



# Arm® Neoverse™ N2 reference design

Revision: Release D

## Technical Overview

### Non-Confidential

Copyright © 2021–2022 Arm Limited (or its affiliates).  
All rights reserved.

### Issue 04

102337\_0000\_04\_en

# Arm® Neoverse™ N2 reference design

## Technical Overview

Copyright © 2021–2022 Arm Limited (or its affiliates). All rights reserved.

## Release information

### Document history

Issue	Date	Confidentiality	Change
0000-01	25 January 2021	Confidential	Initial release
0000-02	12 May 2021	Confidential	Second release
0000-03	4 October 2021	Confidential	Third release
0000-04	15 January 2022	Non-Confidential	Fourth release

## Proprietary Notice

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

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

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

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND

REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

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

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

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

Copyright © 2021–2022 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

## Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

## Product Status

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

## Feedback

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

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

## Inclusive language commitment

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

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

To report offensive language in this document, email [terms@arm.com](mailto:terms@arm.com).

# Contents

<b>1 Introduction.....</b>	<b>10</b>
1.1 Product revision status.....	10
1.2 Intended audience.....	10
1.3 Conventions.....	10
1.4 Additional reading.....	12
<b>2 Overview of Neoverse N2 reference design.....</b>	<b>15</b>
2.1 Product documentation.....	16
2.2 Compliance.....	16
<b>3 Hardware and topology.....</b>	<b>17</b>
3.1 IP in RD-N2.....	17
3.2 System architecture.....	17
3.3 Interconnect block.....	20
3.3.1 CMN-700 Coherent Mesh Network.....	23
3.3.2 NI-700 Non-coherent Interconnect.....	26
3.3.3 NIC-450 Network Interconnect.....	27
3.4 Processor block.....	27
3.5 Interrupt block.....	30
3.5.1 GIC Interrupt Translation Service options.....	31
3.5.2 I/O Virtualization block.....	34
3.7 Debug block (debug and trace).....	38
3.7.1 External debugger connectivity.....	39
3.7.2 Debug authentication.....	40
3.7.3 System debug support.....	41
3.7.4 Application core debug support.....	42
3.7.5 SCP core debug support.....	43
3.7.6 MCP core debug support.....	45
3.7.7 ROM tables.....	46
3.7.8 Trace sources.....	48
3.7.9 Cross triggers.....	49
3.7.10 System Trace Macrocell.....	50
3.7.11 Self-hosted debug.....	52

3.7.12 Debug through functional I/O interfaces.....	53
3.8 MSCP block.....	53
3.8.1 System Control Processor block.....	54
3.8.2 Manageability Control Processor block.....	59
3.8.3 Message communication between processors.....	61
3.9 Clock Control block.....	62
3.10 Peripheral block.....	63
3.11 Dynamic Memory block.....	64
<b>4 Functional description.....</b>	<b>67</b>
4.1 Clocks.....	67
4.1.1 Input clocks.....	67
4.1.2 Internal clocks.....	68
4.1.3 Output clocks.....	71
4.1.4 PLL lock control.....	72
4.1.5 Clocks by functional block.....	72
4.2 Counters and timers.....	74
4.2.1 System generic counter.....	74
4.2.2 Generic timers.....	74
4.2.3 Watchdog timers.....	77
4.2.4 Power down considerations.....	81
4.3 System power management and domains.....	82
4.3.1 Power and voltage domains.....	83
4.3.2 Power and voltage domain hierarchy.....	86
4.3.3 Compute subsystem power states.....	88
4.3.4 SYSTOP reset control.....	90
4.3.5 PD_CPU <sub>n</sub> power control.....	91
4.3.6 SYSTOP domain reset control sequences.....	92
4.3.7 PD_CPU <sub>n</sub> power domain power control sequences.....	93
4.3.8 Core power and performance management.....	95
4.4 Resets.....	96
4.4.1 Input resets.....	97
4.4.2 Internal resets.....	98
4.4.3 Output resets.....	98
4.4.4 Power policy unit resets.....	99
4.4.5 SCP core Warm reset.....	99

4.4.6 MCP core Warm reset.....	99
4.4.7 Debug through reset.....	100
4.5 Top-level I/O interfaces.....	101
4.5.1 I/O coherent expansion slave interfaces (EXSACEL<0-n>, EXSACELD<0-n>).....	101
4.5.2 I/O coherent expansion slave interfaces with integrated inline TBUs (EXSACELT<0-n>)....	101
4.5.3 Expansion master interfaces (EXMACEL<0-n>, EXMAXI<0-n>).....	102
4.5.4 DFI master interfaces (EXMMEMDFI<0-n>).....	102
4.5.5 GIC expansion interrupts interface (EXGICINT).....	103
4.5.6 LTI expansion slave interfaces (EXSLTI<0-n>).....	103
4.5.7 ATS_DTI expansion slave interfaces (EXSATS<0-n>).....	103
4.5.8 MSI expansion slave interfaces (EXSMSI<0-n>).....	103
4.5.9 STM event expansion interfaces (EXSSTME<0-n>).....	104
4.5.10 CoreSight ATB slave trace expansion interfaces (EXSATB<0-n>).....	104
4.5.11 CoreSight APB master expansion interface (EXMDBGAPB).....	104
4.5.12 CoreSight Cross Trigger Interface (EXCTM).....	104
4.5.13 AMBA CXS expansion master and slave interfaces (EXSCXSB<0-n>, EXMCXSB<0-n>)....	105
4.5.14 Power Q-Channel or P-Channel expansion interfaces (EXPWRQ_DBGTOP<0-n>, EXPWRP_DBGTOP<0-n>, EXPWRQ_SYSTOP<0-n>, EXPWRP_SYSTOP<0-n>).....	105
4.5.15 SCP expansion interrupts (EXSCPINT).....	106
4.5.16 MCP expansion interrupts (EXMCPINT).....	106
4.5.17 SCP PIK expansion interfaces (EXSCPPIK<0-n>).....	106
4.5.18 SCP AON expansion interface (EXMSCPAXI).....	107
4.5.19 MCP AON expansion interface (EXMMCPAXI).....	107
4.5.20 AP, SCP, and MCP UART interfaces (APNSUART, APSUART, SCPUART, MCPUART)....	107
4.5.21 JTAG and SWD interface (JTAGIF).....	107
4.5.22 Debug authentication signals.....	108
4.5.23 Functional override inputs.....	108
4.5.24 Miscellaneous output signals.....	109
4.6 System Boot.....	109
4.6.1 Boot flow overview.....	110
4.7 Security.....	111
4.7.1 Application processor security.....	112
4.7.2 Interconnect block security.....	112
4.7.3 Peripheral block security.....	112
4.7.4 SCP block security.....	113
4.7.5 MCP block security.....	113

4.8 Multichip architecture.....	113
4.8.1 Host-to-host SMP multichip use case.....	115
4.8.2 Host-to-accelerator use case.....	126
4.8.3 Host-to-memory expansion use case.....	128
<b>5 Fixed Virtual Platform.....</b>	<b>129</b>
5.1 About the FVP.....	129
5.2 FVP peripherals.....	129
<b>6 Software stack.....</b>	<b>133</b>
6.1 About the software.....	133
6.2 SCP firmware.....	134
6.2.1 Power control.....	135
6.2.2 SCP boot ROM.....	135
6.3 MCP firmware.....	135
6.3.1 MCP boot ROM.....	135
6.4 Application processor firmware.....	136
6.4.1 Arm Trusted firmware BL1.....	136
6.4.2 Arm Trusted firmware BL2.....	136
6.4.3 Arm Trusted firmware BL31.....	136
6.4.4 Secure runtime services BL32.....	137
6.4.5 Bootloader BL33.....	137
6.5 Linux kernel.....	137
6.5.1 Multiprocessing.....	138
6.5.2 UEFI awareness.....	138
6.5.3 Device drivers.....	138
6.6 Multi-chiplet support.....	138
<b>7 Programmers model.....</b>	<b>139</b>
7.1 About the programmers model.....	139
7.2 Memory maps.....	139
7.2.1 AP system memory map.....	140
7.2.2 PCIe address mapping.....	153
7.2.3 SCP memory map.....	154
7.2.4 MCP memory map.....	156
7.2.5 Address translation table.....	159
7.2.6 Debug memory maps.....	159

7.3 Interrupt maps.....	165
7.3.1 PPI interrupt map.....	165
7.3.2 SCP NVIC interrupt map.....	171
7.3.3 MCP NVIC interrupt map.....	173
7.4 Register descriptions.....	174
7.4.1 System ID registers.....	175
7.4.2 REFCLK counter registers.....	184
7.4.3 Generic timer registers.....	204
7.4.4 System generic timer synchronization registers.....	218
7.4.5 Application Processor Watchdog timer registers.....	234
7.4.6 Base SRAM ECC RAS registers.....	249
7.4.7 Message Handling Unit registers.....	259
7.4.8 Core Manager and clock control registers.....	283
7.4.9 System Power Integration Kit registers.....	311
7.4.10 Debug Power Integration Kit registers.....	345
7.4.11 Debug Chain Power Control Logic registers.....	373
7.4.12 MSCP Power Control registers.....	386
7.4.13 DMC manager registers.....	434
7.4.14 PCIe integration control registers.....	438
<b>A Revisions.....</b>	<b>487</b>

# 1 Introduction

## 1.1 Product revision status

For details on the product revision status, see the Arm® Neoverse™ N2 reference design *Release Note*.

## 1.2 Intended audience

The Technical Overview is written for experienced hardware and System-on-Chip (SoC) engineers who might or might not have experience with Arm products. Such engineers typically have experience in writing Verilog and of performing synthesis, but might have limited experience of integrating and implementing Arm products.

## 1.3 Conventions

The following subsections describe conventions used in Arm documents.

### Glossary

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

See the Arm® Glossary for more information: [developer.arm.com/glossary](https://developer.arm.com/glossary).

### Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

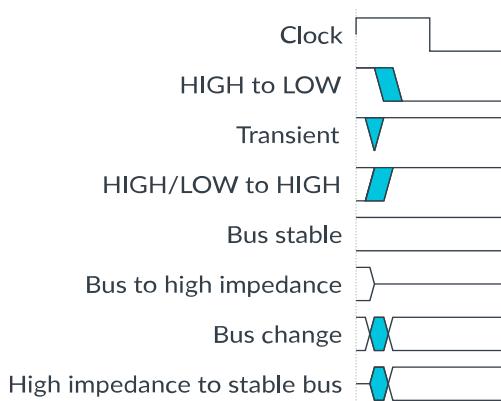
Convention	Use
<i>italic</i>	Citations.
<b>bold</b>	Interface elements, such as menu names.  Signal names.  Terms in descriptive lists, where appropriate.
<code>monospace</code>	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
<b>monospace bold</b>	Language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

Convention	Use
monospace <u>underline</u>	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments.  For example:  MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
<b>SMALL CAPITALS</b>	Terms that have specific technical meanings as defined in the Arm® Glossary. For example, <b>IMPLEMENTATION DEFINED</b> , <b>IMPLEMENTATION SPECIFIC</b> , <b>UNKNOWN</b> , and <b>UNPREDICTABLE</b> .
 Caution	Recommendations. Not following these recommendations might lead to system failure or damage.
 Warning	Requirements for the system. Not following these requirements might result in system failure or damage.
 Danger	Requirements for the system. Not following these requirements will result in system failure or damage.
 Note	An important piece of information that needs your attention.
 Tip	A useful tip that might make it easier, better or faster to perform a task.
 Remember	A reminder of something important that relates to the information you are reading.

## Timing diagrams

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

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

**Figure 1-1: Key to timing diagram conventions**

## Signals

The signal conventions are:

### Signal level

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

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

### Lowercase n

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

## 1.4 Additional reading

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

**Table 1-2: Arm publications**

Document Name	Document ID	Licensee only
AMBA® 4 AXI-Stream Protocol Specification	IHI 0051	No
AMBA® 5 CHI Architecture Specification, issue C	IHI 0050C	No
AMBA® 5 CHI Architecture Specification, issue D	IHI 0050D	No
AMBA® 5 CHI Architecture Specification, issue E	IHI 0050E	No
AMBA® AXI and ACE Protocol Specification	IHI 0022	No
AMBA® CXS Protocol Specification	IHI 0079	No
AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces	IHI 0068	No

Document Name	Document ID	Licensee only
Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile	DDI 0487	No
Arm® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual	DDI 0475	No
Arm® CoreSight™ Base System Architecture, version 1.0	DEN 0068	No
Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual	100806	No
Arm® Cortex®-M7 Processor Technical Reference Manual	DDI 0489	No
Arm® Debug Interface Architecture Specification ADIv6.0	IHI 0074C	No
Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4	IHI 0069	No
Arm® Neoverse™ CMN-700 Coherent Mesh Network Technical Reference Manual	102308	Yes
Arm® Neoverse™ N2 Core Technical Reference Manual	102099	No
Arm® Neoverse™ N2 reference design Analysis Report	PJDOC-1505342170-533091	Yes
Arm® Neoverse™ N2 reference design Implementation Guidelines	PJDOC-1505342170-534961	Yes
Arm® Neoverse™ N2 reference design Release Note	PJDOC-1505342170-533248	Yes
Arm® Neoverse™ N2 reference design System Design	PJDOC-1505342170-531272	Yes
Arm® Power Control System Architecture, version 2.0	DEN 0050C	Yes
Arm® Power Policy Unit Architecture Specification, version 1.1	DEN 0051E	No
Arm® Server Base System Architecture, version 6.0	DEN 0029C	No
Arm® System Memory Management Unit Architecture Specification, SMMU architecture version 3	IHI 0070	No
Arm® Theodul DynamIQ™ Shared Unit Technical Reference Manual	101381	Yes
Arm®v7-M Architecture Reference Manual	DDI 0403	No
CoreSight™ Components Technical Reference Manual	DDI 0314	No
CoreSight™ System Trace Macrocell Technical Reference Manual	DDI 0444	No
Principles of Arm® Memory Maps White Paper	DEN 0001C	No

**Note**

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

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

## 2 Overview of Neoverse N2 reference design

Arm® Neoverse™ N2 reference design (RD-N2) is intended for Silicon Partners (SiPs) that want to create high core-count designs targeted at 5G, enterprise networking, SmartNIC, and cloud computing server applications.

This reference design provides recommended configurations to address Power, Performance, and Area (PPA) targets and other key requirements that are specific to these markets.

RD-N2 provides the following features:

- Thirty-two MP1 Armv9.0-A Arm® Neoverse™ N2 cores with Direct connect and 1MB of dedicated, private L2 cache for each core
- Arm Neoverse CMN-700 6 x 6 mesh interconnect with 32MB System Level Cache (SLC) and 64MB Snoop Filter (SF)
- Eight CCIX 2.0 and CXL 1.1 ports for connections to in-package and off-package accelerators
- Eight CCIX 2.0 links to support in-package die-to-die Multichip Module (MCM) or socket-to-socket Symmetric Multiprocessing (SMP) use cases
- Cache-coherent mesh interconnect with Arm® CoreLink™ NIC-450 Network Interconnect and Arm CoreLink NI-700 Network-on-Chip Interconnect non-coherent interconnects to connect system IP
- Configurations with either eight or four DDR5-5600 40-bit memory interfaces supporting dual-channel DIMM
- System Control Processor (SCP) and Manageability Control Processor (MCP) based on the Arm® Cortex®-M7 processor
- Arm® CoreSight™ debug and trace support
- Targets a representative 5nm process

There are two configurations for RD-N2:

### **CFG32C8M**

This configuration includes eight DDR5 40-bit memory interfaces for dual-channel DIMM.

### **CFG32C4M**

This configuration includes four DDR5 40-bit memory interfaces for dual-channel DIMM.

The memory interfaces are enabled on only one side of the mesh interconnect.

There are no other differences between the two configurations.

## 2.1 Product documentation

RD-N2 includes a set of documentation.

The documentation for RD-N2 comprises:

- *Arm® Neoverse™ N2 reference design System Design*

Introduces RD-N2, giving an overview of its features, architecture, configurations, and performance.

- *Arm® Neoverse™ N2 reference design Technical Overview*

Provides a high-level overview of RD-N2, including the system architecture and the functional description.

- *Arm® Neoverse™ N2 reference design Analysis Report*

Summarizes the performance characteristics of RD-N2, based on modeling, testing, and analysis of the system.

- *Arm® Neoverse™ N2 reference design Implementation Guidelines*

Contains information about the physical implementation of the RD-N2 design.

## 2.2 Compliance

RD-N2 complies with, or includes components that comply with, the following specifications:

- *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile*
- *Arm® Generic Interrupt Controller Architecture Specification, GIC architecture version 3 and version 4*
- *AMBA® 5 CHI Architecture Specification, issue E*
- *Arm® Server Base System Architecture, version 6.0*
- *AMBA® AXI and ACE Protocol Specification*
- *Arm® System Memory Management Unit Architecture Specification, SMMU architecture version 3*
- *Arm® CoreSight™ Base System Architecture, version 1.0*
- *Arm® Power Policy Unit Architecture Specification, version 1.1*
- *Arm® Power Control System Architecture, version 2.0*
- *Arm® Debug Interface Architecture Specification ADIv6.0*

# 3 Hardware and topology

The architecture of RD-N2 supports the design of compute subsystems for infrastructure market segments such as hyperscale data centers, networking, and enterprise storage.

## 3.1 IP in RD-N2

The IP and components in RD-N2 provide a design from which to create a custom SoC.

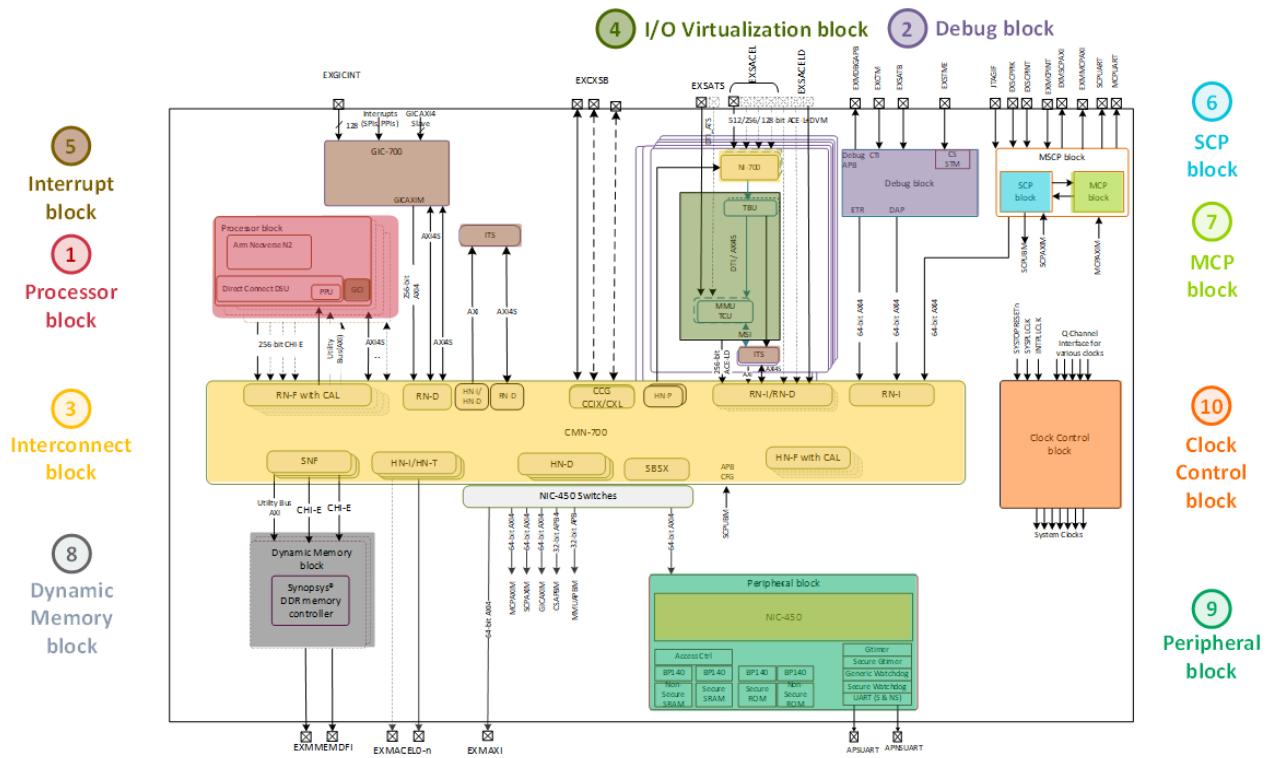
The RD-N2 subsystem contains the following IP:

- Armv9.0-A Arm Neoverse N2 cores
- Arm Neoverse CMN-700 Coherent Mesh Network
- Arm Cortex-M7 System Control Processor
- Arm Cortex-M7 Manageability Control Processor
- Arm CoreLink GIC-700 Generic Interrupt Controller
- Arm CoreLink MMU-700 System Memory Management Unit
- Arm CoreLink PCK-600 Power Control Kit
- Arm CoreLink NI-700 Network-on-Chip Interconnect
- Arm CoreLink NIC-450 Network Interconnect
- Arm® CoreLink™ ADB-400 AMBA® Domain Bridge
- Arm CoreSight System-on-Chip SoC-600
- Arm CoreSight STM-500 System Trace Macrocell
- Clock generators with support for dynamic clock gating
- On-chip ROM, RAM, and other peripherals
- Third-party DDR5-5600 memory controllers

## 3.2 System architecture

RD-N2 is partitioned into functional blocks that are a combination of major IP and the supporting logic around them. Some features of the design incorporate functionality from multiple blocks. This block-based design approach provides flexibility, scalability, and modularity.

The following figure shows the top-level architecture of RD-N2.

**Figure 3-1: RD-N2 architecture block diagram**

The functional blocks in RD-N2 are:

### Interconnect block

Includes coherent and non-coherent interconnects.

CMN-700 is a mesh-based coherent interconnect with multichip support using the CCIX standard.

NI-700 connects to the external I/O masters and provides expansion interfaces for CMN-700.

NIC-450 interfaces connect to system peripherals, rest-of-SoC components, and other subsystem connectivity.

Contains other very low performance interconnects such as AMBA AXI4-Stream Interconnect (IC) and Advanced Peripheral Bus (APB) IC.

For more information, see [Interconnect block](#).

### Processor block

Includes Arm Neoverse N2 cores with Direct connect DynamIQ Shared Unit (DSU) and power management components.

Configured with one core plus bridge per cluster (MP1).

RD-N2 implements 32 instances of the Processor block.

For more information, see [Processor block](#).

## Interrupt block

Handles interrupts in the system.

Includes distributed GIC-700 components, such as GIC Distributor and Redistributor, and Interrupt Translation Services (ITS).

For more information, see [Interrupt block](#).

## I/O Virtualization block

Includes distributed Translation Buffer Unit (TBU) and Translation Control Unit (TCU) components, which provide I/O virtualization for external I/O masters.

For more information, see [I/O Virtualization block](#).

## Debug block

Supports CoreSight debug and trace in all cores in the compute subsystem, in the mesh network, and in other system IP.

For more information, see [Debug block \(debug and trace\)](#).

## MSCP block

Includes a System Control Processor and a Manageability Control Processor, both based on the Cortex-M7 processor.

The SCP controls functions such as boot, reset, clock, and power management.

The MCP handles communication with the Baseboard Manageability Controller (BMC).

Supports the MHUv2.1 inter-processor communication method.

Includes a PCK-600 Power Control Kit that is distributed across the subsystem to support different reset and power domains.

Provides expansion interfaces for adding SoC peripherals such as Shared Peripheral Interrupt (SPI) and I2C.

For more information, see [MSCP block](#).

## Clock Control block

Includes logic to generate the compute subsystem clocks.

Separate clock control logic is used to generate clocks for each core, for the debug logic, and for the memory controllers.

For more information, see [Clock Control block](#).

## Peripheral block

Includes watchdog timers, Secure and Non-secure generic timers, and Secure and Non-secure Universal Asynchronous Receiver-Transmitters (UARTs).

Also contains scratch RAM, boot ROM, and firmware ROM.

For more information, see [Peripheral block](#).

## Dynamic Memory block

Contains the third-party memory controller.

For more information, see [Dynamic Memory block](#).

## 3.3 Interconnect block

The Interconnect block in RD-N2 supports coherent and non-coherent accesses.

### Coherent interconnect

[CMN-700](#) is a mesh-based coherent interconnect with support for AMBA 5 CHI-E, AXI-H, and memory partitioning.

### Non-coherent interconnects

[NI-700](#) connects to the external I/O masters and provides expansion interfaces for CMN-700

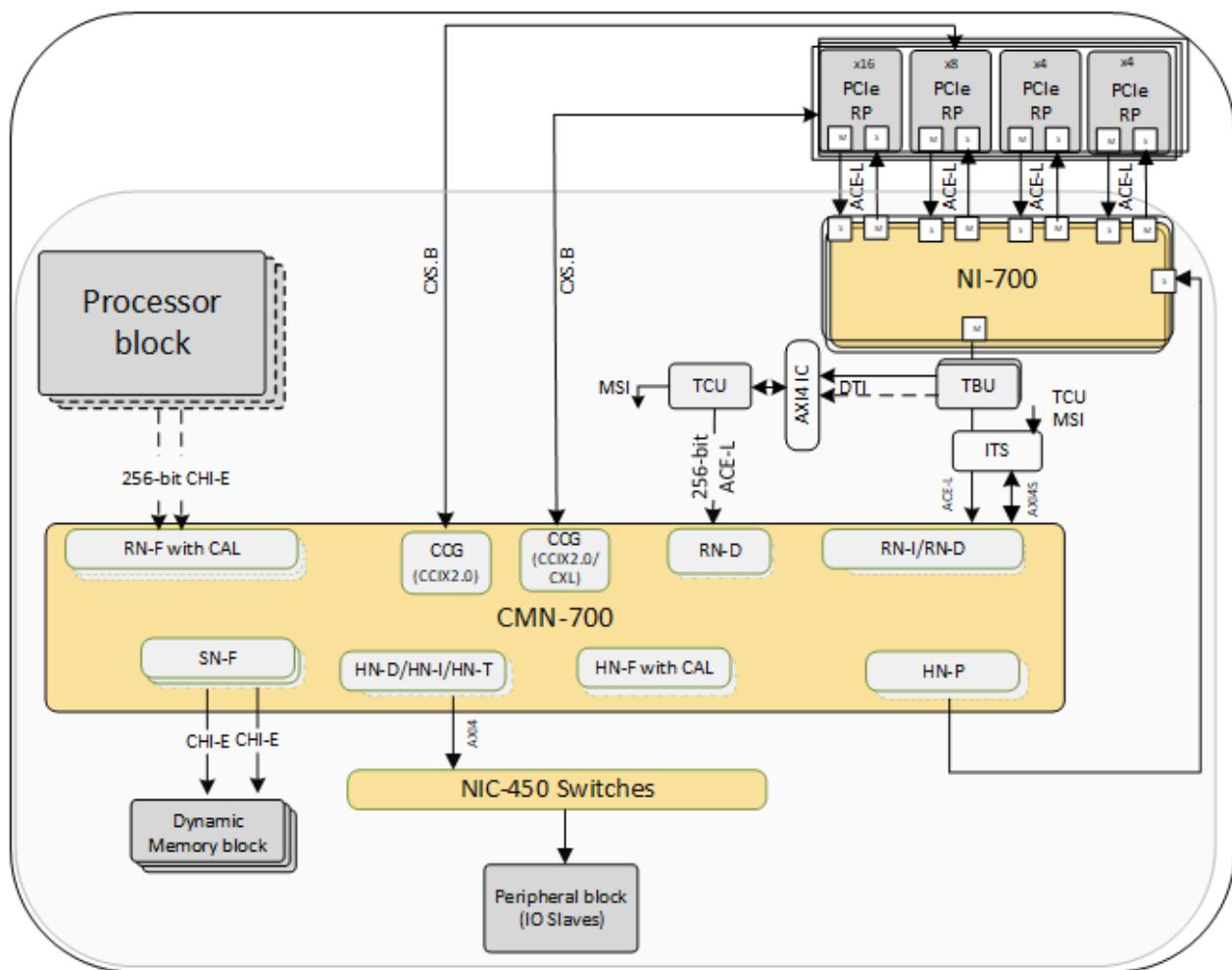
[NIC-450](#) provides interfaces to connect system peripherals, rest-of-SoC components, and other subsystem connectivity.

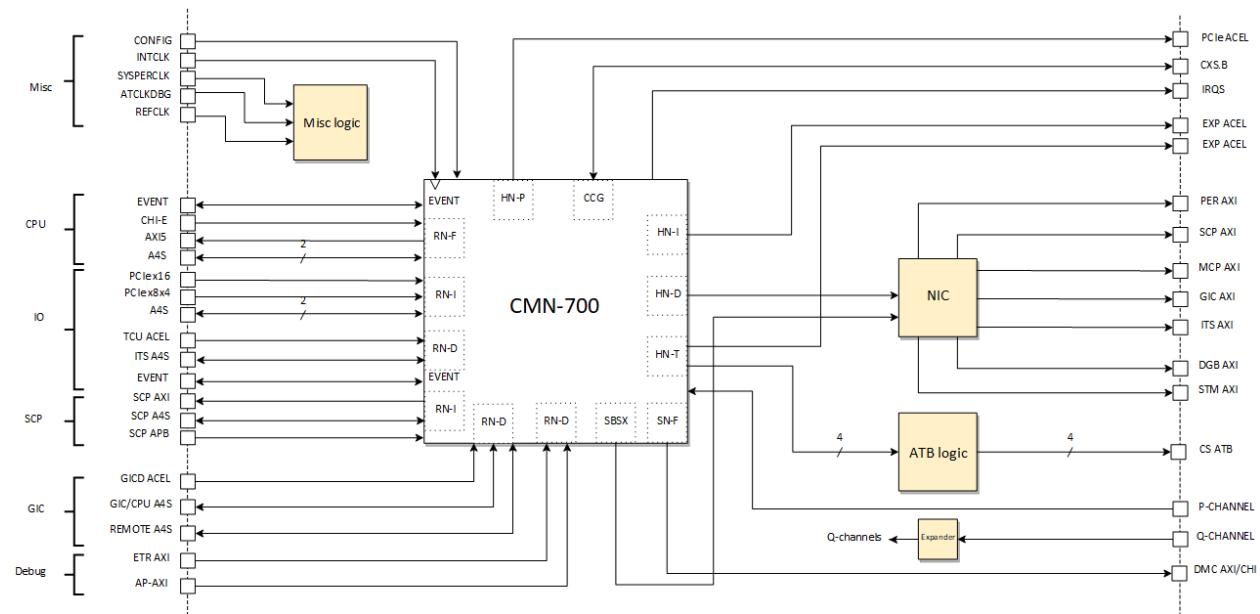
Other very low performance interconnects, such as AMBA AXI4-Stream Interconnect (IC), APB IC.

The Interconnect block contains:

- A coherent mesh network with coherent multichip support using the CCIX and CXL standard
- I/O master and slave extension interfaces
- NIC-450 switches and expanders for Q-Channels
- Miscellaneous blocks, such as clock domain bridges, power management, pipeline slices, and other wrapper glue logic

The following figures illustrate the block diagrams of the Interconnect block interfaces and key IP blocks that are connected to those interfaces.

**Figure 3-2: Connectivity of key IP blocks with the Interconnect block**

**Figure 3-3: CMN-700 interconnect interfaces**

**Note** The final logical structure could differ significantly due to restructuring of the RTL that removes the hierarchical boundaries in favor of physical boundaries. This sort of change might be made to simplify development of the backend implementation flow, such as using a tile-based approach.

The CMN-700 mesh network constitutes the largest part of the overall subsystem. It enables the connection of blocks that have different interfacing requirements to the Interconnect block. Examples of components that have differing interface requirements are the Processor block, memory controllers, and external I/O masters.

The coherent mesh network supports the following interface protocols:

- AMBA® 5 CHI Architecture Specification, issue C
- AMBA® 5 CHI Architecture Specification, issue D
- AMBA® 5 CHI Architecture Specification, issue E
- AMBA ACE-Lite
- AMBA® AXI and ACE Protocol Specification
- AMBA® 4 AXI-Stream Protocol Specification
- AMBA® CXS Protocol Specification

Ancillary components such as the internal NIC provide fine grain decoding of the system address map. These components interface to modules that operate in different clock domains and support dynamic control of clocks that can be gated to reduce power.

### 3.3.1 CMN-700 Coherent Mesh Network

In RD-N2, the CMN-700 implementation addresses the PPA requirements of designs with a high core count. This implementation is targeted at the networking and data center markets.

In the RD-N2 design, CMN-700 implements the following features:

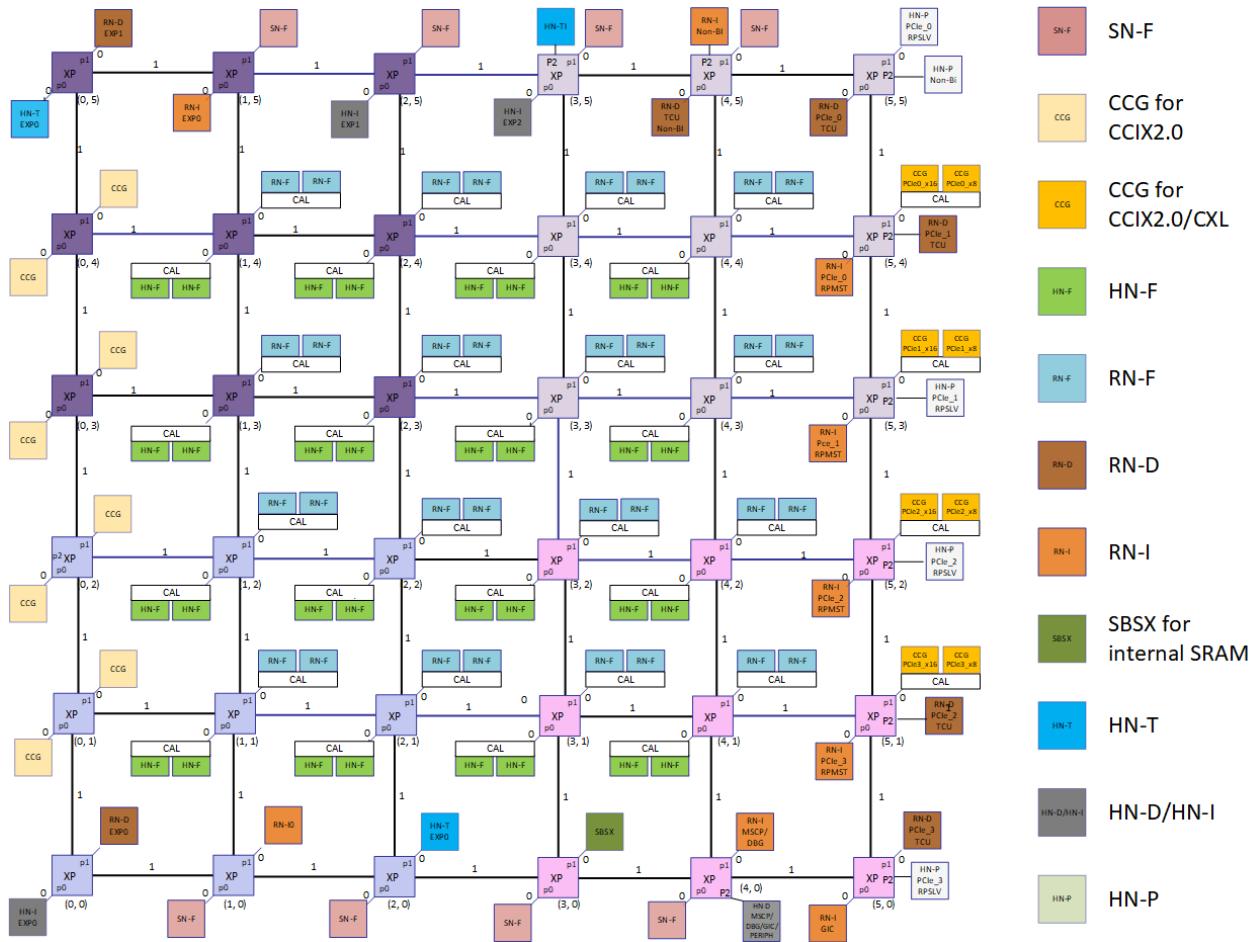
- A mesh size of 6 x 6 for both the CFG32C8M and CFG32C4M configurations. The mesh includes 32 Fully coherent Request Nodes (RN-Fs) with support for a Component Aggregation Layer that can connect to two devices (CAL2).
- An SLC of 32MB. The mesh includes 32 Fully coherent Home Nodes (HN-Fs) with CALs. Each HN-F has a cache size of 1MB.
- In the CFG32C8M configuration, eight Fully coherent Slave Nodes (SN-Fs) for connections to the third-party memory controllers over the native AMBA CHI interface.
- The AMBA interfaces from these nodes are connected to components within the subsystem or are exported as expansion interfaces.
- Configured with a single Data (DAT) and Response (RSP) channel.

CMN-700 also has the following features:

- Support for multiple types of nodes, such as:
  - I/O coherent Request Nodes (RN-Is).
  - I/O coherent Request Nodes with DVM support (RN-Ds).
  - I/O coherent Home Nodes (HN-Is).
  - I/O coherent Home Nodes with PCIe optimization (HN-Ps).
  - I/O coherent Home Nodes with Debug Trace Controller (HN-Ts).
- Support for Configurable Mesh Credit Slices (MCSs), Device Credit Slices (DCSs), and CAL Credit Slices (CCSs).
- Fully compliant with AMBA® 5 CHI Architecture Specification, issue E and AMBA® AXI and ACE Protocol Specification, issue H to optimize coherency and Quality of Service (QoS).
- Configurable option to support all SLC and SF sizes that are available for the mesh to meet various RD-N2 requirements.
- Support for System Cache Group (SCG) and striping capabilities on SLC partitions and memory interfaces.
- Support for Armv9 features.
- Configurable Memory Partitioning and Monitoring (MPAM) for SLC capacity allocation.
- Support for all system-level and interconnect-level QoS features, including Completer Busy (CBusy). Configurable CBusy options are also supported.
- Support for the configurable Request Node System Address Map (RN SAM) QoS override feature.
- Support for Reliability, Availability, and Serviceability (RAS) features and the Performance Monitoring Unit (PMU).

- Support for Debug and Trace (DT).
- Configurable support for static route optimizations around mesh hotspots that override the default XY route path between specific Crosspoints (XPs).
- Device ports with CALs and support for more device ports at the boundary and edge XPs for PPA optimization.
- Support for the CMN-700 Utility Bus (PUB) interface, which is a generic transport layer in the mesh. Request and response fields are left for definition and processing by requestor and responder.
  - Support for PUB with bridges to connect external AMBA AXI4-Stream and AXI devices outside of the CMN-700 interconnect.
  - Support for use of PUB channels to transfer messages between distributed GIC components.
- Support for configurable bus width options, such as 128 bits, 256 bits, and 512 bits on I/O ports.
- Support for Coherent Multichip Link (CML) port-to-port forwarding, which allows the mesh to act as a bridge between two CCIX chips.
- Support for chip-to-chip communication.
  - Eight CML links that support both the CCIX 2.0 and CXL 1.1 protocols for connections to accelerators.
  - Eight CML links with CCIX 2.0 support for chip-to-chip or socket-to-socket connections.

The following figure shows the 6 x 6 CMN-700 interconnect mesh network that is used in RD-N2.

**Figure 3-4: RD-N2 6 x 6 CMN-700 system mesh topology**

The following selections determine the CMN-700 mesh structure:

### Requesting master selection

The number of masters with coherent caches in the SoC determines the number of RN-F ports that are required.

The number of masters without internal coherent caches in the SoC determines the number of RN-D/RN-I components that are required.

### Home Node selection

The SLC and SF size requirements determine the number of HN-F instances. The appropriate number of HN-F nodes are selected in the system according to the performance and area requirements.

The total AMBA AXI slave bandwidth requirements and the physical placement of these slave peripherals determine the number of HN-D and HN-I instances.



The HN-I does not support caching of data read from or written to the downstream AMBA interface I/O slave block. Therefore, any cacheable request that is sent to the HN-I is converted to the appropriate AMBA AXI read or write command and sent to the downstream AMBA interface block. No snoops are sent to RN-Fs in the system. If an RN-F caches data read from or written to the downstream AMBA interface I/O slave block, that data is not cached in the SLC. In this case, hardware coherency is not maintained. Any subsequent access to that data is read from or written to the AMBA interface I/O slave block directly, ignoring the cached data. The data can be cached in Processor block caches.

### 3.3.2 NI-700 Non-coherent Interconnect

NI-700 is used for PCIe system integration to support the PCIe bifurcation feature in RD-N2. The interconnect is configured according to the overall PCIe system functional and performance requirements, bus widths, and queue sizes to meet the PPA requirements.

NI-700 provides:

- A Network on Chip (NoC) based architecture
- Highly configurable and scalable interconnect that extends to higher bandwidths
- A packetized transport layer that allows serialization to reduce wire counts
- Protocol conversion on entry and exit from the AMBA protocol to internal transport for ease of implementation
- Native support for AMBA AXI5, ACE5-Lite, and AHB5 interfaces
- Highly configurable Quality of Service (QoS) features
- Flexible tree or mesh topology
- Reduced routing and area, with easier timing closure
- Configurable options to create a high-performance AMBA protocol-compliant network infrastructure for non-coherent masters, including:
  - Multiple AMBA protocols, such as AXI, AHB, APB, AXI4-Stream
  - Multiple clock domains
  - Single power domain
  - Single voltage domain
  - Variable link data widths
  - Up to 256 slave interfaces
  - Up to 255 master interfaces
  - Hierarchical clock gating
  - Support for complex topologies
  - Non-blocking resource planes

The NI-700 topology in RD-N2 includes multilane PCIe I/O masters with bifurcation modes. The two four-lane, one eight-lane, and one 16-lane PCIe I/O masters are connected through NI-700 to the main CMN-700 interconnect, see [Figure 3-8: Example GIC topology with distributed in-line ITS connectivity](#) on page 33.

PCIe bifurcation involves splitting a single PCIe port into two or more ports with reduced lane widths.

The system functional and performance requirements determine:

- The number of non-coherent interconnects
- The bus width selections
- The number of AMBA AXI Master Network Interfaces (AMNIIs) and AMBA AXI Slave Network Interfaces (ASNIs) that are used in each non-coherent interconnect

### 3.3.3 NIC-450 Network Interconnect

Arm Neoverse N2 reference design includes the NIC-450 network interconnect.

NIC-450 provides:

- A network of AMBA AXI cross bars or switches
- Reduced latency, without the protocol conversion that is associated with Network-on-Chip (NoC) architectures
- A flexible tree topology
- Master and slave extension interfaces for low-bandwidth, low-latency connectivity

In Arm Neoverse N2 reference design, NIC-450 is used to provide connectivity for both internal and external compute subsystem interfaces. NIC-450-based external interfaces enable connection of external masters and slaves to the compute subsystem. For example, NIC-450 enables extension of the interfaces by grouping multiple low-performance peripheral interface connections on one CMN-700 HN-D/HN-I device port.

## 3.4 Processor block

Each Processor block contains a high-performance Neoverse N2 core, a direct connect DSU, the GIC Cluster Interface (GCI), debug, clock, and reset control blocks. The direct connect DSU, with minimal DSU logic, contains asynchronous interface bridges, power management PPUs, and debug logic.

The RD-N2 design has 32 Processor blocks. The cores in the Processor block are also referred to as the Application Processor (AP).

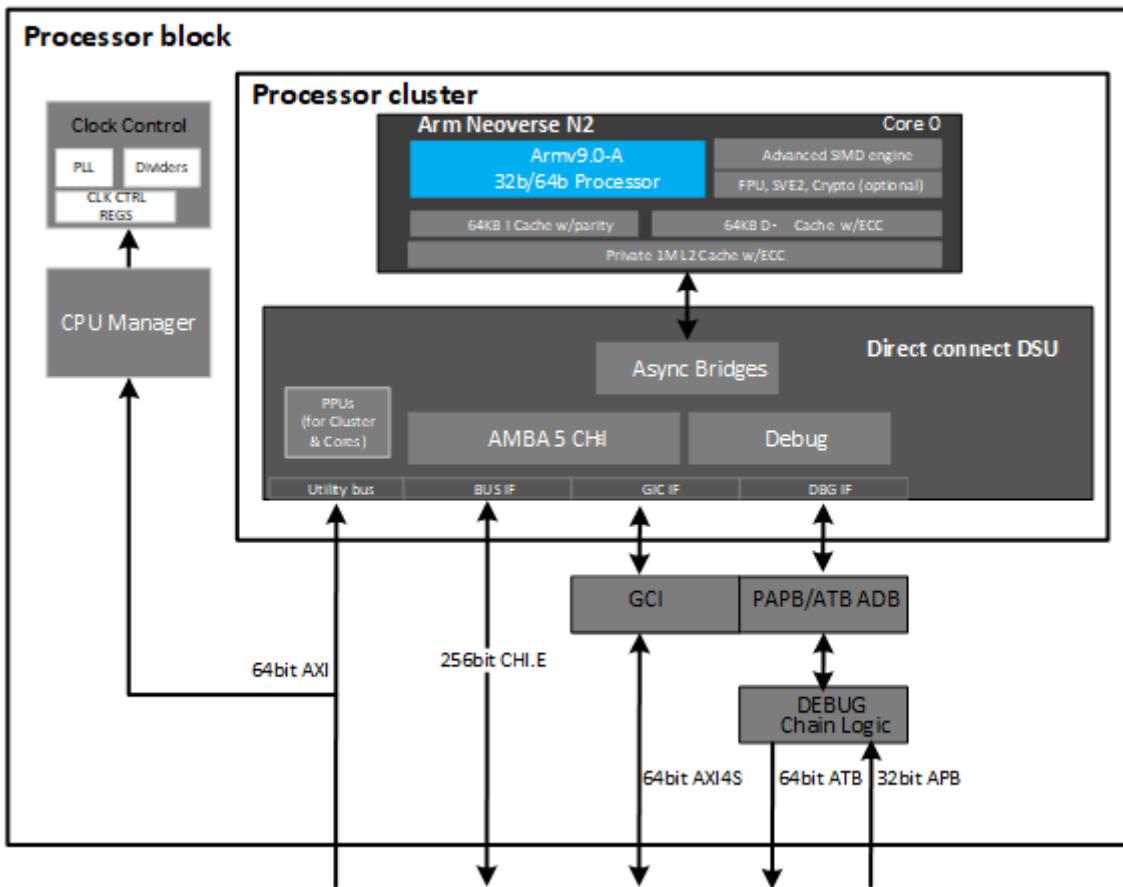
The core is configured in direct connect mode. There is no L3 cache or snoop control logic. There is one core in each cluster, directly connected to the CMN-700 interconnect through a CAL

(Component Aggregation Layer). This connection method is used for MP1 clusters that support direct connect with a coherent AMBA CHI interconnect.

The Processor block includes:

- One MP1 Arm Neoverse N2 core, which supports the following architectural features:
  - Armv9.0-A - 48-bit Physical Address (PA) and 48-bit Virtual Address (VA).
  - Armv9.0-A - Scalable Vector Extensions 2 (SVE2).
  - Armv9.0-A - SVE2 Cryptographic Extensions supporting SHA2-512, Advanced Encryption Standard (AES), Secure Hash Algorithm 3 (SHA-3), the SM3 and SM4 ciphers, and bit permute instructions.
  - Armv9.0-A - Embedded Trace Extension (ETE).
  - Armv9.0-A - Trace Buffer Extension (TRBE).
  - Armv8.x - Memory System Resource Partitioning and Monitoring (MPAM).
  - Armv8.x - Activity Monitors Extension.
  - Armv8.x - Random Number instructions.
  - Armv8.x - CBusy adaptive prefetching mechanism.
- 1MB private L2 cache per core.
- Direct connect to CMN-700 through a CAL.
- Single 256-bit AMBA CHI master interface to CMN-700.
- A utility bus slave interface that is memory-mapped at the Cluster Utility memory region in AP memory map. This 64-bit AMBA AXI5 interface provides a programming interface to access the control registers for various other system components in the cluster and core registers. The SCP and AP cores can access the following control registers:
  - PPUs.
  - Activity monitors in the cores.
  - Maximum Power Mitigation Mechanism (MPMM) registers in the cores.
  - RAS registers.
- A separate voltage domain per core.
- A separate power domain per core and per cluster. There is separate PPU support for core and cluster power management.
- ELA-600 Embedded Logic Analyzer.
- Asynchronous or synchronous options between the DSU and the CMN-700 interconnect.
- Integrated GCI in the cluster power domain. In direct connect mode, one GCI module is supported for each MP1 core to handle all the software-generated interrupts between the cores.

**Figure 3-5: Processor block with a single MP1 big core in direct connect mode**



The Processor block comprises the following interfacing logic outside of the core cluster:

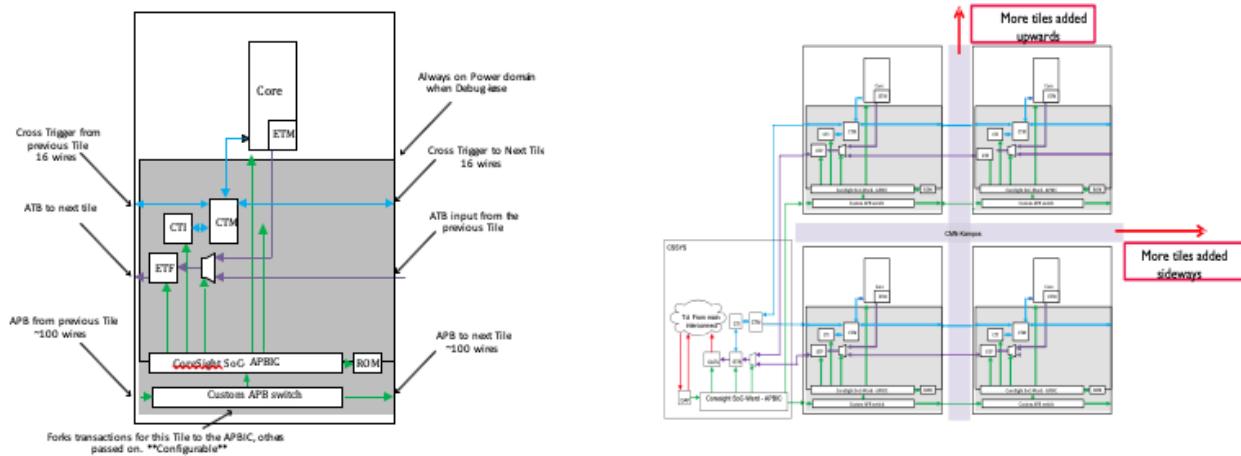
- Integrated GIC Redistributor. A GCI block for each core to support Private Peripheral Interrupts (PPIs) and Software Generated Interrupts (SGIs) between the application cores.
- Clock Control logic to generate all the clocks that are needed for the core and glue logic.
- Core Manager logic, which provides software registers for setting the reset values to the core.

- Daisy chain logic for the following interfaces within each Processor block:
  - AMBA Trace Bus (ATB) trace from each core.
  - Debug APB interface.
  - CoreSight Trigger Interface.
  - Generic counter bus that is routed to all the cores in the system. This routing must be balanced across the system. Configurable register slice logic can be used to add register slices for each core to provide the necessary balancing.
  - Utility bus interface. This interface can be connected through the CMN-700 PUB control communication network to enable the SCP to access Clock Control registers, power management logic, and sensor logic.

These interfaces are daisy chained, with the interfaces originating from the neighboring Processor block and sent to the destination in a daisy-chained manner. This arrangement enables tile-based physical implementation, which is recommended for the optimal physical implementation of a large core-count infrastructure system. For example, a tile can be a group of one or two processor cores, one or two HN-F slices (SLC caches), and an XP. Instead of designing the system as one flat physical implementation, routing channels can be used to create the complete system with a repeatable macro.

The following figure shows an example of debug interfaces daisy chained through each Processor block. The other interfaces are also daisy chained in a similar manner.

**Figure 3-6: Processor block debug daisy chaining with a single MP1 big core in Direct connect mode**



## 3.5 Interrupt block

The RD-N2 Interrupt block contains distributed Generic Interrupt Controller (GIC) components.

RD-N2 supports the following GIC-700 features:

- Arm GICv4.1 architecture.
- Distributed architecture - GIC Distributor, Redistributor, and ITS blocks can be distributed across the system.
  - The number of ITS blocks on the data path is determined based on system functional and performance requirements. Inline ITS, ITS on the side, and system ITS options are supported.
  - Distributed Redistributor (GCI-GIC cluster interface). One block for each core in Direct connect mode to support PPIs and SGIs. The PPI assignments comply with *Arm® Server Base System Architecture, version 6.0*.
  - Single global SPI block.
- Security Extensions.
- Distributed architecture, optimized for up to 128 cores or threads for each die or chip. Support for up to four chips.
- Direct injection of virtualized interrupts, bypassing the hypervisor to deliver Locality-specific Peripheral Interrupts (LPIs) and SGIs to VMs.
- Shared peripheral interrupt supported for up to 1984 interrupts. All interrupts are available for single-chip systems and, equally, between chips for multichip systems.
- Support for up to 16,000 virtual Processing Elements (vPEs).
- P-Channel and Q-Channel support for low-power control.
- Multichip support for SMP use cases.
- MPAM support for memory traffic originating from the GIC.
- Support for the AMBA® AXI and ACE Protocol Specification, issue H bus and all the associated commands.
- PUB interface connectivity can be used for local GIC interrupt communications. This interface is also used for GIC chip-to-chip interrupt communications.
- 128 SPIs can be exported as inputs to the compute subsystem for connection to other SoC components. These interrupts are assumed to be level interrupts. Any edge interrupts must be converted to level interrupts before connecting to these interrupt inputs on the compute subsystem.



Backwards compatibility with the GIC architecture, version 2 programmers model is not supported.

### 3.5.1 GIC Interrupt Translation Service options

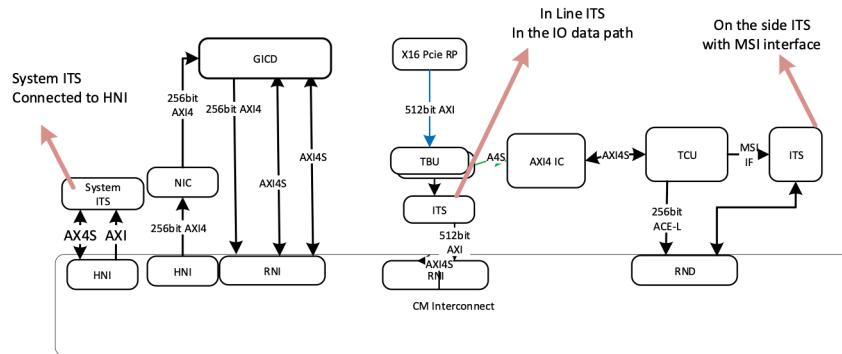
RD-N2 supports the following ITS block arrangements:

- Inline ITS: Integrated before RN-I/RN-D ports inline in the I/O path of the master to system memory.

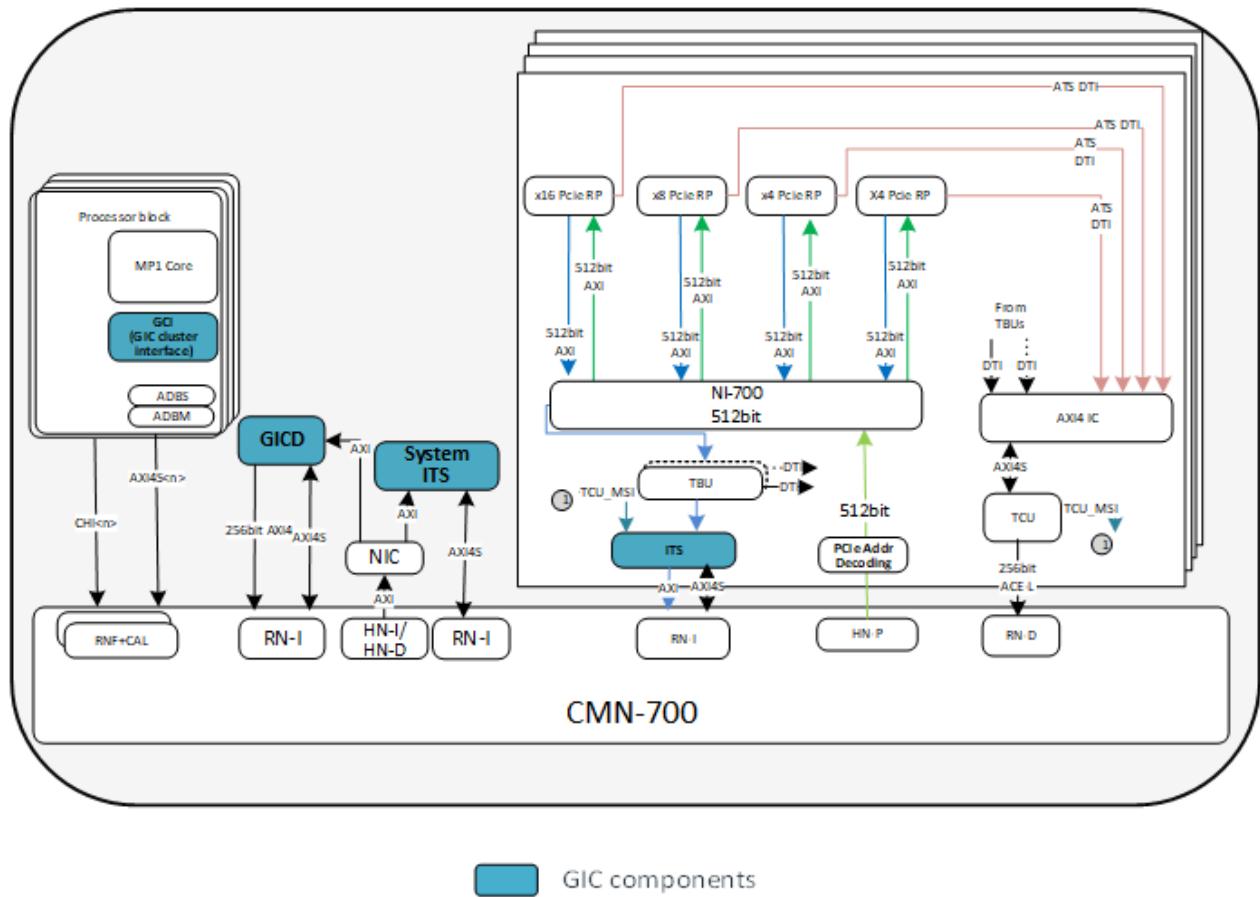
- ITS on the side: Interfaced along the side of the master through the Message Signaled Interrupt (MSI) interface.
- System ITS: Global ITS that is integrated on the CMN-700 HN-I interface. Multiple masters on the local chip or masters on remote chips can send MSI or MSIx targeting to this system ITS. For more information about interrupt routing from a remote accelerator chip to the system ITS on the host, see [Host-to-accelerator use case](#).

The following figure shows inline ITS, ITS on the side, and system ITS block arrangements.

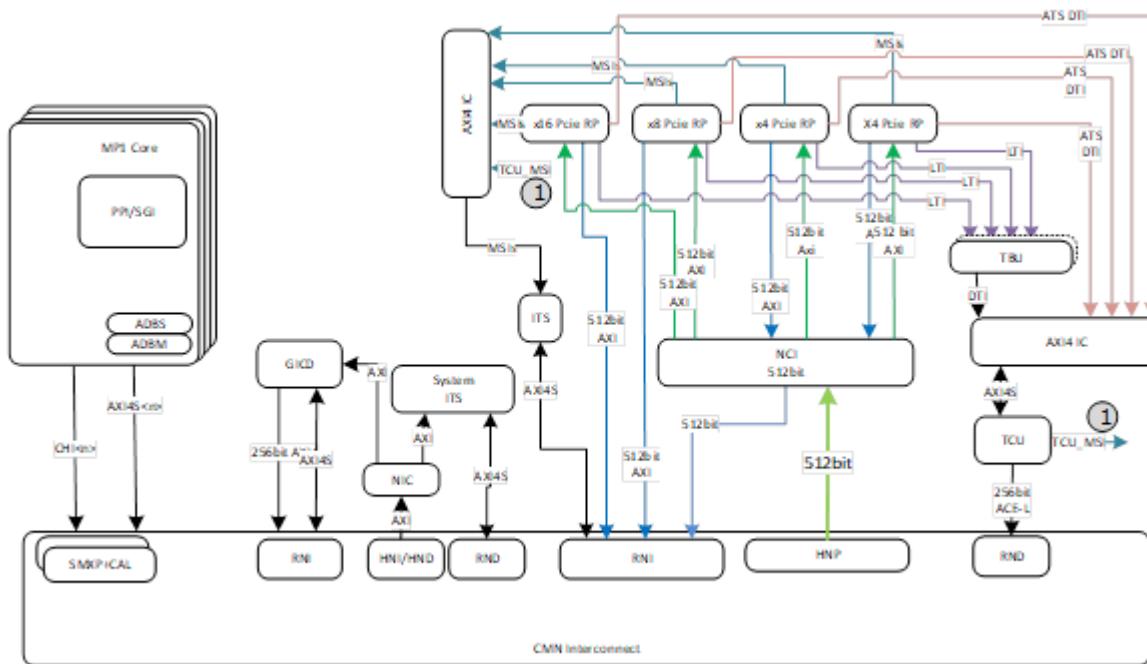
**Figure 3-7: Different ITS types**



RD-N2 supports both inline ITS mode or with MSI interface mode. The mode to implement depends on the system requirements. Both ITS modes can exist in the system for different ITS instances. The following figure shows an example GIC topology with inline ITS. The ITS is connected inline on the AMBA AXI data path.

**Figure 3-8: Example GIC topology with distributed in-line ITS connectivity**

The following figure shows an example GIC topology with distributed ITS using the MSI interface without intercepting the main AMBA AXI data path. In this example, a PCIe Root Complex (PCIe-RC) supports the MSI interface to connect the interrupts directly to the ITS block.

**Figure 3-9: Example GIC topology with MSI interface-based ITS connectivity and system ITS**

## 3.6 I/O Virtualization block

The I/O Virtualization block is implemented by MMU-700 System Memory Management Unit and other IP. This block includes distributed MMU-700 Translation Buffer Unit (TBU) and Translation Control Unit (TCU) components, which provide I/O virtualization for external I/O masters.

