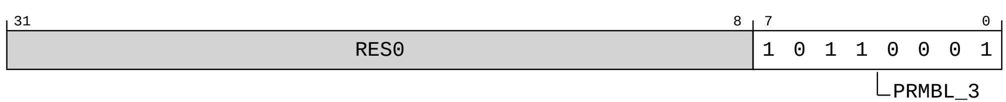


Table B-121: CLUSTERRAS_ERRCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Component identification preamble, segment 3. **`0b10110001`**	`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 summary table provides an overview of **IMPLEMENTATION DEFINED** memory-mapped ppu registers in the core. For more information about a register, click 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 or in the Arm ARM.

Table B-123: Cluster Power Policy Unit registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	See individual bit resets.	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	See individual bit resets.	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	See individual bit resets.	32-bit	Power Status Register	Yes
0x010	PPU_DISR	See individual bit resets.	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	See individual bit resets.	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	See individual bit resets.	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	See individual bit resets.	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	See individual bit resets.	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	See individual bit resets.	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	See individual bit resets.	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	See individual bit resets.	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	See individual bit resets.	32-bit	Additional Interrupt Status Register	Yes
0x040	PPU_IESR	See individual bit resets.	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register	Yes
0x054	PPU_FULRR	See individual bit resets.	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	See individual bit resets.	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	See individual bit resets.	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	See individual bit resets.	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	See individual bit resets.	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0	Yes
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1	Yes

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

B.1.4.1 PPU_PWPR, Power Policy Register

This register enables software to program both power and operating mode policy. It also contains related settings including the enable for dynamic transitions and the lock enable.

This register does not reflect the current power mode value. The current power mode of the domain is reflected in the Power Status Register (ext-PPU_PWSR).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x000

Access type

RW

Reset value

```
xxxx xxx0 xxxx 0000 xxx0 xxx0 xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-57: ext_ppu_pwpr bit assignments

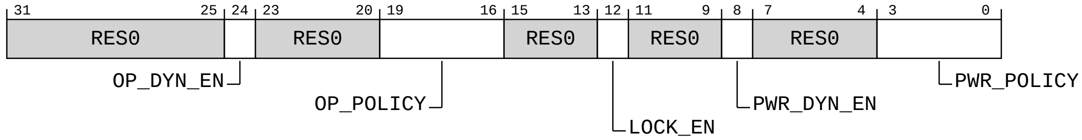


Table B-124: PPU_PWPR bit descriptions

Bits	Name	Description	Reset
[31:25]	**RES0**	Reserved	**RES0**
[24]	OP_DYN_EN	Operating mode dynamic transition enable. **`0b0`** Dynamic transitions disabled for operating modes. **`0b1`** Dynamic transitions enabled for operating modes, allowing transitions to be initiated by changes on operating mode DEVACTIVE inputs.	`0b0`
[23:20]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[19:16]	OP_POLICY	Operating mode policy. When static operating mode transitions are enabled, OP_DYN_EN is set to `0b0`, then this is the target operating mode for the PPU. When dynamic operating mode transitions are enabled, OP_DYN_EN is set to `0b1`, then this is the minimum operating mode for the PPU. All other values are reserved. **`0b0000`** OPMODE_00: ONE_SLICE_SF_ONLY_ON: One L3 Cache slice is operational, the Cache RAM is powered down. **`0b0001`** OPMODE_01: ONE_SLICE_HALF_RAM_ON: One L3 Cache slice is operational, half of the Cache RAMs are powered on. **`0b0011`** OPMODE_03: ONE_SLICE_FULL_RAM_ON: One L3 Cache slice is operational, all of the Cache RAMs are powered on. **`0b0100`** OPMODE_04: ALL_SLICE_SF_ONLY_ON: All L3 Cache slices are operational, the Cache RAMs in each slice are powered down. **`0b0101`** OPMODE_05: ALL_SLICE_HALF_RAM_ON: All L3 Cache slices are operational, half of the Cache RAMs are powered on. **`0b0111`** OPMODE_07: ALL_SLICE_FULL_RAM_ON: All L3 Cache slices are operational, all of the Cache RAMs are powered on. **`0b1000`** OPMODE_08: HALF_SLICE_SF_ONLY_ON: Half L3 Cache slices are operational, the Cache RAMs in each slice are powered down. **`0b1001`** OPMODE_09: HALF_SLICE_HALF_RAM_ON: Half L3 Cache slices are operational, half of the Cache RAMs are powered on. **`0b1011`** OPMODE_0B: HALF_SLICE_FULL_RAM_ON: Half L3 Cache slices are operational, all of the Cache RAMs are powered on.	`0b0000`
[15:13]	**RES0**	Reserved	**RES0**
[12]	LOCK_EN	Lock enable bit for OFF, OFF_EMU, MEM_RET and MEM_RET_EMU power modes. **`0b0`** Lock feature disabled. **`0b1`** Lock feature enabled.	`0b0`
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8]	PWR_DYN_EN	Power mode dynamic transition enable. **0b0** Dynamic transitions disabled for power modes. **0b1** Dynamic transitions enabled for power modes, allowing transitions to be initiated by changes on power mode DEVACTIVE inputs.	0b0
[7:4]	**RES0**	Reserved	**RES0**
[3:0]	PWR_POLICY	Power mode policy. When static power mode transitions are enabled, PWR_DYN_EN is set to `0b0`, this is the target power mode for the PPU. When dynamic power mode transitions are enabled, PWR_DYN_EN is set to `0b1`, this is the minimum power mode for the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0010** MEM_RET. Memory Retention. Logic off with RAM retained. **0b0011** MEM_RET_EMU. Emulated Memory Retention. Logic on with RAM on. This mode is used to emulate the functional condition of MEM_RET without removing power. **0b0101** FULL_RET. Full Retention. Slice logic off with RAM contents retained. **0b0111** FUNC_RET. Functional Retention. Logic on with L3 Cache and Snoop Filter retained. **0b1000** ON. Logic on with RAM on, cluster is functional. **0b1001** WARM_RST. Warm Reset. Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.4.2 PPU_PMER, Power Mode Emulation Enable Register

This register allows software to enable entry into emulated modes.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x004

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-58: ext_ppu_pmer bit assignments

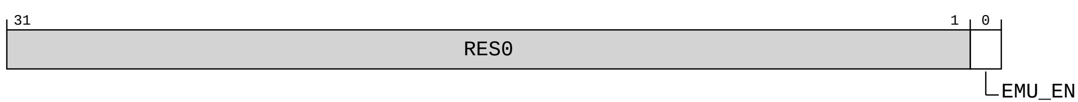


Table B-125: PPU_PMER bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0
[0]	EMU_EN	Power mode emulation enable. **0b0** Power mode emulation disabled. **0b1** Power mode emulation enabled. Transitions to OFF and MEM_RET instead transition to OFF_EMU and MEM_RET_EMU.	0b0

B.1.4.3 PPU_PWSR, Power Status Register

This read-only register contains status information for the power mode, operating mode, dynamic transitions, and lock feature.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x008

Access type

RO

Reset value

```
xxxx xxx0 xxxx 0000 xxx0 xxx0 xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-59: ext_ppu_pwsr bit assignments

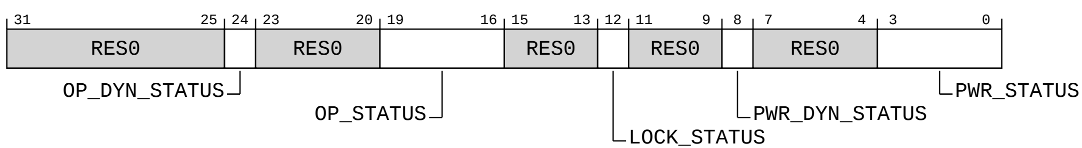


Table B-126: PPU_PWSR bit descriptions

Bits	Name	Description	Reset
[31:25]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[24]	OP_DYN_STATUS	Operating mode dynamic transition status. There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when ext-PPU_PWPR.OP_DYN_EN is programmed. **0b0** Dynamic transitions disabled for operating modes. **0b1** Dynamic transitions enabled for operating modes.	0b0
[23:20]	**RES0**	Reserved	**RES0**
[19:16]	OP_STATUS	Operating mode status. These bits reflect the current operating mode of the PPU. In the OFF, OFF_EMU, DBG_RECOV, and WARM_RST power modes, this field reflects the current programmed OP_POLICY even though the operating mode DEVPSTATE output bits are set to zero. All other values are reserved. **0b0000** OPMODE_00: ONE_SLICE_SF_ONLY_ON: One L3 Cache slice is operational, only the snoop filter RAM instances are active in the slice **0b0001** OPMODE_01: ONE_SLICE_HALF_RAM_ON: One L3 Cache slice is operational, half of the Cache RAMs are powered on. **0b0011** OPMODE_03: ONE_SLICE_FULL_RAM_ON: One L3 Cache slice is operational, all of the Cache RAMs are powered on. **0b0100** OPMODE_04: ALL_SLICE_SF_ONLY_ON: All L3 Cache slices are operational, only the snoop filter RAM instances are active in each slice. **0b0101** OPMODE_05: ALL_SLICE_HALF_RAM_ON: All L3 Cache slices are operational, half of the Cache RAMs are powered on. **0b0111** OPMODE_07: ALL_SLICE_FULL_RAM_ON: All L3 Cache slices are operational, all of the Cache RAMs are powered on. **0b1000** OPMODE_08: HALF_SLICE_SF_ONLY_ON: Half L3 Cache slices are operational, the Cache RAMs in each slice are powered down. **0b1001** OPMODE_09: HALF_SLICE_HALF_RAM_ON: Half L3 Cache slices are operational, half of the Cache RAMs are powered on. **0b1011** OPMODE_0B: HALF_SLICE_FULL_RAM_ON: Half L3 Cache slices are operational, all of the Cache RAMs are powered on.	0b0000
[15:13]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[12]	LOCK_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**
[8]	PWR_DYN_STATUS	Power mode dynamic transition status. There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when ext-PPU_PWPR.DYN_EN is programmed. **0b0** Dynamic transitions disabled for power modes. **0b1** Dynamic transitions enabled for power modes.	0b0
[7:4]	**RES0**	Reserved	**RES0**
[3:0]	PWR_STATUS	Power mode status. These bits reflect the current power mode of the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0010** MEM_RET. Memory Retention. Logic off with RAM retained. **0b0011** MEM_RET_EMU. Emulated Memory Retention. Logic on with RAM on. This mode is used to emulate the functional condition of MEM_RET without removing power. **0b0101** FULL_RET. Full Retention. Slice logic off with RAM contents retained. **0b0111** FUNC_RET. Functional Retention. Logic on with L3 Cache and Snoop Filter retained. **0b1000** ON. Logic on with RAM on, cluster is functional. **0b1001** WARM_RST. Warm Reset. Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.4.4 PPU_DISR, Device Interface Input Current Status Register

This read-only register contains status reflecting the values of the device interface inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x010

Access type

RO

Reset value

```
xxxx 0000 xxxx xxxx xxxx x000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-60: ext_ppu_disr bit assignments

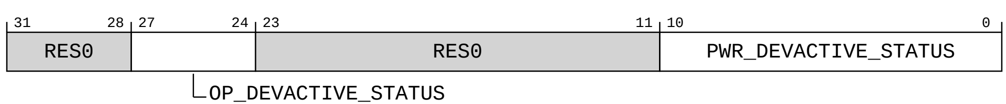


Table B-127: PPU_DISR bit descriptions

Bits	Name	Description	Reset
[31:28]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[27:24]	OP_DEVACTIVE_STATUS	Status of the operating mode DEVPACTIVE inputs. All other values are reserved. **0b0000** Request for OPMODE_00, ONE_SLICE_SF_ONLY_ON. **0b0001** Request for OPMODE_01, ONE_SLICE_HALF_RAM_ON. **0b0011** Request for OPMODE_03, ONE_SLICE_FULL_RAM_ON. **0b0100** Request for OPMODE_04, ALL_SLICE_SF_ONLY_ON. **0b0101** Request for OPMODE_05, ALL_SLICE_HALF_RAM_ON. **0b0111** Request for OPMODE_07, ALL_SLICE_FULL_RAM_ON. **0b1000** Request for OPMODE_08, HALF_SLICE_SF_ONLY_ON. **0b1001** Request for OPMODE_09, HALF_SLICE_HALF_RAM_ON. **0b1011** Request for OPMODE_0B, HALF_SLICE_FULL_RAM_ON.	0b0000
[23:11]	**RES0**	Reserved	**RES0**
[10:0]	PWR_DEVACTIVE_STATUS	Status of the power mode DEVPACTIVE inputs. **0b00000000000** Request for OFF. **0000000001x** Request for OFF_EMU. **000000001xx** Request for MEM_RET. **00000001xxx** Request for MEM_RET_EMU. **000001xxxxx** Request for FULL_RET. **0001xxxxxxx** Request for FUNC_RET. **001xxxxxxxx** Request for ON. **01xxxxxxxxx** Request for WARM_RST. **1xxxxxxxxxx** Request for DBG_RECOV.	0b00000000000

B.1.4.5 PPU_MISR, Miscellaneous Input Current Status Register

This read-only register contains status reflecting the values of miscellaneous inputs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x014

Access type

RO

Reset value

```
xxxx xxxx xxxx xxx0 xxxx xxx0 xxxx xxx0
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-61: ext_ppu_misr bit assignments

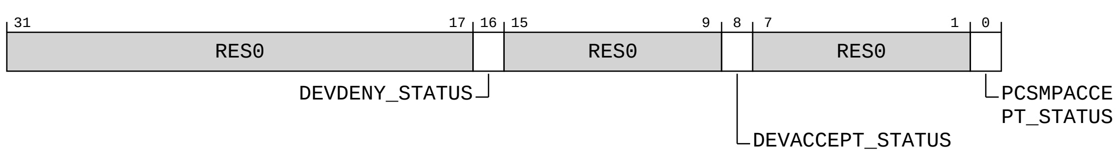


Table B-128: PPU_MISR bit descriptions

Bits	Name	Description	Reset
[31:17]	RES0	Reserved	RES0
[16]	DEVDENY_STATUS	Status of the device interface DEVPDENY inputs. **0b0** DEVPDENY deasserted. **0b1** DEVPDENY asserted.	0b0

Bits	Name	Description	Reset
[15:9]	**RES0**	Reserved	**RES0**
[8]	DEVACCEPT_STATUS	Status of the device interface DEVPACCEPT inputs. **0b0** DEVPACCEPT deasserted. **0b1** DEVPACCEPT asserted.	0b0
[7:1]	**RES0**	Reserved	**RES0**
[0]	PCSMPACCEPT_STATUS	Status of the PCSMPACCEPT inputs. **0b0** PCSMPACCEPT deasserted. **0b1** PCSMPACCEPT asserted.	0b0

B.1.4.6 PPU_STSR, Stored Status Register

This register is reserved for P-Channel PPUs.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x018

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-62: ext_ppu_stsr bit assignments

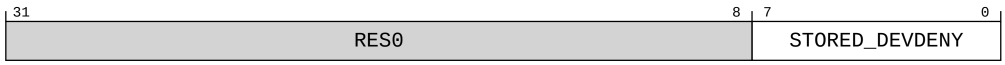


Table B-129: PPU_STSR bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	STORED_DEVDENY	Status of the DEVDENY signals from the last device interface Q-Channel transition. This field is reserved. **0b00000000** Reserved for P-Channel PPUs.	8{x}

B.1.4.7 PPU_UNLK, Unlock Register

This register allows software to unlock the PPU from a locked power mode.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x01C

Access type

UNKNOWNW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-63: ext_ppu_unlk bit assignments

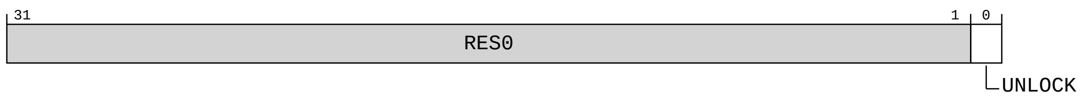


Table B-130: PPU_UNLK bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0
[0]	UNLOCK	When `0b1` is written to this bit the PPU is unlocked from a locked power mode. A read always returns `0b0`.	`x`

B.1.4.8 PPU_PWCR, Power Configuration Register

This register controls enabling and disabling of hardware control inputs to the PPU.

Note

Before software programs the DEVREQEN bits it must configure the PPU for static transitions and ensure the requested power mode has been reached, this means that no further transitions can occur, otherwise behavior is UNPREDICTABLE.

The PWR_DEVACTIVEEN and OP_DEVACTIVEEN fields in this register control the ability of the DEVACTIVE inputs to initiate power mode transitions, but not the ability to generate input edge interrupt events.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x020

Access type

RW

Reset value

```
xxxx 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

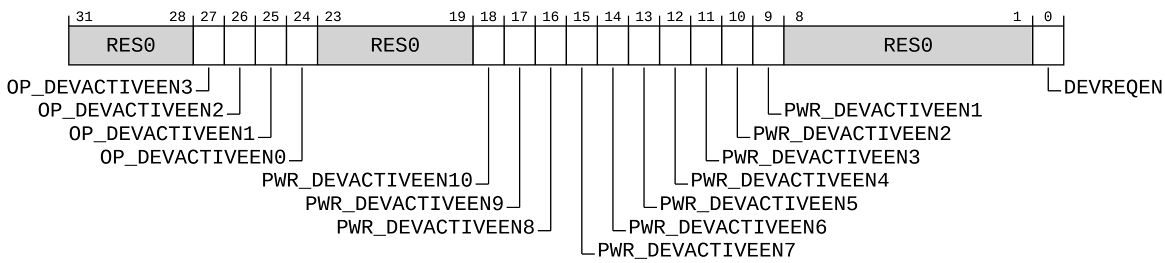


Table B-131: PPU_PWCR bit descriptions

Bits	Name	Description	Reset
[31:28]	**RES0**	Reserved	**RES0**
[27]	OP_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[19] input. **0b0** DEVPACTIVE[19] input (Half L3 Cache Slices active) disabled. **0b1** DEVPACTIVE[19] input (Half L3 Cache Slices active) enabled.	0b1
[26]	OP_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[18] input. **0b0** DEVPACTIVE[18] input (All L3 Cache Slices active) disabled. **0b1** DEVPACTIVE[18] input (All L3 Cache Slices active) enabled.	0b1
[25]	OP_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[17] input. **0b0** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) disabled. **0b1** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) enabled.	0b1
[24]	OP_DEVACTIVEEN0	Enables the operating mode DEVPACTIVE[16] input. **0b0** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) disabled. **0b1** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) enabled.	0b1
[23:19]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[18]	PWR_DEVACTIVEEN10	Enables the operating mode DEVPACTIVE[10] input. **0b0** DEVPACTIVE[10] input (DBG_RECOV) disabled. **0b1** DEVPACTIVE[10] input (DBG_RECOV) enabled.	0b1
[17]	PWR_DEVACTIVEEN9	Enables the operating mode DEVPACTIVE[9] input. **0b0** DEVPACTIVE[9] input (WARM_RST) disabled. **0b1** DEVPACTIVE[9] input (WARM_RST) enabled.	0b1
[16]	PWR_DEVACTIVEEN8	Enables the operating mode DEVPACTIVE[8] input. **0b0** DEVPACTIVE[8] input (ON) disabled. **0b1** DEVPACTIVE[8] input (ON) enabled.	0b1
[15]	PWR_DEVACTIVEEN7	Enables the operating mode DEVPACTIVE[7] input. **0b0** DEVPACTIVE[7] input (FUNC_RET) disabled. **0b1** DEVPACTIVE[7] input (FUNC_RET) enabled.	0b1
[14]	PWR_DEVACTIVEEN6	Enables the operating mode DEVPACTIVE[6] input. **0b1** DEVPACTIVE[6] input (MEM_OFF) enabled.	0b1
[13]	PWR_DEVACTIVEEN5	Enables the operating mode DEVPACTIVE[5] input. **0b0** DEVPACTIVE[5] input (FULL_RET) disabled. **0b1** DEVPACTIVE[5] input (FULL_RET) enabled.	0b1
[12]	PWR_DEVACTIVEEN4	Enables the operating mode DEVPACTIVE[4] input. **0b1** DEVPACTIVE[4] input (LOGIC_RET) enabled.	0b1
[11]	PWR_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[3] input. **0b0** DEVPACTIVE[3] input (MEM_RET_EMU) disabled. **0b1** DEVPACTIVE[3] input (MEM_RET_EMU) enabled.	0b1
[10]	PWR_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[2] input. **0b0** DEVPACTIVE[2] input (MEM_RET) disabled. **0b1** DEVPACTIVE[2] input (MEM_RET) enabled.	0b1

Bits	Name	Description	Reset
[9]	PWR_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[1] input. **0b0** DEVPACTIVE[1] input (OFF_EMU) disabled. **0b1** DEVPACTIVE[1] input (OFF_EMU) enabled.	0b1
[8:1]	**RES0**	Reserved	**RES0**
[0]	DEVREQEN	Device interface handshake enable. **0b0** Device interface handshake disabled for transitions. **0b1** Device interface handshake enabled for transitions.	0b1

B.1.4.9 PPU_PTCR, Power Mode Transition Register

This register contains settings which affect the behaviour of certain power mode transitions.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x024

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-65: ext_ppu_ptcr bit assignments

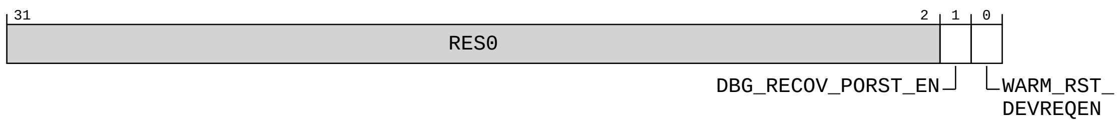


Table B-132: PPU_PTCR bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	DBG_RECOV_PORST_EN	Power-on reset behavior in DBG_RECOV. This bit should not be modified when the PPU is in DBG_RECOV or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **`0b0`** DEVPORESETn is not asserted when in DBG_RECOV. **`0b1`** DEVPORESETn is asserted when in DBG_RECOV.	`0b0`
[0]	WARM_RST_DEVREQEN	Device interface handshake behavior. This bit should not be modified when the PPU is in WARM_RST, or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **`0b0`** The PPU does not perform a device interface handshake when transitioning between ON and WARM_RST. **`0b1`** The PPU performs a device interface handshake when transitioning between ON and WARM_RST.	`0b0`

B.1.4.10 PPU_IMR, Interrupt Mask Register

This register controls the events that assert the interrupt output. Additional event masking controls are in the Additional Interrupt Mask Register (ext-PPU_AIMR), Input Edge Sensitivity Register (ext-PPU_IESR), and the Operating Mode Active Edge Sensitivity Register (ext-PPU_OPSR).

When an interrupt event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x030

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xx11 1010
|    |    |    |    |    |    |    |   |
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

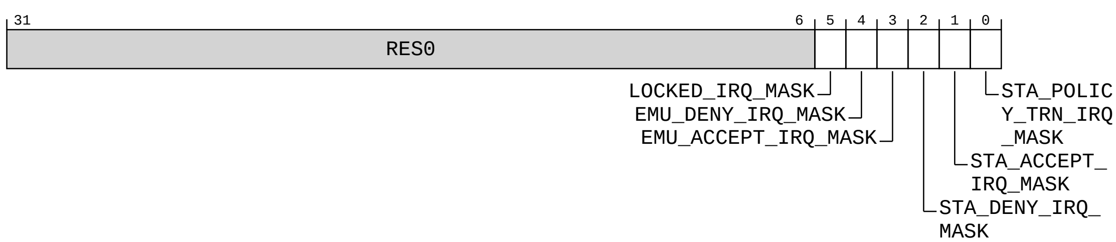


Table B-133: PPU_IMR bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	LOCKED_IRQ_MASK	Locked event mask **0b0** Locked event enabled. **0b1** Locked event masked.	0b1
[4]	EMU_DENY_IRQ_MASK	Emulation transition denial event mask **0b0** Emulation transition denial event enabled. **0b1** Emulation transition denial event masked.	0b1

Bits	Name	Description	Reset
[3]	EMU_ACCEPT_IRQ_MASK	Emulation transition acceptance event mask **0b0** Emulation transition acceptance event enabled. **0b1** Emulation transition acceptance event masked.	0b1
[2]	STA_DENY_IRQ_MASK	Static transition denial event mask **0b0** Static transition denial event enabled. **0b1** Static transition denial event masked.	0b0
[1]	STA_ACCEPT_IRQ_MASK	Static transition acceptance event mask **0b0** Static transition acceptance event enabled. **0b1** Static transition acceptance event masked.	0b1
[0]	STA_POLICY_TRN_IRQ_MASK	Static full policy transition completion event mask **0b0** Static full policy transition completion event enabled. **0b1** Static full policy transition completion event masked.	0b0

B.1.4.11 PPU_AIMR, Additional Interrupt Mask Register

This register controls the events that assert the interrupt output. Additional event masking controls are in the Interrupt Mask Register (ext-PPU_IMR), Input Edge Sensitivity Register (ext-PPU_IESR), and the Operating Mode Active Edge Sensitivity Register (ext-PPU_OPSR).

When an interrupt event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x034

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxx1 1110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-67: ext_ppu_aimr bit assignments

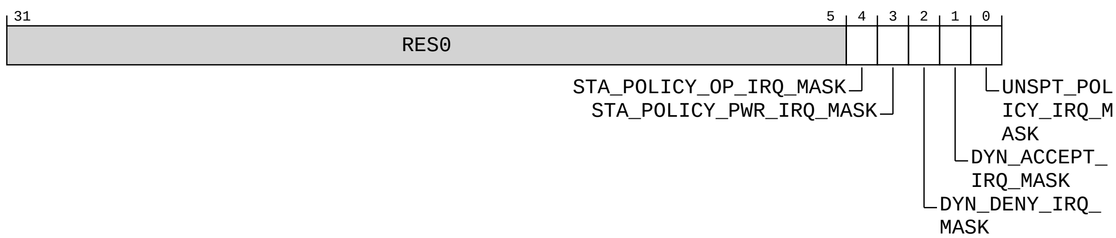


Table B-134: PPU_AIMR bit descriptions

Bits	Name	Description	Reset
[31:5]	RES0	Reserved	RES0
[4]	STA_POLICY_OP_IRQ_MASK	Static operating policy transition completion event mask **0b0** Static operating policy transition completion event enabled. **0b1** Static operating policy transition completion event masked.	0b1
[3]	STA_POLICY_PWR_IRQ_MASK	Static power policy transition completion event mask **0b0** Static power policy transition completion event enabled. **0b1** Static power policy transition completion event masked.	0b1
[2]	DYN_DENY_IRQ_MASK	Dynamic transition denial event mask **0b0** Dynamic transition denial event enabled. **0b1** Dynamic transition denial event masked.	0b1

Bits	Name	Description	Reset
[1]	DYN_ACCEPT_IRQ_MASK	Dynamic transition acceptance event mask **0b0** Dynamic transition acceptance event enabled. **0b1** Dynamic transition acceptance event masked.	0b1
[0]	UNSPT_POLICY_IRQ_MASK	Unsupported policy event mask **0b0** Unsupported policy event enabled. **0b1** Unsupported policy event masked.	0b0

B.1.4.12 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-68: ext_ppu_isr bit assignments

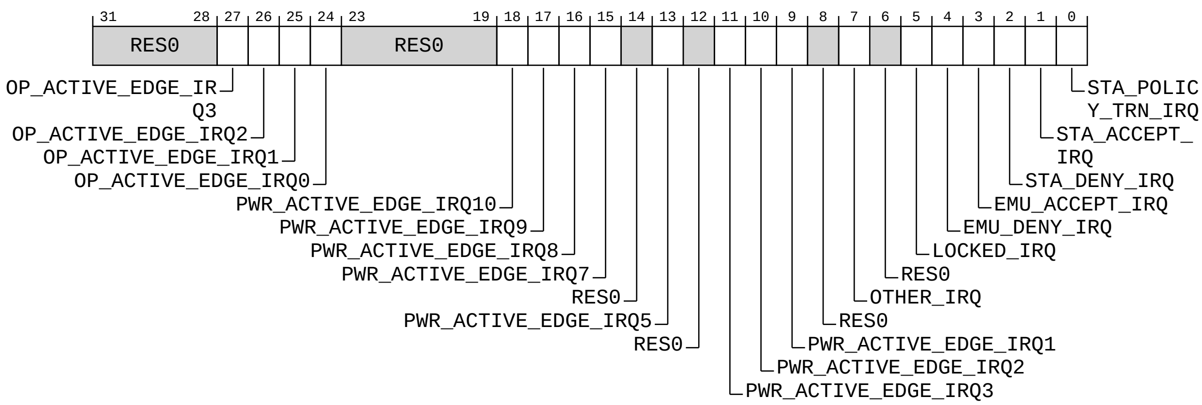


Table B-135: PPU_ISR bit descriptions

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. **0b0** DEVPACTIVE[19] input (Half L3 Cache Slices active) did not assert the interrupt output. **0b1** 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. **0b0** DEVPACTIVE[18] input (All L3 Cache Slices active) did not assert the interrupt output. **0b1** DEVPACTIVE[18] input (All L3 Cache Slices active) asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[25]	OP_ACTIVE_EDGE_IRQ1	Indicates if operating mode DEVPACTIVE[17] input caused the input edge event. **0b0** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) did not assert the interrupt output. **0b1** DEVPACTIVE[17] input (Upper L3 Cache RAMs active) asserted the interrupt output.	0b0
[24]	OP_ACTIVE_EDGE_IRQ0	Indicates if operating mode DEVPACTIVE[16] input caused the input edge event. **0b0** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) did not assert the interrupt output. **0b1** DEVPACTIVE[16] input (Lower L3 Cache RAMs active) asserted the interrupt output.	0b0
[23:19]	**RES0**	Reserved	**RES0**
[18]	PWR_ACTIVE_EDGE_IRQ10	Indicates if power mode DEVPACTIVE[10] input caused the input edge event. **0b0** DEVPACTIVE[10] input (DBG_RECOV) did not assert the interrupt output. **0b1** DEVPACTIVE[10] input (DBG_RECOV) asserted the interrupt output.	0b0
[17]	PWR_ACTIVE_EDGE_IRQ9	Indicates if power mode DEVPACTIVE[9] input caused the input edge event. **0b0** DEVPACTIVE[9] input (WARM_RST) did not assert the interrupt output. **0b1** DEVPACTIVE[9] input (WARM_RST) asserted the interrupt output.	0b0
[16]	PWR_ACTIVE_EDGE_IRQ8	Indicates if power mode DEVPACTIVE[8] input caused the input edge event. **0b0** DEVPACTIVE[8] input (ON) did not assert the interrupt output. **0b1** DEVPACTIVE[8] input (ON) asserted the interrupt output.	0b0
[15]	PWR_ACTIVE_EDGE_IRQ7	Indicates if power mode DEVPACTIVE[7] input caused the input edge event. **0b0** DEVPACTIVE[7] input (FUNC_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[7] input (FUNC_RET) asserted the interrupt output.	0b0
[14]	**RES0**	Reserved	**RES0**
[13]	PWR_ACTIVE_EDGE_IRQ5	Indicates if power mode DEVPACTIVE[5] input caused the input edge event. **0b0** DEVPACTIVE[5] input (FULL_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[5] input (FULL_RET) asserted the interrupt output.	0b0
[12]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset

[11]	PWR_ACTIVE_EDGE_IRQ3	Indicates if power mode DEVPACTIVE[3] input caused the input edge event.
		`0b0`
		DEVPACTIVE[3] input (MEM_RET_EMU) did not assert the interrupt output.
		`0b1`
		DEVPACTIVE[3] input (MEM_RET_EMU) asserted the interrupt output.
			`0b0`

[10]	PWR_ACTIVE_EDGE_IRQ2	Indicates if power mode DEVPACTIVE[2] input caused the input edge event.
		`0b0`
		DEVPACTIVE[2] input (MEM_RET) did not assert the interrupt output.
		`0b1`
		DEVPACTIVE[2] input (MEM_RET) asserted the interrupt output.
			`0b0`

[9]	PWR_ACTIVE_EDGE_IRQ1	Indicates if power mode DEVPACTIVE[1] input caused the input edge event.
		`0b0`
		DEVPACTIVE[1] input (OFF_EMU) did not assert the interrupt output.
		`0b1`
		DEVPACTIVE[1] input (OFF_EMU) asserted the interrupt output.
			`0b0`

[8]	**RES0**	Reserved	**RES0**

[7]	OTHER_IRQ	Indicates there is an interrupt event pending in the Additional Interrupt Status Register (ext-PPU_AISR).
		`0b0`
		No interrupt pending in ext-PPU_AISR.
		`0b1`
		Interrupt pending in ext-PPU_AISR.
			`0b0`

[6]	**RES0**	Reserved	**RES0**

[5]	LOCKED_IRQ	Locked event status.
		`0b0`
		No locked event.
		`0b1`
		A locked event asserted the interrupt output.
			`0b0`

[4]	EMU_DENY_IRQ	Emulated transition denial event status.
		`0b0`
		No emulated transition denial event.
		`0b1`
		An emulated transition denial event asserted the interrupt output.
			`0b0`

[3]	EMU_ACCEPT_IRQ	Emulated transition acceptance event status.
		`0b0`
		No emulated transition acceptance event.
		`0b1`
		An emulated transition acceptance event asserted the interrupt output.
			`0b0`

Bits	Name	Description	Reset
[2]	STA_DENY_IRQ	Static transition denial event status. **0b0** No static transition denial event. **0b1** An static transition denial event asserted the interrupt output.	0b0
[1]	STA_ACCEPT_IRQ	Static transition acceptance event status. **0b0** No static transition acceptance event. **0b1** An static transition acceptance event asserted the interrupt output.	0b0
[0]	STA_POLICY_TRN_IRQ	Static full policy transition completion event status. **0b0** No static full policy transition completion event. **0b1** An static full policy transition completion event asserted the interrupt output.	0b0

B.1.4.13 PPU_AISR, Additional Interrupt Status Register

This register contains information about events causing the assertion of the interrupt output. It is also used to clear interrupt events.

A bit set to 0b1 indicates the event asserted the interrupt output. Multiple events can be active at the same time. When an interrupt event is masked an occurrence of that event does not set the status bit.

A write of 0b1 to an event bit clears that event. A write of 0b0 has no effect. The interrupt output stays HIGH until all status bits in the Interrupt Status Register (ext-PPU_ISR) and the Additional Interrupt Status Register (PPU_AISR) are set to 0b0.

When an interrupt status is set to 0b1 in this register it sets the OTHER_IRQ bit in the Interrupt Status Register (ext-PPU_ISR). Status bits in this register are only cleared by writing to this register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x03C

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxx0 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-69: ext_ppu_aisr bit assignments

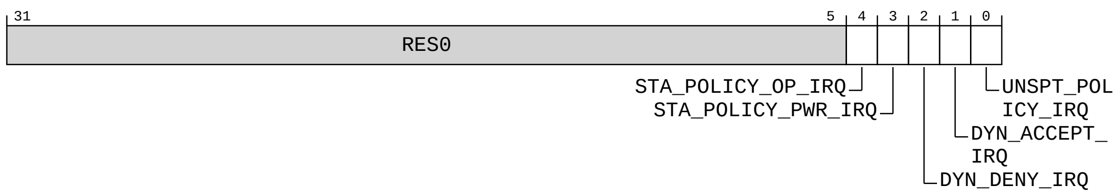


Table B-136: PPU_AISR bit descriptions

Bits	Name	Description	Reset
[31:5]	**RES0**	Reserved	**RES0**
[4]	STA_POLICY_OP_IRQ	Static operating policy transition completion event status **0b0** No static operating policy transition completion event. **0b1** A static operating policy transition completion event asserted the interrupt output.	0b0
[3]	STA_POLICY_PWR_IRQ	Static power policy transition completion event status **0b0** No static power policy transition completion event. **0b1** A static power policy transition completion event asserted the interrupt output.	0b0
[2]	DYN_DENY_IRQ	Dynamic transition denial event status **0b0** No dynamic transition denial event. **0b1** A dynamic transition denial event asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[1]	DYN_ACCEPT_IRQ	Dynamic transition acceptance event status **0b0** No dynamic transition acceptance event. **0b1** A dynamic transition acceptance event asserted the interrupt output.	0b0
[0]	UNSPT_POLICY_IRQ	Unsupported policy event status **0b0** No unsupported policy event. **0b1** An unsupported policy event asserted the interrupt output.	0b0

B.1.4.14 PPU_IESR, Input Edge Sensitivity Register

This register configures the transitions on the power mode DEVPACTIVE inputs that generate an Input Edge interrupt event.

When an event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x040

Access type

RW

Reset value

```
xxxx xxxx xx00 0000 00xx 00xx 0000 00xx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-70: ext_ppu_iesr bit assignments

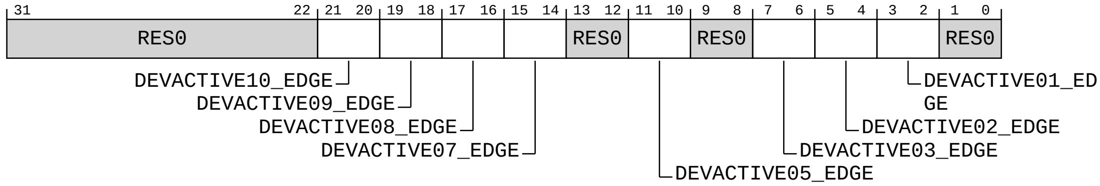


Table B-137: PPU_IESR bit descriptions

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
[17:16]	DEVACTIVE08_EDGE	Configures the transitions on the DEVPACTIVE[8] input (ON) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00

Bits	Name	Description	Reset
[15:14]	DEVACTIVE07_EDGE	Configures the transitions on the DEVPACTIVE[7] input (FUNC_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	`0b00`
[13:12]	**RES0**	Reserved	**RES0**
[11:10]	DEVACTIVE05_EDGE	Configures the transitions on the DEVPACTIVE[5] input (FULL_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	`0b00`
[9:8]	**RES0**	Reserved	**RES0**
[7:6]	DEVACTIVE03_EDGE	Configures the transitions on the DEVPACTIVE[3] input (MEM_RET_EMU) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	`0b00`
[5:4]	DEVACTIVE02_EDGE	Configures the transitions on the DEVPACTIVE[2] input (MEM_RET) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	`0b00`

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. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[1:0]	RES0	Reserved	RES0

B.1.4.15 PPU_OPSR, Operating Mode Active Edge Sensitivity Register

This register configures the transitions on the operating mode DEVPACTIVE inputs that generate an Input Edge interrupt event.

When an event is masked an occurrence of the event does not set the corresponding bit in the interrupt status register.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x044

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-71: ext_ppu_opsr bit assignments

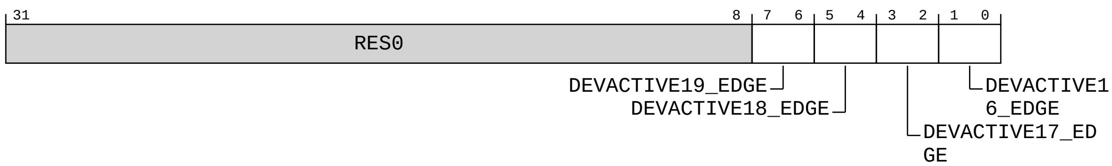


Table B-138: PPU_OPSR bit descriptions

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. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[5:4]	DEVACTIVE18_EDGE	Configures the transitions on the DEVPACTIVE[18] input (All L3 Cache Slices active) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[3:2]	DEVACTIVE17_EDGE	Configures the transitions on the DEVPACTIVE[17] input (Upper L3 Cache RAMs active) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00

Bits	Name	Description	Reset
[1:0]	DEVACTIVE16_EDGE	Configures the transitions on the DEVPACTIVE[16] input (Lower L3 Cache RAMs active) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00

B.1.4.16 PPU_FUNRR, Functional Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x050

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-72: ext_ppu_funrr bit assignments

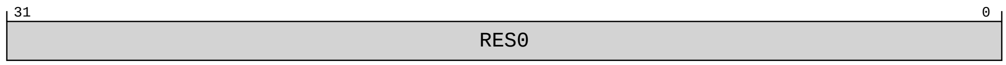


Table B-139: PPU_FUNRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.17 PPU_FULRR, Full Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x054

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-73: ext_ppu_fulrr bit assignments

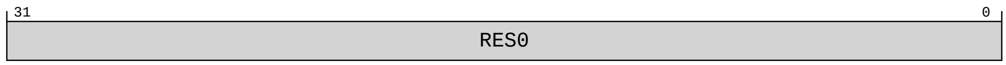


Table B-140: PPU_FULRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.18 PPU_MEMRR, Memory Retention RAM Configuration Register

This register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x058

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-74: ext_ppu_memrr bit assignments

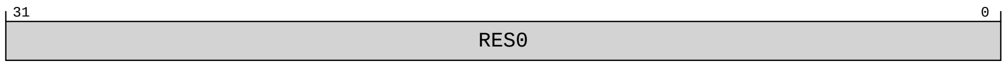


Table B-141: PPU_MEMRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.19 PPU_DCDR0, Device Control Delay Configuration Register 0

This register is used to program device control delay parameters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x170