The RD-N2 system architecture supports the following MMU-700 features:

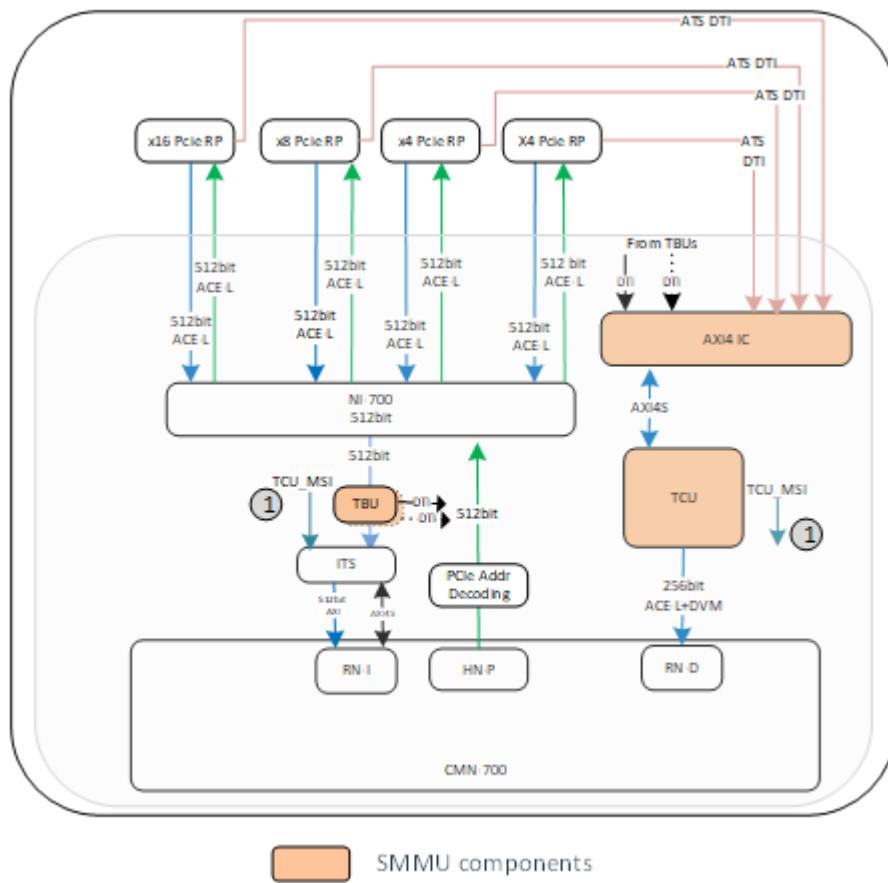
- Compliant with the SMMUv3.2 architecture
- Supports Armv9.0-A and Armv8.x architecture:
  - MPAM - Secure EL2 for multiple Secure OSs and Translation Lookaside Buffer (TLB) range invalidation
  - Hardware Translation Table Update (HTTU) - Support for 52-bit Virtual Addresses (VAs) and Physical Addresses (PAs)
- Includes Distributed Translation Buffer Unit (TBU) and Translation Control Unit (TCU) architecture
- Provides address translation (ATS) functions for I/O master devices such as PCIe
- Supports Direct Translation Interface - Address Translation Services (DTI-ATS) for PCIe
- Provides configurable support for up to eight TBUs per TCU

- Provides single or two-stage address translation for PCIe traffic:
  - Single stage: From VA to PA
  - Two-stage:
    - Stage 1 - from VA to Intermediate Physical Address (IPA)
    - Stage 2 - from IPA to PA
- Supports PCIe features, including ATS and Process Address Space IDs (PASIDs)
- Scalable to millions of active contexts
- Supports ELA-600 for each TCU with all supported trace port features - Support for both internal SRAM buffer mode and trace output mode
- Supports multiple I/O masters, such as x16 PCIe data paths, and shared or independent SMMU systems for each path
- Includes a direct write interface from TCU to ITS for MSI writes

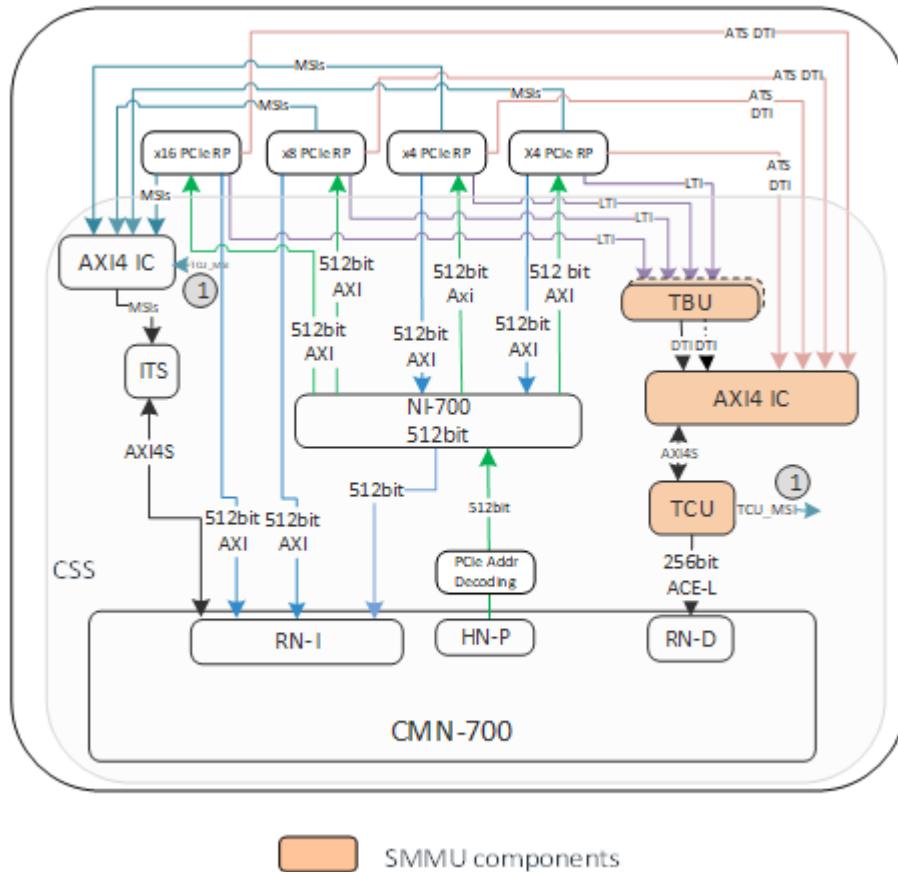
RD-N2 supports both inline and LTI TBU modes, and the choice depends on the system requirements. Both inline and LTI modes can coexist in a system for different TBU instances.

- TaaS LTI mode provides the following benefits over inline TBU mode:
  - Removes write data buffering and improves system PPA.
  - Removes ordering dependency for address translation for I/O masters such as PCIe.
  - Enables PCIe ACS for peer-to-peer traffic through an external PCIe switch.

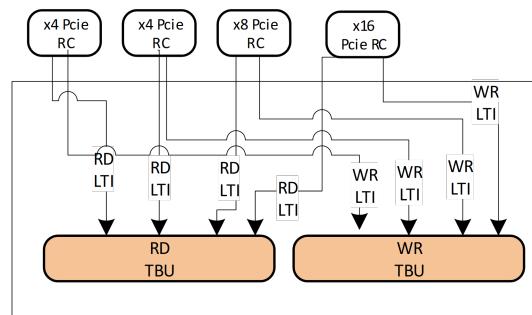
The following figure shows an example SMMU system topology with in-line TBUs in RD-N2 for each I/O master (PCIe) path.

**Figure 3-10: An example SMMU topology with inline TBU mode**

The following figure shows an example SMMU topology with multilane PCIe I/O master connectivity in TaaS LTI configuration mode.

**Figure 3-11: An example SMMU topology with TaaS LTI mode**

In the LTI mode, the I/O master PCIe-RC in the preceding example sends the address translation requests to the TBU through the LTI interface to get the PA. When the PA is returned from the TBU, then the transaction downward path is sent to the interconnect.

**Figure 3-12: Address translation request paths in LTI mode**

In a system, the in-line TBU or LTI mode selection is based on the system features and PPA requirements. For example:

- LTI enables feature such as PCIe ACS for peer-to-peer traffic to be routed through the external PCIe switch.
- LTI offers performance and area optimizations. The performance improvement is achieved by performing address translation ahead in the pipeline before the PCIe ordering, which can improve some latencies in the path. Also, with LTI, there is no write buffering requirement in the TBU.

## 3.7 Debug block (debug and trace)

The debug logic is distributed to provide real-time trace facilities for the application processor cores, MCP, SCP, CMN-700 interconnect, and other compute subsystem components.

The following key debug and trace features are supported:

- Compliant with Arm® CoreSight™ Base System Architecture, version 1.0, Combination C.
- Real-time trace facilities for application processor cores, the SCP, the MCP, and the Interconnect block.
- Full support for CoreSight debug and trace for all application processor cores, the SCP, the MCP, and the CMN-700 interconnect.
- Support for the following CoreSight components:
  - Embedded Trace FIFO (ETF).
  - Embedded Trace Router (ETR).
  - System Trace Macrocell (STM).
  - Cross Trigger Interface (CTI).
  - CoreSight Address Translation Unit (CATU).
- The MSCP debug block is independent of the application processor debug block to enable debug of the clock, reset, and power control devices infrastructure.
- ELA-600 for in-silicon hardware debug.
- Power gating of the CoreSight block.
- Separate cluster debug chain power domain.
- DAP (JTAG) access in the Always-on domain to enable basic debug access when powering up.
- Arm Development Studio, along with transactor from EDA vendors.
- Trace over functional I/O, such as PCIe.
- Multichip debug. For more information, see [Multichip debug](#).

### 3.7.1 External debugger connectivity

The RD-N2 subsystem is connected to the external world through a Debug Access Port (DAP) interface.

A single DAP is used to access the application processor, SCP, and MCP cores. This interface supports the Joint Test Action Group (JTAG) and Serial Wire Debug (SWD) protocols. The DAP is in the Always-on (AON) power domain, which enables basic debug access when powering up.

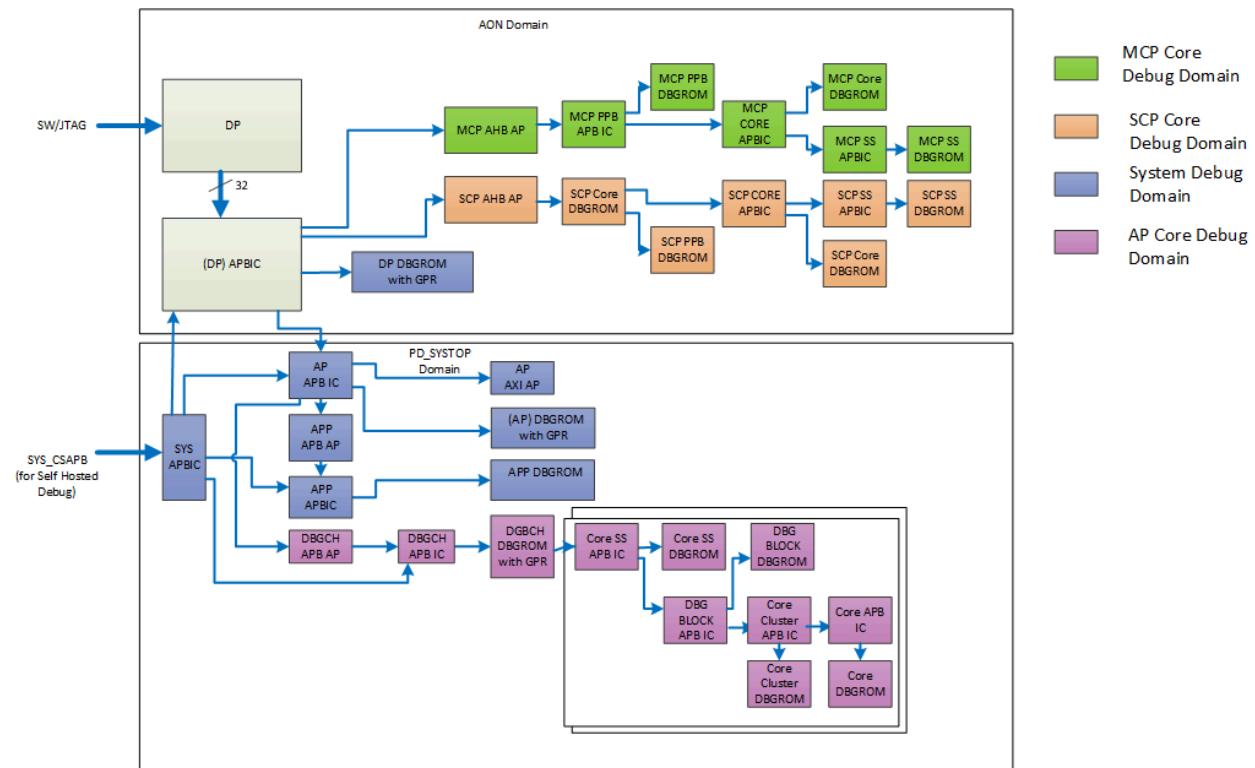
The following external connections can be made to the DAP interface:

- An external debugger, such as Arm DS-5 Debugger
- Interface hardware, such as the Arm DSTREAM debug and trace unit

RD-N2 is debugged using a common debug JTAG interface connected to a DAP. A Serial Wire JTAG Debug Port (SWJ-DP), which combines a JTAG-DP and a SW-DP, enables debug access through either JTAG or SWD. The application processor, SCP, and MCP cores can be debugged independently in the system.

The following functionality is supported through the DAP interface:

- Access to all the debug components in the system
- Cross-trigger support across cores in the system
- The ability to trace out from all the cores and system components to DDR memory through the trace memory controller
- The ability to trace out from all the cores and system components to off-chip memory through external functional I/Os such as PCIe
- Debugger access to the full system memory through a CoreSight debug and trace AMBA AXI-AP interface connected to the CMN-700 interconnect

**Figure 3-13: Architectural view of RD-N2 DAP structure**

Access to SCP and MCP core debug components for self-hosted debug from the application processor cores is optional.

The compute subsystem supports self-hosted debug. Debug monitor software running on the application processor or SCP cores can access all the debug components without an external debugger.

### 3.7.2 Debug authentication

There are different debug domains in the compute subsystem as shown in [Figure 3-13: Architectural view of RD-N2 DAP structure](#) on page 40:

- System debug domain. This domain includes system-level components, such as the CMN-700 interconnect, AXI-AP to access the memory map, etc.
- Application processor core debug domain.
- SCP core debug domain.
- MCP core debug domain.

A different set of authentication signals provides debug authentication for each domain:

- System debug domain components - **SYSDBGGEN**, **SYSNIDEN**, **SYSSPIDEN**, **SYSSPNIDEN**
- Application processor core debug components that are part of all the debug domain chains - **APPDBGGEN**, **APPSPIDEN**
- SCP core debug domain components - **SCPDBGGEN**, **SCPNIDEN**, **SCPSPIDEN**, **SCPSPNIDEN**
- MCP core debug domain components - **MCPDBGGEN**, **MCPNIDEN**, **MCPSPIDEN**, **MCPSPNIDEN**



The application processor cores implement Armv8.5 debug architecture, in which the **NIDEN** and **SPNIDEN** signals are deprecated.

These signals arrive as inputs to the compute subsystem. The instantiation of these signals is **IMPLEMENTATION DEFINED**. SoC implementations must ensure that these signals are controlled by a Secure entity/root of trust to guard against tampering. Alternatively, the signals can be set using fuses implemented as part of the SoC.

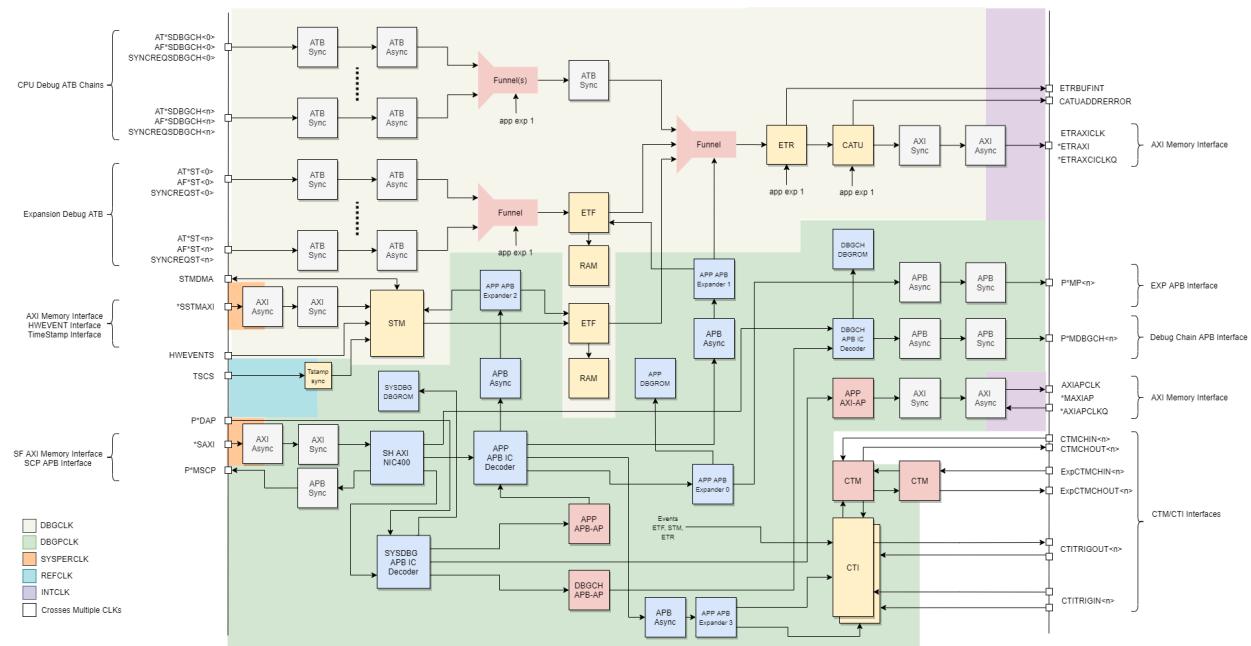
### 3.7.3 System debug support

The system debug logic provides the following debug support:

- Logic to support daisy-chaining of the core debug logic interfaces (Debug APB, Traces, Triggers and Timestamp) to support tile-based floorplan
- An STM that can generate trace data directly from AMBA AXI write transactions (instrumented trace data) and from hardware events (HWEVENT interface) - The STM has its own dedicated STM ETF with 16KB of memory.
- A system trace funnel that takes traces from application processor cores in different daisy chains, the SCP, the MCP, the CMN, and the STM - The traces are funneled to 32KB of system ETF memory.
- Trace data routing to the main system memory through the Embedded Trace Router (ETR)
- A CoreSight Address Translation Unit (CATU), which provides address translation on the AMBA AXI interface from the ETR
- A debug APB network to program the debug peripherals
- A dedicated DAP APB bus interface from the external debugger
- A debug APB interface that provides access to all cores in the various debug chains and debug components in the SCP and MCP blocks
- Trace and trigger components across the system
- Access from all the application processor, SCP, and MCP cores for self-hosted software debug
- An AMBA AXI-AP that allows an external debugger to access the system memory backplane directly without having to go through a processor
- Debug authentication for system debug components through the **SYSDBGGEN**, **SYSNIDEN**, **SYSSPIDEN**, and **SYSSPNIDEN** input signals

- Cross-triggering through a CTI and Cross Trigger Matrix (CTM) across all the components in the system
- Expansion Debug interfaces (Debug APB, Expansion Trace and Triggers) to the SoC for components added outside the compute subsystem
- Separate reset domain for the logic in each debug chain - Every debug chain has its own dedicated PPU for reset control.
- Support for Granular Power Requester (GPR) functionality through GPR registers for SYSTOP power domain reset requests
- Support for multichip debug

**Figure 3-14: Block diagram for Debug block**



### 3.7.4 Application core debug support

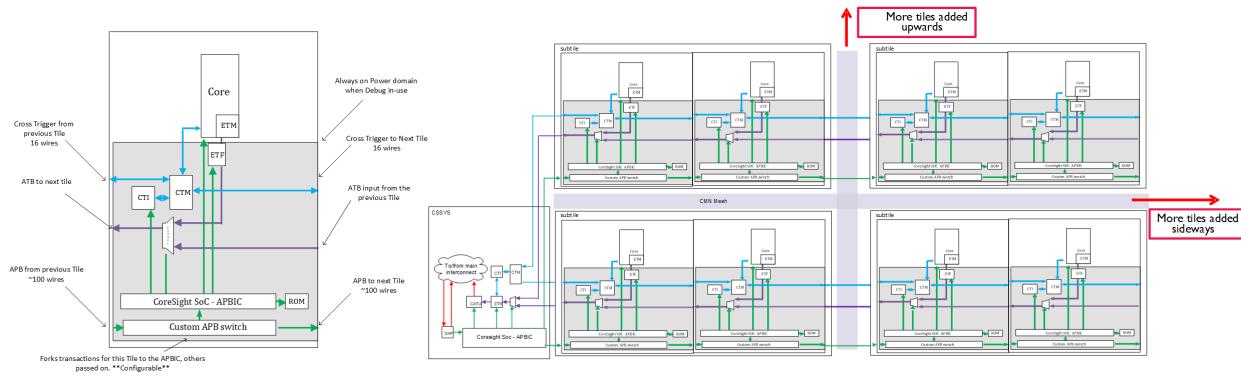
The debug logic for the application processor core is present in the Direct connect DSU. The debug chain logic is in part of the Processor block, outside the core and the Direct connect DSU.

This arrangement provides the following debug support:

- A local debug ROM table that contains a list of components added in the Processor block. This table enables the debugger to determine the components that are implemented locally in the Processor block.
- A debug control with APB register interface that provides access to debug registers in the Processor block. The debug APB is routed to the neighboring core by daisy chaining.
- Embedded 8KB trace FIFO for buffering core traces.

- A trace funnel that arbitrates up to two trace sources for the local core and another for the neighboring core to form a daisy chain.
- A CTI for each processor core that is daisy chained with the CTI of the neighboring core and routed to a system-level CTM.
- Debug authentication for system debug components through the **APPDBGEN**, and **APPSPIDEN** input signals.
- Daisy chaining of the debug channel, debug-related interfaces, and global timestamp synchronization network to connect all the cores to the debug components.

**Figure 3-15: High-level representation of debug daisy chain structure across Processor blocks**



### 3.7.5 SCP core debug support

RD-N2 supports SCP debug.

The following debug support is provided for the SCP core:

- SCP debug ROM tables that enable the debugger to determine which debug components are internal to the Cortex-M7 core and identify others outside the core.
- Components inside the Cortex-M7 core:
  - A Flash Patch and Breakpoint (FPB) unit for implementing hardware breakpoints.
  - A Data Watch-point and Trace (DWT) unit for implementing watchpoints, data tracing, and system profiling.
  - An Instrumentation Trace Macrocell (ITM), which supports printf()-style debugging. This unit generates trace data that is sent to an ATB funnel and to Serial Wire Output (SWO).
  - An Embedded Trace Macrocell (ETM) for instruction tracing.

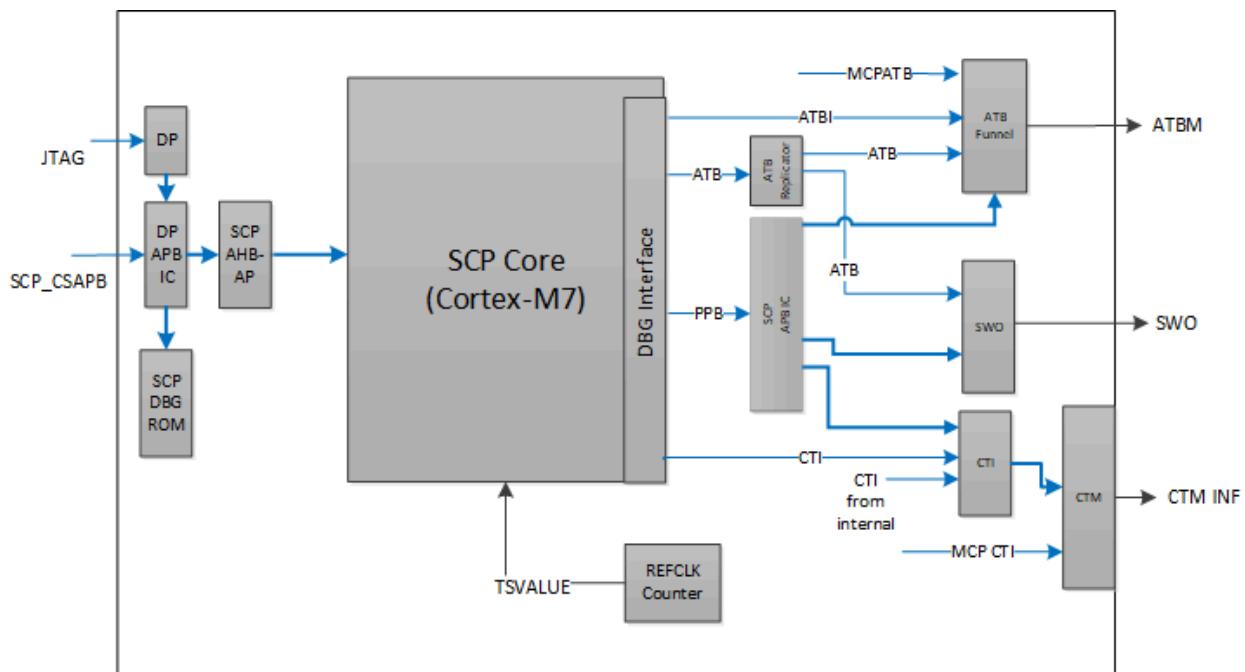
- Components added outside the Cortex-M7 core:
  - A trace funnel to combine the trace sources from the ITM, ETM and MCP down to one trace output (ATBM) before it is sent to the primary CoreSight block.
  - A replicator to send the ITM trace output to either SWO or to the system ETR through the system funnel.
  - A CTI to support cross triggering with other debug components in the system. This CTI also controls the **REFCLK** generic counter, enabling debug triggers to halt the counter.
  - A CTM to connect the CTI triggers from the SCP and MCP to the system funnel, enabling triggering of other system components.
- Timestamp value for the Cortex-M7 core provided by the system generic counter.
- Debug authentication for the SCP core debug domain through the **SCPDBGEN**, **SCPNIDEN**, **SCPSPIDEN**, and **SCPSPNIDEN** input signals.

If you want to trace out the SCP core over the Trace Port Interface Unit (TPIU), you must enable authentication of the system components.



The TPIU that is shown in the SCP code debug domain is used for SWO functionality. The Cortex-M7 TPIU is used for this purpose. Only TRACESWO output is used. This TPIU instance has two ATB input interfaces to support trace sources from both the SCP and MCP cores.

**Figure 3-16: High-level view of SCP core debug domain**



### 3.7.6 MCP core debug support

RD-N2 supports MCP debug.

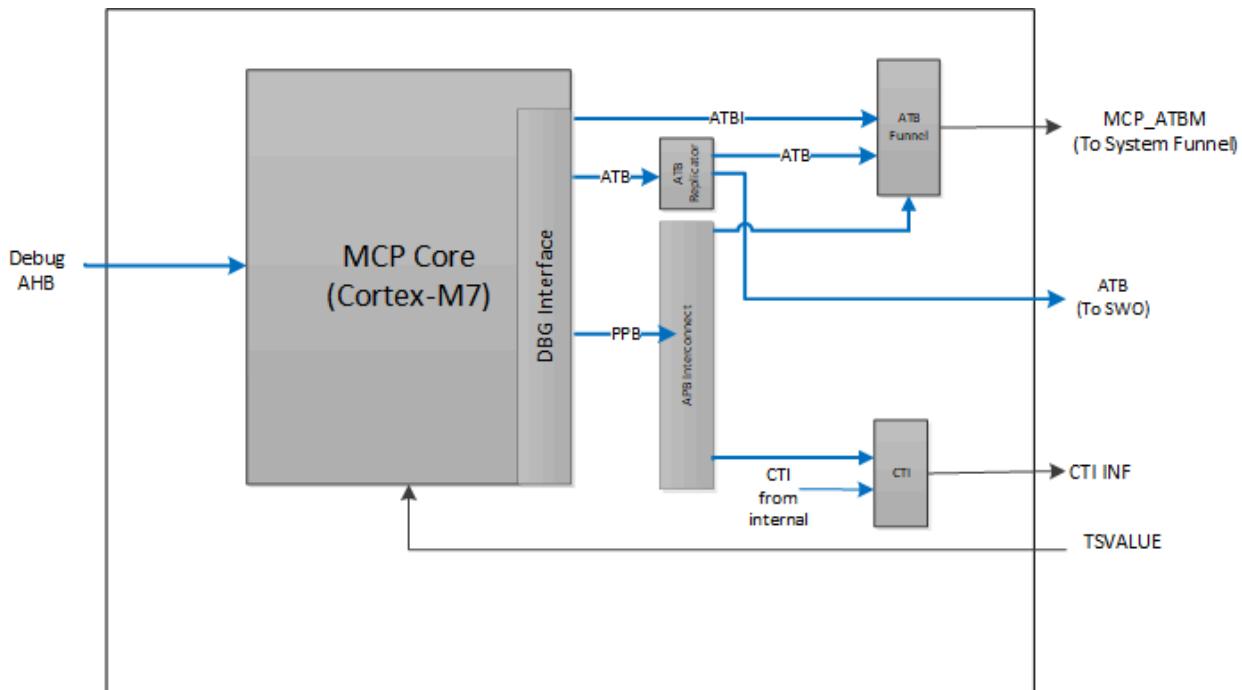
The following debug support is provided for the MCP core:

- MCP debug ROM tables that enable the debugger to determine which debug components are internal to the Cortex-M7 core and identify others outside the core.
- Components inside the Cortex-M7 core:
  - An Flash Patch and Breakpoint (FPB) unit for implementing hardware breakpoints.
  - A Data Watch-point and Trace (DWT) unit for implementing watchpoints, data tracing, and system profiling.
  - An Instrumentation Trace Macrocell (ITM), which supports printf()-style debugging. This unit generates trace data that is sent to an ATB funnel and to SWO.
  - An Embedded Trace Macrocell (ETM) for instruction tracing.
- Components added outside the Cortex-M7 core:
  - A trace funnel to combine the trace sources from the ITM and ETM down to one trace output. This output is muxed with the SCP combined trace output before sending it to the system funnel.
  - A replicator to send the ITM trace output over SWO. The MCP and SCP ITM trace is muxed onto the same SWD output port.
  - A CTI to support cross triggering with other debug components in the system.
  - A CTM to connect the triggers from the core and other components in the MCP block to the system funnel, enabling triggering of other system components.
- Timestamp value for the Cortex-M7 core provided by the system generic counter.
- Debug authentication for the MCP core debug domain through the **MCPDBGEN**, **MCPNIDEN**, **MCPSPIDEN**, and **MCPSPNIDEN** input signals.



If you want to trace out the MCP core over the Trace Port Interface Unit (TPIU), you must enable authentication of the system components.

If you want to trace out the MCP core over SWO, you must enable the SCP core domain authentication signals.

**Figure 3-17: High-level view of MCP core debug domain**

### 3.7.7 ROM tables

The ROM tables hold the locations of debug components. Debuggers can use the tables to determine which components are implemented.

RD-N2 has multiple ROM tables so that external debuggers can discover all the debug components in the system. The following table lists the types of debug components that can be accessed for each ROM table.

**Table 3-1: Components in RD-N2 ROM tables**

Table name	Table location	Debug components
DP Debug ROM	System debug domain	SCP Cortex-M7 AHB AP. MCP Cortex-M7 AHB AP. SYS DBGROM table. GPR registers for power and reset of PD_SYSTOP and PD_DBGTOP power domains.
SYS DBG ROM	System debug domain	APP APB AP. DBGCH APB AP. SYSEMM AXI AP.

Table name	Table location	Debug components
APP DBG ROM	System debug domain	STM ETF. System ETF. Configurable expansion for system funnels. Master funnel. DBGCH funnel. System ETR. STM. System CTI. System CATU.
DBGCH ROM	AP debug domain	CORE<n> SS DBG ROM table.
CORE<n> SS DBG ROM	AP debug domain	ETF. Trace funnel. CTI. Core Debug Block ROM table.
Core Debug Block ROM	AP debug domain	Cluster CTI. PE CTI.
Cluster ROM	AP debug domain	Core0 ROM table. For information about components in the core DBG ROM, see the Arm® Neoverse™ N2 Core Technical Reference Manual.
SCP Core DBG ROM	SCP debug domain	ETM. CTI. SCP SS DBG ROM table.
SCP PPB DBG ROM	SCP debug domain	FPB. DWT. ITM. SCS.
SCP SS DBG ROM	SCP debug domain	SWO. Funnel. CTI. ATB replicator.

Table name	Table location	Debug components
MCP Core DBG ROM	MCP debug domain	ETM. CTI. MCP SS DBG ROM table.
MCP PPB DBG ROM	MCP debug domain	ITM. DWT. FPB. SCS.
MCP SS DBG ROM	MCP debug domain	SWO. Funnel. CTI. ATB replicator.

### 3.7.8 Trace sources

RD-N2 contains various trace sources.

The following trace sources are present:

- Trace sources from each application processor core can be routed to different individual buffers in the DDR memory. The trace sources can also be routed to the system ETR for transfer over functional I/O interfaces, such as PCIe or USB.
- Trace sources from the SCP and MCP ITMs and ETMs are routed to the system ETR for writing to DDR memory or transfer over functional interfaces.
- Traces from the STM are routed to the system ETR for writing to DDR memory or transfer over functional interfaces.
- Trace sources from the CMN-700 interconnect are routed to the system ETR for writing to DDR memory or transfer over functional interfaces.
- Depending on the implementation of the SMMU ELA, trace sources from the SMMU TCU ELA components are routed to the system ETR. The trace sources can be written to DDR memory or transferred over functional interfaces.
- Other trace sources, such as ELA components that are added as part of the SoC.

### 3.7.9 Cross triggers

Cross triggers provide a mechanism for cores and components to trigger each other in a controlled manner.

Triggers to and from a component or core connect to a CTI module. The CTI then maps them onto channels that are connected to one or more CTMs to broadcast the trigger to all the CTIs in the system. The following table lists the components that contain trigger sources and sinks, and details how the triggers are connected.

**Table 3-2: Trigger sources, sinks, and connectivity**

Module name	Source	Sinks	CTI and CTM connectivity
CPU0-CPU <sub>n</sub>	Y	Y	<p>Triggers from each core are connected, in order, to:</p> <ul style="list-style-type: none"> <li>• A CTI inside the application processor core.</li> <li>• A CTM per core to support daisy chaining with the neighboring core.</li> <li>• The system CTM.</li> </ul>
SCP	Y	Y	<p>Triggers from the SCP core are connected to:</p> <ul style="list-style-type: none"> <li>• A CTI inside the SCP core.</li> <li>• A CTM to connect triggers from the SCP core, other components in the SCP block, and trigger interface from the MCP block</li> <li>• The system CTM.</li> </ul>
MCP	Y	Y	<p>Triggers from the MCP core are connected to:</p> <ul style="list-style-type: none"> <li>• A CTI inside the MCP core.</li> <li>• A CTM to connect triggers from the MCP core and other components in the MCP block.</li> <li>• The CTM in the SCP block.</li> <li>• The system CTM.</li> </ul>
STM	Y	Y	<p>The STM provides the trigger outputs TRIGOUTSPE, TRIGOUTSW, TRIGOUTHETE, and ASYNCOUT. Two trigger inputs are required to drive four inputs on the hardware event interface. Two triggers drive HWEVENTS[0] and HWEVENTS[2] directly, and the same two triggers, after inversion, drive HWEVENTS[1] and HWEVENTS[3].</p> <p>These triggers are connected to the system CTM through a CTI.</p>
ETF	Y	Y	<p>Each ETF generates two triggers, FULL and ACQCOMP, and takes trigger inputs to drive FLUSHIN and TRIGIN.</p> <p>These triggers are connected to the system CTM through a CTI.</p>
ETR	Y	Y	<p>The ETR generates two triggers, FULL and ACQCOMP, and takes trigger inputs for FLUSHIN and TRIGIN.</p> <p>These triggers are connected to the system CTM through a CTI.</p>
REFCLK generic counter (REFCLK CNTControl)	N	Y	<p>This generic counter takes two triggers to halt the counter.</p> <p>These triggers are connected to the system CTM through a CTI.</p>

Module name	Source	Sinks	CTI and CTM connectivity
CMN-700 Interconnect	Y	N	The CMN-700 interconnect generates a trigger for the Debug Watch triggers. These triggers are connected to the system CTM through a CTI.
CMN-700 interconnect	N	Y	Trigger request for the CMN-700 interconnect PMU counter snapshot. Connected to CMN PMUSAPSHOTREQ. These triggers are connected to the system CTM through a CTI.
NI-700 interconnect	N	Y	Multiple trigger requests for the NI-700 interconnect PMU counter snapshot. Connected to various NI-700 network PMUSAPSHOTREQ signals. These triggers are connected to the system CTM through a CTI.
MMU-700 TCUn	N	Y	Trigger request for the TCU PMU counter snapshot. Connected to the respective TCU PMUSAPSHOTREQ. There is one request for each TCU in the system. These triggers are connected to the system CTM through a CTI.
MMU-700 TBUn (read/write)	N	Y	Trigger request for the TBU PMU counter snapshot. Connected to the respective TBU PMUSAPSHOTREQ. There is one request for each TBU in the system. These triggers are connected to the system CTM through a CTI.
GIC-700	N	Y	Trigger request for the GIC PMU counter snapshot. Connected to the respective GIC sample_req. These triggers are connected to the system CTM through a CTI.
ELA for each Neoverse N2 core	Y	Y	Triggers from the core ELA are combined with core triggers.
Expansion triggers	Y	Y	Triggers from other components in the SoC, or from other chips in the multichip use case, should be connected to the system CTM. These triggers are exported as expansion trigger interfaces on the compute subsystem.
ELA in each TCU	Y	Y	Triggers from each TCU ELA should be connected to the system CTM.

For more information about CTIs and the CTM, see the CoreSight™ Components Technical Reference Manual.

### 3.7.10 System Trace Macrocell

STM-500 provides high-bandwidth tracing of instrumentation that is embedded into software.

This instrumentation is made up of memory-mapped writes to the STM through its AMBA AXI slave interface. In addition, the STM provides a hardware event interface that generates trace data on the rising edge of the signals on the hardware event interface.

### 3.7.10.1 High-bandwidth trace instrumentation

The STM supports multiple MIPI System Trace Protocol version 2 (STPv2) masters.

For AMBA AXI transactions, the master identification is based on:

- Whether the access is Secure or Non-secure
- The upper bits of the AMBA AXI address

Software stimulus has 128 master IDs allocated, from 0x00 to 0x7F. The lower half of the master ID space is for Secure accesses and the upper half for Non-secure accesses.

The STPv2 master ID[7:0] is constructed from the AWADDR and AWPROT bits of the AMBA AXI stimulus port:

- Master ID[7] = 0b0
- Master ID[6] = AWPROT[1]
- Master ID[5:0] = AWADDR[29:24]

In this ID, master IDs 0–63 are allocated to Secure transactions and master IDs 64–127 to Non-secure transactions.

RD-N2 provides a single 16MB address space for the STM-500 AMBA AXI interface in the application processor memory map. However, the STM provides a separate view of the address space to different masters, grouped by the STPv2 Master IDs shown in the following table.

All the AMBA AXI transactions from different masters target the same 16MB address space. The AMBA AXI transactions use AWUSER bits to specify the master to which the transaction belongs. So, the AWUSER bits are mapped to upper address bits AWADDR[29:24].

When writing traces to the STM using the AMBA AXI slave expansion interface, each external master is identified using the AWUSER[5:0]. For all the masters, the AWUSER[7:6] bits are reserved for SOC integration logic. If any accesses do not belong to any logical system master in the table, default masters are selected.

**Table 3-3: Master AWUSER bits mapped to STM views and associated STPv2 master IDs**

Source	AWUSER[5:0]	STPv2 Master ID[7:0] Secure accesses AWPROT[1] = 0	STPv2 Master ID[7:0] Non-secure accesses AWPROT[1] = 1
Hardware tie-off for all AP cores	0	0	64
Reserved	1–15	1–15	65–79
SoC external I/O masters (to be assigned by SoC integrator)	16–58	16–58	80–122
Default master	59	59	123
MCP	60	60	124
SCP	61	61	125
ETR	62	62	126
DAP (AXI-AP)	63	63	127

### 3.7.10.2 STM hardware event support

STM-500 supports the HWEVENTS[63:32] interface. Internal hardware events in the compute subsystem can be connected to this interface so that event traces can be generated.

The hardware event interface is assigned as follows:

Event Signal	Source	Edge/Level
HWEVENTS[0]	Trigger Output CTI2TRIGOUT[0] from CTI2, rising edge detection	Edge
HWEVENTS[1]	Trigger Output CTI2TRIGOUT[0] from CTI2, falling edge detection	Edge
HWEVENTS[2]	Trigger Output CTI2TRIGOUT[1] from CTI2, rising edge detection	Edge
HWEVENTS[3]	Trigger Output CTI2TRIGOUT[1] from CTI2, falling edge detection	Edge
HWEVENTS[4]	SCP Sleep Entry	Level
HWEVENTS[5]	SCP Sleep Exit	Level
HWEVENTS[6]	MHU APP to MCP Non-Secure interrupt	Level
HWEVENTS[7]	MHU MCP to APP Non-Secure interrupt	Level
HWEVENTS[8]	MHU SCP to MCP Non-Secure interrupt	Level
HWEVENTS[9]	MHU MCP to SCP Non-Secure interrupt	Level
HWEVENTS[10]	MHU APPs to SCP Non-Secure interrupt	Level
HWEVENTS[11]	MHU APPs to SCP Secure interrupt	Level
HWEVENTS[12]	MHU SCP to APPs Non-Secure interrupt	Level
HWEVENTS[13]	MHU SCP to APPs Secure Interrupt	Level
HWEVENTS[31:14]	reserved	n/a
HWEVENTS[63:32]	STM Hardware Events Expansion Interface (EXSSTME) (routed as primary input to subsystem)	Edge/Level

Signals connected to the Hardware Events Expansion Interface must operate in the DBGCLK clock domain. DBGCLK is made available at the subsystem top-level.

For more information about STM-500, see the *CoreSight™ System Trace Macrocell Technical Reference Manual*.

### 3.7.11 Self-hosted debug

The compute subsystem supports self-hosted debug. With self-hosted debug, the debug target runs debug monitor software on any cores in the system. This arrangement avoids a requirement for expensive interface hardware to connect a second host computer.

The debug target can run on any of the application processor cores, the SCP core, or the MCP core. The subsystem allows for self-hosted debug of any of the debug domains. The appropriate debug domain authentication signals must be enabled to allow debugging of the respective debug domains. All the cores have access to all the debug components in compute subsystem provided the respective debug domain authentication signals are enabled.

Debug targets running on any application processor core can debug the application processor debug domain, the SCP core debug domain, or the MCP debug domain. Debug targets running on the SCP or MCP cores can debug their own debug domain or the application processor core debug domain.

Self-hosted debug accesses are marked secure when the subsystem comes out of reset. The security state is configurable by software, where a secure master can allow non-secure masters to access the debug memory map. See [Interconnect block NIC-450 GPV memory map](#) and [Self-hosted memory map](#) for details.

### 3.7.12 Debug through functional I/O interfaces

The debug trace can be output to external storage over functional I/Os such as PCIe or USB.

For PCIe, the ETR can write directly to PCIe device memory or the PCIe device can read the trace from the DDR buffer. It is assumed that the debug components are programmed through JTAG by an external debugger or a debug target running on the one of the cores.



**Note** Because external devices are regarded as Non-secure by default, these devices do not have access to any debug components in the system when the debug access interface is made secure during boot time.

## 3.8 MSCP block

The Manageability and System Control Processor (MSCP) block implements the Cortex-M7-based SCP and MCP. This block connects to all system control and power control logic for the relevant functional blocks.

The Cortex-M7 core is a highly efficient high-performance, embedded processor that features:

- Low interrupt latency
- In-order super-scalar pipeline that means many instructions can be dual-issued, including load/load and load/store instruction pairs because of multiple memory interfaces.
- Memory interfaces that the processor supports include:
  - Tightly-Coupled Memory (TCM) interfaces.
  - Optionally enable Harvard instruction and data caches.
  - AXI master (AXIM) interface.
  - Dedicated low-latency AHB-Lite peripheral (AHBP) interface.
  - AHB-Lite slave (AHBS) interface that provides DMA access to TCMs.
- Optional Memory Protection Unit (MPU) that you can configure to protect regions of memory

- Error Correcting Code (ECC) functionality for error detection and correction is included in the data and instruction caches when implemented.
- TCM interfaces support the implementation of external ECC to provide improved reliability.
- Optionally enable floating-point arithmetic functionality with support for single and double-precision arithmetic.

### 3.8.1 System Control Processor block

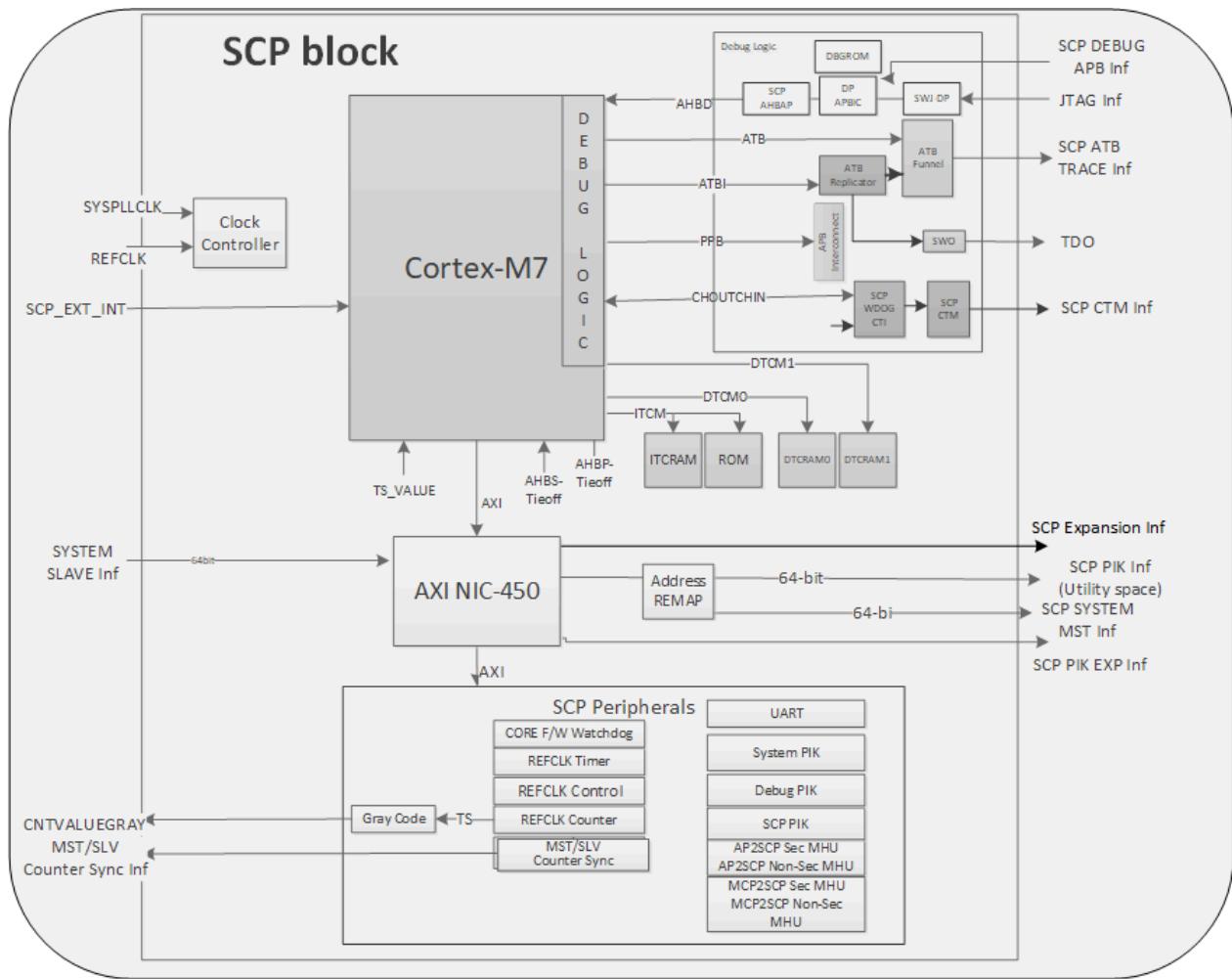
The SCP manages the overall power, clock, reset, and system control of RD-N2.

The System Control Processor contains a Cortex-M7 core that operates only in the secure world.

The SCP has the following functions:

- Provides boot sequencing during system start-up
- Communicates with the MCP to establish trust and the security integrity of the system
- Responsible for all the clock and power controls
- Controls initial configuration and subsequent reset
- Manages transitions between operating points, that is, the external voltage regulator and clock management to support Dynamic Voltage and Frequency Scaling (DVFS)
- Handles hardware wake up requests from components such as timers and interrupts
- Saves and restores all states in the interconnect when powering down or powering up
- Manages a consistent matrix of device states across the whole system
- Controls components that are both internal and external to the SoC - For example, PLL, clock and reset controls, and Power Management Integrated Circuit (PMIC)
- Support for SECDED (Single Error Correction, Double Error Detection) ECC on the RAMs

The following figure shows the architecture of the SCP block. The system is based on a Cortex-M7 core with NIC-450 to extend and support interfaces such as Power Integration Kits (PIKs), watchdogs, timers, and counters.

**Figure 3-18: SCP block architecture**

The SCP supports several peripherals that are connected through the non-coherent interconnect:

- Non-secure Message Handling Unit (MHU) for communication with the MCP and processor cores - For more information, see [Message communication between processors](#).
- Secure MHU for communication with the MCP and processor cores - For more information, see [Message communication between processors](#).
- Tightly coupled SRAM (TCRAMs) for private data and code
- On-chip (Trusted) and Secure boot ROM
- PCK-600 power management logic to control the power and reset for the PD\_SYSTOP, DBGTOP, and DBGCH power domains
- PL011 UART
- BP140 AMBA AXI memory interface
- Timing peripherals
- Watchdog timer

- System generic REFCLK counter. The REFCLK counter is an Arm generic counter that is memory mapped and running on REFCLK
- SCP generic REFCLK timer
- Sensor Control Framework (SCF) data RAM - This peripheral collects data from the various sensors in the chip and the data is written to the RAM. The SCP core can access this data and process.
- Expansion spaces to enable the addition of various AON and sensor components
- Various interrupts from the compute subsystem that are routed to the Cortex-M7 core

## Clock and Reset generation module

The SCP block contains a Clock and Reset Generation (SCP CRG) module that controls the resets to the SCP M7 core and its peripherals, and thus controls the resets for the entire subsystem. It is the first module to come out of reset. For more information about the resets and the reset tree implemented in the compute subsystem, see [Resets](#). The CRG block also has a few clock generators that generate the clocks needed for the SCP block. For more information about the clocks, see [Clocks](#).

## SCP Power Control module

The SCP block also contains an SCP Power Control (SCP PWR CTRL) module that implements these features:

- provides registers to program the clock generation logic within the SCP block.
- provides memory-mapped interface to consolidate interrupts to the SCP M7 core.
- records the watchdog interrupts in non-resettable registers for debug purposes.
- provides the control to SCP M7 core to warm reset just the core. For more information, see [SCP core Warm reset](#).

For more details on the SCP Power Control registers, see [MSCP Power Control registers](#), SCP.

## System Power Integration Kit module

The SCP block also implements a System Power Integration Kit (System PIK) module that implements all the power management control logic related to the SYSTOP domain. This includes a Power Policy Unit (PPU) to control the SYSTOP reset domain, and also an APB register bank used to provide memory-mapped interface to software to control the logic (clocks, resets, etc) in the SYSTOP domain. For more details on the System PIK registers, see [System Power Integration Kit registers](#).

## System ID module

The System ID module implements an memory-mapped APB register bank that provides system-specific ID configuration data. For more details on the System ID registers, see [System ID registers](#).

## System Power Control

Along with SCP Power Control and System PIK modules, the SCP M7 core manages the overall subsystem power, clock and reset logic. Through the CMN-700 PUB network, SCP has memory-mapped access to the Cluster Utility region on all the Processor blocks to control the Core/Cluster

PPUs in the DSU, and other core manager registers. The SCP can also access the DMC PIK interfaces (EXMDMCPPIK\_DMC<x>) over the CMN-700 PUB network using memory-mapped accesses to initialize the DMC and PHY that are present external to the compute subsystem. Also, the compute subsystem provides expansion SCP PIK interfaces (EXMSCPAXI) to provide memory-mapped accesses to expansion PIK modules in the SoC, if needed.

### 3.8.1.1 Address translation

Cortex-M7 processors only access 32-bit address space, so address translation is required for the SCP and MCP to access peripherals on the AP memory map.

Address translations are performed to support SCP and MCP access to:

- The lower 2GB of the local chip application memory.
- Above 2GB of the local chip application memory or application memory region on different chips in the multichip scenario. This access is achieved by windowing into one 1MB window at a time.
- Debug components mapped in the memory region CHIP\_ADDR\_SZ\*CHIPID + (0x04\_0000\_0000-0x04\_3FFF\_FFFF) in the respective chips.

All of the local application memory space or all of the application memory map region on the different chip can be accessed. To achieve this access, the top bits are altered using a value from the SCP or MCP PIK ADDR\_TRANS register. The register that is used depends on the processor that is performing the access.

**Table 3-5: Address translation from the SCP and MCP memory map to the application processor memory map**

Cortex-M7 address space	DBG_ADDR_TRANS[EN]	ADDR_TRANS[EN]	ADDR_TRANS[CMN_ATRANS_EN]	ADDR_TRANS[ADDR_47_20]	Translated address space
0xCB00_0000 - 0xCB0F_FFFF	0	1	X	Program field with bits[47:20] of address on multichip application processor memory map.	Above 2GB on local or remote memory map.  {ADDR_TRAN[47:20] + 0x0_0000} - {ADDR_TRAN[47:20] + 0xF_FFFF}
0xA000_0000 - 0xFFFF_FFFF	1	X	X	Unused	CHIP_ADDR_SZ*CHIPID + ({DBG_ADDR_TRANS[31:16], (ADDR_IN[31:28] - DBG_ADDR_TRANS[15:12]), ADDR_IN[27:0]})
0xA000_0000 - 0xFFFF_FFFF	0	0	0	Unused	Lower 1GB on local chip.  CHIP_ADDR_SZ*CHIPID + (0x0000_0000 - 0x3FFF_FFFF)

Cortex-M7 address space	DBG_ADDR_TRANS[EN]	ADDR_TRANS[EN]	ADDR_TRANS[CMN_ATRANS_EN]	ADDR_TRANS[ADDR_47_20]	Translated address space
0x6000_0000 - 0x9FFF_FFFF	X	X	1	Unused	Translate to CMN configuration space.  CHIP_ADDR_SZ*CHIPID + (0x01_4000_0000 - 0x01_7FFF_FFFF)
0x6000_0000 - 0x9FFF_FFFF	0	0	0	Unused	1GB-2GB on local chip.  CHIP_ADDR_SZ*CHIPID + (0x4000_0000 - 0x7FFF_FFFF)

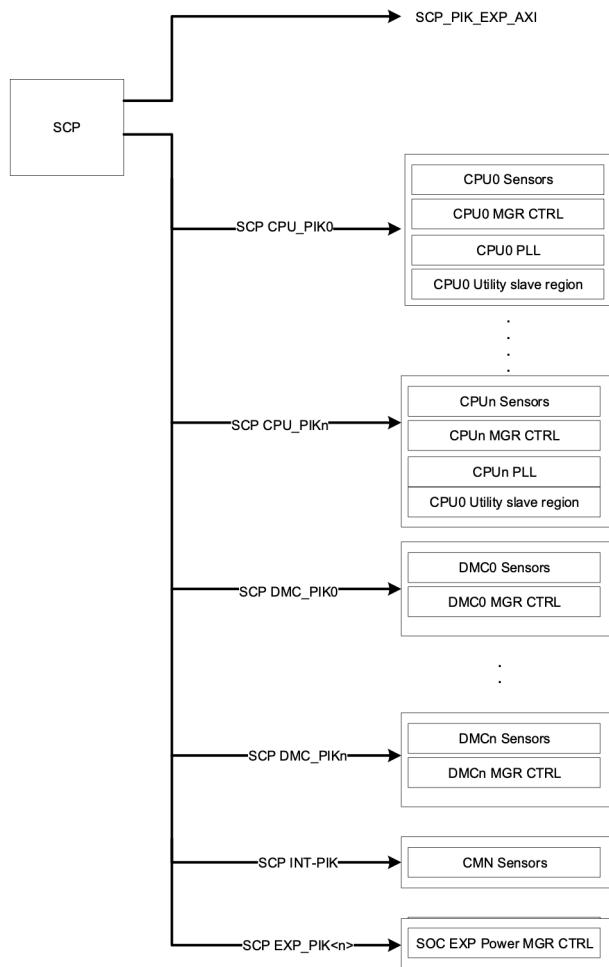
### 3.8.1.2 SCP PIK interface

The SCP PIK interface controls all the sensors and power management logic across the compute subsystem.

This interface exports expansion interfaces out of the compute subsystem to control the power management logic in the SoC. The SCP PIK interface is also used to access the utility slave memory region in each of the processor clusters.



**Note** These interfaces are internal and can be implemented using the CMN-700 PUB channel or routed outside the CMN-700 interconnect using the NI-700 non-coherent interconnect. The SCP PIK expansions are exported as AMBA AXI interfaces.

**Figure 3-19: Example implementation of the SCP PIK interface**

### 3.8.2 Manageability Control Processor block

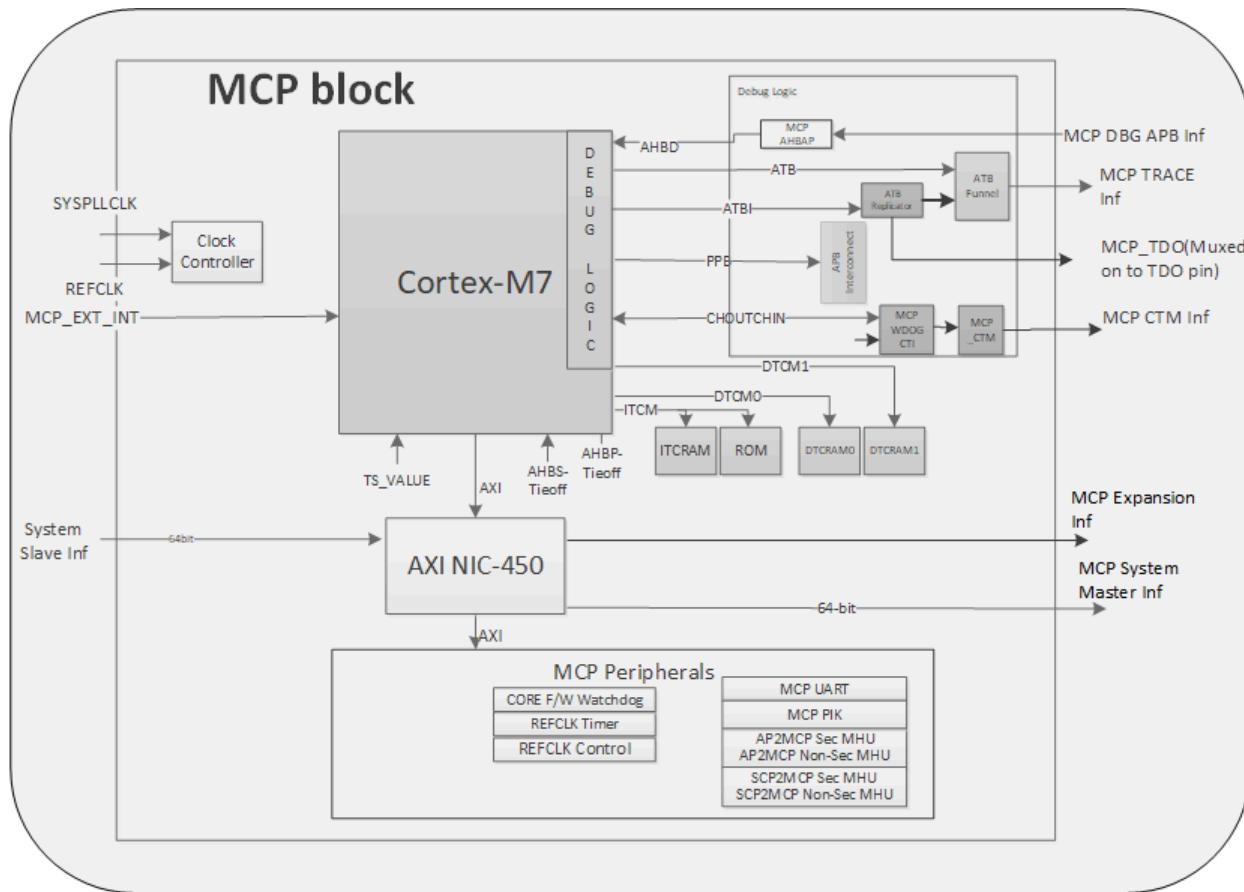
The MCP manages communications with the external Baseboard Management Controller.

The Manageability Control Processor contains a Cortex-M7 core with private peripherals such as timers, UART, instruction, and data memory. The MCP has the following functions:

- Provides boot sequencing during system start-up
- Establishes security integrity with the SCP through software
- Communicates with the external BMC
- Resets the SoC when the BMC sends a reset message or when a reset is indicated through General Purpose Input/Output (GPIO)
- Logs events and communicates with the external BMC
- Along with processor cores, responsible for all the RAS manageability

The following figure shows the architecture of the MCP block.

**Figure 3-20: MCP block architecture**



The MCP core has the following private peripheral components:

- MHU for communication with the SCP and application processor cores - For more information, see [Message communication between processors](#).
- SRAM for private data (512KB) and code (512KB)
- On-chip (Trusted) Secure boot ROM (128KB)
- PIK to handle the reset and power states of the MCP - The MCP PIK generates all the resets and supports various reset functionalities within the MCP.
- MCP UART
- Timing peripherals
- Watchdog timer
- MCP generic REFCLK timer - The REFCLK counter is an Arm generic counter that is memory mapped and running on REFCLK.

The MCP PIK, timing peripherals, watchdog timer, and generic REFCLK timer peripherals, along with the core Warm reset are all the same as for the SCP.

The MCP PIK power control is connected to the PORESETn input. This reset is used to reset all the logic within the SCP and MCP, and therefore the rest of the compute subsystem. The MCP block also provides the option to Warm reset just the MCP core. For more information, see [MCP core Warm reset](#).

The MCP block includes expansion space for adding SoC peripherals that are available for the MCP core.

### 3.8.3 Message communication between processors

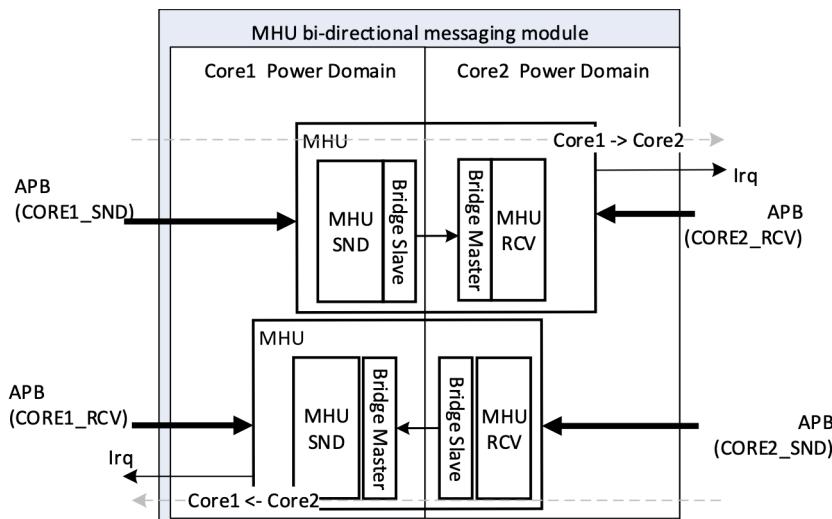
In RD-N2, messages are transferred between processors by using the MHU communication method.

The MHU supports communication between processors through the following features:

- Direct message channel
- Message channel is independent of remote bus infrastructure
- Support for remote interrupt generation
- Provision of isolation between source and target processors

RD-N2 supports the MHUv2.1 versions. Each MHU is implemented as a bidirectional messaging system. The following figure shows the MHU for communication between various application processor cores and the SCP and MCP. The figure also shows the MHU for communication between the SCP and MCP.

**Figure 3-21: MHU module for communication between various cores**



The compute subsystem contains multiple instances of the module that is shown in the preceding figure:

- One secure and one non-secure bi-directional MHU module to support messaging between application processor cores and the SCP
- One non-secure bi-directional MHU module to support messaging between the SCP and MCP
- One non-secure bi-directional MHU module to support messaging between all application processor cores and the MCP

The MHU asserts the following interrupts to be mapped between processors for message passing:

- A non-secure and secure interrupt mapped to one processor
- A non-secure and secure interrupt mapped to the other processor

These interrupts are generated by writing to a corresponding 32-bit register. The use of 32 bits per interrupt line enables software to provide more information about the source of the interrupt. For example, each bit of the register can be associated with an event type that can contribute to raising the interrupt.

Each MHU also has dedicated secure and non-secure channels (each channel is 32-bits), to write request and response messages. The numbers of channels used are configured as such:

- Secure MHU between application processor cores and SCP : 16
- Non-Secure MHU between application processor cores and SCP : 16
- Non-Secure MHU between application processor cores and MCP : 16
- Non-Secure MHU between SCP and MCP : 16

## 3.9 Clock Control block

The Clock Control block contains the clock generation logic for system-level clocks.

This block does not contain clock generation logic for clocks that are internally generated by the Processor block or the Debug block.

The key component of the Clock Control block is the Clock Generation Unit (CGU), which can be individually configured into one of several basic clock structures. For more complex system-level clock generation structures, configured units are connected so that the output clock of one CGU is used as clock source of another CGU.

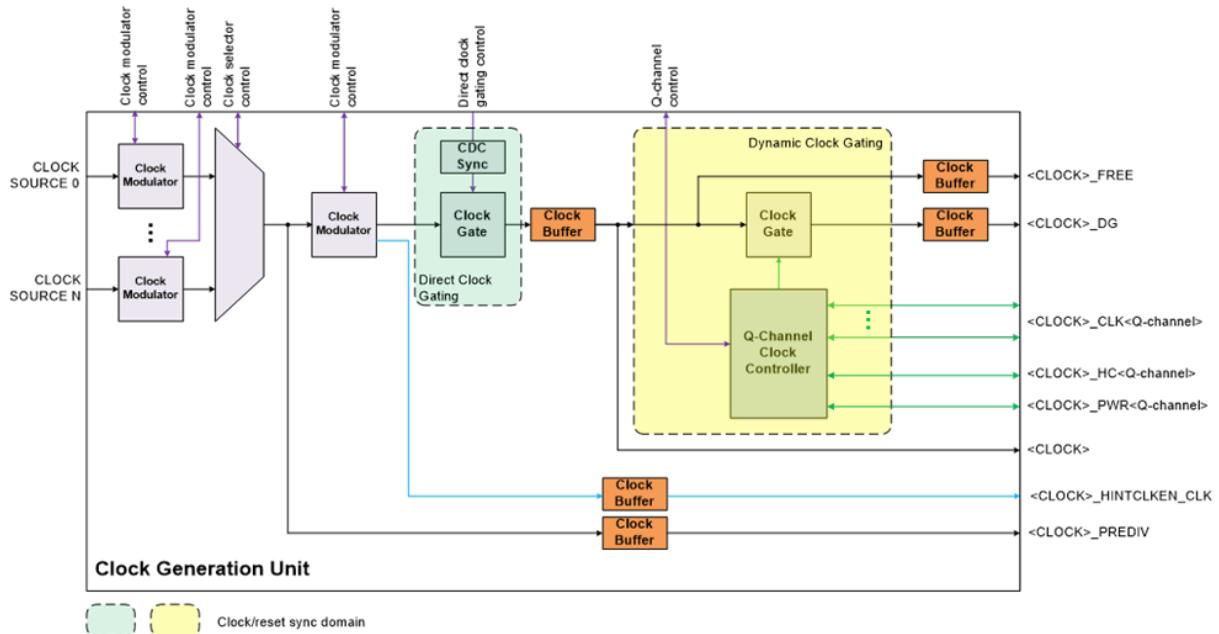
The Clock Control block provides the following configurable features:

- Number of clock sources for each generated clock
- Reset source for each generated clock
- Divider and modulator structure, shared or per-clock source
- Software clock gating
- Hardware dynamic clock gating
- Clock enable hint clock generation for n:1 synchronous divided clocks

- Buffer delay value per CGU
- Clock tree structure

The following figure shows a full CGU. The actual CGU, that is generated, is based on the corresponding clock configuration.

**Figure 3-22: Structure of a full CGU**



## 3.10 Peripheral block

RD-N2 supports multiple peripherals that are used by the application cores.

These peripherals include:

- Generic and watchdog timers:
  - **AP\_REFCLK\_CNTCTL**, the application processor generic timer control
  - Secure application processor watchdog, a generic watchdog timer for Trusted applications
  - Non-secure application processor watchdog, a generic watchdog timer for non-Trusted applications
  - **AP\_REFCLK\_S\_CNTBase1** application processor generic timer - This timer is Secure access only.
  - **AP\_REFCLK\_NS\_CNTBase0** application processor generic timer - This timer permits Non-secure accesses.

- Scratch RAMs:
  - Secure on-chip SRAM that is used by the application processor cores
  - Non-secure on-chip SRAM that is used by the application processor cores
- Application processor boot ROM:
  - Secure boot ROM, which contains the code for initializing the application processor boot process - This ROM is accessible in Secure mode only.
  - Non-secure boot ROM, which contains code that is required during firmware updates - This ROM is accessible in both Secure and Non-secure modes.
- BP140 AMBA AXI internal memory interface that is used to connect to the scratch RAMs and boot ROMs
- PL011 UARTs:
  - Secure application processor UART
  - Non-secure application processor UART

All these peripherals are connected to the main CMN-700 interconnect through NIC-450 components.

## 3.11 Dynamic Memory block

In RD-N2, the Dynamic Memory block supports DDR5.

To use DDR5, the third-party memory controller must support the following key features:

- Native AMBA CHI protocol-based third-party DDR5 controllers
- Two multiport AMBA® 5 CHI Architecture Specification, issue E interfaces with managed QoS to the DDR memory controller
- Configurable data bus widths of 256 bits on the AMBA® 5 CHI Architecture Specification, issue E interfaces
- DFI5.0 PHY interface
- Multiple DFI clock ratio for DDR5 with efficient phase-aware scheduling
- Up to two ranks and 32 logical ranks
- Multichannel DRAM interface, with dual 32-bit and 40-bit channels
- JEDEC standard DDR5 SDRAMs and DIMMs
- DDR5 RDIMM that supports a high-bandwidth design with up to 64 CAM entries for reads and 64 CAM entries for writes - The latency must be as low as eight clock cycles.
- Read reorder buffering to enable maximum scheduling flexibility for strongly ordered command streams
- Phase Aware Scheduling (PAS) that enables any command to issue any DFI phase allowing maximum memory efficiency

- Multiple RAS features.

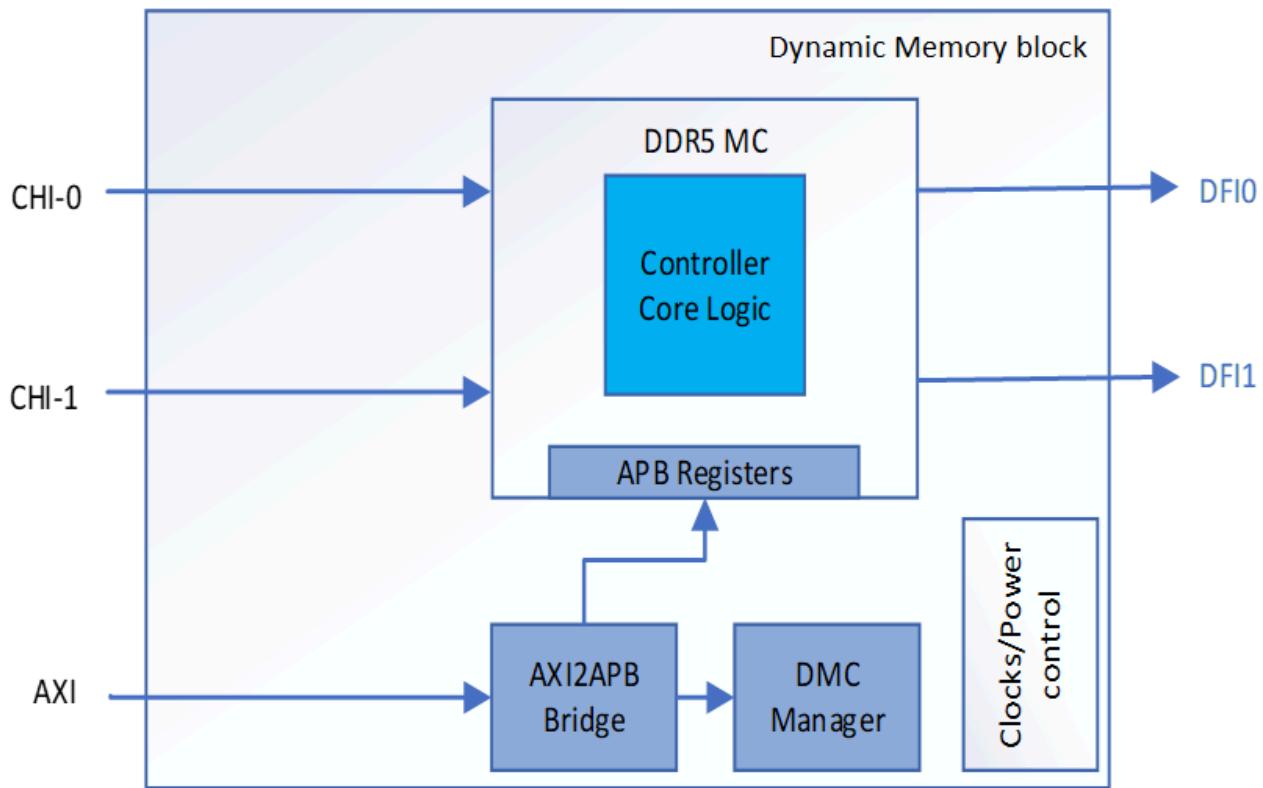
The third-party memory controller must also support the following key features of the Dynamic Memory block:

- CBusy feature in the AMBA CHI protocol-based controller
- TrustZone memory region support in the third-party memory controller itself for the AMBA CHI protocol-based controller
- MPAM support for bandwidth monitoring and allocation - Partition-based bandwidth allocation is controlled by sending the CBusy indication for the partition back to the requestor. It is the responsibility of the requestor to take appropriate action by reducing the number of outstanding transactions for that particular partition.

For more information, see the documentation for the third-party memory controller.

The following figure shows an example CHI protocol-based DDR Dynamic Memory block. It consists of a third-party DDR5 memory controller with two AMBA CHI interface ports configured.

**Figure 3-23: Example DDR Dynamic Memory block**



The Dynamic Memory block lies in the VSYS voltage domain and is completely encapsulated inside the AONTOP power domain and the SYSTOP reset domain.

The Dynamic Memory Controller (DMC) manager provides a memory region to include any logic that must be added outside of the DMC.

For security implementation details, see the documentation for the third-party memory controller.

# 4 Functional description

Components in RD-N2, such as clocks and resets, work across multiple functional blocks.

## 4.1 Clocks

The clocks in RD-N2 are categorized according to their type and the functional blocks in which they are used.

RD-N2 includes the following types of clocks:

- [Input clocks](#). Input clocks must be provided with a source that is external to, or embedded in, the compute subsystem.
- [Internal clocks](#). Internal clocks are generated and used in the compute subsystem.
- [Output clocks](#). Output clocks are source clocks for peripherals and other components that are integrated into an SoC externally to the compute subsystem.

For more information on which input, internal, and output clocks are used in each of the functional blocks, see [Clocks by functional block](#).

### 4.1.1 Input clocks

The following table lists the input clocks in the RD-N2, some of which are driven by Phase Locked Loops (PLLs).

A lock signal input must be provided for each clock input coming from a PLL. The input indicates that the PLL providing the clock is locked and that the clock is available. The SCP uses the PLL lock signals.

Configuration registers for the PLLs must be implemented in the SoC. These registers must be accessible from the SCP. The specification of these registers can be **IMPLEMENTATION DEFINED**.

**Table 4-1: Input clocks**

Clock signal	PLL lock signal	Description
<b>REFCLK</b>	N/A	Main input clock that the SCP and MCP boot when coming out of reset. This clock is always on.  <b>REFCLK</b> is also the slow clock input select option for all the system clocks that are generated internally to the compute subsystem.
<b>CPUPLLCLK&lt; n &gt;</b>	<b>CPUPLLLOCK&lt; n &gt;</b>	CPU PLL. There is one per core, so n = 0–31.  Optionally, an implementation can choose to share a <b>PLL(SHCOREPLLCLK&lt; n &gt;)</b> among group of cores.
<b>INTPLLCLK</b>	<b>INTPLLLOCK</b>	PLL that generates the interconnect clock.
<b>SYSPLLCLK</b>	<b>SYSPLLLOCK</b>	PLL clock input for the main system. <b>SYSPLLCLK</b> is used to generate the clocks for PPUs.

Clock signal	PLL lock signal	Description
DDRPLLCLK	DDRPPLLLOCK	Optional PLL clock input for the DDR Dynamic Memory block if this block is included in the compute subsystem.
EMCLK0- EMCLK<n-1>	N/A	Clock input for external master expansion I/O ports with TBU, NCI, or ITS present on the expansion I/O path. "n" represents the number of external master expansion I/O ports.
EMLTICLK0- EMLTICLK<n-1>	N/A	Clock input for external LTI expansion I/O ports. "n" represents the number of external LTI expansion I/O ports.
EMATSCLK0- EMATSCLK<n-1>	N/A	Clock input for external DTI-ATS expansion I/O ports. "n" represents the number of external DTI-ATS expansion I/O ports.
EMSICLK0- EMSICLK<n-1>	N/A	Clock input for external ITS MSI expansion I/O ports. "n" represents the number of external ITS MSI expansion I/O ports.
SWTCKCLK	N/A	Clock that drives the combined JTAG and SWD interface.
TRACECLKIN	N/A	Optional input clock for the TPIU interface, if the TPIU is implemented.



In general, the RTL implementation can choose to have different names for the signals to match the specific implementation.

RD-N2 is not limited to only the input clocks that are listed in the preceding table. Based on a complete SoC definition or microarchitectural requirements, the number of input clocks to the compute subsystem can vary.

#### 4.1.2 Internal clocks

RD-N2 internally derives clocks that are used in the components of the compute subsystem.

These clocks are only generated when the VSYS.SYSTOP PPU is in the ON state. Each clock can be controlled individually.

The main internally derived clocks for the compute subsystem are shown in the following tables.



Depending on the SoC definition or microarchitectural implementation, the internally derived clocks can change in a compute subsystem. Some clocks might not be used or additional clocks could be needed due to physical implementation changes.

**Table 4-2: Compute subsystem internally generated clocks**

Clock signal	Description	Source clock	Divider	Hardware clock gating	Description
INTCLK	Coherent interconnect clock	REFCLK INTPLLCLK	Y	Y	Clocks the interconnect and other expansion master and slave ports. This clock is generated from INTPLLCLK.  Note that hardware gating can be optional and a choice can be made to do only software clock gating.

Clock signal	Description	Source clock	Divider	Hardware clock gating	Description
INT_ATCLK	CMN-700 trace clock	INTCLK	Y	Y	INTCLK divided by two or four. This clock is gated when INTCLK is gated.
IONCICLK	I/O interface NCI clock	REFCLK SYSPLLCLK	Y	Y	Clocks the NCI network for PCIe and other I/O interfaces.
SYSPERCLK	Interconnect clock	REFCLK SYSPLLCLK	Y	Y	Clocks for the system peripheral and various NIC components. This clock is generated out of SYSPLLCLK.
SYSDBGPCLK	Debug APB clock	REFCLK SYSPLLCLK	Y	N	Clocks the APB interfaces and bridge at the top level.
DBGCLK	Debug clock	REFCLK SYSPLLCLK	Y	N	Debug clock for all system trace and trigger logic, except for the APB interface logic.
DBGCH< n >ATCLK	Debug ATB clock	REFCLK SYSPLLCLK	N	N	Clocks the ATB interfaces from the debug logic to the debug daisy chains.  There is a dedicated clock for each debug daisy chain supported.
DBGCH< n >PCLKDBG	Debug APB clock	REFCLK SYSPLLCLK	N	N	Clocks the APB interfaces for the debug chains.  There is a dedicated APB clock for each debug daisy chain supported.
GICCLK	GIC clock	REFCLK SYSPLLCLK	Y	Y	Clocks the GIC-D and waker logic.  Note that the SPI Collator runs off the free running GICCLK.
TCUCLK	SMMU clock	REFCLK SYSPLLCLK	Y	Y	SMMU TCU clock.  IMPLEMENTATION DEPENDENT. All TCUs can use the same TCUCLK or each TCU can have its own clock.
APUARTCLK	AP UART clock	REFCLK SYSPLLCLK	Y	Y	Clock for the application processor Secure and Non-secure UART.

**Table 4-3: SCP block internally generated clocks**

Clock signal	Description	Source clock	Divider	Hardware clock gating	Description
SCPCORECLK	SCP core clock	REFCLK SYSPLLCLK	Y	Y	Clocks the core of the SCP
SCPPIKCLK	SCP PIK interface clock	REFCLK SYSPLLCLK	Y	Y	Clocks the SCP PIK interfaces
SCP_UARTCLK	UART clock	REFCLK	N	N	Clock for the SCP UART
SCPACLK	SCP expansion clock	REFCLK SYSPLLCLK	Y	Y	Clock for the SCP AON expansion logic

Clock signal	Description	Source clock	Divider	Hardware clock gating	Description
SCP_SYSPIKCLK	System PIK clock	REFCLK SYSPLLCLK	Y	Y	Clock for the system PIK logic

**Table 4-4: MCP block internally generated clocks**

Clock signal	Description	Source clock	Divider	Hardware clock gating	Description
MCPCORECLK	MCP core clock	REFCLK SYSPLLCLK	Y	Y	Clocks the core of the MCP
MCPACLK	MCP expansion clock	REFCLK SYSPLLCLK	Y	Y	Clock for the MCP AON expansion logic
MCP_UARTCLK	UART clock	REFCLK	N	N	Clock for the MCP UART

**Table 4-5: Processor block internally generated clocks**

Clock signal	Description	Source clock	Divider	Hardware clock gating	Description
CORECLK<n>	Main AP core clock	REFCLK CPUPLLCLK<n>	Y	N	Each core is clocked independently of the others. The clock for each core is generated from dedicated CPU PLLs for each core.  Implementations can choose to share a PLL between multiple cores, which is <b>IMPLEMENTATION DEFINED</b> .
CLUS_SCLK<n>	AMBA CHI interface clock	INTCLK	N	N	Clock for the cluster AMBA CHI interface logic.
CLUS_PCLK<n>	Cluster debug APB clock	DBGCHUPPCLK<n>	N	Y	Clocks the DSU debug APB interface logic.
DBGBLK_PCLK<n>	Debug block APB clock	DBGCHUPPCLK<n>	N	Y	Clock from the debug TOP APB interface logic.
CLUS_ATCLK<n>	DSU cluster trace clock	DBGCHUPATCLK<n>	N	Y	Clock for the cluster debug chain CTIs and trace interface logic.
CLUS_GICCLK<n>	Cluster GIC clock	SCLK	Y	Y	Clock for the cluster GCI (PPI/SGI) interface logic.
CLUS_PERIPHCLK<n>	Cluster peripheral clock	CPUPLLCLK<n>	Y	Y	Clock for the peripheral interface logic.
CLUS_PPCLK	Cluster PPU clock	CORECLK<n>	Y	Y	Clock for the cluster PPU logic.



- The divider is only on the faster clock. The **REFCLK** is not divided when it is selected.

- Hardware gating for the clocks can be optional depending on the implementation. The choice can be made to perform only software clock gating, as defined by the clock control registers.

These clocks are asynchronous to each other and can be managed independently using the following controls in the compute subsystem:

- Clock source selection.
- Clock division with all integer ratios between 1 and 16 supported.
- Clock force enabled. When a clock is force enabled, hardware dynamic clock gating is turned off and the clock is always on. This mode is only intended for internal use when debugging clock control software. Some clocks do not have this mode because they do not have any transparent hardware clock gating, so this mode is not needed.

Clock selection is glitchless, and dividers produce a clock with a 50:50 duty cycle.

### 4.1.3 Output clocks

Output clocks are generated internally within the compute subsystem and routed out of the subsystem for external SoC integration purposes.

RD-N2 supports the Q-Channel interfaces to the output clocks to facilitate clock gating, except for the clocks that are suffixed with "\_FREE".

**Table 4-6: Output clocks**

Clock	Description
<b>INTCLK / INTCLKFREE</b>	Output clock for coherent expansion AMBA ACE-Lite slave and master interfaces. SoC integration can choose to run the I/O masters in a SoC-defined clock. In such cases, a domain bridge logic must be enabled accordingly. Output clock for expansion AMBA ACE-Lite master ports, EACEM<n>.
<b>MC&lt;n&gt;_CLK</b>	Memory controller output clock to DDR PHY. "n" represents the number of memory controller interfaces in the system. This clock is from the same source as <b>MC&lt;n&gt;_DFICLK</b> . <b>MC&lt;n&gt;_CLK</b> is optional, depending on the DFI/PHY and clock ratio support requirements, and must be used accordingly. For example, a third-party memory controller might not require this clock.
<b>MC&lt;n&gt;_DFICLK</b>	Optional memory controller DFI output clock for DFI master interfaces. "n" represents the number of memory controller interfaces in the system.
<b>GICCLK</b>	GIC clock. The <b>GICCLK_FREE</b> is a free running clock, not a gated version of <b>GICCLK</b> . Both versions of the clocks are exported out of the subsystem.
<b>GICCLK_FREE</b>	All the incoming GIC_EXT_INT interrupts are assumed to be level sensitive. If the incoming interrupts are edge triggered, then they must be synchronized to this clock domain. The interrupts must be as wide as one clock pulse of this clock.
<b>SYSPERCLK</b>	Output clock for the EXMAXI expansion AMBA AXI4 master interface. The free running version is also supported. Both versions of the clocks are exported out of the subsystem.
<b>SYSPERCLK_FREE</b>	
<b>SCPPIK_CLK</b>	Clocks the SCP expansion PIK interfaces from the SCP block. Both versions of the clocks are exported out of the subsystem.
<b>SCPPIKCLK_FREE</b>	
<b>SYSDBGPCCLK</b>	Output clock for the EXMAPB CoreSight APB master expansion interface.

Clock	Description
<b>DBGCLK</b>	Output clock for the EXSATB< n > CoreSight ATB input expansion interface.
<b>TRACECLKOUT</b>	Optional output clock for the TPIU interface, if the TPIU is implemented.
<b>SCPAXICLK</b>	Output clock for the SCP AMBA AXI expansion interface. Exporting this interface at the subsystem level is just a reference implementation.
<b>MCPAXICLK</b>	Output clock for the MCP AMBA AXI expansion interface. Exporting this interface at the subsystem level is just a reference implementation.
<b>SCPCORECLK</b>	Output clock for the SCP interrupt expansion interface. This clock is used by the EXSCPINT interface.
<b>MCPCORECLK</b>	Output clock for the MCP interrupt expansion interface. This clock is used by the EXMCPINT interface.

#### 4.1.4 PLL lock control

The PLL lock input signals are used to generate lock and unlock interrupts to the SCP Cortex-M7 Nested Vectored Interrupt Controller (NVIC).

#### 4.1.5 Clocks by functional block

The functional blocks in RD-N2 contain various clocks.

##### 4.1.5.1 Interconnect block clocks

There is single clock domain for the coherent mesh network.

The mesh clock **INTCLK** is generated by Clock Control block logic. For more information about **INTCLK**, see [Internal clocks](#).

##### 4.1.5.2 Processor block clocks

Each core in the Processor block is clocked independently of the others.

The clock for each core is generated from dedicated CPU PLLs for each core. For more information about clocks in the Processor block, see [Internal clocks](#).

##### 4.1.5.3 Interrupt block clocks

The distributed GIC components are clocked at different clock domains that are generated by system-level clock control or core clock control.

The key distributed GIC components and their clock domains are shown in the following table.



The source clock in the table can be changed to a different clock based on the specific system implementation.

Note

**Table 4-7: GIC components and their clocks**

Clock	Logic clocked	Description
<b>GICCLK</b>	Main GIC Distributor logic	Generated from the system PLL.
<b>ITS&lt; n &gt;_CLK</b>	Respective ITS logic clock	ITS blocks in the same clock domain as their respective I/O master interface clock domain or the same clock as the AMBA AXI4-Stream IC. Dependent on the implementation. System ITS is clocked with <b>GICCLK</b> .
<b>GCI&lt; n &gt;_CLK</b>	GIC GCI logic	PPIs and SGIs in the GCI are in their respective cluster or core GIC clock domain. Dependent on the implementation. Can be in the CMN-700 interconnect clock domain.
<b>GIC_SPI_CLK</b>	GIC SPI logic	Must be a free running clock. Dependent on the implementation. If a global SPI block is implemented, this clock can be same as the GIC clock, but asynchronous to the main <b>GICCLK</b> .

#### 4.1.5.4 I/O Virtualization block clocks

The distributed SMMU components are clocked at different clock domains that are generated by system-level clock control.

The clocks can be the same, but each TCU clock domain is asynchronous to the others. For more information about the clocks in the I/O Virtualization block, see [Internal clocks](#).

**Table 4-8: SMMU clocks**

Clock	Logic clocked	Description
<b>TCUCLK&lt; n &gt;</b>	Respective TCU block clock	Can be the same clock, but each TCU clock domain is asynchronous to the others.
<b>TCUATSCLK&lt; n &gt;</b>	DTI-ATS clock	Clock for external DTI-ATS interfaces from the PCIe I/O masters. Input <b>EMATSCLK&lt; n &gt;</b> clocks this clock.
<b>TBUAXICLK&lt; n &gt;</b>	Clock for each TBU in Inline mode	The architecture assumes the same clock as the I/O master clock. This clock can be internally generated if the I/O master is internal to the Processor block. Alternatively, the clock can be the same as the I/O AMBA AXI master interface clock <b>EMCLK&lt; n &gt;</b> if the TBU AMBA AXI is ported as an expansion port.
<b>TBULTICLK&lt; n &gt;</b>	Clock for TBU in LTI mode	The architecture assumes the same clock as the I/O master LTI interface clock. This clock can be internally generated if the I/O master is internal to the Processor block. Alternatively, the clock can be the same as the I/O master LTI interface clock <b>EMLTICLK&lt; n &gt;</b> if the TBU LTI is ported as an expansion port.

#### 4.1.5.5 MSCP block clocks

The clock control logic inside the SCP and MCP blocks generates all the clocks that are required by the Cortex-M7 cores. This logic also generates clocks for various peripherals inside the blocks.

For more information, see [Internal clocks](#).

## 4.2 Counters and timers

The following sections describe the counters and timers in RD-N2.

### 4.2.1 System generic counter

The system generic counter is an implementation of the memory-mapped generic counter module that is defined in the Armv8.x architecture.

This counter implements the CNTControlBase and CNTReadBase frames that are defined by the Arm architecture. These frames are mapped in the application processor memory map and reside in the AONTOP power domain.

The CNTControl frame is always Secure. The following features are implemented:

- 64-bit counter
- Single frequency mode – fixed count increment and fixed counter clock frequency
- Counter logic and APB interfaces are synchronous
- Software-enabled halt-on-debug from an external CoreSight CTI
- Security aspects must be handled externally
- Counter output in natural binary

For more information, see the [Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile](#).

### 4.2.2 Generic timers

There are multiple timers in the RD-N2 subsystem.

#### 4.2.2.1 REFCLK time domain

There is a single REFCLK time domain. The application processor cores, the SCP, the MCP, and the Debug System Trace Macrocell (STM) operate in this time domain.

The system generic counter generates a time stamp value for the REFCLK time domain. CoreSight components also use this time stamp for trace generation. The time stamp value must be routed and balanced to all the various application processor, SCP, MCP, and debug components across the system.



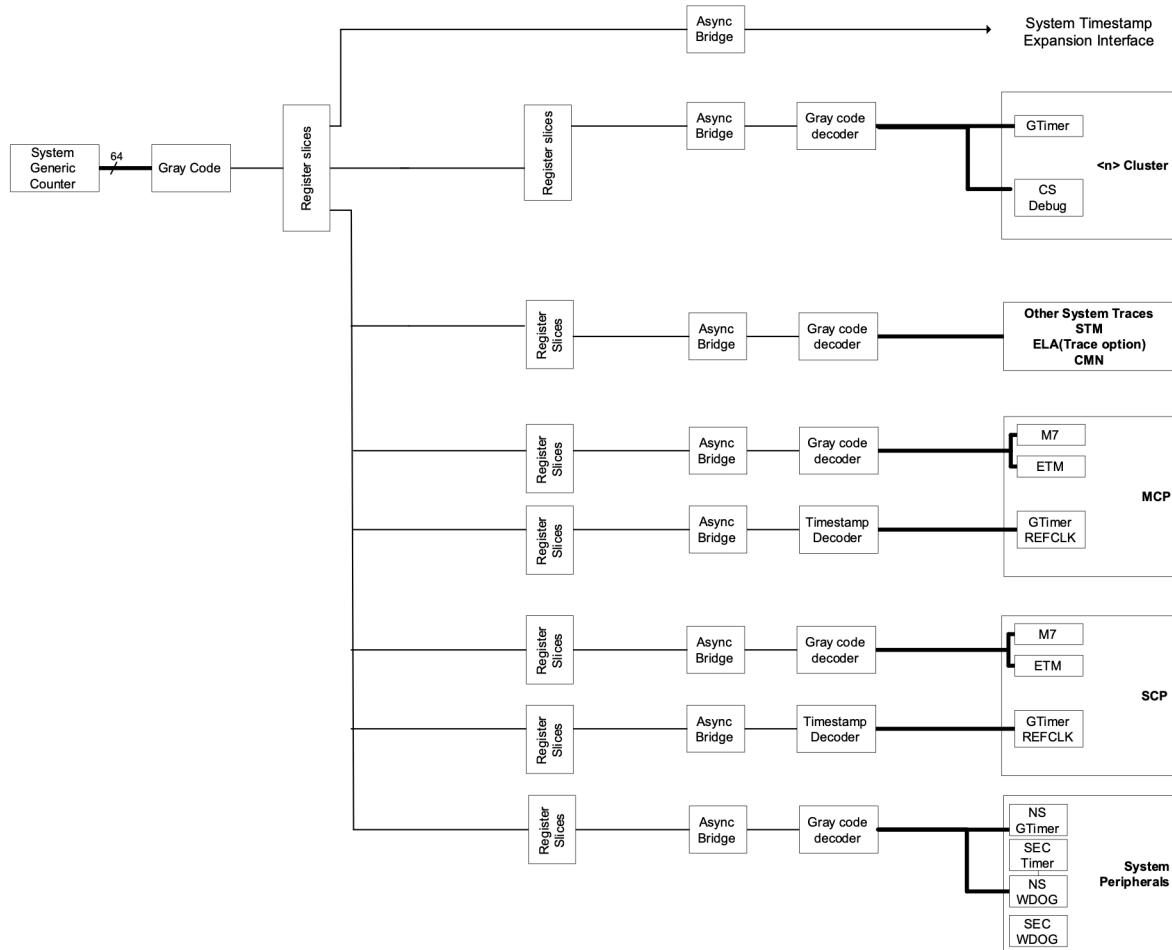
The REFCLK time domain is halted during debug.

Note

In multichip implementations, the REFCLK time domain of each chip is globally synchronized using the global timer synchronization method.

The REFCLK time domain contains several timers:

- All application cores implement the Arm Generic Timer, which is defined in the *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile*. The interrupts from these timers are mapped as PPI interrupts through Redistributor (GIC Cluster interface) in the GIC.
- SCP\_REFCLK generic timer, for private use by the SCP. For more information, see [SCP\\_REFCLK generic timer](#).
- MCP\_REFCLK generic timer, for private use by the MCP. For more information, see [MCP\\_REFCLK generic timer](#).
- Two more REFCLK generic timers, one Secure and one Non-secure, for use solely by the application processors for general purposes. For more information, see [AP\\_REFCLK generic timer](#).
- Non-secure watchdog timer used by the application processor cores.
- Secure watchdog timer used by the application processor cores.

**Figure 4-1: System generic count value distribution**

#### 4.2.2.2 SCP\_REFCLK generic timer

The SCP includes a private memory-mapped Arm Generic Timer, which is defined in the Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile.

The SCP\_REFCLK generic timer provides a single timer frame, without a second view, and without a virtual timer capability. This timer is called the SCP REFCLK generic timer, SCP Timer control frame, and SCP Timer frame in the SCP memory map, and is in the VSYS.AONTOP power domain. The SCP core uses this timer for creating various time events to track the task. It is also used for waking up the SCP core when in Wait for Interrupt (WFI).

### 4.2.2.3 MCP\_REFCLK generic timer

The MCP includes a private memory-mapped Arm Generic Timer, which is defined in the Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile.

The MCP\_REFCLK generic timer provides a single timer frame, without a second view, and without a virtual timer capability. This timer is called the MCP REFCLK generic timer, MCP Timer Control frame and MCP Timer frame in the MCP memory map, and is in the VSYS.AONTOP power domain. The MCP core uses this timer for creating various time events to track the task. It is also used for waking up the MCP core when in WFI.

### 4.2.2.4 AP\_REFCLK Generic Timer

RD-N2 includes two memory-mapped Arm Generic Timers for general-purpose functions, which are defined in the Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile.

Each timer provides single frames, without a second view, and without a virtual timer capability. These timers are called the AP\_REFCLK generic timer, AP Generic Timer Control Frame, AP Secure Generic Timer Frame and AP Non-secure Generic Timer frame in the application processor memory map, and are in the PD\_SYSTOP power domain. Access to the AP Non-secure Generic Timer frame is Non-secure, while the AP Generic Timer control Frame, and AP Secure Generic Timer frame are Secure.

## 4.2.3 Watchdog timers

The following sections describe watchdog timers in RD-N2.

### 4.2.3.1 AP Non-secure watchdog timer

RD-N2 implements a Non-secure watchdog timer that is compliant with the Arm® Server Base System Architecture, version 6.0. The system generic counter provides the timestamp for this timer.

This watchdog generates two interrupts Watchdog Signal0 (WS0) and Watchdog Signal 1 (WS1). The WS0 interrupt is configured as an EL2 interrupt and is routed as a GIC SPI. This interrupt is also sent to the SCP to support logging.

The WS1 interrupt is routed as an SPI and can be configured as EL3 interrupt. It is also sent to the SCP to support logging according to the requirements of the Arm® Server Base System Architecture, version 6.0. The action taken on the raising of WS1 is software **IMPLEMENTATION DEFINED**.

The generic watchdog is managed by two memory-mapped register frames - AP Non-secure Watchdog Control and Refresh frame. See [AP system memory map](#). In the compute subsystem, these frames are accessible by Non-secure accesses from the application processors.

This watchdog timer is clocked by REFCLK.

#### 4.2.3.2 Secure Watchdog Timer

A Secure watchdog timer is available to the application processors in Secure mode. The watchdog timer is compliant with the Arm® Server Base System Architecture, version 6.0. The system generic counter provides the timestamp for this timer.

This watchdog generates two interrupts, Watchdog Signal0 (WS0) and Watchdog Signal 1 (WS1). The WS0 interrupt is configured as an EL2 interrupt and is routed as a GIC SPI. This interrupt is also sent to the SCP to support logging.

The WS1 interrupt is routed as an interrupt to the SCP. The SCP takes appropriate action, which is at minimum a Warm reset of all cores.

The generic watchdog is managed by two memory-mapped register frames - AP Secure Watchdog Control and Refresh Frame. In the RD-N2 subsystem, these frames are accessible by Secure accesses from the application processors.

If a global compute subsystem reset is the response the SCP takes on second expiration, the Trusted Watchdog state is not preserved through reset.

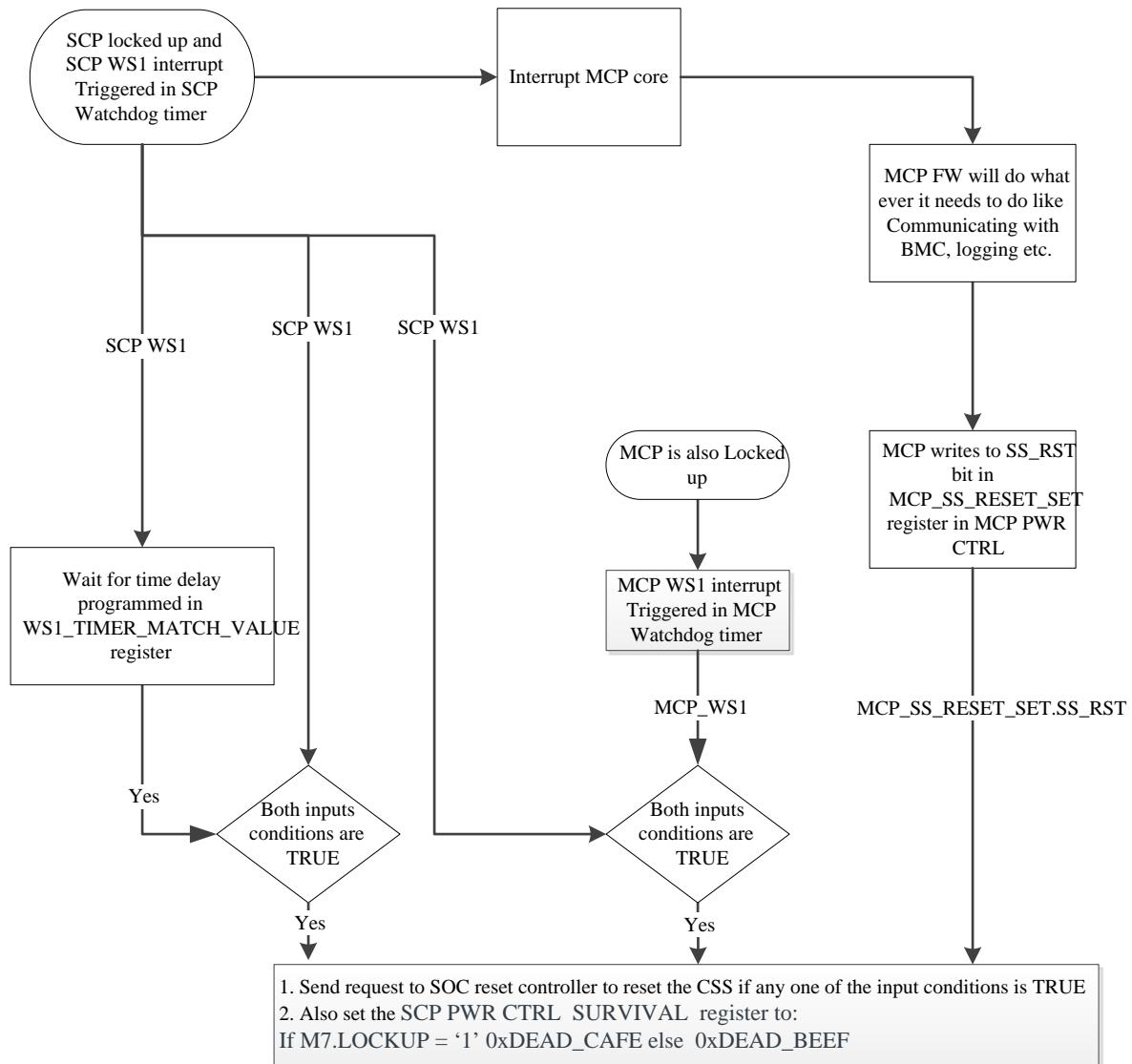
This watchdog timer is clocked by REFCLK.

#### 4.2.3.3 SCP Watchdog Timer

The SCP block includes a Cortex-M System Design Kit watchdog timer that protects against lockups in the firmware.

This watchdog generates two interrupts, Watchdog Signal0 (WS0) and Watchdog Signal 1 (WS1). The WS0 interrupt is routed as an interrupt to the SCP. The WS1 interrupt is routed as an interrupt to MCP.

If the SCP core fails to clear the WS0 interrupt, then the WS1 interrupt will interrupt the MCP core. The following figure shows the flow of the watchdog reset sequence.

**Figure 4-2: SCP watchdog timer reset sequence**

The status for this compute subsystem reset is recorded by hardware in the SCP Power Control SURVIVAL\_RESET\_STATUS register as 0xDEAD\_BEEF.

This watchdog timer is clocked by **REFCLK**.

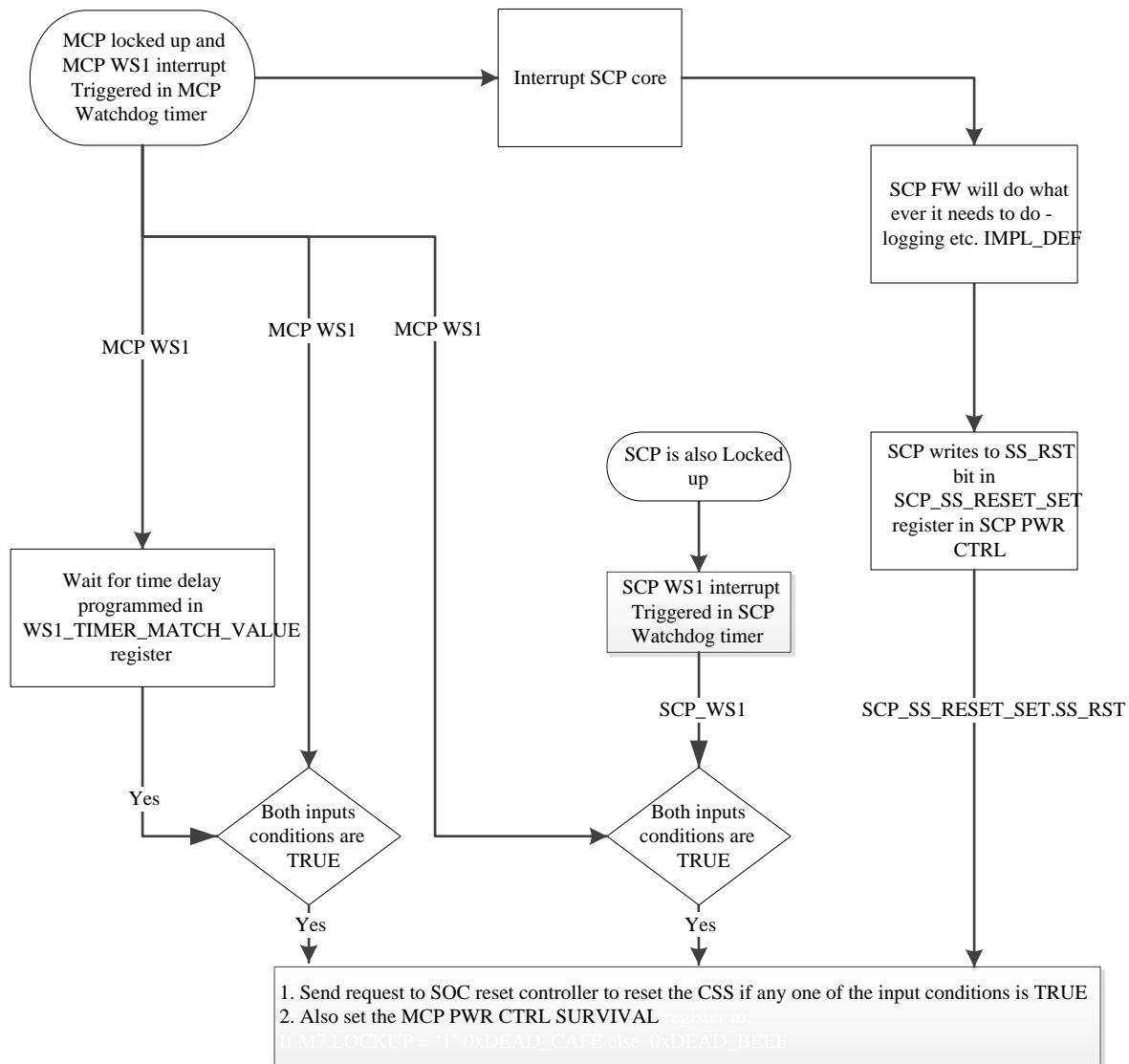
#### 4.2.3.4 MCP Watchdog Timer

The MCP block includes a Cortex-M System Design Kit watchdog timer that protects against lockups in the firmware.

This watchdog generates two interrupts, Watchdog Signal0 (WS0) and Watchdog Signal 1 (WS1). The WS0 interrupt is routed as an interrupt to the MCP. The WS1 interrupt is routed as an interrupt to SCP.

If the MCP core fails to clear the WS0 interrupt, then WS1 interrupt will interrupt the SCP core. The following figure shows the flow of the watchdog reset sequence.

**Figure 4-3: MCP watchdog timer reset sequence**





The status for this compute subsystem reset is recorded in the SCP Power Control SURVIVAL\_RESET\_STATUS register as 0xDEAD\_BEEF.

This watchdog timer is clocked by **REFCLK**.

#### 4.2.3.5 Watchdog security

Both SCP watchdog and Secure watchdog timers are accessible by Secure accesses only. These watchdogs also support halt-on-debug functionality, enabling cross triggers to halt the watchdog.

### 4.2.4 Power down considerations

Each core has Arm architectural timers that reside in the Direct connect DSU, the processor bridge cluster logic.

A core can wake from timer interrupts generated by these timers after entering WFI or Wait for Event (WFE) mode. Cores can enter the WFI and WFE modes without any side effects relating to their timers.

When core and cluster logic is also powered down, the respective core timer state is lost. So, extra steps must be taken to save timer state to AP\_REFCLK timer timers.

There are two models for powering down a core, referred to here as AP Wakeup and SCP Wakeup.

#### 4.2.4.1 AP Wakeup model

In this model, if the Arm architectural timers of a core are also powered down, software running on the applications core must take certain actions.

Software must save the core generic timer state to the AP\_REFCLK timer in the AONTOP power domain and SYSTOP reset domain, and execute WFI.

When the core is in WFI, it is powered down by the hardware. Any application processor interrupts from the AP\_REFCLK timer expire that are mapped to this core wakes up the core.

#### 4.2.4.2 SCP Wakeup model

This model applies when the system goes into the CSS.sleep1 state.

The SCP can wake up on external interrupts that are mapped to the SCP. Alternatively, the SCP can use the state from the SCP\_REFCLK generic timer to schedule wake up of SYSTOP. On receiving an interrupt when the SCP\_REFCLK timer expires, or any external interrupt, the SCP:

- Brings up the SYSTOP domain and restores the state of the system
- Brings up the corresponding core out of reset

For information about the sequence to enter the CSS.SLEEP1 state, see [SYSTOP domain reset control sequences](#).

Before the SCP powers down VSYS.PD\_SYSTOP, if the AP\_REFCLK Generic Timers are intended to be used for system wakeup, save the state of the AP\_REFCLK Generic Timer (subtract PD\_SYSTOP and Core power up time) into SCP\_REFCLK Generic Timer before powering down PD\_SYSTOP.

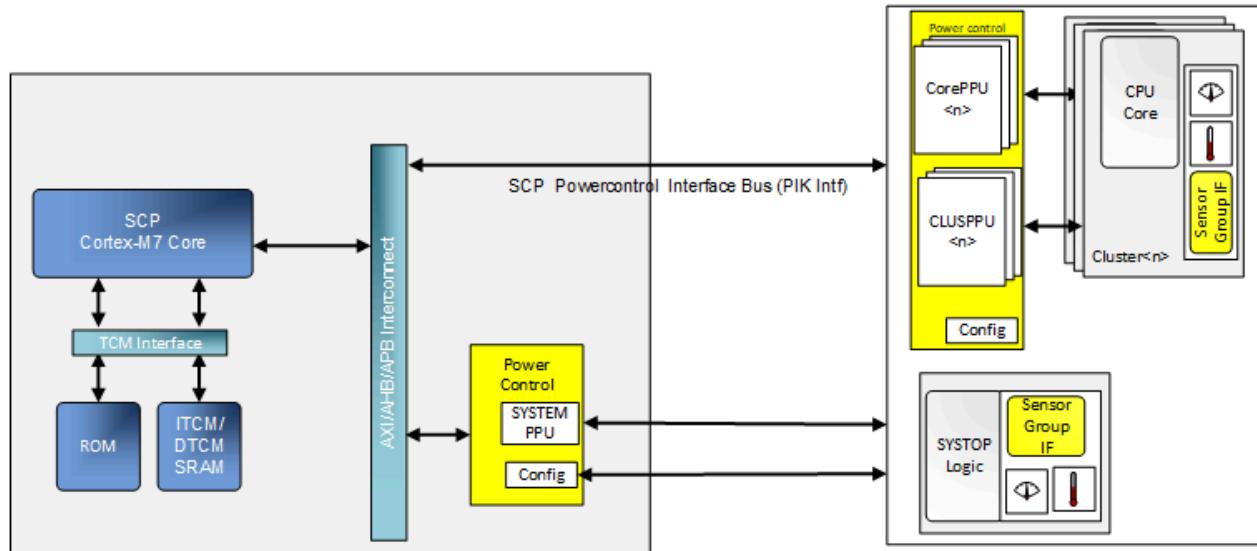
## 4.3 System power management and domains

RD-N2 supports various features for power management.

The following features are supported:

- The SCP controls power, clock, reset, and the static configuration of the subsystem or block
- Multiple voltage domains to allow for DVFS on application cores
- Multiple power-gated regions providing comprehensive leakage management
- Low-power, autonomous wake and sleep mode
- Globally Asynchronous Locally Synchronous (GALS) clocking architecture:
  - All primary clock domains are independent from each other
  - Simplifies frequency scaling and interfacing
  - Enables an accelerated route to a higher quality result for the physical design
- Hierarchical clock gating across the system

The following figure shows an overview of how the distributed power management system has been implemented in RD-N2. The power control blocks such as the MSCP block, the PPUs, and the sensors are distributed across the system.

**Figure 4-4: Distributed power management in RD-N2**

### 4.3.1 Power and voltage domains

RD-N2 supports various voltage domains within the compute subsystem.

The following voltage domains are supported:

- VSYS - Voltage domain for the rest of the compute subsystem. DVFS is not supported.
- VCPUn - Separate voltage domain and DVFS for each core. Optionally, multiple cores can be grouped and share a voltage domain.

RD-N2 supports the following top-level power domains within the voltage domains:

- AONTOP - Power domain for the AON part of the logic, which includes all logic except the cores. This domain is separate but not power gated. AONTOP can be expanded to encompass other logic components in the rest of the SoC.
- SYSTOP - Top-level system logic implemented as a separate reset domain.
- CLUSTOPn - Individual core cluster reset domain. All the logic in the processor bridge is in the SYSTOP reset domain.
- PD\_CCPUn - Power-gated regions. One power domain per core.

The following table shows the logic that resides in the various power domains.

**Table 4-9: Compute subsystem power and reset domains, and PPU implementation**

Domain	Logic in domain	Associated PPU	PPU functionality	Power states supported
SYSTOP	Interconnect block	SYSTOP PPU: PPU for the system logic.  SYSTOP_PPU resides in the SCP block.	Reset Domain Controller.	ON
	Interrupt block		Allows you to reset only the SYSTOP logic.	OFF
	I/O Virtualization block			WARM_RESET
	Dynamic Memory block			
	Peripheral block			
	Clock Control block			
CLUSTOPn	Core cluster-level logic	CLUSTOPn PPU: PPU for core cluster logic.	Reset Domain Controller.	ON
		CLUSTOPn PPU resides in the Direct connect DSU.	Allows you to reset each cluster logic separately.	OFF WARM_RESET DBG_RECov
	Core logic	CPUn PPU: PPU for the processor core logic.  CPUn PPU resides in the Direct connect DSU.	Power Domain Controller.	ON
			Allows you to power gate the corresponding core.	OFF WARM_RESET EMULATED OFF DBG_RECov

### 4.3.1.1 Power domains by functional block

The functional blocks in RD-N2 are in various power domains.

#### 4.3.1.1.1 Interconnect block power domains

The coherent mesh network interconnect has a single power domain and resides in the top-level AONTOP power domain and the SYSTOP reset domain.

Additionally, the SLC and SF RAMs can be power managed independently of AONTOP by the SCP software as part of static power control. Dynamic power transitions are executed autonomously within each HN-F partition by the CMN-700 hardware. For more information about CMN-700 power domain support, see the *Arm® Neoverse™ CMN-700 Coherent Mesh Network Technical Reference Manual*.

The non-coherent interconnect and network interconnect requirements are in the top-level AONTOP region and the SYSTOP reset domain. The AONTOP region is always active and not power gated. Depending on the SoC implementation, the implementation of the power domain can vary.

#### 4.3.1.1.2 Processor block power domains

Each core is in separate power domain, labeled PD\_CPU*n*.

There are separate, dedicated PPU and clock domains for each core. The cluster power domain, CLUSTOP*n*, and the SYSTOP domain are reset domains.

The PPU and clock control logic generates the clock gating and resets required to put the core in clock gated, reset, or low-power domains. The power gating can be implemented by using on-chip power gates, or by turning off the external or on-chip voltage regulators.

#### 4.3.1.1.3 Interrupt block power domains

The global GIC Distributor logic resides in the top-level AONTOP power domain and the SYSTOP reset domain.

The ITS blocks power domain can be in the same domain as the respective I/O master power domains or the AONTOP power domain. The GIC Redistributors are in their respective cluster or Core power domains.

#### 4.3.1.1.4 I/O Virtualization block power domains

All the SMMU logic resides in the top-level AONTOP power domain and SYSTOP reset domain.

External I/O masters such as the PCIe-RC can be part of AONTOP and SYSTOP. Alternatively, external I/O masters can be part of separate SoC-defined power and reset domains. These choices are outside the scope of the RD-N2.

#### 4.3.1.1.5 MSCP block power domains

All the SCP and MCP logic resides in the AONTOP power domain. There are respective domain bridges on the interfaces for power domain crossing to any other power domains that exist.

### 4.3.1.2 Voltage domains by functional block

The functional blocks in RD-N2 are in various voltage domains.

#### 4.3.1.2.1 Interconnect block voltage domains

All the coherent mesh network interconnect logic is in the VSYS voltage domain.

In RD-N2, the non-coherent interconnect and network interconnect requirements are also in the VSYS voltage domain. However, the non-coherent interconnect and network interconnect are not architecturally limited to any one voltage domain. Depending on the SoC implementation, the implementation of the voltage domain can vary.

#### 4.3.1.2.2 Processor block voltage domains

Each core is in a separate voltage domain, labeled VCPUn. For Direct connect mode, the cluster core bridge logic is in the VSYS voltage domain.

#### 4.3.1.2.3 Interrupt block voltage domains

All the GIC logic is in the VSYS voltage domain. The PPIs and SGIs are in the VCPUn domains.



Note

RD-N2 does not strictly limit the GIC logic to the VSYS and VCPUn domains due to the nature of the distributed architecture. The microarchitecture might choose different domains as a trade-off between performance and ease of implementation.

#### 4.3.1.2.4 I/O Virtualization block voltage domains

All the SMMU logic is in the VSYS voltage domain.



Note

As with GIC, the RD-N2 system architecture does not strictly limit the SMMU logic to the VSYS domain due to the nature of the distributed architecture. The microarchitecture might choose different domains as a trade-off between performance and ease of implementation.

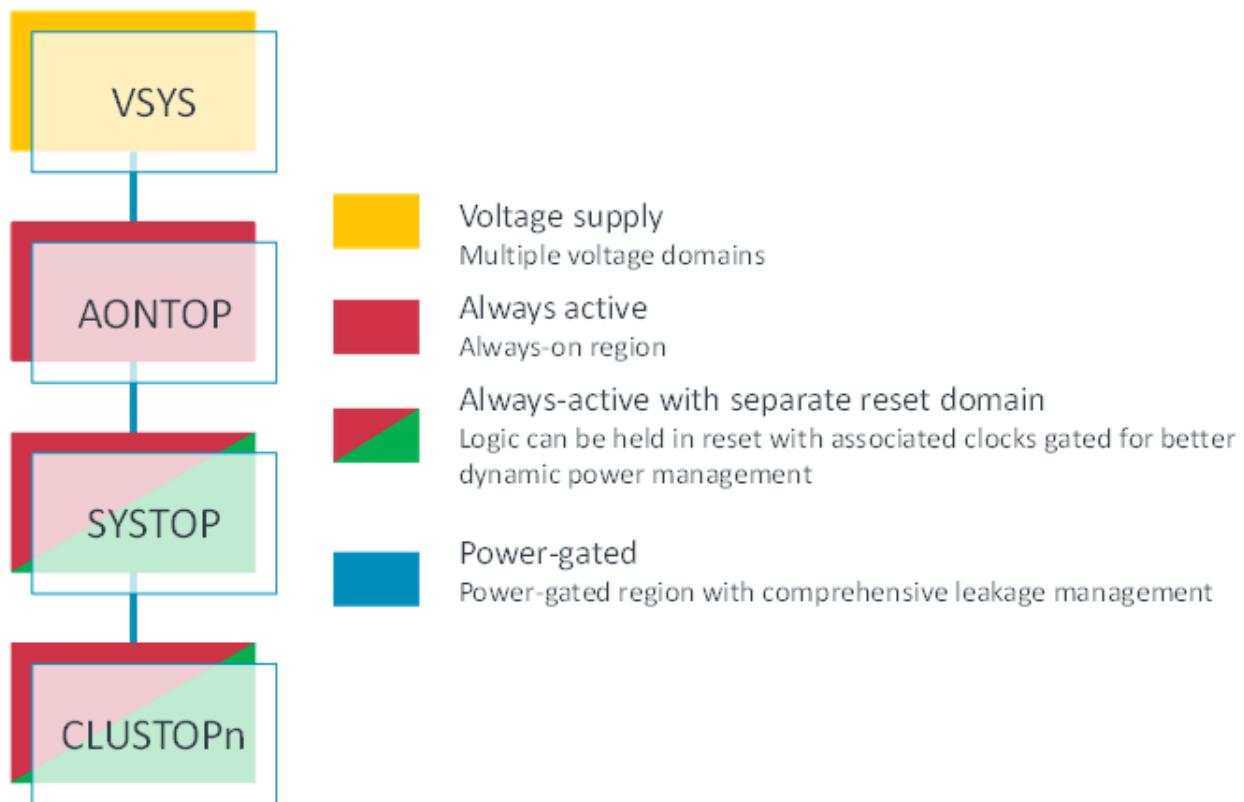
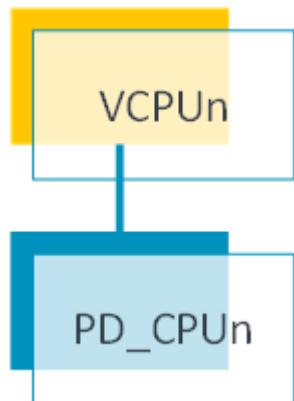
#### 4.3.1.2.5 MSCP block voltage domains

All the SCP and MCP logic is in the VSYS voltage domain.

### 4.3.2 Power and voltage domain hierarchy

A hierarchy of power and voltage domains exists in RD-N2.

The voltage and power domain hierarchies in the following figures are illustrative only of logical relationships between the power-gated domains.

**Figure 4-5: VSYS hierarchy in RD-N2****Figure 4-6: VCPUn hierarchy in RD-N2**

A single PPU or multiple PPUs controls each power domain. PPUs contain a programmer-visible state that the SCP uses to set the power policy and control the power mode of the power domains. For more information about the PPU, see the *Arm® Power Policy Unit Architecture Specification, version 1.1*.

### 4.3.3 Compute subsystem power states

RD-N2 supports various system power and reset states.

The following table shows the system power and reset states that are supported.