Access type

RW

Reset value

```
xxxx xxxx 0000 0000 0000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-75: ext_ppu_dcdr0 bit assignments

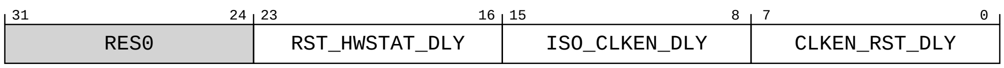


Table B-142: PPU_DCDR0 bit descriptions

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`

B.1.4.20 PPU_DCDR1, Device Control Delay Configuration Register 1

This register is used to program device control delay parameters.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x174

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-76: ext_ppu_dcdr1 bit assignments

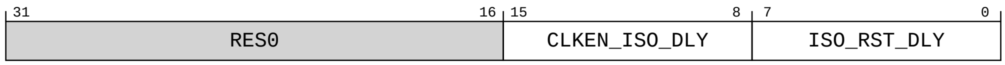


Table B-143: PPU_DCDR1 bit descriptions

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`

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

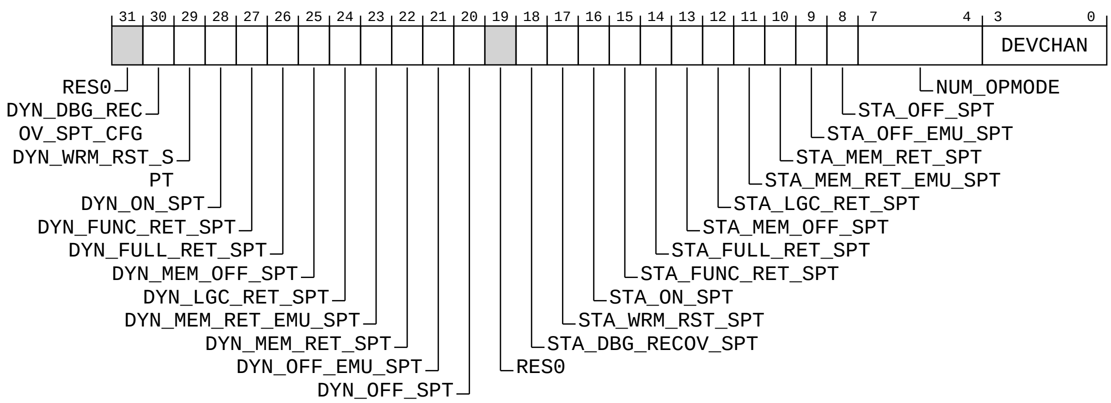


Table B-144: PPU_IDR0 bit descriptions

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 not supported.	0b1
[28]	DYN_ON_SPT	Dynamic ON support. **0b1** Dynamic ON supported.	0b1
[27]	DYN_FUNC_RET_SPT	Dynamic DYN_FUNC_RET_SPT support. **0b1** Dynamic DYN_FUNC_RET_SPT supported.	0b1
[26]	DYN_FULL_RET_SPT	Dynamic DYN_FULL_RET_SPT support. **0b0** Dynamic DYN_FULL_RET_SPT not supported.	0b1
[25]	DYN_MEM_OFF_SPT	Dynamic MEM_OFF support. **0b0** Dynamic MEM_OFF not supported.	0b0

Bits	Name	Description	Reset
[24]	DYN_LGC_RET_SPT	Dynamic LOGIC_RET support. **0b0** Dynamic LOGIC_RET not supported.	0b0
[23]	DYN_MEM_RET_EMU_SPT	Dynamic DYN_MEM_RET_EMU_SPT support. **0b1** Dynamic DYN_MEM_RET_EMU_SPT supported.	0b1
[22]	DYN_MEM_RET_SPT	Dynamic DYN_MEM_RET_SPT support. **0b1** Dynamic DYN_MEM_RET_SPT supported.	0b1
[21]	DYN_OFF_EMU_SPT	Dynamic OFF_EMU support. **0b1** Dynamic OFF_EMU supported.	0b1
[20]	DYN_OFF_SPT	Dynamic OFF support. **0b1** Dynamic OFF supported.	0b1
[19]	RES0	Reserved	RES0
[18]	STA_DBG_RECOV_SPT	DBG_RECOV support. **0b1** DBG_RECOV supported.	0b1
[17]	STA_WRM_RST_SPT	WARM_RST support. **0b1** WRM_RST supported.	0b1
[16]	STA_ON_SPT	ON support. **0b1** ON supported.	0b1
[15]	STA_FUNC_RET_SPT	FUNC_RET support. **0b1** FUNC_RET supported.	0b1
[14]	STA_FULL_RET_SPT	FULL_RET support. **0b0** FULL_RET not supported.	0b1
[13]	STA_MEM_OFF_SPT	MEM_OFF support. **0b0** MEM_OFF not supported.	0b0
[12]	STA_LGC_RET_SPT	LOGIC_RET support. **0b0** LOGIC_RET not supported.	0b0
[11]	STA_MEM_RET_EMU_SPT	MEM_RET_EMU support. **0b1** MEM_RET_EMU supported.	0b1

Bits	Name	Description	Reset
[10]	STA_MEM_RET_SPT	MEM_RET support. **0b1** MEM_RET supported.	0b1
[9]	STA_OFF_EMU_SPT	OFF_EMU support. **0b1** OFF_EMU supported.	0b1
[8]	STA_OFF_SPT	OFF support. **0b1** OFF supported.	0b1
[7:4]	NUM_OPMODE	No. of operating modes supported, minus 1. **0b1111** 16 operating modes supported.	0b1111
[3:0]	DEVCHAN	No. of Device Interface Channels. **0b0000** 0 (P-channel PPU).	0b0000

B.1.4.22 PPU_IDR1, PPU Identification Register 1

This read-only register contains information on the optional features and configurations that are supported by this PPU.

Additional information on optional features can be found in the PPU Identification Register 0 (ext-PPU_IDR0).

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFB4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxx1 x111 x000 x110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-78: ext_ppu_idr1 bit assignments

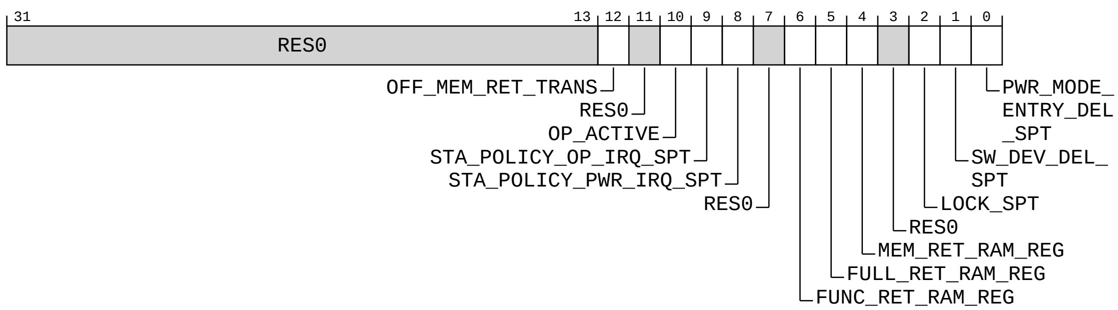


Table B-145: PPU_IDR1 bit descriptions

Bits	Name	Description	Reset
[31:13]	RES0	Reserved	RES0
[12]	OFF_MEM_RET_TRANS	OFF to MEM_RET direct transition. Indicates if direct transitions from OFF to MEM_RET and from OFF_EMU to MEM_RET_EMU are supported. **0b1** OFF to MEM_RET direct transition supported.	0b1
[11]	RES0	Reserved	RES0
[10]	OP_ACTIVE	Operating mode use model for dynamic transitions. **0b1** Independent use model.	0b1
[9]	STA_POLICY_OP_IRQ_SPT	Operating policy transition completion event status. **0b1** Operating policy transition completion events supported.	0b1
[8]	STA_POLICY_PWR_IRQ_SPT	Power policy transition completion event status. **0b1** Power policy transition completion events supported.	0b1
[7]	RES0	Reserved	RES0
[6]	FUNC_RET_RAM_REG	Indicates if the ext-PPU_FUNRR register is present or reserved. **0b0** ext-PPU_FUNRR is reserved.	0b0
[5]	FULL_RET_RAM_REG	Indicates if the ext-PPU_FULRR register is present or reserved. **0b0** ext-PPU_FULRR is reserved.	0b0

Bits	Name	Description	Reset
[4]	MEM_RET_RAM_REG	Indicates if the ext-PPU_MEMRR register is present or reserved. **0b0** ext-PPU_MEMRR is reserved.	0b0
[3]	**RES0**	Reserved	**RES0**
[2]	LOCK_SPT	Indicates if the lock and the lock interrupt event are supported. **0b1** Lock and the lock interrupt event are supported.	0b1
[1]	SW_DEV_DEL_SPT	Software device delay control configuration support. **0b1** Software device delay control configuration supported.	0b1
[0]	PWR_MODE_ENTRY_DEL_SPT	Power mode entry delay support. **0b0** Power mode entry delay not supported.	0b0

B.1.4.23 PPU_IIDR, Implementation Identification Register

This register provides information about the implementer and implementation of the PPU.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFC8

Access type

RO

Reset value

0000 1011 0110 0010 0000 0100 0011 1011

Bit descriptions

Figure B-79: ext_ppu_iidr bit assignments

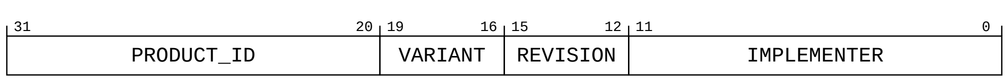


Table B-146: PPU_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	PRODUCT_ID	Value identifying the PPU part. **`0b000010110110`** Power Policy Unit.	`0x0B6`
[19:16]	VARIANT	Value used to distinguish PPU variants, or major revisions of the PPU. **`0b0000`** PPU variant 0. **`0b0001`** PPU variant 1. **`0b0010`** PPU variant 2. **`0b0011`** PPU variant 3. **`0b0100`** PPU variant 4.	`0b0010`
[15:12]	REVISION	Value used to distinguish minor revisions of the PPU. **`0b0000`** PPU revision 0. **`0b0001`** PPU revision 1. **`0b0010`** PPU revision 2. **`0b0011`** PPU revision 3. **`0b0100`** PPU revision 4.	`0b0000`
[11:0]	IMPLEMENTER	Implementer identification. **`0b010000111011`** Arm Limited.	`0x43B`

B.1.4.24 PPU_AIDR, Architecture Identification Register

This register identifies the PPU architecture revision.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFCC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0001 0010
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-80: ext_ppu_aidr bit assignments

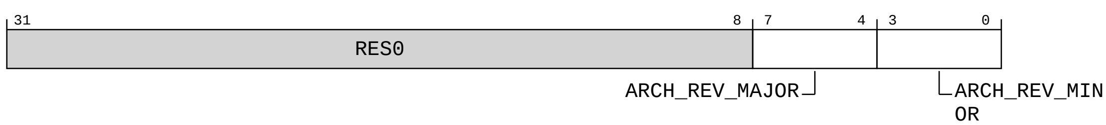


Table B-147: PPU_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	ARCH_REV_MAJOR	PPU architecture major revision. **0b0001** PPU architecture major revision 1.	0b0001
[3:0]	ARCH_REV_MINOR	PPU architecture minor revision. **0b0010** PPU architecture minor revision 2.	0b0010

B.1.4.25 PPU_PIDR4, PPU Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFD0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-81: ext_ppu_pidr4 bit assignments

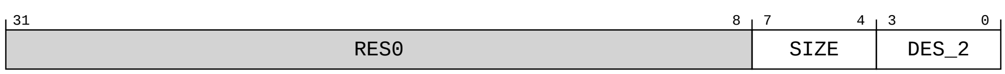


Table B-148: PPU_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	xxxx
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	xxxx

B.1.4.26 PPU_PIDR5, PPU Peripheral Identification Register 5

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFD4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-82: ext_ppu_pidr5 bit assignments

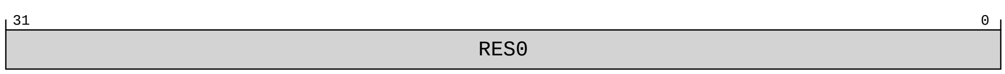


Table B-149: PPU_PIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.27 PPU_PIDR6, PPU Peripheral Identification Register 6

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFD8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-83: ext_ppu_pidr6 bit assignments

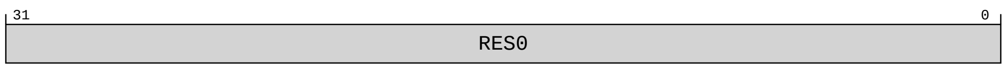


Table B-150: PPU_PIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

B.1.4.28 PPU_PIDR7, PPU Peripheral Identification Register 7

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFDC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |   |
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

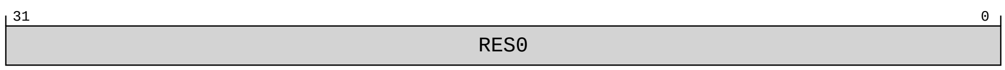


Table B-151: PPU_PIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

B.1.4.29 PPU_PIDR0, PPU Peripheral Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFE0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0110
|    |    |    |    |    |    |    |   |
```

```
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-85: ext_ppu_pidr0 bit assignments

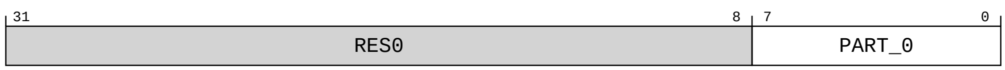


Table B-152: PPU_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. **0b10110110** DSU-C1 Power Policy Unit. Bits [7:0] of part number 0x0B6.	0xB6

B.1.4.30 PPU_PIDR1, PPU Peripheral Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFE4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0000
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-86: ext_ppu_pidr1 bit assignments

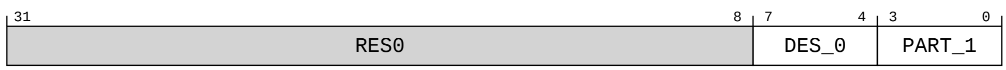


Table B-153: PPU_PIDR1 bit descriptions

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-C1 Power Policy Unit. Bits [11:8] of part number 0x0B6.	0b0000

B.1.4.31 PPU_PIDR2, PPU Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFE8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 1011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-87: ext_ppu_pidr2 bit assignments

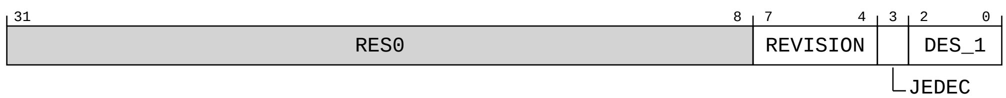


Table B-154: PPU_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. **0b0011** Component major revision 3. **0b0100** Component major revision 4.	0b0010
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

B.1.4.32 PPU_PIDR3, PPU Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFEC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-88: ext_ppu_pidr3 bit assignments

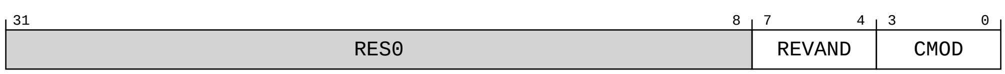


Table B-155: PPU_PIDR3 bit descriptions

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

Bits	Name	Description	Reset
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

B.1.4.33 PPU_CIDR0, PPU Component Identification Register 0

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFF0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 1101
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-89: ext_ppu_cidr0 bit assignments

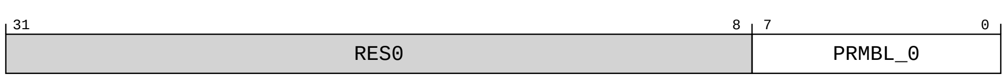


Table B-156: PPU_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

B.1.4.34 PPU_CIDR1, PPU Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFF4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1111 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-90: ext_ppu_cidr1 bit assignments

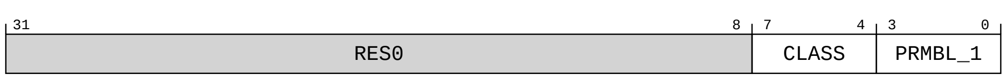


Table B-157: PPU_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	CLASS	CoreSight component class. **0b1111** CoreLink component.	0b1111
[3:0]	PRMBL_1	CoreSight component identification preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.1.4.35 PPU_CIDR2, PPU Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFF8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-91: ext_ppu_cidr2 bit assignments

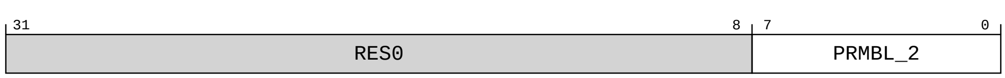


Table B-158: PPU_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	CoreSight component identification preamble. **`0b00000101`** CoreSight component identification preamble.	`0x05`

B.1.4.36 PPU_CIDR3, PPU Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0xFFC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-92: ext_ppu_cidr3 bit assignments

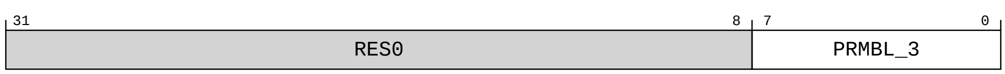


Table B-159: PPU_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **`0b10110001`** CoreSight component identification preamble.	`0xB1`

B.1.5 External 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 C1-DSU. 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.
- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers is treated as **RAZ/WI**.
- Any address that is not documented is treated as **RAZ/WI**.
- The part number is `0x04EE`.
- 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-160: CLUSTERAMU registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0	CLUSTERAMU_AMEVCNTR0 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x4	CLUSTERAMU_AMEVCNTR0 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x8	CLUSTERAMU_AMEVCNTR1 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0xC	CLUSTERAMU_AMEVCNTR1 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x10	CLUSTERAMU_AMEVCNTR2 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0x14	CLUSTERAMU_AMEVCNTR2 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x18	CLUSTERAMU_AMEVCNTR3 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x1C	CLUSTERAMU_AMEVCNTR3 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x20	CLUSTERAMU_AMEVCNTR4 [31:0]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x24	CLUSTERAMU_AMEVCNTR4 [63:32]	See individual bit resets.	32-bit	Activity Monitors Event Counter Registers 0	No
0x400	CLUSTERAMU_AMEVTYPER0	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x404	CLUSTERAMU_AMEVTYPER1	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x408	CLUSTERAMU_AMEVTYPER2	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x40C	CLUSTERAMU_AMEVTYPER3	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0x410	CLUSTERAMU_AMEVTYPER4	See individual bit resets.	32-bit	Activity Monitors Event Type Registers 0	No
0xC00	CLUSTERAMU_AMCNTENSET	See individual bit resets.	32-bit	Activity Monitors Count Enable Set Register 0	No
0xC20	CLUSTERAMU_AMCNTENCLR	See individual bit resets.	32-bit	Activity Monitors Count Enable Clear Register 0	No
0xE00	CLUSTERAMU_AMCFGR	See individual bit resets.	32-bit	Activity Monitors Configuration Register	No
0xE04	CLUSTERAMU_AMCR	See individual bit resets.	32-bit	Activity Monitors Control Register	No
0xE08	CLUSTERAMU_AMIIDR	See individual bit resets.	32-bit	Activity Monitors Implementation Identification Register	No
0xFA8	CLUSTERAMU_AMDEVAFF	See individual bit resets.	64-bit	Activity Monitors Device Affinity Register 0	No
0xFBC	CLUSTERAMU_AMDEVARCH	See individual bit resets.	32-bit	Activity Monitors Device Architecture Register	No
0xFC8	CLUSTERAMU_AMDEVID	See individual bit resets.	32-bit	Cluster Activity Monitors Device ID register	No
0xFCC	CLUSTERAMU_AMDEVTYPE	See individual bit resets.	32-bit	Activity Monitors Device Type Register	No
0xFD0	CLUSTERAMU_AMPIDR4	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 4	No
0xFE0	CLUSTERAMU_AMPIDR0	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 0	No
0xFE4	CLUSTERAMU_AMPIDR1	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 1	No
0xFE8	CLUSTERAMU_AMPIDR2	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 2	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFEC	CLUSTERAMU_AMPIDR3	See individual bit resets.	32-bit	Activity Monitors Peripheral Identification Register 3	No
0xFF0	CLUSTERAMU_AMCIDR0	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 0	No
0xFF4	CLUSTERAMU_AMCIDR1	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 1	No
0xFF8	CLUSTERAMU_AMCIDR2	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 2	No
0xFFC	CLUSTERAMU_AMCIDR3	See individual bit resets.	32-bit	Activity Monitors Component Identification Register 3	No

B.1.5.1 CLUSTERAMU_AMEVCNTR0, Activity Monitors Event Counter Registers 0

Provides access to the architected activity monitor event counters.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offsets (2)

0x0,0x4

Access type

Read

R

Write

W

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-93: ext_clusteramu_amevcntr0 bit assignments

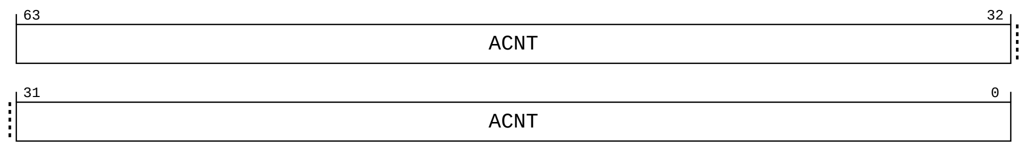


Table B-161: CLUSTERAMU_AMEVCNTR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	ACNT	Architected activity monitor event counter n. Value of architected activity monitor event counter n, where n is the number of this register and is a number from 0 to 3.	0x0000000000000000

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVCNTR<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x0	CLUSTERAMU_AMEVCNTR0	31:0

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
AMU	0x4	AMEVCNTR0	63:32

This interface is accessible as follows:

RO

B.1.5.2 CLUSTERAMU_AMEVCNTR1, Activity Monitors Event Counter Registers 0

Provides access to the architected activity monitor event counters.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offsets (2)

0x8,0xC

Access type

Read

R

Write

W

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-94: ext_clusteramu_amevcntr1 bit assignments

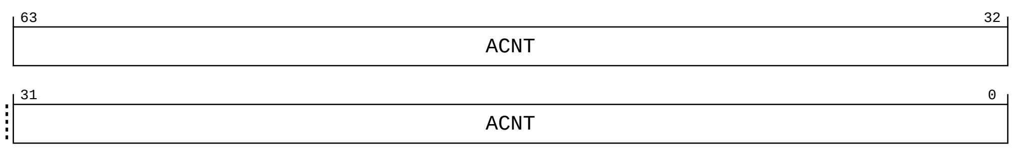


Table B-164: CLUSTERAMU_AMEVCNTR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	ACNT	Architected activity monitor event counter n. Value of architected activity monitor event counter n, where n is the number of this register and is a number from 0 to 3.	`0x0000000000000000`

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVCNTR<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x8	CLUSTERAMU_AMEVCNTR1	31:0

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
AMU	0xC	AMEVCNTR1	63:32

This interface is accessible as follows:

RO

B.1.5.3 CLUSTERAMU_AMEVCNTR2, Activity Monitors Event Counter Registers 0

Provides access to the architected activity monitor event counters.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offsets (2)

0x10,0x14

Access type

Read

R

Write

W

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-95: ext_clusteramu_amevcntr2 bit assignments

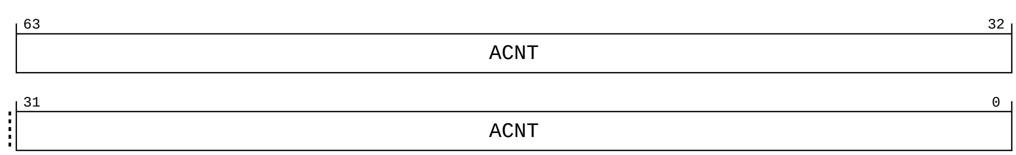


Table B-167: CLUSTERAMU_AMEVCNTR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	ACNT	Architected activity monitor event counter n. Value of architected activity monitor event counter n, where n is the number of this register and is a number from 0 to 3.	0x0000000000000000

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVCNTR<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x10	CLUSTERAMU_AMEVCNTR2	31:0

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
AMU	0x14	AMEVCNTR2	63:32

This interface is accessible as follows:

RO

B.1.5.4 CLUSTERAMU_AMEVCNTR3, Activity Monitors Event Counter Registers 0

Provides access to the architected activity monitor event counters.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offsets (2)

0x18,0x1C

Access type

Read

R

Write

W

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-96: ext_clusteramu_amevcntr3 bit assignments

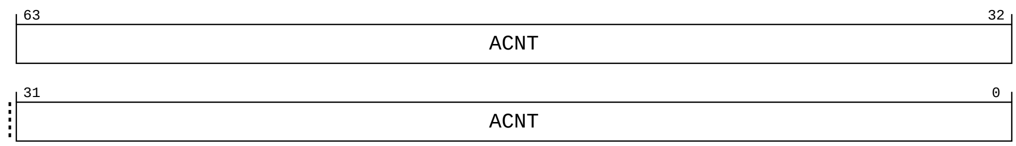


Table B-170: CLUSTERAMU_AMEVCNTR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	ACNT	Architected activity monitor event counter n. Value of architected activity monitor event counter n, where n is the number of this register and is a number from 0 to 3.	0x0000000000000000

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVCNTR<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x18	CLUSTERAMU_AMEVCNTR3	31:0

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
AMU	0x1C	AMEVCNTR3	63:32

This interface is accessible as follows:

RO

B.1.5.5 CLUSTERAMU_AMEVCNTR4, Activity Monitors Event Counter Registers 0

Provides access to the architected activity monitor event counters.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERAMU

Register offsets (2)

0x20,0x24

Access type

Read

R

Write

W

Reset value

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-97: ext_clusteramu_amevcntr4 bit assignments

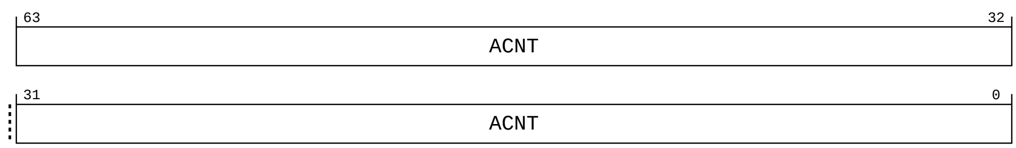


Table B-173: CLUSTERAMU_AMEVCNTR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	ACNT	Architected activity monitor event counter n. Value of architected activity monitor event counter n, where n is the number of this register and is a number from 0 to 3.	`0x0000000000000000`

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVCNTR<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x20	CLUSTERAMU_AMEVCNTR4	31:0

This interface is accessible as follows:

RW

Component	Offset	Instance	Range
AMU	0x24	AMEVCNTR4	63:32

This interface is accessible as follows:

RO

B.1.5.6 CLUSTERAMU_AMEVTYPER0, Activity Monitors Event Type Registers 0

Provides information on the events that an architected activity monitor event counter ext-AMEVCNTR0<n> counts.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x400

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-98: ext_clusteramu_amevtyper0 bit assignments

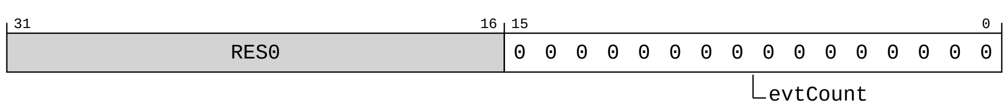


Table B-176: CLUSTERAMU_AMEVTYPER0 bit descriptions

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 the architected activity monitor event counter ext-AMEVCNTR0<n>. The value of this field is architecturally mandated for each architected counter. The following table shows the mapping between required event numbers and the corresponding counters: `0b0000000000000000` L3 cache read hit.	`0x0000`

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVTYPER<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x400	CLUSTERAMU_AMEVTYPER0	None

This interface is accessible as follows:

RO

B.1.5.7 CLUSTERAMU_AMEVTYPER1, Activity Monitors Event Type Registers 0

Provides information on the events that an architected activity monitor event counter ext-AMEVCNTR0<n> counts.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x404

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-99: ext_clusteramu_amevtyper1 bit assignments

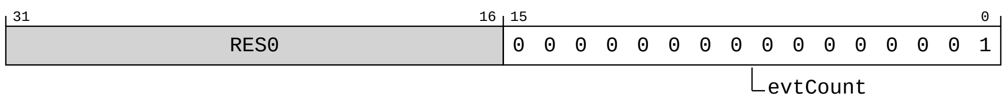


Table B-178: CLUSTERAMU_AMEVTYPER1 bit descriptions

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 the architected activity monitor event counter ext-AMEVCNTR0<n>. The value of this field is architecturally mandated for each architected counter. The following table shows the mapping between required event numbers and the corresponding counters: **0b0000000000000001** L3 cache read miss.	0x0001

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVTYPER<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x404	CLUSTERAMU_AMEVTYPER1	None

This interface is accessible as follows:

RO

B.1.5.8 CLUSTERAMU_AMEVTYPER2, Activity Monitors Event Type Registers 0

Provides information on the events that an architected activity monitor event counter ext-AMEVCNTR0<n> counts.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x408

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0010
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-100: ext_clusteramu_amevtyper2 bit assignments

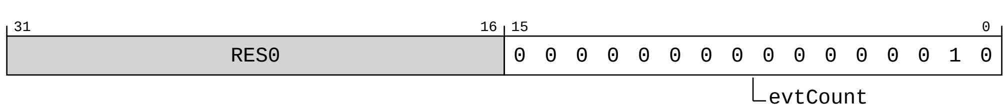


Table B-180: CLUSTERAMU_AMEVTYPER2 bit descriptions

Bits	Name	Description	Reset
[31:16]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[15:0]	evtCount	Event to count. The event number of the event that is counted by the architected activity monitor event counter ext-AMEVCNTR0<n>. The value of this field is architecturally mandated for each architected counter. The following table shows the mapping between required event numbers and the corresponding counters: **`0b0000000000000010`** Post L3 Read occupancy.	`0x0002`

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVTYPER<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x408	CLUSTERAMU_AMEVTYPER2	None

This interface is accessible as follows:

RO

B.1.5.9 CLUSTERAMU_AMEVTYPER3, Activity Monitors Event Type Registers 0

Provides information on the events that an architected activity monitor event counter ext-AMEVCNTR0<n> counts.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x40C

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0011
```

```
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-101: ext_clusteramu_amevtyper3 bit assignments

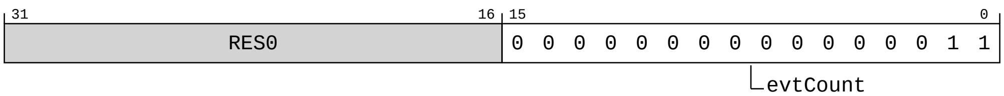


Table B-182: CLUSTERAMU_AMEVTYPER3 bit descriptions

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 the architected activity monitor event counter ext-AMEVCNTR0<n>. The value of this field is architecturally mandated for each architected counter. The following table shows the mapping between required event numbers and the corresponding counters: **0b0000000000000011** Post-L3 Write Transactions.	0x0003

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVTYPER<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x40C	CLUSTERAMU_AMEVTYPER3	None

This interface is accessible as follows:

RO

B.1.5.10 CLUSTERAMU_AMEVTYPER4, Activity Monitors Event Type Registers 0

Provides information on the events that an architected activity monitor event counter ext-AMEVCNTR0<n> counts.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0x410

Access type

Read

R

Write

RESERVED

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-102: ext_clusteramu_amevtyper4 bit assignments

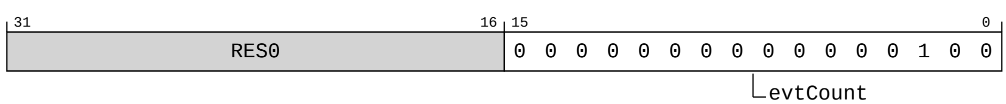


Table B-184: CLUSTERAMU_AMEVTYPER4 bit descriptions

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 the architected activity monitor event counter ext-AMEVCNTR0<n>. The value of this field is architecturally mandated for each architected counter. The following table shows the mapping between required event numbers and the corresponding counters: `0b0000000000000100` Post-L3 Read Transactions.	`0x0004`

Accessibility

If <n> is greater than or equal to the number of architected activity monitor event counters, reads of AMEVTYPER<n> are RAZ. Software must treat reserved accesses as RES0. See 'Access requirements for reserved and unallocated registers'.

Component	Offset	Instance	Range
CLUSTERAMU	0x410	CLUSTERAMU_AMEVTYPER4	None

This interface is accessible as follows:

RO

B.1.5.11 CLUSTERAMU_AMCNTENSET, Activity Monitors Count Enable Set Register 0

Enable control bits for the architected activity monitors event counters, ext-AMEVCNTR0<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xC00

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-103: ext_clusteramu_amcntenset bit assignments

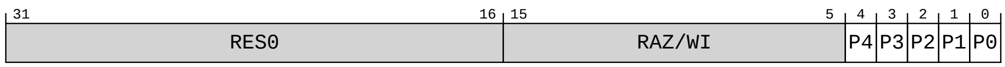


Table B-186: CLUSTERAMU_AMCNTENSET bit descriptions

Bits	Name	Description	Reset
[31:16]	**RES0**	Reserved	**RES0**
[15:5]	**RAZ/WI**	Reserved	**RAZ/WI**
[4]	P4	Activity monitor event counter enable bit for ext-AMEVCNTR0<n>. Possible values of each bit are: **`0b0`** When read, means that ext-AMEVCNTR0<n> is disabled. **`0b1`** When read, means that ext-AMEVCNTR0<n> is enabled.	`0b0`
[3]	P3	Activity monitor event counter enable bit for ext-AMEVCNTR0<n>. Possible values of each bit are: **`0b0`** When read, means that ext-AMEVCNTR0<n> is disabled. **`0b1`** When read, means that ext-AMEVCNTR0<n> is enabled.	`0b0`
[2]	P2	Activity monitor event counter enable bit for ext-AMEVCNTR0<n>. Possible values of each bit are: **`0b0`** When read, means that ext-AMEVCNTR0<n> is disabled. **`0b1`** When read, means that ext-AMEVCNTR0<n> is enabled.	`0b0`
[1]	P1	Activity monitor event counter enable bit for ext-AMEVCNTR0<n>. Possible values of each bit are: **`0b0`** When read, means that ext-AMEVCNTR0<n> is disabled. **`0b1`** When read, means that ext-AMEVCNTR0<n> is enabled.	`0b0`

Bits	Name	Description	Reset
[0]	P0	Activity monitor event counter enable bit for ext-AMEVCNTR0<n>. Possible values of each bit are: **0b0** When read, means that ext-AMEVCNTR0<n> is disabled. **0b1** When read, means that ext-AMEVCNTR0<n> is enabled.	0b0

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xC00	CLUSTERAMU_AMCNTENSET	None

This interface is accessible as follows:

RW

B.1.5.12 CLUSTERAMU_AMCNTENCLR, Activity Monitors Count Enable Clear Register 0

Disable control bits for the architected activity monitors event counters, ext-AMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xC20

Access type

RW

Reset value

```
xxxx xxxx xxxx xxxx 0000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-104: ext_clusteramu_amcntenclr bit assignments

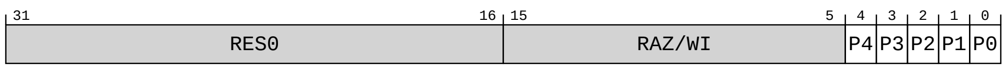


Table B-188: CLUSTERAMU_AMCNTENCLR bit descriptions

Bits	Name	Description	Reset
[31:16]	**RES0**	Reserved	**RES0**
[15:5]	**RAZ/WI**	Reserved	**RAZ/WI**
[4]	P4	Activity monitor event counter disable bit for ext-AMEVCNTR<n>. Possible values of each bit are: **0b0** When read, means that ext-AMEVCNTR<n> is disabled. **0b1** When read, means that ext-AMEVCNTR<n> is enabled.	`0b0`
[3]	P3	Activity monitor event counter disable bit for ext-AMEVCNTR<n>. Possible values of each bit are: **0b0** When read, means that ext-AMEVCNTR<n> is disabled. **0b1** When read, means that ext-AMEVCNTR<n> is enabled.	`0b0`
[2]	P2	Activity monitor event counter disable bit for ext-AMEVCNTR<n>. Possible values of each bit are: **0b0** When read, means that ext-AMEVCNTR<n> is disabled. **0b1** When read, means that ext-AMEVCNTR<n> is enabled.	`0b0`
[1]	P1	Activity monitor event counter disable bit for ext-AMEVCNTR<n>. Possible values of each bit are: **0b0** When read, means that ext-AMEVCNTR<n> is disabled. **0b1** When read, means that ext-AMEVCNTR<n> is enabled.	`0b0`

Bits	Name	Description	Reset
[0]	P0	Activity monitor event counter disable bit for ext-AMEVCNTR<n>. Possible values of each bit are: **0b0** When read, means that ext-AMEVCNTR<n> is disabled. **0b1** When read, means that ext-AMEVCNTR<n> is enabled.	0b0

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xC20	CLUSTERAMU_AMCNTENCLR	None

This interface is accessible as follows:

RW

B.1.5.13 CLUSTERAMU_AMCFGR, Activity Monitors Configuration Register

Global configuration register for the activity monitors.

Provides information on supported features, the number of counter groups implemented, the total number of activity monitor event counters implemented, and the size of the counters. AMCFGR is applicable to both the architected and the auxiliary counter groups.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xE00

Access type

RO

Reset value

```
0000 xxx0 0000 0000 0011 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

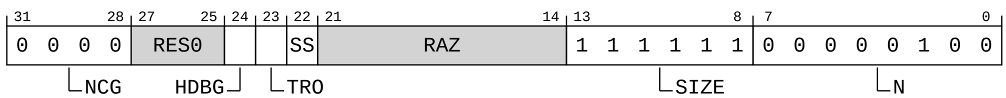


Table B-190: CLUSTERAMU_AMCFGR bit descriptions

Bits	Name	Description	Reset
[31:28]	NCG	Defines the number of counter groups. The number of implemented counter groups is [AMCFGR.NCG + 1]. If the number of implemented auxiliary activity monitor event counters is zero, this field has a value of `0b0000`. Otherwise, this field has a value of `0b0001`. **`0b0000`** Monitor groups are not implemented.	`0b0000`
[27:25]	RES0	Reserved	RES0
[24]	HDBG	Halt on debug. **`0b0`** Halt on debug not supported	`0b0`
[23]	TRO	Trace feature support. **`0b0`** Trace features not supported.	`0b0`
[22]	SS	Snapshot support. **`0b0`** Snapshot not supported.	`0b0`
[21:14]	RAZ	Reserved	RAZ
[13:8]	SIZE	Defines the size of activity monitor event counters. The size of the activity monitor event counters implemented by the Activity Monitors Extension is [AMCFGR.SIZE + 1]. The counters are 64-bit. **Note:** Software also uses this field to determine the spacing of counters in the memory-map. The counters are at doubleword-aligned addresses. **`0b111111`**	`0b111111`

Bits	Name	Description	Reset
[7:0]	N	Defines the number of activity monitor event counters. The total number of counters implemented in all groups by the Activity Monitors Extension is [AMCFGR.N + 1]. **0b00000100** Five event counters implemented.	0x04

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xE00	CLUSTERAMU_AMCFGR	None

This interface is accessible as follows:

RO

B.1.5.14 CLUSTERAMU_AMCR, Activity Monitors Control Register

Global control register for the activity monitors implementation. AMCR is applicable to both the architected and the auxiliary counter groups.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xE04

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-106: ext_clusteramu_amcr bit assignments

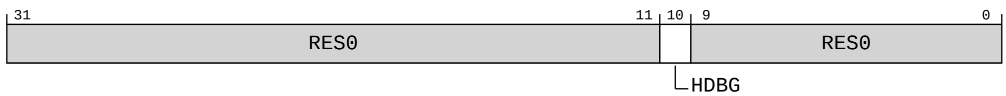


Table B-192: CLUSTERAMU_AMCR bit descriptions

Bits	Name	Description	Reset
[31:11]	RES0	Reserved	RES0
[10]	HDBG	This bit controls whether activity monitor counting is halted when the PE is halted in Debug state. `0b0` Activity monitors do not halt counting when the PE is halted in Debug state.	x
[9:0]	RES0	Reserved	RES0

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xE04	CLUSTERAMU_AMCR	None

This interface is accessible as follows:

RO

B.1.5.15 CLUSTERAMU_AMIIDR, Activity Monitors Implementation Identification Register

Defines the implementer and revisions of the AMU.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xE08

Access type

RO

Reset value

```
0100 1110 1110 0000 xxxx 0100 0011 1011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-107: ext_clusteramu_amiidr bit assignments

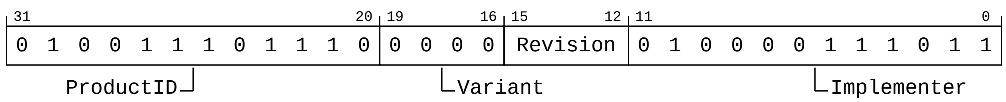


Table B-194: CLUSTERAMU_AMIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	This field is an AMU part identifier. **0b010011101110** DSU-C1 Cluster AMU. If ext-AMPIDR0 is implemented, ext-AMPIDR0.PART_0 matches bits [27:20] of this field. If ext-AMPIDR1 is implemented, ext-AMPIDR1.PART_1 matches bits [31:28] of this field.	`0x4EE`
[19:16]	Variant	This field distinguishes product variants or major revisions of the product. **0b0000** Product Variant 0. If ext-AMPIDR2 is implemented, ext-AMPIDR2.REVISION matches AMIIDR.Variant.	`0b0000`
[15:12]	Revision	This field distinguishes minor revisions of the product. **0b0000** Product Revision 0. **0b0001** Product Revision 1. **0b0010** Product Revision 2. If ext-AMPIDR3 is implemented, ext-AMPIDR3.REVAND matches AMIIDR.Revision.	The reset values can be the following: `0b0000`, `0b0001`, `0b0010`, respective to the value.

Bits	Name	Description	Reset
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the AMU. For an Arm implementation, this field reads as `0x43B`. **`0b010000111011`** Arm implementation. Bits [11:8] contain the JEP106 continuation code of the implementer. Bit 7 is **RES0** Bits [6:0] contain the JEP106 identity code of the implementer. If ext-AMPIDR4 is implemented, ext-AMPIDR4.DES_2 matches bits [11:8] of this field. If ext-AMPIDR2 is implemented, ext-AMPIDR2.DES_1 matches bits [6:4] of this field. If ext-AMPIDR1 is implemented, ext-AMPIDR1.DES_0 matches bits [3:0] of this field.	`0x43B`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xE08	CLUSTERAMU_AMIIDR	None

This interface is accessible as follows:

RO

B.1.5.16 CLUSTERAMU_AMDEVAFF, Activity Monitors Device Affinity Register 0

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

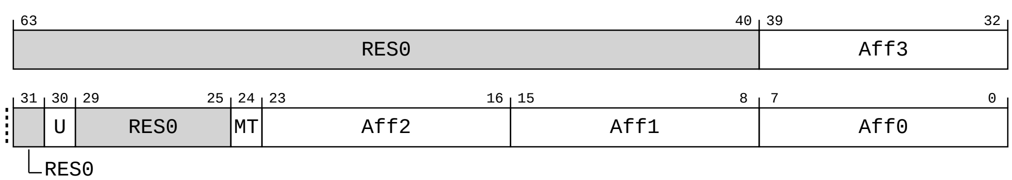


Table B-196: CLUSTERAMU_AMDEVAFF bit descriptions

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. **`0b0`** Processor is part of a multiprocessor system.	`0b0`
[29:25]	**RES0**	Reserved	**RES0**
[24]	MT	Indicates whether the lowest level of affinity consists of logical PEs that are implemented using a multithreading type approach. **`0b0`** Activity of PEs at the lowest affinity level is largely independent.	`0b0`
[23:16]	Aff2	Affinity level 2. Value read from the CFGMPIDRAFF2 configuration pins.	`8{x}`
[15:8]	Aff1	Affinity level 1. **`0b10000000`** Affinity with all cores in cluster.	`0x80`

Bits	Name	Description	Reset
[7:0]	Aff0	Affinity level 0. **0b00000000** Affinity with all core threads in cluster.	0x00

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFA8	CLUSTERAMU_AMDEVAFF	None

This interface is accessible as follows:

RO

B.1.5.17 CLUSTERAMU_AMDEVARCH, Activity Monitors Device Architecture Register

Identifies the programmers' model architecture of the AMU 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

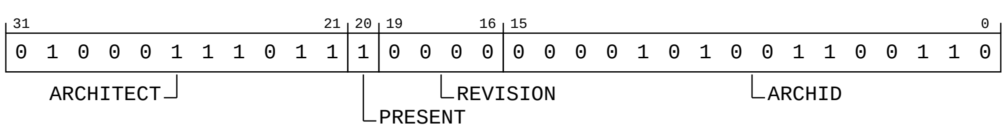


Table B-198: CLUSTERAMU_AMDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Defines the architecture of the component. For AMU, this is Arm Limited. Bits [31:28] are the JEP106 continuation code, `0x4`. Bits [27:21] are the JEP106 ID code, `0x3B`. **`0b01000111011`**	`0b01000111011`
[20]	PRESENT	Indicates that the DEVARCH is present. **`0b1`**	`0b1`
[19:16]	REVISION	Defines the architecture revision. For architectures defined by Arm this is the minor revision. **`0b0000`** Architecture revision is AMUv1. All other values are reserved.	`0b0000`
[15:0]	ARCHID	Defines this part to be an AMU component. For architectures defined by Arm this is further subdivided. For AMU: • Bits [15:12] are the architecture version, `0x0`. • Bits [11:0] are the architecture part number, `0xA66`. This corresponds to AMU architecture version AMUv1. **`0b0000101001100110`**	`0x0A66`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFBC	CLUSTERAMU_AMDEVARCH	None