**Table 4-10: Compute subsystem power and reset states**

SYSTOP PPU (reset controller)	DBGTOP PPU (reset controller)	Debug chain PPU (reset controller)	PD_CPUn PPU (power controller)	CLUSTOPn PPU (reset controller)	Equivalent CSS power states	Description
WARM_RESET	OFF or WARM_RESET	OFF or WARM_RESET	OFF	OFF	CSS.SLEEP0	System logic is reset while the debug logic is not used, that is, held in reset. If the debug chain PPU is in the WARM_RESET state, the debug PPU must also be in the WARM_RESET state. If the debug chain PPU is in the OFF state, the debug PPU can be in the WARM_RESET or OFF states.
WARM_RESET	ON	Any	OFF	OFF	CSS.SLEEP0	System logic is reset while the debug logic is up. Allows debug of AONTOP (SYSTOP reset domain) through a Warm reset of the system logic PPU.
ON	OFF or WARM_RESET	OFF or WARM_RESET	OFF	OFF	CSS.RUN or CSS.SLEEP0	Debug logic can remain held in reset until it is needed. All clocks generated in the debug element are gated. However, debug-related clocks that are generated by the clocking element (DBGCLK, SYSDBGPCCLK) are running as they are in the AONTOP power domain (SYSTOP reset domain). These clocks are required to clock parts of the MSCP debug logic. If the debug chain PPU is in the WARM_RESET state, the debug PPU can be in the ON or WARM_RESET states. The debug PPU cannot be in an OFF state when the debug chain PPU is in the ON or WARM_RESET states. If the debug chain PPU is in the OFF state, the debug PPU can be in the ON, WARM_RESET, or OFF states.
ON	ON	OFF or WARM_RESET	OFF	OFF	CSS.RUN or CSS.SLEEP0	Debug chains can remain held in reset until needed. Associated clocks (DBGCH<k>ATCLK, DBGCH<k>PCLKDBG) are gated.
ON	ON	ON	OFF	OFF	CSS.RUN or CSS.SLEEP0	Debug chains fully operational.
OFF	OFF	OFF	OFF	OFF	CSS.SLEEP1 (if VSYS is ON) or CSS.OFF (if VSYS is OFF)	If VSYS is ON, power is still provided to the AONTOP power domain.

The following guidelines apply to RD-N2.

- The power states are compute subsystem specific and do not define the state for the SoC logic outside the compute subsystem boundary.

- The CMN-700 SLC memories can be configured independently to be in the ON and OFF states without power gating. Additional power domains are managed internally within CMN-700 by:
  - Software through the Global Programmers View (GPV) space for static transitions.
  - Power management logic hardware within CMN-700 for dynamic transitions.

These CMN-700 internal domains are:

- HN-F SLC RAM0 power domain
- HN-F SLC RAM1 power domain
- SFRAM snoop filter RAM power domain

For more information, see the *Arm® Neoverse™ CMN-700 Coherent Mesh Network Technical Reference Manual*.

Memory retention of SLCRAMx and SFRAM is not supported in this implementation.

The following table shows the power states that each PPU supports.

**Table 4-11: Compute subsystem PPU power states**

Power Policy Unit	Power states supported	Description
System logic PPU	ON OFF WARM_RESET	PPU for the system logic. SYSTOP is not implemented as a power domain, so this PPU acts only as a reset controller.  Static only.  Note that the power state of CMN-700 memories is controlled through software. Software can program CMN-700 registers to change the state of the SLC and SF RAMs when PD_SYSTOP is in the ON state.  A Warm reset resets all the logic in SYSTOP, including all the cores.
CLUSTOPn_PPU	ON OFF WARM_RESET DBG_RECov EMULATED OFF	CLUSTOP domain within the Direct connect DSU, which is made part of the AONTOP power domain (SYSTOP reset domain). The cluster PPU within the Direct connect DSU operates only as a reset controller. The cluster top PPU must be controlled by software. The software must ensure that the CLUSTOP PPU is set to the OFF state before SYSTOP can be changed from the ON state to the OFF state.  For more information about power mode support, see the <i>Arm® Theodul DynamIQ™ Shared Unit Technical Reference Manual</i> .

Power Policy Unit	Power states supported	Description
CPUn PPU	ON OFF WARM_RESET EMULATED OFF DBG_RECov	PPU for the processor core. One per core.

#### 4.3.4 SYSTOP reset control

This domain is an always-active power domain, but with a separate SYSTOP reset domain.

The SYSTOP logic is implemented as a separate reset domain and the system PPU provides all the reset controls. The system PPU implements ON and OFF states, where the OFF state means that the logic is in reset and clocks can be turned off.

The PPU for SYSTOP is implemented in the MSCP Always-on power domain.

The System PPU manages power control of the following logic that resides in this reset domain:

- Interconnect block: CMN, NI, and NIC
- Interrupt block
- I/O Virtualization block
- Dynamic Memory block
- Peripheral block
- Clock Control block

The following power domains are managed internally within CMN-700 by software through the GPV space for static transitions, and by power management logic within CMN-700 for dynamic transitions:

- HN-F SLC RAM0 power domain. SLC tag/data RAMs bits[7:0] within HN-F partitions. The RAMs in each HN-F partition can be independently controlled.
- HN-F SLC RAM1 power domain. SLC tag/data RAMs bits[15:8] within HN-F partitions. The RAMs in each HN-F partition can be independently controlled.
- SFRAM power domain. Power domain for snoop filter RAM in CMN.

The SLC RAM and the SF RAM support the different power states defined in the Arm® Neoverse™ CMN-700 Coherent Mesh Network Technical Reference Manual.

All the top-level AMBA AXI and ACE-Lite interfaces terminate in the AONTOP power region and SYSTOP reset region. To facilitate reset of SYSTOP, components outside the compute

subsystem must handshake with the system logic PPU to quiesce the system components before going into the OFF state. The handshake is performed through an expansion power Q-Channel or P-Channel. For more information, see [Power Q-Channel or P-Channel expansion interfaces \(EXPWRQ\\_DBGTOP<0-n>, EXPWRP\\_DBGTOP<0-n>, EXPWRQ\\_SYSTOP<0-n>, EXPWRP\\_SYSTOP<0-n>\)](#).

For more information about the Q-Channel and P-Channel protocol, see the AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces.

### 4.3.5 PD\_CPUUn power control

PD\_CPUUn is a power domain per core.

The power gating for each core can be implemented with on-chip power gates or by turning off external or on-chip voltage regulators. There is a separate dedicated PPU per core that generates control logic to put the core in low-power state. This change prepares the core for entering the corresponding CPUUn power gating mode. The corresponding core PPU must be in the OFF state before:

- The on-chip power gates are turned off dynamically by hardware
- The on-chip or external voltage regulators are turned off by the SCP core



An implementation can choose not to have power gating per core and group multiple cores into a single power-gated region. The core PPUs in the power-gated region must be in the OFF state before the power is gated. When all the cores in that power-gated region are in the OFF state, the SCP handles the control sequence for turning off the power.

The following table describes legal power policy state combinations for the VCPUUn voltage domains. It also shows the dependencies to relevant power-gated regions under VSYS and VCPUUn in each configuration. The states of all other power-gated regions can be considered orthogonal to the core and cluster power states.

**Table 4-12: Core power policy states**

VSYS	VCPUUn	PD_CPUUn
ON	ON	ON/OFF
ON	OFF	OFF
OFF	OFF	OFF

The ON power policy encompasses RUN and IDLE (STANDBYWFI) device states.

### 4.3.6 SYSTOP domain reset control sequences

The SYSTOP region is a resettable domain.

The system logic PPU is in the MSCP block, resides in the AONTOP domain, and acts as a reset controller. This PPU supports the ON, OFF, and WARM\_RESET states.

The following steps are required for a system logic PPU transition to the OFF state:

1. Reset state preconditions:

- The architectural state is saved before the application processor core goes into the OFF state.
- All the application processor cores and cluster PPUs are in the OFF state.
- The application processor wake up latency requirements must also permit the exit latency from the SYSTOP OFF state.
- It is assumed that all I/O masters that are controlled by application processor cores are quiesced by disabling them.

2. Save state:

- The SCP saves any system IP states that are not already saved.
- Any state in the SRAM must be saved to memory before the memory controllers shut down.

3. Set CMN-700 SLC and SF memories to the OFF state:

- The SCP programs the respective CMN-700 registers to put the SLC and SF memories into the OFF state.

4. Shut down memory controller and self-refresh by the DDR:

- All memory controllers must be placed in an architectural low-power state.

5. Set the system logic PPU to the OFF state:

- The SCP sets to the OFF state the POLICY field in the Power Policy register of the system PPU that controls SYSTOP. The SCP waits for the PPU interrupts to indicate that SYSTOP is entering the OFF state.

6. Stop PLLs related to the SYSTOP logic:

- The SCP stops the PLLs. If a PLL shares external logic that remains active, then it must remain running.

The following steps are required to manage a SYSTOP PPU transition from the OFF state to the ON state:

1. Start system PLLs:

- The SCP restarts any required PLLs, waits for lock interrupts, and switches the clock muxes to PLL clock source, as required.

2. Set the system logic PPU to the ON state:

- The SCP sets to the ON state the POLICY field in the system PPU that controls SYSTOP. Completion of the reset sequence is indicated by a PPU interrupt.

3. Start the memory controllers:
  - The SCP programs the memory controllers, performs the DDR PHY start-up sequences and then configures the memory controller to the active state.
4. Restore the system state:
  - The SCP restores the system IP, SRAM memory state, and memory controller states.

### 4.3.7 PD\_CPUn power domain power control sequences

The following sections describe the power control sequences in the PD\_CPUn power domain.

#### 4.3.7.1 PD\_CPUn power sequence from the OFF to the ON state

The PD\_CPUn power transition from the OFF state to the ON state can be done statically or dynamically.

The following sequence provides a general overview of the tasks that are required to manage the PD\_CPUn power domain transition from the OFF state to the ON state.

In the static transition, the SCP brings the core out of reset on power on reset or whenever SYSTOP transitions from the OFF state to the ON state:

1. The SCP turns on the corresponding VCPU voltage rail through an external PMIC or an integrated voltage regulator.
2. The SCP starts the core PLLs and waits for the respective PLLLOCK interrupt.
3. The SCP core programs the corresponding cluster CLUSTOPn\_PPU, acting as a reset controller, setting the static power mode policy to the ON state.
4. The SCP core programs the corresponding CPUn\_PPU, setting the static power mode policy to the ON state.
5. When the core comes out of reset, it determines the boot type and takes the appropriate action. Note that the way in which the boot type is determined is software **IMPLEMENTATION DEFINED**.



**Note** The preceding sequence programs the cluster before the cores. It is possible to change the order and program the cores before the cluster. However, the cores will not take effect and will not reach the ON power mode state, until after the cluster has reached the ON power mode state.

In a dynamic transition from the OFF state to the ON state, the core wakes up dynamically on various interrupts and events in the system through COREWAKEUPREQUEST<n> or CLUSTERDBGWRUPREQ<n>:

- Core timer interrupts from the timers that reside in the cluster logic. This event is only possible if the cluster power domain is in the ON state. Note that the core timer interrupts wake up the core only when the CLUSTOPn is in the ON state.

- Interrupts from other cores.
- System interrupts, such as application processor timer interrupts and MHU interrupts from the SCP, MCP, and other system peripherals.
- Debug powerup requests (CLUSTERDBGPWRUPREQ<n>).

The transition sequence is as follows:

1. CLUSTOPn\_PPU, if it is in the OFF state, and CPUn\_PPU transition to the ON state on any of the events in the preceding list.
2. The core comes out of reset and restore the architectural state.
3. The coherency domain is entered by asserting **SYSCOREQ** signals.

#### 4.3.7.2 PD\_CPU<sub>n</sub> power sequence from the ON to the OFF state

The PD\_CPU<sub>n</sub> power transition from the ON state to the OFF state can be done statically or dynamically.

In the static transition, the request for static power down to the OFF state can be initiated by:

- The application processor cores or the OS sending a message through the MHU to the SCP core when a task is complete and a transition to a lower power state is required. This message states that PD\_CPU<sub>n</sub> must be powered down and clarifies whether the L2 cache requires flushing using the SCP-controlled L2 cache flush mechanism. The message indicates that the OS is ready to power down, with context saved, at the next STANDBYWFI. When saving context, the core must mask interrupts to the core in the GIC to prevent exit from STANDBYWFI before the power down sequence is complete.
- The SCP sending a message to application processor cores due to system events such as maintaining system Thermal Design Power (TDP), temperature events, and other maintenance events.

The transition sequence is as follows:

1. The SCP core programs the corresponding CPUn\_PPU, setting the static power mode policy to the OFF state.
2. If CLUSTOPn is to be set to the OFF state, the SCP core programs the corresponding CLUSTOPn\_PPU, setting the static power mode policy to the OFF state.
3. When the application processor core is ready to go to a low-power state, it saves the architectural state, flushes the caches, and execute WFI.
4. The PPU starts the transition to the OFF state on receiving the WFI event and sends an interrupt to the SCP when the transition is complete.
5. The SCP can power down the respective core PLLs to save power.

In the dynamic transition, the request for dynamic power down to the OFF state is initiated by a PPU set in dynamic mode and an application processor core executing WFI.

The following steps are required to manage a PD\_CPU<sub>n</sub> power domain dynamic transition from the ON state to the OFF state:

1. The SCP core programs the corresponding CPUn\_PPU, setting the dynamic operating mode policy and the lowest power mode to the OFF state.
2. Before AP core enters low-power mode, software running on the application processor core saves the architectural state and flushes the caches.
3. The application processor core configures the GIC Distributor to disable or reroute interrupts away from the core.
4. The application processor core sends an MHU message to the SCP stating its intent to go to low-power mode and execute a WFI instruction. The message must also indicate whether the cluster CLUSTOPn\_PPU must also be transitioned to the OFF state.
5. The CPUn\_PPU starts the low-power mode transition when it receives the WFI indication. The core is removed from the coherency domain during the power down sequence by deasserting the **SYSCOREQ** signal.
6. The CPUn\_PPU sends an interrupt to the SCP when power down sequence is complete.
7. If CLUSTOPn\_PPU must be set to the OFF state, the transition is initiated statically by the SCP:
  - a. On receiving a message from the application processor core (Step 5), the SCP programs CLUSTOPn\_PPU, setting the static power mode policy to the OFF state.
  - b. When the core is in the OFF state, CLUSTOPn\_PPU starts transitioning to the OFF state.
  - c. The power sequence is completed when the SCP receives an interrupt from CLUSTOPn\_PPU.
8. If the power gating is implemented on-chip, the power is turned off automatically by power gating control logic as part of the PPU transition. Otherwise, if power gating is performed off-chip, the external voltage regulator requires SCP intervention to turn off the power rails.

### 4.3.8 Core power and performance management

The core implements power management features that enable a system to regulate high-activity workloads with the aim of trading off peak performance for improved power efficiency. These features are described in the Supplemental Performance/Power Document that is part of the Arm Neoverse N2 core IP bundle.

The processor features include: Max Power Mitigation Mechanism (MPMM) : Aims to limit the maximum time-averaged power based on a set threshold below the maximum virus workload. Three MPMM parameter sets, called “Gears”, enable the threshold to vary during runtime based on pin control settings. MPMM does not address short-term di/dt demands on the Power Delivery Network (PDN) but can limit the time-averaged power to an expected worst-case real workload.

#### Activity Monitoring Unit (AMU)

A set of AMU counters dedicated to monitoring activity within the processor that are always accessible over the debug APB sideband bus even when the core is powered down. In addition, the AMU can be exposed at various levels of the Exception level 3 software stack based on a boot time configuration. The AMU can be used for software-based power management.

## Performance Defined Power (PDP)

A “smart” autonomous mechanism within the Arm Neoverse N2 core that power gates aspects of the processor microarchitecture based on the running state.

### Event detection and throttling

Targeted to throttle workloads that can operate at significantly higher power levels than typical integer workloads. MPMM is not intended to throttle workloads at or near typical power levels. The events are limited to high-power load-store events and vector unit instructions.

In RD-N2, the Processor block implements PPUs for each core to manage power per core. Pin control is also supported for the cores to control the power states of each individual core independently.

The programming of the CPUPPMCR\_EL3 register drives configuration inputs to the core. SCP power management firmware can program these values during boot and runtime.

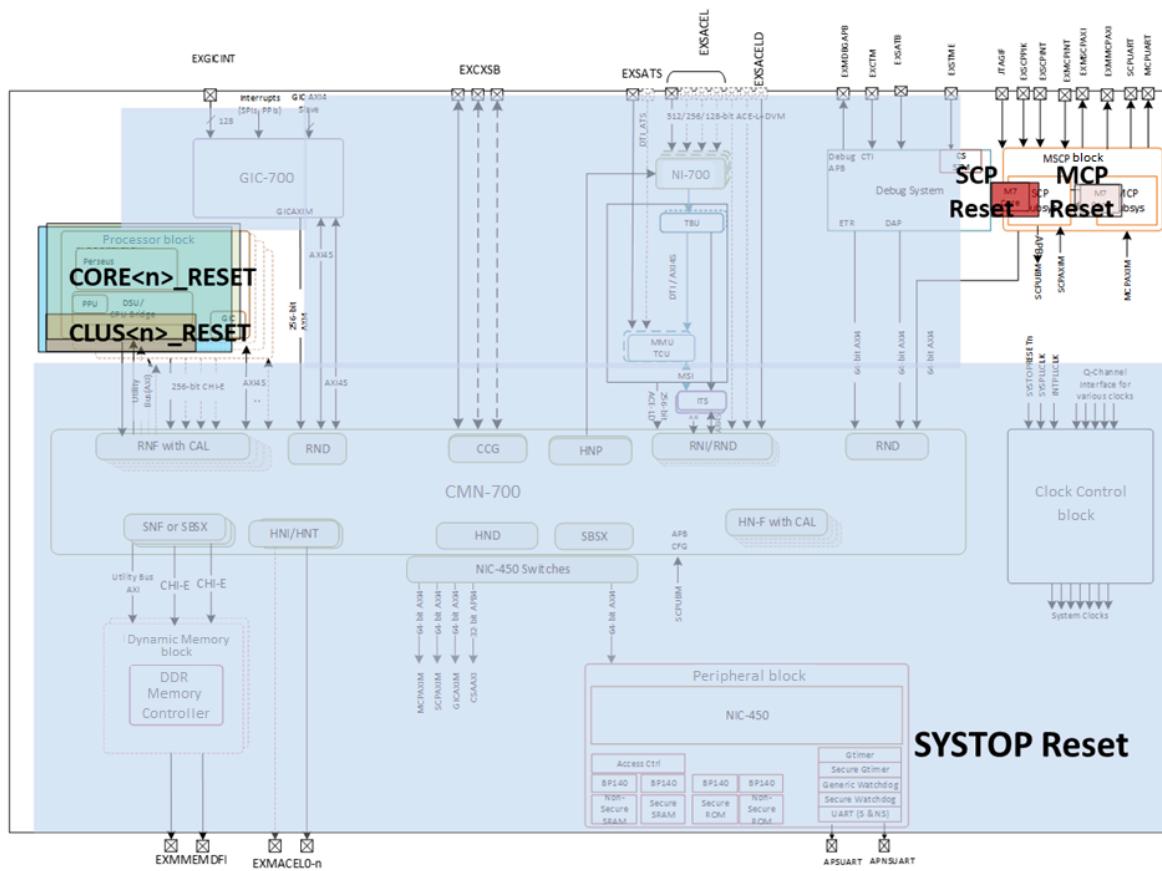
In RD-N2, the following settings are recommended:

- For MPMM:
  - MPMMEN Enabled
  - MPMM Gear 0 (to throttle medium and high-bandwidth vector and viruses)
  - MPMM Gear 1 (to throttle high-bandwidth vector and viruses)
  - MPMM Gear 2 (to throttle power viruses only)
- For PDP:
  - PDP Enabled
  - PDP set points LOW (engages PDP at low aggressiveness)

## 4.4 Resets

This section describes the resets in RD-N2.

The following figure shows the reset domains in the RD-N2 subsystem.

**Figure 4-7: Example compute subsystem system reset domains**

#### 4.4.1 Input resets

RD-N2 has a number of external reset inputs.

External reset inputs:

- **PORESETn** - Power-on reset. Resets the entire compute subsystem.
- **nTRST** - JTAG port reset. Resets the CoreSight DAPs only.
- **nSRST** - System reset. The compute subsystem performs a power-on reset, as if **PORESETn** was asserted, for a defined number of cycles to allow for reset propagation. After this defined number of cycles, the compute subsystem starts to exit the power-on reset condition, but does not release the **SYSRESETn** of the SCP and MCP Cortex-M7 cores. The **SYSRESETn** reset of the Cortex-M7 core is released if **CSYSPWRUPREQ** of the system debug port is asserted. This behavior enables the SCP to handle the request to power up the SYSTOP reset domain. Otherwise, the resets are released when **nSRST** is deasserted.

## 4.4.2 Internal resets

RD-N2 has internal resets such as Warm resets of the SCP and MCP cores, and watchdog resets.

The following table lists the internal reset sources in the MSCP block.

**Table 4-13: MSCP internal reset sources**

Reset type	MCP core	MCP non- core logic	SCP core	SCP non- core logic	SYSTOP logic	CLUSTOPn	PD_CPUn
<b>PORESETn</b>	Y	Y	Y	Y	Y	Y	Y
<b>SCP_CORE_WARM_RESETn</b>	N	N	Y	N	N	N	N
<b>SCP_WATCHDOG_RESETn</b>	Y	Y	Y	Y	Y	Y	Y
<b>SYSTOPRESETn</b>	N	N	N	N	Y	Y	Y
<b>MCP_CORE_WARM_RESETn</b>	Y	N	N	N	N	N	N
<b>MCP_WATCHDOG_RESETn</b>	Y	Y	Y	Y	Y	Y	Y
<b>DBG_TOP_RESETn</b>	N	N	N	N	N	N	N
<b>DBG_CH&lt; n &gt;_RESETn</b>	M	N	N	N	N	N	N



- **PORESETn** is an input to the compute subsystem and not internally generated.
- **SYSTOPRESETn** is the delayed version of the output of the system reset logic (**SYSTOPCLKGENRESETn**). The system clocking logic gets the **SYSTOPCLKGENRESETn**.
- CMN-700 logic needs a delayed version of the **SYSTOPRESETn**.
- All the resets are active-LOW signals.

## 4.4.3 Output resets

RD-N2 provides various reset outputs.

Reset outputs:

### SYSTOPRESETn

An active-LOW reset that must be used by any logic that is attached to an interface within the SYSTOP reset domain.

### SCPAXIEXPRESSTn

An active-LOW reset that is synchronous to SCPAXINICCLK in the SCP block. Use this reset for any logic that connects to the SCP expansion master (EXMSCPAXI) interface.

### MCPAXIEXPRESSTn

An active-LOW reset that is synchronous to MCPAXINICCLK in the MCP block. Use this reset for any logic that connects to the MCP expansion master (EXMMCPAXI) interface.

### IO<x>\_PCle\_RESETn

An active-LOW reset signal that must be synchronized to the respective I/O master clock. Use this reset for any logic that connects to the PCIe Root port controller and related logic.

It is expected that this reset is to be combined (and) with the respective power domain reset. If the PCIe logic is in then SYSTOP reset domain then combine this logic with **SYSTOPRESETn**.

#### 4.4.4 Power policy unit resets

The PPU resets come from the respective PPUs within the design. Software controls the values of these resets by selecting different power policies.

For more information, see the *Arm® Power Policy Unit Architecture Specification, version 1.1*.

A PPU has three discrete resets that allow resets to be applied to different areas of the design next state programmed in the PPU by the SCP firmware.

The three PPU resets are:

- DEVPORESETn - Power-on reset for the power domain
- DEVRETRESETn - Warm reset for any retention flip-flops within the power domain
- DEWWARMRESETn - Warm reset for any non-retention flip-flops within the power domain

Not all the PPU resets are used for every power domain.

#### 4.4.5 SCP core Warm reset

The SCP power control is connected to the PORESETn input for the SoC.

SCP core Warm reset can be used to:

- Reset all the logic in the SCP, and therefore the rest of the subsystem.
- Reset the SCP core only, using either the external reset control signal EXTSCPRESETn or SCP firmware to perform the following sequence:
  1. The SCP core writes a message to the SCP Power Control **SURVIVAL\_RESET\_STATUS** register to capture the type of reset that occurred.
  2. The SCP core writes to the internal SYSRESETREQ register bit in the Cortex-M7 core. See *Arm® Cortex®-M7 Processor Technical Reference Manual*.
  3. The SCP core executes WFI.
  4. The SCP Power Control module detects this idle state and generates a reset to Warm reset the SCP.

#### 4.4.6 MCP core Warm reset

The MCP power control is also connected to the PORESETn input for the SoC.

MCP core Warm reset can be used to:

- Reset all the logic within the MCP, and therefore the rest of the subsystem.
- Reset the MCP core only, using either the external reset control signal EXTMCPRESETn or MCP firmware to perform the following sequence:
  1. The MCP core writes a message to the MCP Power Control [SURVIVAL\\_RESET\\_STATUS](#) register to capture the type of reset that occurred.
  2. The MCP core writes to the internal SYSRESETREQ register bit in the Cortex-M7 core. See [Arm® Cortex®-M7 Processor Technical Reference Manual](#).
  3. The MCP core executes WFI.
  4. The MCP Power Control module detects this idle state and generates a reset to Warm reset the MCP.

#### 4.4.7 Debug through reset

Debug tools might need to access the debug components of the SCP and MCP core before the processor leaves reset.

For example, it might be necessary to set watchpoints or breakpoints that cause the processor to enter debug mode immediately after leaving reset. RD-N2 provides this ability using a dedicated debug system reset input, nSRST.

**PORESETn** must be driven HIGH, when nSRST is being used, to enable debug through reset.

The debug system reset, nSRST, must be driven HIGH during normal operation.

When nSRST is driven from HIGH to LOW, the system performs a reset with the same effect as asserting PORESETn, except that the always-ON logic for DAP is not reset.

While nSRST remains LOW, reset control logic in RD-N2 releases all the resets except for SYSRESETn to the SCP and MCP cores. This allows the debugger to access SCP and MCP debug components.

SYSRESETn will be released when nSRST becomes HIGH. The SCP can then handle the power up requests for other power domains, allowing the reset debug components and system logic to be powered up.

For debugger to access AP core debug components before the core is released from reset it is done by SCP programming the CORE<x> PPU as follows:

- Debugger writes to DP ROM GPR registers requesting to power on PD\_DBGTOP, PD\_SYSTOP power domains.
- SCP core will program the DBGTOP PPU and SYSTEM PPU to power the PD\_DBGTOP and PD\_SYSTOP.
- Debugger writes to DBG\_CH ROM GPR registers requesting to power on corresponding PD\_DBGCH<x> power domains.
- SCP core will program the corresponding DBGCH<x> PPU.

- SCP core will program CLUS<x> PPU to go to static ON state and CORE<x> PPU to dynamic ON state.
- Once the CLUS<x> PPU is ON state, debugger can access cluster debug components. Program the cluster debug registers as required and then write to corresponding cluster ROM table GPR registers requesting to power on the core. The CORE<x> PPU will transition to ON state.

You must program the core PPUs to the OFF\_EMU, WARM\_RST or DBG\_RECov power modes to support debug for various debug functionality. For more information, see the Arm® Neoverse™ N2 Core Technical Reference Manual.

## 4.5 Top-level I/O interfaces

This chapter provides an overview of all the top-level I/O interfaces at a compute subsystem level.

### 4.5.1 I/O coherent expansion slave interfaces (EXSACEL<0-n>, EXSACELD<0-n>)

EXSACEL<0-n> and EXSACELD<0-n> are multiple coherent expansion slave interfaces that support AMBA ACE-Lite with or without Distributed Virtual Memory (DVM) support.

These interfaces have configurable bus widths of 512 bits, 256 bits, or 128 bits. EXSACEL<0-n> and EXSACELD<0-n> are connected either directly, through an asynchronous bridge, or through the non-coherent interconnect network to RN-I and RN-D input interfaces.

I/O coherent masters such as PCIe-RC (AMBA ACE-Lite) can be connected to EXSACEL<0-n> and EXSACELD<0-n>. The transactions on these interfaces are physical addresses. Any address translation is performed in the SoC.

The AMBA ACE-Lite with DVM expansion interface can be connected to external TCUs that require DVM support.

These interfaces are synchronized with INTCLK or EMCLK<n> and are in same AONTOP power domain and SYSTOP reset domains. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

### 4.5.2 I/O coherent expansion slave interfaces with integrated inline TBUs (EXSACELT<0-n>)

EXSACELT<0-n> are multiple I/O coherent expansion slave interfaces that support AMBA ACE-Lite with various configurable of 512 bits, 256 bits, or 128 bits. These interfaces connect to the compute subsystem.

I/O coherent masters such as PCIe-RC (AMBA ACE-Lite) can be connected to EXSACELT<0-n>. The transactions on these interfaces are virtual addresses.

These interfaces are synchronized with EMCLK<n> and are in the same AONTOP power and SYSTOP reset domains. Any power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

#### 4.5.3 Expansion master interfaces (EXMACEL<0-n>, EXMAXI<0-n>)

EXMACEL<0-n> and EXMAXI<0-n> are multiple expansion AMBA ACE-Lite or AXI master interfaces that support 512 bits, 256 bits, or 128 bits. AMBA ACE-Lite or AXI slave peripherals connect to these interfaces.

These ports are supported by using HN-I, HN-D, and HN-T nodes in the CMN-700 interconnect.

The AMBA AXI master interface EXMAXI<0-n> is optional and extends from NIC connecting to HN-D nodes. The intended use case is for small systems when there are only HN-D nodes in the mesh.

PCIe slaves must not be connected to the EXMAXI<n> expansion interfaces.

These interfaces are synchronized with SYSPERCLK and are in the same PD\_SYSTOP power and reset domains. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

#### 4.5.4 DFI master interfaces (EXMMEMDFI<0-n>)

The DDR Physical Interfaces (DFIs) EXMMEMDFI<0-n> are exported when the Dynamic Memory block is integrated in the compute subsystem. These interfaces are connected to the DMC PHY.

DFI 4.0 and DFI 5.0-compliant PHY devices connect to these interfaces.

The EXMMEMDFI<0-n> interfaces are synchronized with DMCCLK and are in the same AONTOP power and SYSTOP reset domains. Any power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.



These interfaces are exported only when the Dynamic Memory block is integrated inside the compute subsystem.

In EXMMEMDFI<0-n>, n includes twice the number of DFI memory interfaces to support a 40-bit dual-channel controllers configuration.

## 4.5.5 GIC expansion interrupts interface (EXGICINT)

The GIC Expansion Interrupts (GEI) interface EXGICINT can accept up to 84 interrupts from devices that are external to the compute subsystem.



**Note** These interrupts can be level or edge triggered. If edge triggered, SoC integrators must synchronize the interrupt to the GICCLK domain and extend the interrupt pulse to be asserted for at least one GICCLK cycle.

This interface is synchronous to the GIC Distributor clock (GICCLK).

## 4.5.6 LTI expansion slave interfaces (EXSLTI<0-n>)

To support LTI/TaaS implementation, a configurable number of EXSLTI<0-n> interfaces are exported from internally instantiated TBUs to connect with external I/O masters that support LTI interfaces.

These interfaces are synchronized with EMLTICLK<n> and are in the same AONTOP power and SYSTOP reset domains. Any power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

## 4.5.7 ATS\_DTI expansion slave interfaces (EXSATS<0-n>)

Configurable numbers of EXSATS<0-n> interfaces are exported from internally instantiated TCUs to connect to external I/O masters that support DTS-ATS interfaces.

These interfaces are synchronized with EMATSCLK<n> and are in the same AONTOP power and SYSTOP reset domains. Any power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

## 4.5.8 MSI expansion slave interfaces (EXSMSI<0-n>)

To support LTI/TaaS implementation, a configurable number of EXSMSI<0-n> interfaces are exported from internally instantiated ITS blocks to connect to PCIe-RCs that support MSI interfaces.

These interfaces are synchronized with EMSICLK<n> and are in the same AONTOP power and SYSTOP reset domains. Any power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

## 4.5.9 STM event expansion interfaces (EXSSTME<0-n>)

The EXSSTME<0-n> interfaces can capture 16 level-sensitive events and 16 edge-sensitive events that occur outside the subsystem.

These interfaces are asynchronous to the STM clock domain and in the AONTOP power domain and the SYSTOP reset domain.

## 4.5.10 CoreSight ATB slave trace expansion interfaces (EXSATB<0-n>)

CoreSight components, such as ELA, that are added outside the compute subsystem during SoC integration use the EXSATB<0-n> interfaces to connect to the external trace source.

In CSATBSE<n>, n is the number of expansion interfaces, which is **IMPLEMENTATION DEFINED**.

These interfaces are synchronized with DBGCLK and, if a separate Debug Top power or reset domain is not implemented, are in the AONTOP power domain. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

## 4.5.11 CoreSight APB master expansion interface (EXMDBGAPB)

The CSAPBME interface enables programming of all the debug components that are added outside of the compute subsystem in the rest of the SoC. This extra debug logic is known as the debug expansion block.

These interfaces are synchronized with SYSDBGCLK and, if a separate Debug Top power or reset domain is not implemented, are in the AONTOP power domain. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

## 4.5.12 CoreSight Cross Trigger Interface (EXCTM)

The EXCTM interface enables connection of triggers from sources in the debug expansion block, and then cross triggers with other devices in the subsystem.

These interfaces are asynchronous. Power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

### 4.5.13 AMBA CXS expansion master and slave interfaces (EXSCXSB<0-n>, EXMCXSB<0-n>)

The EXSCXSB<0-n>, EXMCXSB<0-n> interfaces are used for CCIX and CXL connectivity for the chip-to-chip and accelerator use cases.

These interfaces contain AMBA® CXS Protocol Specification, issue B interfaces to support CCIX 2.0 and CXL 1.1, and contain AMBA® CXS Protocol Specification, issue A interfaces to support CCIX 1.1.

A configurable number of AMBA® CXS Protocol Specification, issue A and AMBA® CXS Protocol Specification, issue B master and slave interfaces can be used. For example, a configuration for CCIX 2.0, CCIX 1.x, and CXL protocol link connectivity using various different transport technologies such as PCIe and XSR or USR might include:

- AMBA® CXS Protocol Specification, issue B (CXS\_B), AMBA CXS TX (PCIe\_CXSB\_TX), and AMBA CXS RX (PCIe\_CXSB\_RX) to support CCIX 2.0 or CXL over PCIe.
- AMBA® CXS Protocol Specification, issue B (CXS\_B), AMBA CXS TX (XSR\_CXSB\_TX), and AMBA CXS RX (XSR\_CXSB\_RX) to support CCIX2.0 or CXL over a die-to-die interface like XSR or USR.
- AMBA® CXS Protocol Specification, issue A (CXS\_A), AMBA CXS TX (PCIe\_CXSA\_TX), and AMBA CXS RX (PCIe\_CXSA\_RX) to support CCIX1.x over PCIe.

The EXSCXSB<0-n> interfaces are exported out of the compute subsystem to connect to CCIX and CXL transport parts of the SoC system.

These interfaces are synchronized with INTCLK and are in the AONTOP power and SYSTOP reset domains. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

### 4.5.14 Power Q-Channel or P-Channel expansion interfaces (EXPWRQ\_DBGTOP<0-n>, EXPWRP\_DBGTOP<0-n>, EXPWRQ\_SYSTOP<0-n>, EXPWRP\_SYSTOP<0-n>)

These interfaces are used to perform handshakes between external components and the system logic PPU.

All the top-level interfaces terminate in the AONTOP power and SYSTOP reset region. When the SYSTOP domain is reset, components outside the subsystem must handshake with the system logic PPU to quiesce the system components before going into the OFF state. The handshake is performed through an expansion power Q-Channel or P-Channel. The number of Q-Channels or P-Channels is **IMPLEMENTATION DEFINED**. These interfaces are asynchronous and are in the AONTOP power and SYSTOP reset domains.

If a separate Debug Top power domain is implemented, a similar arrangement is required. In this case, all top-level interfaces that terminate at the Debug Top power domain within the compute subsystem must implement the expansion power Q-Channels or P-Channels. These channels are required to facilitate power gating of the Debug Top power domain.

### 4.5.15 SCP expansion interrupts (EXSCPINT)

The EXSCPINT interface (32 interrupts) enables the addition of interrupt sources from devices that are external to the compute subsystem. The subsystem routes the interrupts to the interrupt controller in the SCP Cortex-M7 processor.



Note

These interrupts can be level or edge triggered. If edge triggered, SoC integrators must synchronize the interrupt to the SCPCORECLK domain and extend the interrupt pulse to be asserted for at one SCPCORECLK cycle.

---

This interface is assumed to be synchronous to SCPCORECLK and is in the AONTOP power domain.

### 4.5.16 MCP expansion interrupts (EXMCPINT)

The EXMCPINT interface (32 interrupts) enables the addition of interrupt sources from devices that are external to the compute subsystem. The subsystem routes the interrupts to the interrupt controller in the MCP Cortex-M7 processor.



Note

These interrupts can be level or edge triggered. If edge triggered, SoC integrators must synchronize the interrupt to the MCPCORECLK domain and extend the interrupt pulse to be asserted for at least one MCPCORECLK cycle.

---

This interface is assumed to be synchronous to MCPCORECLK and is in the AONTOP power domain.

### 4.5.17 SCP PIK expansion interfaces (EXSCPPIK<0-n>)

The EXSCPPIK<0-n> interfaces are provided to connect to the slave interfaces of modules added in the SoC.

These interfaces connect to the following modules:

- Power management control logic.
- Any other peripheral slave interfaces that are added to the SoC outside the compute subsystem and that are controlled by the SCP. SoC integration determines the number of interfaces to be supported.

The interface type is **IMPLEMENTATION DEFINED**.

These interfaces are synchronized with SCPPIKCLK and are in the same AONTOP power and reset domains. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

### 4.5.18 SCP AON expansion interface (EXMSPAXI)

The EXMSPAXI interface enables the addition of AMBA AXI slave peripherals to the SCP.

The peripherals can be either internal or external to the compute subsystem. The most common slave peripherals are clock, reset, power management, and debug-related functionalities.

This interface is synchronized with SCPAXICLK and is in the same AONTOP power and reset domains. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

### 4.5.19 MCP AON expansion interface (EXMMCPAXI)

The EXMSPAXI interface enables the addition of AMBA AXI slave peripherals to the MCP.

The peripherals can be either internal or external to the compute subsystem. The MCP expansion interface is used to connect peripheral devices such as SPI and I2C.

This interface is synchronized with MCPAXICLK and is in the same AONTOP power and reset domains. Any clock crossing and power domain bridges must be added in the SoC area as necessary and are **IMPLEMENTATION DEFINED**.

### 4.5.20 AP, SCP, and MCP UART interfaces (APNSUART, APSUART, SCPUART, MCPUART)

The UART serial interfaces from various UARTs are exported to connect to I/O pins in the SoC.

### 4.5.21 JTAG and SWD interface (JTAGIF)

The combined JTAG and SWD interface JTAGIF provides connectivity to an external debugger.

The signals for this interface are shown in the following table.

**Table 4-14: JTAGIF interface signals**

Signal	Direction	Description
TDI	Input	Debug data in
TDO	Output	Debug data out
nTRST	Output	JTAG reset
SWDITMS	Input	Combined serial wire data in and debug mode select
SWDO	Output	Serial data out
SWDOEN	Input	Serial data out enable



The SWO is muxed on to TDO in debug mode.

Note

The signals **SWDITMS**, **SWDO** and **SWDOEN** are expected to be used to form a bidirectional **SWDIOTMS** external signal, with **SWDOEN** determining the direction.

This interface is synchronized with SWTCKCLK and is in the AONTOP power domain.

#### 4.5.22 Debug authentication signals

Various debug authentication input interfaces are provided to enable debug for various debug components in the system.

Different sets of authentication input signals provide debug authentication for each domain:

- System debug domain components – **SYSDBGGEN**, **SYSNIDEN**, **SYSSPIDEN**, and **SYSSPNIDEN**
- SCP core debug domain components – **SCPDBGGEN**, **SCPNIDEN**, **SCPSPIDEN**, and **SCPSPNIDEN**
- MCP core debug domain components – **MCPDBGGEN**, **MCPNIDEN**, **MCPSPIDEN**, and **MCPSPNIDEN**
- Application processor core debug components that are part of all the debug domain chains – **APPDBGGEN**, **APPSPIDEN**



- DBGEN - Invasive Debug Enable
- NIDEN - Non-Invasive Debug Enable
- SPIDEN - Secure Privilege Invasive Debug Enable
- SPNIDEN - Non-Secure Privilege Invasive Debug Enable

#### 4.5.23 Functional override inputs

Various input signals are provided to turn off some of the functionality within the compute subsystem.

The following table lists these input signals.

**Table 4-15: Override inputs**

Signal	Direction	Description
ELADISABLE	Input	<p>Connect to <b>ELADISABLE</b> signals on all the application processor cores.</p> <p>Connect to the TCU <b>ELADISABLE</b> signal.</p> <p>When this input is HIGH, access to ELAs is disabled.</p> <p>This input must be connected to a fuse bit or driven by a register bit set by a Secure entity in the SoC. This choice is SoC <b>IMPLEMENTATION DEFINED</b>.</p>
CRYPTODISABLE	Input	<p>Connect to <b>CRYPTODISABLE</b> signals on all the application processor cores.</p> <p>When this input is HIGH, cryptographic functionality in the application processor cores is disabled.</p> <p>This input must be connected to a fuse bit or driven by a register bit set by a Secure entity in the SoC. This choice is SoC <b>IMPLEMENTATION DEFINED</b>.</p>
TCU_SECURE_DISABLE	Input	<p>Connect to <b>sec_override</b> signals on all the TCUs.</p> <p>When this input is HIGH, certain registers are accessible to Non-secure accesses from reset.</p> <p>This input must be connected to a fuse bit or driven by a register bit set by a Secure entity in the SoC. This choice is SoC <b>IMPLEMENTATION DEFINED</b>.</p>

## 4.5.24 Miscellaneous output signals

There are other miscellaneous output signals from the compute subsystem.

The following table lists these output signals.

**Table 4-16: Miscellaneous output signals**

Signal	Direction	Description
ELASTOPCLOCK_OUT	Output	<p>The <b>ELASTOPCLK</b> signal from all the ELAs in the compute subsystem are ORed and sent as an output to the SoC.</p> <p>This output can be used by DFT logic to prevent all the PLLs in the system from being able to scan dump the state of the system. The implementation of the DFT logic is SoC <b>IMPLEMENTATION DEFINED</b>.</p>

## 4.6 System Boot

RD-N2 supports the Secure board boot.

To support Secure boot, the compute subsystem provides:

- A Cortex-M7 based SCP and MCP designed to function as a Trusted block. This block includes:
  - A local on-chip (Trusted) boot ROM.
  - A local on-chip Secure SRAM to execute the main SCP and MCP firmware loaded in from external non-volatile memory.

- For application processor core Secure boot, support for the following features:
  - On-chip Secure boot ROM intended for storing Trusted boot code.
  - On-chip Non-secure ROM intended for storing Non-secure boot code.
  - On-chip Secure SRAM intended for storing Trusted data.

#### 4.6.1 Boot flow overview

This section provides an overview of an example compute subsystem boot flow.

To boot-up, the SCP and MCP assumption is that the compute subsystem must be supplied with voltage and input clocks before the **PORESETn** reset is deasserted:

1. Both the SCP and the MCP come out of reset and start booting from their respective boot ROMs.
2. The SCP optionally copies the flash image from external flash to scratch RAM, which could be part of the SoC.
3. The SCP programs SYSPLL.
4. The SCP and MCP wait until SYSPLL is locked.
  - a. The SCP waits until the SYSPLL lock interrupt is asserted.
  - b. The SCP sends an MHU message response to the MCP to indicate that the SYSPLL is locked.
5. If the SCP and MCP firmware is going to authenticate the image:
  - a. Both the SCP and MCP load external images from SoC RAM or off-chip flash into their respective Tightly Coupled Instruction RAMs (ITCRAMs).
  - b. Optionally, the SCP and MCP authenticate the respective images.
6. The SCP and MCP jump to the respective ITCRAMs.
7. As part of the boot image, the SCP sends a mailbox command through the MHU to the MCP and stalls.
8. The MCP responds to the mailbox command from the SCP.
9. Both the SCP and the MCP establish security trust by exchanging keys, which is all handled in firmware.
10. The SCP stalls when trust is established and waits for the MCP to send a further message to bring the rest of system out of reset.
11. The SCP enables all the application processor PLLs.
12. The SCP powers up PD\_SYSTOP.
13. The CMN-700 interconnect and the memory controllers are initialized, and the PHYs are trained.
14. The SCP powers up one of the cores in the cluster, such as CPU0.
15. The powered up core starts booting from the Secure ROM in the AONTOP domain. The Secure ROM contains Trusted Firmware BL1.

16. If the application processor is performing the authentication, as part of the BL1 image, the application processor copies the image from flash to the compute subsystem Secure scratch RAM. The application processor then authenticates the image.



The software architecture defines the rest of the software boot flow steps.

Note

## 4.7 Security

RD-N2 supports multiple security features to enable system-level security.

Security features are implemented by application processors, GIC, CMN, non-coherent interconnects, SCP and MCP cores, and the associated CoreSight components. RD-N2 has the following key hardware requirements for Trusted boot, Trusted OS, and Trusted applications:

- Trusted cores
- GIC, an interrupt controller that supports Secure and Non-secure interrupts
- SMMU, a system memory management unit that supports translation tables for Secure and Non-secure processes
- CMN-700 and all non-coherent interconnects (NI, NIC), which support the transfer of the Secure and Non-secure (NS) flag that defines the security settings of accesses through the network
- Support for the debug enable control signals **DBGEN**, **NIDEN**, **SPIDEN**, and **SPNIDEN** to permit Secure debug



Cores that implement the Armv9.0-A debug architecture do not have **NIDEN** or **SPNIDEN** inputs.

Note

- Trusted on-chip boot ROM, RAM, and Trusted off-chip DRAM
- The SCP and MCP blocks, including their internal RAM and ROM memories, are treated as inherently Trusted and Secure
  - The SCP and MCP boot from private on-chip Trusted ROM so that boot code cannot be modified
  - Any access from the SCP to the rest of the system is also Trusted and Secure

## 4.7.1 Application processor security

The application processors in the compute subsystem support the Armv8.4 and Armv9.0 architectures, including Security Extensions.

Support for the Cryptographic Extensions is a configurable option. The selected option applies to all cores in the system. For example, the Cryptographic Extensions are optional in the RD-N2 cores.

The **CRYPTODISABLE** input signal to the core or cluster is exposed as a top-level subsystem input to allow the SoC to control availability of a selected Security Extension.



Export control laws can require disabling of high-performance encryption capabilities such as these Security Extensions.

**Note** The presence of the Armv9-A Cryptographic Extensions in an implementation is subject to export license controls. The implementation of Scalable Vector Extensions and Cryptographic Extensions are subject to export license controls.

## 4.7.2 Interconnect block security

CMN-700 and the non-coherent interconnects in RD-N2 support the transfer of the NS flag from all master interfaces to slave interfaces.

The NS flag is transferred through either:

- The request flit on the AMBA CHI interfaces
- The AxPROT[1] bit on the AMBA ACE-Lite, AXI5, AXI4, and AXI3 interfaces
- PPROT on the APB interfaces

These values define the security setting of an access.

The REQFLIT.NS field or AxPROT[1] specifies whether a read or write transaction is Secure or Non-secure.

Within RD-N2, peripheral accesses are placed into the Secure or Non-secure world by either:

- The peripheral itself, interpreting the NS flag
- Slave ports in NIC and NI, either permanently as Secure or Non-secure, or software-configurable using the respective programmers model registers

## 4.7.3 Peripheral block security

The NS flag from all master to slave interfaces through the AxPROT[1] bit on the AMBA ACE-Lite, AXI4, and AXI3 interfaces is connected to various peripherals. These peripherals define the security setting of an access.

## 4.7.4 SCP block security

The SCP core is treated as a Secure core. All accesses from the SCP core are assumed to be Secure.

Any Secure data that the SCP stores, such as Secure state save and restore data, must be stored in the SCP internal private SRAM or Secure on-chip SRAM. The SCP private SRAM is not accessible from outside the SCP.

In the SCP, the Cortex-M7 core and the associated debug logic do not directly support TrustZone. The core only implements the **DBGEN** and **NIDEN** inputs. However, the SCP is inherently a Trusted block and therefore the **DBGEN** and **NIDEN** inputs are driven as follows:

- **SPIDEN** drives **DBGEN** in the SCP
- **SPNIDEN** drives **NIDEN** in the SCP

## 4.7.5 MCP block security

The MCP core is treated as a non-Secure core. All the accesses from MCP core are non-Secure.

In the MCP, the Cortex-M7 core and the associated debug logic do not directly support TrustZone. The core only implements the **DBGEN** and **NIDEN** inputs. However, the MCP is inherently a Non-secure block and therefore the **DBGEN** and **NIDEN** inputs are driven as follows:

- **DBGEN** drives **DBGEN** in the MCP
- **NIDEN** drives **NIDEN** in the MCP

## 4.8 Multichip architecture

RD-N2 supports chip-to-chip cache-coherent interfaces for multichip use cases.

The following chip-to-chip protocols are supported:

- The CCIX 2.0 protocol is used for connecting to compliant device accelerators and memory devices, or for the host-to-host Symmetric Multiprocessing (SMP) use case.
- The CXL 1.1 protocol is used for connecting to accelerators and memory devices that are compliant with these protocols.

In both cases, the transport interface between the CMN-700 interconnect and the physical interface is an AMBA® CXS Protocol Specification, issue B interface.

Memory coherency is maintained between all interconnected chips. All cache-coherent and remote memory traffic between the chips uses these protocols. The physical transport can be PCIe or other low-latency die-to-die transports, such as XSR or USR, and HBM technologies.

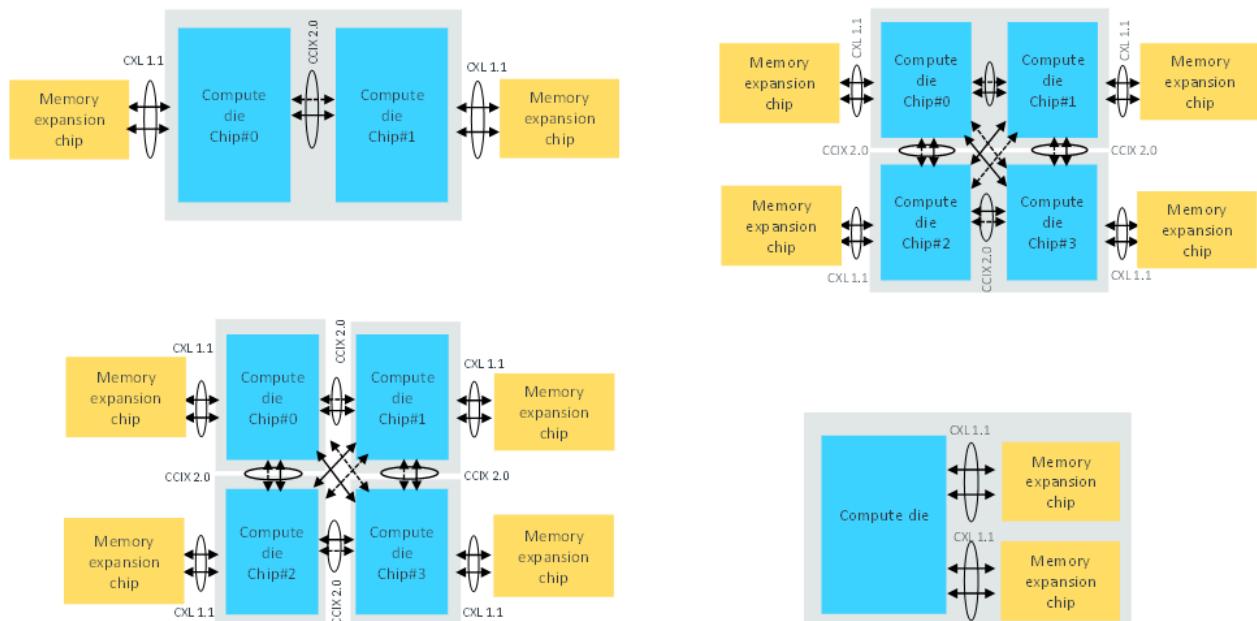
RD-N2 supports the following topologies:

## Host-to-host SMP multichip connections

In this use case, host-to-host connections are used for SMP in a multichip environment. Here, SMP does not mean that all cores in the system are identical. Instead, in this context, SMP indicates that a single hypervisor or operating system is running across the cores in the multichip system. RD-N2 supports up to four chiplets.

The following figure shows examples for die-to-die in-package or off-package multichip topologies. The CCIX 2.0 protocol is used over a 64-byte flit interface or over implementation-dependent die-to-die interfaces.

**Figure 4-8: Die-to-die in-package and off-package multichip connectivity**

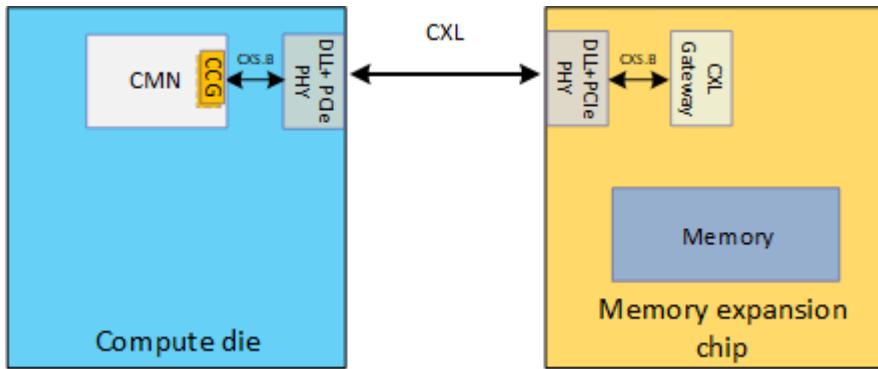


## Host-to-accelerator connections

This use case enables hardware-coherent caches in accelerators and maintains a consistent view of shared data for both processor and accelerator accesses without explicit intervention from software. Caching allows implicit movement of shared data in a system for further reuse or modification, based on the shared access patterns of the processor or accelerator. This approach avoids the overhead of software migrating or maintaining multiple or modified copies between them.

## Host-to-memory expansion connections

In this use case, the CCIX or CXL connections are used to add further memory capacity to a system. As a result, expansion of the system memory domain beyond the host-attached memory is enabled. The host memory manager can optionally allocate and manage peripheral-attached memory in the same manner that host memory is allocated and managed, as part of system memory.

**Figure 4-9: Expansion memory chip connected over optimized CXL PCIe PHY**

## 4.8.1 Host-to-host SMP multichip use case

RD-N2 supports up to 4 chiplets.

### 4.8.1.1 Identification of each chip

A single-chip or multichip system is identified by the SID\_CHIP\_ID[MULTI\_CHIP\_MODE] register bit field. Each chip has a tie off to indicate whether the system is single chip or multichip, and the chip ID for each chip.

This information is captured in the SID\_CHIP\_ID[CHIP\_ID] register bit field. All the cores in the system have access to the SID\_CHIP\_ID register. The chip with SID\_CHIP\_ID[CHIP\_ID] set to 0 is the master and all the other chips are slaves.

### 4.8.1.2 Address mapping for multichip systems

RD-N2 supports separate Non-Uniform Memory Access (NUMA) regions for each chip where the group of HN-Fs on each chip form a coherent subdomain.

Each chip is allocated a different memory region. The CMN-700 interconnect on each chip is programmed with its own RN SAM at boot time.

Each chip is assigned a chunk of memory space. If all the chips are identical, the chips have same memory map regions with different base addresses.

For a quad-chip example, the following memory regions are assigned to each chip:

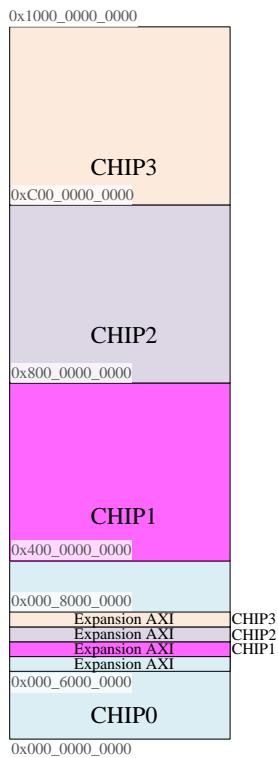
- Chip 0: 0x000\_0000\_0000 – 0x3FF\_FFFF\_FFFF
- Chip 1: 0x400\_0000\_0000 – 0x7FF\_FFFF\_FFFF
- Chip 2: 0x800\_0000\_0000 – 0xCFF\_FFFF\_FFFF
- Chip 3: 0xC00\_0000\_0000 – 0xFFFF\_FFFF\_FFFF



The only exception involves the lower 4GB space, which is a part of the application processor memory expansion region. Some portion of this region can be carved out for allocation to external PCIe devices non-prefetchable MMIO space connected to slave chips. This example shows the memory region split into 4TB chunks for each chip. Depending on the number of chips and the system requirements, the memory can be split into different blocks.

The following figure shows the memory regions allocated to each chip for this example.

**Figure 4-10: Memory regions allocated to each chip in a quad-chip example**



The expansion AMBA AXI mapping shown in the lower 4GB of the preceding figure is just an example.

### 4.8.1.3 Data communication between chips

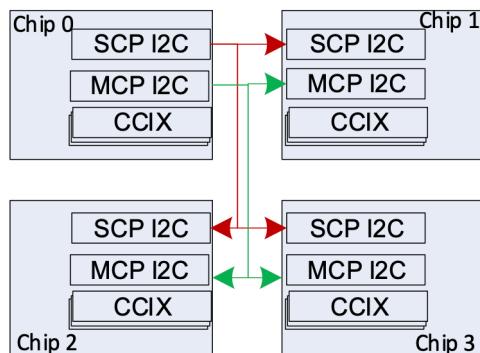
The main data interface between the chips is provided by CCIX links for coherent and non-coherent data communication.

The data flow between the chips over the CCIX links includes:

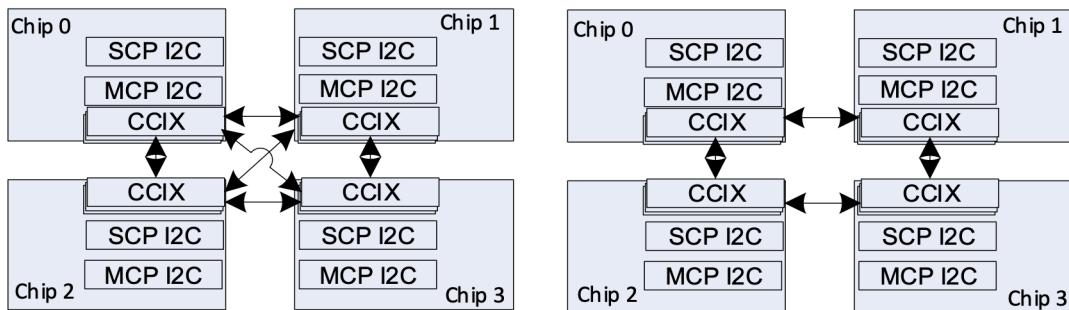
- Memory traffic to the remote memory
- Cache-coherent traffic between chips
- SMMU DVM (translation table invalidation requests) broadcast to remote chips
- Interrupt messages communicated between chips

Before the CCIX links are enabled, serial interface communication must be supported between SCPs and MCPs on master and slave chips for synchronization and exchange of configuration information. To address this requirement, the compute subsystem provides SCP and MCP expansion interfaces to add support for serial interfaces such as I2C at the SoC level. The following figure shows I2C serial interface connections between multiple chips. This interface is used by the SCP and MCP to communicate with SCPs and MCPs in other chips before the CCIX interface is enabled. The I2C interface in the master chip is configured in master mode and all the chips are in slave mode.

**Figure 4-11: I2C serial interface between SCPs and MCPs on master and slave chips**



The following figure shows CCIX connections between multiple chips. The chips can be fully connected or sparsely connected. If they are sparsely connected, CCIX CMN-700 supports CCIX port forwarding, which forwards the transactions from one chip to another when they are connected directly.

**Figure 4-12: Fully and sparsely connected CCIX connections**

#### 4.8.1.4 Interrupt routing between the chips

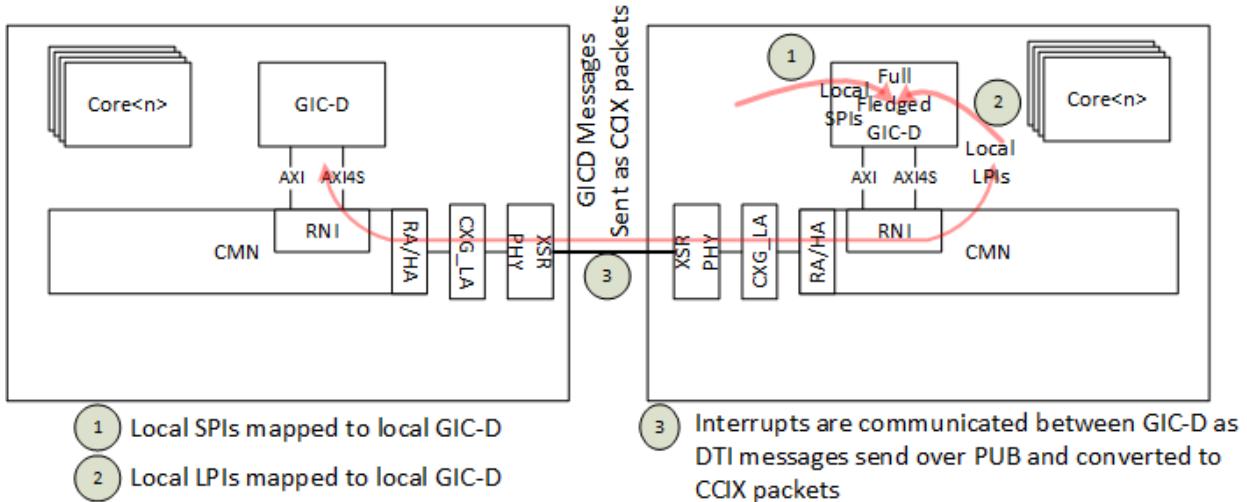
Each chip has an individual distributed GIC-D mapped at a different physical address map in their own local memory map region.

From a software perspective, there is single physical address for GIC-D for the whole system. There is a single global GIC-D instance at the same base address (master chip address region) for all chips. CMN-700 RN SAM address mapping is programmed to map this GIC-D address space to its local GIC-D.

The local GIC-D on each chip services all interrupts. If the interrupt is mapped to a remote core, it routes the interrupt information as a message to the remote GIC-D to interrupt the respective core in that chip.

Interrupt routing from one chip to another is performed by sending messages between the chips over a CCIX link interface. The GIC messages from GIC-D to CCIX are routed over the CMN-700 PUB interface. CMN-700 packetizes the GIC messages and sends them to other chips over the CCIX interface. The following figure shows the interrupt routing for a dual-chip use case.

**Figure 4-13: Interrupt routing in multichip scenarios: in-package accelerator attached through XSR**



#### 4.8.1.5 Generic timer synchronization between the chips

The system generic count value is gray coded and routed to each core. The value is balanced so that each application core gets the same generic time count.

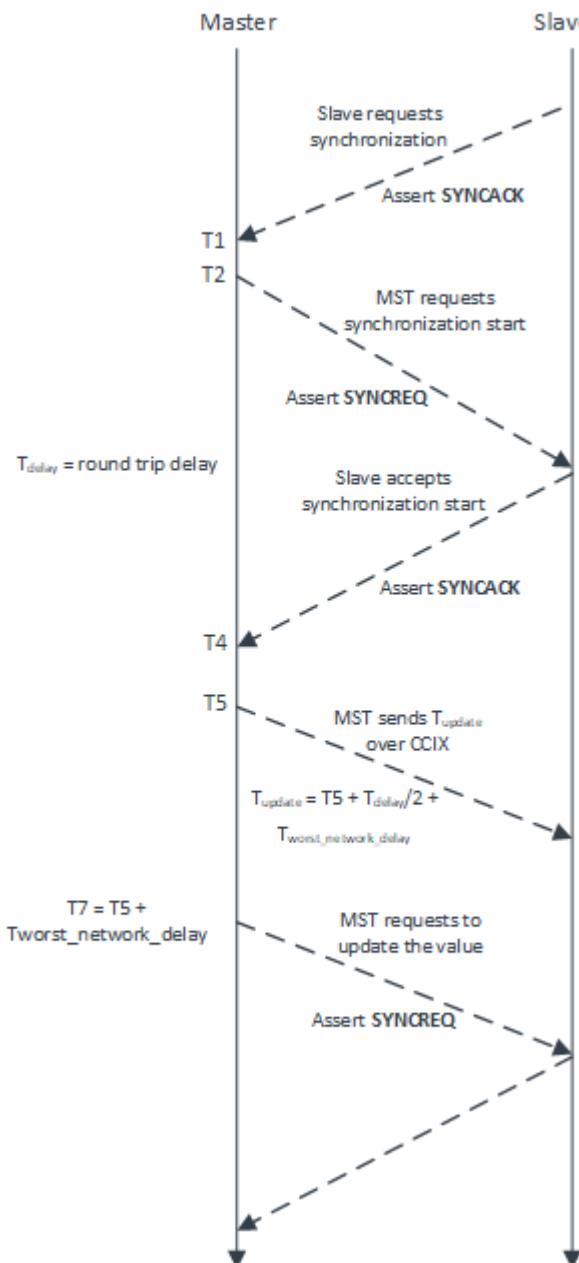
It is difficult to balance these count values because all cores across the multichip can see the same value. So, each chip or platform has its local generic counter. The master socket has a master generic time count. Slave sockets synchronize their local generic time count with the master generic time count. This process is performed by using a dedicated two-pin interface (**SYNCREQ** and **SYNCACK**) between each master and slave pair, and sending messages over CCIX.

Messages are sent as device writes over CCIX to memory-mapped registers when:

- The master sends future  $T_{\text{update}}$  time values to the slave
- The slave uses the interface to reset the master state machine for any error conditions

**SYNCREQ/SYNCACK** four-phase handshake signals are used when:

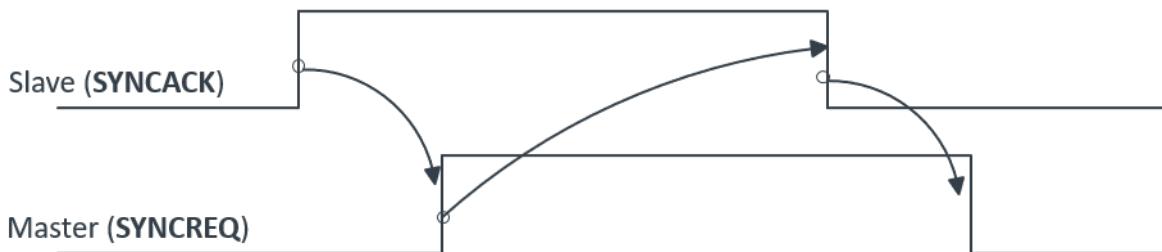
- The slave initiates a request to the master to start the synchronization process
- The master initiates the start of the synchronization process.
- The master indicates an update of the slave counter with the value that was sent previously.
- After the slave socket comes out of reset, it synchronizes with the master socket to initialize the local REGCLK time. After the local REGCLK time is initialized, the slave socket synchronizes with the master socket at regular intervals that are programmed by software.

**Figure 4-14: Synchronization process between master and slave**

Synchronization process between master and slave:

1. The slave initiates synchronization at T0:
  - The slave asserts **SYNCACK** to indicate to the master to start synchronization.
  - The master acknowledges by asserting **SYNCREQ**.

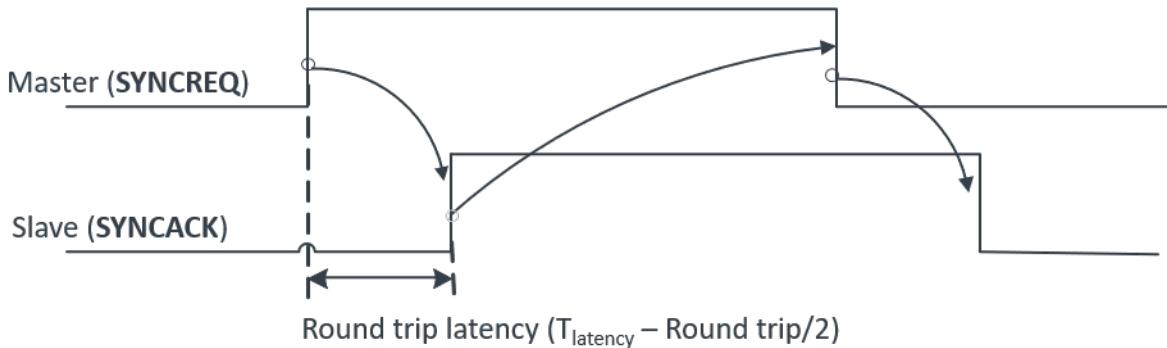
**Figure 4-15: Synchronization process at T0**



2. The master initiates synchronization by asserting **SYNCREQ** @T2. The slave accepts the synchronization by asserting the **SYNCACK** @T3.

Delay value on the dedicated pins between master and slave:  $T_{\text{delay}} = (T4 - T2)/2$

**Figure 4-16: Synchronization process at T2**



3. The master sends a device write with  $T_{\text{updvalue}}$  @T5 over CCIX. The slave receives @T6.

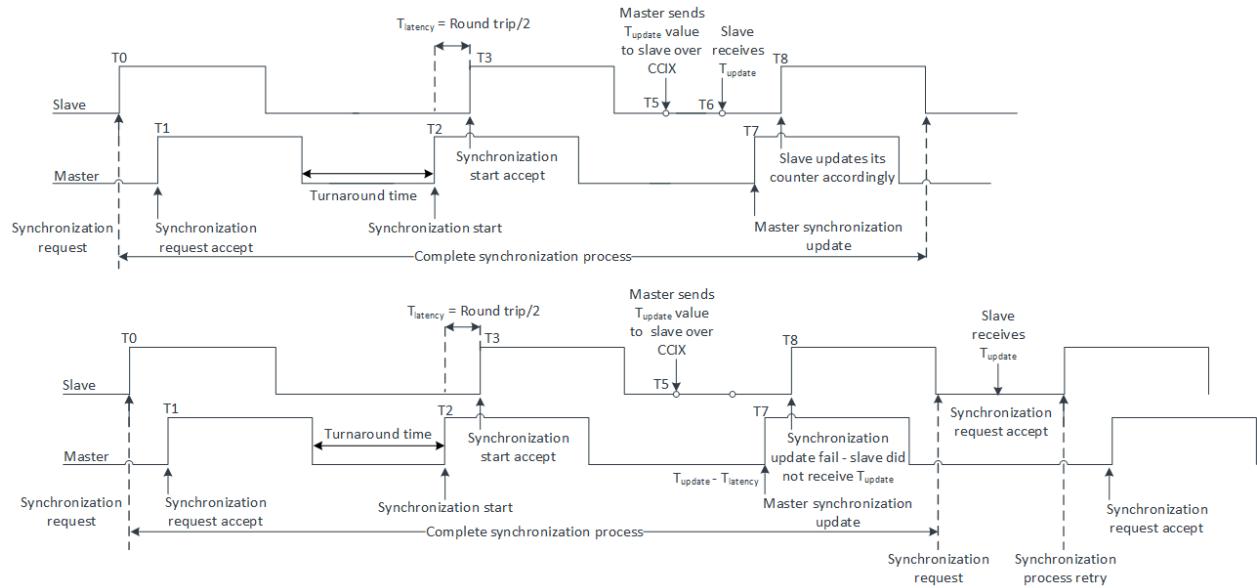
Future time value:  $T_{\text{updvalue}} = T5 + X$ , where  $X = T_{\text{delay}} + T_{\text{ccix\_network\_delay}}$

4. Master @ $T7 = T_{\text{updvalue}} - T_{\text{delay}}$  initiates the synchronization update to the slave by asserting **SYNCREQ**. The slave acknowledges the synchronization update @T8 by asserting **SYNCACK**.
  - If  $T_{\text{updtslave}} < T_{\text{slave}}$  @T8, raise an interrupt to software. Alternatively, stall the slave counter for  $(T_{\text{updtslave}} - T8)$  cycles, then update the counter with  $T_{\text{updtslave}}$  and continue.
  - If  $|T_{\text{updtslave}} - T_{\text{slave}}| > \text{OFFSET\_THRESHOLD}$  raise an interrupt to the SCP.
  - Otherwise, the slave updates its local time with the previously received  $T_{\text{updvalue}}$ .
  - If at T8 the slave does not receive a  $T_{\text{update}}$  value from the master, the slave waits until receiving the value. The slave initiates another synchronization immediately and ignores the received  $T_{\text{update}}$  value.

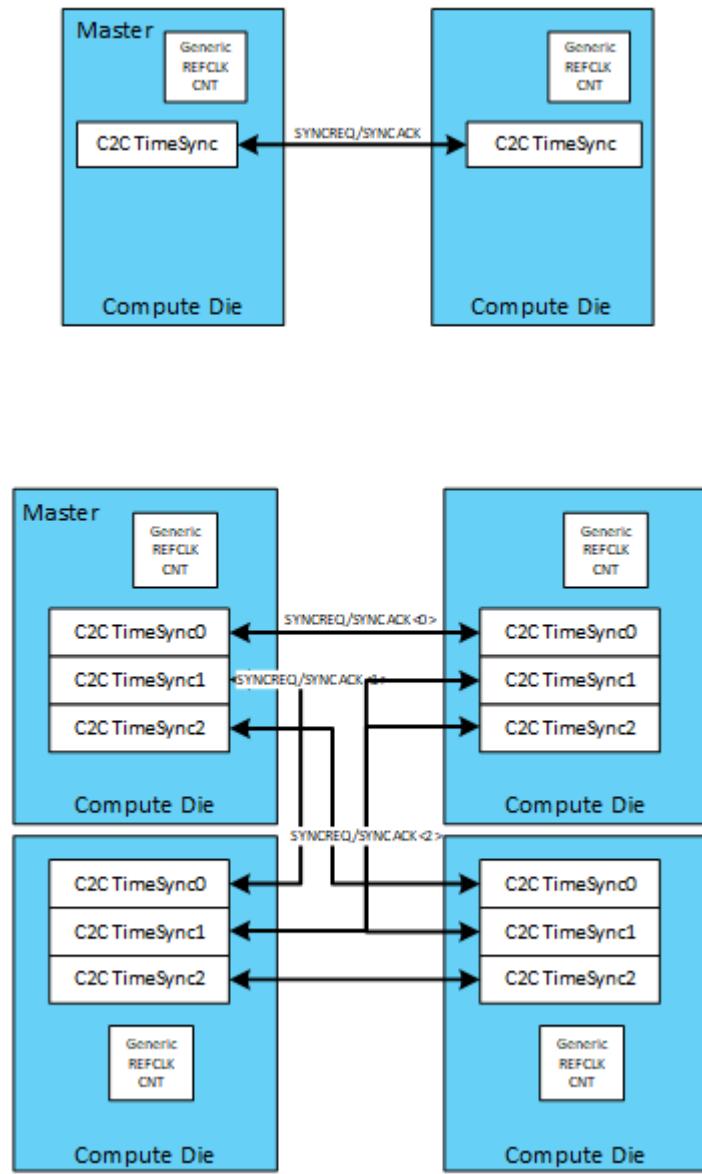
The process is repeated at regular time intervals that are programmed by the software.

The following figure shows the synchronization process for both the pass and failed cases when the slave does not receive the  $T_{update}$  value. Each phase of the process times out if the slave or the master does not respond for the time duration programmed in the respective `SLV<n>_GCNT_TIMEOUT` register.

**Figure 4-17: Synchronization process between the master and slave for the pass and fail cases**



The following figure shows the block diagram for timer synchronization logic that resides on master and slave SoCs. This logic is duplicated between each master and slave pair. So, there are three instances of this block for a multichip system with four chips.

**Figure 4-18: Timer synchronization architecture block diagram for dual and quadruple chiplets**

#### 4.8.1.6 Multichip debug

This section describes the debug and trace architecture for multichip implementations.

The following topics are covered:

- [Debug access to all chips](#)
- [Cross-triggering between chips](#)
- [Trace output from all chips](#)

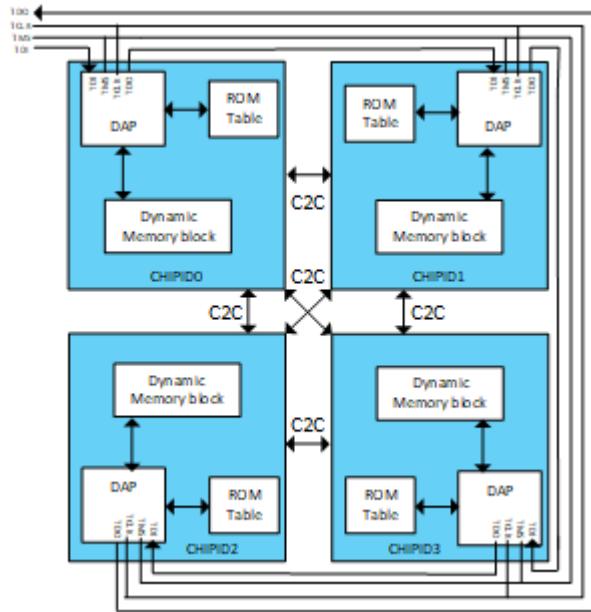
#### 4.8.1.6.1 Debug access to all chips

In a multichip system using CoreSight debug and trace, each chip has local DAPs that are used to access the memory and debug component of that chip.

The multichip system might include CCIX links providing a single address space so that each DAP can access memory on any other DAP. However, each chip retains its own DAP and these DAPs are connected together as follows:

- JTAG is connected to each chip in a daisy chain fashion for debug access to all the chips.

**Figure 4-19: Multichip debug and trace: JTAG daisy chaining for a quadruple chip system**

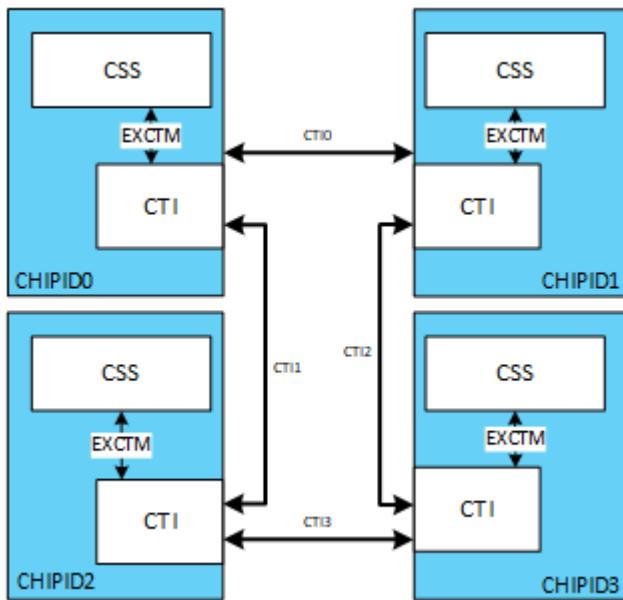


- Each chip has its own ROM table for discovery. There are no links between ROM tables on different chips.
- During early system bring-up, debug access does not rely on the C2C interfaces.

#### 4.8.1.6.2 Cross-triggering between chips

In a multichip system, triggers are passed from one chip to other chips by passing the triggers through dedicated wires using the CTI interface.

Such an arrangement requires four wires between each chip.

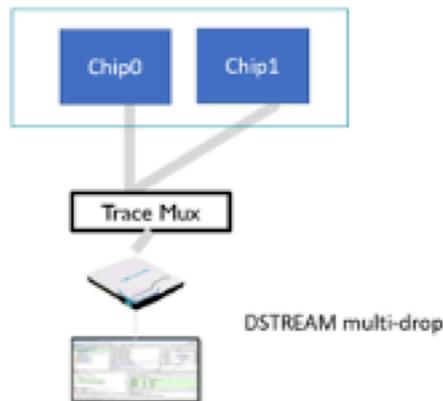
**Figure 4-20: CTI interface connectivity for a quadruple chip system**

#### 4.8.1.6.3 Trace output from all chips

RD-N2 support tracing through a dedicated TPIU interface or through functional I/O interfaces, such as PCIe or USB interfaces.

Tracing through a TPIU interface is performed as follows:

- There are individual trace outputs from each chip.
- Muxing all the interfaces using Trace Mux on the board to connect to a single DSTREAM as shown in the following figure. (Describing the details of Trace Mux is out of the scope of this document.)

**Figure 4-21: Trace outputs from all chips**

Tracing through functional interfaces is performed by using the following methods:

- The trace from all the chips can be written into DDR memory. The trace data from the memory buffer can be moved to off-chip memory storage using PCIe or USB.
- CoreSight ETR can directly write to the off-chip memory through a PCIe interface. These writes are done as MMIO writes to the storage device connected to the PCIe interface.

#### 4.8.1.7 Multichip SMP boot sequence

There is a specific boot sequence in multichip systems.

The following table shows the boot sequence.

**Table 4-17: Multichip SMP boot sequence**

SCP/MCP master chip	SCP/MCP slave chips
SCP/MCP follow the boot sequence described in <a href="#">Boot flow overview</a> .	SCP/MCP follow the boot sequence described in <a href="#">Boot flow overview</a> .
Enable the system PLLs and SYSTOP.	Enable the system PLLs and SYSTOP.
Program the CMN-700 for local routing.	Program the CMN-700 for local routing.
Initialize local DDR memories.	Initialize local DDR memories.
Check for the slave over I2C.	Enter the idle state and wait for the master to synchronize over I2C.
1. Wait until the slave is ready. 2. Check with the slaves over I2C.	1. Wait until the slave is ready. 2. Check with the slaves over I2C.
Request slave chip configuration over I2C with the slaves.	Slave chip configuration over I2C with the master.
Bring up CCIX/PCIe links.	Bring up CCIX/PCIe links.
Program the CCIX and RN SAM registers.	Program the CCIX and RN SAM registers.
Synchronize the timers.	Slave initiates to synchronize timers.
Power up one of the AP cores to boot the OS.	Enter the idle state.

#### 4.8.2 Host-to-accelerator use case

RD-N2 supports different accelerator device types.

The following device types are supported:

##### Host communication and discovery of the accelerator

The main communication between the chips can happen through CCIX or CXL interfaces. If the physical interface is PCIe, all CCIX and CXL device discovery is performed through PCIe after the PCIe link is up and running. If the physical interface is a die-to-die interface, such as XSR or USR, it is assumed that the accelerator has a microcontroller like the SCP. In this case, the accelerator configuration is communicated with the host SCP through the sideband serial interface. The physical interface is brought up and the discovery and setup of the CCIX and CXL links is completed.

## Accelerator address region mapping

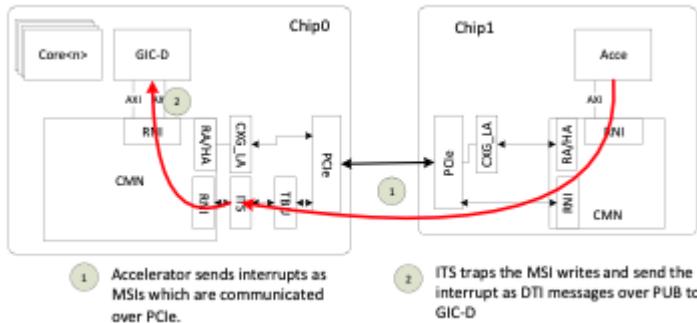
The host core assigns a memory region to the accelerator chip depending on the device type. The CMN-700 interconnect RN SAM is programmed accordingly to route the transactions to the accelerator through a CCIX or CXL link.

## Interrupt routing from the accelerator to the host core

The accelerator communicates the interrupts that are generated by sending them as MSI or MSIx to the host core. The flow of the MSI or MSIx depends on whether the device is CXL or PCIe plus CCIX.

The following figure shows the flow of the interrupts between the accelerator and the host core for CXL devices or PCIe plus CCIX devices connected over a PCIe physical interface.

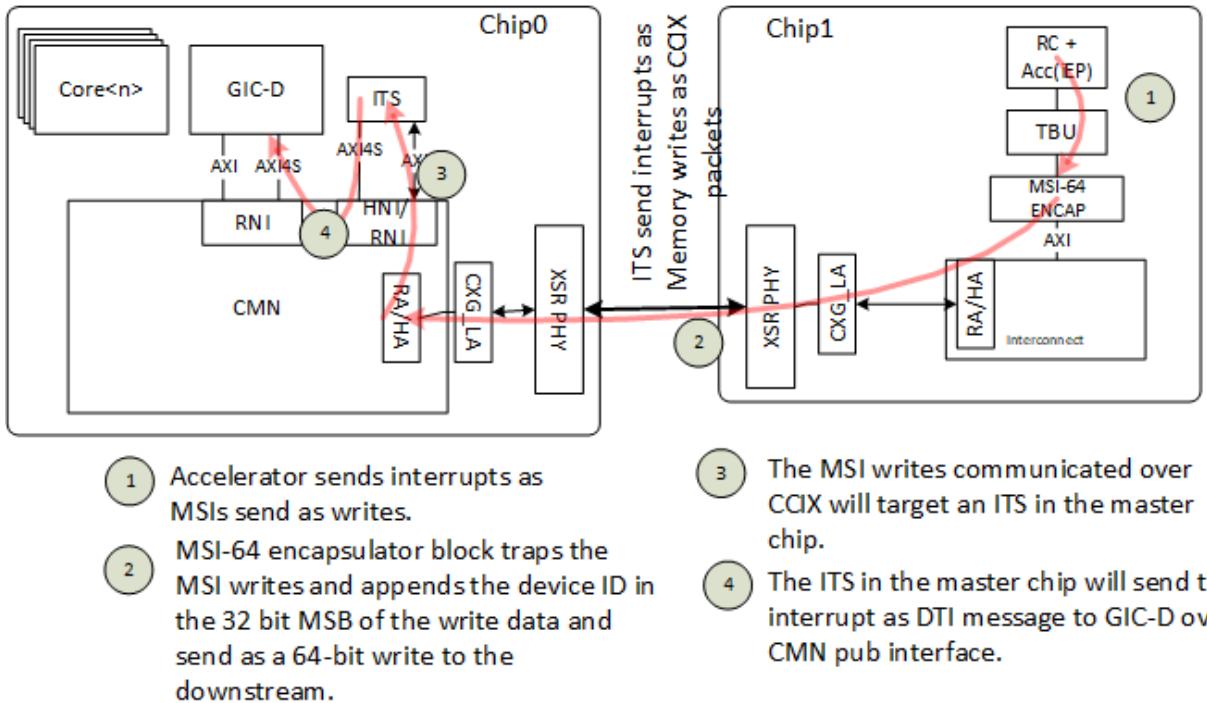
**Figure 4-22: Interrupt routing from the accelerator to the host core**



In this case, Chip0 is the main host and Chip1 is the accelerator. The main host has the global GIC Distributor. All the interrupts from the accelerator are communicated as MSI and MSIx that are sent over the PCIe interface. The MSI is trapped by the ITS in the path and the interrupt is sent to GIC-D. This arrangement is same for both PCIe-based accelerators and CCIX/CXL accelerators.

The following figure shows the flow of interrupts between the accelerator and the host core for CCIX devices connected over die-to-die interfaces such as XSR or USR.

**Figure 4-23: Interrupt routing for CCIX devices connected over die-to-die interfaces**



As before, Chip0 is the main host and Chip1 is the accelerator. The accelerator is connected directly through a die-to-die interface such as XSR or USR. The main host has the global GIC Distributor and the accelerator includes an MSI-64-like wrapper. All the interrupt generators inside the accelerator are communicated as MSI. The MSI write is trapped by the MSI-64 wrapper and appends the devicelID in the upper 32 bits of the write data. The data is sent as a 64-bit write downstream, which is sent over the CCIX interface as a CCIX packet. The MSI write directs to the ITS on Chip0. The ITS then routes the write to GIC-D on the host chip.

### 4.8.3 Host-to-memory expansion use case

CCIX or CXL allows for expansion of the system memory domain beyond the host attached memory.

The host memory manager can optionally allocate and manage peripheral attached memory in the same manner that host memory is allocated and managed, as part of system memory.

So, with memory expansion, a host can expand its memory capacity and support new memory technologies beyond what is available with the native memory capabilities of the host. Note that the view of the peripheral attached memory of the host is consistent with the existing view of memory in a multinode host system. This view being the 25 NUMA memory model.

# 5 Fixed Virtual Platform

The Fixed Virtual Platform (FVP) models much of the Arm IP in RD-N2. The FVP enables partners to develop ahead of hardware availability and to explore the design from a software perspective.

## 5.1 About the FVP

Two configurations of RD-N2 are supported:

- The RD-N2 FVP models CFG32C4M, which is a single-chiplet system with the number of Arm Neoverse N2 cores reduced to 16.
- The RD-N2 quad-chiplet FVP models a reduced-size variant of CFG32C4M, consisting of four compute subsystems linked by CMN-700 CML. It provides a functional model of a quad-chiplet system. Each subsystem contains one IOMACRO instance, 4 Arm Neoverse N2 cores for a total of 16 cores in the FVP.

The FVP models the following IP components:

- Arm Neoverse N2 MP1 cores
- CMN-700
- Multiple NIC-450 interconnects
- GIC-700
- MMU-700
- Arm Cortex-M7 SCP and MCP cores
- Memory access path towards DRAM which includes a TrustZone controller.



The FVP does not model every component that RD-N2 describes. For example, the FVP does not model the CoreSight technology components.

---

The RD-N2 FVP drives system architecture and software standardization. The models provide software and binaries of proprietary firmware that reduce the amount of work that is required for SoC development.

## 5.2 FVP peripherals

The RD-N2 FVP includes peripherals that the software payload requires to run.

The SoC peripherals area and board peripherals area in the RD-N2 memory map are mapped to an expansion AMBA AXI region. Therefore, these mappings are not defined in the reference design.

The peripherals are organized into two layers:

- SoC. The SoC peripherals represent peripherals that might be added to a compute subsystem in an SoC design. The RD-N2 SoC model is based on the Juno Arm Development Platform (ADP).
- Board. The board peripherals represent peripherals that might be present on the board onto which the SoC is mounted. The RD-N2 board model is based on the Juno (ADP).

In the quad-chiplet system described in [About the FVP](#), each compute subsystem has SoC and Board layers dedicated to that subsystem. Addressing for each chip is defined in [AP system memory map](#).

- Chip 0: 0x000\_0000\_0000 – 0x3FF\_FFFF\_FFFF
- Chip 1: 0x400\_0000\_0000 – 0x7FF\_FFFF\_FFFF
- Chip 2: 0x800\_0000\_0000 – 0xBFF\_FFFF\_FFFF
- Chip 3: 0xC00\_0000\_0000 – 0xFFFF\_FFFF\_FFFF

A sideband communication channel is required to coordinate multi-chiplet software boot over CMN-700. The FVP implements this using the MHU device, but Arm recommends using a solution such as I2C in hardware.

**Table 5-1: FVP SoC peripherals**

Name	Base address	Size	Description
SMC0 interface	0x00_0800_0000	64MB	Routed to the board.
SMC1 interface	0x010_5000_0000	256MB	Routed to the board.
PCIe Config	0x10_1000_0000	256MB	-
PCIe 32-bit Memory	0x00_6000_0000	512MB	-
PCIe 64-bit Memory	0x40_0000_0000	258GB	-
DMA MMU-400	0x00_E00_0000	64KB	-
HDLCD1 MMU-400	0x00_E01_0000	64KB	-
HDLCD0 MMU-400	0x00_E02_0000	64KB	-
SoC Interconnect NIC-400 GPV	0x00_ED0_0000	1MB	-
Surge detector	0x00_EE5_0000	4KB	Dummy APB.
TRNG	0x00_EE6_0000	4KB	-
Trusted Non-volatile counters	0x00_EE7_0000	4KB	-
Trusted Root-Key storage	0x00_EE8_0000	4KB	-
Secure I2C	0x00_EE9_0000	512 bytes	There is no Secure I2C component in the FVP. Instead, a PL061 GPIO is mapped as a dummy component.
DDR PHY 0-31	0x00_EB0_0000	2MB	Dummy APB.
DMA Secure	0x00_EF0_0000	64KB	-
DMA Non-Secure	0x00_EF1_0000	64KB	-
PCIE Macro	0x00_EF2_0000	64KB	-
PCIE Root Port	0x00_EF3_0000	64KB	-

Name	Base address	Size	Description
HDLCD1	0x00_EF5_0000	4KB	-
HDLCD0	0x00_EF6_0000	4KB	-
UART 0	0x00_EF7_0000	4KB	-
UART 1	0x00_EF8_0000	4KB	-
I2S	0x00_EF9_0000	1KB	There is no I2S component in the FVP. Instead, a PL061 GPIO is mapped as a dummy component.
I2C	0x00_EFA_0000	256 bytes	There is no I2C component in the FVP. Instead, a PL061 GPIO is mapped as a dummy component.
PL354	0x00_EFD_0000	64KB	Arm® PrimeCell® SMC dual SRAM memory interface (PL354).
System override Registers	0x00_EFF_0000	64KB	-
AP configuration	0x00_EFE_0000	64KB	Granular Power Requester (GPR).

**Table 5-2: FVP board peripherals**

Name	Base address	Size	Description
NOR Flash 0	0x00_0800_0000	64MB	-
NOR Flash 1	0x010_5000_0000	64MB	-
NOR Flash 2	0x010_5400_0000	64MB	-
Ethernet	0x010_5C00_0000	64MB	Non-PCI Ethernet controller (SMSC 91C111)
System Registers	0x00_0C01_0000	64KB	-
SP810 Sysctrl	0x00_0C02_0000	64KB	-
MCI	0x00_0C05_0000	64KB	Arm PrimeCell Multimedia Card Interface (PL180)
KMI 0	0x00_0C06_0000	64KB	Arm PrimeCell PS2 Keyboard/Mouse Interface (PL050)
KMI 1	0x00_0C07_0000	64KB	Arm PrimeCell PS2 Keyboard/Mouse Interface (PL050)
UART 0	0x00_0C09_0000	64KB	Arm PrimeCell UART (PL011)
UART 1	0x00_0C0A_0000	64KB	Arm PrimeCell UART (PL011)
Watchdog	0x00_0C0F_0000	64KB	Arm Watchdog Module (SP805)
Dual Timer	0x00_0C11_0000	64KB	Arm Dual-Timer Module (SP804)
Virtio Block Device	0x00_0C13_0000	64KB	-
Virtio Net Device	0x00_0C15_0000	64KB	-
GPIO 2Wire (DVI)	0x00_0C16_0000	64KB	Arm PrimeCell General Purpose Input/Output (PL061)
RTC0	0x00_0C17_0000	64KB	Arm PrimeCell Real Time Clock (PL031)
RTC1	0x00_0C18_0000	64KB	Arm PrimeCell Real Time Clock (PL031)
GPIO 0	0x00_0C1D_0000	64KB	Arm PrimeCell General Purpose Input/Output (PL061)
GPIO 1	0x00_0C1E_0000	64KB	Arm PrimeCell General Purpose Input/Output (PL061)
DRAM	0x00_8000_0000	2GB	-
DRAM	0x80_8000_0000	6GB	-

**Table 5-3: Interrupt map at the SoC layer**

Interrupt ID	Source
403	UART 0

Interrupt ID	Source
404	UART 1
405	HDLCD controller 0
406	SMC PL354 interface 0
413	HDLCD controller 1
418-425	DMA0 IRQ7-0
426	DMA0 IRQ abort
427	TRNG

**Table 5-4: Interrupt map at the board layer**

Interrupt offset	Source
387	RTC1
388	RTCO
389	UART0 (board)
390	UART1 (board)
391	KMI1
392	GPIO0
393	GPIO1
394	I2C GPIO
395	MCIINTRO
396	MCIINTR1
397	SMSC 91C111
458	Virtio block device
460	Virtio net
483	RTCC
484	WDT
485	KMIO
486	Dual timer
487	System registers
488	System register – USB
489	System register – Tile
490	System register – Push button
491	System register – Ethernet

The FVP is used with the RD-N2 software package.

For instructions on setting up and running the FVP, see the RD-N2 Software Bundle Readme.

# 6 Software stack

The integrated software stack provides a starting point to modify, extend, and develop the software stack for an SoC based on the RD-N2.

## 6.1 About the software

The Board Support Package (BSP) software contains firmware and kernel components that can be run on the associated FVP.

The BSP consists of:

### **SCP firmware**

Controls the system and manages the power.

### **MCP firmware**

Performs minimal MCP element initialization.

### **Application processor firmware**

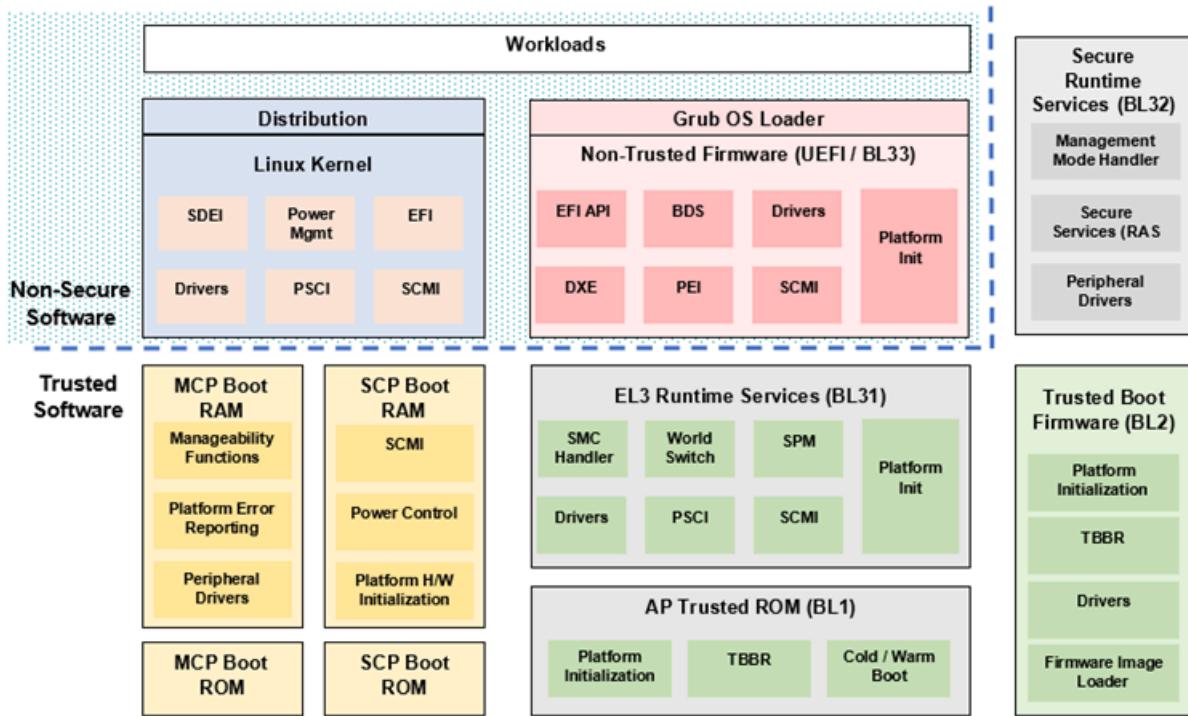
Comprising the Arm Trusted Firmware and Unified Extensible Firmware Interface (UEFI), loads the OS and provides platform services to the OS. The supported OS is a variant of Linux.

### **Linux kernel**

Contains the following kernel features, which are required to exploit and demonstrate RD-N2 hardware features:

- Linux device drivers for all the I/O components.
- Support for virtio disk.

The following figure shows a high-level overview of the software components and their features.

**Figure 6-1: RD-N2 software components and features**

## 6.2 SCP firmware

The SCP is a compute unit running in the Always-on power domain of RD-N2 that is responsible for low-level system management. The SCP is a Cortex-M7 processor with a set of dedicated peripherals and interfaces that you can extend.

The SCP firmware supports:

- Powerup sequence and system startup
- Initial hardware configuration
- Clock and regulator management
- Servicing power state requests from the OS Power Management (OSPM) software
- Multichiplet operation

The SCP firmware manages the overall power, clock, reset, and system control of RD-N2. This firmware is an inherently Trusted part of the software. All the memory that the firmware uses for execution and private storage is internal RAM to prevent tampering.

## 6.2.1 Power control

This block is the generic interface to the SCP from the application processor.

The power control performs the following functions:

- Inquires about the capabilities of the system and individual devices
- Obtains or sets the state of the whole RD-N2 subsystem and individual devices under SCP control
- Obtains or sets the performance level of the processors

The SCP provides a well-defined API for extending the standard commands in the power control interface.

## 6.2.2 SCP boot ROM

The SCP boot ROM code executes after RD-N2 exits from a Cold reset. This boot ROM configures the initial state of the hardware.

For example, the SCP boot ROM can define:

- The processor cores that are released from reset
- The clocks that are available
- The state of the PMIC

## 6.3 MCP firmware

The MCP is a Cortex-M7 processor compute unit running in the AON power domain of RD-N2. The MCP firmware is intended to control all the manageability functions and RAS features.

The MCP firmware can also be extended to communicate with an external BMC to provide event logging and communication of alerts.

## 6.3.1 MCP boot ROM

The MCP boot ROM code executes after RD-N2 exits from a Cold reset. This boot ROM configures the initial state of the hardware.

For example, the MCP boot ROM can:

- Control the MCP block clocks
- Establish communication with the SCP

## 6.4 Application processor firmware

The application processor firmware consists of the code that is required to boot RD-N2 up to the point where the OS execution starts.

The firmware contains the code that is required to:

- Set up the initial security environment
- Support runtime processor power state control using the Power State Coordination Interface (PSCI)
- Load Linux from boot media

### 6.4.1 Arm Trusted firmware BL1

The application processor contains an on-chip trusted ROM that executes the boot code on an RD-N2 SoC.

The application processor boot ROM is fixed for the lifetime of the device and executes minimal code to:

- Maximize robustness
- Reduce the risk of security vulnerabilities

### 6.4.2 Arm Trusted firmware BL2

Arm application processor Trusted firmware BL2 executes in on-chip Trusted RAM, where content is loaded from non-volatile storage.

Subsequent application processor firmware images are stored in DRAM after the application processor is initialized.

#### 6.4.2.1 TrustZone initialization, TZ Init

The AP Trusted RAM firmware initializes any Trusted world resources that the AP Trusted boot ROM either does not initialize, or any Trusted world resources that require more initialization, for example the memory controller.

### 6.4.3 Arm Trusted firmware BL31

The Arm Trusted firmware BL31 consists of the PSCI, Secure monitor framework, and Secure partition manager.

#### 6.4.3.1 Power state coordination interface

The application processor Trusted RAM firmware defines a Secure Monitor Call (SMC) interface to support rich OS power management in accordance with the PSCI system software on Arm systems.

The Linux CPU idle framework uses PSCI to power up or down the application processor cores.

#### 6.4.3.2 Secure monitor framework

The Secure monitor framework handles all SMCs in the application processor Trusted RAM firmware.

The framework handles the transition to Trusted world execution and distributes the SMCs to the correct SMC handler. The following organizations can own these SMC handlers:

- Arm
- Silicon partner (SiP)
- Original Design Manufacturer (ODM)
- Original Equipment Manufacturer (OEM)

#### 6.4.3.3 Secure partition manager

The Secure partition manager framework can be extended to pass the RAS event that is generated in the Secure world to the Normal world.

### 6.4.4 Secure runtime services BL32

Secure runtime services execute at S-ELO and interface with the Secure partition manager at EL3.

The use of Secure runtime services on a platform is optional. Typical uses of Secure services include Secure firmware-first error handling and support for Secure variable service.

### 6.4.5 Bootloader BL33

Bootloaders such as UEFI-compliant EDK II provide the interface between an operating system and platform firmware. EDK II allows early platform configuration and execution of operating system loaders such as grub.

## 6.5 Linux kernel

The RD-N2 Linux kernel contains the subsystem-specific features that demonstrate the capabilities of RD-N2.

The kernel contains the following features:

- Multiprocessing

- UEFI awareness
- Device drivers

### 6.5.1 Multiprocessing

Linux can view all processors at the same time, which enables the task scheduler to make the best use of the cores.

In addition to the multiprocessing mechanics, the Linux kernel contains optimizations to permit efficient operation on RD-N2. The optimizations include tuning to ensure that threads are scheduled efficiently across all application processor cores. These optimizations are validated against multiple use cases.

### 6.5.2 UEFI awareness

The kernel contains the functionality that is required to configure the system using UEFI, ACPI, and SMBIOS tables that the underlying firmware provides.

### 6.5.3 Device drivers

The RD-N2 Linux kernel contains various device drivers.

The following device drivers are included:

- Generic timer
- UART
- GIC (GICv3)
- Virtio network controller
- PCIe-RC and AHCI

For instructions on how to set up and run the software stack and for the User Guide, see the Arm® Neoverse™ N2 reference design Release Note.

## 6.6 Multi-chiplet support

SCP firmware supports configuring the platform for multi-chiplet operation. SCP configures CMN-700 to set up a coherent link with the other chips on the platform. SCP also uses a low-speed sideband interface to communicate inter-chip control messages such as power on/off and performance control. Using this coherent link across the chips, Linux kernel boots the application cores on all of the chips.

# 7 Programmers model

The Programmers Model describes the RD-N2 memory locations, register details, and bit description needed to program software that runs on the subsystem hardware.

## 7.1 About the programmers model

This section describes the functions and programmers model of the RD-N2.

When using the programmers model, adhere to the following guidelines:

- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in unpredictable behavior.
- Unless otherwise stated in the accompanying text:
  - Do not modify undefined register bits.
  - Ignore undefined register bits on reads.
  - Unless otherwise specified, all register bits are reset to a logic 0 by a system or power up reset.

The following describes the access type:

**RW**

Read and write.

**RO**

Read-only.

**WO**

Write-only.

**RAZ**

Read as zero.

**WI**

Writes ignored.

**RW1C**

Read and write 1 to clear.

## 7.2 Memory maps

The SCP and MCP have their own memory maps that differ from the AP memory map.

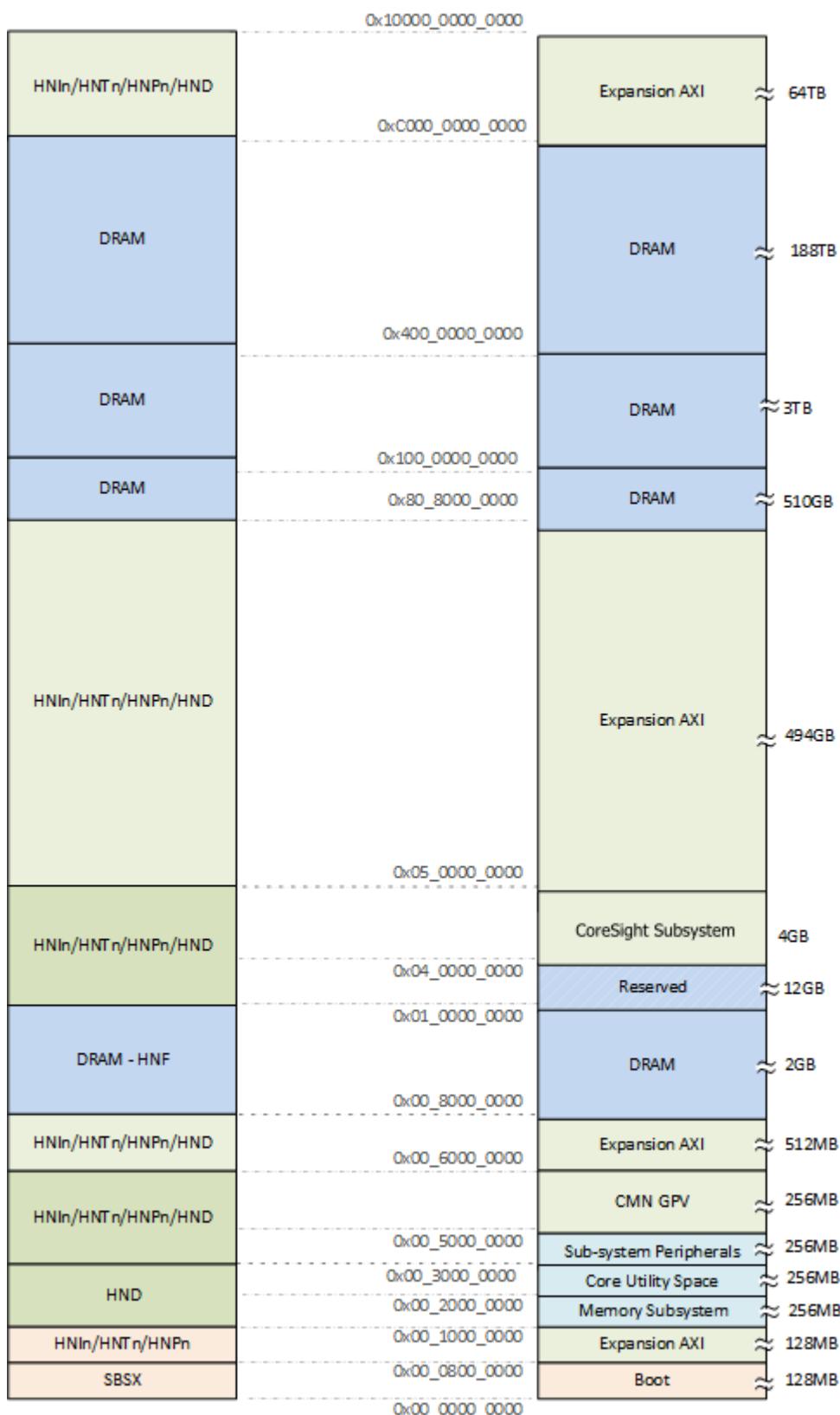
## 7.2.1 AP system memory map

Memory map for the Application Processors.

The AP memory map is visible to the following:

- APs
- SCP through address remapping
- MCP through address remapping
- Embedded Trace Router
- Coherent Expansion AXI slave interfaces
- AP DAP
- Devices in the remote chips through CCIX/CXL interfaces

The following figure shows a top-level representation of the AP memory map.

**Figure 7-1: RN SAM application processor memory map**



The memory map does not adhere to the *Principles of Arm® Memory Maps White Paper* standard.

Unless explicitly stated otherwise:

- Where a region is used to map a peripheral or device, it is possible that the peripheral or device occupies less than the region size used. For example, a peripheral might only occupy 4KB out of the 64KB of the region that is reserved for it. Access to an unmapped region causes a Decode Error (DECERR) response.
- Accesses to reserved areas within the memory map also results in a DECERR response.
- When accessing areas occupied by peripherals or devices, it is the peripherals or devices themselves that determine the response to return. These accesses can include unmapped or reserved area within the areas occupied by the peripheral or device.
- Any Non-secure access targeting Secure regions causes a DECERR response. Note: APB slaves will return SLVERR response for Secure access failed case also.

The memory map in the following tables shows the overall security attributes associated to each area of memory. These are split into the following groups:

- Always Secure Access: A component or region that is only accessible to Secure transactions. Any Non-secure access targeting these components causes a DECERR response. Note: APB slaves will return SLVERR response for Secure access failed case also.
- Non-secure Access: A component or region that is accessible to both Secure and Non-secure transactions.
- Programmable Access security (also referred to as “SW securable”): Components or regions that are defined to be independently software configurable can be changed between the above two states by trusted software. These attributes can be configured in the network interconnect, NIC-450 or in the component itself, and the default state is Secure access only from reset.
- User Defined: These attributes are areas which are mapped to expansion interfaces and the components outside of the RD-N2 subsystem defines their access security. These components must use the **ARPROT[1]** or **AWPROT[1]** bits provided on the expansion interfaces to determine the security permission of each access. Any accesses that fail any external security checks must result in a DECERR response.

For a multichip case, each chip is assigned an equal amount of memory space size of 4TB. Each chip has same memory map with a different base address. The SID\_CHIP\_ID determines the base address.

The application memory base address of each chip is shown in the following calculation:

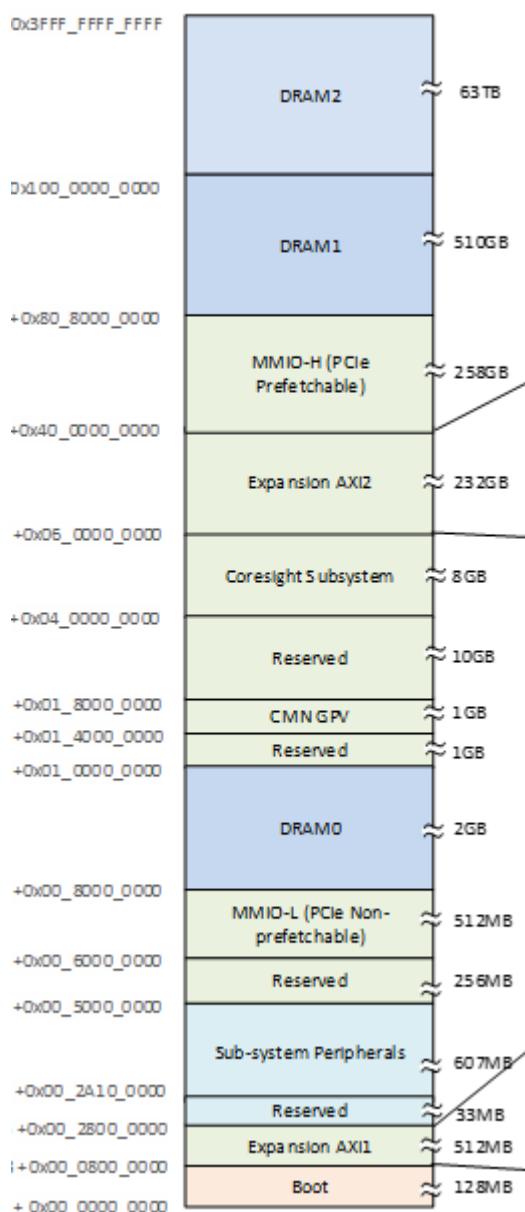
$$\text{AP\_BASE\_ADDRESS} = \text{SID\_CHIP\_ID} * 4\text{TB} + 0x000\_0000\_0000$$

For Quad-chip design, each chip is assigned the following regions:

- Chip 0: 0x000\_0000\_0000 – 0x3FF\_FFFF\_FFFF
- Chip 1: 0x400\_0000\_0000 – 0x7FF\_FFFF\_FFFF

- Chip 2: 0x800\_0000\_0000 – 0xBFF\_FFFF\_FFFF
- Chip 3: 0xC00\_0000\_0000 – 0xFFFF\_FFFF\_FFFF

**Figure 7-2: Application processor memory map for a multi-chip use case**



The following table gives more details of the application processor memory map.

**Table 7-1: Memory map for the application processors**

Start address	End address	Size	Description	Access control
0x00_0000_0000	0x00_03FF_FFFF	64MB	AP Secure BOOT ROM	Secure

Start address	End address	Size	Description	Access control
0x00_0400_0000	0x00_04FF_FFFF	16MB	AP Secure RAM	Secure
0x00_0500_0000	0x00_05FF_FFFF	16MB	AP Non-secure Boot ROM	Non-secure
0x00_0600_0000	0x00_07FF_FFFF	32MB	AP Non-secure RAM	Non-secure
0x00_0800_0000	0x00_0FFF_FFFF	128MB	AP expansion for SoC components	Non-secure
0x00_1000_0000	0x00_1FFF_FFFF	256MB	Memory region for DMC controllers	Secure
0x00_2000_0000	0x00_21FF_FFFF	32MB	Memory region for all cluster Utility space, see <a href="#">Cluster Utility memory map</a> for details	See <a href="#">Cluster Utility memory map</a>
0x00_2200_0000	0x00_27FF_FFFF	96MB	Reserved	-
0x00_2800_0000	0x00_2A0F_FFFF	33MB	Reserved	-
0x00_2A10_0000	0x00_2A1F_FFFF	1MB	Reserved GPV space for various NICs implemented in the system. The memory region for individual NICs is IMPL_DEF for each system.	Secure
0x00_2A20_0000	0x00_2A3F_FFFF	2MB	Reserved GPV space for various NICs implemented for debug system. The memory region for individual NICs is IMPL_DEF for each system.	Non-secure
0x00_2A40_0000	0x00_2A40_FFFF	64KB	AP Non-secure UART	Non-secure
0x00_2A41_0000	0x00_2A41_FFFF	64KB	AP Secure UART	Secure
0x00_2A42_0000	0x00_2A42_FFFF	64KB	Reserved	Secure
0x00_2A43_0000	0x00_2A43_FFFF	64KB	System Generic Counter control frame (GENERIC REFCLK CNT Control)	Secure
0x00_2A44_0000	0x00_2A44_FFFF	64KB	AP Non-secure WatchDog control frame	Non-secure
0x00_2A45_0000	0x00_2A45_FFFF	64KB	AP Non-secure WatchDog refresh frame	Non-secure
0x00_2A46_0000	0x00_2A47_FFFF	128KB	Reserved	Non-secure
0x00_2A48_0000	0x00_2A48_FFFF	64KB	AP Secure/trusted WatchDog control frame	Secure
0x00_2A49_0000	0x00_2A49_FFFF	64KB	AP Secure/trusted WatchDog refresh frame	Secure
0x00_2A4A_0000	0x00_2A4A_FFFF	64KB	<a href="#">System ID registers</a> (SID)	Non-secure
0x00_2A4B_0000	0x00_2A4B_FFFF	64KB	AP Secure RAM ECC Error record block (AP_S_RAM_ECC)	Secure
0x00_2A4C_0000	0x00_2A4C_FFFF	64KB	AP Non-Secure RAM ECC Error record block (AP_NS_RAM_ECC)	Non-secure
0x00_2A4D_0000	0x00_2A4D_FFFF	64KB	SCP Secure RAM Error record block	Secure
0x00_2A4E_0000	0x00_2A4E_FFFF	64KB	SCP Non Secure RAM Error record block	Non-secure
0x00_2A4F_0000	0x00_2A4F_FFFF	64KB	MCP Secure RAM Error record block	Secure
0x00_2A50_0000	0x00_2A50_FFFF	64KB	MCP Non Secure RAM Error record block	Non-secure
0x00_2A51_0000	0x00_2A7F_FFFF	3008KB	Reserved	-
0x00_2A80_0000	0x00_2A80_FFFF	64KB	System Generic Counter read frame	Non-secure
0x00_2A81_0000	0x00_2A81_FFFF	64KB	AP Generic Timer Control Frame (AP_REFCLK_CNTCTL)	Non-secure
0x00_2A82_0000	0x00_2A82_FFFF	64KB	AP Secure Generic Timer control base frame	Secure
0x00_2A83_0000	0x00_2A83_FFFF	64KB	AP Non-secure Generic Timer control base frame	Non-secure
0x00_2A84_0000	0x00_2A8F_FFFF	768KB	Reserved	-
0x00_2A90_0000	0x00_2A90_FFFF	64KB	AP to SCP Non-secure MHU send frame	Non-secure
0x00_2A91_0000	0x00_2A91_FFFF	64KB	AP to SCP Non-secure MHU receive frame	Non-secure

Start address	End address	Size	Description	Access control
0x00_2A92_0000	0x00_2A92_FFFF	64KB	AP to SCP Secure MHU send frame	Secure
0x00_2A93_0000	0x00_2A93_FFFF	64KB	AP to SCP Secure MHU receive frame	Secure
0x00_2A94_0000	0x00_2A9F_FFFF	768KB	Reserved	-
0x00_2AA0_0000	0x00_2AA0_FFFF	64KB	AP to MCP Non-secure MHU send frame	Non-secure
0x00_2AA1_0000	0x00_2AA1_FFFF	64KB	AP to MCP Non-secure MHU receive frame	Non-secure
0x00_2AA2_0000	0x00_2AFF_FFFF	6016KB	Reserved	Non-secure
0x00_2B00_0000	0x00_2B0F_FFFF	1MB	This 1MB window is used with address translation enabled in SCP and MCP	Non-secure
0x00_2B10_0000	0x00_2B10_FFFF	64KB	Base address for timer synchronization master side to send the update value.	Secure
0x00_2B11_0000	0x00_2EFF_FFFF	64448KB	Reserved	Non-secure
0x00_2F00_0000	0x00_2FFF_FFFF	16MB	Reserved	Non-secure
0x00_3000_0000	0x00_309C_FFFF	10048KB	GIC IP	Per-access Secure
0x00_3800_0000	0x00_309D_FFFF	246MB	Reserved	Non-secure
0x00_4000_0000	0x00_4FFF_FFFF	256MB	See <a href="#">I/O Virtualization block memory map</a>	-
0x00_5000_0000	0x00_5FFF_FFFF	256MB	Reserved space for CMN-700 configuration space	Per-access Secure
0x00_6000_0000	0x00_7FFF_FFFF	512MB	SoC expansion reserved for PCIe. See <a href="#">PCIe MMIO and ECAM memory regions</a> for details	Per-access Secure
0x00_8000_0000	0x00_FFFF_FFFF	2GB	DDR memory space	SW SECURABLE
0x01_0000_0000	0x01_3FFF_FFFF	1GB	Reserved	-
0x01_4000_0000	0x01_7FFF_FFFF	1GB	CMN-700 GPV space (configuration space for CMN-700)	Per-access Secure
0x01_8000_0000	0x03_FFFF_FFFF	10GB	Reserved	-
0x04_0000_0000	0x05_FFFF_FFFF	8GB	Debug memory region for software self-hosted. See <a href="#">Self-hosted memory map</a> for details	SW SECURABLE
0x06_0000_0000	0x10_0FFF_FFFF	40.25GB	Reserved	Non-secure
0x10_1000_0000	0x10_4FFF_FFFF	1GB	PCIe NCI Memory space 2. See <a href="#">PCIe MMIO and ECAM memory regions</a> for details	Per-access Secure
0x10_5000_0000	0x3F_FFFF_FFFF	190.75GB	AP expansion for SoC components	Per-access Secure
0x40_0000_0000	0x80_7FFF_FFFF	258GB	PCIe NCI Memory space 3. See <a href="#">PCIe MMIO and ECAM memory regions</a> for details	Per-access Secure
0x80_8000_0000	0xFF_FFFF_FFFF	510GB	DRAM	SW SECURABLE
0x100_0000_0000	0x3FF_FFFF_FFFF	3TB	DRAM	SW SECURABLE
0x400_0000_0000	0xBFFF_FFFF_FFFF	188TB	DRAM	SW SECURABLE
0xC000_0000_0000	0xFFFF_FFFF_FFFF	64TB	AP expansion for SoC components	Per-access Secure

### 7.2.1.1 Cluster Utility memory map

The following table shows how the memory range is decoded for different processor blocks.

**Table 7-2: Memory range decoding in processor blocks**

Instance	Start address	End address
Processor block 0	0x00_2000_0000	0x00_200F_FFFF
Processor block 1	0x00_2010_0000	0x00_201F_FFFF
Processor block 2	0x00_2020_0000	0x00_202F_FFFF
Processor block 3-31	0x00_2030_0000	0x00_21FF_FFFF

Each Processor block is assigned a Cluster Affinity based on its location in the CMN-700 mesh. The processor block numbers are marked on the CMN-700 mesh diagram. See mesh diagram [Figure 3-4: RD-N2 6 x 6 CMN-700 system mesh topology](#) on page 25 for details.

The 1MB region for each Processor block is mapped as shown in the following table.

**Table 7-3: Processor block memory map**

Start - End Offset	Register Group Name	Memory Group Size	Access Control	Further information
0x00_0000 - 0x00_FFFF	Cluster control	64KB	Secure	See Arm® Theodul DynamIQ™ Shared Unit Technical Reference Manual
0x01_0000 - 0x01_FFFF	Reserved	64KB	-	
0x02_0000 - 0x02_FFFF	Reserved	64KB	-	
0x03_0000 - 0x03_FFFF	Cluster PPU	64KB	Secure	See Arm® Theodul DynamIQ™ Shared Unit Technical Reference Manual
0x04_0000 - 0x04_FFFF	Reserved	64KB	-	
0x05_0000 - 0x05_FFFF	Core Manager and clock control registers	64KB	Secure	
0x06_0000 - 0x07_FFFF	Processor block partner expansion module	128KB	Non-secure	This expansion module used by SoC integrator for Core PLL, DVF logic, sensors, and so on
0x08_0000 - 0x08_FFFF	Core 0 PPU	64KB	Secure	See Arm® Theodul DynamIQ™ Shared Unit Technical Reference Manual
0x09_0000 - 0x09_FFFF	Core 0 Activity Monitors Unit (AMU)	64KB	Secure	See Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile for Activity Monitors External register views
0x0A_0000 - 0x0A_FFFF	Reserved	64KB	-	
0x0B_0000 - 0x0B_FFFF	Core 0 Power Performance Management (PPM)	64KB	Secure	See Arm® Neoverse™ N2 Core Technical Reference Manual for Power Performance Managements registers
0x0C_0000 - 0x0F_FFFF	Reserved	256KB	-	

## 7.2.1.2 Interconnect block NIC-450 GPV memory map

The following table shows the memory map for the NIC-450 GPV.