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

```
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-110: ext_clusteramu_amdevid bit assignments

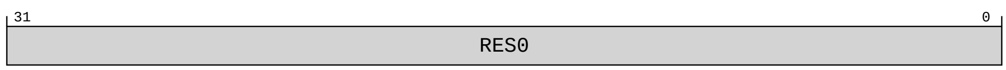


Table B-200: CLUSTERAMU_AMDEVID bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

Accessibility

This interface is accessible as follows:

RO

B.1.5.19 CLUSTERAMU_AMDEVTYPE, Activity Monitors Device Type Register

Indicates to a debugger that this component is part of a PE's activity monitor interface.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFCC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0001 0110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-111: ext_clusteramu_amdevtype bit assignments

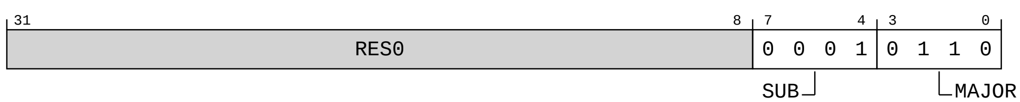


Table B-201: CLUSTERAMU_AMDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SUB	Subtype. `0b0001` Component within a PE.	`0b0001`
[3:0]	MAJOR	Major type. `0b0110` Activity monitor component	`0b0110`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFCC	CLUSTERAMU_AMDEVTYPE	None

This interface is accessible as follows:

RO

B.1.5.20 CLUSTERAMU_AMPIDR4, Activity Monitors Peripheral Identification Register 4

Provides information to identify an activity monitors component.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

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-112: ext_clusteramu_ampidr4 bit assignments

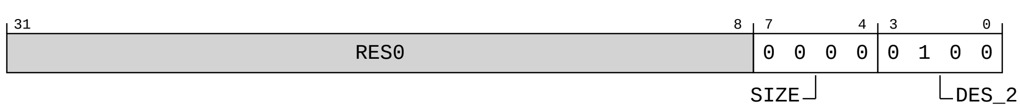


Table B-203: CLUSTERAMU_AMPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	SIZE	Size of the component. Log_2 of the number of 4KB pages from the start of the component to the end of the component ID registers. **0b0000**	0b0000
[3:0]	DES_2	Designer. JEP106 continuation code, least significant nibble. For Arm Limited, this field is 0b0100. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFD0	CLUSTERAMU_AMPIDR4	None

This interface is accessible as follows:

RO

B.1.5.21 CLUSTERAMU_AMPIDR0, Activity Monitors Peripheral Identification Register 0

Provides information to identify an activity monitors component.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFE0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1110 1110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-113: ext_clusteramu_ampidr0 bit assignments

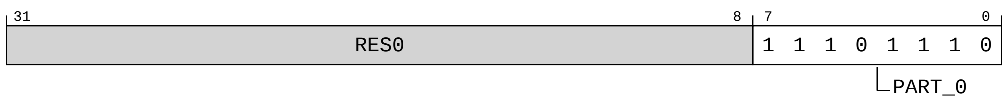


Table B-205: CLUSTERAMU_AMPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number, least significant byte. **`0b11101110`** DSU-C1 DSU Cluster AMU. Bits [7:0] of part number 0x4EE.	`0xEE`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFE0	CLUSTERAMU_AMPIDR0	None

This interface is accessible as follows:

RO

B.1.5.22 CLUSTERAMU_AMPIDR1, Activity Monitors Peripheral Identification Register 1

Provides information to identify an activity monitors component.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-114: ext_clusteramu_ampidr1 bit assignments

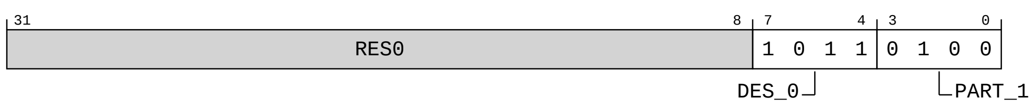


Table B-207: CLUSTERAMU_AMPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	Designer, least significant nibble of JEP106 ID code. For Arm Limited, this field is `0b1011`. **`0b1011`** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	`0b1011`
[3:0]	PART_1	Part number, most significant nibble. **`0b0100`** DSU-C1 DSU Cluster AMU. Bits [11:8] of part number 0x4EE.	`0b0100`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFE4	CLUSTERAMU_AMPIDR1	None

This interface is accessible as follows:

RO

B.1.5.23 CLUSTERAMU_AMPIDR2, Activity Monitors Peripheral Identification Register 2

Provides information to identify an activity monitors component.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFE8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 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

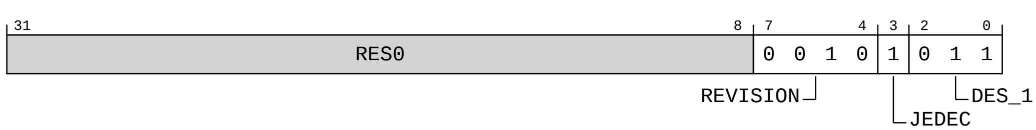


Table B-209: CLUSTERAMU_AMPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVISION	Part major revision. Parts can also use this field to extend Part number to 16-bits. **0b0010** Component major revision 2.	0b0010
[3]	JEDEC	Indicates a JEP106 identity code is used. **0b1**	0b1
[2:0]	DES_1	Designer, most significant bits of JEP106 ID code. For Arm Limited, this field is 0b011. **0b011** Arm Limited. Bits [7:4] of JEP106 identification code 0x3B.	0b011

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFE8	CLUSTERAMU_AMPIDR2	None

This interface is accessible as follows:

RO

B.1.5.24 CLUSTERAMU_AMPIDR3, Activity Monitors Peripheral Identification Register 3

Provides information to identify an activity monitors component.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

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-116: ext_clusteramu_ampidr3 bit assignments

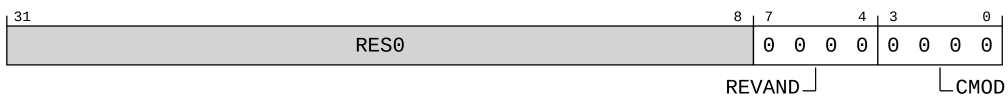


Table B-211: CLUSTERAMU_AMPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Part minor revision. Parts using ext-AMPIDR2.REVISION as an extension to the Part number must use this field as a major revision number. **`0b0000`** Component minor revision 0.	`0b0000`
[3:0]	CMOD	Customer modified. Indicates someone other than the Designer has modified the component. **`0b0000`** The component is not modified from the original design.	`0b0000`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFEC	CLUSTERAMU_AMPIDR3	None

This interface is accessible as follows:

RO

B.1.5.25 CLUSTERAMU_AMCIDR0, Activity Monitors Component Identification Register 0

Provides information to identify an activity monitors component.

For more information, see *About the Component identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

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-117: ext_clusteramu_amcidr0 bit assignments

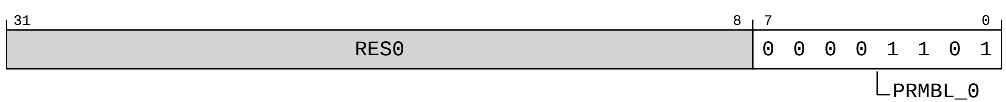


Table B-213: CLUSTERAMU_AMCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	Preamble. **0b00001101**	0x0D

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFF0	CLUSTERAMU_AMCIDR0	None

This interface is accessible as follows:

RO

B.1.5.26 CLUSTERAMU_AMCIDR1, Activity Monitors Component Identification Register 1

Provides information to identify an activity monitors component.

For more information, see *About the Component identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFF4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-118: ext_clusteramu_amcidr1 bit assignments

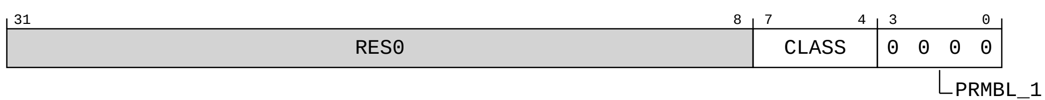


Table B-215: CLUSTERAMU_AMCIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	CLASS	Component class. **0b1001** CoreSight component. Other values are defined by the CoreSight Architecture. This field reads as `0x9`.	`xxxx`
[3:0]	PRMBL_1	Preamble. **0b0000**	`0b0000`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFF4	CLUSTERAMU_AMCIDR1	None

This interface is accessible as follows:

RO

B.1.5.27 CLUSTERAMU_AMCIDR2, Activity Monitors Component Identification Register 2

Provides information to identify an activity monitors component.

For more information, see *About the Component identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

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_amcidr2 bit assignments

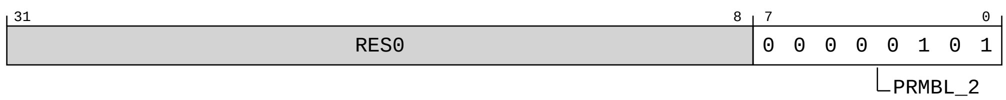


Table B-217: CLUSTERAMU_AMCIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	Preamble. **0b00000101**	0x05

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFF8	CLUSTERAMU_AMCIDR2	None

This interface is accessible as follows:

RO

B.1.5.28 CLUSTERAMU_AMCIDR3, Activity Monitors Component Identification Register 3

Provides information to identify an activity monitors component.

For more information, see *About the Component identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERAMU

Register offset

0xFFC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

> **Note**
>
> Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-120: ext_clusteramu_amcidr3 bit assignments

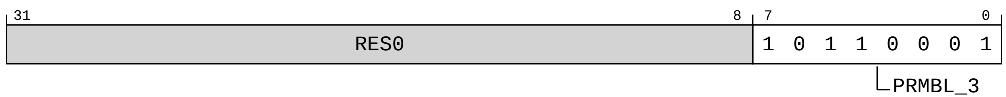


Table B-219: CLUSTERAMU_AMCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Preamble. **`0b10110001`**	`0xB1`

Accessibility

Component	Offset	Instance	Range
CLUSTERAMU	0xFFC	CLUSTERAMU_AMCIDR3	None

This interface is accessible as follows:

RO

B.1.6 External CLUSTERCTL registers summary

The CLUSTERCTL registers are only accessible from memory-mapped accesses on the utility bus. Only the CLUSTERCTL_CLUSTERRECOV register is functional in the C1-DSU. The other registers are present but have no functional effect when written.

The summary table provides an overview of **IMPLEMENTATION DEFINED** memory-mapped CLUSTERCTL registers in the C1-DSU. For more information about a register, click on the register name in the table

---

Note

- Only the CLUSTERCTL_CLUSTERRECOV register is functional in the C1-DSU.
- The CLUSTERCTL 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 CLUSTERCTL registers is `0x050000`.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-221: CLUSTERCTL registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0000	CLUSTERCTL_CLUSTERSLCFR	See individual bit resets.	64-bit	Cluster Split/Lock Configuration Feature Register	Yes
0x0008	CLUSTERCTL_CLUSTERSLCTLR	See individual bit resets.	64-bit	Cluster Split/Lock Configuration Control Register	Yes
0x0010	CLUSTERCTL_CLUSTERSLSTAT	See individual bit resets.	64-bit	Cluster Split/Lock Configuration Status Register	Yes
0x0018	CLUSTERCTL_CLUSTERPPUACTLR	See individual bit resets.	64-bit	Cluster Split/Lock PPU Chicken bit register	Yes
0x0020	CLUSTERCTL_CORESLCFR	See individual bit resets.	64-bit	Core Split/Lock Configuration Feature Register	Yes
0x0030	CLUSTERCTL_CORESLCTLR	See individual bit resets.	64-bit	Core Split/Lock Configuration Control Register	Yes
0x0040	CLUSTERCTL_CORESLSTAT	See individual bit resets.	64-bit	Core Split/Lock Configuration Status Register	Yes
0x0050	CLUSTERCTL_CLUSTERRECOV	See individual bit resets.	64-bit	Cluster Recovery Control Register	Yes
0x0060	CLUSTERCTL_CLUSTERWRITEKEY	See individual bit resets.	64-bit	Cluster Write Key Register	Yes

B.1.6.1 CLUSTERCTL_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

CLUSTERCTL

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    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-121: ext_clusterctl_clusterslcfr bit assignments

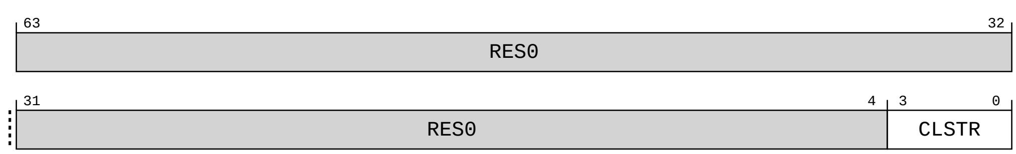


Table B-222: CLUSTERCTL_CLUSTERSLCFR bit descriptions

Bits	Name	Description	Reset
[63:4]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[3:0]	CLSTR	Indicates the Cluster level support for Split/Lock. **0b0001** Cluster only supports LOCKED mode (including HYBRID mode). **0b0100** Cluster only supports SPLIT mode. **0b0101** Cluster supports operation in SPLIT or MIXED mode.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.6.2 CLUSTERCTL_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

CLUSTERCTL

Register offset

0x0008

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-122: ext_clusterctl_clusterslctlr bit assignments

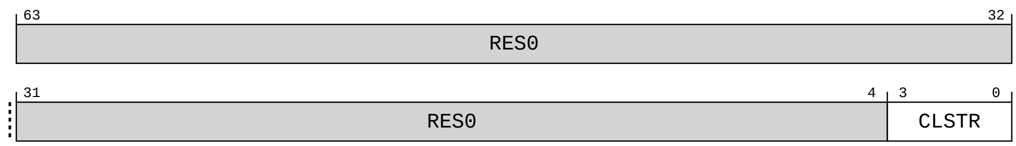


Table B-223: CLUSTERCTL_CLUSTERSLCTLR bit descriptions

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. **`0b0000`** Cluster is to be used in SPLIT mode. **`0b0001`** Cluster is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`

Accessibility

This interface is accessible as follows:

RW

B.1.6.3 CLUSTERCTL_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

CLUSTERCTL

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-123: ext_clusterctl_clusterslstat bit assignments

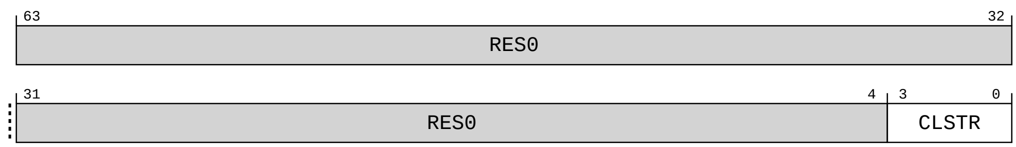


Table B-224: CLUSTERCTL_CLUSTERSLSTAT bit descriptions

Bits	Name	Description	Reset
[63:4]	RES0	Reserved	RES0
[3:0]	CLSTR	Indicates the current Split/Lock status for the Cluster. **0b0000** Cluster is powered off. **0b0001** Cluster is powered on in LOCKED mode (includes HYBRID mode). **0b0100** Cluster is powered on in SPLIT mode.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.1.6.4 CLUSTERCTL_CLUSTERPPUACTLR, Cluster Split/Lock PPU Chicken bit register

Control register for disabling/enabling PPU hierarchical clock gating.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERCTL

Register offset

0x0018

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-124: ext_clusterctl_clusterppuactlr bit assignments

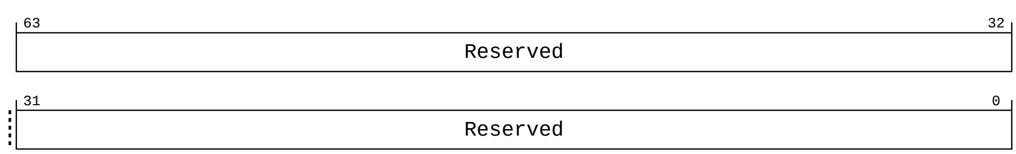


Table B-225: CLUSTERCTL_CLUSTERPPUACTLR bit descriptions

Bits	Name	Description	Reset
[63:0]	Reserved	Reserved for Arm internal use	64{x}

Accessibility

This interface is accessible as follows:

RW

B.1.6.5 CLUSTERCTL_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

CLUSTERCTL

Register offset

0x0020

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-125: ext_clusterctl_coreslcfr bit assignments

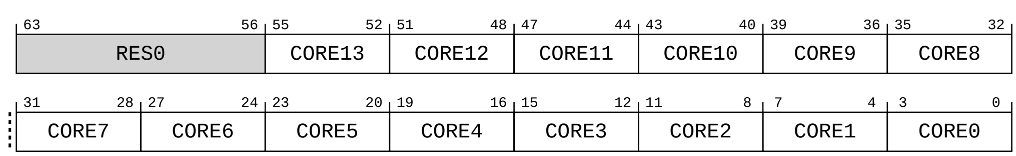


Table B-226: CLUSTERCTL_CORESLCFR bit descriptions

Bits	Name	Description	Reset
[63:56]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[55:52]	CORE13	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[51:48]	CORE12	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[47:44]	CORE11	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx

Bits	Name	Description	Reset
[43:40]	CORE10	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[39:36]	CORE9	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[35:32]	CORE8	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx

Bits	Name	Description	Reset
[31:28]	CORE7	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[27:24]	CORE6	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[23:20]	CORE5	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx

Bits	Name	Description	Reset
[19:16]	CORE4	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[15:12]	CORE3	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[11:8]	CORE2	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx

Bits	Name	Description	Reset
[7:4]	CORE1	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx
[3:0]	CORE0	Indicates core N support for Split/Lock. **0b0000** Core not present. **0b0001** Core supports operation only in LOCKED mode as the primary core. **0b0100** Core supports operation in SPLIT mode only. **0b0101** Core supports both SPLIT and LOCKED mode, as the primary core. **0b0111** Core supports both SPLIT and LOCKED mode, as the redundant core.	xxxx

Accessibility

This interface is accessible as follows:

RO

B.1.6.6 CLUSTERCTL_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

CLUSTERCTL

Register offset

0x0030

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Indicates request settings for core N Split/Lock

Figure B-126: ext_clusterctl_coreslctlr bit assignments

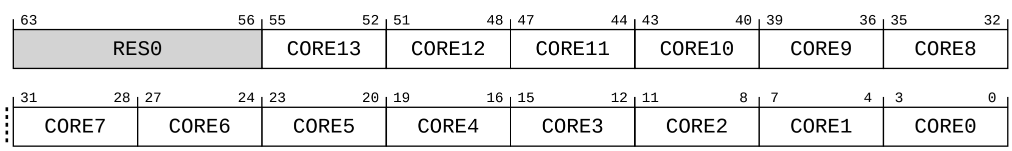


Table B-227: CLUSTERCTL_CORESLCTLR bit descriptions

Bits	Name	Description	Reset
[63:56]	**RES0**	Reserved	**RES0**
[55:52]	CORE13	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`

Bits	Name	Description	Reset
[51:48]	CORE12	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`
[47:44]	CORE11	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`
[43:40]	CORE10	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`
[39:36]	CORE9	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`

Bits	Name	Description	Reset
[35:32]	CORE8	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	xxxx
[31:28]	CORE7	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	xxxx
[27:24]	CORE6	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	xxxx
[23:20]	CORE5	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	xxxx

Bits	Name	Description	Reset
[19:16]	CORE4	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`
[15:12]	CORE3	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`
[11:8]	CORE2	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`
[7:4]	CORE1	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`

Bits	Name	Description	Reset
[3:0]	CORE0	Indicates core N 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 relevant bit of CORESLDEFAULT pin. **`0b0000`** Core is to be used in SPLIT mode. **`0b0001`** Core is to be used in LOCKED mode (includes HYBRID mode).	`xxxx`

Accessibility

This interface is accessible as follows:

RW

B.1.6.7 CLUSTERCTL_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

CLUSTERCTL

Register offset

0x0040

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Indicates the current Split/Lock status for the core.

Figure B-127: ext_clusterctl_coreslstat bit assignments

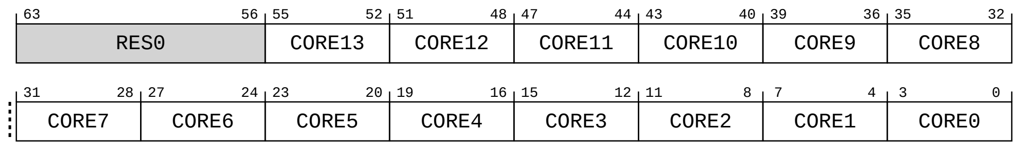


Table B-228: CLUSTERCTL_CORESLSTAT bit descriptions

Bits	Name	Description	Reset
[63:56]	**RES0**	Reserved	**RES0**
[55:52]	CORE13	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[51:48]	CORE12	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[47:44]	CORE11	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx

Bits	Name	Description	Reset
[43:40]	CORE10	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[39:36]	CORE9	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[35:32]	CORE8	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[31:28]	CORE7	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx

Bits	Name	Description	Reset
[27:24]	CORE6	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[23:20]	CORE5	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[19:16]	CORE4	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[15:12]	CORE3	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx

Bits	Name	Description	Reset
[11:8]	CORE2	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[7:4]	CORE1	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx
[3:0]	CORE0	Indicates the current Split/Lock status for core N. **0b0000** Core not present or powered off. **0b0001** Core powered in LOCKED mode as the primary core. **0b0011** Core powered in LOCKED mode as the redundant core. **0b0100** Core powered in SPLIT mode	xxxx

Accessibility

This interface is accessible as follows:

RO

B.1.6.8 CLUSTERCTL_CLUSTERRECOV, Cluster Recovery Control Register

Debug recovery and warm reset request register.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

CLUSTERCTL

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-128: ext_clusterctl_clusterrecov bit assignments

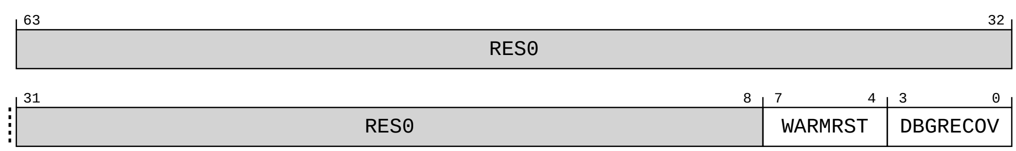


Table B-229: CLUSTERCTL_CLUSTERRECOV bit descriptions

Bits	Name	Description	Reset
[63:8]	RES0	Reserved	RES0
[7:4]	WARMRST	**0b0000** Cores and Cluster in normal operation. **0b0001** Cores and Cluster entering WARM_RST. **0b0011** Cores and Cluster entering WARM_RST, ready to remain there. **0b0100** Cores and Cluster leaving WARM_RST. **0b0111** Cores and Cluster all in WARM_RST (or OFF/OFF_EMU).	0b0000

Bits	Name	Description	Reset
[3:0]	DBGRECOV	**0b0000** Cores and Cluster in normal operation. **0b0001** Cores and Cluster entering DBG_RECOV. **0b0011** Cores and Cluster entering DBG_RECOV, ready to remain there. **0b0100** Cores and Cluster leaving DBG_RECOV. **0b0111** Cores and Cluster all in DBG_RECOV (or OFF/OFF_EMU).	0b0000

Accessibility

This interface is accessible as follows:

RW

B.1.6.9 CLUSTERCTL_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

CLUSTERCTL

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-129: ext_clusterctl_clusterwritekey bit assignments

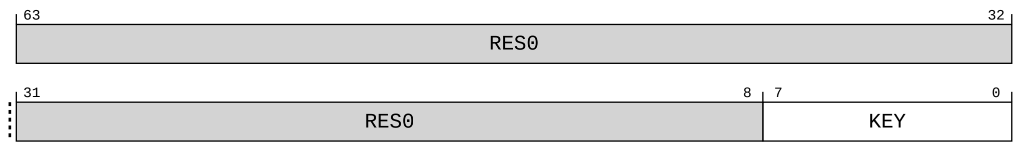


Table B-230: CLUSTERCTL_CLUSTERWRITEKEY bit descriptions

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 and SME2 unit PPU registers summary

The summary table provides an overview of **IMPLEMENTATION DEFINED** memory-mapped ppu registers in the core. For more information about a register, click 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 or in the Arm ARM.

Table B-231: Core Power Policy Unit registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	PPU_PWPR	See individual bit resets.	32-bit	Power Policy Register	Yes
0x004	PPU_PMER	See individual bit resets.	32-bit	Power Mode Emulation Enable Register	Yes
0x008	PPU_PWSR	See individual bit resets.	32-bit	Power Status Register	Yes
0x010	PPU_DISR	See individual bit resets.	32-bit	Device Interface Input Current Status Register	Yes
0x014	PPU_MISR	See individual bit resets.	32-bit	Miscellaneous Input Current Status Register	Yes
0x018	PPU_STSR	See individual bit resets.	32-bit	Stored Status Register	Yes
0x01C	PPU_UNLK	See individual bit resets.	32-bit	Unlock Register	Yes
0x020	PPU_PWCR	See individual bit resets.	32-bit	Power Configuration Register	Yes
0x024	PPU_PTCR	See individual bit resets.	32-bit	Power Mode Transition Register	Yes
0x030	PPU_IMR	See individual bit resets.	32-bit	Interrupt Mask Register	Yes
0x034	PPU_AIMR	See individual bit resets.	32-bit	Additional Interrupt Mask Register	Yes
0x038	PPU_ISR	See individual bit resets.	32-bit	Interrupt Status Register	Yes
0x03C	PPU_AISR	See individual bit resets.	32-bit	Additional Interrupt Status Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x040	PPU_IESR	See individual bit resets.	32-bit	Input Edge Sensitivity Register	Yes
0x044	PPU_OPSR	See individual bit resets.	32-bit	Operating Mode Active Edge Sensitivity Register	Yes
0x050	PPU_FUNRR	See individual bit resets.	32-bit	Functional Retention RAM Configuration Register	Yes
0x054	PPU_FULRR	See individual bit resets.	32-bit	Full Retention RAM Configuration Register	Yes
0x058	PPU_MEMRR	See individual bit resets.	32-bit	Memory Retention RAM Configuration Register	Yes
0x170	PPU_DCDR0	See individual bit resets.	32-bit	Device Control Delay Configuration Register 0	Yes
0x174	PPU_DCDR1	See individual bit resets.	32-bit	Device Control Delay Configuration Register 1	Yes
0xFB0	PPU_IDR0	See individual bit resets.	32-bit	PPU Identification Register 0	Yes
0xFB4	PPU_IDR1	See individual bit resets.	32-bit	PPU Identification Register 1	Yes
0xFC8	PPU_IIDR	See individual bit resets.	32-bit	Implementation Identification Register	Yes
0xFCC	PPU_AIDR	See individual bit resets.	32-bit	Architecture Identification Register	Yes
0xFD0	PPU_PIDR4	See individual bit resets.	32-bit	PPU Peripheral Identification Register 4	Yes
0xFD4	PPU_PIDR5	See individual bit resets.	32-bit	PPU Peripheral Identification Register 5	Yes
0xFD8	PPU_PIDR6	See individual bit resets.	32-bit	PPU Peripheral Identification Register 6	Yes
0xFDC	PPU_PIDR7	See individual bit resets.	32-bit	PPU Peripheral Identification Register 7	Yes
0xFE0	PPU_PIDR0	See individual bit resets.	32-bit	PPU Peripheral Identification Register 0	Yes
0xFE4	PPU_PIDR1	See individual bit resets.	32-bit	PPU Peripheral Identification Register 1	Yes
0xFE8	PPU_PIDR2	See individual bit resets.	32-bit	PPU Peripheral Identification Register 2	Yes
0xFEC	PPU_PIDR3	See individual bit resets.	32-bit	PPU Peripheral Identification Register 3	Yes
0xFF0	PPU_CIDR0	See individual bit resets.	32-bit	PPU Component Identification Register 0	Yes
0xFF4	PPU_CIDR1	See individual bit resets.	32-bit	PPU Component Identification Register 1	Yes
0xFF8	PPU_CIDR2	See individual bit resets.	32-bit	PPU Component Identification Register 2	Yes
0xFFC	PPU_CIDR3	See individual bit resets.	32-bit	PPU Component Identification Register 3	Yes

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-130: ext_ppu_pwpr bit assignments

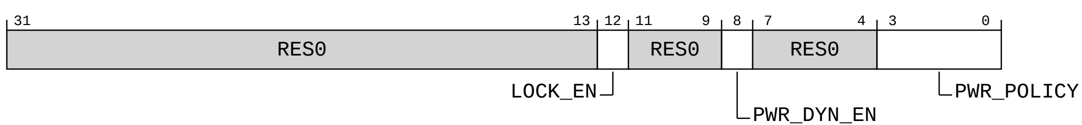


Table B-232: PPU_PWPR bit descriptions

Bits	Name	Description	Reset
[31:13]	**RES0**	Reserved	**RES0**
[12]	LOCK_EN	Lock enable bit for OFF and OFF_EMU power modes **`0b0`** Lock feature disabled. **`0b1`** Lock feature enabled.	`0b0`
[11:9]	**RES0**	Reserved	**RES0**
[8]	PWR_DYN_EN	Power mode dynamic transition enable. **`0b0`** Dynamic transitions disabled for power modes. **`0b1`** Dynamic transitions enabled for power modes, allowing transitions to be initiated by changes on power mode DEVACTIVE inputs.	`0b0`
[7:4]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[3:0]	PWR_POLICY	Power mode policy. When static power mode transitions are enabled, PWR_DYN_EN is set to `0b0`, this is the target power mode for the PPU. When dynamic power mode transitions are enabled, PWR_DYN_EN is set to `0b1`, this is the minimum power mode for the PPU. All other values are reserved. **`0b0000`** OFF. Logic off and RAM off. **`0b0001`** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **`0b0101`** FULL_RET. Full Retention. Logic and RAM in retention. **`0b0111`** FUNC_RET. Functional Retention. Floating-point/Vector logic retained, rest of the core logic and RAM on, core is functional. **`0b1000`** ON. Logic on with RAM on, core is functional. **`0b1001`** WARM_RST. Warm Reset. Warm reset application with logic and RAM on. **`0b1010`** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	`0b0000`

B.1.7.2 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

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-131: ext_ppu_pmer bit assignments

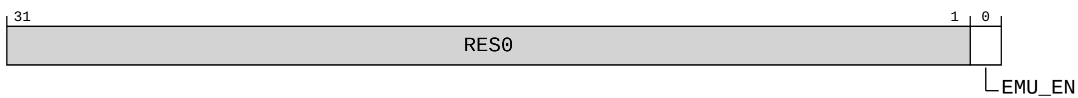


Table B-233: PPU_PMER bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	EMU_EN	Power mode emulation enable. **0b0** Power mode emulation disabled. **0b1** Power mode emulation enabled. Transitions to OFF and MEM_RET instead transition to OFF_EMU and MEM_RET_EMU.	0b0

B.1.7.3 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

```
xxxx xxxx xxxx xxxx xxx0 xxx0 xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-132: ext_ppu_pwsr bit assignments

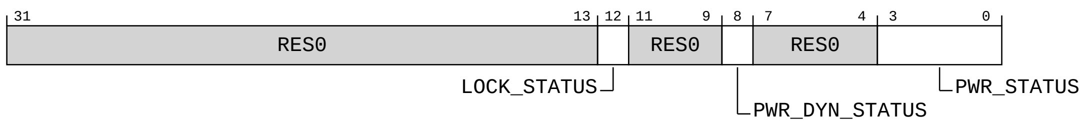


Table B-234: PPU_PWSR bit descriptions

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**
[8]	PWR_DYN_STATUS	Power mode dynamic transition status. There might be a delay in dynamic transitions becoming active or inactive if the PPU is transitioning when PPU_PWPR.DYN_EN is programmed. **`0b0`** Dynamic transitions disabled for power modes. **`0b1`** Dynamic transitions enabled for power modes.	`0b0`
[7:4]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[3:0]	PWR_STATUS	Power mode status. These bits reflect the current power mode of the PPU. All other values are reserved. **0b0000** OFF. Logic off and RAM off. **0b0001** OFF_EMU. Emulated Off. Logic on with RAM on. This mode is used to emulate the functional condition of OFF without removing power. **0b0101** FULL_RET. Full Retention. Logic and RAM in retention. **0b0111** FUNC_RET. Functional Retention. Floating-point/Vector logic retained, rest of the core logic and RAM on, core is functional. **0b1000** ON. Logic on with RAM on, core is functional. **0b1001** WARM_RST. Warm Reset. Warm reset application with logic and RAM on. **0b1010** DBG_RECOV. Debug Recovery Reset. Warm reset application with logic and RAM on.	0b0000

B.1.7.4 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

```
xxxx xxxx xxxx xxxx xxxx x000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-133: ext_ppu_disr bit assignments

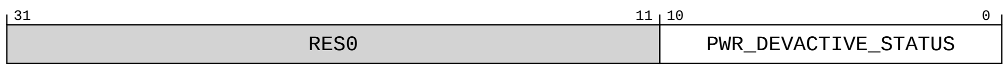


Table B-235: PPU_DISR bit descriptions

Bits	Name	Description	Reset
[31:11]	**RES0**	Reserved	**RES0**
[10:0]	PWR_DEVACTIVE_STATUS	Status of the power mode DEVPACTIVE inputs. **0b00000000000** Minimum mode OFF. **0000000001x** Minimum mode OFF_EMU. **000001xxxxx** Minimum mode FULL_RET. **0001xxxxxxx** Minimum mode FUNC_RET. **001xxxxxxxx** Minimum mode ON. **01xxxxxxxxx** Minimum mode WARM_RST. **1xxxxxxxxxx** Minimum mode DBG_RECOV.	0b00000000000

B.1.7.5 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-134: ext_ppu_misr bit assignments

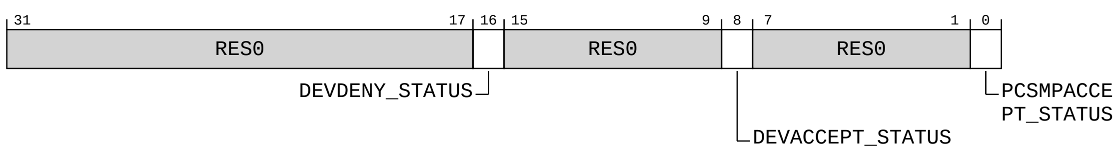


Table B-236: PPU_MISR bit descriptions

Bits	Name	Description	Reset
[31:17]	**RES0**	Reserved	**RES0**
[16]	DEVDENY_STATUS	Status of the device interface DEVPDENY inputs. **0b0** DEVPDENY deasserted. **0b1** DEVPDENY asserted.	0b0
[15:9]	**RES0**	Reserved	**RES0**
[8]	DEVACCEPT_STATUS	Status of the device interface DEVPACCEPT inputs. **0b0** DEVPACCEPT deasserted. **0b1** DEVPACCEPT asserted.	0b0
[7:1]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[0]	PCSMPACCEPT_STATUS	Status of the PCSMPACCEPT inputs. **0b0** PCSMPACCEPT deasserted. **0b1** PCSMPACCEPT asserted.	0b0

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

```
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-135: ext_ppu_stsr bit assignments

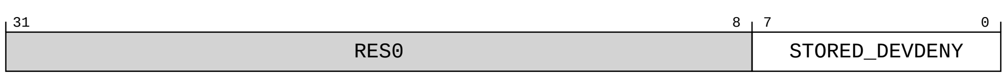


Table B-237: PPU_STSR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	STORED_DEVDENY	Status of the DEVDENY signals from the last device interface Q-Channel transition. This field is reserved. **`0b00000000`** Reserved for P-Channel PPUs.	`8{x}`

B.1.7.7 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

```
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-136: ext_ppu_unlk bit assignments

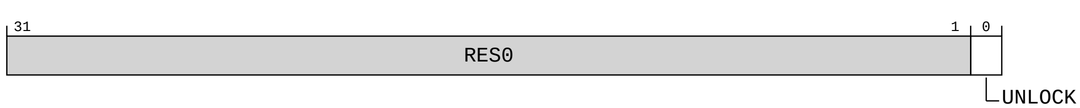


Table B-238: PPU_UNLK bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0
[0]	UNLOCK	When `0b1` is written to this bit the PPU is unlocked from a locked power mode. A read always returns `0b0`.	`x`

B.1.7.8 PPU_PWCR, Power Configuration Register

This register controls enabling and disabling of hardware control inputs to the PPU.

Note

Before software programs the DEVREQEN bits it must configure the PPU for static transitions and ensure the requested power mode has been reached, this means that no further transitions can occur, otherwise behavior is UNPREDICTABLE.

The PWR_DEVACTIVEEN and OP_DEVACTIVEEN fields in this register control the ability of the DEVACTIVE inputs to initiate power mode transitions, but not the ability to generate input edge interrupt events.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

ppu

Register offset

0x020

Access type

RW

Reset value

```
xxxx xxxx 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-137: ext_ppu_pwcr bit assignments

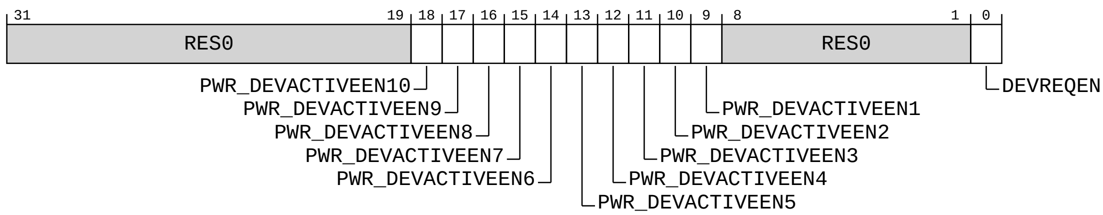


Table B-239: PPU_PWCR bit descriptions

Bits	Name	Description	Reset
[31:19]	RES0	Reserved	RES0
[18]	PWR_DEVACTIVEEN10	Enables the operating mode DEVPACTIVE[10] input. **0b0** DEVPACTIVE[10] input (DBG_RECOV) disabled. **0b1** DEVPACTIVE[10] input (DBG_RECOV) enabled.	0b1
[17]	PWR_DEVACTIVEEN9	Enables the operating mode DEVPACTIVE[9] input. **0b0** DEVPACTIVE[9] input (WARM_RST) disabled. **0b1** DEVPACTIVE[9] input (WARM_RST) enabled.	0b1
[16]	PWR_DEVACTIVEEN8	Enables the operating mode DEVPACTIVE[8] input. **0b0** DEVPACTIVE[8] input (ON) disabled. **0b1** DEVPACTIVE[8] input (ON) enabled.	0b1
[15]	PWR_DEVACTIVEEN7	Enables the operating mode DEVPACTIVE[7] input. **0b0** DEVPACTIVE[7] input (FUNC_RET) disabled. **0b1** DEVPACTIVE[7] input (FUNC_RET) enabled.	0b1
[14]	PWR_DEVACTIVEEN6	Enables the operating mode DEVPACTIVE[6] input. **0b1** DEVPACTIVE[6] input (MEM_OFF) enabled.	0b1
[13]	PWR_DEVACTIVEEN5	Enables the operating mode DEVPACTIVE[5] input. **0b0** DEVPACTIVE[5] input (FULL_RET) disabled. **0b1** DEVPACTIVE[5] input (FULL_RET) enabled.	0b1

Bits	Name	Description	Reset
[12]	PWR_DEVACTIVEEN4	Enables the operating mode DEVPACTIVE[4] input. **0b1** DEVPACTIVE[4] input (LOGIC_RET) enabled.	0b1
[11]	PWR_DEVACTIVEEN3	Enables the operating mode DEVPACTIVE[3] input. **0b1** DEVPACTIVE[3] input (MEM_RET_EMU) enabled.	0b1
[10]	PWR_DEVACTIVEEN2	Enables the operating mode DEVPACTIVE[2] input. **0b1** DEVPACTIVE[2] input (MEM_RET) enabled.	0b1
[9]	PWR_DEVACTIVEEN1	Enables the operating mode DEVPACTIVE[1] input. **0b0** DEVPACTIVE[1] input (OFF_EMU) disabled. **0b1** DEVPACTIVE[1] input (OFF_EMU) enabled.	0b1
[8:1]	RES0	Reserved	RES0
[0]	DEVREQEN	Device interface handshake enable. **0b0** Device interface handshake disabled for transitions. **0b1** Device interface handshake enabled for transitions.	0b1

B.1.7.9 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

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
```

31	27	23	19	15	11	7	3	0

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-138: ext_ppu_ptcr bit assignments

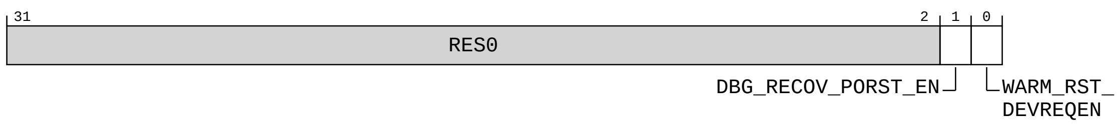


Table B-240: PPU_PTCR bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	DBG_RECOV_PORST_EN	Power-on reset behavior in DBG_RECOV. This bit should not be modified when the PPU is in DBG_RECOV or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **0b0** DEVPORESETn is not asserted when in DBG_RECOV. **0b1** DEVPORESETn is asserted when in DBG_RECOV.	0b0
[0]	WARM_RST_DEVREQEN	Device interface handshake behavior. This bit should not be modified when the PPU is in WARM_RST, or if the PPU is performing a transition, otherwise PPU behavior is **UNPREDICTABLE**. **0b0** The PPU does not perform a device interface handshake when transitioning between ON and WARM_RST. **0b1** The PPU performs a device interface handshake when transitioning between ON and WARM_RST.	0b0

B.1.7.10 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-139: ext_ppu_imr bit assignments

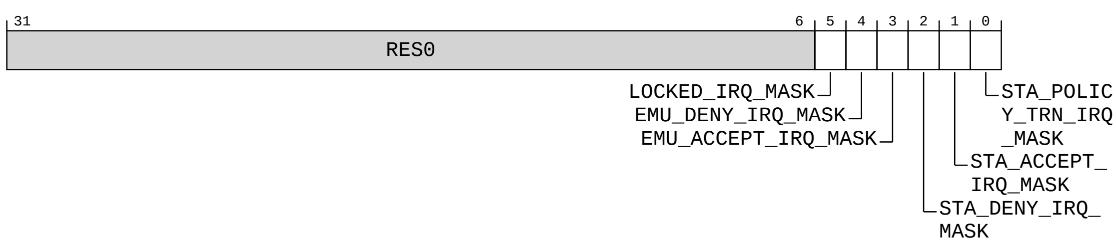


Table B-241: PPU_IMR bit descriptions

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5]	LOCKED_IRQ_MASK	Locked event mask **0b0** Locked event enabled. **0b1** Locked event masked.	0b1

Bits	Name	Description	Reset
[4]	EMU_DENY_IRQ_MASK	Emulation transition denial event mask **0b0** Emulation transition denial event enabled. **0b1** Emulation transition denial event masked.	0b1
[3]	EMU_ACCEPT_IRQ_MASK	Emulation transition acceptance event mask **0b0** Emulation transition acceptance event enabled. **0b1** Emulation transition acceptance event masked.	0b1
[2]	STA_DENY_IRQ_MASK	Static transition denial event mask **0b0** Static transition denial event enabled. **0b1** Static transition denial event masked.	0b0
[1]	STA_ACCEPT_IRQ_MASK	Static transition acceptance event mask **0b0** Static transition acceptance event enabled. **0b1** Static transition acceptance event masked.	0b1
[0]	STA_POLICY_TRN_IRQ_MASK	Static full policy transition completion event mask **0b0** Static full policy transition completion event enabled. **0b1** Static full policy transition completion event masked.	0b0

B.1.7.11 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 x110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-140: ext_ppu_aimr bit assignments

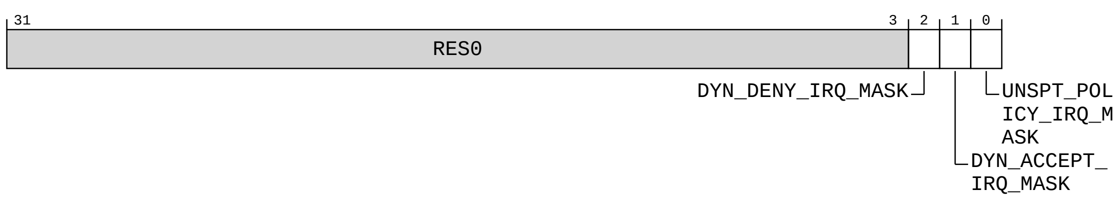


Table B-242: PPU_AIMR bit descriptions

Bits	Name	Description	Reset
[31:3]	RES0	Reserved	RES0
[2]	DYN_DENY_IRQ_MASK	Dynamic transition denial event mask **0b0** Dynamic transition denial event enabled. **0b1** Dynamic transition denial event masked.	0b1
[1]	DYN_ACCEPT_IRQ_MASK	Dynamic transition acceptance event mask **0b0** Dynamic transition acceptance event enabled. **0b1** Dynamic transition acceptance event masked.	0b1

Bits	Name	Description	Reset
[0]	UNSPT_POLICY_IRQ_MASK	Unsupported policy event mask **0b0** Unsupported policy event enabled. **0b1** Unsupported policy event masked.	0b0

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-141: ext_ppu_isr bit assignments

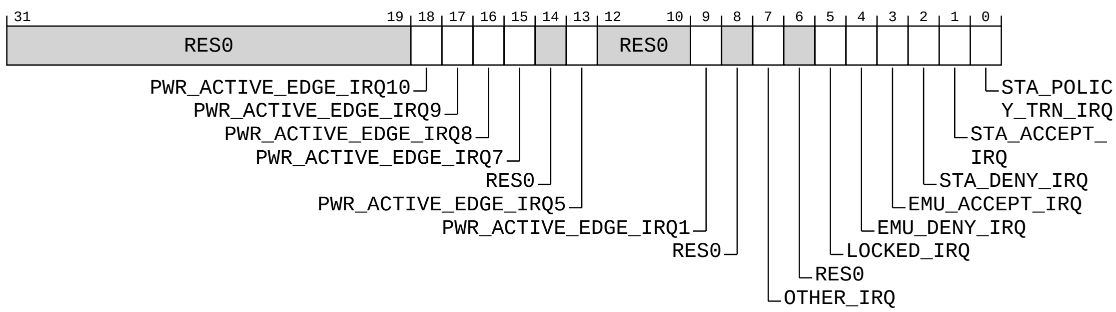


Table B-243: PPU_ISR bit descriptions

Bits	Name	Description	Reset
[31:19]	**RES0**	Reserved	**RES0**
[18]	PWR_ACTIVE_EDGE_IRQ10	Indicates if power mode DEVPACTIVE[10] input caused the input edge event. **0b0** DEVPACTIVE[10] input (DBG_RECOV) did not assert the interrupt output. **0b1** DEVPACTIVE[10] input (DBG_RECOV) asserted the interrupt output.	0b0
[17]	PWR_ACTIVE_EDGE_IRQ9	Indicates if power mode DEVPACTIVE[9] input caused the input edge event. **0b0** DEVPACTIVE[9] input (WARM_RST) did not assert the interrupt output. **0b1** DEVPACTIVE[9] input (WARM_RST) asserted the interrupt output.	0b0
[16]	PWR_ACTIVE_EDGE_IRQ8	Indicates if power mode DEVPACTIVE[8] input caused the input edge event. **0b0** DEVPACTIVE[8] input (ON) did not assert the interrupt output. **0b1** DEVPACTIVE[8] input (ON) asserted the interrupt output.	0b0
[15]	PWR_ACTIVE_EDGE_IRQ7	Indicates if power mode DEVPACTIVE[7] input caused the input edge event. **0b0** DEVPACTIVE[7] input (FUNC_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[7] input (FUNC_RET) asserted the interrupt output.	0b0
[14]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[13]	PWR_ACTIVE_EDGE_IRQ5	Indicates if power mode DEVPACTIVE[5] input caused the input edge event. **0b0** DEVPACTIVE[5] input (FULL_RET) did not assert the interrupt output. **0b1** DEVPACTIVE[5] input (FULL_RET) asserted the interrupt output.	0b0
[12:10]	**RES0**	Reserved	**RES0**
[9]	PWR_ACTIVE_EDGE_IRQ1	Indicates if power mode DEVPACTIVE[1] input caused the input edge event. **0b0** DEVPACTIVE[1] input (OFF_EMU) did not assert the interrupt output. **0b1** DEVPACTIVE[1] input (OFF_EMU) asserted the interrupt output.	0b0
[8]	**RES0**	Reserved	**RES0**
[7]	OTHER_IRQ	Indicates there is an interrupt event pending in the Additional Interrupt Status Register (PPU_AISR). **0b0** No interrupt pending in PPU_AISR. **0b1** Interrupt pending in PPU_AISR.	0b0
[6]	**RES0**	Reserved	**RES0**
[5]	LOCKED_IRQ	Locked event status. **0b0** No locked event. **0b1** A locked event asserted the interrupt output.	0b0
[4]	EMU_DENY_IRQ	Emulated transition denial event status. **0b0** No emulated transition denial event. **0b1** An emulated transition denial event asserted the interrupt output.	0b0
[3]	EMU_ACCEPT_IRQ	Emulated transition acceptance event status. **0b0** No emulated transition acceptance event. **0b1** An emulated transition acceptance event asserted the interrupt output.	0b0
[2]	STA_DENY_IRQ	Static transition denial event status. **0b0** No static transition denial event. **0b1** A static transition denial event asserted the interrupt output.	0b0

Bits	Name	Description	Reset
[1]	STA_ACCEPT_IRQ	Static transition acceptance event status. **0b0** No static transition acceptance event. **0b1** A static transition acceptance event asserted the interrupt output.	0b0
[0]	STA_POLICY_TRN_IRQ	Static full policy transition completion event status. **0b0** No static full policy transition completion event. **0b1** A static full policy transition completion event asserted the interrupt output.	0b0

B.1.7.13 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-142: ext_ppu_aisr bit assignments

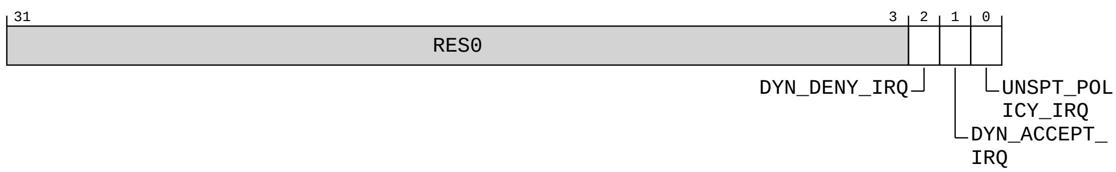


Table B-244: PPU_AISR bit descriptions

Bits	Name	Description	Reset
[31:3]	**RES0**	Reserved	**RES0**
[2]	DYN_DENY_IRQ	Dynamic transition denial event status **0b0** No dynamic transition denial event. **0b1** A dynamic transition denial event asserted the interrupt output.	0b0
[1]	DYN_ACCEPT_IRQ	Dynamic transition acceptance event status **0b0** No dynamic transition acceptance event. **0b1** A dynamic transition acceptance event asserted the interrupt output.	0b0
[0]	UNSPT_POLICY_IRQ	Unsupported policy event status **0b0** No unsupported policy event. **0b1** An unsupported policy event asserted the interrupt output.	0b0

B.1.7.14 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

```
xxxx xxxx xx00 0000 00xx 00xx xxxx 00xx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-143: ext_ppu_iesr bit assignments

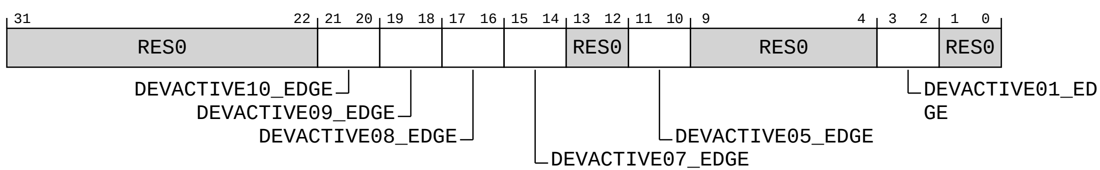


Table B-245: PPU_IESR bit descriptions

Bits	Name	Description	Reset
[31:22]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[21:20]	DEVACTIVE10_EDGE	Configures the transitions on the DEVPACTIVE[10] input (DBG_RECOV) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[19:18]	DEVACTIVE09_EDGE	Configures the transitions on the DEVPACTIVE[9] input (WARM_RST) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[17:16]	DEVACTIVE08_EDGE	Configures the transitions on the DEVPACTIVE[8] input (ON) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[15:14]	DEVACTIVE07_EDGE	Configures the transitions on the DEVPACTIVE[7] input (ON) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[13:12]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[11:10]	DEVACTIVE05_EDGE	Configures the transitions on the DEVPACTIVE[5] input (ON) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[9:4]	RES0	Reserved	RES0
[3:2]	DEVACTIVE01_EDGE	Configures the transitions on the DEVPACTIVE[1] input (OFF_EMU) that generate an Input Edge interrupt event. **0b00** Event masked. **0b01** Rising edge of event generates an interrupt. **0b10** Falling edge of event generates an interrupt. **0b11** Both edges of event generate an interrupt.	0b00
[1:0]	RES0	Reserved	RES0

B.1.7.15 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

```
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-144: ext_ppu_opsr bit assignments

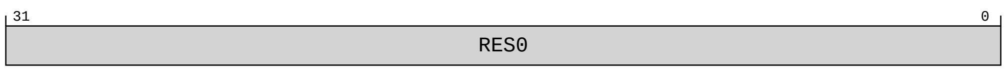


Table B-246: PPU_OPSR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

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

```
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-145: ext_ppu_funrr bit assignments

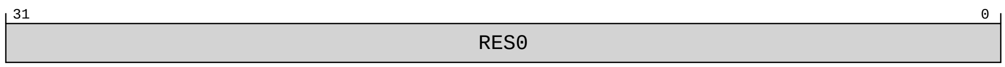


Table B-247: PPU_FUNRR bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

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

```
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-146: ext_ppu_fulrr bit assignments

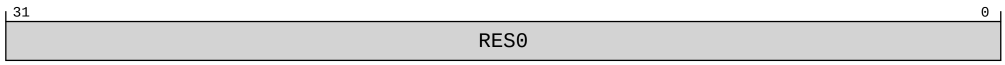


Table B-248: PPU_FULRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

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

```
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-147: ext_ppu_memrr bit assignments

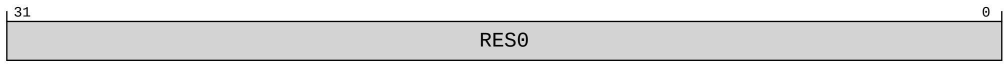


Table B-249: PPU_MEMRR bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

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

```
xxxx xxxx 0000 0000 0000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-148: ext_ppu_dcdr0 bit assignments

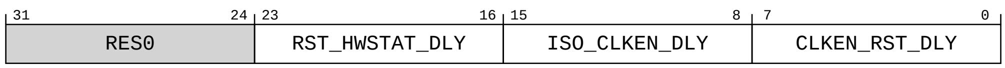


Table B-250: PPU_DCDR0 bit descriptions

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`

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

```
xxxx xxxx xxxx xxxx 0000 0000 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_dcdr1 bit assignments