**Table 7-4: NIC-450 GPV memory map**

Start - End offset	Region	Register block type	Further information
0x00_0000 - 0x00_0FFF	Address control registers	Address control	See NIC-450 TRM
0x00_1000 - 0x00_1FFF	Peripheral ID registers	Peripheral ID	See NIC-450 TRM
0x00_2000 - 0x00_2FFF	SCP master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to SCP peripherals in MSCP block
0x00_3000 - 0x00_3FFF	MCP master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to MCP peripherals in MSCP block
0x00_4000 - 0x00_4FFF	Base master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to Peripheral block
0x00_5000 - 0x00_5FFF	STM master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to STM
0x00_6000 - 0x00_6FFF	ITS master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to System ITS
0x00_7000 - 0x00_7FFF	GIC master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to GIC peripherals in MSCP block
0x00_8000 - 0x00_8FFF	Debug master interface registers	Master Interface/ AMIB	Master interface from Interconnect block to Debug block for self-hosted accesses
0x00_9000 - 0x04_1FFF	Reserved	Reserved	
0x04_2000 - 0x04_2FFF	HND slave interface registers	Slave interface/ ASIB	Slave interface from HND in CMN-700
0x04_3000 - 0x04_3FFF	SBSX slave interface registers	Slave interface/ ASIB	Slave interface from Boot SBSX node in CMN-700
0x04_4000 - 0x0F_FFFF	Reserved	Reserved	

For more details about registers within each memory frame in the table above, see Arm® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual.

### 7.2.1.3 I/O Virtualization block memory map

The following table shows the memory map within each I/O Virtualization block memory frame.

**Table 7-5: Memory map for the I/O Macros**

Start address	End address	Description	Access Control
0x00_4000_0000	0x00_40BD_FFFF	TCU0 and connected TBUs	Per-access Secure
0x00_40BE_0000	0x00_40BF_FFFF	Reserved	-
0x00_40C0_0000	0x00_40CF_FFFF	NCI GPV for x16 PCIe Bifurcation Path0	Per-access Secure
0x00_40D0_0000	0x00_40D0_FFFF	PCIe Integration control registers	Boot Secure
0x00_40D1_0000	0x00_40DF_FFFF	Reserved for other components connected to NCI in the PCIe subsystem in the CSS	-
0x00_40E0_0000	0x00_40FF_FFFF	Expansion interface for PHY and other components configuration register space. (512KB for each controller PCIe0- 3).	Per-access Secure
0x00_4100_0000	0x00_41FF_FFFF	Expansion interface for x16 PCIe Bifurcation path0 (PCIe0-3 controller config space (4MB for each controller). This is the region assigned to the PCIe specific controller registers (not the ECAM space RP registers). Map this as the same expansion interface as above PCIe PHY space or maps to main PCIe CTRL slave expansion space when this 4M region is programmed in PCIe_CTRL_x*_CFG_START/END_ADDR registers.	Per-access Secure
0x00_4200_0000	0x00_42BD_FFFF	TCU1 and connected TBUs	Per-access Secure
0x00_42BE_0000	0x00_42BF_FFFF	Reserved	-
0x00_42C0_0000	0x00_42CF_FFFF	NCI GPV for x16 PCIe Bifurcation Path1	Per-access Secure
0x00_42D0_0000	0x00_42D0_FFFF	PCIe Integration control registers	Boot Secure
0x00_42D1_0000	0x00_42DF_FFFF	Reserved for other components connected to NCI in the PCIe subsystem in the CSS	-
0x00_42E0_0000	0x00_40FF_FFFF	Expansion interface for PHY and other components configuration register space. (512KB for each controller PCIe4- 7).	Per-access Secure
0x00_4300_0000	0x00_43FF_FFFF	Expansion interface for x16 PCIe Bifurcation path0 (PCIe4-7 controller config space (4MB for each controller). This is the region assigned to the PCIe specific controller registers(not the ECAM space RP registers). Map this as the same expansion interface as above PCIe PHY space or maps to main PCIe CTRL slave expansion space when this 4M region is programmed in PCIe_CTRL_x*_CFG_START/END_ADDR registers.	Per-access Secure
0x00_4400_0000	0x00_44BD_FFFF	TCU2 and connected TBUs	Per-access Secure
0x00_44BE_0000	0x00_44BF_FFFF	Reserved	-
0x00_44C0_0000	0x00_44CF_FFFF	NCI GPV for x16 PCIe Bifurcation Path1	Per-access Secure

Start address	End address	Description	Access Control
0x00_44D0_0000	0x00_44D0_FFFF	PCIe Integration control registers	Boot Secure
0x00_44D1_0000	0x00_44DF_FFFF	Reserved for other components connected to NCI in the PCIe subsystem in the CSS	-
0x00_44E0_0000	0x00_44FF_FFFF	Expansion interface for PHY and other components configuration register space. (512KB for each controller PCIe8- 11).	Per-access Secure
0x00_4500_0000	0x00_45FF_FFFF	Expansion interface for x16 PCIe Bifurcation path0 (PCIe8-11 controller config space (4MB for each controller). This is the region assigned to the PCIe specific controller registers(not the ECAM space RP registers). Map this as the same expansion interface as above PCIe PHY space or maps to main PCIe CTRL slave expansion space when this 4M region is programmed in PCIe_CTRL_x*_CFG_START/END_ADDR registers.	Per-access Secure
0x00_4600_0000	0x00_46BD_FFFF	TCU3 and connected TBUs	Per-access Secure
0x00_46BE_0000	0x00_46BF_FFFF	Reserved	-
0x00_46C0_0000	0x00_46CF_FFFF	NCI GPV for x16 PCIe Bifurcation Path1	Per-access Secure
0x00_46D0_0000	0x00_46D0_FFFF	PCIe Integration control registers	Boot Secure
0x00_46D1_0000	0x00_46DF_FFFF	Reserved for other components connected to NCI in the PCIe subsystem in the CSS	-
0x00_46E0_0000	0x00_46FF_FFFF	Expansion interface for PHY and other components configuration register space. (512KB for each controller PCIe12- 15).	Per-access Secure
0x00_4700_0000	0x00_47FF_FFFF	Expansion interface for x16 PCIe Bifurcation path0 (PCIe12-15 controller config space (4MB for each controller). This is the region assigned to the PCIe specific controller registers(not the ECAM space RP registers). Map this as the same expansion interface as above PCIe PHY space or maps to main PCIe CTRL slave expansion space when this 4M region is programmed in PCIe_CTRL_x*_CFG_START/END_ADDR registers.	Per-access Secure
0x00_4800_0000	0x00_4FFF_FFFF	Reserved	Per-access Secure

### 7.2.1.4 Self-hosted memory map

Memory map for Self-hosted debug

**Table 7-6: Memory map for Self-Hosted debug**

Base address	End address	Size	Peripheral
0x4_0000_0000	0x4_0000_0FFF	4KB	APP Debug ROM
0x4_0000_1000	0x4_0000_FFFF	-	Reserved (APP APPBIC)
0x4_0001_0000	0x4_0001_0FFF	4KB	STM ETF
0x4_0001_1000	0x4_0001_FFFF	-	Reserved (APP APPBIC)
0x4_0002_0000	0x4_0002_0FFF	4KB	System ETF
0x4_0002_1000	0x4_0008_FFFF	-	Reserved (APP APPBIC)

Base address	End address	Size	Peripheral
0x4_0009_0000	0x4_0009_0FFF	4KB	System Funnel
0x4_0009_1000	0x4_0009_FFFF	-	Reserved (APP APBIC)
0x4_000A_0000	0x4_000A_0FFF	4KB	Master Funnel
0x4_000A_1000	0x4_000A_FFFF	-	Reserved (APP APBIC)
0x4_000B_0000	0x4_000B_0FFF	4KB	DBGCH Funnel
0x4_000B_1000	0x4_0010_FFFF	-	Reserved (APP APBIC)
0x4_0011_0000	0x4_0011_0FFF	4KB	Reserved (APP APBIC)
0x4_0011_1000	0x4_0011_FFFF	-	Reserved (APP APBIC)
0x4_0012_0000	0x4_0012_0FFF	4KB	System ETR
0x4_0012_1000	0x4_0013_FFFF	-	Reserved (APP APBIC)
0x4_0014_0000	0x4_0014_0FFF	4KB	System CTI
0x4_0014_1000	0x4_0015_FFFF	-	Reserved (APP APBIC)
0x4_0016_0000	0x4_0016_0FFF	4KB	System CATU
0x4_0016_1000	0x4_0016_FFFF	-	Reserved (APP APBIC)
0x4_0017_0000	0x4_0017_0FFF	4KB	STM Debug APB
0x4_0017_1000	0x4_0017_FFFF	-	Reserved (APP APBIC)
0x4_0018_0000	0x4_0018_0FFF	-	Reserved (APP APBIC)
0x4_0018_1000	0x4_003F_FFFF	-	Reserved (APP APBIC)
0x4_0040_0000	0x4_004F_FFFF	1MB	APB Expansion Regions
0x4_0050_0000	0x4_3FFF_FFFF	-	Reserved
0x4_4000_0000	0x4_400_00FFF	4KB	DP Debug ROM
0x4_4000_1000	0x4_4000_FFFF	-	Reserved (DP APBIC)
0x4_4001_0000	0x4_4401_1FFF	8KB	SCP AHB-AP
0x4_4001_2000	0x4_4001_FFFF	-	Reserved (DP APBIC)
0x4_4002_0000	0x4_4002_1FFF	8KB	MCP AHB-AP
0x4_4002_2000	0x4_7FFF_FFFF	-	Reserved (DP APBIC)
0x4_8000_0000	0x4_8007_FFFF	512KB	Reserved (SYS APBIC)
0x4_8008_0000	0x4_8008_0FFF	4KB	SYSDBG DEBUG ROM
0x4_8008_1000	0x4_8008_FFFF	-	Reserved (SYS APBIC)
0x4_8009_0000	0x4_8009_1FFF	8KB	APP APB-AP
0x4_8009_2000	0x4_8009_FFFF	-	Reserved (SYS APBIC)
0x4_800A_0000	0x4_800A_1FFF	8KB	SYSMEM AXI-AP
0x4_800A_2000	0x4_800A_FFFF	-	Reserved (SYS APBIC)
0x4_800B_0000	0x4_800B_0FFF	4KB	Reserved
0x4_800B_1000	0x4_800B_FFFF	-	Reserved (SYS APBIC)
0x4_800C_0000	0x4_800C_0FFF	4KB	Reserved
0x4_800C_1000	0x4_800C_FFFF	-	Reserved (SYS APBIC)
0x4_800D_0000	0x4_800D_0FFF	4KB	Reserved
0x4_800D_1000	0x4_800D_FFFF	-	Reserved (SYS APBIC)
0x4_800E_0000	0x4_800E_1FFF	8KB	DBGCH APB-AP

Base address	End address	Size	Peripheral
0x4_800E_2000	0x4_FFFF_FFFF	-	Reserved (SYS APBIC)
0x5_0000_0000	0x5_7EFF_FFFF	127*16MB	Cluster 0-126 Debug APB - 16MB for each; See <a href="#">Core&lt;n&gt; SS APB IC</a>
0x5_7F00_0000	0x5_7FEF_FFFF	15MB	Cluster 127 DSU Debug Block
0x5_7FF0_0000	0x5_7FFE_FFFF	-	Reserved
0x5_7FFF_0000	0x5_7FFF_0FFF	4KB	Cluster 127 Core SS DBGROM
0x5_7FFF_1000	0x5_7FFF_1FFF	4KB	Reserved
0x5_7FFF_2000	0x5_7FFF_2FFF	4KB	Cluster 127 CTI
0x5_7FFF_3000	0x5_7FFF_3FFF	4KB	Cluster 127 Trace Funnel
0x5_7FFF_4000	0x5_7FFF_4FFF	4KB	Cluster 127 ETF
0x5_7FFF_5000	0x5_7FFF_EFFF	-	Reserved
0x5_7FFF_F000	0x5_7FFF_FFFF	4KB	DBGCH ROM TABLE
0x5_8000_0000	0x5_FFFF_FFFF	-	Reserved

### 7.2.1.5 PCIe MMIO and ECAM memory regions

PCIe requires two memory regions.

#### Enhanced Configuration Access Method memory region

PCI devices have a set of registers referred to as configuration space and PCI Express introduces extended configuration space for devices. Configuration space registers are mapped to memory locations. PCIe devices are referred to by the bus, device, and function number.

Each PCIe device function requires 4K bytes of the configuration space. There could be 256 buses, each with up to 32 devices, each supporting eight functions in a device PCIe tree .

Each PCIe device tree is called a segment. So you need total 256MB of contiguous memory region for the config space for this Enhanced Configuration Access method (ECAM) region allocated for each segment. ECAM space to be split across different RPs that are in the same segments depending on allocation of the bus numbers for the devices behind it.

Also you could have multiple segments in the single or multichip systems. For example you could have following:

- Single PCIe segment across 4 chips
- Each chip has its own PCIe segment. Separate 256MB of ECAM region for each chip. Within each chip the 256MB is split across how many ever RPs.
- Multiple segments within the chips.
- Peer-to-Peer traffic is possible only between the devices that are in the segment

#### PCIe memory-mapped address space

Each PCIe device will have internal memory which needs to be memory mapped so that it is visible to the system memory. During PCIe discovery, software will get the information on how much memory space requirements from device configuration registers. Some of the legacy devices can decode 32bit addresses, so they require memory region allocated in lower

4GB space. So architecture must provide two memory regions one in the lower 4GB space called MMIOL and another above 4G space called MMIOH.

Also PCIe device might need non-prefetchable PCIe Memory-Mapped (MMIO) space which has to be in the lower 4GB space and prefetchable MMIO space which can be anywhere in the system memory. Each device non-prefetchable memory needs to be allocated from the MMIOL region and prefetchable memory can be allocated from MMIOH region.

All this PCIe memory requirements needs to be supported by NI-700 if the outgoing I/O path from CMN-700 to PCIe root port includes NCI network.

The following table shows how the various memory regions are allocated.

**Table 7-7: Memory map for PCIe MMIO and ECAM regions through I/O Macro**

Start address	End address	Name	Description	Remapping
0x00_6000_0000	0x00_7FFF_FFFF	PCIe NCI Memory space1	MMIOL - SoC expansion reserved for PCIe. This address region can be mapped to each of the PCIe path using RNSAM to particular HNP connected to respective X16 path. This address region is used for PCIe non-prefetchable memory space.	Software to program the address region allocated to each HNP is split according to the devices needs connected behind each x16 path and programmed in the corresponding PCIe_CTRL<x>_MMIOL_START_ADDR and PCIe_CTRL<x>_MMIOL_END_ADDR registers
0x10_1000_0000	0x10_1FFF_FFFF	ECAM0 - PCIe NCI Memory space2	ECAM0 Memory region - This region is reserved for ECAM space. This address region is split across multiple HNPs as per required on what devices are connected behind each PCIe controller. This is mapped to each PCIe path using RNSAM to particular HNP connected to respective X16 path.	Software to program the address region allocated to each HNP is split according to the devices connected behind each x16 path and programmed in the corresponding PCIe_CTRL<x>_ECAM_START_ADDR and PCIe_CTRL<x>_ECAM_END_ADDR registers
0x10_2000_0000	0x10_2FFF_FFFF	ECAM1 - PCIe NCI Memory space2	ECAM1 Memory region - This region is reserved for ECAM space. This address region is split across multiple HNPs as per required on what devices are connected behind each PCIe controller. This is mapped to each PCIe path using RNSAM to particular HNP connected to respective X16 path.	Software to program the address region allocated to each HNP is split according to the devices connected behind each x16 path and programmed in the corresponding PCIe_CTRL<x>_ECAM_START_ADDR and PCIe_CTRL<x>_ECAM_END_ADDR registers
0x10_3000_0000	0x10_3FFF_FFFF	ECAM2 - PCIe NCI Memory space2	ECAM2+D13 Memory region - This region is reserved for ECAM space. This address region is split across multiple HNPs as per required on what devices are connected behind each PCIe controller. This is mapped to each PCIe path using RNSAM to particular HNP connected to respective X16 path.	Software to program the address region allocated to each HNP is split according to the devices connected behind each x16 path and programmed in the corresponding PCIe_CTRL<x>_ECAM_START_ADDR and PCIe_CTRL<x>_ECAM_END_ADDR registers
0x10_4000_0000	0x10_4FFF_FFFF	ECAM3 - PCIe NCI Memory space2	ECAM3 Memory region - This region is reserved for ECAM space. This address region is split across multiple HNPs as per required on what devices are connected behind each PCIe controller. This is mapped to each PCIe path using RNSAM to particular HNP connected to respective X16 path.	Software to program the address region allocated to each HNP is split according to the devices connected behind each x16 path and programmed in the corresponding PCIe_CTRL<x>_ECAM_START_ADDR and PCIe_CTRL<x>_ECAM_END_ADDR registers

Start address	End address	Name	Description	Remapping
0x40_0000_0000	0x80_7FFF_FFFF	PCIe NCI Memory space3	MMIOH - This memory region is used as MMIO space. It can be used as prefetchable MMIO space or some part of it can be also used as non-prefetchable memory with address translated to lower 4GB.	Software needs to program the address region allocated to each HNP is split and programmed in either of the corresponding: - PCIe_CTRL<x>_MMIOH_START_ADDR and PCIe_CTRL<x>_MMIOH_END_ADDR registers - PCIe_CTRL<x>_MMIOH2L_TR_START_ADDR and PCIe_CTRL<x>_MMIOH2L_TR_END_ADDR registers

## 7.2.2 PCIe address mapping

The RNSAM will be programmed to map the various PCIe address regions(ECAM, MMIOL, MMIOH regions) to respective HNPs connected to each x16 PCIe path.

Second level of address mapping is done to map the address regions to various PCIe controllers supporting the PCIe bifurcation in the x16 PCIe path. This is done by programming the respective PCIe\_CTRL\_x\*\_START\_ADDR and PCIe\_CTRL\_x\*\_END\_ADDR. If the incoming address from the CMN-700 mesh matches the range that is programmed in the PCIe\_CTRL\_x\*\_START\_ADDR and PCIe\_CTRL\_x\*\_END\_ADDR registers, then the address is modified and sent to the NCI. The following table specifies how the address is modified.

**Table 7-8: PCIe Address mapping**

Match condition = Falls in the range programmed in:	If CMN-700 mesh support 52bit address. Set address bits[55:52]	If CMN-700 mesh support 48bit address. Set address bits[51:48]
PCIe_CTRL_x4_0*	Set the address bit[55] = 1	Set the address bit[51] = 1
PCIe_CTRL_x4_1*	Set the address bit[54] = 1	Set the address bit[50] = 1
PCIe_CTRL_x8*	Set the address bit[53] = 1	Set the address bit[49] = 1
PCIe_CTRL_x16*	Set the address bit[52] = 1	Set the address bit[48] = 1

In addition to setting the above address bit, if the address falls in the range programmed in PCIe\_CTRL\_x\*\_MMIOH2L\_TR\_START\_ADDR and PCIe\_CTRL\_x\*\_MMIOH2L\_TR\_END\_ADDR registers replace the MSB bits [51:32] or [47:32] with '0's and set [31]=1.



These MSB bits from NCI are dropped from the address sent to the respective PCIe controllers.

## 7.2.3 SCP memory map

The SCP is a Cortex-M7 based subsystem which implements a 32-bit address space.

The Cortex-M7 uses a fixed high-level memory map as specified in the *Arm® v7-M Architecture Reference Manual*. The SCP core is always treated as a trusted core. See the *Arm® v7-M Architecture Reference Manual* for recommendations regarding the use of these memory areas.

**Table 7-9: Memory map for the SCP**

Start - End address	Size	Description	Access control	Further information
0x0000_0000 - 0x0001_FFFF	128KB	SCP Boot ROM	Secure	See <a href="#">System Control Processor block</a>
0x0002_0000 - 0x007F_FFFF	8064KB	Reserved for SCP Boot ROM	-	-
0x0080_0000 - 0x0087_FFFF	512KB	SCP Instruction RAM	Secure	See <a href="#">System Control Processor block</a>
0x0088_0000 - 0x00FF_FFFF	7680KB	Reserved for SCP Instruction RAM	-	-
0x0100_0000 - 0x0FFF_FFFF	240MB	SCP Private Expansion Area 1	Secure	Accesses are routed on SCP expansion AXI4 Master interface EXMSPAXI
0x1000_0000 - 0x1FFF_FFFF	256MB	Reserved	-	-
0x2000_0000 - 0x2007_FFFF	512KB	SCP Tightly Coupled Data RAM	Secure	See <a href="#">System Control Processor block</a>
0x2008_0000 - 0x20FF_FFFF	15872KB	Reserved	-	-
0x2100_0000 - 0x2FFF_FFFF	240MB	Reserved	-	-
0x3000_0000 - 0x3FFF_FFFF	256MB	SCP Private Expansion Area 2	Secure	Accesses are routed on SCP expansion AXI4 Master interface EXMSPAXI
0x4000_0000 - 0x43FF_FFFF	64MB	SCP Private Expansion Area 3	Secure	Accesses are routed on SCP expansion AXI4 Master interface EXMSPAXI
0x4400_0000 - 0x4400_0FFF	4KB	SCP Timer control frame	Secure	See <a href="#">Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile</a> - Memory-Mapped Timer Components module for CNTCTLBase registers. Also, see <a href="#">Generic timer registers</a> for implementation-specific timer registers.
0x4400_1000 - 0x4400_1FFF	4KB	SCP Timer frame	Secure	See <a href="#">Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile</a> - Memory-Mapped Timer Components module for CNTBaseN registers. Also, see <a href="#">Generic timer registers</a> for implementation-specific timer registers.

Start - End address	Size	Description	Access control	Further information
0x4400_2000 - 0x4400_2FFF	4KB	SCP UART	Secure	-
0x4400_3000 - 0x4400_5FFF	12KB	Reserved	-	-
0x4400_6000 - 0x4400_6FFF	4KB	SCP Watchdog Timer	Secure	-
0x4400_7000 - 0x44FF_FFFF	16356KB	Reserved	-	-
0x4500_0000 - 0x4500_FFFF	64KB	AP to SCP Non-secure MHU send frame	Secure	See <a href="#">Message Handling Unit registers</a>
0x4501_0000 - 0x4501_FFFF	64KB	AP to SCP Non-secure MHU receive frame	Secure	See <a href="#">Message Handling Unit registers</a>
0x4502_0000 - 0x4502_FFFF	64KB	AP to SCP Secure MHU send frame	Secure	See <a href="#">Message Handling Unit registers</a>
0x4503_0000 - 0x4503_FFFF	64KB	AP to SCP Secure MHU receive frame	Secure	See <a href="#">Message Handling Unit registers</a>
0x4504_0000 - 0x457F_FFFF	7936KB	Reserved	-	-
0x4580_0000 - 0x4580_FFFF	64KB	SCP to MCP Non-secure MHU send frame	Secure	See <a href="#">Message Handling Unit registers</a>
0x4581_0000 - 0x4581_FFFF	64KB	SCP to MCP Non-secure MHU receive frame	Secure	See <a href="#">Message Handling Unit registers</a>
0x4582_0000 - 0x46FF_FFFF	24448KB	Reserved	-	-
0x4700_0000 - 0x4700_FFFF	64KB	Reserved	-	-
0x4701_0000 - 0x4701_FFFF	64KB	Reserved	-	-
0x4702_0000 - 0x4702_FFFF	64KB	Reserved	-	-
0x4703_0000 - 0x4FFF_FFFF	147264KB	Reserved	-	-

Start - End address	Size	Description	Access control	Further information
0x5000_0000 - 0x5000_FFFF	64KB	SCP Power Control registers	Secure	-
0x5001_0000 - 0x5001_FFFF	64KB	Reserved	-	-
0x5002_0000 - 0x5002_FFFF	64KB	Debug Power Integration Kit registers	Secure	-
0x5003_0000 - 0x5003_FFFF	64KB	Reserved	-	-
0x5004_0000 - 0x5004_FFFF	64KB	System Power Integration Kit registers	Secure	-
0x5005_0000 - 0x50EF_FFFF	15040KB	Reserved	-	-
0x50F0_0000 - 0x50FF_FFFF	1MB	Expansion area reserved for SoC Power control Expansion	Secure	Accesses are routed on SCP expansion AXI4 Master interface EXMSCPAXI
0x5100_0000 - 0x5FFF_FFFF	240MB	Reserved	-	-
0x6000_0000 - 0x9FFF_FFFF	1GB	Memory region to access AP Memory map.	Secure	See <a href="#">Address Translation</a>
0xA000_0000 - 0xDFFF_FFFF	1GB	Memory region to access AP Memory map.	Secure	See <a href="#">Address Translation</a>
0xE000_0000 - 0xE00F_FFFF	1MB	Cortex-M7 Debug Memory region	Secure	See <a href="#">SCP Cortex-M7 Private Peripheral Bus</a>
0xE010_0000 - 0xFFFF_FFFF	511MB	Reserved	-	-

## 7.2.4 MCP memory map

The MCP is a Cortex-M7 based subsystem that implements a 32-bit address space.

The Cortex-M7 uses a fixed high-level memory map as specified in the *Arm® v7-M Architecture Reference Manual* (which also contains recommendations regarding the use of these memory areas).

**Table 7-10: Memory map for the MCP**

Start - End address	Size	Description	Access control	Further information
0x0000_0000 - 0x0001_FFFF	128KB	MCP Boot ROM	Non-secure	See <a href="#">Manageability Control Processor block</a>
0x0002_0000 - 0x007F_FFFF	8064KB	Reserved for MCP Boot ROM	-	-
0x0080_0000 - 0x0087_FFFF	512KB	MCP Instruction RAM	Non-secure	See <a href="#">Manageability Control Processor block</a>
0x0088_0000 - 0x00FF_FFFF	7680MB	Reserved for MCP Instruction RAM	-	-
0x0100_0000 - 0x1FFF_FFFF	496MB	MCP Expansion memory region 1	Non-secure	Accesses are routed on MCP expansion AXI4 Master interface EXMMCPAXI
0x2000_0000 - 0x2007_FFFF	512KB	MCP Tightly Coupled Data RAM	Non-secure	See <a href="#">Manageability Control Processor block</a>
0x2008_0000 - 0x20FF_FFFF	15872KB	Reserved	-	-
0x2100_0000 - 0x3FFF_FFFF	496MB	MCP SoC expansion region 2	Non-secure	Accesses are routed on MCP expansion AXI4 Master interface EXMMCPAXI
0x4000_0000 - 0x43FF_FFFF	64MB	MCP SoC expansion region 3	Non-secure	Accesses are routed on MCP expansion AXI4 Master interface EXMMCPAXI
0x4400_0000 - 0x455F_FFFF	22MB	Reserved	-	-
0x4560_0000 - 0x4560_FFFF	64KB	MCP to SCP Non-secure MHU send frame	Non-secure	See <a href="#">Message Handling Unit registers</a>
0x4561_0000 - 0x4561_FFFF	64KB	MCP to SCP Non-secure MHU receive frame	Non-secure	See <a href="#">Message Handling Unit registers</a>
0x4562_0000 - 0x47FF_FFFF	42880KB	Reserved	-	-
0x4800_0000 - 0x4BFF_FFFF	64MB	MCP SoC expansion region 4	Non-secure	Accesses are routed on MCP expansion AXI4 Master interface EXMMCPAXI

Start - End address	Size	Description	Access control	Further information
0x4C00_0000 - 0x4C00_0FFF	4KB	MCP Timer control frame	Non-secure	This is a private timer used only by the MCP. See <a href="#">Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile - Memory-Mapped Timer Components module</a> for CNTCTLBase registers. Also, see <a href="#">Generic timer registers</a> for implementation-specific timer registers.
0x4C00_1000 - 0x4C00_1FFF	4KB	MCP Timer frame	Non-secure	This is a private timer used only by the MCP. See <a href="#">Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile - Memory-Mapped Timer Components module</a> for CNTBaseN registers. Also, see <a href="#">Generic timer registers</a> for implementation-specific timer registers.
0x4C00_2000 - 0x4C00_2FFF	4KB	MCP UART	Non-secure	-
0x4C00_3000 - 0x4C00_5FFF	12KB	Reserved	-	-
0x4C00_6000 - 0x4C00_6FFF	4KB	MCP Watchdog Timer	Non-secure	-
0x4C00_7000 - 0x4C3F_FFFF	4068KB	Reserved	-	-
0x4C40_0000 - 0x4C40_FFFF	64KB	MCP to AP Non-secure MHU send frame	Non-secure	See <a href="#">Message Handling Unit registers</a>
0x4C41_0000 - 0x4C41_FFFF	64KB	MCP to AP Non-secure MHU receive frame	Non-secure	See <a href="#">Message Handling Unit registers</a>
0x4C42_0000 - 0x4FFF_FFFF	61312KB	Reserved	-	-
0x5000_0000 - 0x5000_FFFF	64KB	MCP Power Control registers	Non-secure	-
0x5001_0000 - 0x507F_FFFF	8128KB	Reserved	-	-
0x5080_0000 - 0x5FFF_FFFF	248MB	Reserved	-	-
0x6000_0000 - 0x9FFF_FFFF	1GB	Memory region to access AP Memory map.	Non-secure	See <a href="#">Address Translation</a>
0xA000_0000 - 0xDFFF_FFFF	1GB	Memory region to access AP Memory map.	Non-secure	See <a href="#">Address Translation</a>

Start - End address	Size	Description	Access control	Further information
0xE000_0000 - 0xE00F_FFFF	1MB	Cortex-M7 Debug Memory region	Non-secure	See MCP Cortex-M7 Private Peripheral Bus
0xE010_0000 - 0xFFFF_FFFF	511MB	Reserved	-	-

## 7.2.5 Address translation table

Cortex-M7 address space

**Table 7-11: Address translations from SCP and MCP regions to application processor core regions**

Cortex-M7 address space	DBG_ADDR_TRANS[EN]	ADDR_TRANS[EN]	ADDR_TRANS[CMN_ATRANS_EN]	ADDR_TRANS[ADDR_47_20]	Translated address space
0xCB00_0000 - 0xCB0F_FFFF	0	1	X	Program field with bits[47:20] of address on multichip application processor memory map.	Above 2GB on local or remote memory map.  {ADDR_TRAN[47:20] + 0x0_0000} - {ADDR_TRAN[47:20] + 0xF_FFFF}
0xA000_0000 - 0xDFFF_FFFF	1	X	X	Unused	CHIP_ADDR_SZ*CHIPID + ({DBG_ADDR_TRANS[31:16], ADDR_IN[31:28]} - {DBG_ADDR_TRANS[15:12]}, ADDR_IN[27:0])
0xA000_0000 - 0xDFFF_FFFF	0	0	0	Unused	Lower 1GB on local chip.  CHIP_ADDR_SZ*CHIPID + (0x0000_0000 - 0x3FFF_FFFF)
0x6000_0000 - 0x9FFFF_FFFF	X	X	1	Unused	Translate to CMN-700 configuration space.  CHIP_ADDR_SZ*CHIPID + (0x01_4000_0000 - 0x01_7FFF_FFFF)
0x6000_0000 - 0x9FFFF_FFFF	0	0	0	Unused	1GB-2GB on local chip.  CHIP_ADDR_SZ*CHIPID + (0x4000_0000 - 0x7FFF_FFFF)

## 7.2.6 Debug memory maps

Memory map for the Debug interfaces

### 7.2.6.1 DP IC debug memory map

Debug memory map for debug components behind the main DP IC.

For details on the peripherals, see Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual.

**Table 7-12: Debug memory map for debug components behind the main DP IC.**

Base address	End address	Size	Peripheral
0x0000_0000	0x0000_0FFF	4KB	DP Debug ROM
0x0000_1000	0x0000_FFFF	-	Reserved
0x0001_0000	0x0001_1FFF	8KB	SCP AHB-AP
0x0001_2000	0x0001_FFFF	-	Reserved
0x0002_0000	0x0002_1FFF	8KB	MCP AHB-AP
0x0002_2000	0x0007_FFFF	-	Reserved
0x0008_0000	0x000F_FFFF	512KB	SYSDBG APB IC, see <a href="#">Debug memory for SYSDBG APB IC</a>
0x0010_0000	0x001F_FFFF	-	Reserved
0x0020_0000	0xFFFF_FFFF	-	Reserved

### 7.2.6.2 Debug memory for SYSDBG APB IC

Debug memory map for all the components behind SYSDBG APB IC.

For details on the peripherals, see Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual.

**Table 7-13: Debug memory map for all the components behind SYSDBG APB IC.**

Base address	End address	Size	Peripheral	Exp
0x00080000	0x00080FFF	4KB	SYSDBG Debug ROM	0
0x00081000	0x0008FFFF	-	Reserved	-
0x00090000	0x00091FFF	8KB	APP APB-AP	0
0x00092000	0x0009FFFF	-	Reserved	-
0x000A0000	0x000A1FFF	8KB	SYSMEM AXI-AP	0
0x000A2000	0x000AFFFF	-	Reserved	-
0x000B0000	0x000B0FFF	4KB	Reserved	1
0x000B1000	0x000BFFFF	-	Reserved	-
0x000C0000	0x000C0FFF	4KB	Reserved	1
0x000C1000	0x000CFFFF	-	Reserved	-
0x000D0000	0x000D0FFF	4KB	Reserved	1
0x000D1000	0x000DFFFF	-	Reserved	-
0x000E0000	0x000E1FFF	8KB	DBGCH APB-AP	0

Base address	End address	Size	Peripheral	Exp
0x000E2000	0x000FFFFF	-	Reserved	-

### 7.2.6.3 Debug domain memory map for APP APB IC

Memory map for debug components with in the system debug domain behind APP APB IC.

For details on the peripherals, see *Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual*.

**Table 7-14: Memory map for debug components with in the system debug domain behind APP APB IC.**

Base address	End address	Size	Peripheral	Exp
0x00000000	0x00000FFF	4KB	APP Debug ROM	0
0x00001000	0x0000FFFF	-	Reserved	-
0x00010000	0x00010FFF	4KB	STM ETF	2
0x00011000	0x0001FFFF	-	Reserved	-
0x00020000	0x00020FFF	4KB	System ETF	1
0x00021000	0x0008FFFF	-	Reserved	-
0x00090000	0x00090FFF	4KB	System Funnel	1
0x00091000	0x0009FFFF	-	Reserved	-
0x000A0000	0x000A0FFF	4KB	Master Funnel	1
0x000A1000	0x000AFFFF	-	Reserved	-
0x000B0000	0x000B0FFF	4KB	DBGCH Funnel	1
0x000B1000	0x0010FFFF	-	Reserved	-
0x00110000	0x00110FFF	4KB	Replicator	1
0x00111000	0x0011FFFF	-	Reserved	-
0x00120000	0x00120FFF	4KB	ETR	1
0x00121000	0x0013FFFF	-	Reserved	-
0x00140000	0x00140FFF	4KB	CTI	3
0x00141000	0x0015FFFF	-	Reserved	-
0x00160000	0x00160FFF	4KB	CATU	1
0x00161000	0x0016FFFF	-	Reserved	-
0x00170000	0x00170FFF	4KB	Debug Element STM Debug	2
0x00171000	0x0017FFFF	-	Reserved	-
0x00180000	0x00180FFF	-	Reserved	0
0x00181000	0x003FFFFFF	-	Reserved	-
0x00400000	0x004FFFFF	1MB	APB Expansion Regions	0
0x00500000	0x4FFFFFFF	-	Reserved	-

## 7.2.6.4 Debug memory map for DBGCHx APB IC

Memory map for debug components behind DBGCHx APB IC.

**Table 7-15: Memory map for debug components behind DBGCHx APB IC.**

Base address	End address	Size	Peripheral
0x00000000	0x7FFFFFFF	127*16MB	Cluster CoreSight Memory Map. See <a href="#">Core&lt;n&gt; SS APB IC</a> for Clusters 0-126. - 16MB for each cluster
0x7F000000	0x7FEFFFFF	15MB	Cluster 127 DSU Debug Block
0x7FF00000	0x7FFFFFFF	-	Reserved
0x7FFF0000	0x7FFF0FFF	4KB	Cluster 127 Core SS DBGROM
0x7FFF1000	0x7FFF1FFF	4KB	Reserved
0x7FFF2000	0x7FFF2FFF	4KB	Cluster 127 CTI
0x7FFF3000	0x7FFF3FFF	4KB	Cluster 127 Trace Funnel
0x7FFF4000	0x7FFF4FFF	4KB	Cluster 127 ETF
0x7FFF5000	0x7FFF5FFF	-	Reserved
0x7FFF6000	0x7FFFFFFF	4KB	DBGCH ROM TABLE
0x80000000	0xFFFFFFF	2GB	Reserved

## 7.2.6.5 SCP Cortex-M7 Private Peripheral Bus

Memory map for SCP Cortex-M7 Private Peripheral Bus (PPB).

For details on the peripherals, see [Arm® Cortex®-M7 Processor Technical Reference Manual](#).

**Table 7-16: Memory map for SCP Cortex-M7 Private Peripheral Bus**

Base address	End address	Size	Peripheral
0x0_E000_0000	0x0_E000_0FFF	4KB	ITM
0x0_E000_1000	0x0_E000_1FFF	4KB	DWT
0x0_E000_2000	0x0_E000_2FFF	4KB	FPB
0x0_E000_3000	0x0_E000_DFFF	44KB	Reserved
0x0_E000_E000	0x0_E000_EFFF	4KB	SCS
0x0_E000_F000	0x0_E003_FFFF	196KB	Reserved
0x0_E004_0000	0x0_E004_0FFF	4KB	Reserved
0x0_E004_1000	0x0_E004_1FFF	4KB	ETM
0x0_E004_2000	0x0_E004_2FFF	4KB	CTI
0x0_E004_3000	0x0_E00F_DFFF	748KB	SCP EPPB APBIC
0x0_E00F_E000	0x0_E00F_EFFF	4KB	SCP Core ROM TABLE
0x0_E00F_F000	0x0_E00F_FFFF	4KB	SCP PPB ROM TABLE

## 7.2.6.6 SCP core debug components

Memory map for SCP core debug components.

For details on the peripherals, see Arm® Cortex®-M7 Processor Technical Reference Manual.

**Table 7-17: Memory map for SCP core debug components.**

Base address	End address	Size	Peripheral
0x0_E004_0000	0x0_E004_0FFF	-	Reserved
0x0_E004_1000	0x0_E004_1FFF	4KB	ETM
0x0_E004_2000	0x0_E004_2FFF	4KB	CTI
0x0_E004_3000	0x0_E004_FFFF	-	Reserved
0x0_E005_0000	0x0_E006_FFFF	128KB	SCP SS APB IC
0x0_E005_1000	0x0_E00F_DFFF	-	Reserved
0x0_E00F_E000	0x0_E00F_EFFF	4KB	SCP Core ROM TABLE

## 7.2.6.7 SCP SS APB IC

Memory map for SCP APB IC.

For details on the peripherals, see Arm® Cortex®-M7 Processor Technical Reference Manual.

**Table 7-18: Memory map for SCP APB IC**

Base address	End address	Size	Peripheral
0x0_E005_0000	0x0_E005_0FFF	4KB	SCP SS ROM BASE
0x0_E005_1000	0x0_E005_1FFF	4KB	Reserved
0x0_E005_2000	0x0_E005_2FFF	4KB	Reserved
0x0_E005_3000	0x0_E005_3FFF	4KB	SWO
0x0_E005_4000	0x0_E005_4FFF	4KB	SCP FUNNEL
0x0_E005_5000	0x0_E005_5FFF	4KB	SCP ATB REPLICATOR
0x0_E005_6000	0x0_E005_6FFF	4KB	CTI
0x0_E005_6000	0x0_E006_FFFF	-	Reserved

## 7.2.6.8 MCP Cortex-M7 Private Peripheral Bus

Memory map for MCP Cortex-M7 Private Peripheral Bus (PPB).

For details on the peripherals, see Arm® Cortex®-M7 Processor Technical Reference Manual.

**Table 7-19: Memory map for MCP Cortex-M7 Private Peripheral Bus**

Base address	End address	Size	Peripheral
0x0_E000_0000	0x0_E000_0FFF	4KB	ITM

Base address	End address	Size	Peripheral
0x0_E000_1000	0x0_E000_1FFF	4KB	DWT
0x0_E000_2000	0x0_E000_2FFF	4KB	FPB
0x0_E000_3000	0x0_E000_DFFF	44KB	Reserved
0x0_E000_E000	0x0_E000_EFFF	4KB	SCS
0x0_E000_F000	0x0_E003_FFFF	196KB	Reserved
0x0_E004_0000	0x0_E004_0FFF	4KB	Reserved
0x0_E004_1000	0x0_E004_1FFF	4KB	ETM
0x0_E004_2000	0x0_E004_2FFF	4KB	CTI
0x0_E004_3000	0x0_E00F_DFFF	748KB	MCP EPPB APBIC
0x0_E00F_E000	0x0_E00F_EFFF	4KB	MCP Core ROM TABLE
0x0_E00F_F000	0x0_E00F_FFFF	4KB	MCP PPB ROM TABLE

### 7.2.6.9 MCP core debug components

Memory map for MCP core debug components.

For details on the peripherals, see Arm® Cortex®-M7 Processor Technical Reference Manual.

**Table 7-20: Memory map for MCP core debug components.**

Base address	End address	Size	Peripheral
0x0_E004_0000	0x0_E004_0FFF	-	Reserved
0x0_E004_1000	0x0_E004_1FFF	4KB	ETM
0x0_E004_2000	0x0_E004_2FFF	4KB	CTI
0x0_E004_3000	0x0_E004_FFFF	-	Reserved
0x0_E005_0000	0x0_E006_FFFF	128KB	MCP SS APB IC
0x0_E005_1000	0x0_E00F_DFFF	-	Reserved
0x0_E00F_E000	0x0_E00F_EFFF	4KB	MCP Core ROM TABLE

### 7.2.6.10 MCP SS APB IC

Memory map for Manageability Control Processor APB IC.

For details on the peripherals, see Arm® Cortex®-M7 Processor Technical Reference Manual.

**Table 7-21: Memory map for MCP SS APB IC**

Base address	End address	Size	Peripheral
0x0_E005_0000	0x0_E005_0FFF	4KB	MCP SS ROM BASE
0x0_E005_1000	0x0_E005_1FFF	4KB	Reserved
0x0_E005_2000	0x0_E005_2FFF	4KB	Reserved
0x0_E005_3000	0x0_E005_3FFF	4KB	Reserved

Base address	End address	Size	Peripheral
0x0_E005_4000	0x0_E005_4FFF	4KB	MCP FUNNEL
0x0_E005_5000	0x0_E005_5FFF	4KB	MCP ATB REPLICATOR
0x0_E005_6000	0x0_E005_6FFF	4KB	CTI
0x0_E005_6000	0x0_E006_FFFF	-	Reserved

### 7.2.6.11 Core<n> SS APB IC

Memory map for Debug components connected to the core SS APB IC.

For details on the peripherals, see *Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual*.

**Table 7-22: Memory map for Debug components connected to the core SS APB IC**

Base address	End address	Size	Peripheral
0x0000_0000	0x00EF_FFFF	15MB	CORE DEBUG BLOCK APBIC
0x00F0_0000	0x00FE_FFFF	-	Reserved
0x00FF_0000	0x00FF_0FFF	4KB	CORE SS DBGROM
0x00FF_1000	0x00FF_1FFF	-	Reserved
0x00FF_2000	0x00FF_2FFF	4KB	CTI
0x00FF_3000	0x00FF_3FFF	4KB	TRACE FUNNEL
0x00FF_4000	0x00FF_4FFF	4KB	ETF
0x00FF_5000	0x00FF_FFFF	-	Reserved

## 7.3 Interrupt maps

This section describes the RD-N2 interrupt maps.

### 7.3.1 PPI interrupt map

The Generic Interrupt Controller Architecture defines two different types of interrupts. PPIs exist separately for every microprocessor, while SPIs are shared for all microprocessors.

PPI and SPI interrupts have configurable options, including number of interrupts, Edge or Level triggered, and Polarity (that is, active-LOW, active-HIGH, or Rising edge). The GIC supports 48 PPIs interrupts per core and total of 1984 SPI interrupts. The SPI interrupts are split across multichip. So for a quad chip, each chip will have support for 480 (have to be multiples of 32 interrupts for each).

For a quad multichip system the SPI are split as shown in the following table:

**Table 7-23: Multichip Interrupt ID allocation**

Chip#	CHIP_START_INTID	CHIP_END_INTID
Chip0	32	511
Chip1	512	991
Chip2	4096	4575
Chip3	4576	5055

**Table 7-24: Application Processor PPIs**

Interrupt ID	Interrupt Name	Description	Trigger	Polarity
16	nERRIRQ	Core Error Interrupt	Level	Active-LOW
17	nFaultIRQ	Core Fault Interrupt	Level	Active-LOW
18	nTBEIRQ	Trace Buffer Extension interrupt request.	Level	Active-LOW
19	CNTHVS	Secure EL2 virtual timer interrupt (if PEs are Armv8.4 or greater).	Level	Active-LOW
20	CNTHPS	Secure EL2 physical timer interrupt (if PEs are Armv8.4 or greater)	Level	Active-LOW
21	PMBIRQ	Statistical Profiling Interrupt, if Statistical Profiling Extensions are implemented	Level	Active-LOW
22	COMMIRQn	Debug Communications Channel receive or transmit request	Level	Active-LOW
23	PMUIRQn	PMU interrupt	Level	Active-LOW
24	CTIIRQ	CTI Interrupt	Edge	rising_edge
25	VCPUMNTIRQn	Virtual Maintenance Interrupt	Level	Active-LOW
26	CNTHPIRQn	Hypervisor Non-secure EL2 Timer event	Level	Active-LOW
27	CNTVIRQn	Virtual Timer event	Level	Active-LOW
28	CNTHVIRQn	EL2 virtual timer	Level	Active-LOW
29	CNTPSIRQn	Secure PL1 Physical Timer event	Level	Active-LOW
30	CNTPNSIRQn	Non-secure PL1 Physical Timer event	Level	Active-LOW
31	RESERVED	-	-	
1087-1056	RESERVED	Reserved	Level	Active-LOW (must be tied HIGH if unused)

**Table 7-25: Application Processor SPIs**

Interrupt ID Range for each CHIPID0	Interrupt ID (Offset from CHIP_START_INTID)	Source	Description	Trigger	Polarity
15:0	[\$NUM_DMC]+0	dmc<n>_combined_err_intr	Combined interrupts from each DMC controller. Where "n" is 0 to NUM_DMC-1 Controllers.	Level	Active-HIGH

Interrupt ID Range for each CHIPID0	Interrupt ID (Offset from CHIP_START_INTID)	Source	Description	Trigger	Polarity
31:16	31:16	Reserved	-	Level	Active-HIGH (must be tied LOW)
32	32	MCP2APMHU_NS	MCP2AP MHU Non-Secure interrupt from the receiver frame	Level	Active-HIGH
40:33	40:33	Reserved	-	Level	Active-HIGH (must be tied LOW)
44:41	[\$NUM_DTC]+41	CMN-700 INTREQPMU_DTC[\$NUM_DTC]	CMN-700 PMU Count Overflow Interrupt. NUM_DTC = 1 to 4	Level	Active-HIGH
45	45	INTREQERRS	Secure error handling interrupt from CMN-700	Level	ACTIVE-High
46	46	INTREQFAULTS	Secure Fault handling interrupt from CMN-700	Level	ACTIVE-High
47	47	INTREQFAULTNS	Non Secure Fault handling interrupt from CMN-700	Level	ACTIVE-High
48	48	INTREQPMU	PMU count overflow interrupt	Level	ACTIVE-High
49	49	INTREQMPAMERRNS	Non-secure Memory Partitioning And Monitoring (MPAM) fault handling interrupt	Level	ACTIVE-High
50	50	INTREQMPAMERRS	Secure Memory Partitioning And Monitoring (MPAM) fault handling interrupt	Level	ACTIVE-High
63:51	63:51	Reserved	-	Level	Active-HIGH (must be tied LOW)
64	64	CATU	CATUADDRERROR interrupt	Level	Active-HIGH
65	65	ETR	ETRBUFINT interrupt	Level	Active-HIGH
66	66	SCP2AP MHU Non Secure interrupt	SCP2AP MHU Non-Secure interrupt from the receiver frame	Level	Active-HIGH

Interrupt ID Range for each CHIPID0	Interrupt ID (Offset from CHIP_START_INTID)	Source	Description	Trigger	Polarity
67	67	SCP2AP MHU Secure interrupt	SCP2AP MHU Secure interrupt from the receiver frame	Level	Active-HIGH
71:68	71:68	Reserved	CMN-700 PMU Count Overflow Interrupt. NUM_DTC = 1 to 4	Level	Active-HIGH (must be tied LOW)
72	72	STM-500	STM-500 Synchronization Interrupt	Edge	rising_edge
73	73	CTI	CTI Trigger output 6 from CTI2	Edge	rising_edge
74	74	CTI	CTI Trigger output 7 from CTI2	Edge	rising_edge
75	75	APP Secure Watchdog	Secure Watchdog interrupt(WS0)	Level	Active-HIGH
76	76	AP_REFCLK Generic Timer (Secure)	AP_REFCLK Generic Timer Interrupt (Secure)	Level	Active-HIGH
77	77	AP_REFCLK Generic Timer (Non- Secure)	AP_REFCLK Generic Timer Interrupt (Non-Secure)	Level	Active-HIGH
78	78	AP_NSWDOG_WSO	AP Non-secure Watchdog timer interrupt WSO	Level	Active-HIGH
79	79	AP_NSWDOG_WS1	AP Non-secure Watchdog timer interrupt WS1	Level	Active-HIGH
80	80	AP_UART_NS	AP Non-Secure UART interrupt	Level	Active-HIGH
81	81	AP_UART_S	AP Secure UART interrupt	Level	Active-HIGH
85:82	85:82	Reserved	CMN-700 PMU Count Overflow Interrupt. NUM_DTC = 1 to 4	Level	Active-HIGH (must be tied LOW)
133:86	5*[+\$NUM_TCU+1]+86 : 6*[\$NUM_TCU]+86	MMUTCU[\$NUM_TCU]_PMU_IRPT	PMU interrupt. NUM_TCU = 0 to 7. Note: This interrupt allocation start lowest Interrupt ID in this range.	Edge	Rising Edge

Interrupt ID Range for each CHIPID0	Interrupt ID (Offset from CHIP_START_INTID)	Source	Description	Trigger	Polarity
		MMUTCU[\$NUM_TCU]_EVENT_Q_IRPT_S	Event Queue Secure interrupt, indicating Event Queue Non-Empty or Overflow. These Secure interrupts can be wired or sent as write messages to set the SPI interrupt.	Edge	Rising Edge
		MMUTCU[\$NUM_TCU]_CMD_SYNC_IRPT_S	SYNC Complete Secure interrupt. These Secure interrupts can be wired or sent as write messages to set the SPI interrupt.	Edge	Rising Edge
		MMUTCU[\$NUM_TCU]_GLOBAL_IRPT_S	Global Secure interrupt. These Secure interrupts can be wired or sent as write messages to set the SPI interrupt.	Edge	Rising Edge
		MMUTCU[\$NUM_TCU]_ERRI	TCU RAS ERI interrupt	Edge	Rising Edge
		MMUTCU[\$NUM_TCU]_FHI	TCU RAS FHI interrupt	Edge	Rising Edge
325:134	2*[\$NUM_TCU]+[\$NUM_TBU_[\$NUM_TCU]]+134: 3*[\$NUM_TCU]+[\$NUM_TBU_[\$NUM_TCU]]+134  NUM_TCU = 0 to 7 NUM_TBU_[\$NUM_TCU] = 0 to 7	MMUTBU[\$NUM_TCU][\$NUM_TBU_[\$NUM_TCU]]_PMU_IRPT	TBU PMU Interrupt.  Note: This interrupt allocation start lowest Interrupt ID in this range The LSB is MMUTBU[0][0][\0]]_PMU_IRPT Note: Each interrupt is the combination of the output from the read & write TBUs. For individual TBU status, read the ITOP_TBU register (see ??? for more details)	Edge	Rising Edge

Interrupt ID Range for each CHIPID0	Interrupt ID (Offset from CHIP_START_INTID)	Source	Description	Trigger	Polarity
		MMUTBU[\$NUM_TCU][\$NUM_TBU_[\$NUM_TCU]]_RAS_ERI	TBU RAS ERI Interrupt Note: Each interrupt is the combination of the output from the read & write TBUs. For individual TBU status, read the ITOP_TBU register (see ??? for more details)	Edge	Rising Edge
		MMUTBU[\$NUM_TCU][\$NUM_TBU_[\$NUM_TCU]]_RAS_FHI	TBU RAS FHI Interrupt Note: Each interrupt is the combination of the output from the read & write TBUs. For individual TBU status, read the ITOP_TBU register (see ??? for more details)	Edge	Rising Edge
341:326	[\$NUM_PCIE_NCI]+326 : [\$NUM_PCIE_NCI]+326	PCIe[\$NUM_PCIE_NCI]_NCI_NS_IN T,	Non-secure interrupt for PCIe<x> NCI networks, where NUM_PCIE_NCI = 0 to 7  Note: This interrupt allocation start lowest Interrupt ID in this range. PCIe0_NCI_NS_INT is allocated the LSB of this interrupt range.	Level	Active-High
		PCIe[\$NUM_PCIE_NCI]_NCI_S_INT	Secure interrupt for PCIe<x> NCI networks, where NUM_PCIE_NCI = 0 to 7		
349:342	349:342	{PCIe[\$NUM_PCIE_NCI]_NCI_nPMU_INTERRUPT[\$NUM_NCI_CLKDOMAIN]}>	Consolidated PMU counter overflow interrupts from various clock domains from each PCIe<x> NCI networks, where NUM_PCIE_NCI = 0 to 7	Level	Active-Low

## 7.3.2 SCP NVIC interrupt map

The SCP receives interrupts from four sources. These interrupts are routed to the NVIC that is included in the Cortex-M7 processor, where they can be managed by software.

The four sources of interrupts are:

- Applications processor system wakeup interrupts
- CoreSight power and reset request interrupts
- Internal SCP interrupts
- Expansion SCP interrupts

**Table 7-26: Interrupt map for the SCP NVIC**

ID	Source	Description	Trigger	Polarity
NMI	SCP Generic Watchdog	SCP Watchdog(WS0)	Level	ACTIVE-High
0	CDBGPWRREQ_INT	Debug Power up request from Debugger	Level	ACTIVE-High
1	CSYSPWRREQ_INT	SYSTOP Power up request from Debugger	Level	ACTIVE-High
2	CDBGRSTREQ_INT	Debug Reset request from Debugger	Level	ACTIVE-High
3	CSYSRSTREQ_INT	SYSTOP Reset request from Debugger	Level	ACTIVE-High
4	RESERVED	-	Level	Active-High (tie off 1'b0)
12:5	DBGCH<0-7>PWRREQ_INT	Debug Chain<x> power up request from the Debugger	Level	ACTIVE-High
20:13	DBGCH<0-7>RSTREQ_INT	Debug Chain<x> Reset request from the Debugger	Level	ACTIVE-High
32:21	RESERVED	-	-	Active-High (tie off 1'b0)
33	SCP REFCLK Generic Timer	SCP Timer interrupt	Level	ACTIVE-High
34	GENTIM_SYNC_INST0	System generic timer synchronization interrupt for SYNCNT_INST0	Level	ACTIVE-HIGH
35	GENTIM_SYNC_INT1	System generic timer synchronization interrupt for SYNCNT_INST1	Level	ACTIVE-HIGH
36	GENTIM_SYNC_INT2	System generic timer synchronization interrupt SYNCNT_INST2	Level	ACTIVE-HIGH
37	CTI	CTI Trigger 0	Edge	ACTIVE-High
38	CTI	CTI Trigger 1	Edge	ACTIVE-High
39	GIC_FAULT_INT	GIC Fault Interrupt	Level	ACTIVE-High
40	GIC_ERR_INT	GIC Error INT	Level	ACTIVE-High
41	RESERVED	-	-	Active-High (tie off 1'b0)
42	SCP_UART_INT	SCP UART interrupt	Level	ACTIVE-High
43	RESERVED	-	-	Active-High (tie off 1'b0)
44	AP_NSWDG_WSO	AP Non-secure Watchdog timer interrupt WSO	Level	ACTIVE-High
45	AP_NSWDG_WS1	AP Non-secure Watchdog timer interrupt WS1	Level	ACTIVE-High
46	AP_SWDOG_WSO	AP Secure Watchdog timer interrupt WSO	Level	ACTIVE-High

ID	Source	Description	Trigger	Polarity
47	AP_SWDOG_WS1	AP Secure Watchdog timer interrupt WS1	Level	ACTIVE-High
48	AP_UART_NS	AP Non-Secure UART interrupt	Level	ACTIVE-High
49	AP_UART_S	AP Secure UART interrupt	Level	ACTIVE-High
50	Consolidated CPU Core Power Policy Units	Consolidated CPU PPU Interrupt for 0-31 cores	Level	ACTIVE-High
51	Consolidated CPU Core Power Policy Units	Consolidated CPU PPU Interrupt for 32-63 cores	Level	ACTIVE-High
52	Consolidated CPU Core Power Policy Units	Consolidated CPU PPU Interrupt for 64-95 cores	Level	ACTIVE-High
53	Consolidated CPU Core Power Policy Units	Consolidated CPU PPU Interrupt for 96-127 cores	Level	ACTIVE-High
54	RESERVED	-	Level	Active-High (tie off 1'b0)
55	Consolidated CPU Core PLL Lock	Consolidated CPU PLL Lock for PLLs 0-31, 64-95	Level	ACTIVE-High
56	Consolidated CPU Core PLL Lock	Consolidated CPU PLL Lock for PLLs 32-63, 96-127	Level	ACTIVE-High
57	Consolidated CPU PLL Unlock	Consolidated CPU PLL Unlock for PLLs 0-31, 64-95	Level	ACTIVE-High
58	Consolidated CPU PLL Unlock	Consolidated CPU PLL Unlock for PLLs 32-63, 96-127	Level	ACTIVE-High
59	Consolidated CPU Cluster PPU interrupt 0	Consolidated CPU cluster PPU Interrupt for clusters 0-31	Level	ACTIVE-LOW
60	Consolidated CPU Cluster PPU interrupt 1	Consolidated CPU cluster PPU Interrupt for clusters 32-63	Level	ACTIVE-LOW
61	Consolidated CPU Cluster PPU interrupt 2	Consolidated CPU cluster PPU Interrupt for clusters 64-95	Level	ACTIVE-LOW
62	Consolidated CPU Cluster PPU interrupt 3	Consolidated CPU cluster PPU Interrupt for clusters 96-127	Level	ACTIVE-LOW
63	Consolidated Cluster SCF INTO	Reserved for consolidated cluster SCF Interrupt for clusters 0-31	Level	ACTIVE-LOW
64	Consolidated Cluster SCF INT1	Reserved for consolidated cluster SCF Interrupt for clusters 32-63	Level	ACTIVE-LOW
65	Consolidated Cluster SCF INT2	Reserved for consolidated cluster SCF Interrupt for clusters 64-95	Level	ACTIVE-LOW
66	Consolidated Cluster SCF INT3	Reserved for consolidated cluster SCF Interrupt for clusters 96-127	Level	ACTIVE-LOW
81:67	RESERVED	-	Level	Active-High (tie off 1'b0)
82	AP2SCP MHU Non-secure int	AP2SCP MHU High Priority Non-Secure interrupt	Level	ACTIVE-High
83	AP2SCP MHU Secure int	AP2SCP MHU Secure interrupt	Level	ACTIVE-High
84	MCP2SCP MHU Non-secure Int	MCP2SCP MHU High Priority Int	Level	ACTIVE-High
93:85	RESERVED	-	Level	Active-High (tie off 1'b0)
94	MMU_TCU_RAS_INT	Consolidated MMU RAS_CRI/ERI/FHI interrupts from all the TCUs	Level	ACTIVE-High
95	MMU_TBU_RAS_INT	Consolidated TBU RAS_CRI/ERI/FHI interrupts from all the TBUs	Level	ACTIVE-High
96	INTREQPPU	PPU interrupt from CMN-700	Level	ACTIVE-High

ID	Source	Description	Trigger	Polarity
97	INTREQERRNS	Non Secure error handling interrupt from CMN-700	Level	ACTIVE-High
98	INTREQERRS	Secure error handling interrupt from CMN-700	Level	ACTIVE-High
99	INTREQFAULTS	Secure Fault handling interrupt from CMN-700	Level	ACTIVE-High
100	INTREQFAULTNS	Non Secure Fault handling interrupt from CMN-700	Level	ACTIVE-High
101	INTREQPMU	PMU count overflow interrupt	Level	ACTIVE-High
119:102	RESERVED	-	Level	Active-High (tie off 1'b0)
127:120	DBGCH[0-7]_PPU_INT	Debug Chain Power control PPU Interrupts	Level	ACTIVE-High
129:128	RESERVED	-	Level	Active-High (tie off 1'b0)
130	DEBTOP_PPU_INT	Debug PIK Interrupt	Level	ACTIVE-High
131	LOGIC_PPU_INT	LOGIC PPU Interrupt	Level	ACTIVE-High
135:132	RESERVED	-	Level	Active-High (tie off 1'b0)
136	DPU_PPU_INT	Display PPU Interrupt	Level	ACTIVE-High
137	GPU_PPU_INT	GPU PPU Interrupt	Level	ACTIVE-High
138	RESERVED	-	Level	Active-High (tie off 1'b0)
139	MCP WS1	MCP Watch dog reset	Level	ACTIVE-High
179:140	RESERVED	-	Level	Active-High (tie off 1'b0)
204:180	dmc<n>_combined_err_intr	Combined interrupts from each DMC controller. Where "n" is 0 to NUM_DMC-1	Level	ACTIVE-High
	dmc<n>_combine_temp_intr		Level	ACTIVE-High
207:205	RESERVED	-	-	Active-High (tie off 1'b0)
239:208	External Interrupts	32 Interrupts for SCP SoC Expansion	-	ACTIVE-High

### 7.3.3 MCP NVIC interrupt map

The MCP receives interrupts from three sources. These interrupts are routed to the NVIC that is included in the Cortex-M7 processor, where they can be managed by software.

The three sources of interrupts are;

- CoreSight power and reset request interrupts
- Internal MCP interrupts
- Expansion MCP interrupts

**Table 7-27: Interrupt map for the MCP NVIC**

ID	Source	Description	Trigger	Polarity
NMI	MCP Generic Watchdog	MCP Watchdog(WS0)	Level	Active-High

ID	Source	Description	Trigger	Polarity
32:1	RESERVED	-	Level	Active-High (tie off 1'b0)
33	MCP REFCLK Generic Timer	REFCLK Physical Timer interrupt	Level	Active-High
34	Non-secure AP2MCP MHU	MHU Non-Secure interrupt	Level	Active-High
36:35	RESERVED	-	Level	Active-High
37	CTI	CTI Trigger 0	Edge	Rising & Falling Edge
38	CTI	CTI Trigger 1	Edge	Rising & Falling Edge
39	GIC_FAULT_INT	GIC Fault Interrupt	Level	ACTIVE-High
40	GIC_ERR_INT	GIC Error INT	Level	ACTIVE-High
41	RESERVED	-	Level	Active-High (tie off 1'b0)
42	MCP_UART_INT	MCP UART interrupt	Level	ACTIVE-High
83:43	RESERVED	-	Level	Active-High (tie off 1'b0)
84	MCP2SCP MHU Non-secure Int	MCP2SCP MHU High Priority Int	Level	Active-High
93:85	RESERVED	-	Level	Active-High (tie off 1'b0)
94	MMU_TCU_RAS_INT	Consolidated MMU RAS_CRI/ERI/FHI interrupts from all the TCUs	Level	ACTIVE-High
95	MMU_TBU_RAS_INT	Consolidated TBU RAS_CRI/ERI/FHI interrupts from all the TBUs	Level	ACTIVE-High
96	RESERVED	-	Level	ACTIVE-High
97	INTREQERRNS	Non-secure error handling interrupt from Porter	Level	ACTIVE-High
98	INTREQFAULTNS	Non-secure Fault handling interrupt from CMN-700	Level	ACTIVE-High
138:99	RESERVED	-	Level	Active-High (tie off 1'b0)
139	SCP WS1	SCP Watch dog reset	Level	Active-High
207:140	RESERVED	-	Level	Active-High
239:208	External Interrupts	32 Interrupts for MCP SoC Expansion	Level	Active-High

## 7.4 Register descriptions

This section contains information that can be used to program the subsystem. Each description provides details about a register, such as configurations, attributes, and bit assignments.

## 7.4.1 System ID registers

The System ID (SID) registers allow the embedding of identification and configuration information in a system.

The following table lists the registers and their corresponding address offsets for the SID. SID\_BASE is the base address of the SID. All registers support Secure and Non-secure accesses.

**Table 7-28: System ID register summary**

Offset	Name	Description	Type	Reset	Width
SID_BASE + 0x0040	SID_SYSTEM_ID	System ID System Identification	RO	0x0047_77B6	32-bit
SID_BASE + 0x0050	SID_SOC_ID	System ID SoC Identification	RO	CFG_DEF	32-bit
SID_BASE + 0x0060	SID_CHIP_ID	System ID Chip Identification	RO	CFG_DEF	32-bit
SID_BASE + 0x0070	SID_SYSTEM_CFG	System ID System Configuration	RO	CFG_DEF	32-bit
SID_BASE + 0x0FD0	SID PID_4	System ID Peripheral ID 4	RO	0x0000_0004	32-bit
SID_BASE + 0x0FE0	SID PID_0	System ID Peripheral ID 0	RO	0x0000_00BC	32-bit
SID_BASE + 0x0FE4	SID PID_1	System ID Peripheral ID 1	RO	0x0000_00B0	32-bit
SID_BASE + 0x0FE8	SID PID_2	System ID Peripheral ID 2	RO	0x0000_000B	32-bit
SID_BASE + 0x0FEC	SID PID_3	System ID Peripheral ID 3	RO	0x0000_0000	32-bit
SID_BASE + 0x0FF0	COMPID0	System ID Component ID 0	RO	0x0000_000D	32-bit
SID_BASE + 0x0FF4	COMPID1	System ID Component ID 1	RO	0x0000_00F0	32-bit
SID_BASE + 0x0FF8	COMPID2	System ID Component ID 2	RO	0x0000_0005	32-bit
SID_BASE + 0x0FFC	COMPID3	System ID Component ID 3	RO	0x0000_00B1	32-bit

### 7.4.1.1 SID\_SYSTEM\_ID, System ID System Identification register

The SID\_SYSTEM\_ID register contains version information for the subsystem.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System ID registers

##### Address offset

SID\_BASE + 0x0040

##### Type

RO

##### Reset value

0x0047\_77B7

## Bit descriptions

**Table 7-29: SID\_SYSTEM\_ID bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	VARIANT_NUMBER	Specifies the configuration code for the compute subsystem.	RO	0x0
[27:23]	RESERVED	Reserved	RAZ/WI	-
[22:12]	DESIGNER_ID	Specifies the identification code for the subsystem designer. 0x477 for Arm.	RO	0x477
[11:0]	PART_NUMBER	Specifies the part number for the subsystem.	RO	0x7B7

## 7.4.1.2 SID\_SOC\_ID, System ID SoC Identification register

The SID\_SOC\_ID register contains version information for the SoC that integrates the subsystem. The value of this register depends on the SID\_SOC\_ID signal.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

System ID registers

#### Address offset

SID\_BASE + 0x0050

#### Type

RO

#### Reset value

CFG\_DEF

## Bit descriptions

**Table 7-30: SID\_SOC\_ID bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	SOC_TREVISION	Soc Target Revision. The value is driven from the tie-off primary input port - SOC_TREVISION	RO	CFG_DEF
[27:12]	SOC_TPARTNO	SoC Target Part Number. The value is driven from the tie-off primary input port - SOC_TPARTNO	RO	CFG_DEF
[11:1]	SOC_DESIGNER_ID	SoC Designer ID, based on the 11-bit JEDEC JEP106 continuation and identity code. The value is driven from the tie-off primary input port - SOC_DESIGNER	RO	CFG_DEF
[0]	RESERVED	Reserved	RA1	0x1

### 7.4.1.3 SID\_CHIP\_ID, System ID Chip Identification register

The SID\_CHIP\_ID register contains information about the node when there are multiple sockets. The value of this register depends on the CHIPID and MULTI\_CHIP\_MODE signals.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System ID registers

##### Address offset

SID\_BASE + 0x0060

##### Type

RO

##### Reset value

CFG\_DEF

#### Bit descriptions

**Table 7-31: SID\_CHIP\_ID bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:24]	MAJOR REVISION	Specifies the major revision number for the compute subsystem.	RO	0x0
[23:20]	MINOR REVISION	Specifies the minor revision number for the compute subsystem.	RO	0x0
[19:9]	RESERVED	Reserved	RAZ/WI	-
[8]	MULTI_CHIP_MODE	Specifies the chip mode, as indicated by the input signals into the compute subsystem. <ul style="list-style-type: none"> <li>• 0 - single-chip mode</li> <li>• 1 - multi-chip mode</li> </ul>	RO	CFG_DEF
[7:0]	CHIP_ID	Specifies the CHIP_ID tie-off value in multi-chip mode. In single-chip mode, this value is 0.	RO	CFG_DEF

### 7.4.1.4 SID\_SYSTEM\_CFG, System ID System Configuration register

The SID\_SYSTEM\_CFG register contains information about the subsystem configuration when multiple variants that use the same compute subsystem are supported. For example, when two DDR channel and four DDR channel variants are supported. The value of this register depends on the SID\_SYSTEM\_CFG signal.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System ID registers

### Address offset

SID\_BASE + 0x0070

### Type

RO

### Reset value

CFG\_DEF

## Bit descriptions

**Table 7-32: SID\_SYSTEM\_CFG bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	SYSTEM_CONFIG_INFO	CFG_DEF. Dependent on the implementation.	RO	-

## 7.4.1.5 SID PID\_4, System ID Peripheral ID 4 register

The SID PID\_4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System ID registers

### Address offset

SID\_BASE + 0x0FD0

### Type

RO

### Reset value

0x00000004

## Bit descriptions

**Table 7-33: SID PID\_4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of continuation characters in the manufacturer identity code. 0x4 for Arm.	RO	0x4

### 7.4.1.6 SID PID\_0, System ID Peripheral ID 0 register

The SID PID\_0 register contains the first eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System ID registers](#)

##### Address offset

SID\_BASE + 0x0FE0

##### Type

RO

##### Reset value

0x000000BC

## Bit descriptions

**Table 7-34: SID PID\_0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xBC

### 7.4.1.7 SID PID\_1, System ID Peripheral ID 1 register

The SID PID\_1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System ID registers

### Address offset

SID\_BASE + 0x0FE4

### Type

RO

### Reset value

0x000000B0

## Bit descriptions

**Table 7-35: SID PID\_1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral. Set to 0xB.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x0

## 7.4.1.8 SID PID\_2, System ID Peripheral ID 2 register

The SID PID\_2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System ID registers

### Address offset

SID\_BASE + 0x0FE8

### Type

RO

### Reset value

0x0000000B

## Bit descriptions

**Table 7-36: SID PID\_2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x0
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral. Set to 0x3 for Arm.	RO	0b011

## 7.4.1.9 SID PID\_3, System ID Peripheral ID 3 register

The SID PID\_3 register provides information about any modifications to the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[System ID registers](#)

#### Address offset

SID\_BASE + 0x0FEC

#### Type

RO

#### Reset value

0x00000000

## Bit descriptions

**Table 7-37: SID PID\_3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.1.10 COMPID0, System ID Component ID 0 register

The COMPID0 register contains segment 0 of the preamble to the system ID component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System ID registers](#)

##### Address offset

SID\_BASE + 0x0FF0

##### Type

RO

##### Reset value

0x0000000D

#### Bit descriptions

**Table 7-38: COMPID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_0	Specifies segment 0 of the preamble to the code that identifies the system ID component class.	RO	0x0D

### 7.4.1.11 COMPID1, System ID Component ID 1 register

The COMPID1 register contains segment 1 of the preamble and the system ID component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System ID registers](#)

##### Address offset

SID\_BASE + 0x0FF4

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-39: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the system ID component class.	RO	0xF
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the system ID component class.	RO	0x0

**7.4.1.12 COMPID2, System ID Component ID 2 register**

The COMPID2 register contains segment 2 of the preamble to the system ID component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System ID registers](#)**Address offset**

SID\_BASE + 0xFF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-40: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the system ID component class.	RO	0x05

### 7.4.1.13 COMPID3, System ID Component ID 3 register

The COMPID3 register contains segment 3 of the preamble to the system ID component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System ID registers

##### Address offset

SID\_BASE + 0x0FFC

##### Type

RO

##### Reset value

0x000000B1

#### Bit descriptions

**Table 7-41: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the system ID component class.	RO	0xB1

### 7.4.2 REFCLK counter registers

The REFCLK counter registers provide access to and control of the counter.

The REFCLK counter is an implementation of the memory-mapped counter that the *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile* defines. This counter implements both the standard REFCLK CNTControl frame and the REFCLK CNTRead frame in the application processor memory map. The base address and accessibility of these frames are defined in [Memory maps](#).

The REFCLK counter implements four additional registers in the CNTControl frame that are not defined in the *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile*. The following table lists these extra registers and their address offsets in the CNTControl frame.

**Table 7-42: Implementation-specific REFCLK counter CNTControl register summary**

Offset	Name	Description	Type	Reset	Width
0x00C0	CNTENCR	Counter Enable Control.	RW	0x00	32-bit
0x00C4	Reserved	Reserved	RAZ/WI	-	32-bit
0x00C8	CNTSAMPVAL_L	Counter Sample Counter Lower Value.	RO	0x00	32-bit
0x00CC	CNTSAMPVAL_U	Counter Sample Counter Upper Value.	RO	0x00	32-bit
0x00D0	CNTINCR	Counter Increment Control.	RW	0x00	32-bit
0x0FD0	CNTControl PID_4	CNTControl Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	CNTControl PID_0	CNTControl Peripheral ID 0	RO	0x0000009C	32-bit
0x0FE4	CNTControl PID_1	CNTControl Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	CNTControl PID_2	CNTControl Peripheral ID 2	RO	0x0000001B	32-bit
0x0FEC	CNTControl PID_3	CNTControl Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	COMPID0	CNTControl Component ID 0	RO	0x0000000D	32-bit
0x0FF4	COMPID1	CNTControl Component ID 1	RO	0x000000F0	32-bit
0x0FF8	COMPID2	CNTControl Component ID 2	RO	0x00000005	32-bit
0x0FFC	COMPID3	CNTControl Component ID 3	RO	0x000000B1	32-bit

The following table lists the Peripheral ID and Component ID registers in the CNTReadBase frame.

**Table 7-43: Implementation-specific REFCLK counter CNTReadBase register summary**

Offset	Name	Description	Type	Reset	Width
0x0FD0	CNTReadBase PID_4	CNTReadBase Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	CNTReadBase PID_0	CNTReadBase Peripheral ID 0	RO	0x0000009D	32-bit
0x0FE4	CNTReadBase PID_1	CNTReadBase Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	CNTReadBase PID_2	CNTReadBase Peripheral ID 2	RO	0x0000000B	32-bit
0x0FEC	CNTReadBase PID_3	CNTReadBase Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	COMPID0	CNTReadBase Component ID 0	RO	0x0000000D	32-bit
0x0FF4	COMPID1	CNTReadBase Component ID 1	RO	0x000000F0	32-bit
0x0FF8	COMPID2	CNTReadBase Component ID 2	RO	0x00000005	32-bit
0x0FFC	COMPID3	CNTReadBase Component ID 3	RO	0x000000B1	32-bit

#### 7.4.2.1 CNTENCR, Counter Enable Control register

The CNTENCR register specifies how the REFCLK generic counter is enabled and disabled.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

**Functional group**

REFCLK counter registers

**Address offset**

0xC0

**Type**

RW

**Reset value**

0x00

**Bit descriptions****Table 7-44: CNTENCR bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	ENCTRL	Controls the operation of the Counter Control register EN bit. <ul style="list-style-type: none"> <li>When this bit is set to 0, the counter is enabled and disabled immediately.</li> <li>If the setting is 1, enabling of the counter is delayed until just after the next rising edge at the REFCLK. Disabling of the counter is not delayed.</li> </ul>	RW	-

**7.4.2.2 CNTSAMPVAL\_L, Counter Sample Counter Lower Value register**

The CNTSAMPVAL\_L register enables read back of the lower word of the counter value sampled on the rising of edge of the RFCLK.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

REFCLK counter registers

**Address offset**

0xC8

**Type**

RO

**Reset value**

0x00

## Bit descriptions

**Table 7-45: CNTSAMPVAL\_L bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CNTSAMPVAL_L	Displays bits[31:0] of the REFCLK-sampled counter value.	RO	0x00

### 7.4.2.3 CNTSAMPVAL\_U, Counter Sample Counter Upper Value register

The CNTSAMPVAL\_U register enables read back of the upper word of the counter value sampled on the rising of edge of the RFCLK.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[REFCLK counter registers](#)

##### Address offset

0xCC

##### Type

RO

##### Reset value

0x00

## Bit descriptions

**Table 7-46: CNTSAMPVAL\_U bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CNTSAMPVAL_U	Displays bits[63:32] of the REFCLK-sampled counter value.	RO	0x00

### 7.4.2.4 CNTINCR, Counter Increment Control register

The CNTINCR register specifies the increment step value for the counter.

This register is added to mimic incrementing of the counter at 1GHz, which is a requirement of the Arm® Server Base System Architecture, version 6.0. Depending on the REFCLK frequency, the step value must be programmed so that the counter appears to be incremented at a clock frequency of 1GHz.



- Arm® Server Base System Architecture, version 6.0 has a requirement SBSA\_CNT\_019 that states: 'A system compatible with level 5 will implement a generic counter which counts in nanosecond units. This means that, to the operating system the reported frequency will be 1GHz.'
- We recommend the following combinations of REFCLK and INCR\_VALUE to meet this requirement: (125MHz \* 8), (62.5MHz \* 16) or (31.25MHz \* 32).

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

REFCLK counter registers

### Address offset

0xD0

### Type

RW

### Reset value

0x00

## Bit descriptions

**Table 7-47: CNTINCR bit descriptions**

Bits	Name	Description	Type	Reset
[31:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	INCR_VALUE	Specifies the value that the counter increments with every clock tick. <ul style="list-style-type: none"> <li>The programmed increment value must be a power of two.</li> <li>For REFCLK, the increment must give an apparent counter update frequency of 1GHz.</li> </ul>	RW	-

## 7.4.2.5 CNTControl PID\_4, CNTControl Peripheral ID 4 register

The CNTControl PID\_4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[REFCLK counter registers](#)

### Address offset

0x0FDO

### Type

RO

### Reset value

0x00000004

## Bit descriptions

**Table 7-48: CNTControl PID\_4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code. 0x4 for Arm.	RO	0x4

## 7.4.2.6 CNTControl PID\_0, CNTControl Peripheral ID 0 register

The CNTControl PID\_0 register contains the first eight bits of the identifier for the peripheral.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[REFCLK counter registers](#)

### Address offset

0x0FE0

### Type

RO

### Reset value

0x0000009C

## Bit descriptions

**Table 7-49: CNTControl PID\_0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x9C

## 7.4.2.7 CNTControl PID\_1, CNTControl Peripheral ID 1 register

The CNTControl PID\_1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

REFCLK counter registers

#### Address offset

0x0FE4

#### Type

RO

#### Reset value

0x000000B0

## Bit descriptions

**Table 7-50: CNTControl PID\_1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral. Set to 0xB.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x0

## 7.4.2.8 CNTControl PID\_2, CNTControl Peripheral ID 2 register

The CNTControl PID\_2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

REFCLK counter registers

### Address offset

0x0FE8

### Type

RO

### Reset value

0x00000001B

## Bit descriptions

**Table 7-51: CNTControl PID\_2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x1
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral. Set to 0x3 for Arm.	RO	0b011

## 7.4.2.9 CNTControl PID\_3, CNTControl Peripheral ID 3 register

The CNTControl PID\_3 register provides information about any modifications to the peripheral.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

REFCLK counter registers

### Address offset

0x0FEC

### Type

RO

### Reset value

0x00000000

## Bit descriptions

**Table 7-52: CNTControl PID\_3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

## 7.4.2.10 COMPIDO, CNTControl Component ID 0 register

The COMPIDO register contains segment 0 of the preamble to the REFCLK counter CNTControl frame class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

REFCLK counter registers

#### Address offset

0x0FF0

#### Type

RO

#### Reset value

0x0000000D

## Bit descriptions

**Table 7-53: COMPIDO bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_O	Specifies segment 0 of the preamble to the code that identifies the REFCLK counter CNTControl frame class.	RO	0x0D

### 7.4.2.11 COMPID1, CNTControl Component ID 1 register

The COMPID1 register contains segment 1 of the preamble and the REFCLK counter CNTControl frame class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

REFCLK counter registers

##### Address offset

0x0FF4

##### Type

RO

##### Reset value

0x00000000

#### Bit descriptions

**Table 7-54: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the system ID component class.	RO	0xF
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the REFCLK counter CNTControl frame class.	RO	0x0

### 7.4.2.12 COMPID2, CNTControl Component ID 2 register

The COMPID2 register contains segment 2 of the preamble to the REFCLK counter CNTControl frame class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

REFCLK counter registers

**Address offset**

0x0FF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-55: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the REFCLK counter CNTControl frame component class.	RO	0x05

[7.4.2.13 COMPID3, CNTControl Component ID 3 register](#)

The COMPID3 register contains segment 3 of the preamble to the REFCLK counter CNTControl frame class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[REFCLK counter registers](#)**Address offset**

0x0FFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-56: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the REFCLK counter CNTControl frame component class.	RO	0xB1

### 7.4.2.14 CNTReadBase PID\_4, CNTReadBase Peripheral ID 4 register

The CNTReadBase PID\_4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

REFCLK counter registers

##### Address offset

0x0FD0

##### Type

RO

##### Reset value

0x0000\_0004

#### Bit descriptions

**Table 7-57: CNTReadBase PID\_4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of continuation characters in the manufacturer identity code. 0x4 for Arm.	RO	0x4

### 7.4.2.15 CNTReadBase PID\_0, CNTReadBase Peripheral ID 0 register

The CNTReadBase PID\_0 register contains the first eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

REFCLK counter registers

**Address offset**

0x0FE0

**Type**

RO

**Reset value**

0x0000009D

**Bit descriptions****Table 7-58: CNTReadBase PID\_0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x9D

**7.4.2.16 CNTReadBase PID\_1, CNTReadBase Peripheral ID 1 register**

The CNTReadBase PID\_1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[REFCLK counter registers](#)**Address offset**

0x0FE4

**Type**

RO

**Reset value**

0x000000B0

**Bit descriptions****Table 7-59: CNTReadBase PID\_1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral. Set to 0xB.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x0

### 7.4.2.17 CNTReadBase PID\_2, CNTReadBase Peripheral ID 2 register

The CNTReadBase PID\_2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

REFCLK counter registers

##### Address offset

0x0FE8

##### Type

RO

##### Reset value

0x0000000B

#### Bit descriptions

**Table 7-60: CNTReadBase PID\_2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x1
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral. Set to 0x3 for Arm.	RO	0b011

### 7.4.2.18 CNTReadBase PID\_3, CNTReadBase Peripheral ID 3 register

The CNTReadBase PID\_3 register provides information about any modifications to the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

REFCLK counter registers

**Address offset**

0x0FEC

**Type**

RO

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-61: CNTReadBase PID\_3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

**7.4.2.19 COMPIDO, CNTReadBase Component ID 0 register**

The COMPIDO register contains segment 0 of the preamble to the REFCLK Counter CNTReadBase frame class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[REFCLK counter registers](#)**Address offset**

0x0FF0

**Type**

RO

**Reset value**

0x0000000D

**Bit descriptions****Table 7-62: COMPIDO bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[7:0]	PRMBL_0	Specifies segment 0 of the preamble to the code that identifies the REFCLK Counter CNTReadBase frame class.	RO	0x0D

#### 7.4.2.20 COMPID1, CNTReadBase Component ID 1 register

The COMPID1 register contains segment 1 of the preamble and the REFCLK Counter CNTReadBase frame class identifier.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[REFCLK counter registers](#)

###### Address offset

0x0FF4

###### Type

RO

###### Reset value

0x00000000

##### Bit descriptions

**Table 7-63: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the system ID component class.	RO	0xF
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the REFCLK Counter CNTReadBase frame class.	RO	0x0

#### 7.4.2.21 COMPID2, CNTReadBase Component ID 2 register

The COMPID2 register contains segment 2 of the preamble to the REFCLK Counter CNTReadBase frame class identifier.

##### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

REFCLK counter registers

### Address offset

0x0FF8

### Type

RO

### Reset value

0x00000005

## Bit descriptions

**Table 7-64: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the REFCLK Counter CNTReadBase frame class.	RO	0x05

## 7.4.2.22 COMPID3, CNTReadBase Component ID 3 register

The COMPID3 register contains segment 3 of the preamble to the REFCLK Counter CNTReadBase frame class identifier.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

REFCLK counter registers

### Address offset

0x0FFC

### Type

RO

### Reset value

0x000000B1

## Bit descriptions

**Table 7-65: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the REFCLK Counter CNTReadBase frame class.	RO	0xB1

### 7.4.2.23 Counter scaling extension support

RD-N2 supports optional counter scaling for the system generic global counter, as introduced in Armv8.4. This support is optional and is **IMPLEMENTATION DEFINED**.

To support counter scaling, a scaling register is added to the memory-mapped counter module that is described in the *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile*. The register allows the frequency of the counter that is generated to be scaled from the basic frequency that is reported in the counter ID mechanisms. The effect of this scaling is visible in all values of the generic counter that is presented in the system.

In implementations with this feature, the counter count value is extended to 88 bits. The count is expressed as a fixed-point number with a 64-bit integer value and a 24-bit fractional value. The 64-bit integer value is read from the Counter Count Value register and this value is distributed as the generic count.

To support counter scaling, the following changes are made to registers at the CNTControlBase memory map:

- The [Counter Scale \(CNTSCR\) register](#) is added to the memory-mapped counter module as part of the CNTControlBase memory map to the *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile*.
- The [Counter Identification \(CNTID\) register](#) is added as part of the CNTControlBase memory map.
- The Scale Enable (SCEN) control bit is added to the [Counter Control \(CNTCR\) register](#).

When counter scaling is implemented, the generic counter must still be balanced across the system. As the counter increment value that is specified in the CNTSCR.ScaleVal field is programmable, distribution using Gray coding is not possible. The whole 88-bit or 64-bit binary value must be distributed as required. Balancing of the generic counter value is **IMPLEMENTATION DEFINED**.

### 7.4.2.23.1 CNTSCR, Counter Scale register

The CNTSCR register specifies a scale value that is added to the counter value for every counter tick. The scale value is added only when counter scaling is enabled.

#### Configurations

This register is present only when Armv8.4 counter scaling is implemented.

#### Attributes

##### Width

32-bit

##### Functional group

Generic timer memory-mapped registers

##### Address offset

0x10

##### Type

RW

##### Reset value

UNKNOWN

#### Bit descriptions

**Table 7-66: CNTSCR bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	ScaleVal	<p>When counter scaling is enabled, specifies a value to add to the counter value for every counter tick. The counter tick is determined by the inverse of the current operating frequency of the generic counter. The scale value is expressed as an unsigned fixed-point number with an 8-bit integer value and a 24-bit fractional value.</p> <p>The value of this field can only be changed when the system counter is disabled, that is, CNTCR.EN is set to 0. If the scale value is changed when the system counter is enabled, the counter value becomes UNKNOWN and remains UNKNOWN on future ticks of the clock.</p>	RW	UNKNOWN

### 7.4.2.23.2 CNTID, Counter Identification register

The CNTID register indicates whether counter scaling is implemented.

#### Configurations

This register is present only when Armv8.4 counter scaling is implemented.

#### Attributes

##### Width

32-bit

**Functional group**

Generic timer memory-mapped registers

**Address offset**

0x1C

**Type**

RO

**Bit descriptions****Table 7-67: CNTID bit descriptions**

Bits	Name	Description	Type	Reset
[31:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	CNTSC	<p>Shows whether counter scaling is implemented. A value of 0b0000 indicates that counter scaling is not implemented, while a setting of 0b0001 specifies that counter scaling is implemented. All other values are Reserved.</p> <p>When a value is written to the CNTCV register in the CNTControlBase frame of the memory-mapped counter module, the accumulated fraction information is reset to zero.</p>	RO	-

**7.4.2.23.3 CNTCR, Counter Control register**

The CNTCR register enables the counter and counter scaling, allows counter frequency change requests, and controls counter debug behavior.

**Configurations**

This register is present only when Armv8.4 counter scaling is implemented.

**Attributes****Width**

32-bit

**Functional group**

Generic timer memory-mapped registers

**Address offset**

0x000

**Type**

RW

**Reset value**

UNKNOWN

## Bit descriptions

**Table 7-68: CNTCR bit descriptions**

Bits	Name	Description	Type	Reset
[31:18]	RESERVED	Reserved	RAZ/WI	-
[17:8]	FCREQ	Specifies the entry number to select in the Frequency modes table for a counter frequency change. Selecting an unimplemented entry or an entry that contains 0 has no effect on the counter.  Implementations of this field are only required to contain values up to the maximum number of entries in the Frequency modes table. The maximum number of table entries is IMPLEMENTATION DEFINED, with an upper limit of 1004.	RW	0x0
[7:3]	RESERVED	Reserved	RAZ/WI	-
[2]	SCEN	When Armv8.4 counter scaling is implemented, enables and disables this function. A value of 0 indicates that counter scaling is not enabled and the counter value is incremented by 0x1.0000000 for each counter tick. When this field is set to 1, counter scaling is enabled and the counter is incremented by the value of the CNTSCR.ScaleVal field for each counter tick.  The value of this field can only be changed when the system counter is disabled, that is, CNTCR.EN is set to 0. If the value of this bit is changed when the system counter is enabled, the counter value becomes UNKNOWN and remains UNKNOWN on future ticks of the clock.	RW	UNKNOWN
[1]	HDBG	Specifies whether halt-on-debug signals halt the system counter. When this bit is set to 0, the system counter ignores halt-on-debug signals. If the setting is 1, system counter updates are halted when a halt-on-debug signal is asserted.	RW	UNKNOWN
[0]	EN	Enables and disables the system counter. A value of 0 indicates that the counter is disabled, while setting this field to 1 enables the counter.	RW	0b0

### 7.4.3 Generic timer registers

The generic timer registers provide access to and control of the timer.

The RD-N2 supports various timers. The timer base addresses are mapped in the memory maps for the application processors, System Control Processor, and Manageability Control Processor:

- The SCP REFCLK Generic Timer is mapped in the SCP memory map.
- The MCP REFCLK Generic Timer is mapped in the MCP memory map.
- Two additional REFCLK generic timers are mapped in the AP memory map. These two extra timers, one Secure and one Non-secure, are solely for general-purpose use by the application processors.

In addition, the system generic timer synchronization logic is mapped in the SCP memory map. This logic is not a timer. However, the registers that are related to the logic are used to synchronize the system generic count value across the chips.

The following table lists the implementation-specific generic timer registers for CNTCTLBase memory frame. See the *Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile - Memory-mapped Timer component* for the complete set of registers.

**Table 7-69: Implementation-specific Generic timer CNTCTLBase register summary**

Offset	Name	Description	Type	Reset	Width
0xFD0	CNTPIDR4	Timer Control Peripheral ID 4	RO	0x00000004	32-bit
0xFE0	CNTPIDR0	Timer Control Peripheral ID 0	RO	0x000000A0	32-bit
0xFE4	CNTPIDR1	Timer Control Peripheral ID 1	RO	0x000000B0	32-bit
0xFE8	CNTPIDR2	Timer Control Peripheral ID 2	RO	0x0000000B	32-bit
0xFEC	CNTPIDR3	Timer Control Peripheral ID 3	RO	0x00000000	32-bit
0xFF0	CNTCIDR0	Timer Control Component ID 0	RO	0x0000000D	32-bit
0xFF4	CNTCIDR1	Timer Control Component ID 1	RO	0x000000F0	32-bit
0xFF8	CNTCIDR2	Timer Control Component ID 2	RO	0x00000005	32-bit
0xFFC	CNTCIDR3	Timer Control Component ID 3	RO	0x000000B1	32-bit

The following table lists the implementation-specific generic timer registers for CNTBaseN memory frame. See the *Arm® Architecture Reference Manual Armv8*, for *Armv8-A architecture profile - Memory-mapped Timer component* for the complete set of registers.

**Table 7-70: Implementation-specific Generic timer CNTBaseN register summary**

Offset	Name	Description	Type	Reset	Width
0xFD0	CNTPIDR4	Timer Peripheral ID 4	RO	0x00000004	32-bit
0xFE0	CNTPIDR0	Timer Peripheral ID 0	RO	0x000000A2	32-bit
0xFE4	CNTPIDR1	Timer Peripheral ID 1	RO	0x000000B0	32-bit
0xFE8	CNTPIDR2	Timer Peripheral ID 2	RO	0x0000000B	32-bit
0xFEC	CNTPIDR3	Timer Peripheral ID 3	RO	0x00000000	32-bit
0xFF0	CNTCIDR0	Timer Component ID 0	RO	0x0000000D	32-bit
0xFF4	CNTCIDR1	Timer Component ID 1	RO	0x000000F0	32-bit
0xFF8	CNTCIDR2	Timer Component ID 2	RO	0x00000005	32-bit
0xFFC	CNTCIDR3	Timer Component ID 3	RO	0x000000B1	32-bit

### 7.4.3.1 CNTPIDR4, Timer Control Peripheral ID 4 register

The CNTPIDR4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Generic timer registers

**Address offset**

0xFD0

**Type**

RO

**Reset value**

0x00000004

**Bit descriptions****Table 7-71: CNTPIDR4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved.	RAZ/WI	0x0
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

**7.4.3.2 CNTPIDR0, Timer Control Peripheral ID 0 register**

The CNTPIDR0 register contains the first eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Generic timer registers](#)**Address offset**

0xFE0

**Type**

RO

**Reset value**

0x000000A0

**Bit descriptions****Table 7-72: CNTPIDR0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xA0

### 7.4.3.3 CNTPIDR1, Timer Control Peripheral ID 1 register

The CNTPIDR1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Generic timer registers

##### Address offset

0xFE4

##### Type

RO

##### Reset value

0x000000B0

#### Bit descriptions

**Table 7-73: CNTPIDR1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

### 7.4.3.4 CNTPIDR2, Timer Control Peripheral ID 2 register

The CNTPIDR2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**

Generic timer registers

**Address offset**

0xFE8

**Type**

RO

**Reset value**

0x0000000B

**Bit descriptions****Table 7-74: CNTPIDR2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block. The value is defined by the implementation.	RO	0x0
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	0x1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

**7.4.3.5 CNTPIDR3, Timer Control Peripheral ID 3 register**

The CNTPIDR3 register provides information about any modifications to the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Generic timer registers

**Address offset**

0xFEC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-75: CNTPIDR3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.3.6 CNTCIDR0, Timer Control Component ID 0 register

The CNTCIDR0 register contains segment 0 of the preamble to the Timer Control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Generic timer registers

##### Address offset

0xFF0

##### Type

RO

##### Reset value

0x0000000D

#### Bit descriptions

**Table 7-76: CNTCIDR0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_0	Specifies segment 0 of the preamble to the code that identifies the Timer Control component class.	RO	0x0D

### 7.4.3.7 CNTCIDR1, Timer Control Component ID 1 register

The CNTCIDR1 register contains segment 1 of the preamble and the Timer Control component class identifier.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Generic timer registers

### Address offset

0xFF4

### Type

RO

### Reset value

0x000000F0

## Bit descriptions

**Table 7-77: CNTC IDR1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the counter-timer component class	RO	0x0
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the Timer Control component class	RO	0xF0

## 7.4.3.8 CNTC IDR2, Timer Control Component ID 2 register

The CNTC IDR2 register contains segment 2 of the preamble to the Timer Control component class identifier.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Generic timer registers

### Address offset

0xFF8

### Type

RO

### Reset value

0x00000005

## Bit descriptions

**Table 7-78: CNTCIDR2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the Timer Control component class.	RO	0x05

## 7.4.3.9 CNTCIDR3, Timer Control Component ID 3 register

The CNTCIDR3 register contains segment 3 of the preamble to the Timer Control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Generic timer registers

#### Address offset

0xFFC

#### Type

RO

#### Reset value

0x000000B1

## Bit descriptions

**Table 7-79: CNTCIDR3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the Timer control component class.	RO	0xB1

## 7.4.3.10 CNTPIDR4, Timer Peripheral ID 4 register

The CNTPIDR4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Generic timer registers

### Address offset

0xFD0

### Type

RO

### Reset value

0x00000004

## Bit descriptions

**Table 7-80: CNTPIDR4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

## 7.4.3.11 CNTPIDR0, Timer Peripheral ID 0 register

The CNTPIDR0 register contains the first eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Generic timer registers

### Address offset

0xFE0

### Type

RO

### Reset value

0x000000A2

## Bit descriptions

**Table 7-81: CNTPIDR0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xA2

### 7.4.3.12 CNTPIDR1, Timer Peripheral ID 1 register

The CNTPIDR1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Generic timer registers

### Address offset

0xFE4

### Type

RO

### Reset value

0x000000B0

## Bit descriptions

**Table 7-82: CNTPIDR1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

### 7.4.3.13 CNTPIDR2, Timer Peripheral ID 2 register

The CNTPIDR2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Generic timer registers

##### Address offset

0xFE8

##### Type

RO

##### Reset value

0x0000000B

#### Bit descriptions

**Table 7-83: CNTPIDR2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block. The value is defined by the implementation.	RO	0x0
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	0x1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

### 7.4.3.14 CNTPIDR3, Timer Peripheral ID 3 register

The CNTPIDR3 register provides information about any modifications to the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Generic timer registers

**Address offset**

0xFEC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-84: CNTPIDR3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

**7.4.3.15 CNTCIDR0, Timer Component ID 0 register**

The CNTCIDR0 register contains segment 0 of the preamble to the Timer component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Generic timer registers](#)**Address offset**

0xFF0

**Type**

RO

**Reset value**

0x0000000D

**Bit descriptions****Table 7-85: CNTCIDR0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_0	Specifies segment 0 of the preamble to the code that identifies the Timer component class.	RO	0x0D

### 7.4.3.16 CNTCIDR1, Timer Component ID 1 register

The CNTCIDR1 register contains segment 1 of the preamble and the Timer component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Generic timer registers](#)

##### Address offset

0xFF4

##### Type

RO

##### Reset value

0x000000F0

#### Bit descriptions

**Table 7-86: CNTCIDR1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the counter-timer component class	RO	0x0
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the Timer component class	RO	0xF0

### 7.4.3.17 CNTCIDR2, Timer Component ID 2 register

The CNTCIDR2 register contains segment 2 of the preamble to the Timer component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Generic timer registers](#)

**Address offset**

0xFF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-87: CNTCIDR2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the Timer component class.	RO	0x05

[7.4.3.18 CNTCIDR3, Timer Component ID 3 register](#)

The CNTCIDR3 register contains segment 3 of the preamble to the Timer component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Generic timer registers](#)**Address offset**

0xFFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-88: CNTCIDR3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the Timer component class.	RO	0xB1

## 7.4.4 System generic timer synchronization registers

The system generic timer synchronization registers allow balancing of system generic count values across multichip systems.

Depending on the number of chips in the system, there are multiple instances of the registers. For example, a four-chip system contains three instances of these registers.

**Table 7-89: System generic timer synchronization register summary**

Offset	Name	Description	Type	Reset	Width
0x0000	MST_GCNT_SYNC_CTRL	Master Generic Counter Synchronization Control	RW	0x0	32-bit
0x0004	SLVCHIP_GCNT_SYNC_CTRL	Slave Generic Counter Synchronization Control	RW	0x0	32-bit
0x0008	SLVCHIP_GCNT_OFFSET_THRESHOLD	Slave Generic Counter Offset Threshold	RW	0x0	32-bit
0x000C	SLVCHIP_GCNT_INT_STATUS	Slave Generic Counter Interrupt Status	RW1C	0x0	32-bit
0x0010	GCNT_TIMEOUT	Generic Counter Timeout	RW	0x0	32-bit
0x0014	SLVCHIP_GCNT_SYNC_INTERVAL	Slave Generic Counter Synchronization Interval	RW	0x0	32-bit
0x0018	SLVCHIP_GCNT_UPDT_VAL_L	Slave Generic Counter Update Time Lower Value	R	0x0	32-bit
0x001C	SLVCHIP_GCNT_UPDT_VAL_U	Slave Generic Counter Update Time Upper Value	R	0x0	32-bit
0x0020	GCNT_SYNC_STATUS	Generic Counter Synchronization Status	RW	0x0	32-bit
0x0024	SLVCHIP_GCNT_NW_DLY	Slave Generic Counter Network Delay	RW	0x0	32-bit
0x0028	SLVCHIP_GCNT_RETRY_CNT	Slave Generic Counter Retry Count	RW	0x0	32-bit
0x002C	SLVCHIP_GCNT_MSTS LV_DIFF_L	Slave Generic Counter Time value difference (Lower 32 bits)	R	0x0	32-bit
0x0030	SLVCHIP_GCNT_MSTS LV_DIFF_U	Slave Generic Counter Time value difference (Upper 32 bits)	R	0x0	32-bit
0x0FD0	PID_4	Generic Counter Synchronization Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PID_0	Generic Counter Synchronization Peripheral ID 0	RO	0x00000098	32-bit
0x0FE4	PID_1	Generic Counter Synchronization Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID_2	Generic Counter Synchronization Peripheral ID 2	RO	0x0000001B	32-bit
0x0FEC	PID_3	Generic Counter Synchronization Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	COMP_ID0	Generic Counter Synchronization Component ID 0	RO	0x0000000D	32-bit
0x0FF4	COMP_ID1	Generic Counter Synchronization Component ID 1	RO	0x000000F0	32-bit
0x0FF8	COMP_ID2	Generic Counter Synchronization Component ID 2	RO	0x00000005	32-bit
0x0FFC	COMP_ID3	Generic Counter Synchronization Component ID 3	RO	0x000000B1	32-bit

### 7.4.4.1 MST\_GCNT\_SYNC\_CTRL, Master Generic Counter Synchronization Control register

Master Generic Counter Synchronization Control.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System generic timer synchronization registers

### Address offset

0x0000

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-90: MST\_GCNT\_SYNC\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:7]	RESERVED	Reserved	RAZ/WI	-
[5:0]	MSTUPDT_ADDR_OFFSET	Indicates which slave chip this master instance is communicating with. This will be used to offset the SYNCNT_MSTUPDT_ADDR that is allocated in chip0 address space. The address used to send the Tupdate value is MSTUPDT_ADDR_OFFSET * 0x400_0000_0000 + SYNCNT_MSTUPDT_ADDR.	RW	-
[6:1]	TURN_AR_TIME	Number of clocks to wait between back to back start of the transactions as noted in the diagram Figure 6-7 Shows the sync process between master and slave for both pass and fail cases master	RW	-
[0]	EN	Enable sync process.	RW	

## 7.4.4.2 SLVCHIP\_GCNT\_SYNC\_CTRL, Slave Generic Counter Synchronization Control register

Slave Generic Counter Synchronization Control.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System generic timer synchronization registers

### Address offset

0x0004

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-91: SLVCHIP\_GCNT\_SYNC\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:7]	RESERVED	Reserved	RAZ/WI	-
[6:2]	TURN_AR_TIME	Number of clocks to wait between back to back start of the transactions for retries	RW	-
[1]	EN_SYNC_IMM	Enables immediate synchronization process.  This starts the sync process immediately. Software can use this instead of continuous mode.	RW	-
[0]	EN	Enable sync process.	RW	-

**7.4.4.3 SLVCHIP\_GCNT\_OFFSET\_THRESHOLD, Slave Generic Counter Offset Threshold register**

Slave Generic Counter Offset Threshold register.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System generic timer synchronization registers](#)**Address offset**

0x0008

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-92: SLVCHIP\_GCNT\_OFFSET\_THRESHOLD bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	THRSHLD_VAL	Threshold value to be compared.	RW	-

#### 7.4.4.4 SLVCHIP\_GCNT\_INT\_STATUS, Slave Generic Counter Interrupt Status register

Slave Generic Counter Interrupt Status.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

System generic timer synchronization registers

###### Address offset

0x000C

###### Type

RW1C

###### Reset value

0x0

##### Bit descriptions

**Table 7-93: SLVCHIP\_GCNT\_INT\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:5]	RESERVED	Reserved	RAZ/WI	-
[4]	Threshold failed interrupt	The time difference between Master and slave is greater than the programmed threshold values.	RW1C	-
[3]	Time out interrupt	Time out interrupt when Master does not respond to sync request.	RW1C	-
[2]	Sync failed interrupt status	Sync failed when software request by setting EN_SYNC_IMM register field.	RW1C	-
[1]	Sync passed interrupt status	Sync Passed when software request by setting EN_SYNC_IMM register field.	RW1C	-
[0]	Sync failed after number of retries specified	Sync failed after n-number of retries.	RW1C	-

#### 7.4.4.5 GCNT\_TIMEOUT, Generic Counter Timeout register

Used for phase of the synchronization process.

##### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System generic timer synchronization registers

### Address offset

0x00010

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-94: GCNT\_TIMEOUT bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	TIMEOUT_VAL	Synchronization timeout value.	RW	-

## 7.4.4.6 SLVCHIP\_GCNT\_SYNC\_INTERVAL, Slave Generic Counter Synchronization Interval register

Time interval to perform synchronization process.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System generic timer synchronization registers

### Address offset

0x00014

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-95: SLVCHIP\_GCNT\_SYNC\_INTERVAL register bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	SYNC_INTRVL	Interval time to start the next sync process	RW	-

### 7.4.4.7 SLVCHIP\_GCNT\_UPDT\_VAL\_L, Slave Generic Counter Update Lower Value register

Time value bits[31:0] sent from the Master Counter.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

##### Address offset

0x0018

##### Type

R

##### Reset value

0x0

## Bit descriptions

**Table 7-96: SLVCHIP\_GCNT\_UPDT\_VAL\_L bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	TUPDT_L	Lower 32 bits of Tupdate value	R	-

### 7.4.4.8 SLVCHIP\_GCNT\_UPDT\_VAL\_U, Slave Generic Counter Update Upper Value register

Time value bits[63:32] sent from the Master Counter.

#### Configurations

This register is available in all configurations.

## Attributes

**Width**

32-bit

**Functional group**

System generic timer synchronization registers

**Address offset**

0x001C

**Type**

R

**Reset value**

0x0

## Bit descriptions

**Table 7-97: SLVCHIP\_GCNT\_UPDT\_VAL\_U bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	TUPDT_U	Upper 32 bits of Tupdate value	R	-

## 7.4.4.9 GCNT\_SYNC\_STATUS, Generic Counter Synchronization Status register

Status of Current Sync process.

## Configurations

This register is available in all configurations.

## Attributes

**Width**

32-bit

**Functional group**

System generic timer synchronization registers

**Address offset**

0x0020

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-98: GCNT\_SYNC\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:4]	RESERVED	Reserved	RAZ/WI	-
[3:2]	prev_sync_status	<p>Indicates the status of current synchronization process.</p> <ul style="list-style-type: none"> <li>0b00: No sync process completed</li> <li>0b01: Previous sync process was successful</li> <li>0b10: Error in the previous sync process</li> <li>0b11: Reserved</li> </ul>	RW	-
[1:0]	cur_sync_status	<p>Indicates the status of current synchronization process.</p> <ul style="list-style-type: none"> <li>0b00: First sync sequence is not started</li> <li>0b01: Sync in progress</li> <li>0b10: Idle period between sync processes</li> <li>0b11: Reserved</li> </ul>	RW	-

## 7.4.4.10 SLVCHIP\_GCNT\_NW\_DLY, Slave Generic Counter Network Delay register

Network delay through CCIX from master to slave.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

System generic timer synchronization registers

#### Address offset

0x0024

#### Type

RW

#### Reset value

0x0

## Bit descriptions

**Table 7-99: SLVCHIP\_GCNT\_NW\_DLY bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	NW_DLY	Network delay value between master and slave using CCIX	RW	-

### 7.4.4.11 SLVCHIP\_GCNT\_RETRY\_CNT, Slave Generic Counter Retry Count register

Retry count if the update is not successful if sync process failed due to slave not receiving the Tupdate value in time.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

##### Address offset

0x0028

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-100: SLVCHIP\_GCNT\_RETRY\_CNT bit descriptions**

Bits	Name	Description	Type	Reset
[31:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	RETRY_COUNT	Retries before raising the interrupt and go into stall mode	RW	-

### 7.4.4.12 SLVCHIP\_GCNT\_MSTSLV\_DIFF\_L, Slave Generic Counter Time value difference (Lower 32 bits)

Lower 32 bits of difference between Tupdate and Tslave when synchronization is failed due to being higher than the programmed threshold.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

**Address offset**

0x002C

**Type**

R

**Reset value**

0x0

**Bit descriptions****Table 7-101: SLVCHIP\_GCNT\_MSTSLV\_DIFF\_L bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	MSTSLV_DIFF_VALU	Lower 32 bits of difference between Tupdate and Tslave when synchronization is failed due to being higher than the programmed threshold.	R	-

**7.4.4.13 SLVCHIP\_GCNT\_MSTSLV\_DIFF\_U, Slave Generic Counter Time value difference (Upper 32 bits)**

Upper 32 bits of difference between Tupdate and Tslave when synchronization is failed due to being higher than the programmed threshold.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System generic timer synchronization registers](#)**Address offset**

0x0030

**Type**

R

**Reset value**

0x0

**Bit descriptions****Table 7-102: SLVCHIP\_GCNT\_MSTSLV\_DIFF\_U bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	TUPDT_U	Upper 32 bits of difference between Tupdate and Tslave when synchronization is failed due to being higher than the programmed threshold.	R	-

### 7.4.4.14 PID\_4, Generic Counter Synchronization Peripheral ID 4 register

The PID\_4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

##### Address offset

0x0FD0

##### Type

RO

##### Reset value

0x00000004

#### Bit descriptions

**Table 7-103: PID\_4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	-
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	-

### 7.4.4.15 PID\_0, Generic Counter Synchronization Peripheral ID 0 register

The PID\_0 register contains the first eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

**Address offset**

0x0FE0

**Type**

RO

**Reset value**

0x00000098

**Bit descriptions****Table 7-104: PID\_0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. Set to 0x98.	RO	-

[7.4.4.16 PID\\_1, Generic Counter Synchronization Peripheral ID 1 register](#)

The PID\_1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System generic timer synchronization registers](#)**Address offset**

0x0FE4

**Type**

RO

**Reset value**

0x000000B0

**Bit descriptions****Table 7-105: PID\_1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral. Set to 0xB for Arm.	RO	-
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral. Set to 0x0.	RO	-

### 7.4.4.17 PID\_2, Generic Counter Synchronization Peripheral ID 2 register

The CNTPIDR2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

##### Address offset

0x0FE8

##### Type

RO

##### Reset value

0x0000001B

#### Bit descriptions

**Table 7-106: PID\_2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block. Set to 0x0 for rOp0.	RO	-
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	-
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral. Set to 0x3 for Arm.	RO	-

### 7.4.4.18 PID\_3, Generic Counter Synchronization Peripheral ID 3 register

The PID\_3 register is Reserved.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

**Address offset**

0x0FEC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-107: PID\_3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	RESERVED	Reserved	RAZ/WI	-

**7.4.4.19 COMP\_ID0, Generic Counter Synchronization Component ID 0 register**

The COMP\_ID0 register contains segment 0 of the generic counter synchronization component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System generic timer synchronization registers](#)**Address offset**

0x0FF0

**Type**

RO

**Reset value**

0x0000000D

**Bit descriptions****Table 7-108: COMP\_ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/ WI	-
[7:0]	COMP_ID0	Specifies segment 0 of the code that identifies the generic counter synchronization component class. Reads as 0x0D.	RO	-

### 7.4.4.20 COMP\_ID1, Generic Counter Synchronization Component ID 1 register

The COMP\_ID1 register contains segment 1 of the generic counter synchronization component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

##### Address offset

0x0FF4

##### Type

RO

##### Reset value

0x000000F0

#### Bit descriptions

**Table 7-109: COMP\_ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID1	Specifies segment 1 of the code that identifies the generic counter synchronization component class. Reads as 0xF0.	RO	-

### 7.4.4.21 COMP\_ID2, Generic Counter Synchronization Component ID 2 register

The COMP\_ID2 register contains segment 2 of the generic counter synchronization component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

System generic timer synchronization registers

**Address offset**

0x0FF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-110: COMP\_ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID2	Specifies segment 2 of the code that identifies the generic counter synchronization component class. Reads as 0x05.	RO	-

**7.4.4.22 COMP\_ID3, Generic Counter Synchronization Component ID 3 register**

The COMP\_ID3 register contains segment 3 of the generic counter synchronization component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

System generic timer synchronization registers

**Address offset**

0x0FFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-111: COMP\_ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[7:0]	COMP_ID3	Specifies segment 3 of the code that identifies the generic counter synchronization component class. Reads as 0xB1.	RO	-

## 7.4.5 Application Processor Watchdog timer registers

The AP Watchdog timer registers provide identification information.

The AP Watchdog timer is an implementation of the memory-mapped timer that is defined by the Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile. This timer is compliant with the Arm® Server Base System Architecture, version 6.0. For more information about the registers, see the Arm® CoreSight™ Base System Architecture, version 1.0 Platform Design Document.

AP Watchdog timer has two memory frames - Watchdog Control and Watchdog Refresh. The following table lists the implementation specific registers both these frames.

**Table 7-112: AP Watchdog Control implementation-specific register summary**

Offset	Name	Description	Type	Reset	Width
0x0FCC	IIDR	AP Watchdog Control Interface Identification Register	RO	0x0001243B	32-bit
0x0FD0	PID4	AP Watchdog Control Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PID0	AP Watchdog Control Peripheral ID 0	RO	0x000000B1	32-bit
0x0FE4	PID1	AP Watchdog Control Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID2	AP Watchdog Control Peripheral ID 2	RO	0x0000002B	32-bit
0x0FEC	PID3	AP Watchdog Control Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	COMPID0	AP Watchdog Control Component ID 0 Register	RO	0x0000000D	32-bit
0x0FF4	COMPID1	AP Watchdog Control Component ID 1 Register	RO	0x000000F0	32-bit
0x0FF8	COMPID2	AP Watchdog Control Component ID 2 Register	RO	0x00000005	32-bit
0x0FFC	COMPID3	AP Watchdog Control Component ID 3 Register	RO	0x000000B1	32-bit

**Table 7-113: AP Watchdog Refresh implementation-specific register summary**

Offset	Name	Description	Type	Reset	Width
0x0FCC	IIDR	AP Watchdog Refresh Interface Identification Register	RO	0x0001243B	32-bit
0x0FD0	PID4	AP Watchdog Refresh Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PID0	AP Watchdog Refresh Peripheral ID 0	RO	0x000000B0	32-bit
0x0FE4	PID1	AP Watchdog Refresh Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID2	AP Watchdog Refresh Peripheral ID 2	RO	0x0000002B	32-bit
0x0FEC	PID3	AP Watchdog Refresh Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	COMPID0	AP Watchdog Refresh Component ID 0 Register	RO	0x0000000D	32-bit
0x0FF4	COMPID1	AP Watchdog Refresh Component ID 1 Register	RO	0x000000F0	32-bit
0x0FF8	COMPID2	AP Watchdog Refresh Component ID 2 Register	RO	0x00000005	32-bit
0x0FFC	COMPID3	AP Watchdog Refresh Component ID 3 Register	RO	0x000000B1	32-bit

### 7.4.5.1 IIDR, AP Watchdog Control Interface Identification Register

The IIDR is a read-only identification register that the Generic Watchdog architecture requires.

A register that is loaded post reset drives the identification fields. This approach ensures that the logic is preserved and that Engineering Change Orders (ECOs) can be applied, if necessary.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Application Processor Watchdog timer registers](#)

##### Address offset

0x0FCC

##### Type

RO

##### Reset value

0x00012343B

#### Bit descriptions

**Table 7-114: IIDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	RESERVED	Reserved	RAZ/WI	-
[23:20]	RESERVED	Reserved	RAZ/WI	-
[19:16]	Architecture version, v1	Specifies an architecture-defined value. Set to 0x1.	RO	0x1
[15:12]	An implementation defined revision number for the component	Specifies the version number for an Arm® Server Base System Architecture, version 6.0 compliant revision. Set to 0x2.	RO	0x2
[11:0]	Implementer JEP106 code	Specifies the JEDEC JEP106 implementer identification code for the AP Watchdog timer.	RO	0x43B

### 7.4.5.2 PID4, AP Watchdog Control Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FDO

### Type

RO

### Reset value

0x00000004

## Bit descriptions

**Table 7-115: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2. Set to 0x0.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

## 7.4.5.3 PID0, AP Watchdog Control Peripheral ID 0 register

The PID0 register contains the first eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FE0

### Type

RO

### Reset value

0x000000B1

## Bit descriptions

**Table 7-116: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral.	RO	0xB1

## 7.4.5.4 PID1, AP Watchdog Control Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Application Processor Watchdog timer registers

#### Address offset

0x0FE4

#### Type

RO

#### Reset value

0x000000B0

## Bit descriptions

**Table 7-117: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x0

## 7.4.5.5 PID2, AP Watchdog Control Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FE8

### Type

RO

### Reset value

0x0000002B

## Bit descriptions

**Table 7-118: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x2
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

## 7.4.5.6 PID3, AP Watchdog Control Peripheral ID 3 register

The PID3 register provides information about any modifications to the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FEC

### Type

RO

### Reset value

0x00000000

## Bit descriptions

**Table 7-119: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

## 7.4.5.7 COMPIDO, AP Watchdog Control Component ID 0 register

The COMPIDO register contains segment 0 of the preamble to the AP Watchdog Control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Application Processor Watchdog timer registers](#)

#### Address offset

0x0FF0

#### Type

RO

#### Reset value

0x0000000D

## Bit descriptions

**Table 7-120: COMPIDO bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_O	Specifies segment 0 of the preamble to the code that identifies the AP Watchdog Control component class.	RO	0x0D

### 7.4.5.8 COMPID1, AP Watchdog Control Component ID 1 register

The COMPID1 register contains segment 1 of the preamble and the AP Watchdog Control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Application Processor Watchdog timer registers

##### Address offset

0x0FF4

##### Type

RO

##### Reset value

0x000000F0

#### Bit descriptions

**Table 7-121: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the AP Watchdog Control component class	RO	-
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the AP Watchdog Control component class	RO	0xF0

### 7.4.5.9 COMPID2, AP Watchdog Control Component ID 2 register

The COMPID2 register contains segment 2 of the preamble to the AP Watchdog Control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Application Processor Watchdog timer registers

**Address offset**

0x0FF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-122: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the AP Watchdog Control component class.	RO	0x05

**7.4.5.10 COMPID3, AP Watchdog Control Component ID 3 register**

The COMPID3 register contains segment 3 of the preamble to the AP Watchdog Control component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Application Processor Watchdog timer registers](#)**Address offset**

0x0FFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-123: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the AP Watchdog Control component class.	RO	0xB1

### 7.4.5.11 IIDR, AP Watchdog Refresh Interface Identification Register

The IIDR is a read-only identification register that the Generic Watchdog architecture requires.

A register that is loaded post reset drives the identification fields. This approach ensures that the logic is preserved and that Engineering Change Orders (ECOs) can be applied, if necessary.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Application Processor Watchdog timer registers](#)

##### Address offset

0x0FCC

##### Type

RO

##### Reset value

0x00012343B

#### Bit descriptions

**Table 7-124: IIDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	RESERVED	Reserved	RAZ/WI	-
[23:20]	RESERVED	Reserved	RAZ/WI	-
[19:16]	Architecture version, v1	Specifies an architecture-defined value. Set to 0x1.	RO	0x1
[15:12]	An implementation defined revision number for the component	Specifies the version number for a revision compliant with Arm® Server Base System Architecture, version 6.0. Set to 0x2.	RO	0x2
[11:0]	Implementer JEP106 code	Specifies the JEDEC JEP106 implementer identification code for the AP Watchdog timer.	RO	0x43B

### 7.4.5.12 PID4, AP Watchdog Refresh Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FDO

### Type

RO

### Reset value

0x00000004

## Bit descriptions

**Table 7-125: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2. Set to 0x0.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

## 7.4.5.13 PID0, AP Watchdog Refresh Peripheral ID 0 register

The PID0 register contains the first eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FE0

### Type

RO

### Reset value

0x000000B0

## Bit descriptions

**Table 7-126: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral.	RO	0xB0

## 7.4.5.14 PID1, AP Watchdog Refresh Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Application Processor Watchdog timer registers

#### Address offset

0x0FE4

#### Type

RO

#### Reset value

0x000000B0

## Bit descriptions

**Table 7-127: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x0

## 7.4.5.15 PID2, AP Watchdog Refresh Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FE8

### Type

RO

### Reset value

0x0000002B

## Bit descriptions

**Table 7-128: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x2
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

## 7.4.5.16 PID3, AP Watchdog Refresh Peripheral ID 3 register

The PID3 register provides information about any modifications to the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Application Processor Watchdog timer registers

### Address offset

0x0FEC

### Type

RO

### Reset value

0x00000000

## Bit descriptions

**Table 7-129: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

## 7.4.5.17 COMPIDO, AP Watchdog Refresh Component ID 0 register

The COMPIDO register contains segment 0 of the preamble to the AP Watchdog Refresh component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Application Processor Watchdog timer registers](#)

#### Address offset

0x0FF0

#### Type

RO

#### Reset value

0x0000000D

## Bit descriptions

**Table 7-130: COMPIDO bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_O	Specifies segment 0 of the preamble to the code that identifies the AP Watchdog Refresh component class.	RO	0x0D

### 7.4.5.18 COMPID1, AP Watchdog Refresh Component ID 1 register

The COMPID1 register contains segment 1 of the preamble and the AP Watchdog Refresh component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Application Processor Watchdog timer registers

##### Address offset

0x0FF4

##### Type

RO

##### Reset value

0x000000F0

#### Bit descriptions

**Table 7-131: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the AP Watchdog Refresh component class	RO	-
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the AP Watchdog Refresh component class	RO	0xF0

### 7.4.5.19 COMPID2, AP Watchdog Refresh Component ID 2 register

The COMPID2 register contains segment 2 of the preamble to the AP Watchdog Refresh component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Application Processor Watchdog timer registers

**Address offset**

0x0FF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-132: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the AP Watchdog Refresh component class.	RO	0x05

**7.4.5.20 COMPID3, AP Watchdog Refresh Component ID 3 register**

The COMPID3 register contains segment 3 of the preamble to the AP Watchdog Refresh component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Application Processor Watchdog timer registers](#)**Address offset**

0x0FFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-133: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the AP Watchdog Refresh component class.	RO	0xB1

## 7.4.6 Base SRAM ECC RAS registers

The following table summarizes the error record registers that captures the ECC error status for Secure and Non-secure RAMs.

There are six frames of these register banks.

- AP Secure RAM ECC RAS registers
- AP Non-secure RAM ECC RAS registers
- SCP Secure RAM ECC RAS registers
- SCP Non-secure RAM ECC RAS registers
- MCP Secure RAM ECC RAS registers
- MCP Non-secure RAM ECC RAS registers

**Table 7-134: Base SRAM ECC RAS register summary**

Offset	Name	Description	Type	Reset	Width
0x000	RAMECC_ERRSTATUS	Secure SRAM error status	RW	0x00000000	32-bit
0x004	RAMECC_ERRCTRL	Secure SRAM error mask	RW	0x00000000	32-bit
0x008	RAMECC_ERRCODE	Secure SRAM error code	RO	0x00000000	32-bit
0x00C	RAMECC_ERRADDR	Secure SRAM error address	RO	0x00000000	32-bit
0x0FD0	PID4	Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PIDO	Peripheral ID 0	RO	0x000000BE	32-bit
0x0FE4	PID1	Peripheral ID 1	RO	0x000000B7	32-bit
0x0FE8	PID2	Peripheral ID 2	RO	0x0000000B	32-bit
0x0FEC	PID3	Peripheral ID 3	RO	0x00000000	32-bit
0xFF0	COMPID0	Component ID 0 Register	RO	0x0000000D	32-bit
0xFF4	COMPID1	Component ID 1 Register	RO	0x000000F0	32-bit
0xFF8	COMPID2	Component ID 2 Register	RO	0x00000005	32-bit
0xFFC	COMPID3	Component ID 3 Register	RO	0x000000B1	32-bit

### 7.4.6.1 ECC\_ERRSTATUS, Secure SRAM error status

This register captures the error status of Secure SRAM.

#### Configurations

This register is available in all configurations.