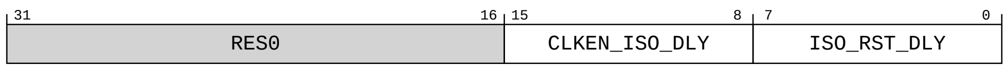


Table B-251: PPU_DCDR1 bit descriptions

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

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-150: ext_ppu_idr0 bit assignments

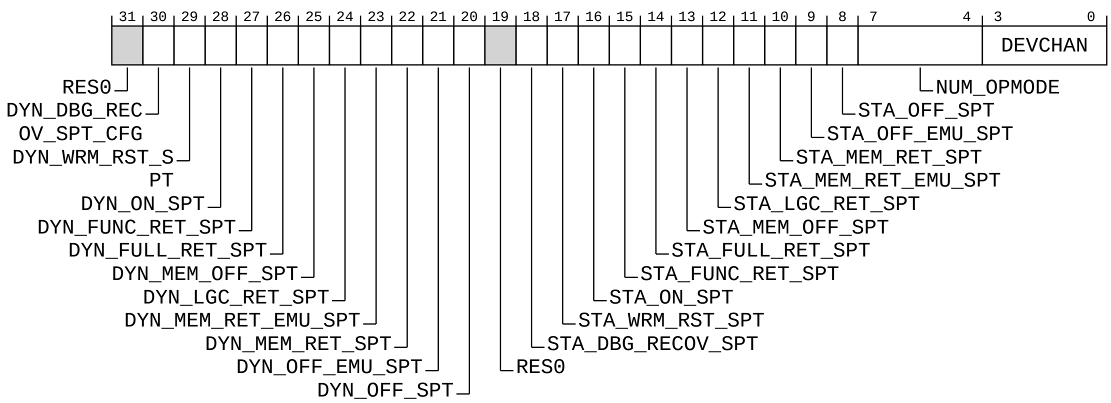


Table B-252: PPU_IDR0 bit descriptions

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
[28]	DYN_ON_SPT	Dynamic ON support. **0b1** Dynamic ON supported.	0b1
[27]	DYN_FUNC_RET_SPT	Dynamic DYN_FUNC_RET_SPT support. **0b1** Dynamic DYN_FUNC_RET_SPT supported.	0b1
[26]	DYN_FULL_RET_SPT	Dynamic DYN_FULL_RET_SPT support. **0b1** Dynamic DYN_FULL_RET_SPT supported.	0b1
[25]	DYN_MEM_OFF_SPT	Dynamic MEM_OFF support. **0b0** Dynamic MEM_OFF not supported.	0b0

Bits	Name	Description	Reset
[24]	DYN_LGC_RET_SPT	Dynamic LOGIC_RET support. **0b0** Dynamic LOGIC_RET not supported.	0b0
[23]	DYN_MEM_RET_EMU_SPT	Dynamic DYN_MEM_RET_EMU_SPT support. **0b0** Dynamic DYN_MEM_RET_EMU_SPT not supported.	0b0
[22]	DYN_MEM_RET_SPT	Dynamic DYN_MEM_RET_SPT support. **0b0** Dynamic DYN_MEM_RET_SPT not supported.	0b0
[21]	DYN_OFF_EMU_SPT	Dynamic OFF_EMU support. **0b1** Dynamic OFF_EMU supported.	0b1
[20]	DYN_OFF_SPT	Dynamic OFF support. **0b1** Dynamic OFF supported.	0b1
[19]	RES0	Reserved	RES0
[18]	STA_DBG_RECOV_SPT	DBG_RECOV support. **0b1** DBG_RECOV supported.	0b1
[17]	STA_WRM_RST_SPT	WARM_RST support. **0b1** WRM_RST supported.	0b1
[16]	STA_ON_SPT	ON support. **0b1** ON supported.	0b1
[15]	STA_FUNC_RET_SPT	FUNC_RET support. **0b1** FUNC_RET supported.	0b1
[14]	STA_FULL_RET_SPT	FULL_RET support. **0b1** FULL_RET supported.	0b1
[13]	STA_MEM_OFF_SPT	MEM_OFF support. **0b0** MEM_OFF not supported.	0b0
[12]	STA_LGC_RET_SPT	LOGIC_RET support. **0b0** LOGIC_RET not supported.	0b0
[11]	STA_MEM_RET_EMU_SPT	MEM_RET_EMU support. **0b0** MEM_RET_EMU not supported.	0b0

Bits	Name	Description	Reset
[10]	STA_MEM_RET_SPT	MEM_RET support. **0b0** MEM_RET not supported.	0b0
[9]	STA_OFF_EMU_SPT	OFF_EMU support. **0b1** OFF_EMU supported.	0b1
[8]	STA_OFF_SPT	OFF support. **0b1** OFF supported.	0b1
[7:4]	NUM_OPMODE	No. of operating modes supported, minus 1. **0b0000** 1 operating mode supported.	0b0000
[3:0]	DEVCHAN	No. of Device Interface Channels. **0b0000** 0 (P-channel PPU).	0b0000

B.1.7.22 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    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-151: ext_ppu_idr1 bit assignments

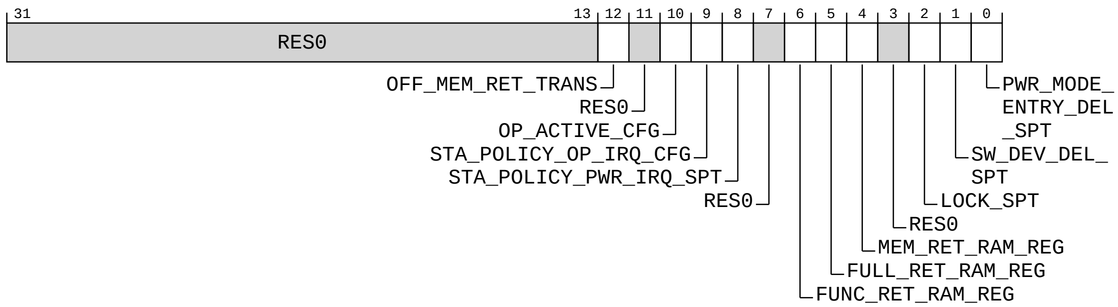


Table B-253: PPU_IDR1 bit descriptions

Bits	Name	Description	Reset
[31:13]	**RES0**	Reserved	**RES0**
[12]	OFF_MEM_RET_TRANS	OFF to MEM_RET direct transition. Indicates if direct transitions from OFF to MEM_RET and from OFF_EMU to MEM_RET_EMU are supported. **0b0** OFF to MEM_RET direct transition not supported.	0b0
[11]	**RES0**	Reserved	**RES0**
[10]	OP_ACTIVE_CFG	Operating mode use model for dynamic transitions. **0b0** Ladder use model.	0b0
[9]	STA_POLICY_OP_IRQ_CFG	Operating policy transition completion event status. **0b0** Operating policy transition completion events not supported.	0b0
[8]	STA_POLICY_PWR_IRQ_SPT	Power policy transition completion event status. **0b0** Power policy transition completion events not supported.	0b0
[7]	**RES0**	Reserved	**RES0**
[6]	FUNC_RET_RAM_REG	Indicates if the PPU_FUNRR register is present or reserved. **0b0** PPU_FUNRR is reserved.	0b0
[5]	FULL_RET_RAM_REG	Indicates if the PPU_FULRR register is present or reserved. **0b0** PPU_FULRR is reserved.	0b0

Bits	Name	Description	Reset
[4]	MEM_RET_RAM_REG	Indicates if the PPU_MEMRR register is present or reserved. **0b0** PPU_MEMRR is reserved.	0b0
[3]	**RES0**	Reserved	**RES0**
[2]	LOCK_SPT	Indicates if the lock and the lock interrupt event are supported. **0b1** Lock and the lock interrupt event are supported.	0b1
[1]	SW_DEV_DEL_SPT	Software device delay control configuration support. **0b1** Software device delay control configuration supported.	0b1
[0]	PWR_MODE_ENTRY_DEL_SPT	Power mode entry delay support. **0b0** Power mode entry delay not supported.	0b0

B.1.7.23 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-152: ext_ppu_iidr bit assignments

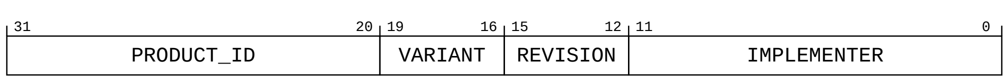


Table B-254: PPU_IIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	PRODUCT_ID	Value identifying the PPU part. **0b000010110110** Power Policy Unit.	0x0B6
[19:16]	VARIANT	Value used to distinguish PPU variants, or major revisions of the PPU. **0b0000** PPU variant 0. **0b0001** PPU variant 1. **0b0010** PPU variant 2. **0b0011** PPU variant 3. **0b0100** PPU variant 4.	0b0010
[15:12]	REVISION	Value used to distinguish minor revisions of the PPU. **0b0000** PPU revision 0. **0b0001** PPU revision 1. **0b0010** PPU revision 2. **0b0011** PPU revision 3. **0b0100** PPU revision 4.	0b0000
[11:0]	IMPLEMENTER	Implementer identification. **0b010000111011** Arm Limited.	0x43B

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

```
xxxx xxxx xxxx xxxx xxxx xxxx 0001 0010
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-153: ext_ppu_aidr bit assignments

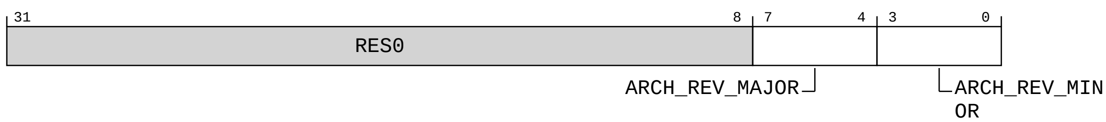


Table B-255: PPU_AIDR bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	ARCH_REV_MAJOR	PPU architecture major revision. **0b0001** PPU architecture major revision 1.	0b0001
[3:0]	ARCH_REV_MINOR	PPU architecture minor revision. **0b0010** PPU architecture minor revision 2.	0b0010

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

```
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-154: ext_ppu_pidr4 bit assignments

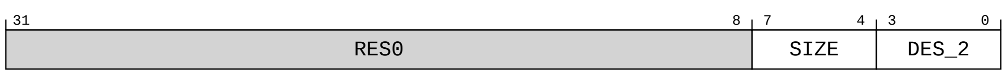


Table B-256: PPU_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	xxxx
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	xxxx

B.1.7.26 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

```
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-155: ext_ppu_pidr5 bit assignments

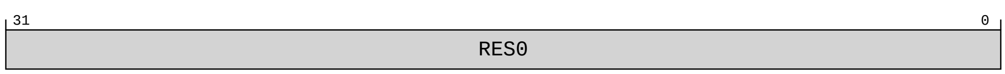


Table B-257: PPU_PIDR5 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

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

```
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-156: ext_ppu_pidr6 bit assignments

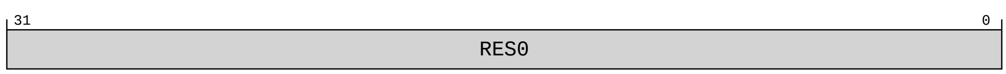


Table B-258: PPU_PIDR6 bit descriptions

Bits	Name	Description	Reset
[31:0]	RES0	Reserved	RES0

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

```
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-157: ext_ppu_pidr7 bit assignments

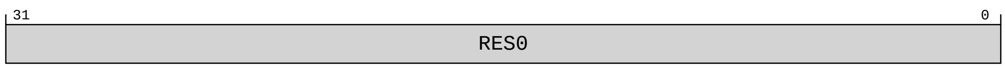


Table B-259: PPU_PIDR7 bit descriptions

Bits	Name	Description	Reset
[31:0]	**RES0**	Reserved	**RES0**

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

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0110
|    |    |    |    |    |    |    |  |
```

```
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-158: ext_ppu_pidr0 bit assignments

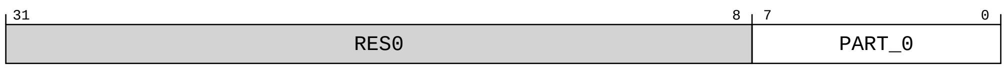


Table B-260: PPU_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. **0b10110110** Core Power Policy Unit. Bits [7:0] of part number 0x0B6.	0xB6

B.1.7.30 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

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0000
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-159: ext_ppu_pidr1 bit assignments

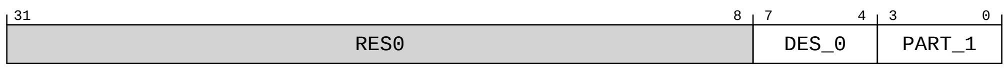


Table B-261: PPU_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0000** Core Power Policy Unit. Bits [11:8] of part number 0x0B6.	0b0000

B.1.7.31 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

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 1011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-160: ext_ppu_pidr2 bit assignments

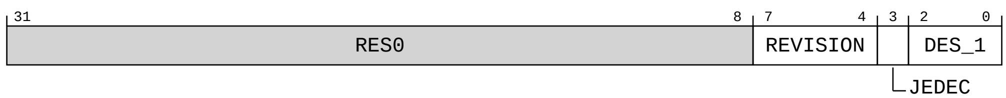


Table B-262: PPU_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. **`0b0000`** Component major revision 0. **`0b0001`** Component major revision 1. **`0b0010`** Component major revision 2. **`0b0011`** Component major revision 3. **`0b0100`** Component major revision 4.	`0b0010`
[3]	JEDEC	JEDEC assignee. **`0b1`** JEDEC-assignee values is used.	`0b1`
[2:0]	DES_1	JEP106 identification code bits [6:4]. **`0b011`** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	`0b011`

B.1.7.32 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
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-161: ext_ppu_pidr3 bit assignments

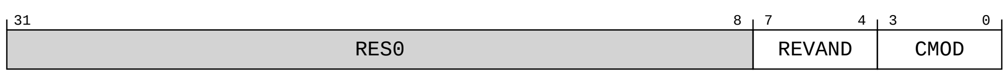


Table B-263: PPU_PIDR3 bit descriptions

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

Bits	Name	Description	Reset
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

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

```
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-162: ext_ppu_cidr0 bit assignments

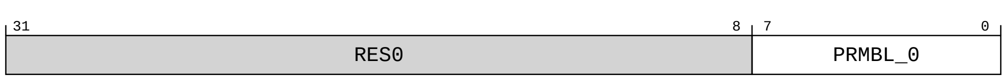


Table B-264: PPU_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

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

```
xxxx xxxx xxxx xxxx xxxx xxxx 1111 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-163: ext_ppu_cidr1 bit assignments

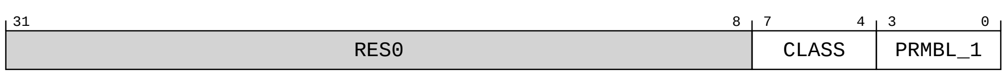


Table B-265: PPU_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	CLASS	CoreSight component class. **0b1111** CoreLink component.	0b1111
[3:0]	PRMBL_1	CoreSight component identification preamble. **0b0000** CoreSight component identification preamble.	0b0000

B.1.7.35 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

```
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-164: ext_ppu_cidr2 bit assignments

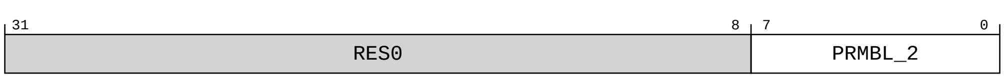


Table B-266: PPU_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	CoreSight component identification preamble. **`0b00000101`** CoreSight component identification preamble.	`0x05`

B.1.7.36 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

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-165: ext_ppu_cidr3 bit assignments

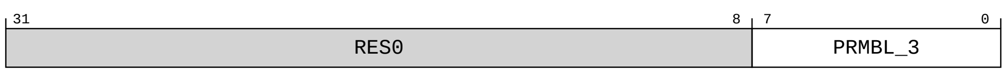


Table B-267: PPU_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **`0b10110001`** CoreSight component identification preamble.	`0xB1`

B.2 Registers accessed over the Debug APB bus

This section contains the descriptions for all the external registers in the C1-DSU accessed over the Debug APB bus.

B.2.1 External cluster, core, and SME2 unit CTI registers summary

The cluster Cross Trigger Interface (CTI) registers, core CTI registers, and SME2 unit CTI registers are only accessible using memory-mapped accesses over the Debug APB interface.

The summary table provides an overview of all the cluster, core, and SME2 unit CTI registers. For more information about a register, click on the register name in the table.

Note

- Registers that differ in descriptions and values, between the cluster, core, and SME2 unit, are indicated in the Identical core CTI and SME2 unit CTI column. These registers are the CTIPIDR0-4 registers, and the CTIDEVAFF0-1 registers.
- The CTI registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the C1-DSU is configured for Direct connect all these registers are present.
- If the C1-DSU is enabled for Realm Management Extension (RME) all these registers are present.
- The cluster CTI part number is `0x4F0`.
- The core and SME2 unit CTI part numbers are `0x4ED` and the CTIDEVAFF0-1 registers can be used to identify which core or SME2 unit the CTI is associated with.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-268: CTI registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core and SME2 unit CTI
0x000	CTICONTROL	See individual bit resets.	32-bit	CTI Control register	Yes, for core and SME2 unit CTI only	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core and SME2 unit CTI
0x010	CTIINTACK	See individual bit resets.	32-bit	CTI Output Trigger Acknowledge register	Yes, for core and SME2 unit CTI only	Yes
0x014	CTIAPPSET	See individual bit resets.	32-bit	CTI Application Trigger Set register	Yes, for core and SME2 unit CTI only	Yes
0x018	CTIAPPCLEAR	See individual bit resets.	32-bit	CTI Application Trigger Clear register	Yes, for core and SME2 unit CTI only	Yes
0x01C	CTIAPPPULSE	See individual bit resets.	32-bit	CTI Application Pulse register	Yes, for core and SME2 unit CTI only	Yes
0x20	CTIINEN0	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x24	CTIINEN1	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x28	CTIINEN2	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x2C	CTIINEN3	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x30	CTIINEN4	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x34	CTIINEN5	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x38	CTIINEN6	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x3C	CTIINEN7	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x40	CTIINEN8	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x44	CTIINEN9	See individual bit resets.	32-bit	CTI Input Trigger to Output Channel Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xA0	CTIOUTEN0	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xA4	CTIOUTEN1	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xA8	CTIOUTEN2	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xAC	CTIOUTEN3	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xB0	CTIOUTEN4	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xB4	CTIOUTEN5	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xB8	CTIOUTEN6	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xBC	CTIOUTEN7	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0xC0	CTIOUTEN8	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core and SME2 unit CTI
0xC4	CTIOUTEN9	See individual bit resets.	32-bit	CTI Input Channel to Output Trigger Enable registers	Yes, for core and SME2 unit CTI only	Yes
0x130	CTITRIGINSTATUS	See individual bit resets.	32-bit	CTI Trigger In Status register	Yes, for core and SME2 unit CTI only	Yes
0x134	CTITRIGOUTSTATUS	See individual bit resets.	32-bit	CTI Trigger Out Status register	Yes, for core and SME2 unit CTI only	Yes
0x138	CTICHINSTATUS	See individual bit resets.	32-bit	CTI Channel In Status register	Yes, for core and SME2 unit CTI only	Yes
0x13C	CTICHOUTSTATUS	See individual bit resets.	32-bit	CTI Channel Out Status register	Yes, for core and SME2 unit CTI only	Yes
0x140	CTIGATE	See individual bit resets.	32-bit	CTI Channel Gate Enable register	Yes, for core and SME2 unit CTI only	Yes
0x150	CTIDEVCTL	See individual bit resets.	32-bit	CTI Device Control register	Yes, for core and SME2 unit CTI only	Yes
0xFA0	CTICLAIMSET	See individual bit resets.	32-bit	CTI CLAIM Tag Set register	Yes, for core and SME2 unit CTI only	Yes
0xFA4	CTICLAIMCLR	See individual bit resets.	32-bit	CTI CLAIM Tag Clear register	Yes, for core and SME2 unit CTI only	Yes
0xFA8	CTIDEVAFF0	See individual bit resets.	32-bit	CTI Device Affinity register 0	Yes, for core and SME2 unit CTI only	No, see individual register
0xFAC	CTIDEVAFF1	See individual bit resets.	32-bit	CTI Device Affinity register 1	Yes, for core and SME2 unit CTI only	No, see individual register
0xFB8	CTIAUTHSTATUS	See individual bit resets.	32-bit	CTI Authentication Status register	Yes, for core and SME2 unit CTI only	Yes
0xFBC	CTIDEVARCH	See individual bit resets.	32-bit	CTI Device Architecture register	Yes, for core and SME2 unit CTI only	Yes
0xFC0	CTIDEVID2	See individual bit resets.	32-bit	CTI Device ID register 2	Yes, for core and SME2 unit CTI only	Yes
0xFC4	CTIDEVID1	See individual bit resets.	32-bit	CTI Device ID register 1	Yes, for core and SME2 unit CTI only	Yes
0xFC8	CTIDEVID	See individual bit resets.	32-bit	CTI Device ID register 0	Yes, for core and SME2 unit CTI only	Yes
0xFCC	CTIDEVTYPE	See individual bit resets.	32-bit	CTI Device Type register	Yes, for core and SME2 unit CTI only	Yes
0xFD0	CTIPIDR4	See individual bit resets.	32-bit	CTI Peripheral Identification Register 4	Yes, for core and SME2 unit CTI only	Yes
0xFE0	CTIPIDR0	See individual bit resets.	32-bit	CTI Peripheral Identification Register 0	Yes, for core and SME2 unit CTI only	No, see individual register
0xFE4	CTIPIDR1	See individual bit resets.	32-bit	CTI Peripheral Identification Register 1	Yes, for core and SME2 unit CTI only	Yes
0xFE8	CTIPIDR2	See individual bit resets.	32-bit	CTI Peripheral Identification Register 2	Yes, for core and SME2 unit CTI only	Yes
0xFEC	CTIPIDR3	See individual bit resets.	32-bit	CTI Peripheral Identification Register 3	Yes, for core and SME2 unit CTI only	Yes
0xFF0	CTICIDR0	See individual bit resets.	32-bit	CTI Component Identification Register 0	Yes, for core and SME2 unit CTI only	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect	Identical core and SME2 unit CTI
0xFF4	CTICIDR1	See individual bit resets.	32-bit	CTI Component Identification Register 1	Yes, for core and SME2 unit CTI only	Yes
0xFF8	CTICIDR2	See individual bit resets.	32-bit	CTI Component Identification Register 2	Yes, for core and SME2 unit CTI only	Yes
0xFFC	CTICIDR3	See individual bit resets.	32-bit	CTI Component Identification Register 3	Yes, for core and SME2 unit CTI only	Yes

B.2.1.1 CTICONTROL, CTI Control register

Controls whether the CTI is enabled.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0x000

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-166: ext_cticontrol bit assignments

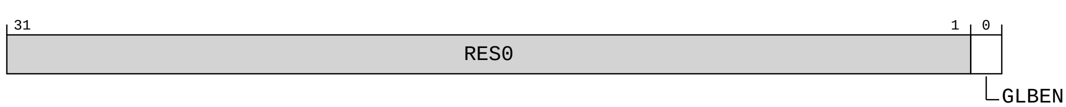


Table B-269: CTICONTROL bit descriptions

Bits	Name	Description	Reset
[31:1]	RES0	Reserved	RES0
[0]	GLBEN	Enables or disables the CTI mapping functions. Possible values of this field are: **0b0** CTI mapping functions and application trigger disabled. **0b1** CTI mapping functions and application trigger enabled. When GLBEN is 0, the input channel to output trigger, input trigger to output channel, and application trigger functions are disabled and do not signal new events on either output triggers or output channels. If a previously asserted output trigger has not been acknowledged, it is CONSTRAINED UNPREDICTABLE which of the following occurs: • The output trigger remains asserted after the mapping functions are disabled. • The output trigger is deasserted after the mapping functions are disabled. All output triggers are disabled by CTI reset. If the ECT supports multicycle channel events any existing output channel events will be terminated.	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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-167: ext_ctiintack bit assignments

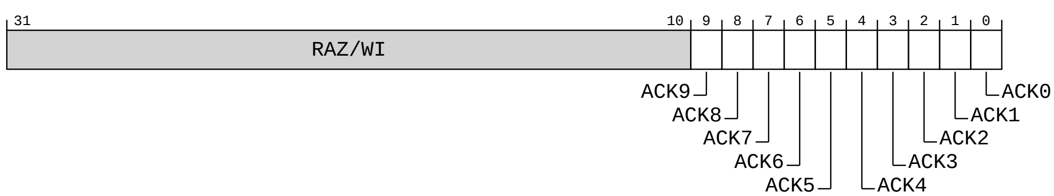


Table B-271: CTIINTACK bit descriptions

Bits	Name	Description	Reset
[31:10]	RAZ/WI	Reserved	RAZ/WI

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>Reset</th></tr>
<tr><td>[9]</td><td>ACK9</td><td>Acknowledge for output trigger <n>.

Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field.

If any of the following is true, writes to ACK<n> are ignored:
• n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers.
• Output trigger n is not active.
• The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active.

Otherwise, if any of the following are true, ACK<n> is RES0:
• Output trigger n is not implemented.
• Output trigger n is not connected.
• Output trigger n is self-acknowledging and does not require software acknowledge.

Otherwise, the behavior on writes to ACK<n> is as follows:
0b0
No effect
0b1
Deactivate the trigger.</td><td>x</td></tr>
<tr><td>[8]</td><td>ACK8</td><td>Acknowledge for output trigger <n>.

Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field.

If any of the following is true, writes to ACK<n> are ignored:
• n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers.
• Output trigger n is not active.
• The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active.

Otherwise, if any of the following are true, ACK<n> is RES0:
• Output trigger n is not implemented.
• Output trigger n is not connected.
• Output trigger n is self-acknowledging and does not require software acknowledge.

Otherwise, the behavior on writes to ACK<n> is as follows:
0b0
No effect
0b1
Deactivate the trigger.</td><td>x</td></tr>
</table>

Bits	Name	Description	Reset
[7]	ACK7	Acknowledge for output trigger <n>. Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. Otherwise, if any of the following are true, ACK<n> is **RES0**: • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **`0b0`** No effect **`0b1`** Deactivate the trigger.	`x`
[6]	ACK6	Acknowledge for output trigger <n>. Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. Otherwise, if any of the following are true, ACK<n> is **RES0**: • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **`0b0`** No effect **`0b1`** Deactivate the trigger.	`x`

Bits	Name	Description	Reset
[5]	ACK5	Acknowledge for output trigger <n>. Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. Otherwise, if any of the following are true, ACK<n> is **RES0**: • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **`0b0`** No effect **`0b1`** Deactivate the trigger.	`x`
[4]	ACK4	Acknowledge for output trigger <n>. Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. Otherwise, if any of the following are true, ACK<n> is **RES0**: • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **`0b0`** No effect **`0b1`** Deactivate the trigger.	`x`

Bits	Name	Description	Reset
[3]	ACK3	Acknowledge for output trigger <n>. Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. Otherwise, if any of the following are true, ACK<n> is **RES0**: • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **`0b0`** No effect **`0b1`** Deactivate the trigger.	`x`
[2]	ACK2	Acknowledge for output trigger <n>. Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. If any of the following is true, writes to ACK<n> are ignored: • n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers. • Output trigger n is not active. • The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active. Otherwise, if any of the following are true, ACK<n> is **RES0**: • Output trigger n is not implemented. • Output trigger n is not connected. • Output trigger n is self-acknowledging and does not require software acknowledge. Otherwise, the behavior on writes to ACK<n> is as follows: **`0b0`** No effect **`0b1`** Deactivate the trigger.	`x`

Bits	Name	Description	Reset

[1]	ACK1	Acknowledge for output trigger <n>.
		
		Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field.
		
		If any of the following is true, writes to ACK<n> are ignored:
		• n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers.
		• Output trigger n is not active.
		• The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active.
		
		Otherwise, if any of the following are true, ACK<n> is **RES0**:
		• Output trigger n is not implemented.
		• Output trigger n is not connected.
		• Output trigger n is self-acknowledging and does not require software acknowledge.
		
		Otherwise, the behavior on writes to ACK<n> is as follows:
		`0b0`
		No effect
		`0b1`
		Deactivate the trigger.
			`x`

[0]	ACK0	Acknowledge for output trigger <n>.
		
		Bits [31:N] are RAZ/WI. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field.
		
		If any of the following is true, writes to ACK<n> are ignored:
		• n >= ext-CTIDEVID.NUMTRIG, the number of implemented triggers.
		• Output trigger n is not active.
		• The channel mapping function output, as controlled by ext-CTIOUTEN<n>, is still active.
		
		Otherwise, if any of the following are true, ACK<n> is **RES0**:
		• Output trigger n is not implemented.
		• Output trigger n is not connected.
		• Output trigger n is self-acknowledging and does not require software acknowledge.
		
		Otherwise, the behavior on writes to ACK<n> is as follows:
		`0b0`
		No effect
		`0b1`
		Deactivate the trigger.
			`x`

Accessibility

A debugger must read ext-CTITRIGOUTSTATUS to confirm that the output trigger has been acknowledged before generating any event that must be ordered after the write to CTIINTACK, such as a write to CTIAPPPULSE to activate another trigger.

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 the application triggers.

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 xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-168: ext_ctiappset bit assignments

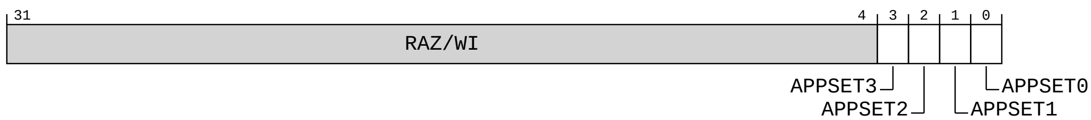


Table B-273: CTIAPPSET bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	APPSET3	Application trigger <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Reading this means the application trigger is inactive. Writing this has no effect.
		
		`0b1`
		Reading this means the application trigger is active. Writing this sets the corresponding application trigger to 1 and generates a channel event.
		
		If the ECT does not support multicycle channel events, use of CTIAPPSET is deprecated and the debugger must only use ext-CTIAPPPULSE.
			`x`

[2]	APPSET2	Application trigger <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Reading this means the application trigger is inactive. Writing this has no effect.
		
		`0b1`
		Reading this means the application trigger is active. Writing this sets the corresponding application trigger to 1 and generates a channel event.
		
		If the ECT does not support multicycle channel events, use of CTIAPPSET is deprecated and the debugger must only use ext-CTIAPPPULSE.
			`x`

[1]	APPSET1	Application trigger <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Reading this means the application trigger is inactive. Writing this has no effect.
		
		`0b1`
		Reading this means the application trigger is active. Writing this sets the corresponding application trigger to 1 and generates a channel event.
		
		If the ECT does not support multicycle channel events, use of CTIAPPSET is deprecated and the debugger must only use ext-CTIAPPPULSE.
			`x`

Bits	Name	Description	Reset
[0]	APPSET0	Application trigger <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Reading this means the application trigger is inactive. Writing this has no effect. **`0b1`** Reading this means the application trigger is active. Writing this sets the corresponding application trigger to 1 and generates a channel event. If the ECT does not support multicycle channel events, use of CTIAPPSET is deprecated and the debugger must only use ext-CTIAPPPULSE.	`x`

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 the application triggers.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-169: ext_ctiappclear bit assignments

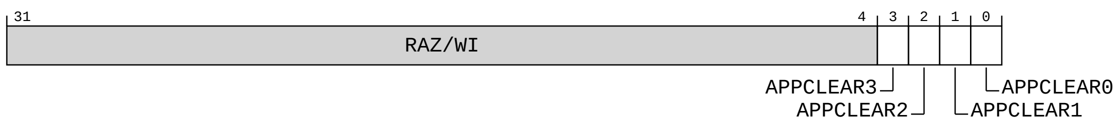


Table B-275: CTIAPPCLEAR bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**

[3]	APPCLEAR3	Application trigger <x> disable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Writing to this bit has the following effect:
		`0b0`
		No effect.
		`0b1`
		Clear corresponding application trigger to 0 and clear the corresponding channel event.
		
		If the ECT does not support multicycle channel events, use of CTIAPPCLEAR is deprecated and the debugger must only use ext-CTIAPPPULSE.
			x

[2]	APPCLEAR2	Application trigger <x> disable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Writing to this bit has the following effect:
		`0b0`
		No effect.
		`0b1`
		Clear corresponding application trigger to 0 and clear the corresponding channel event.
		
		If the ECT does not support multicycle channel events, use of CTIAPPCLEAR is deprecated and the debugger must only use ext-CTIAPPPULSE.
			x

Bits	Name	Description	Reset
[1]	APPCLEAR1	Application trigger <x> disable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Writing to this bit has the following effect: **`0b0`** No effect. **`0b1`** Clear corresponding application trigger to 0 and clear the corresponding channel event. If the ECT does not support multicycle channel events, use of CTIAPPCLEAR is deprecated and the debugger must only use ext-CTIAPPPULSE.	`x`
[0]	APPCLEAR0	Application trigger <x> disable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Writing to this bit has the following effect: **`0b0`** No effect. **`0b1`** Clear corresponding application trigger to 0 and clear the corresponding channel event. If the ECT does not support multicycle channel events, use of CTIAPPCLEAR is deprecated and the debugger must only use ext-CTIAPPPULSE.	`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-170: ext_ctiapppulse bit assignments

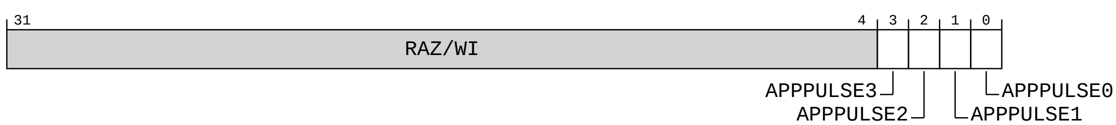


Table B-277: CTIAPPPULSE bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	APPPULSE3	Generate event pulse on ECT channel <x>. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Writing to this bit has the following effect: **`0b0`** No effect. **`0b1`** Channel <x> event pulse generated. • The CTIAPPPULSE operation does not affect the state of the application trigger. If the channel is active, either because of an earlier event or from the application trigger, then the value written to CTIAPPPULSE might have no effect. • Multiple pulse events that occur close together might be merged into a single pulse event.	x
[2]	APPPULSE2	Generate event pulse on ECT channel <x>. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Writing to this bit has the following effect: **`0b0`** No effect. **`0b1`** Channel <x> event pulse generated. • The CTIAPPPULSE operation does not affect the state of the application trigger. If the channel is active, either because of an earlier event or from the application trigger, then the value written to CTIAPPPULSE might have no effect. • Multiple pulse events that occur close together might be merged into a single pulse event.	x
[1]	APPPULSE1	Generate event pulse on ECT channel <x>. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Writing to this bit has the following effect: **`0b0`** No effect. **`0b1`** Channel <x> event pulse generated. • The CTIAPPPULSE operation does not affect the state of the application trigger. If the channel is active, either because of an earlier event or from the application trigger, then the value written to CTIAPPPULSE might have no effect. • Multiple pulse events that occur close together might be merged into a single pulse event.	x

Bits	Name	Description	Reset
[0]	APPPULSE0	Generate event pulse on ECT channel <x>. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Writing to this bit has the following effect: **`0b0`** No effect. **`0b1`** Channel <x> event pulse generated. • The CTIAPPPULSE operation does not affect the state of the application trigger. If the channel is active, either because of an earlier event or from the application trigger, then the value written to CTIAPPPULSE might have no effect. • Multiple pulse events that occur close together might be merged into a single pulse event.	`x`

Accessibility

It is CONSTRAINED UNPREDICTABLE whether a write to CTIAPPPULSE generates an event on a channel if CTICONTROL.GLBEN is 0.

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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-171: ext_ctiinen0 bit assignments

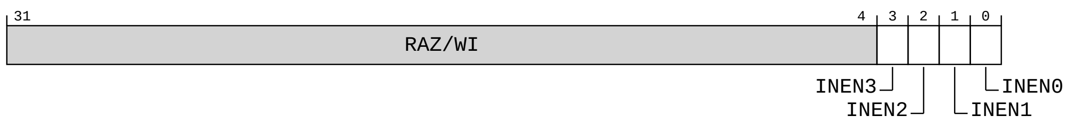


Table B-279: CTIINEN0 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x24

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-172: ext_ctiinen1 bit assignments

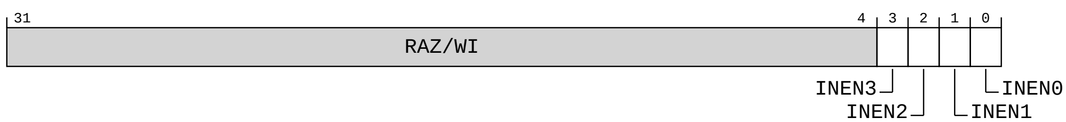


Table B-281: CTIINEN1 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	INEN3	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

[2]	INEN2	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

Bits	Name	Description	Reset

[1]	INEN1	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

[0]	INEN0	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x28

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-173: ext_ctiinen2 bit assignments

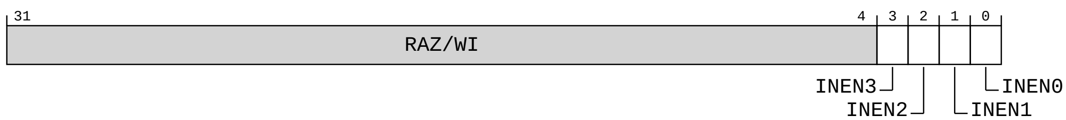


Table B-283: CTIINEN2 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x2C

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-174: ext_ctiinen3 bit assignments

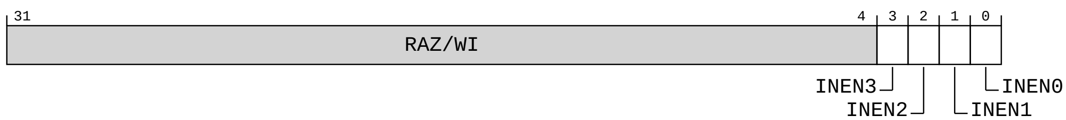


Table B-285: CTIINEN3 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x30

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-175: ext_ctiinen4 bit assignments

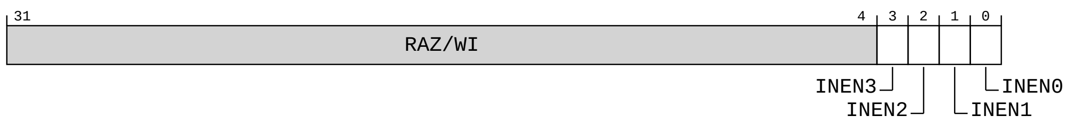


Table B-287: CTIINEN4 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/ WI	Reserved	RAZ/ WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-176: ext_ctiinen5 bit assignments

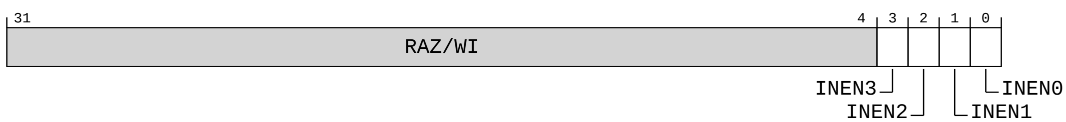


Table B-289: CTIINEN5 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x38

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-177: ext_ctiinen6 bit assignments

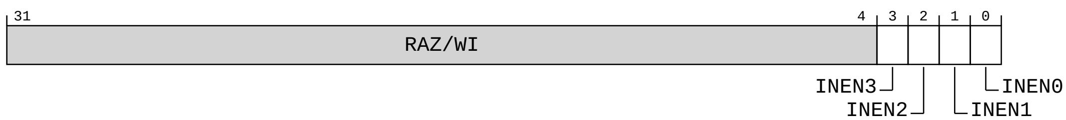


Table B-291: CTIINEN6 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	INEN3	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

[2]	INEN2	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x3C

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-178: ext_ctiinen7 bit assignments

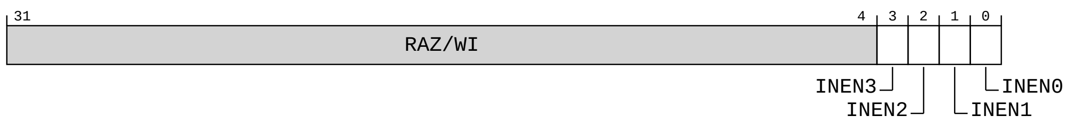


Table B-293: CTIINEN7 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	INEN3	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

[2]	INEN2	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

Bits	Name	Description	Reset

[1]	INEN1	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

[0]	INEN0	Input trigger <n> to output channel <x> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		`0b0`
		Input trigger <n> will not generate an event on output channel <x>.
		`0b1`
		Input trigger <n> will generate an event on output channel <x>.
			x

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x40

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-179: ext_ctiinen8 bit assignments

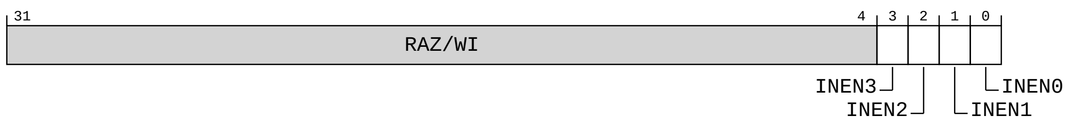


Table B-295: CTIINEN8 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input trigger <n> will not generate an event on output channel <x>. **`0b1`** Input trigger <n> will generate an event on output channel <x>.	`x`

Accessibility

Component	Offset	Instance	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

If input trigger n is not implemented or not connected, CTIINEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0x44

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-180: ext_ctiinen9 bit assignments

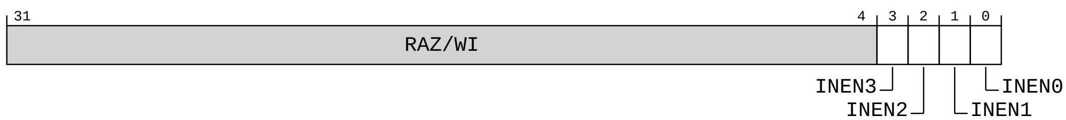


Table B-297: CTIINEN9 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/ WI	Reserved	RAZ/ WI
[3]	INEN3	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[2]	INEN2	Input trigger <n> to output channel <x> enable. Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Bits	Name	Description	Reset
[1]	INEN1	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x
[0]	INEN0	Input trigger <n> to output channel <x> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Input trigger <n> will not generate an event on output channel <x>. **0b1** Input trigger <n> will generate an event on output channel <x>.	x

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

Attributes

Width

32

Component

CTI

Register offset

0xA0

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-181: ext_ctiouten0 bit assignments

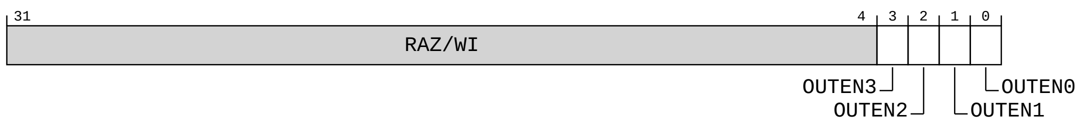


Table B-299: CTIOUTEN0 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	OUTEN3	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[2]	OUTEN2	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

Bits	Name	Description	Reset
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-182: ext_ctiouten1 bit assignments

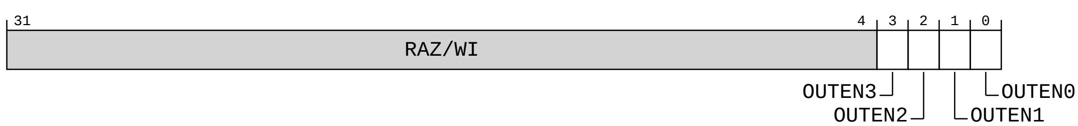


Table B-301: CTIOUTEN1 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	OUTEN3	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

Bits	Name	Description	Reset
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-183: ext_ctiouten2 bit assignments

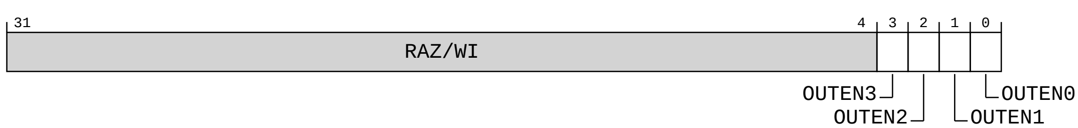


Table B-303: CTIOUTEN2 bit descriptions

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. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-184: ext_ctiouten3 bit assignments

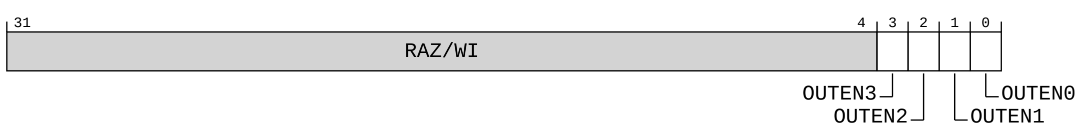


Table B-305: CTIOUTEN3 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	OUTEN3	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[2]	OUTEN2	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[1]	OUTEN1	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[0]	OUTEN0	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-185: ext_ctiouten4 bit assignments

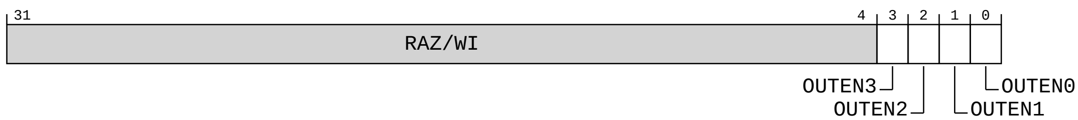


Table B-307: CTIOUTEN4 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset

[0]	OUTEN0	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			`x`

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-186: ext_ctiouten5 bit assignments

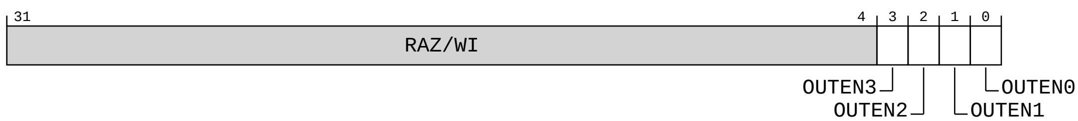


Table B-309: CTIOUTEN5 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`

Bits	Name	Description	Reset
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-187: ext_ctiouten6 bit assignments

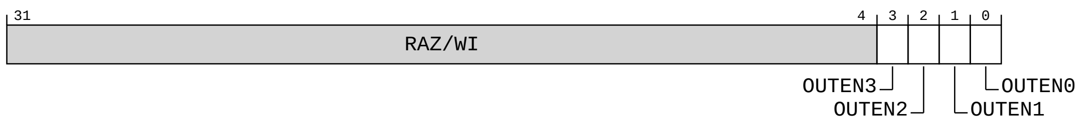


Table B-311: CTIOUTEN6 bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/ WI**
[3]	OUTEN3	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Bits	Name	Description	Reset
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** An event on input channel <x> will not cause output trigger <n> to be asserted. **0b1** An event on input channel <x> will cause output trigger <n> to be asserted.	x

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-188: ext_ctiouten7 bit assignments

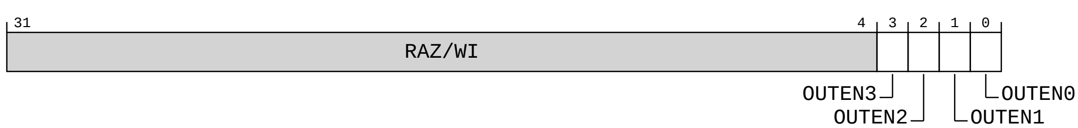


Table B-313: CTIOUTEN7 bit descriptions

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. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[2]	OUTEN2	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[1]	OUTEN1	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`
[0]	OUTEN0	Input channel <x> to output trigger <n> enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** An event on input channel <x> will not cause output trigger <n> to be asserted. **`0b1`** An event on input channel <x> will cause output trigger <n> to be asserted.	`x`

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-189: ext_ctiouten8 bit assignments

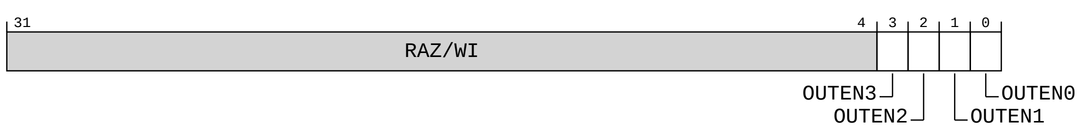


Table B-315: CTIOUTEN8 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	OUTEN3	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[2]	OUTEN2	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[1]	OUTEN1	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[0]	OUTEN0	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

Accessibility

Component	Offset	Instance	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

If output trigger n is not implemented or not connected, CTIOUTEN<n> is RES0.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-190: ext_ctiouten9 bit assignments

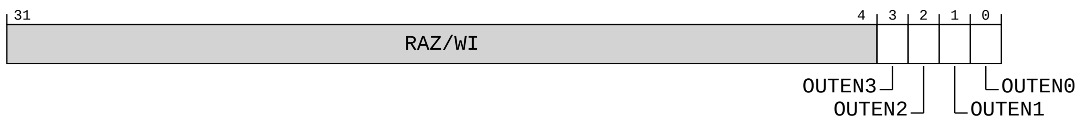


Table B-317: CTIOUTEN9 bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

[3]	OUTEN3	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[2]	OUTEN2	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

[1]	OUTEN1	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are RAZ/WI. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			x

Bits	Name	Description	Reset

[0]	OUTEN0	Input channel <x> to output trigger <n> enable.
		
		Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field.
		
		Possible values of this bit are:
		`0b0`
		An event on input channel <x> will not cause output trigger <n> to be asserted.
		`0b1`
		An event on input channel <x> will cause output trigger <n> to be asserted.
			`x`

Accessibility

Component	Offset	Instance	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 00xx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-191: ext_ctitriginstatus bit assignments

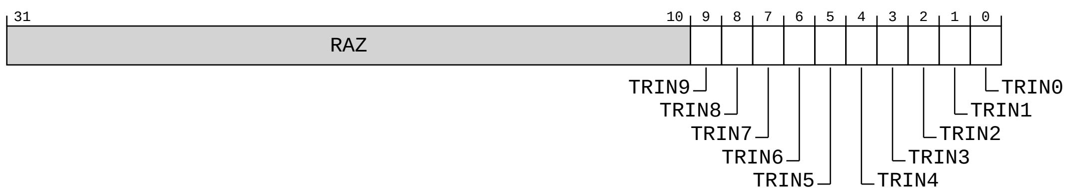


Table B-319: CTITRIGINSTATUS bit descriptions

Bits	Name	Description	Reset
[31:10]	RAZ	Reserved	RAZ
[9]	TRIN9	Trigger input <n> status. Bits [31:N] are RAZ. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **0b0** Input trigger n is inactive. **0b1** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	x
[8]	TRIN8	Trigger input <n> status. Bits [31:N] are RAZ. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **0b0** Input trigger n is inactive. **0b1** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	x

Bits	Name	Description	Reset
[7]	TRIN7	Trigger input <n> status. Bits [31:N] are RAZ. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **`0b0`** Input trigger n is inactive. **`0b1`** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	`x`
[6]	TRIN6	Trigger input <n> status. Bits [31:N] are RAZ. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **`0b0`** Input trigger n is inactive. **`0b1`** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	`x`
[5]	TRIN5	Trigger input <n> status. Bits [31:N] are RAZ. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **`0b0`** Input trigger n is inactive. **`0b1`** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	`x`
[4]	TRIN4	Trigger input <n> status. Bits [31:N] are RAZ. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **`0b0`** Input trigger n is inactive. **`0b1`** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	`x`

Bits	Name	Description	Reset
[3]	TRIN3	Trigger input <n> status. Bits [31:N] are **RAZ**. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **0b0** Input trigger n is inactive. **0b1** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	x
[2]	TRIN2	Trigger input <n> status. Bits [31:N] are **RAZ**. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **0b0** Input trigger n is inactive. **0b1** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	x
[1]	TRIN1	Trigger input <n> status. Bits [31:N] are **RAZ**. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **0b0** Input trigger n is inactive. **0b1** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	x
[0]	TRIN0	Trigger input <n> status. Bits [31:N] are **RAZ**. N is the number of CTI triggers implemented as defined by the ext-CTIDEVID.NUMTRIG field. **0b0** Input trigger n is inactive. **0b1** Input trigger n is active. Not implemented and not-connected input triggers are always inactive. It is IMPLEMENTATION DEFINED whether an input trigger that does not support multicycle events can be observed as active.	x

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 any **IMPLEMENTATION DEFINED** trigger interface logic. For output triggers that are self-acknowledging, this is only meaningful if the CTI implements multicycle channel events.

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 00xx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-192: ext_ctitrigoutstatus bit assignments

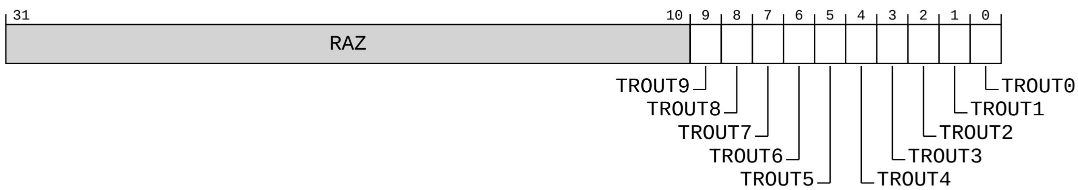


Table B-321: CTITRIGOUTSTATUS bit descriptions

Bits	Name	Description	Reset
[31:10]	**RAZ**	Reserved	**RAZ**
[9]	TROUT9	Trigger output <n> status. Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG. If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are: **`0b0`** Output trigger n is inactive. **`0b1`** Output trigger n is active. Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.	`x`
[8]	TROUT8	Trigger output <n> status. Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG. If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are: **`0b0`** Output trigger n is inactive. **`0b1`** Output trigger n is active. Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.	`x`

Bits	Name	Description	Reset

[7]	TROUT7	Trigger output <n> status.
		
		Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG.
		
		If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are:
		`0b0`
		Output trigger n is inactive.
		`0b1`
		Output trigger n is active.
		
		Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.
			`x`

[6]	TROUT6	Trigger output <n> status.
		
		Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG.
		
		If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are:
		`0b0`
		Output trigger n is inactive.
		`0b1`
		Output trigger n is active.
		
		Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.
			`x`

[5]	TROUT5	Trigger output <n> status.
		
		Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG.
		
		If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are:
		`0b0`
		Output trigger n is inactive.
		`0b1`
		Output trigger n is active.
		
		Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.
			`x`

[4]	TROUT4	Trigger output <n> status.
		
		Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG.
		
		If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are:
		`0b0`
		Output trigger n is inactive.
		`0b1`
		Output trigger n is active.
		
		Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.
			`x`

Bits	Name	Description	Reset
[3]	TROUT3	Trigger output <n> status. Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG. If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are: **`0b0`** Output trigger n is inactive. **`0b1`** Output trigger n is active. Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.	`x`
[2]	TROUT2	Trigger output <n> status. Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG. If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are: **`0b0`** Output trigger n is inactive. **`0b1`** Output trigger n is active. Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.	`x`
[1]	TROUT1	Trigger output <n> status. Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG. If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are: **`0b0`** Output trigger n is inactive. **`0b1`** Output trigger n is active. Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.	`x`
[0]	TROUT0	Trigger output <n> status. Bits [31:N] are **RAZ**. N is the value in ext-CTIDEVID.NUMTRIG. If n < N, and output trigger <n> is implemented and connected, and either the trigger is not self-acknowledging or the CTI implements multicycle channel events, then permitted values for TROUT<n> are: **`0b0`** Output trigger n is inactive. **`0b1`** Output trigger n is active. Otherwise when n < N it is **IMPLEMENTATION DEFINED** whether TROUT<n> behaves as described here or is RAZ.	`x`

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 xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-193: ext_ctichinstatus bit assignments

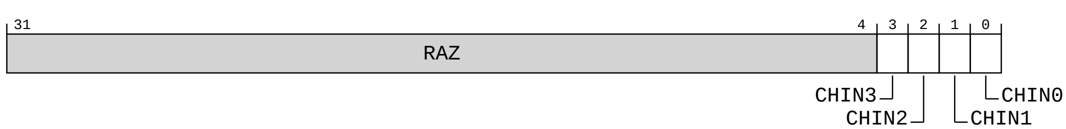


Table B-323: CTICHINSTATUS bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ	Reserved	RAZ
[3]	CHIN3	Input channel <n> status. Bits [31:N] are RAZ. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input channel <n> is inactive. **`0b1`** Input channel <n> is active. If the ECT channels do not support multicycle events then it is IMPLEMENTATION DEFINED whether an input channel can be observed as active.	x
[2]	CHIN2	Input channel <n> status. Bits [31:N] are RAZ. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input channel <n> is inactive. **`0b1`** Input channel <n> is active. If the ECT channels do not support multicycle events then it is IMPLEMENTATION DEFINED whether an input channel can be observed as active.	x
[1]	CHIN1	Input channel <n> status. Bits [31:N] are RAZ. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input channel <n> is inactive. **`0b1`** Input channel <n> is active. If the ECT channels do not support multicycle events then it is IMPLEMENTATION DEFINED whether an input channel can be observed as active.	x
[0]	CHIN0	Input channel <n> status. Bits [31:N] are RAZ. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **`0b0`** Input channel <n> is inactive. **`0b1`** Input channel <n> is active. If the ECT channels do not support multicycle events then it is IMPLEMENTATION DEFINED whether an input channel can be observed as active.	x

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 xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-194: ext_ctichoutstatus bit assignments

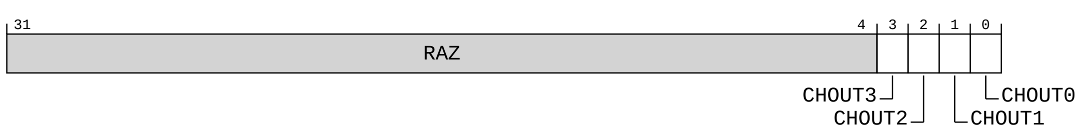


Table B-325: CTICHOUTSTATUS bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ	Reserved	RAZ
[3]	CHOUT3	Output channel <n> status. Bits [31:N] are RAZ. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** Output channel <n> is inactive. **0b1** Output channel <n> is active. If the ECT channels do not support multicycle events then it is IMPLEMENTATION DEFINED whether an output channel can be observed as active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	x
[2]	CHOUT2	Output channel <n> status. Bits [31:N] are RAZ. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **0b0** Output channel <n> is inactive. **0b1** Output channel <n> is active. If the ECT channels do not support multicycle events then it is IMPLEMENTATION DEFINED whether an output channel can be observed as active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	x

Bits	Name	Description	Reset
[1]	CHOUT1	Output channel <n> status. Bits [31:N] are **RAZ**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** Output channel <n> is inactive. **`0b1`** Output channel <n> is active. If the ECT channels do not support multicycle events then it is **IMPLEMENTATION DEFINED** whether an output channel can be observed as active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	x
[0]	CHOUT0	Output channel <n> status. Bits [31:N] are **RAZ**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. Possible values of this bit are: **`0b0`** Output channel <n> is inactive. **`0b1`** Output channel <n> is active. If the ECT channels do not support multicycle events then it is **IMPLEMENTATION DEFINED** whether an output channel can be observed as active. **Note:** The value in CTICHOUTSTATUS is after gating by the channel gate. For more information, see ext-CTIGATE.	x

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 xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-195: ext_ctigate bit assignments

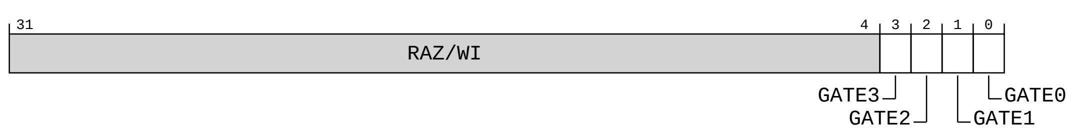


Table B-327: CTIGATE bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	GATE3	Channel <x> gate enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Disable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. **0b1** Enable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. If GATE<x> is set to 0, no new events will be propagated to the ECT, and if the ECT supports multicycle channel events any existing output channel events will be terminated.	`x`
[2]	GATE2	Channel <x> gate enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Disable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. **0b1** Enable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. If GATE<x> is set to 0, no new events will be propagated to the ECT, and if the ECT supports multicycle channel events any existing output channel events will be terminated.	`x`
[1]	GATE1	Channel <x> gate enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Disable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. **0b1** Enable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. If GATE<x> is set to 0, no new events will be propagated to the ECT, and if the ECT supports multicycle channel events any existing output channel events will be terminated.	`x`
[0]	GATE0	Channel <x> gate enable. Bits [31:N] are **RAZ/WI**. N is the number of ECT channels implemented as defined by the ext-CTIDEVID.NUMCHAN field. **0b0** Disable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. **0b1** Enable output and, if ext-CTIDEVID.INOUT == `0b01`, input channel <x> propagation. If GATE<x> is set to 0, no new events will be propagated to the ECT, and if the ECT supports multicycle channel events any existing output channel events will be terminated.	`x`

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

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-196: ext_ctidevctl bit assignments

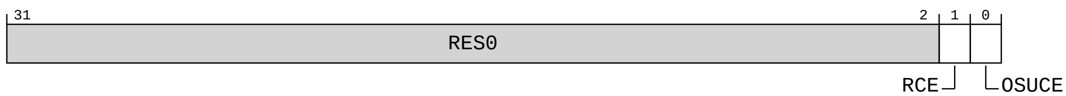


Table B-329: CTIDEVCTL bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	RCE	Reset Catch Enable. **`0b0`** Reset Catch debug event disabled. **`0b1`** Reset Catch debug event enabled.	`0b0`
[0]	OSUCE	OS Unlock Catch Enable **`0b0`** OS Unlock Catch debug event disabled. **`0b1`** OS Unlock Catch debug event enabled.	`0b0`

Accessibility

Component	Offset	Instance	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

```
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-197: ext_cticlaimset bit assignments

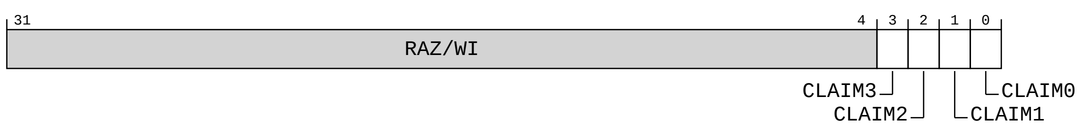


Table B-331: CTICLAIMSET bit descriptions

Bits	Name	Description	Reset
[31:4]	**RAZ/WI**	Reserved	**RAZ/WI**
[3]	CLAIM3	CLAIM tag set bit. If x is less than the **IMPLEMENTATION DEFINED** number of CLAIM tags, this field is RAO and the behavior on writes is: **0b0** No action. **0b1** Indirectly set CLAIM<x> tag to 1. A single write to CTICLAIMSET can set multiple tags to 1. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	x [5]

[5] An External Debug reset clears the CLAIM tag bits to 0.

Bits	Name	Description	Reset
[2]	CLAIM2	CLAIM tag set bit. If x is less than the **IMPLEMENTATION DEFINED** number of CLAIM tags, this field is RAO and the behavior on writes is: **`0b0`** No action. **`0b1`** Indirectly set CLAIM<x> tag to 1. A single write to CTICLAIMSET can set multiple tags to 1. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [6]
[1]	CLAIM1	CLAIM tag set bit. If x is less than the **IMPLEMENTATION DEFINED** number of CLAIM tags, this field is RAO and the behavior on writes is: **`0b0`** No action. **`0b1`** Indirectly set CLAIM<x> tag to 1. A single write to CTICLAIMSET can set multiple tags to 1. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [7]
[0]	CLAIM0	CLAIM tag set bit. If x is less than the **IMPLEMENTATION DEFINED** number of CLAIM tags, this field is RAO and the behavior on writes is: **`0b0`** No action. **`0b1`** Indirectly set CLAIM<x> tag to 1. A single write to CTICLAIMSET can set multiple tags to 1. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [8]

Accessibility

Component	Offset	Instance	Range
CTI	0xFA0	CTICLAIMSET	None

This interface is accessible as follows:

When SoftwareLockStatus()

RO

[6] An External Debug reset clears the CLAIM tag bits to 0.
[7] An External Debug reset clears the CLAIM tag bits to 0.
[8] An External Debug reset clears the CLAIM tag bits to 0.

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 CLAIM tag 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

```
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-198: ext_cticlaimclr bit assignments

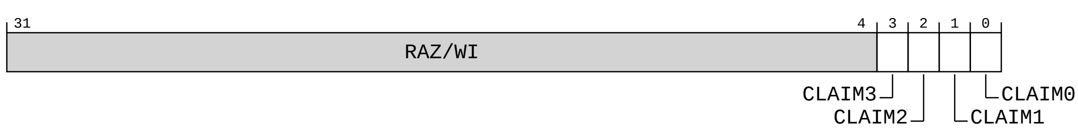


Table B-333: CTICLAIMCLR bit descriptions

Bits	Name	Description	Reset
[31:4]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[3]	CLAIM3	CLAIM tag clear bit. Reads return the value of CLAIM<x>, writes have the following behavior: **`0b0`** No action. **`0b1`** Indirectly clear CLAIM<x> to 0. A single write to CTICLAIMCLR can clear multiple tags to 0. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [9]
[2]	CLAIM2	CLAIM tag clear bit. Reads return the value of CLAIM<x>, writes have the following behavior: **`0b0`** No action. **`0b1`** Indirectly clear CLAIM<x> to 0. A single write to CTICLAIMCLR can clear multiple tags to 0. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [10]
[1]	CLAIM1	CLAIM tag clear bit. Reads return the value of CLAIM<x>, writes have the following behavior: **`0b0`** No action. **`0b1`** Indirectly clear CLAIM<x> to 0. A single write to CTICLAIMCLR can clear multiple tags to 0. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [11]

[9] An External Debug reset clears the CLAIM tag bits to 0.
[10] An External Debug reset clears the CLAIM tag bits to 0.
[11] An External Debug reset clears the CLAIM tag bits to 0.

Bits	Name	Description	Reset
[0]	CLAIM0	CLAIM tag clear bit. Reads return the value of CLAIM<x>, writes have the following behavior: **`0b0`** No action. **`0b1`** Indirectly clear CLAIM<x> to 0. A single write to CTICLAIMCLR can clear multiple tags to 0. If x is greater than or equal to the IMPLEMENTATION DEFINED number of CLAIM tags, this bit is **RAZ/WI**.	`x` [12]

Accessibility

Component	Offset	Instance	Range
CTI	0xFA4	CTICLAIMCLR	None

This interface is accessible as follows:

When SoftwareLockStatus()

RO

When !SoftwareLockStatus()

RW

B.2.1.34 CTIDEVAFF0, CTI Device Affinity register 0

Copy of the low half of the PE AArch64-MPIDR_EL1 register that allows a debugger to determine which PE in a multiprocessor system the CTI component relates to.

Configurations

Not applicable.

Attributes

Width

32

Component

CTI

Register offset

0xFA8

Access type

RO

[12] An External Debug reset clears the CLAIM tag bits to 0.

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-199: ext_ctidevaff0 bit assignments

31 0
MPIDR_EL1lo

Table B-335: CTIDEVAFF0 bit descriptions

Bits	Name	Description	Reset

[31:0]	MPIDR_EL1lo	Identifier for associated core, SME2 unit or cluster for the CTI:
		Cluster
		`{0x00, CLUSTERIDAFF2[7:0], 0x80, 0x00}`
		Core or SME2 unit
		`{0x81, CLUSTERIDAFF2[7:0], Core/SME2 unit identification number[7:0], 0x00}`
		For the cores this value corresponds to a read-only copy of the upper half of AArch64-MPIDR_EL1, as seen from the highest implemented Exception level.
			`32{x}`

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

Not applicable.

Attributes

Width

32

Component

CTI

Register offset

0xFAC

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-200: ext_ctidevaff1 bit assignments

Table B-337: CTIDEVAFF1 bit descriptions

Bits	Name	Description	Reset
[31:0]	MPIDR_EL1hi	Identifier for associated core, SME2 unit or cluster for the CTI: **Cluster** `{24'd0, CLUSTERIDAFF3[7:0]}` **Core or SME2 unit** `{24'd0, CLUSTERIDAFF3[7:0]}` For the cores this value corresponds to a read-only copy of the upper half of AArch64-MPIDR_EL1, as seen from the highest implemented Exception level.	`32{x}`

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 **IMPLEMENTATION DEFINED** authentication interface for CTI.

Configurations

This register is OPTIONAL, and is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFB8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-201: ext_ctiauthstatus bit assignments

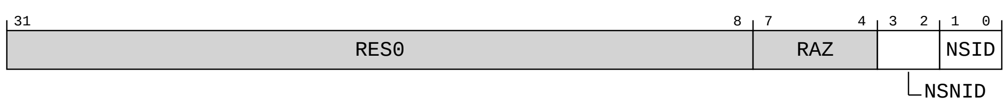


Table B-339: CTIAUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	**RAZ**	Reserved	**RAZ**
[3:2]	NSNID	If EL3 is implemented, this field holds the same value as ext-DBGAUTHSTATUS_EL1.NSNID.	xx
[1:0]	NSID	If EL3 is implemented, this field holds the same value as ext-DBGAUTHSTATUS_EL1.NSID.	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

If the CTI is CTIv1, this register is OPTIONAL. If the CTI is CTIv2, this register is mandatory.

Arm recommends that the CTI is CTIv2.

In an Armv8.5 compliant implementation, the CTI must be CTIv2.

If this register is not implemented, ext-CTIDEVAFF0 and ext-CTIDEVAFF1 are also not implemented.

Attributes

Width

32

Component

CTI

Register offset

0xFBC

Access type

RO

Reset value

```
0100 0111 0111 xxxx 0001 1010 0001 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

> **Note**
>
> Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-202: ext_ctidevarch bit assignments

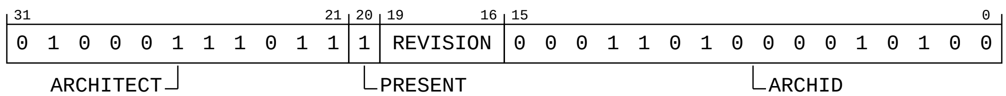


Table B-341: CTIDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Defines the architecture of the component. For CTI, this is Arm Limited. Bits [31:28] are the JEP106 continuation code, `0x4`. Bits [27:21] are the JEP106 ID code, `0x3B`. **`0b01000111011`**	`0b01000111011`
[20]	PRESENT	Indicates that the DEVARCH is present. **`0b1`**	`0b1`
[19:16]	REVISION	Revision. Defines the architecture revision of the component. **`0b0000`** First revision. **`0b0001`** As 0b0000, and also adds support for ext-CTIDEVCTL. All other values are reserved.	The reset values can be the following: `0b0000`, `0b0001`, respective to the value.
[15:0]	ARCHID	Defines this part to be an Armv8 debug component. For architectures defined by Arm this is further subdivided. For CTI: • Bits [15:12] are the architecture version, `0x1`. • Bits [11:0] are the architecture part number, `0xA14`. This corresponds to CTI architecture version CTIv2. **`0b0001101000010100`**	`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

```
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_ctidevid2 bit assignments

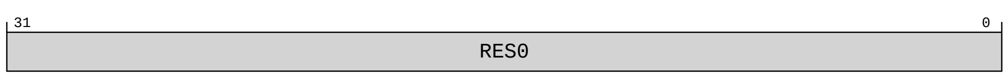


Table B-343: CTIDEVID2 bit descriptions

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-204: ext_ctidevid1 bit assignments

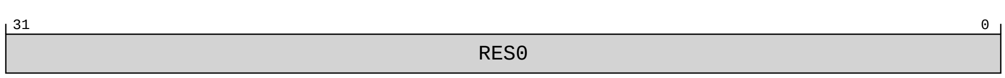


Table B-345: CTIDEVID1 bit descriptions

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 xxxx xx00 0100 xx00 1010 xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-205: ext_ctidevid bit assignments

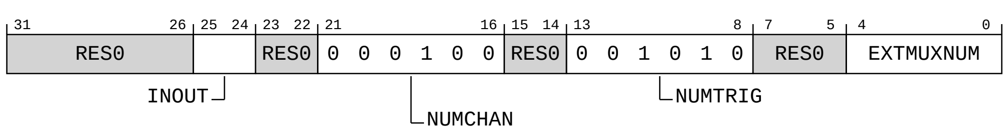


Table B-347: CTIDEVID bit descriptions

Bits	Name	Description	Reset
[31:26]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[25:24]	INOUT	Input/output options. Indicates presence of the input gate. `0b00` - ext-CTIGATE does not mask propagation of input events from external channels. `0b01` - ext-CTIGATE masks propagation of input events from external channels. All other values are reserved. **`0b01`** ext-CTIGATE masks propagation of input events from external channels.	`xx`
[23:22]	**RES0**	Reserved	**RES0**
[21:16]	NUMCHAN	Number of ECT channels implemented. **`0b000100`** 4 channels (0..3) implemented.	`0b000100`
[15:14]	**RES0**	Reserved	**RES0**
[13:8]	NUMTRIG	Number of triggers implemented. **`0b001010`** 10 triggers (0..9) implemented.	`0b001010`
[7:5]	**RES0**	Reserved	**RES0**
[4:0]	EXTMUXNUM	Number of multiplexors available on triggers. This value is used in conjunction with External Control register, ext-ASICCTL.	`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 PE's cross-trigger interface.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CTI

Register offset

0xFCC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0001 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-206: ext_ctidevtype bit assignments

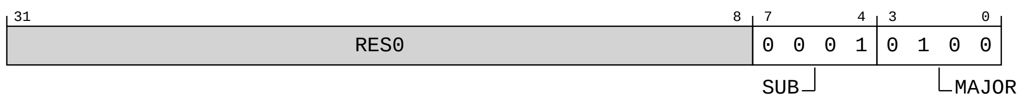


Table B-349: CTIDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SUB	Subtype. Indicates this is a component within a PE. **0b0001**	0b0001
[3:0]	MAJOR	Major type. Indicates this is a cross-trigger component. **0b0100**	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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

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 xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-207: ext_ctipidr4 bit assignments

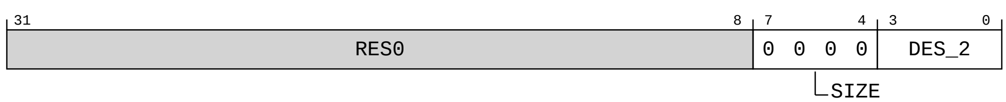


Table B-351: CTIPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SIZE	Size of the component. Log_2 of the number of 4KB pages from the start of the component to the end of the component ID registers. **Cluster, core, or SME2 unit** `0b0000`	`0b0000`
[3:0]	DES_2	Designer, JEP106 continuation code, least significant nibble. For Arm Limited, this field is: **Cluster, core, or SME2 unit** `0b0100`	`xxxx`

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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFE0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1110 1110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-208: ext_ctipidr0 bit assignments

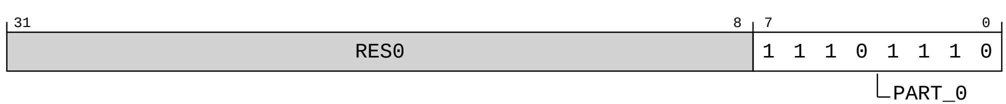


Table B-353: CTIPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset

[7:0]	PART_0	Cross Trigger Interface. Bits [7:0] of part number.
		Cluster
		`0b11101110`
		Cluster Cross Trigger Interface. Bits [7:0] of part number `0x4EE`.
		Core or SME2 unit
		`0b11101101`
		Core or SME2 unit Cross Trigger Interface. Bits [7:0] of part number `0x4ED`.
			Cluster
			`0xEE`
			Core or SME2 unit
			`0xED`

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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFE4

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-209: ext_ctipidr1 bit assignments

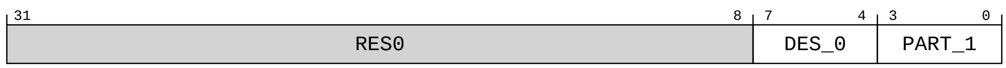


Table B-355: CTIPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	Designer, least significant nibble of JEP106 ID code. For Arm Limited, this field is: **Cluster, core, or SME2 unit** `0b1011`	`1011`
[3:0]	PART_1	Part number, most significant nibble: **Cluster, core, or SME2 unit** `0b0100`	`0100`

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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFE8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx 1xxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-210: ext_ctipidr2 bit assignments

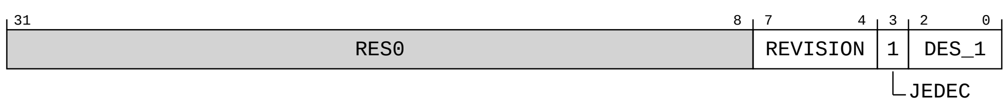


Table B-357: CTIPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Part major revision. Parts can also use this field to extend Part number to 16-bits: **Cluster, core, or SME2 unit** `0b0010`	`0010`
[3]	JEDEC	Indicates a JEP106 identity code is used. **Cluster, core, or SME2 unit** `0b1`	`0b1`
[2:0]	DES_1	Designer, most significant bits of JEP106 ID code. For Arm Limited, this field is: **Cluster, core, or SME2 unit** `0b011` .	`xxx`

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.

For more information, see *About the Peripheral identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFEC

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-211: ext_ctipidr3 bit assignments

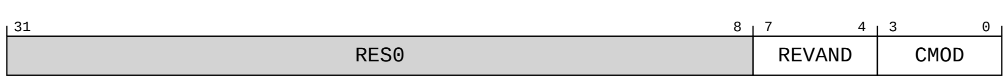


Table B-359: CTIPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	REVAND	Part minor revision. Parts using ext-CTIPIDR2.REVISION as an extension to the Part number must use this field as a major revision number. **Cluster, core, or SME2 unit** `0b0000`	`0000`
[3:0]	CMOD	Customer modified. Indicates someone other than the Designer has modified the component.	`xxxx`

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.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

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-212: ext_cticidr0 bit assignments

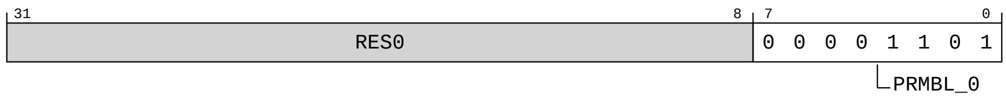


Table B-361: CTICIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	Preamble. `0b00001101`	`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.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFF4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-213: ext_cticidr1 bit assignments

Table B-363: CTICIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	Component class. **`0b1001`** CoreSight component. Other values are defined by the CoreSight Architecture. This field reads as `0x9`.	`xxxx`
[3:0]	PRMBL_1	Preamble. **RAZ**. **`0b0000`**	`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.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-214: ext_cticidr2 bit assignments

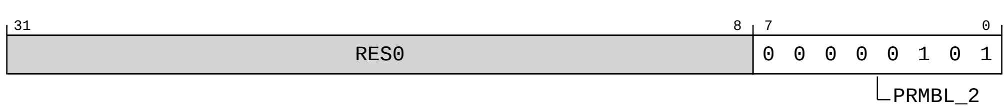


Table B-365: CTICIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	Preamble. `0b00000101`	`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.

For more information, see *About the Component Identification scheme* in the *Arm® Architecture Reference Manual for A-profile architecture*.

Configurations

This register is required for CoreSight compliance.

Attributes

Width

32

Component

CTI

Register offset

0xFFC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-215: ext_cticidr3 bit assignments

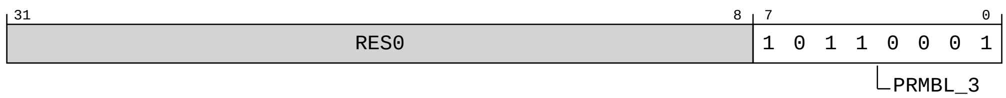


Table B-367: CTICIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Preamble. `0b10110001`	`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.

Note

- The cluster ROM table registers are treated as **RAZ/WI** if the register is marked Reserved.
- Any address that is not documented is treated as **RAZ/WI**.
- If the C1-DSU is configured for Direct connect, all these registers are present.
- If the C1-DSU is enabled for Realm Management Extension (RME) all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.
- The cluster ROM table registers part number is `0x4F0`.

Table B-369: CLUSTERROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	CLUSTERROM_ROMENTRY0	See individual bit resets.	32-bit	Cluster ROM table entry 0	Yes
0x004	CLUSTERROM_ROMENTRY1	See individual bit resets.	32-bit	Cluster ROM table entry 1	Yes
0x8	CLUSTERROM_ROMENTRY2	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0xC	CLUSTERROM_ROMENTRY3	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x10	CLUSTERROM_ROMENTRY4	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x14	CLUSTERROM_ROMENTRY5	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x18	CLUSTERROM_ROMENTRY6	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x1C	CLUSTERROM_ROMENTRY7	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x20	CLUSTERROM_ROMENTRY8	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x24	CLUSTERROM_ROMENTRY9	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x28	CLUSTERROM_ROMENTRY10	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x2C	CLUSTERROM_ROMENTRY11	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x30	CLUSTERROM_ROMENTRY12	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x34	CLUSTERROM_ROMENTRY13	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x38	CLUSTERROM_ROMENTRY14	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x3C	CLUSTERROM_ROMENTRY15	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x40	CLUSTERROM_ROMENTRY16	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0x44	CLUSTERROM_ROMENTRY17	See individual bit resets.	32-bit	Cluster ROM table entry	Yes
0xA00	CLUSTERROM_DBGPCR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA04	CLUSTERROM_DBGPCR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA08	CLUSTERROM_DBGPCR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA0C	CLUSTERROM_DBGPCR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xA10	CLUSTERROM_DBGPCR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA14	CLUSTERROM_DBGPCR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA18	CLUSTERROM_DBGPCR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA1C	CLUSTERROM_DBGPCR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA20	CLUSTERROM_DBGPCR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA24	CLUSTERROM_DBGPCR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA28	CLUSTERROM_DBGPCR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA2C	CLUSTERROM_DBGPCR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA30	CLUSTERROM_DBGPCR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA34	CLUSTERROM_DBGPCR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA38	CLUSTERROM_DBGPCR14	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA3C	CLUSTERROM_DBGPCR15	See individual bit resets.	32-bit	Cluster ROM table Debug Power Control Register	Yes
0xA80	CLUSTERROM_DBGPSR0	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA84	CLUSTERROM_DBGPSR1	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA88	CLUSTERROM_DBGPSR2	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA8C	CLUSTERROM_DBGPSR3	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA90	CLUSTERROM_DBGPSR4	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA94	CLUSTERROM_DBGPSR5	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA98	CLUSTERROM_DBGPSR6	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xA9C	CLUSTERROM_DBGPSR7	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAA0	CLUSTERROM_DBGPSR8	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAA4	CLUSTERROM_DBGPSR9	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAA8	CLUSTERROM_DBGPSR10	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0xAAC	CLUSTERROM_DBGPSR11	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAB0	CLUSTERROM_DBGPSR12	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAB4	CLUSTERROM_DBGPSR13	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xAB8	CLUSTERROM_DBGPSR14	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xABC	CLUSTERROM_DBGPSR15	See individual bit resets.	32-bit	Cluster ROM table Debug Power Status Register	Yes
0xC00	CLUSTERROM_PRIDR0	See individual bit resets.	32-bit	Cluster ROM table Power Request ID Register 0	Yes
0xFB8	CLUSTERROM_AUTHSTATUS	See individual bit resets.	32-bit	Cluster ROM table Authentication Status Register	Yes
0xFBC	CLUSTERROM_DEVARCH	See individual bit resets.	32-bit	Cluster ROM table Device Architecture Register	Yes
0xFC8	CLUSTERROM_DEVID	See individual bit resets.	32-bit	Cluster ROM table Device Configuration Register	Yes
0xFCC	CLUSTERROM_DEVTYPE	See individual bit resets.	32-bit	Cluster ROM table Device Type Register	Yes
0xFD0	CLUSTERROM_PIDR4	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 4	Yes
0xFE0	CLUSTERROM_PIDR0	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 0	Yes
0xFE4	CLUSTERROM_PIDR1	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 1	Yes
0xFE8	CLUSTERROM_PIDR2	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 2	Yes
0xFEC	CLUSTERROM_PIDR3	See individual bit resets.	32-bit	Cluster ROM table Peripheral Identification Register 3	Yes
0xFF0	CLUSTERROM_CIDR0	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 0	Yes
0xFF4	CLUSTERROM_CIDR1	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 1	Yes
0xFF8	CLUSTERROM_CIDR2	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 2	Yes
0xFFC	CLUSTERROM_CIDR3	See individual bit resets.	32-bit	Cluster ROM table Component Identification Register 3	Yes

B.2.2.1 CLUSTERROM_ROMENTRY0, Cluster ROM table entry 0

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x000

Access type

RO

Reset value

```
0000 0000 0000 0001 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-216: ext_clusterrom_romentry0 bit assignments

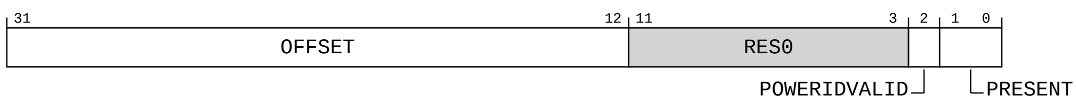


Table B-370: CLUSTERROM_ROMENTRY0 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000000010000** Cluster PMU table at address 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. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM entry is present.	0b11

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

```
0000 0000 0000 0010 0000 xxxx xxxx x0xx
|    |    |    |    |    |    |    |   |
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_romentry1 bit assignments

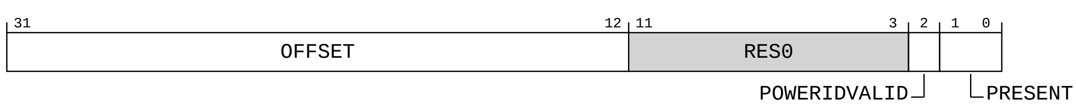


Table B-371: CLUSTERROM_ROMENTRY1 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000000000100000** Cluster ELA at address 0x3_0000 in the Cluster Debug APB address map.	0x00020
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b10** ELA is not present so the ROM entry is not present. **0b11** ELA is present so the ROM entry is present.	xx

Accessibility

This interface is accessible as follows:

RO

B.2.2.3 CLUSTERROM_ROMENTRY2, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x8

Access type

RO

Reset value

```
0000 0000 0000 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
```

31	27	23	19	15	11	7	3	0

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-218: ext_clusterrom_romentry2 bit assignments

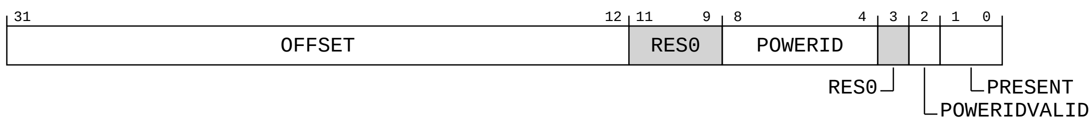


Table B-372: CLUSTERROM_ROMENTRY2 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000000001110000** Core or Complex or SME2 unit ROM table at address 0x0008_0000	0x00070
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.4 CLUSTERROM_ROMENTRY3, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xC

Access type

RO

Reset value

```
0000 0000 0000 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-219: ext_clusterrom_romentry3 bit assignments

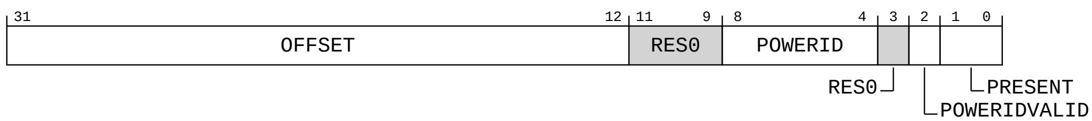


Table B-373: CLUSTERROM_ROMENTRY3 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000000011110000** Core or Complex or SME2 unit ROM table at address 0x0010_0000	0x000F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.5 CLUSTERROM_ROMENTRY4, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x10

Access type

RO

Reset value

```
0000 0000 0001 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-220: ext_clusterrom_romentry4 bit assignments

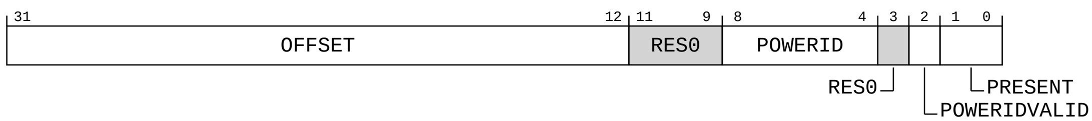


Table B-374: CLUSTERROM_ROMENTRY4 bit descriptions

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. **0b00000000000101110000** Core or Complex or SME2 unit ROM table at address 0x0018_0000	0x00170
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.6 CLUSTERROM_ROMENTRY5, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x14

Access type

RO

Reset value

```
0000 0000 0001 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-221: ext_clusterrom_romentry5 bit assignments

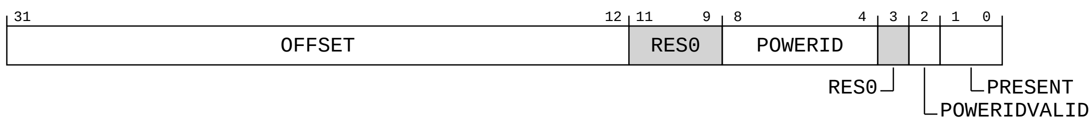


Table B-375: CLUSTERROM_ROMENTRY5 bit descriptions

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. **0b00000000000111110000** Core or Complex or SME2 unit ROM table at address 0x0020_0000	0x001F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	`0b11`

Accessibility

This interface is accessible as follows:

RO

B.2.2.7 CLUSTERROM_ROMENTRY6, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x18

Access type

RO

Reset value

```
0000 0000 0010 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-222: ext_clusterrom_romentry6 bit assignments

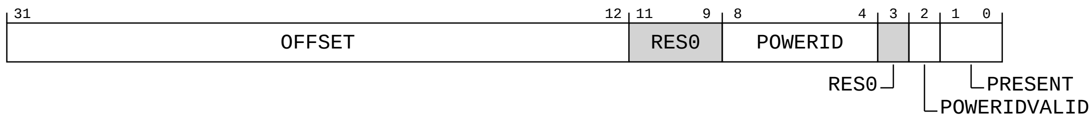


Table B-376: CLUSTERROM_ROMENTRY6 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000001001110000** Core or Complex or SME2 unit ROM table at address 0x0028_0000	0x00270
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.8 CLUSTERROM_ROMENTRY7, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x1C

Access type

RO

Reset value

```
0000 0000 0010 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-223: ext_clusterrom_romentry7 bit assignments

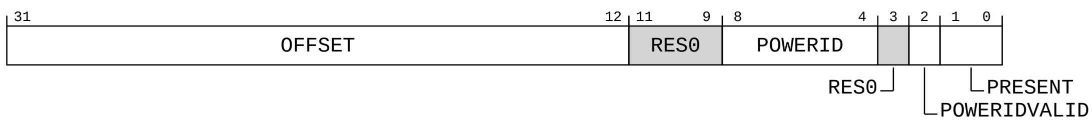


Table B-377: CLUSTERROM_ROMENTRY7 bit descriptions

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. **0b00000000001011110000** Core or Complex or SME2 unit ROM table at address 0x0030_0000	0x002F0
[11:9]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.9 CLUSTERROM_ROMENTRY8, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x20

Access type

RO

Reset value

```
0000 0000 0011 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-224: ext_clusterrom_romentry8 bit assignments

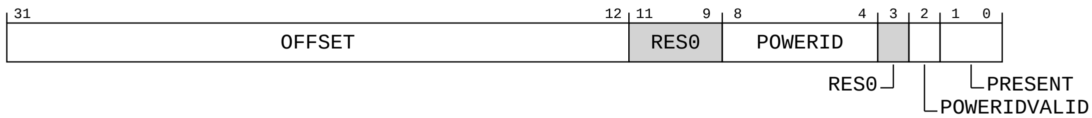


Table B-378: CLUSTERROM_ROMENTRY8 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000001101110000** Core or Complex or SME2 unit ROM table at address 0x0038_0000	0x00370
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.10 CLUSTERROM_ROMENTRY9, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x24

Access type

RO

Reset value

```
0000 0000 0011 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-225: ext_clusterrom_romentry9 bit assignments

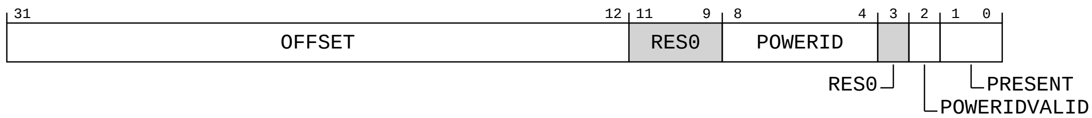


Table B-379: CLUSTERROM_ROMENTRY9 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). The value of this field depends on the cluster configuration. **0b00000000001111110000** Core or Complex or SME2 unit ROM table at address 0x0040_0000	0x003F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.11 CLUSTERROM_ROMENTRY10, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x28

Access type

RO

Reset value

```
0000 0000 0100 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

> **Note**
>
> Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-226: ext_clusterrom_romentry10 bit assignments

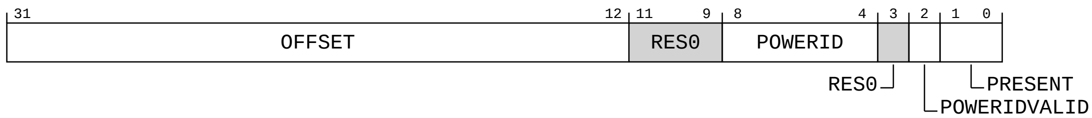


Table B-380: CLUSTERROM_ROMENTRY10 bit descriptions

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. **0b00000000010001110000** Core or Complex or SME2 unit ROM table at address 0x0048_0000	0x00470
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.12 CLUSTERROM_ROMENTRY11, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x2C

Access type

RO

Reset value

```
0000 0000 0100 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-227: ext_clusterrom_romentry11 bit assignments

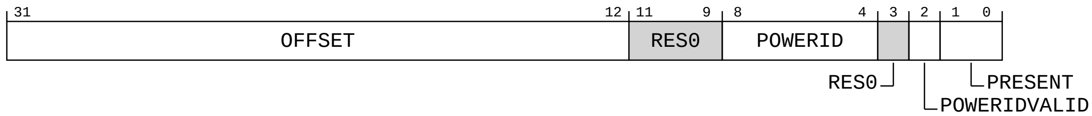


Table B-381: CLUSTERROM_ROMENTRY11 bit descriptions

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. **0b00000000010011110000** Core or Complex or SME2 unit ROM table at address 0x0050_0000	0x004F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.13 CLUSTERROM_ROMENTRY12, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x30

Access type

RO

Reset value

```
0000 0000 0101 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-228: ext_clusterrom_romentry12 bit assignments

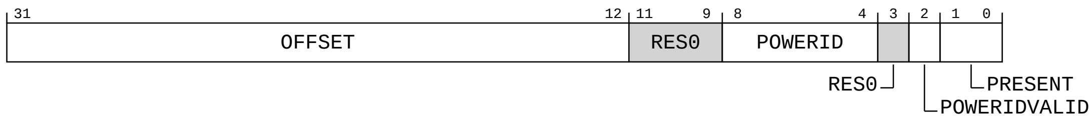


Table B-382: CLUSTERROM_ROMENTRY12 bit descriptions

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. **0b00000000010101110000** Core or Complex or SME2 unit ROM table at address 0x0058_0000	0x00570
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.14 CLUSTERROM_ROMENTRY13, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x34

Access type

RO

Reset value

```
0000 0000 0101 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-229: ext_clusterrom_romentry13 bit assignments

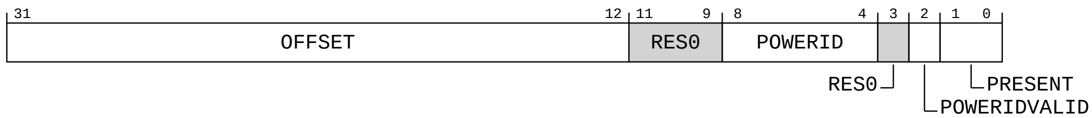


Table B-383: CLUSTERROM_ROMENTRY13 bit descriptions

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. **0b00000000010111110000** Core or Complex or SME2 unit ROM table at address 0x0060_0000	0x005F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.15 CLUSTERROM_ROMENTRY14, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x38

Access type

RO

Reset value

```
0000 0000 0110 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-230: ext_clusterrom_romentry14 bit assignments

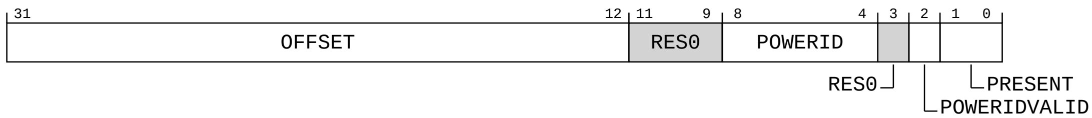


Table B-384: CLUSTERROM_ROMENTRY14 bit descriptions

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). The value of this field depends on the cluster configuration. **0b00000000011001110000** Core or Complex or SME2 unit ROM table at address 0x0068_0000	0x00670
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.16 CLUSTERROM_ROMENTRY15, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x3C

Access type

RO

Reset value

```
0000 0000 0110 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-231: ext_clusterrom_romentry15 bit assignments

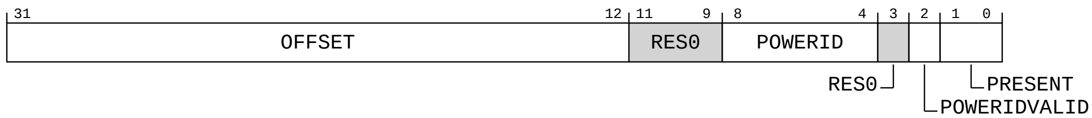


Table B-385: CLUSTERROM_ROMENTRY15 bit descriptions

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. **0b00000000011011110000** Core or Complex or SME2 unit ROM table at address 0x0070_0000	0x006F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.17 CLUSTERROM_ROMENTRY16, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x40

Access type

RO

Reset value

```
0000 0000 0111 0111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-232: ext_clusterrom_romentry16 bit assignments

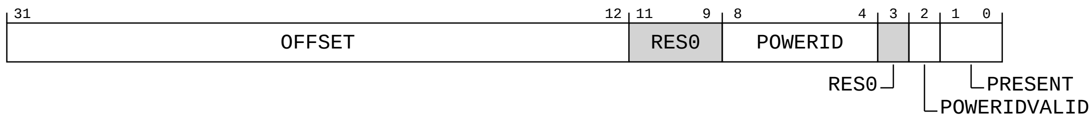


Table B-386: CLUSTERROM_ROMENTRY16 bit descriptions

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. **0b00000000011101110000** Core or Complex or SME2 unit ROM table at address 0x0078_0000	`0x00770`
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **`0b00`** The ROM entry is not present and this is the final entry in the ROM table. **`0b01`** Reserved. **`0b10`** The ROM entry is not present and this is not the final entry in the ROM table. **`0b11`** The ROM entry is present.	`0b11`

Accessibility

This interface is accessible as follows:

RO

B.2.2.18 CLUSTERROM_ROMENTRY17, Cluster ROM table entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0x44

Access type

RO

Reset value

```
0000 0000 0111 1111 0000 xxx0 0000 x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-233: ext_clusterrom_romentry17 bit assignments

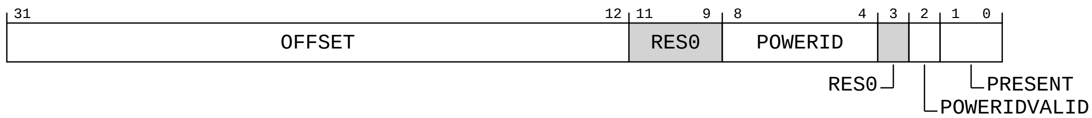


Table B-387: CLUSTERROM_ROMENTRY17 bit descriptions

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. **0b00000000011111110000** Core or Complex or SME2 unit ROM table at address 0x0080_0000	0x007F0
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCOMPLEX0 power domain. **`0b00001`** PDCOMPLEX1 power domain. **`0b00010`** PDCOMPLEX2 power domain. **`0b00011`** PDCOMPLEX3 power domain. **`0b00100`** PDCOMPLEX4 power domain. **`0b00101`** PDCOMPLEX5 power domain. **`0b00110`** PDCOMPLEX6 power domain. **`0b00111`** PDCOMPLEX7 power domain. **`0b01000`** PDCOMPLEX8 power domain. **`0b01001`** PDCOMPLEX9 power domain. **`0b01010`** PDCOMPLEX10 power domain. **`0b01011`** PDCOMPLEX11 power domain. **`0b01100`** PDCOMPLEX12 power domain. **`0b01101`** PDCOMPLEX13 power domain.	`0b00000`
[3]	RES0	Reserved	RES0
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b0`** A power domain ID is not provided. **`0b1`** The POWERID field provides a power domain ID.	`0b0`

Bits	Name	Description	Reset
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. The value of this field depends on the cluster configuration. **0b00** The ROM entry is not present and this is the final entry in the ROM table. **0b01** Reserved. **0b10** The ROM entry is not present and this is not the final entry in the ROM table. **0b11** The ROM entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.2.19 CLUSTERROM_DBGPCR0, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 0.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA00

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-234: ext_clusterrom_dbgpcr0 bit assignments

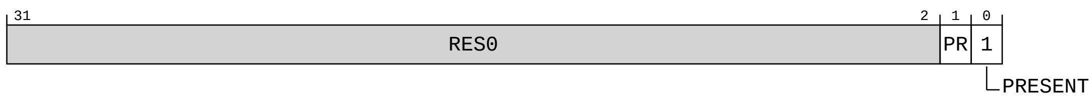


Table B-388: CLUSTERROM_DBGPCR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.20 CLUSTERROM_DBGPCR1, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 1.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA04

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-235: ext_clusterrom_dbgpcr1 bit assignments

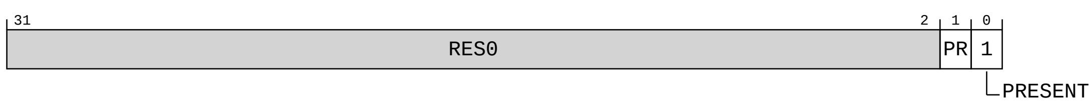


Table B-389: CLUSTERROM_DBGPCR1 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.21 CLUSTERROM_DBGPCR2, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 2.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA08

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-236: ext_clusterrom_dbgpcr2 bit assignments

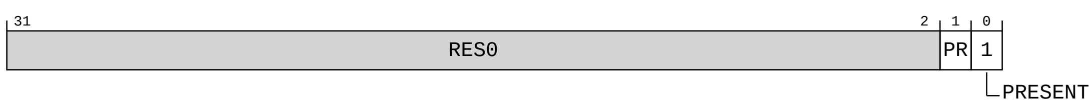


Table B-390: CLUSTERROM_DBGPCR2 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for target core, complex or SME2 unit power domain. **0b1** Power is requested for target core, complex or SME2 unit power domain.	0b0

Bits	Name	Description	Reset
[0]	PRESENT	Power request implemented. **0b1** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	0b1

Accessibility

This interface is accessible as follows:

RW

B.2.2.22 CLUSTERROM_DBGPCR3, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 3.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA0C

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-237: ext_clusterrom_dbgpcr3 bit assignments

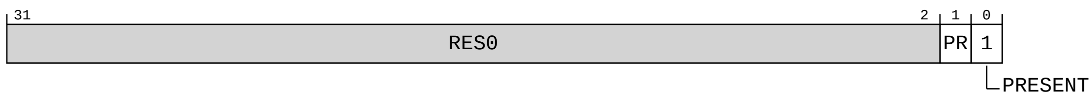


Table B-391: CLUSTERROM_DBGPCR3 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **0b0** Power is not requested for target core, complex or SME2 unit power domain. **0b1** Power is requested for target core, complex or SME2 unit power domain.	0b0
[0]	PRESENT	Power request implemented. **0b1** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	0b1

Accessibility

This interface is accessible as follows:

RW

B.2.2.23 CLUSTERROM_DBGPCR4, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 4.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA10

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-238: ext_clusterrom_dbgpcr4 bit assignments

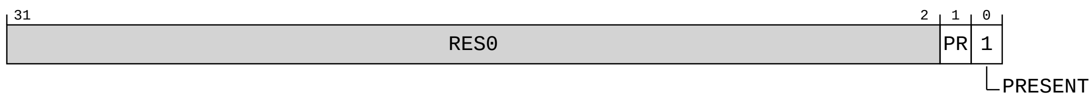


Table B-392: CLUSTERROM_DBGPCR4 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.24 CLUSTERROM_DBGPCR5, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 5.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA14

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-239: ext_clusterrom_dbgpcr5 bit assignments

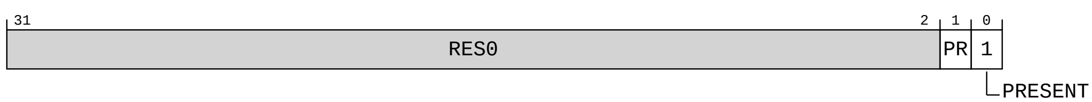


Table B-393: CLUSTERROM_DBGPCR5 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.25 CLUSTERROM_DBGPCR6, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 6.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA18

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-240: ext_clusterrom_dbgpcr6 bit assignments

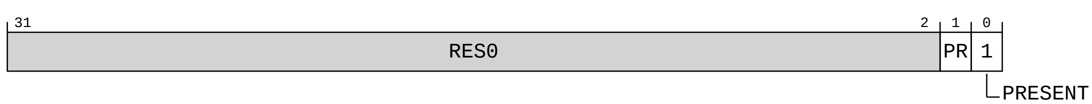


Table B-394: CLUSTERROM_DBGPCR6 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.26 CLUSTERROM_DBGPCR7, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 7.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA1C

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-241: ext_clusterrom_dbgpcr7 bit assignments

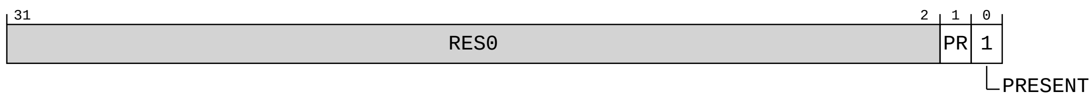


Table B-395: CLUSTERROM_DBGPCR7 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.27 CLUSTERROM_DBGPCR8, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 8.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA20

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-242: ext_clusterrom_dbgpcr8 bit assignments

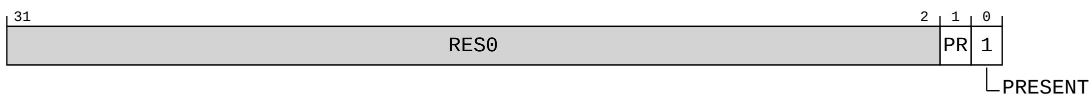


Table B-396: CLUSTERROM_DBGPCR8 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for target core, complex or SME2 unit power domain. **0b1** Power is requested for target core, complex or SME2 unit power domain.	0b0
[0]	PRESENT	Power request implemented. **0b1** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	0b1

Accessibility

This interface is accessible as follows:

RW

B.2.2.28 CLUSTERROM_DBGPCR9, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 9.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA24

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-243: ext_clusterrom_dbgpcr9 bit assignments

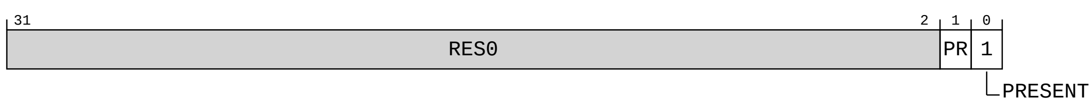


Table B-397: CLUSTERROM_DBGPCR9 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**

[1]	PR	Power Request.
		`0b0`
		Power is not requested for target core, complex or SME2 unit power domain.
		`0b1`
		Power is requested for target core, complex or SME2 unit power domain.
			`0b0`

[0]	PRESENT	Power request implemented. `0b1` Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.29 CLUSTERROM_DBGPCR10, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 10.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA28

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-244: ext_clusterrom_dbgpcr10 bit assignments

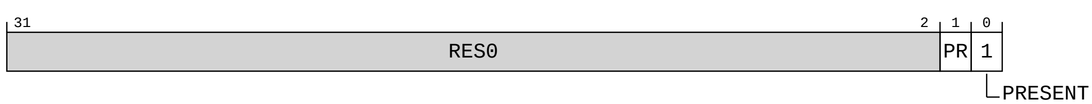


Table B-398: CLUSTERROM_DBGPCR10 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1]	PR	Power Request. **0b0** Power is not requested for target core, complex or SME2 unit power domain. **0b1** Power is requested for target core, complex or SME2 unit power domain.	0b0
[0]	PRESENT	Power request implemented. **0b1** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	0b1

Accessibility

This interface is accessible as follows:

RW

B.2.2.30 CLUSTERROM_DBGPCR11, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 11.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA2C

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-245: ext_clusterrom_dbgpcr11 bit assignments

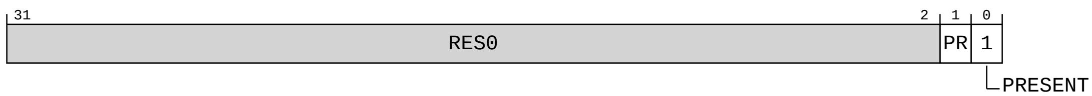


Table B-399: CLUSTERROM_DBGPCR11 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.31 CLUSTERROM_DBGPCR12, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 12.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA30

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-246: ext_clusterrom_dbgpcr12 bit assignments

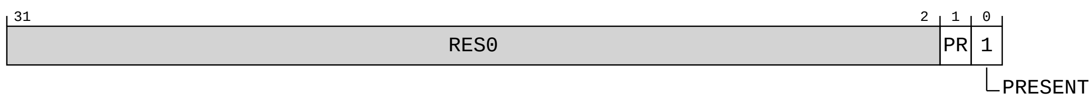


Table B-400: CLUSTERROM_DBGPCR12 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for target core, complex or SME2 unit power domain. **0b1** Power is requested for target core, complex or SME2 unit power domain.	0b0
[0]	PRESENT	Power request implemented. **0b1** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	0b1

Accessibility

This interface is accessible as follows:

RW

B.2.2.32 CLUSTERROM_DBGPCR13, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 13.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA34

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-247: ext_clusterrom_dbgpcr13 bit assignments

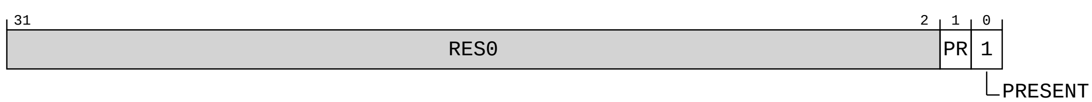


Table B-401: CLUSTERROM_DBGPCR13 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.33 CLUSTERROM_DBGPCR14, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 14.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA38

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-248: ext_clusterrom_dbgpcr14 bit assignments

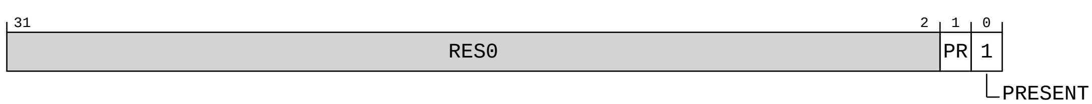


Table B-402: CLUSTERROM_DBGPCR14 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1]	PR	Power Request. **0b0** Power is not requested for target core, complex or SME2 unit power domain. **0b1** Power is requested for target core, complex or SME2 unit power domain.	0b0
[0]	PRESENT	Power request implemented. **0b1** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	0b1

Accessibility

This interface is accessible as follows:

RW

B.2.2.34 CLUSTERROM_DBGPCR15, Cluster ROM table Debug Power Control Register

Controls power requests for PDCOMPLEX 15.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA3C

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx01
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-249: ext_clusterrom_dbgpcr15 bit assignments

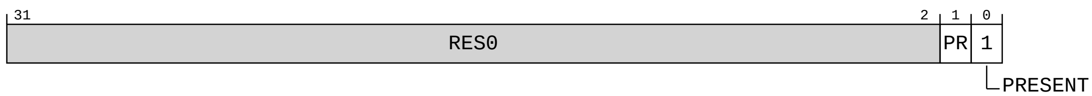


Table B-403: CLUSTERROM_DBGPCR15 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1]	PR	Power Request. **`0b0`** Power is not requested for target core, complex or SME2 unit power domain. **`0b1`** Power is requested for target core, complex or SME2 unit power domain.	`0b0`
[0]	PRESENT	Power request implemented. **`0b1`** Power request for target core, complex, or SME2 unit power domain is included in the PPU power control.	`0b1`

Accessibility

This interface is accessible as follows:

RW

B.2.2.35 CLUSTERROM_DBGPSR0, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 0.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA80

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-250: ext_clusterrom_dbgpsr0 bit assignments

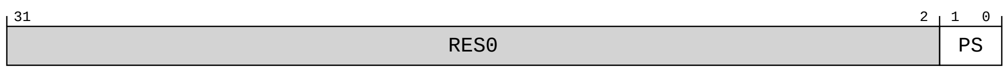


Table B-404: CLUSTERROM_DBGPSR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.36 CLUSTERROM_DBGPSR1, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 1.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA84

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-251: ext_clusterrom_dbgpsr1 bit assignments

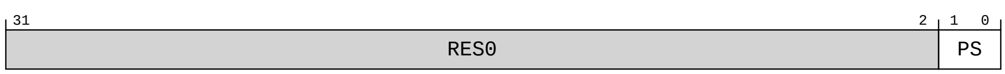


Table B-405: CLUSTERROM_DBGPSR1 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.37 CLUSTERROM_DBGPSR2, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 2.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA88

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-252: ext_clusterrom_dbgpsr2 bit assignments

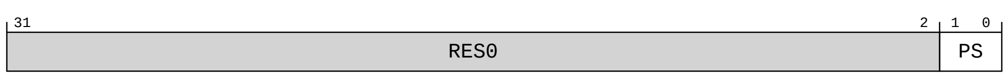


Table B-406: CLUSTERROM_DBGPSR2 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.38 CLUSTERROM_DBGPSR3, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 3.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA8C

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-253: ext_clusterrom_dbgpsr3 bit assignments

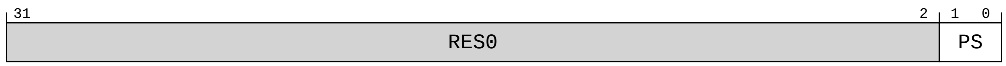


Table B-407: CLUSTERROM_DBGPSR3 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.39 CLUSTERROM_DBGPSR4, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 4.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA90

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-254: ext_clusterrom_dbgpsr4 bit assignments

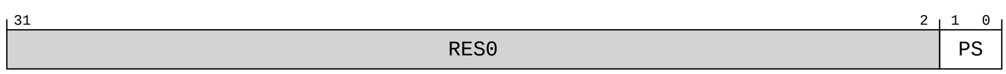


Table B-408: CLUSTERROM_DBGPSR4 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.40 CLUSTERROM_DBGPSR5, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 5.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA94

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-255: ext_clusterrom_dbgpsr5 bit assignments

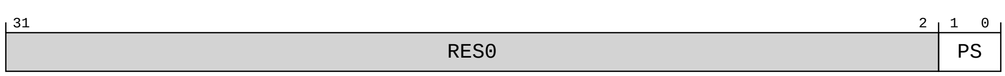


Table B-409: CLUSTERROM_DBGPSR5 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.41 CLUSTERROM_DBGPSR6, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 6.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA98

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-256: ext_clusterrom_dbgpsr6 bit assignments

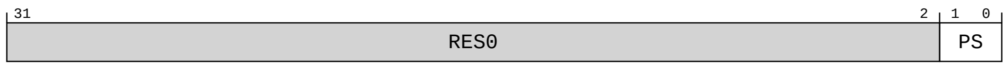


Table B-410: CLUSTERROM_DBGPSR6 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **`0b00`** Target core, complex, or SME2 unit debug power domain might not be powered. **`0b01`** Target core, complex, or SME2 unit debug power domain is powered. **`0b10`** Reserved. **`0b11`** Target core, complex, or SME2 unit 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.2.2.42 CLUSTERROM_DBGPSR7, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 7.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xA9C

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-257: ext_clusterrom_dbgpsr7 bit assignments

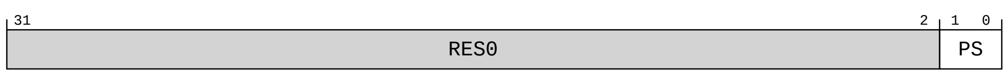


Table B-411: CLUSTERROM_DBGPSR7 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.43 CLUSTERROM_DBGPSR8, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 8.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAA0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-258: ext_clusterrom_dbgpsr8 bit assignments

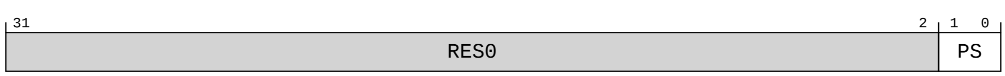


Table B-412: CLUSTERROM_DBGPSR8 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.44 CLUSTERROM_DBGPSR9, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 9.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAA4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-259: ext_clusterrom_dbgpsr9 bit assignments

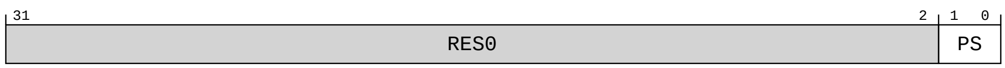


Table B-413: CLUSTERROM_DBGPSR9 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.45 CLUSTERROM_DBGPSR10, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 10.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAA8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-260: ext_clusterrom_dbgpsr10 bit assignments

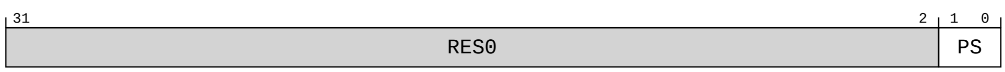


Table B-414: CLUSTERROM_DBGPSR10 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.46 CLUSTERROM_DBGPSR11, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 11.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAAC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-261: ext_clusterrom_dbgpsr11 bit assignments

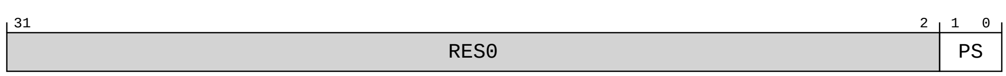


Table B-415: CLUSTERROM_DBGPSR11 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.47 CLUSTERROM_DBGPSR12, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 12.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAB0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-262: ext_clusterrom_dbgpsr12 bit assignments

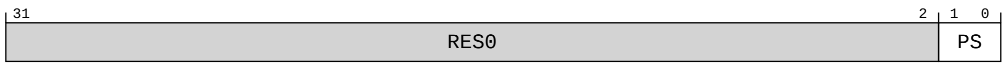


Table B-416: CLUSTERROM_DBGPSR12 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.48 CLUSTERROM_DBGPSR13, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 13.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAB4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-263: ext_clusterrom_dbgpsr13 bit assignments

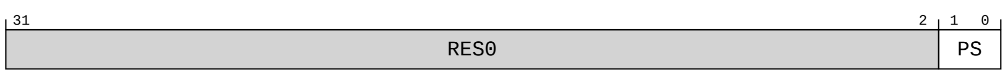


Table B-417: CLUSTERROM_DBGPSR13 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.49 CLUSTERROM_DBGPSR14, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 14.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xAB8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-264: ext_clusterrom_dbgpsr14 bit assignments

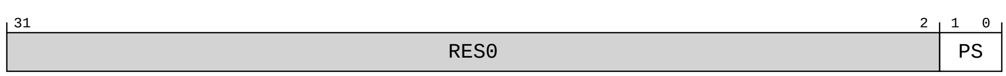


Table B-418: CLUSTERROM_DBGPSR14 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.50 CLUSTERROM_DBGPSR15, Cluster ROM table Debug Power Status Register

Indicates the power status for PDCOMPLEX 15.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xABC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note: Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-265: ext_clusterrom_dbgpsr15 bit assignments

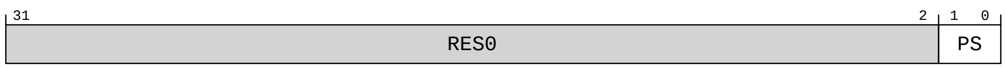


Table B-419: CLUSTERROM_DBGPSR15 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1:0]	PS	Power Status. **0b00** Target core, complex, or SME2 unit debug power domain might not be powered. **0b01** Target core, complex, or SME2 unit debug power domain is powered. **0b10** Reserved. **0b11** Target core, complex, or SME2 unit 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.2.2.51 CLUSTERROM_PRIDR0, Cluster ROM table Power Request ID Register 0

Indicates the features of the power request functionality.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xC00

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xx00 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-266: ext_clusterrom_pridr0 bit assignments

Table B-420: CLUSTERROM_PRIDR0 bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	SYSRR	System reset request functionality present. **`0b0`** The system reset request functionality is not implemented.	`0b0`
[4]	DBGRR	Debug reset request functionality present. **`0b0`** The debug reset request functionality is not implemented.	`0b0`
[3:0]	VERSION	Version of the power request functionality. **`0b0001`** The power request functionality version 0, and the per-core controls for power requests (e.g. ext-CLUSTERROM_DBGPCR0 and ext-CLUSTERROM_DBGPSR0), are implemented.	`0b0001`

Accessibility

This interface is accessible as follows:

RO

B.2.2.52 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-267: ext_clusterrom_authstatus bit assignments

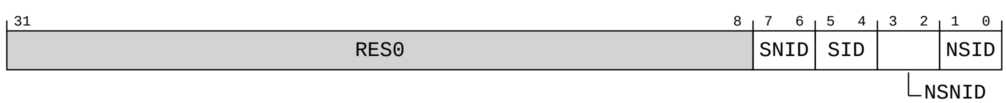


Table B-421: CLUSTERROM_AUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:6]	SNID	Secure Non-invasive Debug. **0b00** Debug level is not supported. ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled(). This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug. **0b00** Debug level is not supported.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug. **0b11** Supported and enabled.	0b11
[1:0]	NSID	Non-secure Invasive Debug. **0b00** Debug level is not supported.	0b00

Accessibility

This interface is accessible as follows:

RO

B.2.2.53 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-268: ext_clusterrom_devarch bit assignments

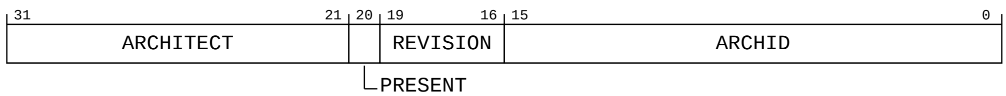


Table B-422: CLUSTERROM_DEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **`0b01000111011`** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	`0b01000111011`
[20]	PRESENT	Present. **`0b1`** DEVARCH information present.	`0b1`
[19:16]	REVISION	Revision. **`0b0000`** Revision 0.	`0b0000`
[15:0]	ARCHID	Architecture ID. **`0b0000101011110111`** ROM Table v0. The debug tool must inspect ext-CLUSTERROM_DEVTYPE and ext-CLUSTERROM_DEVID to determine further information about the ROM Table.	`0x0AF7`

Accessibility

This interface is accessible as follows:

RO

B.2.2.54 CLUSTERROM_DEVID, Cluster ROM table Device Configuration Register

Indicates the capabilities of the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFC8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xx10 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-269: ext_clusterrom_devid bit assignments

Table B-423: CLUSTERROM_DEVID bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	PRR	Power Request functionality included. **0b1** Power Request functionality included. ext-CLUSTERROM_PRIDR0 is implemented.	0b1
[4]	SYSMEM	System memory present. **0b0** System memory is not present on the bus.	0b0
[3:0]	FORMAT	ROM format. **0b0000** 32-bit format 0.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.2.55 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-270: ext_clusterrom_devtype bit assignments

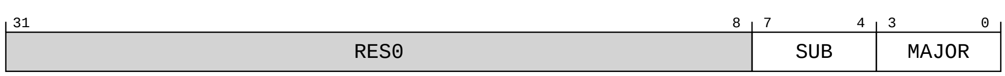


Table B-424: CLUSTERROM_DEVTYPE bit descriptions

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.2.2.56 CLUSTERROM_PIDR4, Cluster ROM table Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFD0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-271: ext_clusterrom_pidr4 bit assignments

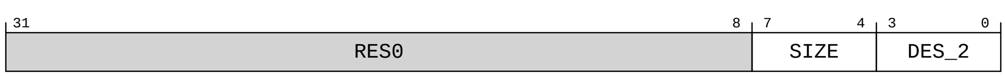


Table B-425: CLUSTERROM_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[7:4]	SIZE	4KB count. `0b0000` The component uses a single 4KB block.	`0b0000`
[3:0]	DES_2	JEP106 continuation code. `0b0100` Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	`0b0100`

Accessibility

This interface is accessible as follows:

RO

B.2.2.57 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 1110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-272: ext_clusterrom_pidr0 bit assignments

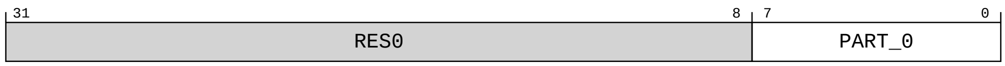


Table B-426: CLUSTERROM_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PART_0	Part number bits [7:0]. **`0b11101110`** DSU-C1 Cluster ROM table. Bits [7:0] of part number 0x4EE.	`0xEE`

Accessibility

This interface is accessible as follows:

RO

B.2.2.58 CLUSTERROM_PIDR1, Cluster ROM table Peripheral Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFE4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

> **Note**
>
> Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-273: ext_clusterrom_pidr1 bit assignments

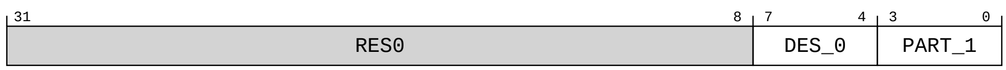


Table B-427: CLUSTERROM_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0100** DSU-C1 Cluster ROM table. Bits [11:8] of part number 0x4EE.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.2.2.59 CLUSTERROM_PIDR2, Cluster ROM table Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFE8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 1011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-274: ext_clusterrom_pidr2 bit assignments

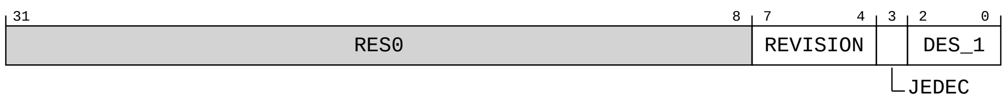


Table B-428: CLUSTERROM_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. For DSU-C1: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r0p1. • Major revision 2 corresponds to r0p2.	0b0010
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

Accessibility

This interface is accessible as follows:

RO

B.2.2.60 CLUSTERROM_PIDR3, Cluster ROM table Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFEC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-275: ext_clusterrom_pidr3 bit assignments

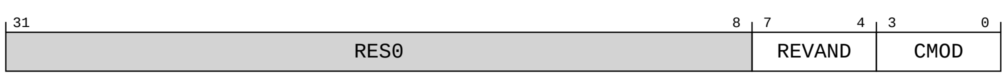


Table B-429: CLUSTERROM_PIDR3 bit descriptions

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. **0b0000** The component is not modified from the original design.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.2.61 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-276: ext_clusterrom_cidr0 bit assignments

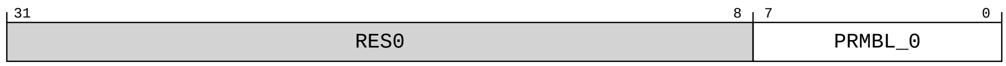


Table B-430: CLUSTERROM_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

B.2.2.62 CLUSTERROM_CIDR1, Cluster ROM table Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFF4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-277: ext_clusterrom_cidr1 bit assignments

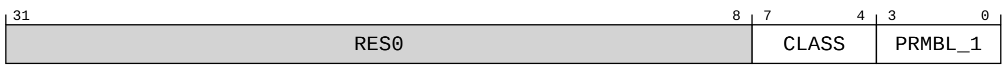


Table B-431: CLUSTERROM_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	CLASS	CoreSight component class. **`0b1001`** CoreSight component.	`0b1001`
[3:0]	PRMBL_1	CoreSight component identification preamble. **`0b0000`** CoreSight component identification preamble.	`0b0000`

Accessibility

This interface is accessible as follows:

RO

B.2.2.63 CLUSTERROM_CIDR2, Cluster ROM table Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFF8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-278: ext_clusterrom_cidr2 bit assignments

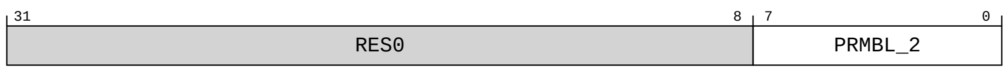


Table B-432: CLUSTERROM_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_2	CoreSight component identification preamble. **0b00000101** CoreSight component identification preamble.	0x05

Accessibility

This interface is accessible as follows:

RO

B.2.2.64 CLUSTERROM_CIDR3, Cluster ROM table Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

CLUSTERROM

Register offset

0xFFC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-279: ext_clusterrom_cidr3 bit assignments

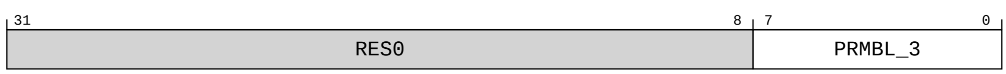


Table B-433: CLUSTERROM_CIDR3 bit descriptions

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, implemented with the following C1-DSU implementation parameters:
    - DIRECT_CONNECT is set to FALSE.
    - NUM_CORES 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**.
- If the C1-DSU is configured for Direct connect, all these registers are present.
- If the C1-DSU is enabled for Realm Management Extension (RME) all these registers are present.
- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-434: DBROM registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x000	DBROM_ROMENTRY0	See individual bit resets.	32-bit	DebugBlock ROM table Entry 0	Yes
0x4	DBROM_ROMENTRY1	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x8	DBROM_ROMENTRY2	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0xC	DBROM_ROMENTRY3	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x10	DBROM_ROMENTRY4	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x14	DBROM_ROMENTRY5	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x18	DBROM_ROMENTRY6	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x1C	DBROM_ROMENTRY7	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x20	DBROM_ROMENTRY8	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x24	DBROM_ROMENTRY9	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x28	DBROM_ROMENTRY10	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x2C	DBROM_ROMENTRY11	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x30	DBROM_ROMENTRY12	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x34	DBROM_ROMENTRY13	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes

Offset	Name	Reset	Width	Description	Present in Direct connect
0x38	DBROM_ROMENTRY14	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x3C	DBROM_ROMENTRY15	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x40	DBROM_ROMENTRY16	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0x44	DBROM_ROMENTRY17	See individual bit resets.	32-bit	DebugBlock ROM table Entry	Yes
0xA00	DBROM_DBGPCR0	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Control Register 0	Yes
0xA80	DBROM_DBGPSR0	See individual bit resets.	32-bit	DebugBlock ROM table Debug Power Status Register 0	Yes
0xC00	DBROM_PRIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Power Request ID Register 0	Yes
0xFB8	DBROM_AUTHSTATUS	See individual bit resets.	32-bit	DebugBlock ROM table Authentication Status Register	Yes
0xFBC	DBROM_DEVARCH	See individual bit resets.	32-bit	DebugBlock ROM table Device Architecture Register	Yes
0xFC8	DBROM_DEVID	See individual bit resets.	32-bit	DebugBlock ROM table Device Configuration Register	Yes
0xFCC	DBROM_DEVTYPE	See individual bit resets.	32-bit	DebugBlock ROM table Device Type Register	Yes
0xFD0	DBROM_PIDR4	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 4	Yes
0xFE0	DBROM_PIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 0	Yes
0xFE4	DBROM_PIDR1	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 1	Yes
0xFE8	DBROM_PIDR2	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 2	Yes
0xFEC	DBROM_PIDR3	See individual bit resets.	32-bit	DebugBlock ROM table Peripheral Identification Register 3	Yes
0xFF0	DBROM_CIDR0	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 0	Yes
0xFF4	DBROM_CIDR1	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 1	Yes
0xFF8	DBROM_CIDR2	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 2	Yes
0xFFC	DBROM_CIDR3	See individual bit resets.	32-bit	DebugBlock ROM table Component Identification Register 3	Yes

B.2.3.1 DBROM_ROMENTRY0, DebugBlock ROM table Entry 0

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x000

Access type

RO

Reset value

```
0000 0000 0000 0001 0000 xxx0 0000 x111
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-280: ext_dbrom_romentry0 bit assignments

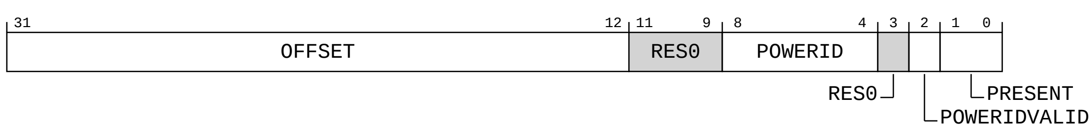


Table B-435: DBROM_ROMENTRY0 bit descriptions

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). **0b00000000000000010000** Cluster ROM table at address 0x0001_0000	0x00010
[11:9]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[8:4]	POWERID	The power domain ID of the component. This field is only valid if the POWERIDVALID field is `0b1`. The value of this field depends on the cluster configuration. **`0b00000`** PDCLUSTER power domain.	`0b00000`
[3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. The value of this field depends on the cluster configuration. **`0b1`** The POWERID field provides a power domain ID.	`0b1`
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **`0b11`** The ROM Entry is present.	`0b11`

Accessibility

This interface is accessible as follows:

RO

B.2.3.2 DBROM_ROMENTRY1, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x4

Access type

RO

Reset value

```
0000 0000 0000 0100 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-281: ext_dbrom_romentry1 bit assignments

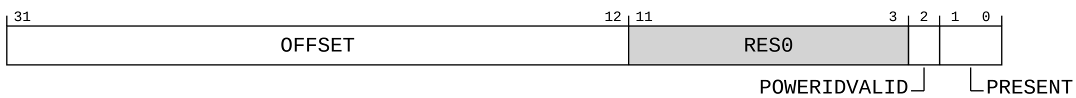


Table B-436: DBROM_ROMENTRY1 bit descriptions

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). **0b00000000000001000000** Cluster CTI at address 0x0004_0000	`0x00040`
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	`0b0`
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	`0b11`

Accessibility

This interface is accessible as follows:

RO

B.2.3.3 DBROM_ROMENTRY2, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x8

Access type

RO

Reset value

```
0000 0000 0000 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-282: ext_dbrom_romentry2 bit assignments

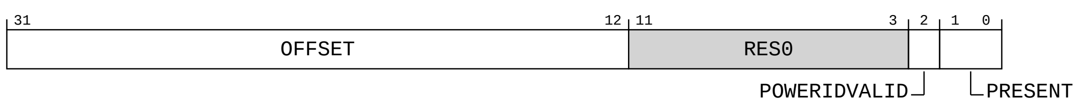


Table B-437: DBROM_ROMENTRY2 bit descriptions

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). **0b00000000000011110000** Core or SME2 unit CTI at address 0x000F_0000	0x000F0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.4 DBROM_ROMENTRY3, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xC

Access type

RO

Reset value

```
0000 0000 0001 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-283: ext_dbrom_romentry3 bit assignments

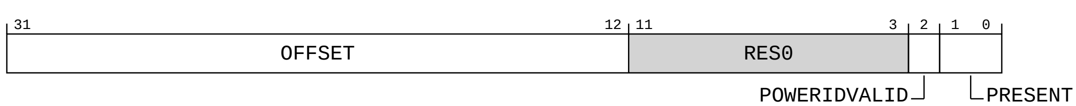


Table B-438: DBROM_ROMENTRY3 bit descriptions

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). **0b00000000000101110000** Core or SME2 unit CTI at address 0x0017_0000	0x00170
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.5 DBROM_ROMENTRY4, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x10

Access type

RO

Reset value

```
0000 0000 0001 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-284: ext_dbrom_romentry4 bit assignments

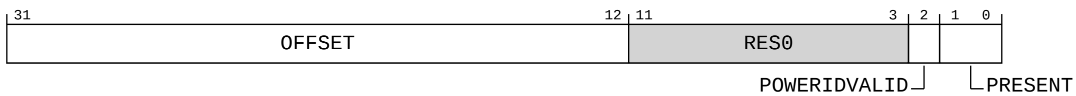


Table B-439: DBROM_ROMENTRY4 bit descriptions

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). **0b00000000000111110000** Core or SME2 unit CTI at address 0x001F_0000	0x001F0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.6 DBROM_ROMENTRY5, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x14

Access type

RO

Reset value

```
0000 0000 0010 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-285: ext_dbrom_romentry5 bit assignments

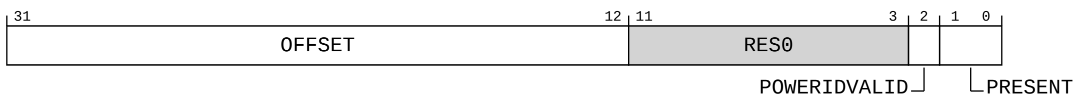


Table B-440: DBROM_ROMENTRY5 bit descriptions

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). **0b00000000001001110000** Core or SME2 unit CTI at address 0x0027_0000	0x00270
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.7 DBROM_ROMENTRY6, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x18

Access type

RO

Reset value

```
0000 0000 0010 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-286: ext_dbrom_romentry6 bit assignments

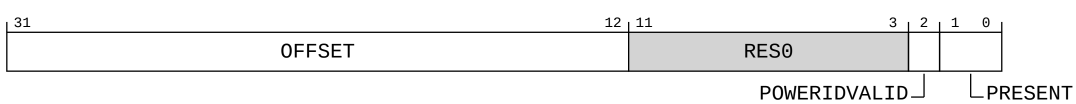


Table B-441: DBROM_ROMENTRY6 bit descriptions

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). **0b00000000001011110000** Core or SME2 unit CTI at address 0x002F_0000	0x002F0
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.8 DBROM_ROMENTRY7, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x1C

Access type

RO

Reset value

```
0000 0000 0011 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-287: ext_dbrom_romentry7 bit assignments

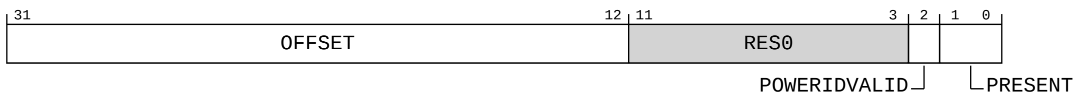


Table B-442: DBROM_ROMENTRY7 bit descriptions

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). **0b00000000001101110000** Core or SME2 unit CTI at address 0x0037_0000	0x00370
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.9 DBROM_ROMENTRY8, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x20

Access type

RO

Reset value

```
0000 0000 0011 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-288: ext_dbrom_romentry8 bit assignments

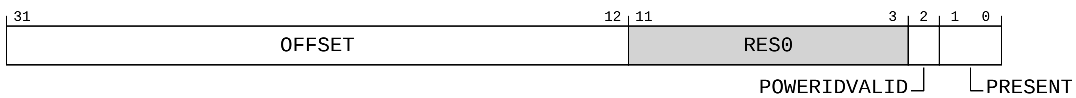


Table B-443: DBROM_ROMENTRY8 bit descriptions

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). **0b00000000001111110000** Core or SME2 unit CTI at address 0x003F_0000	0x003F0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.10 DBROM_ROMENTRY9, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x24

Access type

RO

Reset value

```
0000 0000 0100 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-289: ext_dbrom_romentry9 bit assignments

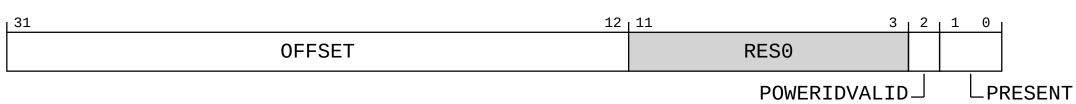


Table B-444: DBROM_ROMENTRY9 bit descriptions

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). **0b00000000010001110000** Core or SME2 unit CTI at address 0x0047_0000	0x00470
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.11 DBROM_ROMENTRY10, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x28

Access type

RO

Reset value

```
0000 0000 0100 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-290: ext_dbrom_romentry10 bit assignments

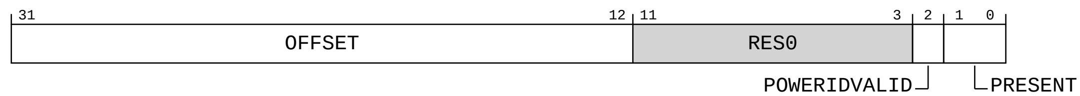


Table B-445: DBROM_ROMENTRY10 bit descriptions

Bits	Name	Description	Reset
[31:12]	OFFSET	The component address, relative to the base address of this ROM Table. The component address is calculated using the following equation: Component Address = ROM Table Base Address + (OFFSET << 12). **0b00000000010011110000** Core or SME2 unit CTI at address 0x004F_0000	0x004F0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.12 DBROM_ROMENTRY11, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x2C

Access type

RO

Reset value

```
0000 0000 0101 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-291: ext_dbrom_romentry11 bit assignments

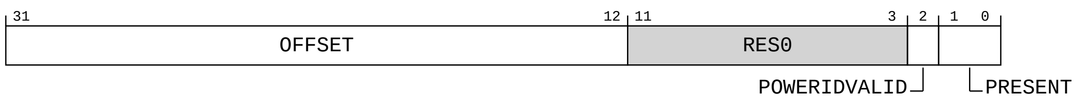


Table B-446: DBROM_ROMENTRY11 bit descriptions

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). **0b00000000010101110000** Core or SME2 unit CTI at address 0x0057_0000	0x00570
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.13 DBROM_ROMENTRY12, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x30

Access type

RO

Reset value

```
0000 0000 0101 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-292: ext_dbrom_romentry12 bit assignments

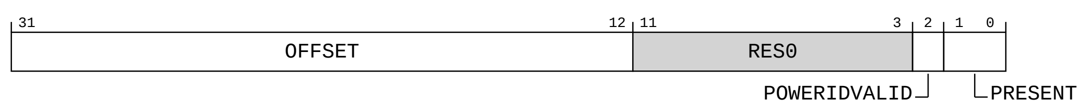


Table B-447: DBROM_ROMENTRY12 bit descriptions

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). **0b00000000010111110000** Core or SME2 unit CTI at address 0x005F_0000	0x005F0
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.14 DBROM_ROMENTRY13, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x34

Access type

RO

Reset value

```
0000 0000 0110 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-293: ext_dbrom_romentry13 bit assignments

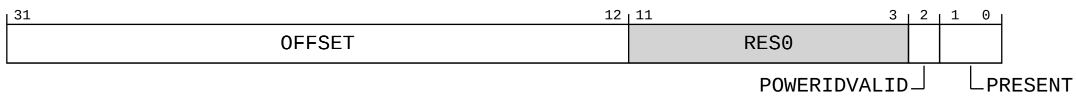


Table B-448: DBROM_ROMENTRY13 bit descriptions

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). **0b00000000011001110000** Core or SME2 unit CTI at address 0x0067_0000	0x00670
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.15 DBROM_ROMENTRY14, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x38

Access type

RO

Reset value

```
0000 0000 0110 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-294: ext_dbrom_romentry14 bit assignments

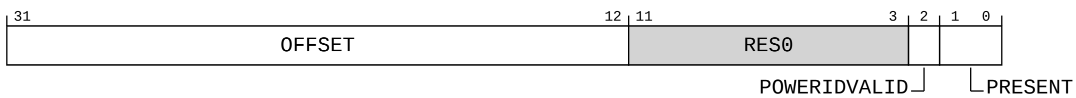


Table B-449: DBROM_ROMENTRY14 bit descriptions

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). **0b00000000011011110000** Core or SME2 unit CTI at address 0x006F_0000	0x006F0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.16 DBROM_ROMENTRY15, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x3C

Access type

RO

Reset value

```
0000 0000 0111 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-295: ext_dbrom_romentry15 bit assignments

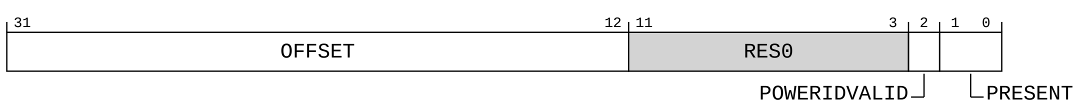


Table B-450: DBROM_ROMENTRY15 bit descriptions

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). **0b00000000011101110000** Core or SME2 unit CTI at address 0x0077_0000	0x00770
[11:3]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.17 DBROM_ROMENTRY16, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x40

Access type

RO

Reset value

```
0000 0000 0111 1111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-296: ext_dbrom_romentry16 bit assignments

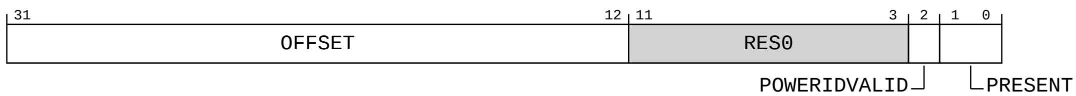


Table B-451: DBROM_ROMENTRY16 bit descriptions

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). **0b00000000011111110000** Core or SME2 unit CTI at address 0x007F_0000	0x007F0
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.18 DBROM_ROMENTRY17, DebugBlock ROM table Entry

Provides the address offset for one CoreSight component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0x44

Access type

RO

Reset value

```
0000 0000 1000 0111 0000 xxxx xxxx x011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-297: ext_dbrom_romentry17 bit assignments

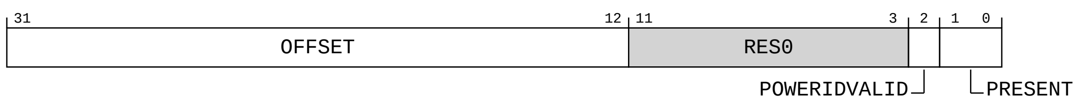


Table B-452: DBROM_ROMENTRY17 bit descriptions

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). **0b00000000100001110000** Core or SME2 unit CTI at address 0x0087_0000	0x00870
[11:3]	**RES0**	Reserved	**RES0**
[2]	POWERIDVALID	Indicates if the Power domain ID field contains a Power domain ID. **0b0** A power domain ID is not provided.	0b0
[1:0]	PRESENT	Indicates whether an entry is present at this location in the ROM Table. **0b11** The ROM Entry is present.	0b11

Accessibility

This interface is accessible as follows:

RO

B.2.3.19 DBROM_DBGPCR0, DebugBlock ROM table Debug Power Control Register 0

Controls power requests for PDCLUSTER.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xA00

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-298: ext_dbrom_dbgpcr0 bit assignments

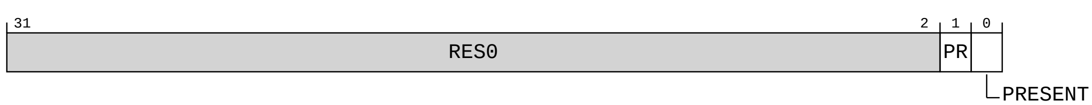


Table B-453: DBROM_DBGPCR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	RES0	Reserved	RES0
[1]	PR	Power Request. **0b0** Power is not requested for PDCLUSTER. **0b1** Power is requested for PDCLUSTER.	x

Bits	Name	Description	Reset
[0]	PRESENT	Power request implemented. `0b1` Power request for PDCLUSTER is implemented.	x

Accessibility

This interface is accessible as follows:

RW

B.2.3.20 DBROM_DBGPSR0, DebugBlock ROM table Debug Power Status Register 0

Indicates the power status for PDCLUSTER.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xA80

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xx00
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-299: ext_dbrom_dbgpsr0 bit assignments

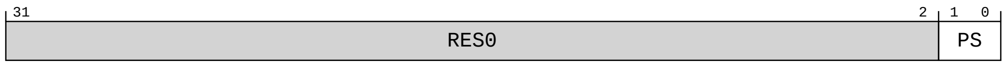


Table B-454: DBROM_DBGPSR0 bit descriptions

Bits	Name	Description	Reset
[31:2]	**RES0**	Reserved	**RES0**
[1:0]	PS	Power Status. **`0b00`** PDCLUSTER might not be powered. **`0b01`** PDCLUSTER is powered.	`0b00`

Accessibility

This interface is accessible as follows:

RO

B.2.3.21 DBROM_PRIDR0, DebugBlock ROM table Power Request ID Register 0

Indicates the features of the power request functionality.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xC00

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xx00 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-300: ext_dbrom_pridr0 bit assignments

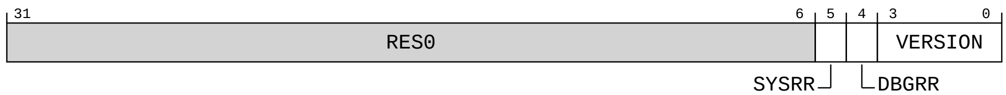


Table B-455: DBROM_PRIDR0 bit descriptions

Bits	Name	Description	Reset
[31:6]	**RES0**	Reserved	**RES0**
[5]	SYSRR	System reset request functionality present. **`0b0`** The system reset request functionality is not implemented.	`0b0`
[4]	DBGRR	Debug reset request functionality present. **`0b0`** The debug reset request functionality is not implemented.	`0b0`
[3:0]	VERSION	Version of the power request functionality. **`0b0001`** The power request functionality version 0, and the ext-DBROM_DBGPCR0, ext-DBROM_DBGPSR0, which provide controls for power requests, are implemented.	`0b0001`

Accessibility

This interface is accessible as follows:

RO

B.2.3.22 DBROM_AUTHSTATUS, DebugBlock ROM table Authentication Status Register

Provides information about the state of the authentication interface for debug.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFB8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 1100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-301: ext_dbrom_authstatus bit assignments

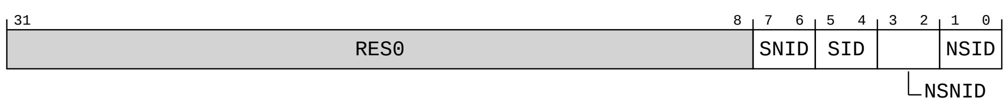


Table B-456: DBROM_AUTHSTATUS bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:6]	SNID	Secure Non-invasive Debug. **0b00** Debug level is not supported. ExternalSecureNoninvasiveDebugEnabled() == ExternalSecureInvasiveDebugEnabled(). This field has the same value as the SID field.	0b00
[5:4]	SID	Secure Invasive Debug. **0b00** Debug level is not supported.	0b00
[3:2]	NSNID	Non-secure Non-invasive Debug. **0b11** Supported and enabled.	0b11
[1:0]	NSID	Non-secure Invasive Debug. **0b00** Debug level is not supported.	0b00

Accessibility

This interface is accessible as follows:

RO

B.2.3.23 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-302: ext_dbrom_devarch bit assignments

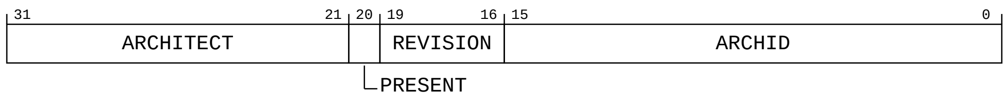


Table B-457: DBROM_DEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **`0b01000111011`** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	`0b01000111011`
[20]	PRESENT	Present. **`0b1`** DEVARCH information present.	`0b1`

Bits	Name	Description	Reset
[19:16]	REVISION	Revision. **0b0000** Revision 0.	`0b0000`
[15:0]	ARCHID	Architecture ID. **0b0000101011110111** ROM Table v0. The debug tool must inspect ext-DBROM_DEVTYPE and ext-DBROM_DEVID to determine further information about the ROM Table.	`0x0AF7`

Accessibility

This interface is accessible as follows:

RO

B.2.3.24 DBROM_DEVID, DebugBlock ROM table Device Configuration Register

Indicates the capabilities of the component.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFC8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xx10 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-303: ext_dbrom_devid bit assignments

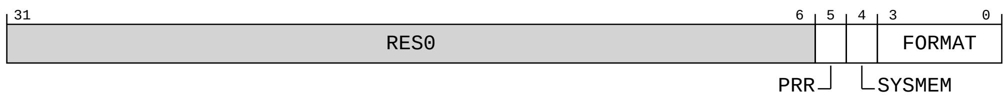


Table B-458: DBROM_DEVID bit descriptions

Bits	Name	Description	Reset
[31:6]	RES0	Reserved	RES0
[5]	PRR	Power Request functionality included. **0b1** Power Request functionality included. ext-DBROM_PRIDR0 is implemented.	0b1
[4]	SYSMEM	System memory present. **0b0** System memory is not present on the bus.	0b0
[3:0]	FORMAT	ROM format. **0b0000** 32-bit format 0.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.3.25 DBROM_DEVTYPE, DebugBlock ROM table Device Type Register

A debugger can use DEVTYPE to obtain information about a component that has an unrecognized part number.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFCC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-304: ext_dbrom_devtype bit assignments

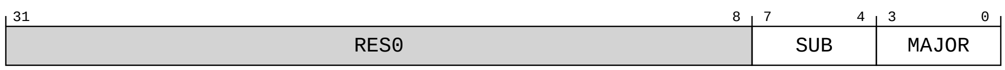


Table B-459: DBROM_DEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SUB	Sub number **0b0000** Other, undefined.	0b0000
[3:0]	MAJOR	Major number **0b0000** Miscellaneous.	0b0000

Accessibility

This interface is accessible as follows:

RO

B.2.3.26 DBROM_PIDR4, DebugBlock ROM table Peripheral Identification Register 4

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFD0

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-305: ext_dbrom_pidr4 bit assignments

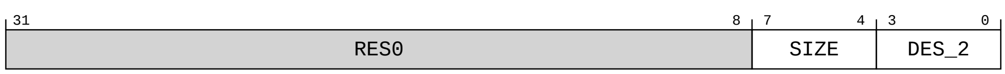


Table B-460: DBROM_PIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	SIZE	4KB count. **0b0000** The component uses a single 4KB block.	0b0000
[3:0]	DES_2	JEP106 continuation code. **0b0100** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	0b0100

Accessibility

This interface is accessible as follows:

RO

B.2.3.27 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 1101
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-306: ext_dbrom_pidr0 bit assignments

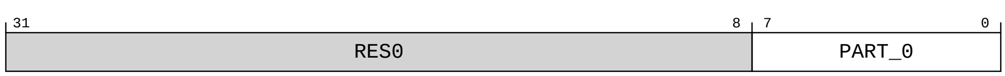


Table B-461: DBROM_PIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. **0b11101011** DSU-C1 DebugBlock ROM table. Bits [7:0] of part number 0x4ED.	0xED

Accessibility

This interface is accessible as follows:

RO

B.2.3.28 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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-307: ext_dbrom_pidr1 bit assignments

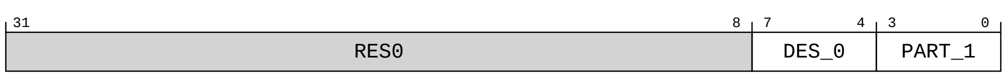


Table B-462: DBROM_PIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	DES_0	JEP106 identification code bits [3:0]. **`0b1011`** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	`0b1011`

Bits	Name	Description	Reset
[3:0]	PART_1	Part number bits [11:8]. **`0b0100`** DSU-C1 DebugBlock ROM table. Bits [11:8] of part number 0x4ED.	`0b0100`

Accessibility

This interface is accessible as follows:

RO

B.2.3.29 DBROM_PIDR2, DebugBlock ROM table Peripheral Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFE8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 1011
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-308: ext_dbrom_pidr2 bit assignments

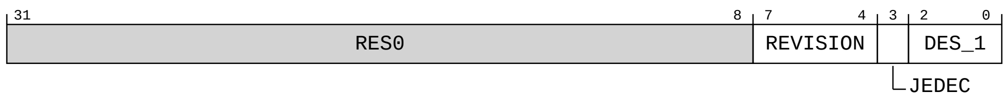


Table B-463: DBROM_PIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. **`0b0000`** Component major revision 0. **`0b0001`** Component major revision 1. **`0b0010`** Component major revision 2. For DSU-C1: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r0p1. • Major revision 2 corresponds to r0p2.	`0b0010`
[3]	JEDEC	JEDEC assignee. **`0b1`** JEDEC-assignee values is used.	`0b1`
[2:0]	DES_1	JEP106 identification code bits [6:4]. **`0b011`** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	`0b011`

Accessibility

This interface is accessible as follows:

RO

B.2.3.30 DBROM_PIDR3, DebugBlock ROM table Peripheral Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFEC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-309: ext_dbrom_pidr3 bit assignments

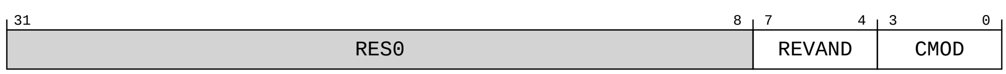


Table B-464: DBROM_PIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0.	0b0000
[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.2.3.31 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-310: ext_dbrom_cidr0 bit assignments

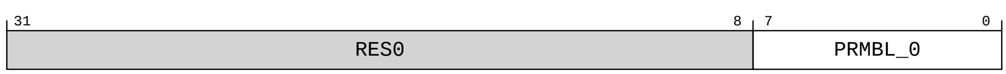


Table B-465: DBROM_CIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PRMBL_0	CoreSight component identification preamble. **0b00001101** CoreSight component identification preamble.	0x0D

Accessibility

This interface is accessible as follows:

RO

B.2.3.32 DBROM_CIDR1, DebugBlock ROM table Component Identification Register 1

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFF4

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-311: ext_dbrom_cidr1 bit assignments

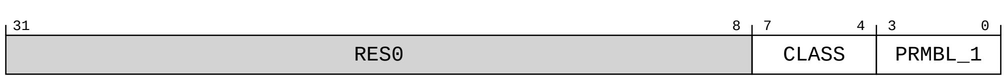


Table B-466: DBROM_CIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	CLASS	CoreSight component class. **0b1001** CoreSight component.	0b1001

Bits	Name	Description	Reset
[3:0]	PRMBL_1	CoreSight component identification preamble. **`0b0000`** CoreSight component identification preamble.	`0b0000`

Accessibility

This interface is accessible as follows:

RO

B.2.3.33 DBROM_CIDR2, DebugBlock ROM table Component Identification Register 2

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFF8

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0000 0101
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-312: ext_dbrom_cidr2 bit assignments

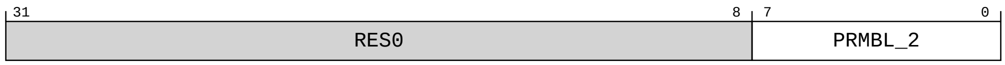


Table B-467: DBROM_CIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	CoreSight component identification preamble. **`0b00000101`** CoreSight component identification preamble.	`0x05`

Accessibility

This interface is accessible as follows:

RO

B.2.3.34 DBROM_CIDR3, DebugBlock ROM table Component Identification Register 3

Provides CoreSight discovery information.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

DBROM

Register offset

0xFFC

Access type

RO

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-313: ext_dbrom_cidr3 bit assignments

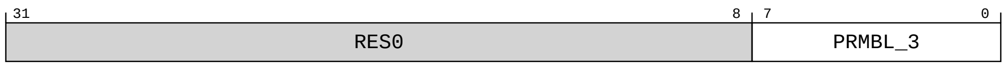


Table B-468: DBROM_CIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	CoreSight component identification preamble. **`0b10110001`** CoreSight component identification preamble.	`0xB1`

Accessibility

This interface is accessible as follows:

RO

B.2.4 External cluster performance-monitors 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**.
- If the C1-DSU is configured for Direct connect, none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- If the C1-DSU is enabled for Realm Management Extension (RME), none of these registers are present, and any access to these registers are treated as **RAZ/WI**.
- The part number is `0x4EE`.

- For registers without a listed reset value refer to the individual field resets documented on the register description pages.

Table B-469: performance-monitors registers summary

Offset	Name	Reset	Width	Description	Present in Direct connect
0x0	CLUSTERPMU_PMEVCNTR0	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x8	CLUSTERPMU_PMEVCNTR1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x10	CLUSTERPMU_PMEVCNTR2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x18	CLUSTERPMU_PMEVCNTR3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x20	CLUSTERPMU_PMEVCNTR4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x28	CLUSTERPMU_PMEVCNTR5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Registers	No
0x400	CLUSTERPMU_PMEVTYPER0	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x404	CLUSTERPMU_PMEVTYPER1	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x408	CLUSTERPMU_PMEVTYPER2	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x40C	CLUSTERPMU_PMEVTYPER3	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x410	CLUSTERPMU_PMEVTYPER4	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x414	CLUSTERPMU_PMEVTYPER5	See individual bit resets.	32-bit	Cluster Performance Monitors Event Type Registers	No
0x600	CLUSTERPMU_PMEVCNTSR0	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x608	CLUSTERPMU_PMEVCNTSR1	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x610	CLUSTERPMU_PMEVCNTSR2	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x618	CLUSTERPMU_PMEVCNTSR3	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x620	CLUSTERPMU_PMEVCNTSR4	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x628	CLUSTERPMU_PMEVCNTSR5	See individual bit resets.	64-bit	Cluster Performance Monitors Event Count Snapshot Registers	No
0x638	CLUSTERPMU_PMSSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Status register	No
0x640	CLUSTERPMU_PMOVSSR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Status Snapshot register	No
0xC00	CLUSTERPMU_PMCNTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Set register	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xC20	CLUSTERPMU_PMCNTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Count Enable Clear register	No
0xC40	CLUSTERPMU_PMINTENSET	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Set register	No
0xC60	CLUSTERPMU_PMINTENCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Interrupt Enable Clear register	No
0xC80	CLUSTERPMU_PMOVSCLR	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Clear register	No
0xCC0	CLUSTERPMU_PMOVSSET	See individual bit resets.	32-bit	Cluster Performance Monitors Overflow Flag Status Set register	No
0xE00	CLUSTERPMU_PMCFGR	See individual bit resets.	32-bit	Cluster Performance Monitors Configuration Register	No
0xE04	CLUSTERPMU_PMCR	See individual bit resets.	32-bit	Cluster Performance Monitors Control Register	No
0xE08	CLUSTERPMU_PMIIDR	See individual bit resets.	32-bit	Cluster Performance Monitors Implementation Identification register	No
0xE20	CLUSTERPMU_PMCEID0	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 0	No
0xE24	CLUSTERPMU_PMCEID1	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 1	No
0xE28	CLUSTERPMU_PMCEID2	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 2	No
0xE2C	CLUSTERPMU_PMCEID3	See individual bit resets.	32-bit	Cluster Performance Monitors Common Event Identification register 3	No
0xE30	CLUSTERPMU_PMSSCR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Capture register	No
0xE38	CLUSTERPMU_PMSSRR	See individual bit resets.	32-bit	Cluster Performance Monitors Snapshot Reset register	No
0xFA8	CLUSTERPMU_PMDEVAFF0	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 0	No
0xFAC	CLUSTERPMU_PMDEVAFF1	See individual bit resets.	32-bit	Cluster Performance Monitors Device Affinity register 1	No
0xFB8	CLUSTERPMU_PMAUTHSTATUS	See individual bit resets.	32-bit	Cluster Performance Monitors Authentication Status register	No
0xFBC	CLUSTERPMU_PMDEVARCH	See individual bit resets.	32-bit	Cluster Performance Monitors Device Architecture register	No
0xFC8	CLUSTERPMU_PMDEVID	See individual bit resets.	32-bit	Cluster Performance Monitors Device ID register	No
0xFCC	CLUSTERPMU_PMDEVTYPE	See individual bit resets.	32-bit	Cluster Performance Monitors Device Type register	No
0xFD0	CLUSTERPMU_PMPIDR4	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 4	No
0xFE0	CLUSTERPMU_PMPIDR0	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 0	No
0xFE4	CLUSTERPMU_PMPIDR1	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 1	No

Offset	Name	Reset	Width	Description	Present in Direct connect
0xFE8	CLUSTERPMU_PMPIDR2	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 2	No
0xFEC	CLUSTERPMU_PMPIDR3	See individual bit resets.	32-bit	Cluster Performance Monitors Peripheral Identification Register 3	No
0xFF0	CLUSTERPMU_PMCIDR0	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 0	No
0xFF4	CLUSTERPMU_PMCIDR1	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 1	No
0xFF8	CLUSTERPMU_PMCIDR2	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 2	No
0xFFC	CLUSTERPMU_PMCIDR3	See individual bit resets.	32-bit	Cluster Performance Monitors Component Identification Register 3	No

B.2.4.1 CLUSTERPMU_PMEVCNTR0, Cluster Performance Monitors Event Count Registers

Holds event counter 0, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-314: ext_clusterpmu_pmevcntr0 bit assignments

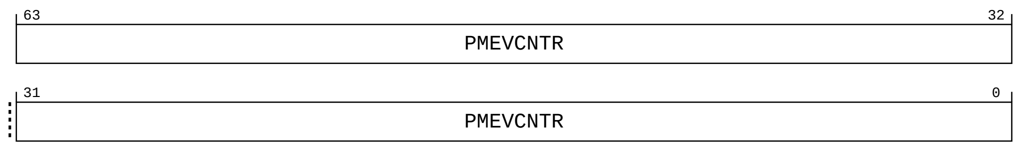


Table B-470: CLUSTERPMU_PMEVCNTR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 0.	`64{x}`

B.2.4.2 CLUSTERPMU_PMEVCNTR1, Cluster Performance Monitors Event Count Registers

Holds event counter 1, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-315: ext_clusterpmu_pmevcntr1 bit assignments

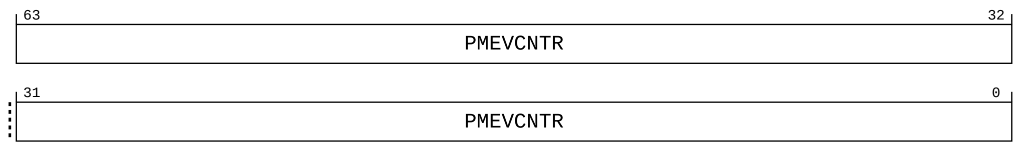


Table B-471: CLUSTERPMU_PMEVCNTR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 1.	`64{x}`

B.2.4.3 CLUSTERPMU_PMEVCNTR2, Cluster Performance Monitors Event Count Registers

Holds event counter 2, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

Register offset

0x10

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-316: ext_clusterpmu_pmevcntr2 bit assignments

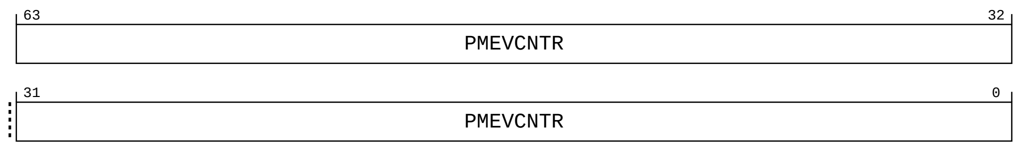


Table B-472: CLUSTERPMU_PMEVCNTR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 2.	`64{x}`

B.2.4.4 CLUSTERPMU_PMEVCNTR3, Cluster Performance Monitors Event Count Registers

Holds event counter 3, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-317: ext_clusterpmu_pmevcntr3 bit assignments

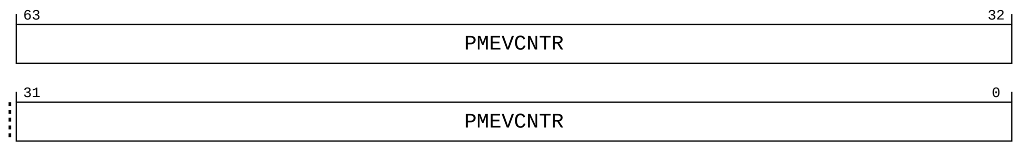


Table B-473: CLUSTERPMU_PMEVCNTR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 3.	64{x}

B.2.4.5 CLUSTERPMU_PMEVCNTR4, Cluster Performance Monitors Event Count Registers

Holds event counter 4, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-318: ext_clusterpmu_pmevcntr4 bit assignments

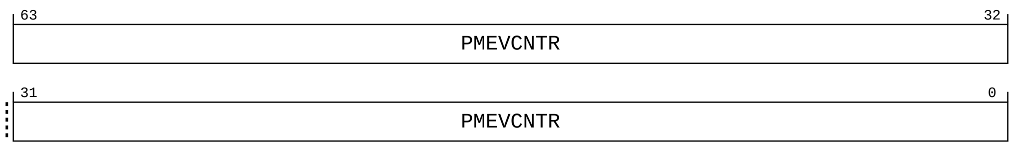


Table B-474: CLUSTERPMU_PMEVCNTR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 4.	`64{x}`

B.2.4.6 CLUSTERPMU_PMEVCNTR5, Cluster Performance Monitors Event Count Registers

Holds event counter 5, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

Register offset

0x28

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-319: ext_clusterpmu_pmevcntr5 bit assignments

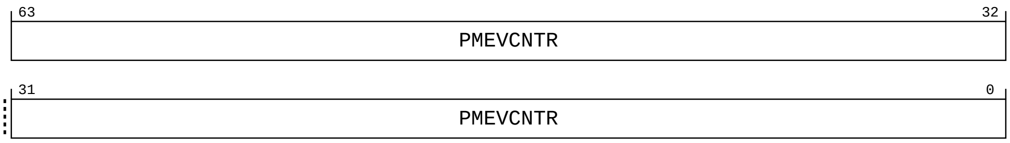


Table B-475: CLUSTERPMU_PMEVCNTR5 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTR	Event counter 5.	`64{x}`

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

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-320: ext_clusterpmu_pmevtyper0 bit assignments

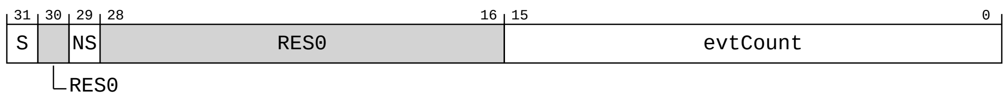


Table B-476: CLUSTERPMU_PMEVTYPER0 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **`0b0`** Count Secure events. **`0b1`** Do not count Secure events.	`x`
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	`x`
[28:16]	**RES0**	Reserved	**RES0**
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	`16{x}`

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

performance-monitors

Register offset

0x404

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-321: ext_clusterpmu_pmevtyper1 bit assignments

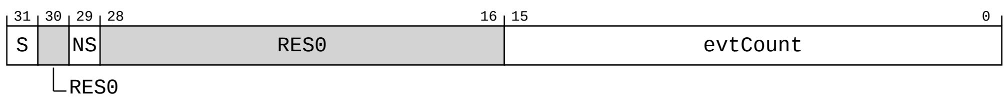


Table B-477: CLUSTERPMU_PMEVTYPER1 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[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}

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

performance-monitors

Register offset

0x408

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-322: ext_clusterpmu_pmevtyper2 bit assignments

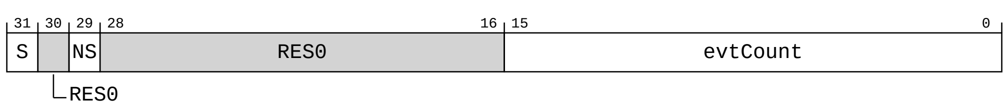


Table B-478: CLUSTERPMU_PMEVTYPER2 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	RES0	Reserved	RES0

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}

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

performance-monitors

Register offset

0x40C

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-323: ext_clusterpmu_pmevtyper3 bit assignments

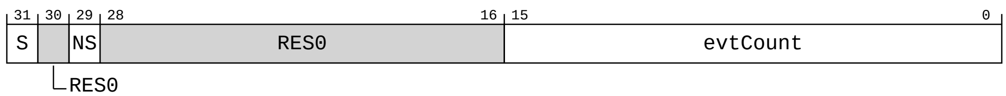


Table B-479: CLUSTERPMU_PMEVTYPER3 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[15:0]	evtCount	Event to count. The event number of the event that is counted by event counter ext-CLUSTERPMU_PMEVCNTR<n>. Software must program this field with an event that is supported by the Cluster.	16{x}

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

performance-monitors

Register offset

0x410

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-324: ext_clusterpmu_pmevtyper4 bit assignments

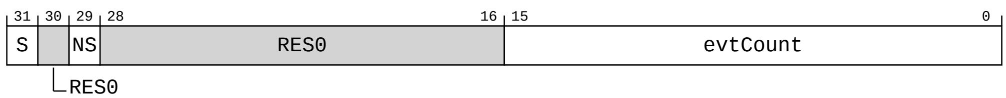


Table B-480: CLUSTERPMU_PMEVTYPER4 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	**RES0**	Reserved	**RES0**
[29]	NS	Non-secure events filtering bit. Controls counting of events generated by Non-secure transactions. Possible values are: NS == S If the value of this bit equals the value of the P bit then count Non-secure events. NS != S If the value of this bit does not equal the value of the P bit then do not count Non-secure events.	x
[28:16]	**RES0**	Reserved	**RES0**
[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}

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

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-325: ext_clusterpmu_pmevtyper5 bit assignments

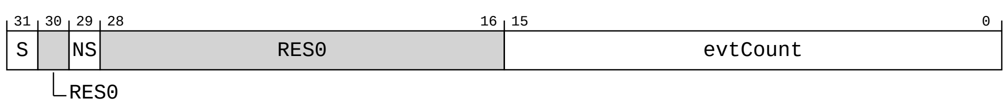


Table B-481: CLUSTERPMU_PMEVTYPER5 bit descriptions

Bits	Name	Description	Reset
[31]	S	Secure events filtering bit. Controls counting of events that are generated by Secure transactions. **0b0** Count Secure events. **0b1** Do not count Secure events.	x
[30]	RES0	Reserved	RES0

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}

B.2.4.13 CLUSTERPMU_PMEVCNTSR0, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 0, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-326: ext_clusterpmu_pmevcntsr0 bit assignments

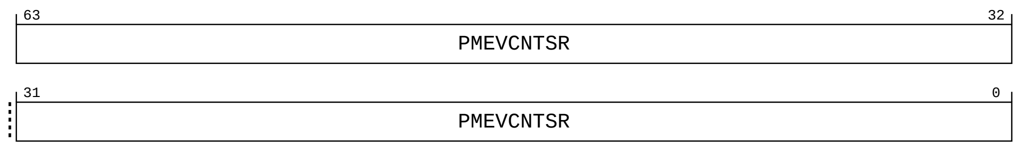


Table B-482: CLUSTERPMU_PMEVCNTSR0 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 0.	64{x}

B.2.4.14 CLUSTERPMU_PMEVCNTSR1, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 1, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-327: ext_clusterpmu_pmevcntsr1 bit assignments

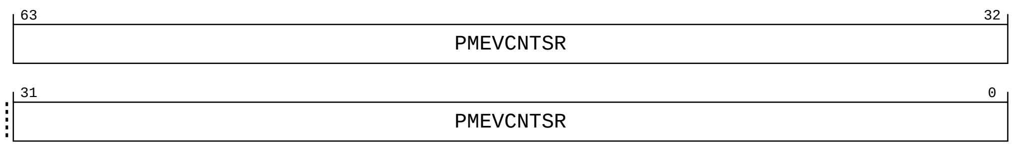


Table B-483: CLUSTERPMU_PMEVCNTSR1 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 1.	`64{x}`

B.2.4.15 CLUSTERPMU_PMEVCNTSR2, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 2, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-328: ext_clusterpmu_pmevcntsr2 bit assignments

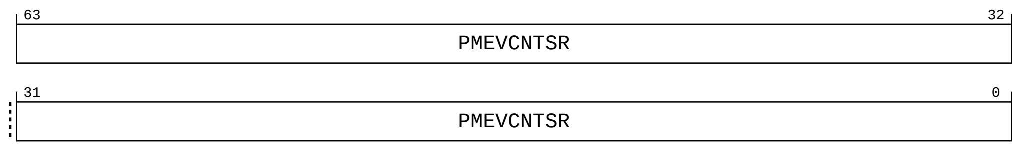


Table B-484: CLUSTERPMU_PMEVCNTSR2 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 2.	`64{x}`

B.2.4.16 CLUSTERPMU_PMEVCNTSR3, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 3, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

Register offset

0x618

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 B-329: ext_clusterpmu_pmevcntsr3 bit assignments

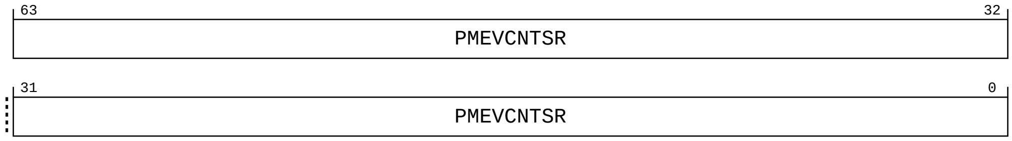


Table B-485: CLUSTERPMU_PMEVCNTSR3 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 3.	`64{x}`

B.2.4.17 CLUSTERPMU_PMEVCNTSR4, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 4, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

Register offset

0x620

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |  |
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-330: ext_clusterpmu_pmevcntsr4 bit assignments

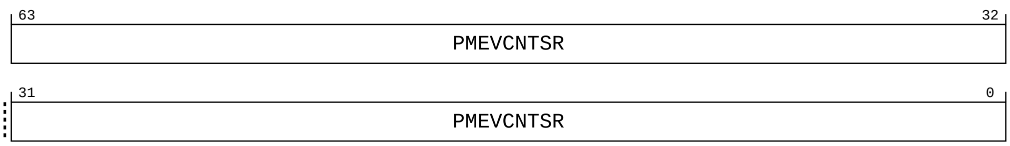


Table B-486: CLUSTERPMU_PMEVCNTSR4 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 4.	`64{x}`

B.2.4.18 CLUSTERPMU_PMEVCNTSR5, Cluster Performance Monitors Event Count Snapshot Registers

Holds event counter 5, which counts events.

Configurations

This register is available in all configurations.

Attributes

Width

64

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-331: ext_clusterpmu_pmevcntsr5 bit assignments

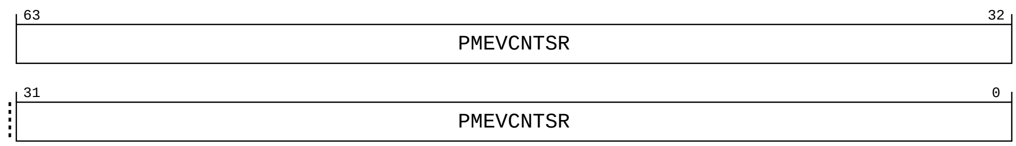


Table B-487: CLUSTERPMU_PMEVCNTSR5 bit descriptions

Bits	Name	Description	Reset
[63:0]	PMEVCNTSR	Event counter 5.	`64{x}`

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

performance-monitors

Register offset

0x638

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx1
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-332: ext_clusterpmu_pmsssr bit assignments

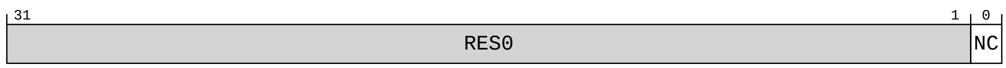


Table B-488: CLUSTERPMU_PMSSSR bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**
[0]	NC	No capture. Indicates whether the PMU counters have been captured. **`0b0`** PMU counters captured. **`0b1`** PMU counters not captured.	`0b1`

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_PMOVSSET and ext-CLUSTERPMU_PMOVSCLR.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0x640

Access type

See bit descriptions

Reset value

```
x000 0000 0000 0000 0000 0000 00xx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-333: ext_clusterpmu_pmovssr bit assignments

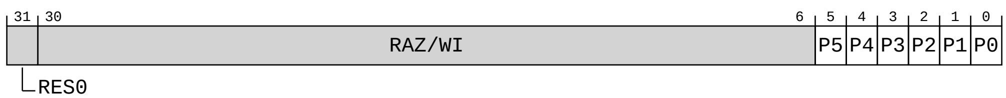


Table B-489: CLUSTERPMU_PMOVSSR bit descriptions

Bits	Name	Description	Reset
[31]	**RES0**	Reserved	**RES0**
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**

[5]	P5	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.
			`x`

[4]	P4	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.
			`x`

[3]	P3	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.
			`x`

Bits	Name	Description	Reset
[2]	P2	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are RAZ/WI. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[1]	P1	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are RAZ/WI. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[0]	P0	Event counter overflow bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are RAZ/WI. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x

B.2.4.21 CLUSTERPMU_PMCNTENSET, Cluster Performance Monitors Count Enable Set register

Enables any implemented event counters AArch64-IMP_CLUSTERPMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xC00

Access type

See bit descriptions

Reset value

```
x000 0000 0000 0000 0000 0000 00xx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-334: ext_clusterpmu_pmcntenset bit assignments

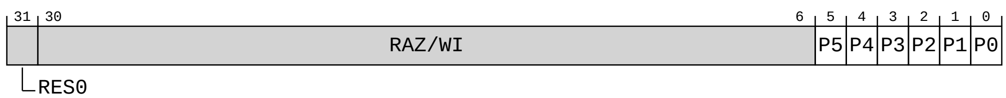


Table B-490: CLUSTERPMU_PMCNTENSET bit descriptions

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 **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.
			`x`

[4]	P4	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>.
		
		If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.
			`x`

Bits	Name	Description	Reset
[3]	P3	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[2]	P2	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[1]	P1	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x
[0]	P0	Event counter enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> event counter is enabled. When written, enables ext-CLUSTERPMU_PMEVCNTR<n>.	x

B.2.4.22 CLUSTERPMU_PMCNTENCLR, Cluster Performance Monitors Count Enable Clear register

Disables any implemented event counters AArch64-IMP_CLUSTERPMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xC20

Access type

See bit descriptions

Reset value

```
x000 0000 0000 0000 0000 0000 00xx xxxx
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-335: ext_clusterpmu_pmcntenclr bit assignments

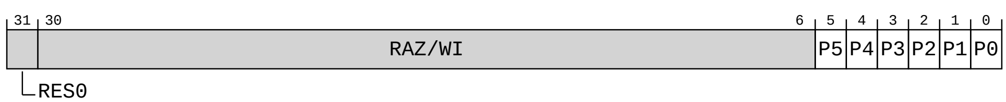


Table B-491: CLUSTERPMU_PMCNTENCLR bit descriptions

Bits	Name	Description	Reset
[31]	**RES0**	Reserved	**RES0**
[30:6]	**RAZ/ WI**	Reserved	**RAZ/ WI**

[5]	P5	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>.
		
		If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.
			`x`

[4]	P4	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>.
		
		If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.
			`x`

Bits	Name	Description	Reset
[3]	P3	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **`0b1`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	`x`
[2]	P2	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **`0b1`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	`x`
[1]	P1	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **`0b1`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	`x`
[0]	P0	Event counter disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is disabled. When written, has no effect. **`0b1`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> is enabled. When written, disables ext-CLUSTERPMU_PMEVCNTR<n>.	`x`

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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-336: ext_clusterpmu_pmintenset bit assignments

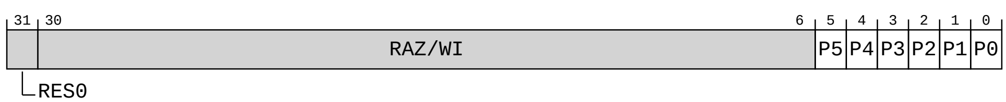


Table B-492: CLUSTERPMU_PMINTENSET bit descriptions

Bits	Name	Description	Reset
[31]	**RES0**	Reserved	**RES0**
[30:6]	**RAZ/WI**	Reserved	**RAZ/WI**
[5]	P5	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x

Bits	Name	Description	Reset
[4]	P4	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[3]	P3	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[2]	P2	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[1]	P1	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x
[0]	P0	Event counter overflow interrupt request enable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **0b1** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, enables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	x

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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-337: ext_clusterpmu_pmintenclr bit assignments

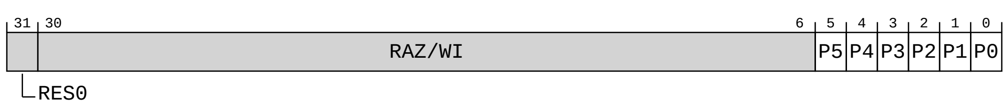


Table B-493: CLUSTERPMU_PMINTENCLR bit descriptions

Bits	Name	Description	Reset
[31]	RES0	Reserved	RES0
[30:6]	RAZ/WI	Reserved	RAZ/WI

Bits	Name	Description	Reset
[5]	P5	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	`x`
[4]	P4	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	`x`
[3]	P3	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	`x`
[2]	P2	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	`x`
[1]	P1	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>. If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect. **`0b1`** When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.	`x`

Bits	Name	Description	Reset

[0]	P0	Event counter overflow interrupt request disable bit for ext-CLUSTERPMU_PMEVCNTR<n>.
		
		If ext-CLUSTERPMU_PMCFGR.N is less than 31, bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is disabled. When written, has no effect.
		`0b1`
		When read, means that the ext-CLUSTERPMU_PMEVCNTR<n> event counter interrupt request is enabled. When written, disables the ext-CLUSTERPMU_PMEVCNTR<n> interrupt request.
			x

B.2.4.25 CLUSTERPMU_PMOVSCLR, 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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xC80

Access type

See bit descriptions

Reset value

```
x000 0000 0000 0000 0000 0000 00xx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-338: ext_clusterpmu_pmovsclr bit assignments

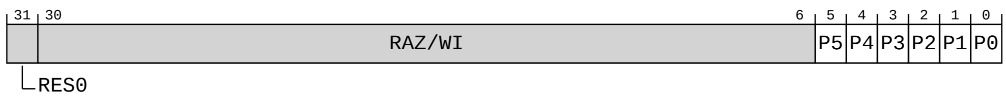


Table B-494: CLUSTERPMU_PMOVSCLR bit descriptions

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 **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.
			`x`

[4]	P4	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.
			`x`

[3]	P3	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.
			`x`

[2]	P2	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**.
		`0b0`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect.
		`0b1`
		When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.
			`x`

Bits	Name	Description	Reset
[1]	P1	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **`0b1`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	`x`
[0]	P0	Event counter overflow clear bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **`0b0`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **`0b1`** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, clears the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 0.	`x`

B.2.4.26 CLUSTERPMU_PMOVSSET, Cluster Performance Monitors Overflow Flag Status Set register

Sets the state of the overflow bit for each of the implemented event counters AArch64-PMEVCNTR<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-339: ext_clusterpmu_pmovsset bit assignments

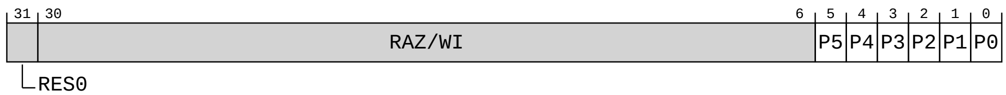


Table B-495: CLUSTERPMU_PMOVSSET bit descriptions

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_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[4]	P4	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[3]	P3	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x

Bits	Name	Description	Reset
[2]	P2	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[1]	P1	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x
[0]	P0	Event counter overflow set bit for ext-CLUSTERPMU_PMEVCNTR<n>. Bits [30:ext-CLUSTERPMU_PMCFGR.N] are **RAZ/WI**. **0b0** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has not overflowed since this bit was last cleared. When written, has no effect. **0b1** When read, means that ext-CLUSTERPMU_PMEVCNTR<n> has overflowed since this bit was last cleared. When written, sets the ext-CLUSTERPMU_PMEVCNTR<n> overflow bit to 1.	x

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

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-340: ext_clusterpmu_pmcfgr bit assignments

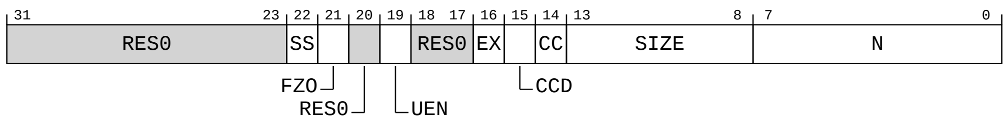


Table B-496: CLUSTERPMU_PMCFGR bit descriptions

Bits	Name	Description	Reset
[31:23]	**RES0**	Reserved	**RES0**
[22]	SS	Snapshot supported. **0b1** Snapshot supported.	0b1
[21]	FZO	Freeze on overflow supported. **0b1** Freeze on overflow supported.	0b1
[20]	**RES0**	Reserved	**RES0**
[19]	UEN	User-mode Enable Register supported. **0b0** User-mode Enable Register not supported.	0b0
[18:17]	**RES0**	Reserved	**RES0**
[16]	EX	Export supported. **0b0** ext-CLUSTERPMU_PMCR.X is **RES0**.	0b0
[15]	CCD	Cycle counter has prescale. **0b0** ext-CLUSTERPMU_PMCR.D is **RES0**.	0b0
[14]	CC	Dedicated cycle counter. **0b0** Dedicated cycle counter ext-CLUSTERPMU_PMCCNTR is not supported.	0b0

Bits	Name	Description	Reset
[13:8]	SIZE	Size of counters, minus one. This field defines the size of the largest counter implemented by the Performance Monitors Unit. This field is used by software to determine the spacing of the counters in the memory-map. **0b111111** The largest counter is 64-bits. Counters are at doubleword-aligned addresses.	0b111111
[7:0]	N	Number of counters implemented. **0b00000101** Six event counters implemented.	0x05

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_PMCFGR and the ID registers.

Attributes

Width

32

Component

performance-monitors

Register offset

0xE04

Access type

inconsistent

Reset value

```
xxxx xxxx xxxx xxxx xxxx xx0x xxx0 xx00
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-341: ext_clusterpmu_pmcr bit assignments

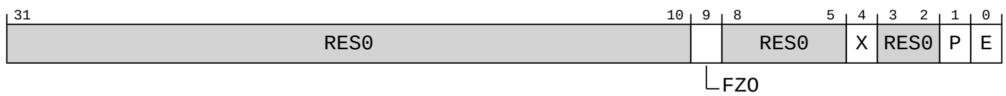


Table B-497: CLUSTERPMU_PMCR bit descriptions

Bits	Name	Description	Reset
[31:10]	RES0	Reserved	RES0
[9]	FZO	Freeze on overflow. **0b0** Freeze on overflow disabled. **0b1** 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. **0b0** Cluster PMU events are not exported externally.	0b0
[3:2]	RES0	Reserved	RES0
[1]	P	Event counter reset. This bit is WO. **0b0** No action. **0b1** Reset all event counters to zero. This bit is always RAZ. **Note:** Resetting the event counters does not change the event counter overflow bits.	0b0
[0]	E	Enable. **0b0** All event counters are disabled. **0b1** All event counters can be enabled by ext-CLUSTERPMU_PMCNTENSET. This bit is RW.	0b0

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

performance-monitors

Register offset

0xE08

Access type

See bit descriptions

Reset value

0100 1110 1110 0000 0010 0100 0011 1011

Bit descriptions

Figure B-342: ext_clusterpmu_pmiidr bit assignments

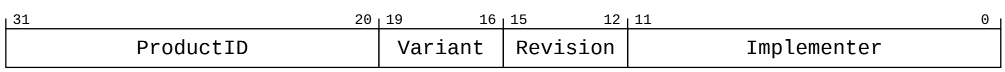


Table B-498: CLUSTERPMU_PMIIDR bit descriptions

Bits	Name	Description	Reset
[31:20]	ProductID	Value identifying the PMU Component. **`0b010011101110`** DSU-C1 Cluster PMU.	`0x4EE`
[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. **`0b0010`** Product revision 2.	`0b0010`

Bits	Name	Description	Reset
[11:0]	Implementer	Contains the JEP106 code of the company that implemented the PMU Component. For an Arm implementation, bits[11:0] are `0x43B`. **`0b010000111011`** Arm implementation.	`0x43B`

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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xE20

Access type

See bit descriptions

Reset value

0010 0110 0000 0010 0000 0000 0000 0000

Bit descriptions

Figure B-343: ext_clusterpmu_pmceid0 bit assignments

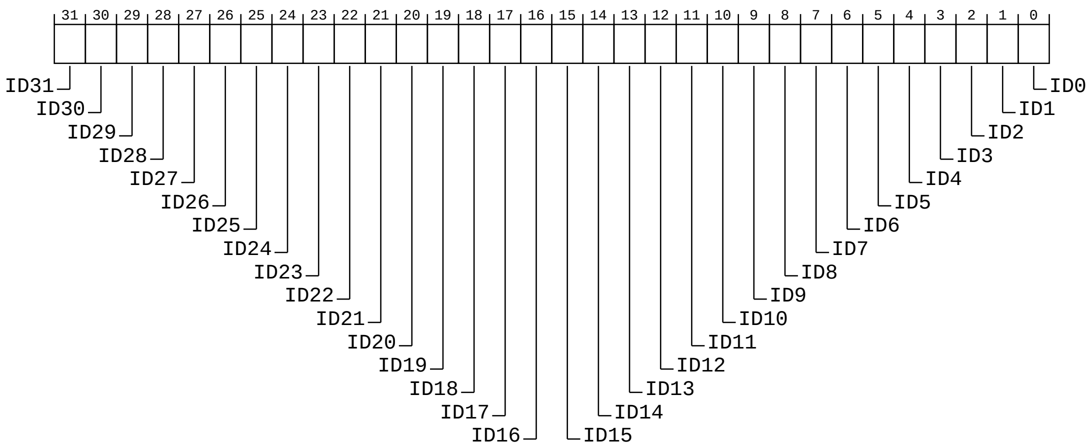


Table B-499: CLUSTERPMU_PMCEID0 bit descriptions

Bits	Name	Description	Reset
[31]	ID31	Common event `0x001F` implemented. **`0b0`** Event 0x001F not implemented.	`0b0`
[30]	ID30	Common event `0x001E` implemented. **`0b0`** CHAIN event not implemented.	`0b0`
[29]	ID29	Common event `0x001D` implemented. **`0b1`** BUS_CYCLES event implemented.	`0b1`
[28]	ID28	Common event `0x001C` implemented. **`0b0`** Event 0x001C not implemented.	`0b0`
[27]	ID27	Common event `0x001B` implemented. **`0b1`** Event 0x001B not implemented.	`0b0`
[26]	ID26	Common event `0x001A` implemented. **`0b1`** MEMORY_ERROR event implemented.	`0b1`
[25]	ID25	Common event `0x0019` implemented. **`0b1`** BUS_ACCESS event implemented.	`0b1`
[24]	ID24	Common event `0x0018` implemented. **`0b0`** Event 0x0018 not implemented.	`0b0`

Bits	Name	Description	Reset
[23]	ID23	Common event `0x0017` implemented. **`0b0`** Event 0x0017 not implemented.	`0b0`
[22]	ID22	Common event `0x0016` implemented. **`0b0`** Event 0x0016 not implemented.	`0b0`
[21]	ID21	Common event `0x0015` implemented. **`0b0`** Event 0x0015 not implemented.	`0b0`
[20]	ID20	Common event `0x0014` implemented. **`0b0`** Event 0x0014 not implemented.	`0b0`
[19]	ID19	Common event `0x0013` implemented. **`0b0`** Event 0x0013 not implemented.	`0b0`
[18]	ID18	Common event `0x0012` implemented. **`0b0`** Event 0x0012 not implemented.	`0b0`
[17]	ID17	Common event `0x0011` implemented. **`0b1`** CYCLES event implemented.	`0b1`
[16]	ID16	Common event `0x0010` implemented. **`0b0`** Event 0x0010 not implemented.	`0b0`
[15]	ID15	Common event `0x000F` implemented. **`0b0`** Event 0x000F not implemented.	`0b0`
[14]	ID14	Common event `0x000E` implemented. **`0b0`** Event 0x000E not implemented.	`0b0`
[13]	ID13	Common event `0x000D` implemented. **`0b0`** Event 0x000D not implemented.	`0b0`
[12]	ID12	Common event `0x000C` implemented. **`0b0`** Event 0x000C not implemented.	`0b0`
[11]	ID11	Common event `0x000B` implemented. **`0b0`** Event 0x000B not implemented.	`0b0`
[10]	ID10	Common event `0x000A` implemented. **`0b0`** Event 0x000A not implemented.	`0b0`

Bits	Name	Description	Reset
[9]	ID9	Common event `0x0009` implemented. **0b0** Event 0x0009 not implemented.	0b0
[8]	ID8	Common event `0x0008` implemented. **0b0** Event 0x0008 not implemented.	0b0
[7]	ID7	Common event `0x0007` implemented. **0b0** Event 0x0007 not implemented.	0b0
[6]	ID6	Common event `0x0006` implemented. **0b0** Event 0x0006 not implemented.	0b0
[5]	ID5	Common event `0x0005` implemented. **0b0** Event 0x0005 not implemented.	0b0
[4]	ID4	Common event `0x0004` implemented. **0b0** Event 0x0004 not implemented.	0b0
[3]	ID3	Common event `0x0003` implemented. **0b0** Event 0x0003 not implemented.	0b0
[2]	ID2	Common event `0x0002` implemented. **0b0** Event 0x0002 not implemented.	0b0
[1]	ID1	Common event `0x0001` implemented. **0b0** Event 0x0001 not implemented.	0b0
[0]	ID0	Common event `0x0000` implemented. **0b0** Event 0x0000 not implemented.	0b0

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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xE24

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0001 1110 0000 0000

Bit descriptions

Figure B-344: ext_clusterpmu_pmceid1 bit assignments

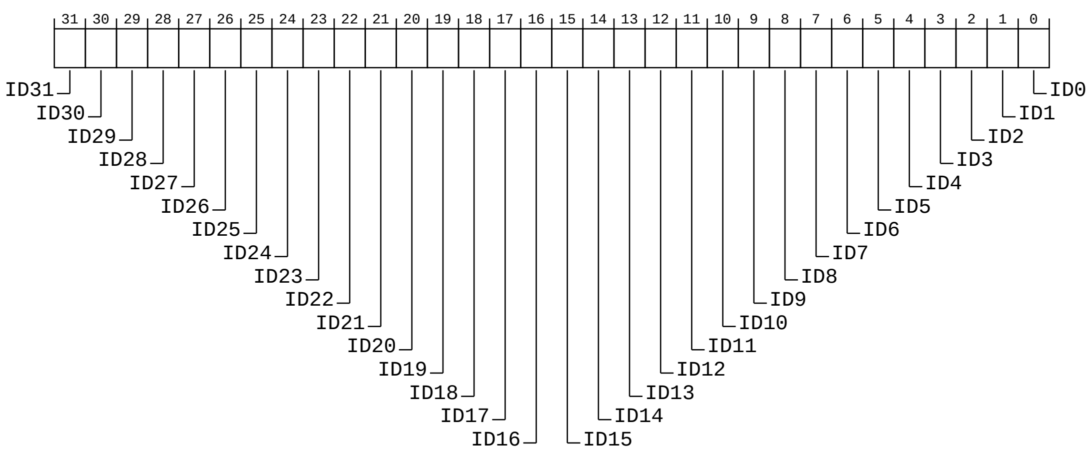


Table B-500: CLUSTERPMU_PMCEID1 bit descriptions

Bits	Name	Description	Reset
[31]	ID31	Common event `0x003F` implemented. **`0b0`** Event 0x003F not implemented.	`0b0`
[30]	ID30	Common event `0x003E` implemented. **`0b0`** Event 0x003E not implemented.	`0b0`
[29]	ID29	Common event `0x003D` implemented. **`0b0`** Event 0x003D not implemented.	`0b0`

Bits	Name	Description	Reset
[28]	ID28	Common event `0x003C` implemented. **0b0** Event 0x003C not implemented.	0b0
[27]	ID27	Common event `0x003B` implemented. **0b0** Event 0x003B not implemented.	0b0
[26]	ID26	Common event `0x003A` implemented. **0b0** Event 0x003A not implemented.	0b0
[25]	ID25	Common event `0x0039` implemented. **0b0** Event 0x0039 not implemented.	0b0
[24]	ID24	Common event `0x0038` implemented. **0b0** Event 0x0038 not implemented.	0b0
[23]	ID23	Common event `0x0037` implemented. **0b0** Event 0x0037 not implemented.	0b0
[22]	ID22	Common event `0x0036` implemented. **0b0** Event 0x0036 not implemented.	0b0
[21]	ID21	Common event `0x0035` implemented. **0b0** Event 0x0035 not implemented.	0b0
[20]	ID20	Common event `0x0034` implemented. **0b0** Event 0x0034 not implemented.	0b0
[19]	ID19	Common event `0x0033` implemented. **0b0** Event 0x0033 not implemented.	0b0
[18]	ID18	Common event `0x0032` implemented. **0b0** Event 0x0032 not implemented.	0b0
[17]	ID17	Common event `0x0031` implemented. **0b0** Event 0x0031 not implemented.	0b0
[16]	ID16	Common event `0x0030` implemented. **0b0** Event 0x0030 not implemented.	0b0
[15]	ID15	Common event `0x002F` implemented. **0b0** Event 0x002F not implemented.	0b0

Bits	Name	Description	Reset
[14]	ID14	Common event `0x002E` implemented. **0b0** Event 0x002E not implemented.	0b0
[13]	ID13	Common event `0x002D` implemented. **0b0** Event 0x002D not implemented.	0b0
[12]	ID12	Common event `0x002C` implemented. **0b1** L3D_CACHE_WB event implemented.	0b1
[11]	ID11	Common event `0x002B` implemented. **0b1** L3D_CACHE event implemented.	0b1
[10]	ID10	Common event `0x002A` implemented. **0b1** L3D_CACHE_REFILL event implemented.	0b1
[9]	ID9	Common event `0x0029` implemented. **0b1** L3D_CACHE_ALLOCATE event implemented.	0b1
[8]	ID8	Common event `0x0028` implemented. **0b0** Event 0x0028 not implemented.	0b0
[7]	ID7	Common event `0x0027` implemented. **0b0** Event 0x0027 not implemented.	0b0
[6]	ID6	Common event `0x0026` implemented. **0b0** Event 0x0026 not implemented.	0b0
[5]	ID5	Common event `0x0025` implemented. **0b0** Event 0x0025 not implemented.	0b0
[4]	ID4	Common event `0x0024` implemented. **0b0** Event 0x0024 not implemented.	0b0
[3]	ID3	Common event `0x0023` implemented. **0b0** Event 0x0023 not implemented.	0b0
[2]	ID2	Common event `0x0022` implemented. **0b0** Event 0x0022 not implemented.	0b0
[1]	ID1	Common event `0x0021` implemented. **0b0** Event 0x0021 not implemented.	0b0

Bits	Name	Description	Reset
[0]	ID0	Common event `0x0020` implemented. **`0b0`** Event 0x0020 not implemented.	`0b0`

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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xE28

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-345: ext_clusterpmu_pmceid2 bit assignments

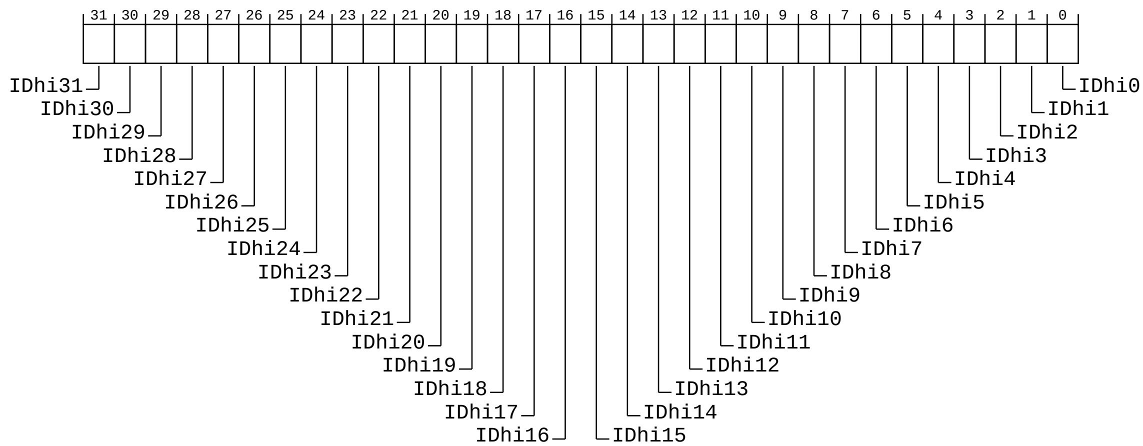


Table B-501: CLUSTERPMU_PMCEID2 bit descriptions

Bits	Name	Description	Reset
[31]	IDhi31	Common event `0x401F` implemented. **0b0** Event 0x401F not implemented.	0b0
[30]	IDhi30	Common event `0x401E` implemented. **0b0** Event 0x401E not implemented.	0b0
[29]	IDhi29	Common event `0x401D` implemented. **0b0** Event 0x401D not implemented.	0b0
[28]	IDhi28	Common event `0x401C` implemented. **0b0** Event 0x401C not implemented.	0b0
[27]	IDhi27	Common event `0x401B` implemented. **0b0** Event 0x401B not implemented.	0b0
[26]	IDhi26	Common event `0x401A` implemented. **0b0** Event 0x401A not implemented.	0b0
[25]	IDhi25	Common event `0x4019` implemented. **0b0** Event 0x4019 not implemented.	0b0
[24]	IDhi24	Common event `0x4018` implemented. **0b0** Event 0x4018 not implemented.	0b0

Bits	Name	Description	Reset
[23]	IDhi23	Common event `0x4017` implemented. **0b0** Event 0x4017 not implemented.	`0b0`
[22]	IDhi22	Common event `0x4016` implemented. **0b0** Event 0x4016 not implemented.	`0b0`
[21]	IDhi21	Common event `0x4015` implemented. **0b0** Event 0x4015 not implemented.	`0b0`
[20]	IDhi20	Common event `0x4014` implemented. **0b0** Event 0x4014 not implemented.	`0b0`
[19]	IDhi19	Common event `0x4013` implemented. **0b0** Event 0x4013 not implemented.	`0b0`
[18]	IDhi18	Common event `0x4012` implemented. **0b0** Event 0x4012 not implemented.	`0b0`
[17]	IDhi17	Common event `0x4011` implemented. **0b0** Event 0x4011 not implemented.	`0b0`
[16]	IDhi16	Common event `0x4010` implemented. **0b0** Event 0x4010 not implemented.	`0b0`
[15]	IDhi15	Common event `0x400F` implemented. **0b0** Event 0x400F not implemented.	`0b0`
[14]	IDhi14	Common event `0x400E` implemented. **0b0** Event 0x400E not implemented.	`0b0`
[13]	IDhi13	Common event `0x400D` implemented. **0b0** Event 0x400D not implemented.	`0b0`
[12]	IDhi12	Common event `0x400C` implemented. **0b0** Event 0x400C not implemented.	`0b0`
[11]	IDhi11	Common event `0x400B` implemented. **0b0** Event 0x400B not implemented.	`0b0`
[10]	IDhi10	Common event `0x400A` implemented. **0b0** Event 0x400A not implemented.	`0b0`

Bits	Name	Description	Reset
[9]	IDhi9	Common event `0x4009` implemented. **0b0** Event 0x4009 not implemented.	`0b0`
[8]	IDhi8	Common event `0x4008` implemented. **0b0** Event 0x4008 not implemented.	`0b0`
[7]	IDhi7	Common event `0x4007` implemented. **0b0** Event 0x4007 not implemented.	`0b0`
[6]	IDhi6	Common event `0x4006` implemented. **0b0** Event 0x4006 not implemented.	`0b0`
[5]	IDhi5	Common event `0x4005` implemented. **0b0** Event 0x4005 not implemented.	`0b0`
[4]	IDhi4	Common event `0x4004` implemented. **0b0** Event 0x4004 not implemented.	`0b0`
[3]	IDhi3	Common event `0x4003` implemented. **0b0** Event 0x4003 not implemented.	`0b0`
[2]	IDhi2	Common event `0x4002` implemented. **0b0** Event 0x4002 not implemented.	`0b0`
[1]	IDhi1	Common event `0x4001` implemented. **0b0** Event 0x4001 not implemented.	`0b0`
[0]	IDhi0	Common event `0x4000` implemented. **0b0** Event 0x4000 not implemented.	`0b0`

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

This register is available in all configurations.

Attributes

Width

32

Component

performance-monitors

Register offset

0xE2C

Access type

See bit descriptions

Reset value

0000 0000 0000 0000 0000 0000 0000 0000

Bit descriptions

Figure B-346: ext_clusterpmu_pmceid3 bit assignments

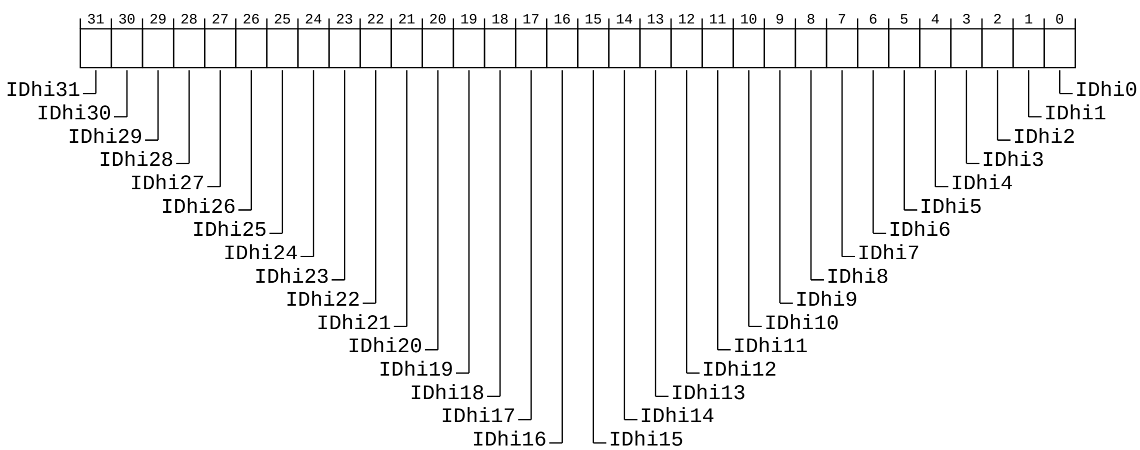


Table B-502: CLUSTERPMU_PMCEID3 bit descriptions

Bits	Name	Description	Reset
[31]	IDhi31	Common event `0x403F` implemented. **`0b0`** Event 0x403F not implemented.	`0b0`
[30]	IDhi30	Common event `0x403E` implemented. **`0b0`** Event 0x403E not implemented.	`0b0`
[29]	IDhi29	Common event `0x403D` implemented. **`0b0`** Event 0x403D not implemented.	`0b0`

Bits	Name	Description	Reset
[28]	IDhi28	Common event `0x403C` implemented. **`0b0`** Event 0x403C not implemented.	`0b0`
[27]	IDhi27	Common event `0x403B` implemented. **`0b0`** Event 0x403B not implemented.	`0b0`
[26]	IDhi26	Common event `0x403A` implemented. **`0b0`** Event 0x403A not implemented.	`0b0`
[25]	IDhi25	Common event `0x4039` implemented. **`0b0`** Event 0x4039 not implemented.	`0b0`
[24]	IDhi24	Common event `0x4038` implemented. **`0b0`** Event 0x4038 not implemented.	`0b0`
[23]	IDhi23	Common event `0x4037` implemented. **`0b0`** Event 0x4037 not implemented.	`0b0`
[22]	IDhi22	Common event `0x4036` implemented. **`0b0`** Event 0x4036 not implemented.	`0b0`
[21]	IDhi21	Common event `0x4035` implemented. **`0b0`** Event 0x4035 not implemented.	`0b0`
[20]	IDhi20	Common event `0x4034` implemented. **`0b0`** Event 0x4034 not implemented.	`0b0`
[19]	IDhi19	Common event `0x4033` implemented. **`0b0`** Event 0x4033 not implemented.	`0b0`
[18]	IDhi18	Common event `0x4032` implemented. **`0b0`** Event 0x4032 not implemented.	`0b0`
[17]	IDhi17	Common event `0x4031` implemented. **`0b0`** Event 0x4031 not implemented.	`0b0`
[16]	IDhi16	Common event `0x4030` implemented. **`0b0`** Event 0x4030 not implemented.	`0b0`
[15]	IDhi15	Common event `0x402F` implemented. **`0b0`** Event 0x402F not implemented.	`0b0`

Bits	Name	Description	Reset
[14]	IDhi14	Common event `0x402E` implemented. **0b0** Event 0x402E not implemented.	0b0
[13]	IDhi13	Common event `0x402D` implemented. **0b0** Event 0x402D not implemented.	0b0
[12]	IDhi12	Common event `0x402C` implemented. **0b0** Event 0x402C not implemented.	0b0
[11]	IDhi11	Common event `0x402B` implemented. **0b0** Event 0x402B not implemented.	0b0
[10]	IDhi10	Common event `0x402A` implemented. **0b0** Event 0x402A not implemented.	0b0
[9]	IDhi9	Common event `0x4029` implemented. **0b0** Event 0x4029 not implemented.	0b0
[8]	IDhi8	Common event `0x4028` implemented. **0b0** Event 0x4028 not implemented.	0b0
[7]	IDhi7	Common event `0x4027` implemented. **0b0** Event 0x4027 not implemented.	0b0
[6]	IDhi6	Common event `0x4026` implemented. **0b0** Event 0x4026 not implemented.	0b0
[5]	IDhi5	Common event `0x4025` implemented. **0b0** Event 0x4025 not implemented.	0b0
[4]	IDhi4	Common event `0x4024` implemented. **0b0** Event 0x4024 not implemented.	0b0
[3]	IDhi3	Common event `0x4023` implemented. **0b0** Event 0x4023 not implemented.	0b0
[2]	IDhi2	Common event `0x4022` implemented. **0b0** Event 0x4022 not implemented.	0b0
[1]	IDhi1	Common event `0x4021` implemented. **0b0** Event 0x4021 not implemented.	0b0

Bits	Name	Description	Reset
[0]	IDhi0	Common event `0x4020` implemented. **`0b0`** Event 0x4020 not implemented.	`0b0`

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

performance-monitors

Register offset

0xE30

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxx0
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-347: ext_clusterpmu_pmsscr bit assignments

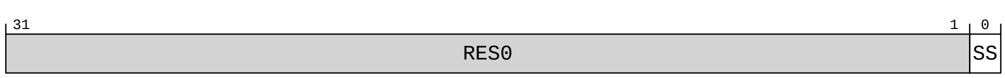


Table B-503: CLUSTERPMU_PMSSCR bit descriptions

Bits	Name	Description	Reset
[31:1]	**RES0**	Reserved	**RES0**

Bits	Name	Description	Reset
[0]	SS	Capture now. The possible values for writing to this bit are: **0b0** Ignored. **0b1** Initiate a capture immediately.	0b0

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

performance-monitors

Register offset

0xE38

Access type

See bit descriptions

Reset value

```
x000 0000 0000 0000 0000 0000 00xx xxxx
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-348: ext_clusterpmu_pmssrr bit assignments

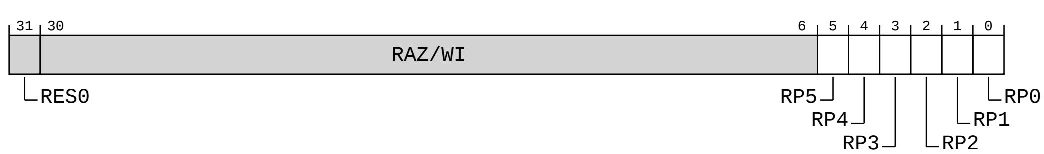


Table B-504: CLUSTERPMU_PMSSRR bit descriptions

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 >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[4]	RP4	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[3]	RP3	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[2]	RP2	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x
[1]	RP1	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **0b0** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **0b1** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	x

Bits	Name	Description	Reset
[0]	RP0	Reset performance counter. For each bit [x], if x >= ext-CLUSTERPMU_PMCR.N, the number of implemented counters, then RP[x] is **RAZ/WI**. Otherwise, indicates whether ext-PMEVCNTR<x> and ext-PMOVSR[x] are to be reset after a capture. **`0b0`** Do not reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture. **`0b1`** Reset ext-CLUSTERPMU_PMEVCNTR<n> and ext-CLUSTERPMU_PMOVSR[x] on capture.	`x`

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

performance-monitors

Register offset

0xFA8

Access type

See bit descriptions

Reset value

```
x0xx xxx0 xxxx xxxx 1000 0000 0000 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-349: ext_clusterpmu_pmdevaff0 bit assignments

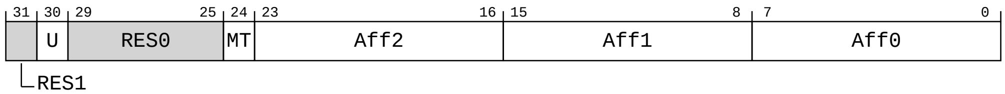


Table B-505: CLUSTERPMU_PMDEVAFF0 bit descriptions

Bits	Name	Description	Reset
[31]	**RES1**	Reserved	**RES1**
[30]	U	Uniprocessor/Multiprocessor system. **`0b0`** Processor is part of a multiprocessor system.	`0b0`
[29:25]	**RES0**	Reserved	**RES0**
[24]	MT	Indicates whether the lowest level of affinity consists of logical PEs that are implemented using a multithreading type approach. **`0b0`** Performance of PEs at the lowest affinity level is largely independent.	`0b0`
[23:16]	Aff2	Affinity level 2. Value read from the CFGMPIDRAFF2 configuration pins.	`8{x}`
[15:8]	Aff1	Affinity level 1. **`0b10000000`** Affinity with all cores in cluster.	`0x80`
[7:0]	Aff0	Affinity level 0. **`0b00000000`** Affinity with all core threads in cluster.	`0x00`

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

performance-monitors

Register offset

0xFAC

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-350: ext_clusterpmu_pmdevaff1 bit assignments

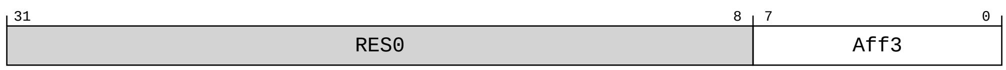


Table B-506: CLUSTERPMU_PMDEVAFF1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	Aff3	Affinity level 3. Value read from the CFGMPIDRAFF3 configuration pins.	8{x}

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

performance-monitors

Register offset

0xFB8

Access type

See bit descriptions

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-351: ext_clusterpmu_pmauthstatus bit assignments

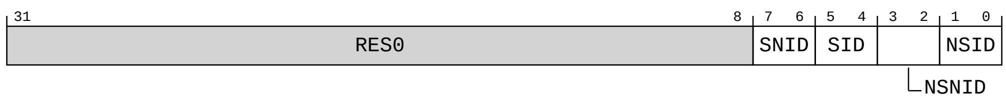


Table B-507: CLUSTERPMU_PMAUTHSTATUS bit descriptions

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

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

performance-monitors

Register offset

0xFBC

Access type

See bit descriptions

Reset value

0100 0111 0111 0000 0010 1010 0001 0110

Bit descriptions

Figure B-352: ext_clusterpmu_pmdevarch bit assignments

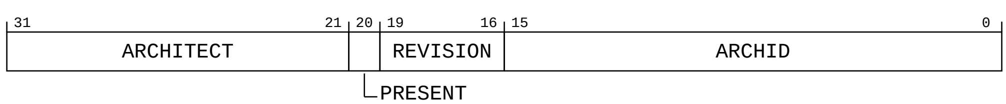


Table B-508: CLUSTERPMU_PMDEVARCH bit descriptions

Bits	Name	Description	Reset
[31:21]	ARCHITECT	Architect. **`0b01000111011`** JEP106 continuation code 0x4, ID code 0x3B. Arm Limited.	`0b01000111011`
[20]	PRESENT	Present. **`0b1`** DEVARCH information present.	`0b1`
[19:16]	REVISION	Revision. **`0b0000`** Revision 0.	`0b0000`
[15:0]	ARCHID	Architecture ID. **`0b0010101000010110`** Processor Performance Monitor (PMU) architecture PMUv3.	`0x2A16`

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

performance-monitors

Register offset

0xFC8

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx xxxx 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-353: ext_clusterpmu_pmdevid bit assignments

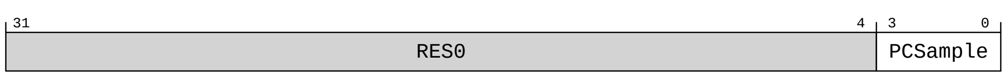


Table B-509: CLUSTERPMU_PMDEVID bit descriptions

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

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

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-354: ext_clusterpmu_pmdevtype bit assignments

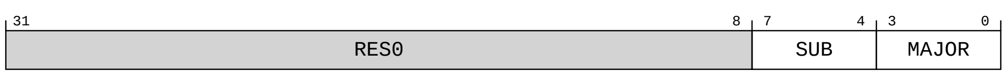


Table B-510: CLUSTERPMU_PMDEVTYPE bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	SUB	Subtype. **0b0001** Associated with a processor.	0b0001

Bits	Name	Description	Reset
[3:0]	MAJOR	Major type. **0b0110** Performance Monitor.	0b0110

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

performance-monitors

Register offset

0xFD0

Access type

See bit descriptions

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-355: ext_clusterpmu_pmpidr4 bit assignments

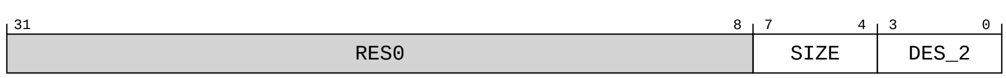


Table B-511: CLUSTERPMU_PMPIDR4 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0

Bits	Name	Description	Reset
[7:4]	SIZE	4KB count. **`0b0000`** The component uses a single 4KB block.	`0b0000`
[3:0]	DES_2	JEP106 continuation code. **`0b0100`** Arm Limited. Number of 0x7F bytes in full JEP106 code 0x7F 0x7F 0x7F 0x7F 0x3B.	`0b0100`

B.2.4.43 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

performance-monitors

Register offset

0xFE0

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1110 1110
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-356: ext_clusterpmu_pmpidr0 bit assignments

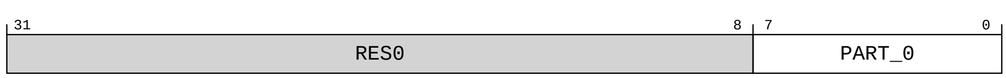


Table B-512: CLUSTERPMU_PMPIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:0]	PART_0	Part number bits [7:0]. **0b11101110** DSU-C1 Cluster PMU. Bits [7:0] of part number 0x4EE.	0xEE

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

performance-monitors

Register offset

0xFE4

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0100
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-357: ext_clusterpmu_pmpidr1 bit assignments

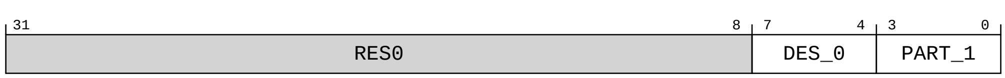


Table B-513: CLUSTERPMU_PMPIDR1 bit descriptions

Bits	Name	Description	Reset
[31:8]	RES0	Reserved	RES0
[7:4]	DES_0	JEP106 identification code bits [3:0]. **0b1011** Arm Limited. Bits [3:0] of JEP106 identification code 0x3B.	0b1011
[3:0]	PART_1	Part number bits [11:8]. **0b0100** DSU-C1 Cluster PMU. Bits [11:8] of part number 0x4EE.	0b0100

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

performance-monitors

Register offset

0xFE8

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 0010 1011
|    |    |    |    |    |    |    |  |
31   27   23   19   15   11   7    3  0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-358: ext_clusterpmu_pmpidr2 bit assignments

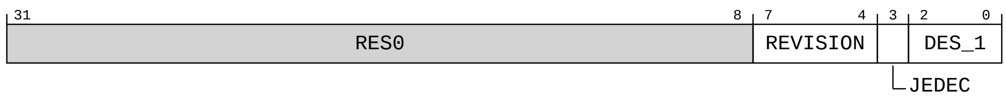


Table B-514: CLUSTERPMU_PMPIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVISION	Component major revision. **0b0000** Component major revision 0. **0b0001** Component major revision 1. **0b0010** Component major revision 2. For DSU-C1: • Major revision 0 corresponds to r0p0. • Major revision 1 corresponds to r0p1. • Major revision 2 corresponds to r0p2.	0b0010
[3]	JEDEC	JEDEC assignee. **0b1** JEDEC-assignee values is used.	0b1
[2:0]	DES_1	JEP106 identification code bits [6:4]. **0b011** Arm Limited. Bits [6:4] of JEP106 identification code 0x3B.	0b011

B.2.4.46 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

performance-monitors

Register offset

0xFEC

Access type

See bit descriptions

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-359: ext_clusterpmu_pmpidr3 bit assignments

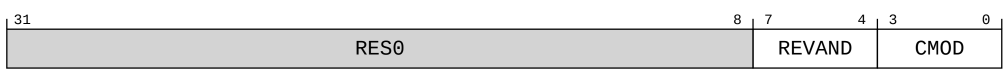


Table B-515: CLUSTERPMU_PMPIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:4]	REVAND	Component minor revision. **0b0000** Component minor revision 0.	0b0000
[3:0]	CMOD	Customer Modified. **0b0000** The component is not modified from the original design.	0b0000

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

performance-monitors

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
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-360: ext_clusterpmu_pmcidr0 bit assignments

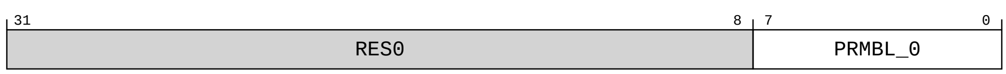


Table B-516: CLUSTERPMU_PMCIDR0 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_0	Preamble. **0b00001101** CoreSight component identification preamble.	0x0D

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

performance-monitors

Register offset

0xFF4

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1001 0000
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-361: ext_clusterpmu_pmcidr1 bit assignments

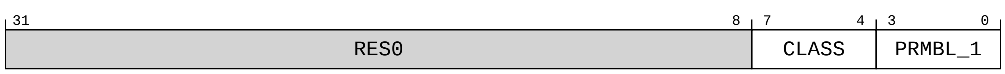


Table B-517: CLUSTERPMU_PMCIDR1 bit descriptions

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

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

performance-monitors

Register offset

0xFF8

Access type

See bit descriptions

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-362: ext_clusterpmu_pmcidr2 bit assignments

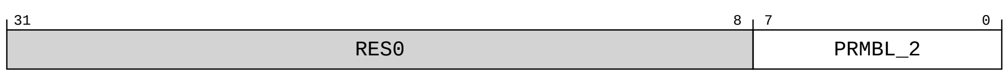


Table B-518: CLUSTERPMU_PMCIDR2 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_2	Preamble. **0b00000101** CoreSight component identification preamble.	0x05

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

performance-monitors

Register offset

0xFFC

Access type

See bit descriptions

Reset value

```
xxxx xxxx xxxx xxxx xxxx xxxx 1011 0001
|    |    |    |    |    |    |    |   |
31   27   23   19   15   11   7    3   0
```

Note

Where the reset reads xxxx, see individual bits.

Bit descriptions

Figure B-363: ext_clusterpmu_pmcidr3 bit assignments

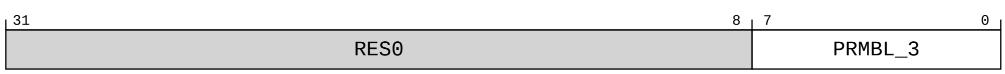


Table B-519: CLUSTERPMU_PMCIDR3 bit descriptions

Bits	Name	Description	Reset
[31:8]	**RES0**	Reserved	**RES0**
[7:0]	PRMBL_3	Preamble. **0b10110001** CoreSight component identification preamble.	0xB1

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

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 r0p2, 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
0002-05	10 September 2025	Non-Confidential	Second early access release for r0p2
0002-04	30 May 2025	Confidential	First early access release for r0p2
0001-03	29 July 2024	Confidential	First early access release for r0p1
0000-02	29 February 2024	Confidential	First limited access release for r0p0
0000-01	30 November 2023	Confidential	First beta release for r0p0

The Change history tables describe the technical changes between released issues of this document in reverse order. Issue numbers match the revision history in Document release information on page 1046.

Table 2: Issue 0000-01

Change	Location
First Beta release for r0p0	-

Table 3: Differences between issue 0000-01 and issue 0000-02

Change	Location
First confidential limited access release for r0p0	-
Editorial changes	Throughout document
Updated parameter definition of PE	1.7 Core, complex, SME2 unit, and processing element numbering on page 28
Updated signal names in the figure	2.4 Interfaces on page 38
Clarification on when to use WARM_RST mode	4.10.5 Warm reset mode on page 85
Clarifications	4.3.2 Off mode (OFF) on page 54
Clarifications	4.10.3 Emulated off mode on page 84
Updated with information about Debug recovery and Warm reset	4.10.5 Warm reset mode on page 85
Updated step 2 to drive nRESET LOW for a minimum of three PPUCLK cycles.	5.8.5 Cold reset of the whole cluster, including the PPUs, retaining cache contents for debug on page 110
In Method 1, step 5, clarification of the conditions for not setting the core to DBG_RECOV power mode.	5.8.6 Reset of the cluster, excluding the PPUs, retaining the cache contents for debug on page 111
Clarification on the number of bits the PARTID field supports.	6.3 Memory System Resource Partitioning and Monitoring control on page 121
Updated the AXI read and write address IDs to 2-bits.	8.2.3 AXI id bit setting on page 152
Clarifications for INCR burst for more than one transfer, transactions marked as FIXED, and write transfers with none, some, or all byte strobes being LOW.	10.5 AXI 64-bit peripheral port transactions on page 175
Clarifications for SME2 unit with Embedded Cross Trigger	14.7 Embedded Cross Trigger overview on page 220
Clarifications added for SME2 unit	14.7.1 CTI triggers on page 221
Added in the SQNAP bit field and clarifications for DCC and EFC bit fields.	A.2.5 IMP_CLUSTERECTLR_EL1, Cluster Extended Control Register on page 313 and B.1.1.11 CLUSTERECTLR, Cluster Extended Control Register on page 413
Clarifications added for QNAP bit field. Clarifications for SME2 unit on various bit fields.	A.2.6 IMP_CLUSTERPWRCTLR_EL1, Cluster Power Control Register on page 318 and B.1.1.3 CLUSTERPWRCTLR, Cluster Power Control Register on page 394
SLC bit field updated to 2 bits.	A.2.8 IMP_CLUSTERPWRSTAT_EL1, Cluster Power Status Register on page 326
Clarifications to PF bit	A.2.13 IMP_CLUSTERBUSQOS_EL1, Cluster Bus QoS Control Register on page 336 and B.1.1.8 CLUSTERBUSQOS, Cluster Bus QoS Control Register on page 404
MPAM register descriptions updated to conform to new architectural specifications	B.1.2 External MPAM registers summary on page 423
Cluster RAS descriptions updated to conform to new architectural specifications	B.1.3 External cluster RAS registers summary on page 458

Change	Location
Renamed prefix to CLUSTERCTL for register names	B.1.6 External CLUSTERCTL registers summary on page 630 and throughout the document
CTI register descriptions clarified and updated.	B.2.1 External cluster, core, and SME2 unit CTI registers summary on page 705
Cluster ROM registers descriptions clarified	B.2.2 External cluster ROM registers summary on page 808
Debug ROM registers descriptions clarified	B.2.3 External debug ROM registers summary on page 920

Table 4: Differences between issue 0000-02 and issue 0001-03

Change	Location
First confidential early access release for r0p1	-
Editorial changes	Throughout document
Added the revision entry for r0p1	1.8 Product revisions on page 30
Updated the cluster interfaces protocol description for AMBA AXI and DebugBlock interfaces image	2.4 Interfaces on page 38
Updated the conditions related to SME2 unit PPU programming	4.3.1 On mode (ON) on page 54
Updated the register access on the core PPU	5.3 Utility bus accesses on page 93
Updated the recommendation on using RMR for Warm reset	5.8.2 Core software initiated Warm reset of an individual core on page 106
Updated the list on CHI bitfield encodings	7.8 CHI transactions on page 143
Added a new section	9.6 ACP user bits on page 168
Updated the condition on SME2 unit being powered down	12.2.2 Cluster, core, and SME2 unit PPU register access on page 201
Updated the RAM bit description	14.1 Cache debug on page 207
Updated the CLUSTERCDBG table on RAM access	14.1 Cache debug on page 207
Updated the block diagram	14.6 DebugBlock subcomponents on page 218
Updated the PID and other columns in the table.	14.10 CoreSight component identification on page 226

Table 5: Differences between issue 0001-03 and issue 0002-04

Change	Location
First confidential early access release for r0p2	-
Editorial changes	Throughout document
Added the revision entry for r0p2	1.8 Product revisions on page 30
Updated the recommendation on using RMR for Warm reset	5.8.2 Core software initiated Warm reset of an individual core on page 106
Updated the condition on SME2 unit being powered down	12.2.2 Cluster, core, and SME2 unit PPU register access on page 201
Updated the block diagram	14.6 DebugBlock subcomponents on page 218

Table 6: Differences between issue 0002-04 and issue 0002-05

Change	Location
First non-confidential early access release for r0p2	-
Updated for product name changes	Throughout 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.

Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
italic	Citations.
bold	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 underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
SMALL CAPITALS	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Caution

We recommend the following. If you do not follow these recommendations your system might not work.

Warning

Your system requires the following. If you do not follow these requirements your system will not work.

Danger

You are at risk of causing permanent damage to your system or your equipment, or of harming yourself.

Note

This information is important and needs your attention.

Tip

This information might help you perform a task in an easier, better, or faster way.

Remember

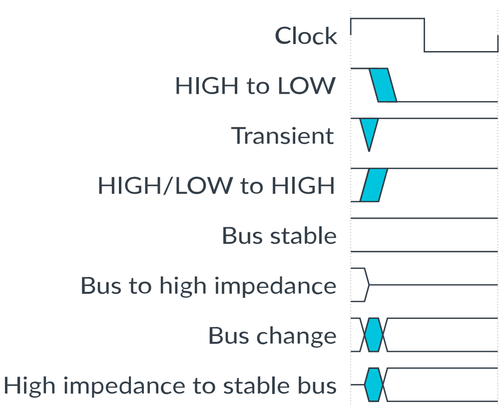
This information reminds you of something important relating to the current content.

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.

Register descriptions

Reset definitions

Replication Operator {}

Verilog replication operators are used for reset values over 8-bits.

For example, `{16{1'b0}}` indicates a binary value of 16 zeros.

x

Resets that are unknown are indicated with `x`.

Useful resources

This document contains information that is specific to this product. See the following resources for other useful information.

Arm documents are available on developer.arm.com/documentation.

Confidential documents are only available to licensees, when logged in. Each document link in the tables below provides direct access to the online version of the document.

Table 1: Arm publications

Arm product resources	Document ID	Confidentiality
Arm® C1-DynamIQ™ Shared Unit Configuration and Integration Manual	107805	Confidential
Arm® C1-Scalable Matrix Extension 2 Configuration and Integration Manual	107832	Confidential
Arm® C1-Scalable Matrix Extension 2 Technical Reference Manual	107831	Non-confidential
Arm® Corelink® PCK-600 Power Control Kit Technical Reference Manual	101150	Non-confidential
Arm® CoreSight™ DAP-Lite2 Technical Reference Manual	100572	Non-confidential
Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual	101088	Non-confidential
Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual	100806	Non-confidential

Arm architecture and specifications	Document ID	Confidentiality
AMBA® APB Protocol Specification	IHI 0024	Non-Confidential
AMBA® ATB Protocol Specification	IHI 0032	Non-Confidential
AMBA® AXI Protocol Specification	IHI 0022	Non-Confidential
Arm® Embedded Trace Macrocell Architecture Specification ETMv4.0 to ETM4.6	IHI 0064	Non-confidential
AMBA® CHI Architecture Specification	IHI 0050	Non-Confidential
Arm® Architecture Reference Manual for A-profile architecture	DDI 0487	Non-Confidential
Arm® CoreSight™ Architecture Specification v3.0	IHI 0029	Non-Confidential
Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4	IHI 0069	Non-Confidential
Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification	IHI 0099	Non-Confidential
Arm® Power Control System Architecture	DEN 0050	Non-confidential
Arm® Power Policy Unit Architecture Specification	DEN 0051	Non-confidential
Arm® Reliability, Availability, and Serviceability (RAS) System Architecture	IHI 0100	Non-Confidential

Non-Arm resources	Document ID	Organization
IEEE, Standard for Access and Control of Instrumentation Embedded within a Semiconductor Device	1687-2014	IEEE www.ieee.org
IEEE, Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems	1801-2018	IEEE www.ieee.org
IEEE, Standard for Test Access Port and Boundary-Scan Architecture	1149.1-2013	IEEE www.ieee.org