#### Attributes

#### Width

32-bit

**Functional group**

Base SRAM ECC RAS registers

**Address offset**

0x000

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-135: ECC\_ERRSTATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	OF	Multibit error occurred. Write a 1 to clear this bit and clear the MULTIBIT TYPE also.	RW	-
[1]	UE	Uncorrectable and uncontainable error have occurred. Write a 1 to clear this bit and clear the ERRCODE also.	RW	-
[0]	CE	Correctable error has occurred. Write a 1 to clear this bit and clear the ERRCODE field also.	Rw	-

**7.4.6.2 ECC\_ERRCTRL, Secure SRAM error mask**

This control register enables ECC checking and error injection on Secure SRAMs. The error mask bits in the register can be programmed to enable/disable interrupt generation for each type of errors.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Base SRAM ECC RAS registers

**Address offset**

0x004

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-136: ECC\_ERRCTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/ WI	-
[7]	SLVERR_CTRL	0 - Send OKAY read response back to the source on UE. 1 - Send SLVERR back to the source on UE in addition to poisoning the read data.	RW	-
[6:5]	INJECT_ERROR	Od00 - Do not inject any error Od01 - Inject a correctable error on the RAM read data of next read transaction that comes after this bit is set.  Od10 - Inject a Uncorrectable error on the RAM read data of next read transaction that comes after this bit is set.  Od11 - Reserved.  Follow the same error checking and recording the error.  Reset these bits after the injection of the error.  If this bit is set before the error record is cleared it will generate a OF error.	RW	-
[4]	OF_MASK	0 - Generate an interrupt when a OF error is seen. 1 - Do not generate an interrupt when a OF error is seen	RW	-
[3]	UE_MASK	0 - Generate an interrupt when a UE occurs. 1 - Do not generate an interrupt when a EE error occurs	RW	-
[2]	CE_MASK	Multi-bit error occurred. Write a 1 to clear this bit and clear the MULTIBIT TYPE also.	RW	-
[1]	RESERVED	Reserved	RAZ/ WI	-
[0]	RAM_ECC_EN	0 - Disable ECC checking. 1 - Enable ECC checking.  This enables checking and sending response for respective core.	RW	-

### 7.4.6.3 RAMECC\_ERRCODE, Secure SRAM error code

This register captures the error code of Secure SRAM errors.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Base SRAM ECC RAS registers

**Address offset**

0x008

**Type**

RO

**Reset value**

0x0

**Bit descriptions****Table 7-137: ECC\_ERRCODE bit descriptions**

Bits	Name	Description	Type	Reset
[31:5]	RESERVED	Reserved	RAZ/WI	-
[4:3]	MULTIBIT_TYPE	Type of the last multibit error type <ul style="list-style-type: none"><li>• 0x00 – No error</li><li>• 0x01 – Last multibit error is Correctable error</li><li>• 0x10 – Last multibit error is Uncorrectable error</li></ul>	RO	-
[2:0]	ERRCODE	Error code <ul style="list-style-type: none"><li>• 000 – No Error</li><li>• 001 – Error occurred on Read data</li><li>• 010 – Error received on Write data(Partial writes)</li><li>• 011 – Error occurred through poison on the write data</li></ul>	RO	-

**7.4.6.4 RAMECC\_ERRADDR, Secure SRAM error address**

This register captures the address of the Secure SRAM location where the error occurred.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Base SRAM ECC RAS registers](#)**Address offset**

0x00C

**Type**

RO

**Reset value**

0x0

## Bit descriptions

**Table 7-138: ECC\_ERRCODE bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	ERRORADDR	Address of the RAM location that error occurred	RO	-

## 7.4.6.5 PID4, Base SRAM ECC Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Base SRAM ECC RAS registers](#)

#### Address offset

0x0FD0

#### Type

RO

#### Reset value

0x00000004

## Bit descriptions

**Table 7-139: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2. Set to 0x0.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

## 7.4.6.6 PID0, Base SRAM ECC Peripheral ID 0 register

The PID0 register contains the first eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Base SRAM ECC RAS registers

### Address offset

0x0FE0

### Type

RO

### Reset value

0x000000BE

## Bit descriptions

**Table 7-140: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral.	RO	0xBE

## 7.4.6.7 PID1, Base SRAM ECC Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Base SRAM ECC RAS registers

### Address offset

0x0FE4

### Type

RO

### Reset value

0x000000B7

## Bit descriptions

**Table 7-141: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x7

## 7.4.6.8 PID2, Base SRAM ECC Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Base SRAM ECC RAS registers

#### Address offset

0x0FE8

#### Type

RO

#### Reset value

0x0000000B

## Bit descriptions

**Table 7-142: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x0
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

### 7.4.6.9 PID3, Base SRAM ECC Peripheral ID 3 register

The PID3 register provides information about any modifications to the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Base SRAM ECC RAS registers

##### Address offset

0x0FEC

##### Type

RO

##### Reset value

0x00000000

#### Bit descriptions

**Table 7-143: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.6.10 COMPIDO, Base SRAM ECC Component ID 0 register

The COMPIDO register contains segment 0 of the preamble to the Base SRAM ECC component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Base SRAM ECC RAS registers

**Address offset**

0x0FF0

**Type**

RO

**Reset value**

0x0000000D

**Bit descriptions****Table 7-144: COMPID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_0	Specifies segment 0 of the preamble to the code that identifies the Base SRAM ECC component class.	RO	0x0D

[7.4.6.11 COMPID1, Base SRAM ECC Component ID 1 register](#)

The COMPID1 register contains segment 1 of the preamble and the Base SRAM ECC component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Base SRAM ECC RAS registers](#)**Address offset**

0x0FF4

**Type**

RO

**Reset value**

0x000000F0

**Bit descriptions****Table 7-145: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the Base SRAM ECC component class	RO	-
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the Base SRAM ECC component class	RO	0xF0

### 7.4.6.12 COMPID2, Base SRAM ECC Component ID 2 register

The COMPID2 register contains segment 2 of the preamble to the Base SRAM ECC component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Base SRAM ECC RAS registers

##### Address offset

0x0FF8

##### Type

RO

##### Reset value

0x00000005

#### Bit descriptions

**Table 7-146: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the Base SRAM ECC component class.	RO	0x05

### 7.4.6.13 COMPID3, Base SRAM ECC Component ID 3 register

The COMPID3 register contains segment 3 of the preamble to the Base SRAM ECC component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Base SRAM ECC RAS registers

##### Address offset

0x0FFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-147: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the Base SRAM ECC component class.	RO	0xB1

## 7.4.7 Message Handling Unit registers

The MHU registers enable the configuration and operation of the Sender and Receiver.

The MHU implementation in RD-N2 only supports 32-bit word-aligned accesses. Unaligned accesses are treated as **RAZ/WI**.

The MHU is made up of two memory-mapped register frames. The first frame is used by the Sender of the transfer, while the Receiver uses the second frame. The registers in each frame are listed in the following tables.

**Table 7-148: MHU Sender frame register summary**

Offset	Name	Description	Type	Reset	Width
0x000 - 0xF7C	Sender channel window registers	MHU Sender channel window, see <a href="#">MHU channel window registers</a>	-	-	32-bit
0xF80	<a href="#">MHU_CFG</a>	Message Handling Unit Configuration	RO	See note 1	32-bit
0xF84	<a href="#">RESP_CFG</a>	MHU Response Configuration	RW	0x00000000	32-bit
0xF88	<a href="#">ACCESS_REQUEST</a>	MHU Access Request	RW	0x00000000	32-bit
0xF8C	<a href="#">ACCESS_READY</a>	MHU Access Ready	RO	0x00000000	32-bit
0xF90	<a href="#">INT_ST</a>	MHU Sender Interrupt Status	RO	0x00000000	32-bit
0xF94	<a href="#">INT_CLR</a>	MHU Sender Interrupt Clear	WO	0x00000000	32-bit
0xF98	<a href="#">INT_EN</a>	MHU Sender Interrupt Enable	RW	0x00000000	32-bit
0xF9C	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFA0	<a href="#">CHCOMB_INT_ST0</a>	MHU Sender Channel combined interrupt status (0-31)	RO	0x00000000	32-bit
0xFA4	<a href="#">CHCOMB_INT_ST1</a>	MHU Sender Channel combined interrupt status (32-63)	RO	0x00000000	32-bit
0xFA8	<a href="#">CHCOMB_INT_ST2</a>	MHU Sender Channel combined interrupt status (64-95)	RO	0x00000000	32-bit
0xFAC	<a href="#">CHCOMB_INT_ST3</a>	MHU Sender Channel combined interrupt status (96-123)	RO	0x00000000	32-bit
0xFB0 - 0xFC4	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFC8	<a href="#">I IDR</a>	MHU Implementer Identification Register	RO	0x0760043B	32-bit

Offset	Name	Description	Type	Reset	Width
0xFCC	AIDR	MHU Architecture Identification Register	RO	0x00000011	32-bit
0x0FD0	PID4	MHU Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PIDO	MHU Peripheral ID 0	RO	0x00000076	32-bit
0x0FE4	PID1	MHU Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID2	MHU Peripheral ID 2	RO	0x0000000B	32-bit
0x0FEC	PID3	MHU Peripheral ID 3	RO	0x00000000	32-bit
0xFF0	COMPIDO	MHU Component ID 0 Register	RO	0x0000000D	32-bit
0xFF4	COMPID1	MHU Component ID 1 Register	RO	0x000000F0	32-bit
0xFF8	COMPID2	MHU Component ID 2 Register	RO	0x00000005	32-bit
0xFFC	COMPID3	MHU Component ID 3 Register	RO	0x000000B1	32-bit

Note 1: The reset value for MHU\_CFG register is equal to the number of channels configured for the particular MHU. See [Message communication between processors](#) for details.

**Table 7-149: MHU Receiver frame register summary**

Offset	Name	Description	Type	Reset	Width
0x000 - 0xF7C	Receiver channel window registers	MHU Receiver channel window, see <a href="#">MHU channel window registers</a>	-	-	-
0xF80	MHU_CFG	Message Handling Unit Configuration	RO	See note 2	32-bit
0xF84 - 0xF8C	-	Reserved	RO	-	-
0xF90	INT_ST	MHU Receiver Interrupt Status	RO	0x00000000	32-bit
0xF94	INT_CLR	MHU Receiver Interrupt Clear	WO	0x00000000	32-bit
0xF98	INT_EN	MHU Receiver Interrupt Enable	RW	0x00000000	32-bit
0xF9C	-	Reserved	-	-	-
0xFA0	CHCOMB_INT_ST0	MHU Receiver Channel combined interrupt status (0-31)	RO	0x00000000	32-bit
0xFA4	CHCOMB_INT_ST1	MHU Receiver Channel combined interrupt status (32-63)	RO	0x00000000	32-bit
0xFA8	CHCOMB_INT_ST2	MHU Receiver Channel combined interrupt status (64-95)	RO	0x00000000	32-bit
0xFAC	CHCOMB_INT_ST3	MHU Receiver Channel combined interrupt status (96-123)	RO	0x00000000	32-bit
0xFC8	IIDR	MHU Implementer Identification Register	RO	0x0760043B	32-bit
0xFCC	AIDR	MHU Architecture Identification Register	RO	0x00000011	32-bit
0x0FD0	PID4	MHU Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PIDO	MHU Peripheral ID 0	RO	0x00000076	32-bit
0x0FE4	PID1	MHU Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID2	MHU Peripheral ID 2	RO	0x0000000B	32-bit
0x0FEC	PID3	MHU Peripheral ID 3	RO	0x00000000	32-bit
0xFF0	COMPIDO	MHU Component ID 0 Register	RO	0x0000000D	32-bit
0xFF4	COMPID1	MHU Component ID 1 Register	RO	0x000000F0	32-bit
0xFF8	COMPID2	MHU Component ID 2 Register	RO	0x00000005	32-bit
0xFFC	COMPID3	MHU Component ID 3 Register	RO	0x000000B1	32-bit

Note 2: The reset value for MHU\_CFG register is equal to the number of channels configured for the particular MHU. See [Message communication between processors](#) for details.

### 7.4.7.1 MHU channel window registers

A channel window is a group of registers. The registers in the channel window vary between the Sender and Receiver frame views.

An MHU implementation can contain between 1 and 124 channels. The number of channels that are implemented can be discovered from the MHU\_CFG.NUM\_CH field. Each channel occupies eight 32-bit words in both the Sender and Receiver register maps. The address space that is allocated to channels that are not implemented is Reserved and treated as **RAZ/WI**.

The following tables list the registers in the Sender and Receiver channel windows.

**Table 7-150: MHU Sender channel window register summary**

Offset	Name	Type	Reset	Width	Description
0x00	CH_ST	RO	-	32-bit	MHU Channel Status
0x04 – 0x08	RESERVED	RAZ/WI	-	32-bit	Reserved
0x0C	CH_SET	WO	-	32-bit	MHU Channel Set
0x10 – 0x18	RESERVED	RAZ/WI	-	32-bit	Reserved

**Table 7-151: MHU Receiver channel window register summary**

Offset	Name	Type	Reset	Width	Description
0x00	CH_ST	RO	-	32-bit	MHU Channel Status
0x04	CH_ST_MSK	RO	-	32-bit	MHU Channel Status Masked
0x08	CH_CLR	WO	-	32-bit	MHU Channel Clear
0x0C	RESERVED	RAZ/WI	-	32-bit	Reserved
0x10	CH_MSK_ST	RO	-	32-bit	MHU Channel Mask Status
0x14	CH_MSK_SET	WO	-	32-bit	MHU Channel Mask Set
0x18	CH_MSK_CLR	WO	-	32-bit	MHU Channel Mask Clear
0x1C	RESERVED	RAZ/WI	-	32-bit	Reserved

#### 7.4.7.1.1 CH\_ST, MHU Channel Status register

The CH\_ST register shows the state of the channel and is part of both the Receiver and Sender channel windows.

If the Receiver frame is reset, then the contents of the CH\_ST register in the Sender frame are also reset. Software is responsible for handling any lost messages when the Receiver frame and CH\_ST register are reset.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MHU channel window registers](#)

### Address offset

0x00

### Type

RO

## Bit descriptions

**Table 7-152: CH\_ST bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAGn, n = 0– 31	<p>Display the status of channel flags. Each bit can be used as an individual flag or bits can be grouped. The way in which the register is used depends on the transport protocol that is employed.</p> <p>Bits in this register are set by writing 0b1 to the corresponding bits in the CH_SET register. Writing 0b1 to bits in the CH_CLR register clears the corresponding bits in the CH_ST register.</p> <p>If software:</p> <ul style="list-style-type: none"> <li>• Sets a bit that is already set, the bit remains set.</li> <li>• Clears a bit that is already cleared, the bit remains cleared.</li> <li>• Sets and clears a bit at the same time, the bit remains set.</li> </ul> <p>Arm strongly recommends that software follows the transport protocols that are defined for the MHU.</p>	RO	0x0000_0000

### 7.4.7.1.2 CH\_ST\_MSK, MHU Channel Status Masked register

The CH\_ST\_MSK register shows the state of the channel with the channel mask applied, and is part of the Receiver channel window.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MHU channel window registers](#)

### Address offset

0x04

**Type**

RO

**Bit descriptions****Table 7-153: CH\_ST\_MSK bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAG_MSKn, n = 0–31	Display the status of channel flags with the mask applied. When this register is nonzero, the interrupt for the channel is asserted. The value in this register is equal to CH_ST and ~CH_MSK_ST at the point at which the read occurs.	RO	0x0000_0000

**7.4.7.1.3 CH\_CLR, MHU Channel Clear register**

The CH\_CLR register resets bits in the Channel Status and Channel Status Masked registers, and is part of the Receiver channel window.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MHU channel window registers](#)**Address offset**

0x08

**Type**

WO

**Bit descriptions****Table 7-154: CH\_CLR bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAG_CLRn, n = 0–31	Clear the channel flags. Writing 0b1 to bits in this register clears the corresponding bits in the CH_ST and CH_ST_MSK registers. Writing 0b0 to bits in this register has no effect. Each bit always reads as 0b0.	WO	0x0000_0000

**7.4.7.1.4 CH\_SET, MHU Channel Set register**

The CH\_SET register writes bits in the Channel Status and Channel Status Mask registers, and is part of the Sender channel window.

**Configurations**

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MHU channel window registers](#)

### Address offset

0x0C

### Type

WO

## Bit descriptions

**Table 7-155: CH\_SET bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAG_SETn, n = 0–31	Set the channel flags. Writing 0b1 to bits in this register sets the corresponding bits in the CH_ST register. Writing 0b0 to bits in this register has no effect. Each bit always reads as 0b0.	WO	0x0000_0000

### 7.4.7.1.5 [CH\\_MSK\\_ST](#), MHU Channel Mask Status register

The CH\_MSK\_ST register shows the state of the channel mask, and is part of the Receiver channel window. The channel mask is used with the Channel Status register to generate the channel status mask.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MHU channel window registers](#)

### Address offset

0x10

### Type

RO

## Bit descriptions

**Table 7-156: CH\_MSK\_ST bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAG_MSKn, n = 0-31	Display the status of channel flag masks. A channel mask bit that is set to 0b0 indicates that the corresponding flag bit is unmasked. When a bit is unmasked, the equivalent bits in the CH_ST and CH_ST_MSK registers have the same value. A channel mask bit that is set to 0b1 indicates that the corresponding flag bit is masked. When a bit is masked, the equivalent bit in the CH_ST_MSK register always reads as 0b0.	RO	0x0000_0000

### 7.4.7.1.6 CH\_MSK\_SET, MHU Channel Mask Set register

The CH\_MSK\_SET register writes bits in the channel mask and is part of the Receiver channel window.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

MHU channel window registers

##### Address offset

0x14

##### Type

WO

## Bit descriptions

**Table 7-157: CH\_MSK\_SET bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAG_MSK_SETn, n = 0-31	Set the channel flag masks. Writing 0b1 to bits in this register sets the corresponding bits in the CH_MSK_ST register. Writing 0b0 to bits in this register has no effect. Each bit always reads as 0b0.	WO	0x0000_0000

### 7.4.7.1.7 CH\_MSK\_CLR, MHU Channel Mask Clear register

The CH\_MSK\_CLR register resets bits in the channel mask and is part of the Receiver channel window.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MHU channel window registers](#)

### Address offset

0x18

### Type

WO

## Bit descriptions

**Table 7-158: CH\_MSK\_CLR bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	FLAG_MSK_CLRn, n = 0-31	Clear the channel flag masks. Writing 0b1 to bits in this register clears the corresponding bits in the CH_MSK_ST register. Writing 0b0 to bits in this register has no effect. Each bit always reads as 0b0.	WO	0x0000_0000

## 7.4.7.2 MHU\_CFG, Message Handling Unit Configuration register

The MHU\_CFG register shows the number of channels that are implemented in the MHU.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[Message Handling Unit registers](#)

### Address offset

0xF80

### Type

RO

## Bit descriptions

**Table 7-159: MHU\_CFG bit descriptions**

Bits	Name	Description	Type	Default
[31:7]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Default
[6:0]	NUM_CH	Specifies the number of MHU channels that are implemented. The value of the field indicates the number of channels, up to a maximum of 124 (0x7C).  The values 0x00, 0x7D, 0x7E, and 0x7F are reserved.	RO	CFG_DEF

### 7.4.7.3 RESP\_CFG, MHU Response Configuration register

The RESP\_CFG register shows the number of channels that are implemented in the MHU.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

##### Address offset

0xF84

##### Type

RW

#### Bit descriptions

**Table 7-160: RESP\_CFG bit descriptions**

Bits	Name	Description	Type	Default
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	NR_RESP	Specifies the response that is generated when the Sender attempts to access any channel window register while the ACCESS_READY.ACC_RDY field is set to 0b0. This setting indicates that the Receiver is not in a state in which it can accept a transfer.  When the Receiver is not ready, channel window register access attempts by the Sender are treated as <b>RAZ/WI</b> . With the RESP_CFG.NR_RESP field set to 0b0, an error is not generated. If this field is set to 0b1, then an error is generated.	RO	CFG_DEF

#### 7.4.7.4 ACCESS\_REQUEST, MHU Access Request register

The ACCESS\_REQUEST register is used by the Sender to require that the Receiver enters a state in which the Receiver can accept a transfer.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[Message Handling Unit registers](#)

###### Address offset

0xF88

###### Type

RW

##### Bit descriptions

**Table 7-161: ACCESS\_REQUEST bit descriptions**

Bits	Name	Description	Type	Default
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	ACC_REQ	Requests that the Receiver prepares to accept a transfer. A setting of 0b0 for this field indicates that the Receiver does not need to prepare for a transfer. If this field is set to 0b1, then the Receiver is requested to prepare to accept a transfer.	RW	0b0

#### 7.4.7.5 ACCESS\_READY, MHU Access Ready register

The ACCESS\_READY register shows whether the Receiver is in a state in which it can accept a transfer from the Sender.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[Message Handling Unit registers](#)

###### Address offset

0xF8C

**Type**

RO

**Bit descriptions****Table 7-162: ACCESS\_READY bit descriptions**

Bits	Name	Description	Type	Default
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	ACC_RDY	Specifies whether the Receiver is able to accept a transfer. A setting of 0b0 for this field indicates that the Receiver is not able to accept a transfer. If this field is set to 0b1, then the Receiver is able to accept a transfer.	RW	0b0

**7.4.7.6 INT\_ST, MHU Sender Interrupt Status register**

The INT\_ST register shows whether the ready to not ready and not ready to ready interrupts have been generated.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0xF90

**Type**

RO

**Bit descriptions****Table 7-163: INT\_ST bit descriptions**

Bits	Name	Description	Type	Default
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	CHCOMB	<p>Channel combined interrupt status.</p> <ul style="list-style-type: none"> <li>• 0b0 – No interrupt has occurred on any Channel.</li> <li>• 0b1 – An interrupt has occurred on at least one Channel.</li> </ul> <p>There is no corresponding bit in the INT_CLR register. To clear this interrupt, software must clear the underlying interrupt.</p>	RO	0b0

Bits	Name	Description	Type	Default
[1]	R2NR	Ready to not ready interrupt status. <ul style="list-style-type: none"><li>• 0b0 – Ready to not ready interrupt has not occurred.</li><li>• 0b1 – Ready to not ready interrupt has occurred.</li></ul>	RO	0b0
[0]	NR2R	Not ready to ready interrupt status. <ul style="list-style-type: none"><li>• 0b0 -Not ready to ready interrupt has not occurred.</li><li>• 0b1 – Not ready to ready interrupt has occurred.</li></ul>	RO	0b0

### 7.4.7.7 INT\_CLR, MHU Sender Interrupt Clear register

The INT\_CLR register resets the ready to not ready and not ready to ready interrupts.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Message Handling Unit registers

##### Address offset

0xF94

##### Type

WO

#### Bit descriptions

**Table 7-164: INT\_CLR bit descriptions**

Bits	Name	Description	Type	Default
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	R2NR	Clears the ready to not ready interrupt. Writing 0b1 to this field clears the ready to not ready interrupt.  Writing 0b0 to this field has no effect.	WO	0b0
[0]	NR2R	Clears the not ready to ready interrupt. Writing 0b1 to this field clears the not ready to ready interrupt.  Writing 0b0 to this field has no effect.	WO	0b0

### 7.4.7.8 INT\_EN, MHU Sender Interrupt Enable register

The INT\_EN register activates and deactivates generation of the ready to not ready and not ready to ready interrupts.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Message Handling Unit registers

##### Address offset

0xF98

##### Type

RW

#### Bit descriptions

**Table 7-165: INT\_EN bit descriptions**

Bits	Name	Description	Type	Default
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	CHCOMB	Channel combined interrupt enable 0b0 – Combined interrupt is disabled. 0b1 – Combined interrupt is enabled.	RO	0b1
[1]	R2NR	Ready to not ready interrupt enable. 0b0 – Ready to not ready interrupt has not occurred.  0b1 – Ready to not ready interrupt has occurred.	RW	0b0
[0]	NR2R	Not ready to ready interrupt enable. 0b0 -Not ready to ready interrupt has not occurred.  0b1 – Not ready to ready interrupt has occurred.	RW	0b0

### 7.4.7.9 INT\_ST, MHU Receiver Interrupt Status register

The INT\_ST register shows whether the ready to not ready and not ready to ready interrupts have been generated.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0xF90

**Type**

RO

**Bit descriptions****Table 7-166: INT\_ST bit descriptions**

Bits	Name	Description	Type	Default
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	CHCOMB	<p>Channel combined interrupt status.</p> <ul style="list-style-type: none"> <li>• 0b0 – No interrupt has occurred on any Channel.</li> <li>• 0b1 – An interrupt has occurred on at least one Channel.</li> </ul> <p>There is no corresponding bit in the INT_CLR register. To clear this interrupt, software must clear the underlying interrupt.</p>	RO	0b0
[1:0]	RESERVED	Reserved	RAZ/WI	-

**7.4.7.10 INT\_CLR, MHU Receiver Interrupt Clear register**

This register is included for completeness and for an orthogonal set of registers between Sender and Receiver. It has no functionality in the receiver.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0xF94

**Type**

WO

## Bit descriptions

**Table 7-167: INT\_CLR bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.7.11 INT\_EN, MHU Receiver Interrupt Enable register

The INT\_EN register activates and deactivates generation of the ready to not ready and not ready to ready interrupts.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

##### Address offset

0xF98

##### Type

RW

## Bit descriptions

**Table 7-168: INT\_EN bit descriptions**

Bits	Name	Description	Type	Default
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	CHCOMB	Channel combined interrupt enable 0b0 – Combined interrupt is disabled. 0b1 – Combined interrupt is enabled.	RO	0b1
[1:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.7.12 CHCOMB\_INT\_ST<0-3>, MHU Sender Channel Combined Interrupt Status 0-3 register

The CHCOMB\_INT\_ST register indicates the channel interrupt status for the respective channels.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[Message Handling Unit registers](#)

### Address offset

0xFA0 – 0xFAC

### Type

RO

## Bit descriptions

**Table 7-169: CHCOMB\_INT\_ST<0-3> bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	CHCOMB_INT_ST{x}	<p>Channel interrupt status. Each bit indicates whether a Channel has a pending interrupt or not.</p> <p>CHCOMB_INT_ST0 has the status for Channels 0 to 31, starting with Channel 0 at bit 0.</p> <p>CHCOMB_INT_ST1 has the status for Channels 32 to 63, starting with Channel 32 at bit 0.</p> <p>CHCOMB_INT_ST2 has the status for Channels 64 to 95, starting with Channel 64 at bit 0.</p> <p>CHCOMB_INT_ST3 has the status for Channels 96 to 123, starting with Channel 96 at bit 0.</p> <p>A bit relating to an unimplemented Channel is Reserved and treated as <b>RAZ/WI</b>.</p> <p>CHCOMB_INT_ST3 bits [31:28] are always Reserved and treated as <b>RAZ/WI</b>.</p>	RO	0x0000_0000

### 7.4.7.13 CHCOMB\_INT\_ST<0-3>, MHU Receiver Channel Combined Interrupt Status 0-3 register

The CHCOMB\_INT\_ST register indicates the channel interrupt status for the respective channels.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[Message Handling Unit registers](#)

**Address offset**

0xFA0 – 0xFAC

**Type**

RO

**Bit descriptions****Table 7-170: CHCOMB\_INT\_ST<0-3> bit descriptions**

Bits	Name	Description	Type	Default
[31:0]	CHCOMB_INT_ST{x}	<p>Channel interrupt status. Each bit indicates whether a Channel has a pending interrupt or not.</p> <p>CHCOMB_INT_ST0 has the status for Channels 0 to 31, starting with Channel 0 at bit 0.</p> <p>CHCOMB_INT_ST1 has the status for Channels 32 to 63, starting with Channel 32 at bit 0.</p> <p>CHCOMB_INT_ST2 has the status for Channels 64 to 95, starting with Channel 64 at bit 0.</p> <p>CHCOMB_INT_ST3 has the status for Channels 96 to 123, starting with Channel 96 at bit 0.</p> <p>A bit relating to an unimplemented Channel is reserved and treated as <b>RAZ/WI</b>.</p> <p>CHCOMB_INT_ST3 bits [31:28] are always reserved and treated as <b>RAZ/WI</b>.</p>	RO	0x0000_0000

**7.4.7.14 IIDR, MHU Implementer Identification Register**

The IIDR contains information about the MHU implementation.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0xFC8

**Type**

RO

**Reset value**

0x0760043B

## Bit descriptions

**Table 7-171: IIDR bit descriptions**

Bits	Name	Description	Type	Default
[31:20]	PRODUCT_ID	Specifies the MHU part identifier.	RO	0x076
[19:16]	VARIANT	Specifies the MHU major revision number.	RO	0x0
[15:12]	REVISION	Specifies the MHU minor revision number.	RO	0x0
[11:0]	IMPLEMENTER	Specifies the JEDEC JEP106 manufacturers identification code for the MHU implementer. Bits[11:8] contain the JEP106 continuation code for the implementer, bit[7] must always be 0, and bits[6:0] give the JEP106 identity code for the implementer.	RO	0x43B

## 7.4.7.15 AIDR, MHU Architecture Identification Register

The AIDR contains the MHU architecture version.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Message Handling Unit registers](#)

#### Address offset

0xFCC

#### Type

RO

## Bit descriptions

**Table 7-172: AIDR bit descriptions**

Bits	Name	Description	Type	Default
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	ARCH_MAJOR_REV	Specifies the MHU major architecture revision number. A value of 0x1 indicates that the MHU conforms to MHU architecture version 2. The setting 0x0 is Reserved.  When the ARCH_MAJOR_REV field is set to 0x0, the values in the ARCH_MINOR_REV field and IIDR register are RAZ. Software must determine in a platform-specific manner the MHU architecture version to which the component conforms.	RO	0x1
[3:0]	ARCH_MINOR_REV	Specifies the MHU minor architecture revision number. A value of 0x0 indicates that the architecture minor revision number is 0. All other values are reserved.	RO	0x0

### 7.4.7.16 PID4, MHU Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

##### Address offset

0x0FD0

##### Type

RO

##### Reset value

0x00000004

#### Bit descriptions

**Table 7-173: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2. Set to 0x0.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

### 7.4.7.17 PID0, MHU Peripheral ID 0 register

The PID0 register contains the first eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

**Address offset**

0x0FE0

**Type**

RO

**Reset value**

0x00000076

**Bit descriptions****Table 7-174: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral.	RO	0x76

[7.4.7.18 PID1, MHU Peripheral ID 1 register](#)

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0x0FE4

**Type**

RO

**Reset value**

0x000000B0

**Bit descriptions****Table 7-175: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral.	RO	0x0

### 7.4.7.19 PID2, MHU Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

##### Address offset

0x0FE8

##### Type

RO

##### Reset value

0x0000000B

#### Bit descriptions

**Table 7-176: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block.	RO	0x0
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

### 7.4.7.20 PID3, MHU Peripheral ID 3 register

The PID3 register provides information about any modifications to the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

**Address offset**

0x0FEC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-177: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[3:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

**7.4.7.21 COMPIDO, MHU Component ID 0 register**

The COMPIDO register contains segment 0 of the preamble to the MHU component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0x0FF0

**Type**

RO

**Reset value**

0x0000000D

**Bit descriptions****Table 7-178: COMPIDO bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_0	Specifies segment 0 of the preamble to the code that identifies the MHU component class.	RO	0x0D

### 7.4.7.22 COMPID1, MHU Component ID 1 register

The COMPID1 register contains segment 1 of the preamble and the MHU component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

##### Address offset

0x0FF4

##### Type

RO

##### Reset value

0x000000F0

#### Bit descriptions

**Table 7-179: COMPID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	CLASS	Specifies a code that identifies the MHU component class	RO	-
[3:0]	PRMBL_1	Specifies segment 1 of the preamble to the code that identifies the MHU component class	RO	0xF0

### 7.4.7.23 COMPID2, MHU Component ID 2 register

The COMPID2 register contains segment 2 of the preamble to the MHU component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Message Handling Unit registers](#)

**Address offset**

0x0FF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-180: COMPID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_2	Specifies segment 2 of the preamble to the code that identifies the MHU component class.	RO	0x05

**7.4.7.24 COMPID3, MHU Component ID 3 register**

The COMPID3 register contains segment 3 of the preamble to the MHU component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Message Handling Unit registers](#)**Address offset**

0x0FFC

**Type**

RO

**Reset value**

0x000000B1

**Bit descriptions****Table 7-181: COMPID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PRMBL_3	Specifies segment 3 of the preamble to the code that identifies the MHU component class.	RO	0xB1

## 7.4.8 Core Manager and clock control registers

The Core Manager and clock control registers enable configuration of reset values and clock settings for core subsystems.

The register summary in the following table provides a fully configured register block.

**Table 7-182: Core Manager and clock control register summary**

Offset	Name	Description	Type	Reset	Width
0x0000 - 0x000C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0010	PE_STATIC_CONFIG	Processing Element Configuration	RW	0x00000000	32-bit
0x0014	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0018	PE_RVBARADDR_LW	Processing Element Reset Vector Base Address Lower Word	RW	0x00000000	32-bit
0x001C	PE_RVBARADDR_UP	Processing Element Reset Vector Base Address Upper Word	RW	0x00000000	32-bit
0x0020 - 0x002C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0030	PE_STATUS	Status register to capture Core PMU signal	RO	0x00000000	32-bit
0x0034 - 0x03FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0400 - 0x07FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0800	CLUS_PPUCLK_CTRL	Cluster PPU Clock Control	RW	0x00000000	32-bit
0x0804	CLUS_PPUCLK_DIV1	Cluster PPU Clock Divider Control	RW	0x000F000F	32-bit
0x0808 - 0x083C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0840	CLUS_GICCLK_CTRL	Cluster GIC Clock Control	RW	0x00000000	32-bit
0x0844	CLUS_GICCLK_DIV1	Cluster GIC Clock Divider Control	RW	0x000F000F	32-bit
0x0848 - 0x084C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0850	CLUS_PERIPHCLK_CTRL	Cluster Peripheral Clock Control	RW	0x00000000	32-bit
0x0854	CLUS_PERIPHCLK_DIV1	Cluster Peripheral Clock Divider Control	RW	0x000F000F	32-bit
0x0858 - 0x085C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0860	CORECLK_CTRL	Core Clock Control	RW	0x00000101	32-bit
0x0864	CORECLK_DIV1	Core Clock Divider Control	RW	0x000F000F	32-bit
0x0868	CORECLK_MOD1	Core Clock Modulator Control	RW	0x101010101	32-bit
0x086C - 0x9FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xA00	CLKFORCE_STATUS	Clock Force Status	RO	0x00000000	32-bit
0xA04	CLKFORCE_SET	Clock Force Set	WO	0x00000000	32-bit
0xA08	CLKFORCE_CLR	Clock Force Clear	WO	0x00000000	32-bit

Offset	Name	Description	Type	Reset	Width
0x0A0C – 0x0FB0	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0FB4	CAP3	Core Capabilities 3	RO	0x00000001	32-bit
0x0FB8	CAP2	Core Capabilities 2	RO	0x00000000	32-bit
0x0FBC	CAP1	Core Capabilities 1	RO	0x00000003	32-bit
0x0FC0	PWR_CTRL_CONFIG	Power Control Configuration	RO	0x00140000	32-bit
0x0FC4 – 0x0FCC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0FD0	PID4	Power Control Peripheral ID 4	RO	0x00000044	32-bit
0x0FD4	PID5	Power Control Peripheral ID 5	RO	0x00000000	32-bit
0x0FD8	PID6	Power Control Peripheral ID 6	RO	0x00000000	32-bit
0x0FDC	PID7	Power Control Peripheral ID 7	RO	0x00000000	32-bit
0x0FE0	PID0	Power Control Peripheral ID 0	RO	0x000000B8	32-bit
0x0FE4	PID1	Power Control Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID2	Power Control Peripheral ID 2	RO	0x0000000B	32-bit
0x0FEC	PID3	Power Control Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	ID0	Power Control Component ID 0 register	RO	0x0000000D	32-bit
0x0FF4	ID1	Power Control Component ID 1 register	RO	0x000000F0	32-bit
0x0FF8	ID2	Power Control Component ID 2 register	RO	0x00000005	32-bit
0x0FFC	ID3	Power Control Component ID 3 register	RO	0x000000B1	32-bit

#### 7.4.8.1 PE\_STATIC\_CONFIG, Processing Element Configuration register

This register drives configuration reset values for the PE. Under normal circumstances, SCP firmware programs these values prior to power up and reset de-assertion of the cluster. The reset values must match the reset values of the specific core.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

Core Manager and clock control registers

###### Address offset

0x0010

###### Type

RW

**Reset value**

0x00000000

**Bit descriptions****Table 7-183: PE Static Configuration bit descriptions**

Bits	Name	Description	Type	Reset
[31:9]	RESERVED	Reserved	RAZ/WI	-
[8]	DISPBLK	Is this bits is set, stalls dispatch indefinitely and is intended to limit forward progress (initiation of new transactions, for example) in certain emergency conditions.	RW	0x0
[7:4]	PDPSTATE	Override value for the following register field when external pin control is enabled. CPUPPMPDPCR_EL1. PDP_EXTMS_SET CPUPPMPDPCR_EL1. PDP_CORE_SET	RW	0x0
[3:2]	MPMSTATE	Override value for the following core register field when external pin control is enabled.  CPUMPMCR_EL3. MPMM_GEAR[1:0]	RW	0x0
[1]	RESERVED	Enable bit for max-power throttling mechanism of each processor. This is override value for CPUMPMCR_EL3. MPMM_EN when external pin control is enabled.	RW	0x0
[0]	CFGEND	Individual processor control of the endianness configuration at reset.  This control is only sampled by the processor during reset.	RW	0x0

### 7.4.8.2 PE\_RVBARADDR\_LW, Processing Element Reset Vector Base Address Lower Word register

This register provides lower 32-bits of the reset value of the RVBAR\_EL3 register in the PE. The value programmed in this register is used as the address from which execution starts when the processor is in 64-bit state. The reset values must match the reset values of the specific core.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0018

**Type**

RW

**Reset value**

0x00000000

## Bit descriptions

**Table 7-184: PE RVBARADDR\_LW bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	RVBAR	<p>Provides bits [31:2] of the reset value of the RVBAR_EL3 register in the PE. This value is used as the address that execution starts from when the processor is in 64-bit state.</p> <p>All other bits of RVBAR_EL3 are zero.</p> <p>This value is used to drive bits [31:2] of all of the RVBARADDRx inputs to the PE.</p> <p>x = 0, 1, 2, 3...</p> <p>These pins are only sampled during reset of the processor.</p>	RW	0x0
[1:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.8.3 PE\_RVBARADDR\_UP, Processing Element Reset Vector Base Address Upper Word register

This register provides upper 32-bits of the reset value of the RVBAR\_EL3 register in the PE. The value programmed in this register is used as the address from which execution starts when the processor is in 64-bit state. The reset values must match the reset values of the specific core.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x001C

##### Type

RW

##### Reset value

0x00000000

## Bit descriptions

**Table 7-185: PE RVBARADDR\_UP bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[11:0]	RVBAR	<p>Provides bits [43:32] of the reset value of the RVBAR_EL3 register in the PE. This value is used as the address that execution starts from when the processor is in 64-bit state.</p> <p>All other bits of RVBAR_EL3 are zero.</p> <p>This value is used to drive bits [43:32] of all of the RVBARADDRx inputs to the PE.</p> <p>x = 0, 1, 2, 3...</p> <p>These pins are only sampled during reset of the processor.</p>	RW	0x0

#### 7.4.8.4 PE\_STATUS, Core PMU signal status register

This register captures status of Core PMU signals that are exported out.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

Core Manager and clock control registers

###### Address offset

0x0030

###### Type

RO

###### Reset value

0x0

##### Bit descriptions

Table 7-186: PE\_STATUS bit descriptions

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	COREINSTRRUN	Indicates core is in running state	RO	-
[0]	COREINSTRRET	Indicates core has retired at least one instruction recently	RO	-

### 7.4.8.5 CLUS\_PPUCLK\_CTRL, Cluster PPU Clock Control register

This register provides the ability to program the number of clock cycles between the CLUS\_PPUCLK not being required and the request to dynamically clock gate it.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x0800

##### Type

RW

##### Reset value

0x0000\_0000

#### Bit descriptions

**Table 7-187: CLUS\_PPUCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This is reserved If dynamic clock gating is not implemented.</p>	RW	-
[23:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.8.6 CLUS\_PPUCLK\_DIV1, Cluster PPU Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (INTCLK) to generate the required output clock (CLUS\_PPUCLK). The current divider value can also be read out from this register.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Core Manager and clock control registers](#)

#### Address offset

0x0804

#### Type

RW

#### Reset value

0x000F\_000F

### Bit descriptions

**Table 7-188: CLUS\_PPUCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently selected clock divider value for CLUS_PPUCLK.  The divider value is the value of CLKDIV_CUR + 1, for example, setting a value of 0 indicates a divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value on source clock INTCLK to generate respective output clock.  The divider value is the value of CLKDIV + 1, for example, setting a value of 0 indicates a divider value of 1.  The divider values must be changed only when the clock source selected is REFCLK.	RW	0x0F

### 7.4.8.7 CLUS\_GICCLK\_CTRL, Cluster GIC Clock Control register

This register provides the ability to program the number of clock cycles between the CLUS\_GICCLK not being required and the request to dynamically clock gate it.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x0840

##### Type

RW

##### Reset value

0x0000\_0000

#### Bit descriptions

**Table 7-189: CLUS\_GICCLK bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.8.8 CLUS\_GICCLK\_DIV1, Cluster GIC Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (INTCLK) to generate the required output clock (CLUS\_GICCLK). The current divider value can also be read out from this register.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Core Manager and clock control registers](#)

#### Address offset

0x0844

#### Type

RW

#### Reset value

0x000F\_000F

### Bit descriptions

**Table 7-190: CLUS\_GICCLK\_DIV bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently selected divider value for CLK_GICCLK.  The divider value is the value of CLKDIV_CUR +, for example, setting a value of 0 indicates a divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value on source clock INTCLK to generate CLUS_GICCLK.  The divider value is the value of CLKDIV + 1, for example, setting a value of 0 indicates a divider value of 1.	RW	0x0F

## 7.4.8.9 CLUS\_PERIPHCLK\_CTRL, Cluster Peripheral Clock Control register

This register provides the ability to program the number of clock cycles between the CLUS\_PERIPHCLK not being required and the request to dynamically clock gate it.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Core Manager and clock control registers

### Address offset

0x0850

### Type

RW

### Reset value

0x0000\_0000

## Bit descriptions

**Table 7-191: CLUS\_PERIPHCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	Number of clock cycles between the clock not being required and the request to dynamically clock gate it.  0x0 - No cycles  0x1 - 1 cycle  ...  0xFF - 255 cycles	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-

## 7.4.8.10 CLUS\_PERIPHCLK\_DIV1, Cluster Peripheral Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (INTCLK) to generate the required output clock (CLUS\_PERIPHCLK). The current divider value can also be read out from this register.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Core Manager and clock control registers

**Address offset**

0x0854

**Type**

RW

**Reset value**

0x000F\_000F

**Bit descriptions****Table 7-192: CLUS\_PERIPHCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently selected clock divider value for CLUS_PERIPHCLK.  The divider value is the value of CLKDIV_CUR + 1, for example, setting a value of 0 indicates a divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value on source clock INTCLK to generate respective output clock.  The divider value is the value of CLKDIV + 1, for example, setting a value of 0 indicates a divider value of 1.	RW	0x0F

### 7.4.8.11 CORE\_CLK\_CTRL, Core Clock Control register

This register provides the ability to program the number of clock cycles between the CORECLK not being required and the request to dynamically clock gate it. The clock source of the CORECLK can also be programmed through this register.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Core Manager and clock control registers](#)**Address offset**

0x0860

**Type**

RW

**Reset value**

0x0000\_0101

**Bit descriptions****Table 7-193: CORECLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - CPU&lt;n&gt;.COREPLLCLK</p> <p>Other values Reserved.</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - CPU&lt;n&gt;.COREPLLCLK</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

**7.4.8.12 CORECLK\_DIV1, Core Clock Divider Control register**

This register provides the ability to request a new clock divider value on the source clock (COREPLLCLK) to generate the required output clock (CORECLK). The current divider value can also be read out from this register.

**Configurations**

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Core Manager and clock control registers

### Address offset

0x0864

### Type

RW

### Reset value

0x000F\_000F

### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-194: CORECLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Current value of the integer divider applied to clock selected by CORExCLK.CLKSELECTnCLK.CLKSELECT  0 - Divide by 1  1 - Divide by 2  ....  1F - Divide by 32	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Select the value of the integer divider applied to clock selected by CORExCLK.CLKSELECTnCLK.CLKSELECT  0 - Divide by 1  1 - Divide by 2  ....  1F - Divide by 32	RW	0x0F

### 7.4.8.13 CORECLK\_MOD1, Core- and Complex Clock Modulator Control register

The CORECLK modulator values (both numerator and denominator) can be programmed through this register. This is an optional finer control to support the fraction of the divided clock to be gated.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Core Manager and clock control registers](#)

##### Address offset

0x0868

##### Type

RW

##### Reset value

0x0101\_0101

#### Bit descriptions

**Table 7-195: CORECLK\_MOD1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	CLKMOD_NUM_CUR	Current value of the clock modulator numerator.	RW	-
[23:16]	CLKMOD_DEN_CUR	Current value of the clock modulator denominator.	RW	-
[15:8]	CLKMOD_NUM	Clock modulator numerator.  Behavior is undefined for the following: <ul style="list-style-type: none"><li>• writing 0 to NUM or DEN fields</li><li>• writing NUM &gt; DEN</li></ul>	RW	-
[7:0]	CLKMOD_DEN	Clock modulator denominator. A value of 0 is RESERVED and the result of writing a 0 to this field is <b>UNPREDICTABLE</b> . If the numerical value of CLKMOD_DEN is smaller than the numerical value of CLKMOD_NUM then that results in the clock being enabled for all clock cycles.  Behavior is undefined for the following: <ul style="list-style-type: none"><li>• writing 0 to NUM or DEN fields</li><li>• writing NUM &gt; DEN</li></ul>	RW	-

### 7.4.8.14 CLKFORCE\_STATUS, Clock Force Status register

This register captures the status (enabled/disabled) of the dynamic clock gating on CLUS\_PPUCLK, CLUS\_PERIPHCLK, CLUS\_GICCLK and CORECLK.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x0A00

##### Type

RO

##### Reset value

0x0

#### Bit descriptions

If a bit reads back as 1 then the associated dynamic clock gating associated with the clock is disabled otherwise it is enabled.

If dynamic clock gating is not implemented for a particular clock, the corresponding CLKFORCE bit is reserved.

**Table 7-196: CLKFORCE\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
Direct Connect Mode: [31:4]	RESERVED	Direct Connect Mode: Reserved	RAZ/WI RO	-
DSU Mode: [11:4], [31:12]	DSU Mode: COMPLEX<n>CLKFORCE,-	DSU Mode: One bit per complex clock. Any unused bits are treated as RESERVED (RAZ/WI), Reserved	RO	
[3]	CLUS_PERIPHCLKFORCE	Clock for CLUS_PERIPHCLK	RO	-
[2]	CLUS_GICCLKFORCE	Clock for CLUS_GICCLK	RO	-
[1]	CLUS_PPUCLKFORCE	Clock for CLUS_PPUCLK	RO	-

Bits	Name	Description	Type	Reset
[0]	Direct Connect Mode: CLUS_CORECLKFORCE DSU Mode: RESERVED	Direct Connect Mode: Clock force for CORECLK DSU Mode: Reserved	RO  RAZ/ WI	-

### 7.4.8.15 CLKFORCE\_SET, Clock Force Set register

This register provides the ability to disable dynamic clock gating on the respective clock.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x0A04

##### Type

WO

##### Reset value

0x0

#### Bit descriptions

Writing a 1 to a bit within the CLKFORCE\_SET register disables any dynamic hardware clock gating for that respective clock, whilst writing 0 to a bit is ignored. The bit allocation is the same as the CLKFORCE\_STATUS register.

If dynamic clock gating is not implemented for a particular clock, the corresponding CLKFORCE bit is reserved.

**Table 7-197: CLKFORCE\_SET bit descriptions**

Bits	Name	Description	Type	Reset
Direct Connect Mode: [31:4]	RESERVED	Direct Connect Mode: Reserved	RAZ/ WI WO	-
DSU Mode: [11:4], [31:12]	DSU Mode: COMPLEX<n>CLKFORCE,-	DSU Mode: One bit per complex clock. Any unused bits are treated as RESERVED (RAZ/WI), Reserved		
[3]	CLUS_PERIPHCLKFORCE	Clock for CLUS_PERIPHCLK	WO	-
[2]	CLUS_GICCLKFORCE	Clock for CLUS_GICCLK	WO	-
[1]	CLUS_PPUCLKFORCE	Clock for CLUS_PPUCLK	WO	-
[0]	Direct Connect Mode: CLUS_CORECLKFORCE DSU Mode: RESERVED	Direct Connect Mode: Clock force for CORECLK DSU Mode: Reserved	WO RAZ/ WI	-

#### 7.4.8.16 CLKFORCE\_CLR, Clock Force Clear register

Writing a 1 to a bit within the CLKFORCE\_CLR register enables the dynamic hardware clocking gating for that respective clock, whilst writing 0 to a bit is ignored. The bit allocation is the same as the CLKFORCE\_STATUS register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

Core Manager and clock control registers

###### Address offset

0x0A08

###### Type

WO

###### Reset value

0x0

## Bit descriptions

If dynamic clock gating is not implemented for a particular clock, the corresponding CLKFORCE bit is reserved.

**Table 7-198: CLKFORCE\_CLR bit descriptions**

Bits	Name	Description	Type	Reset
Direct Connect Mode: [31:4]	RESERVED	Direct Connect Mode: Reserved	RAZ/ WI	-
DSU Mode: [11:4], [31:12]	DSU Mode: COMPLEX<n>CLKFORCE,-	DSU Mode: One bit per complex clock. Any unused bits are treated as RESERVED ( <b>RAZ/WI</b> ), Reserved	WO	-
[3]	CLUS_PERIPHCLKFORCE	Clock for CLUS_PERIPHCLK	WO	-
[2]	CLUS_GICCLKFORCE	Clock for CLUS_GICCLK	WO	-
[1]	CLUS_PPUCLKFORCE	Clock for CLUS_PPUCLK	WO	-
[0]	Direct Connect Mode: CLUS_CORECLKFORCE DSU Mode: RESERVED	Direct Connect Mode: Clock force for CORECLK DSU Mode: Reserved	WO RAZ/ WI	-

### 7.4.8.17 CAP3, Core Capabilities 3 register

Core Capabilities 3.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

RO

##### Address offset

0x0FB4

##### Type

RO

##### Reset value

0x00000000

## Bit descriptions

**Table 7-199: CAP3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	CPUPLL_IMPLEMENTED	Shows whether dedicated CPUPLL is implemented in the design <ul style="list-style-type: none"> <li>• 0 - Dedicated CPUPLL not present</li> <li>• 1 - Dedicated CPUPLL present</li> </ul>	RO	0b1

## 7.4.8.18 CAP2, Core Capabilities 2 register

Core Capabilities 2.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

RO

#### Address offset

0x0FB8

## Bit descriptions

**Table 7-200: CAP2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	-		RO	-
[1:0]	THREADS_CORE0	Number of threads per CPU core <ul style="list-style-type: none"> <li>0 for 1 thread on CPU</li> <li>1 for 2 threads on CPU</li> <li>Other values are reserved</li> </ul>	RO	-

## 7.4.8.19 CAP1, Core Capabilities 1 register

Core Capabilities 1.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

RO

### Address offset

0x0FBC

### Bit descriptions

**Table 7-201: CAP1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	NUM_PE	Number of CPU Processing Elements 0 for 1 PE 1 for 2 PEs ... 15 for 16 PEs	RO	-
[27:2]	RESERVED	Reserved	RAZ/WI	-
DSU Mode: [27:9]				
Direct Connect Mode: [1]	Direct Connect Mode: COREOSYNC	Direct Connect Mode: Indicates if the CPU core 0 is synchronous to the Cluster Clock	RO	-
DSU Mode: [8:1]	DSU Mode: COMPLEX<n>SYNC	DSU Mode: Indicates if COMPLEX<n> is synchronous to the Cluster Clock <ul style="list-style-type: none"> <li>• 0 - Core is ASYNC to the cluster</li> <li>• 1 - Core is SYNC to the cluster</li> </ul>		
[0]	CLUSSYNC	Indicates if the cluster is synchronous to the Interconnect or not <ul style="list-style-type: none"> <li>• 0 - Cluster is ASYNC to the Interconnect</li> <li>• 1 - Cluster is SYNC to the Interconnect</li> </ul> Support only SYNC for direct connect mode.	RO	-

### 7.4.8.20 PWR\_CTRL\_CONFIG, Power Control Configuration register

The power controller logical ID value is captured in this register. The value depends on the chosen configuration.

Value dependent upon chosen configuration.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Core Manager and clock control registers](#)

##### Address offset

0x0FC0

##### Type

RO

##### Reset value

0x0014\_0000

#### Bit descriptions

**Table 7-202: PWR\_CTRL\_CONFIG bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	PCL_ID	Power Control Logical ID. This field is set to 0x0014.	RO	-
[15:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	NO_OF_PPU	Defines the number of PPUs in the power control logic.  This is set to 0 as the PPUs are inside the Core.	RO	-

### 7.4.8.21 PID4, Power Control Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FD0

**Type**

RO

**Reset value**

0x00000044

**Bit descriptions****Table 7-203: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	4KB_count	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x4
[3:0]	jep106_c_code	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

**7.4.8.22 PID5, Power Control Peripheral ID 5 register**

The PID5 register is Reserved.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FD4

**Type**

RO

**Reset value**

0x00000000

## Bit descriptions

**Table 7-204: PID5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.8.23 PID6, Power Control Peripheral ID 6 register

The PID6 register is Reserved.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Core Manager and clock control registers

#### Address offset

0x0FD8

#### Type

RO

#### Reset value

0x00000000

## Bit descriptions

**Table 7-205: PID6 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.8.24 PID7, Power Control Peripheral ID 7 register

The PID7 register is Reserved.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FDC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-206: PID7 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

**7.4.8.25 PID0, Power Control Peripheral ID 0 register**

The PID0 register contains the first eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FE0

**Type**

RO

**Reset value**

0x000000B8

**Bit descriptions****Table 7-207: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/ WI	-
[7:0]	part_number_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xB8

### 7.4.8.26 PID1, Power Control Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Core Manager and clock control registers](#)

##### Address offset

0x0FE4

##### Type

RO

##### Reset value

0x000000B0

#### Bit descriptions

**Table 7-208: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	jep106_id_3_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	part_number_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

### 7.4.8.27 PID2, Power Control Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FE8

**Type**

RO

**Reset value**

0x0000000B

**Bit descriptions****Table 7-209: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	Revision	Specifies the major revision number for the block. The value is defined by the implementation.	RO	0x0
[3]	jedec_used	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0b1
[2:0]	jep106_id_6_4	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

**7.4.8.28 PID3, Power Control Peripheral ID 3 register**

The PID3 register provides information about any modifications to the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FEC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-210: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.8.29 ID0, Power Control Component ID 0 register

The ID0 register contains segment 0 of the power control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x0FF0

##### Type

RO

##### Reset value

0x0000000D

#### Bit descriptions

**Table 7-211: ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_0	Specifies segment 0 of the code that identifies the power control component class. Reads as 0x0D.	RO	0x0D

### 7.4.8.30 ID1, Power Control Component ID 1 register

The ID1 register contains segment 1 of the power control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FF4

**Type**

RO

**Reset value**

0x000000F0

**Bit descriptions****Table 7-212: ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_1	Specifies segment 1 of the code that identifies the power control component class. Reads as 0xF0.	RO	0xF0

**7.4.8.31 ID2, Power Control Component ID 2 register**

The ID2 register contains segment 2 of the power control component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Core Manager and clock control registers

**Address offset**

0x0FF8

**Type**

RO

**Reset value**

0x00000005

**Bit descriptions****Table 7-213: ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_2	Specifies segment 2 of the code that identifies the power control component class. Reads as 0x05.	RO	0x05

### 7.4.8.32 ID3, Power Control Component ID 3 register

The ID3 register contains segment 3 of the power control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Core Manager and clock control registers

##### Address offset

0x0FFC

##### Type

RO

##### Reset value

0x000000B1

#### Bit descriptions

**Table 7-214: ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_3	Specifies segment 3 of the code that identifies the power control component class. Reads as 0xB1.	RO	0xB1

### 7.4.9 System Power Integration Kit registers

The System PIK registers enable configuration of clock settings for the system Power Integration Kit.

The system PIK occupies a 64KB address space, which is split into 4KB blocks as shown in the following table.

**Table 7-215: System PIK address offsets**

Offset	Name	Description
0x0000	PIK Control Registers	Clock and pseudo-static control signals for PD_SYSTOP clocks.
0x1000	SYSTOP_PPU0	Power Policy Unit that is responsible for all SYSTOP system logic. For register information, see the <i>Arm® Power Policy Unit Architecture Specification, version 1.1</i> .
0x2000 - 0xFFFF	RESERVED	Reserved

The following table lists the registers for the system PIK.

**Table 7-216: System PIK register summary**

Offset	Name	Description	Type	Reset	Width
0x000 - 0x81C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x820	INTCLK_CTRL	Cache Coherent Interconnect Clock (INTCLK) Control.	RW	0x0000_0101	32-bit
0x824	INTCLK_DIV1	Cache Coherent Interconnect Clock (INTCLK) Divider Control.	RW	0x000F_000F	32-bit
0x828 - 0x84F	RESERVED	Reserved	RAZ/WI	-	32-bit
0x850	GICCLK_CTRL	GIC Clock (GICCLK) Control.	RW	0x0000_0101	32-bit
0x854	GICCLK_DIV1	GIC Clock (GICCLK) Divider Control.	RW	0x000F_000F	32-bit
0x858 - 0x85F	RESERVED	Reserved	RAZ/WI	-	32-bit
0x860	SCPPIKCLK_CTRL	SCP PIK Clock (SCPPIKCLK) Control.	RW	0x0000_0101	32-bit
0x864	SCPPIKCLK_DIV1	SCP PIK Clock (SCPPIKCLK) Divider Control.	RW	0x000F_000F	32-bit
0x868 - 0x86F	RESERVED	Reserved	RAZ/WI	-	32-bit
0x870	SYSPERCLK_CTRL	System Peripheral Clock (SYSPERCLK) Control.	RW	0x0000_0101	32-bit
0x874	SYSPERCLK_DIV1	System Peripheral Clock (SYSPERCLK) Divider Control.	RW	0x000F_000F	32-bit
0x878 - 0x87F	RESERVED	Reserved	RAZ/WI	-	32-bit
0x880	DMCCLK_CTRL	DMC Clock Control. Only applicable if the DMC controller is integrated inside the CSS.	RW	0x0000_0001	32-bit
0x884	DMCCLK_DIV1	DMC Clock Divider Control. Only applicable if the DMC controller is integrated inside the CSS.	RW	0x0000_001F	32-bit
0x888 - 0x89F	RESERVED	Reserved	RAZ/WI	-	32-bit
0x8A0	APUARTCLK_CTRL	UART Clock (APUARTCLK) Control.	RW	0x0000_0101	32-bit
0x8A4	APUARTCLK_DIV1	UART Clock (APUARTCLK) Divider Control.	RW	0x001F_001F	32-bit
0x8A8 - 0x8AF	RESERVED	Reserved	RAZ/WI	-	32-bit
0x8B0	IONCICLICK_CTRL	I/O NI-700 Network Clock (IO_MACRO.NCICLICK) Control.	RW	0x0000_0101	32-bit
0x8B4	IONCICLICK_DIV1	I/O NI-700 Network Clock (IO_MACRO.NCICLICK) Divider Control.	RW	0x000F_000F	32-bit
0x8B8 - 0x8FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x900	TCU<x>CLK_CTRL	TCU Clock (IO_MACRO.TCUCLK) Control.	RW	0x0000_0101	32-bit
0x904	TCU<x>CLK_DIV1	TCU Clock (IO_MACRO.TCUCLK) Divider Control.	RW	0x001F_001F	32-bit
0x908 - 0x93C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x940	TCUx_CLK_ENABLE	Control register to disable or clock gate the respective TCU clocks.	RW	0x0000_00FF	32-bit
0x944	NCIx_CLK_ENABLE	Control register to disable or clock gate the respective NI-700 clocks.	RW	0x0000_00FF	32-bit
0x940 - 0x9FF	RESERVED	Reserved	RAZ/WI	-	32-bit
0xA00	CLKFORCE_STATUS	System PIK Clock Force Status.	RO	-	32-bit

Offset	Name	Description	Type	Reset	Width
0xA04	CLKFORCE_SET	System PIK Clock Force Set.	WO	-	32-bit
0xA08	CLKFORCE_CLR	System PIK Clock Force Clear.	WO	0x0000_0000	32-bit
0xA0C - 0xB0C	RESERVED	Reserved	RAZ/WI	-	32-bit
0xB10	IOMACRO_OVERRIDE	IOMacro Override.	RW	0x0000_0000	32-bit
0xB10 - 0xFB0	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFC0	PIK_CONFIG	System PIK Configuration.	RO	0x0026_0001	32-bit
0xFC4 - 0xFCF	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFD0	PID4	System PIK Peripheral ID 4.	RO	0x0000_0044	32-bit
0xFD4	PID5	System PIK Peripheral ID 5.	RO	0x0000_0000	32-bit
0xFD8	PID6	System PIK Peripheral ID 6.	RO	0x0000_0000	32-bit
0xFDC	PID7	System PIK Peripheral ID 7.	RO	0x0000_0000	32-bit
0xFE0	PID0	System PIK Peripheral ID 0.	RO	0x0000_00B8	32-bit
0xFE4	PID1	System PIK Peripheral ID 1.	RO	0x0000_00B0	32-bit
0xFE8	PID2	System PIK Peripheral ID 2.	RO	0x0000_000B	32-bit
0xFEC	PID3	System PIK Peripheral ID 3.	RO	0x0000_0000	32-bit
0xFF0	ID0	System PIK Component ID 0.	RO	0x0000_000D	32-bit
0xFF4	ID1	System PIK Component ID 1.	RO	0x0000_00F0	32-bit
0xFF8	ID2	System PIK Component ID 2.	RO	0x0000_0005	32-bit
0xFFC	ID3	System PIK Component ID 3.	RO	0x0000_00B1	32-bit

#### 7.4.9.1 INTCLK\_CTRL, Cache Coherent Interconnect Clock Control register

This register provides the ability to program the number of clock cycles between the INTCLK not being required and the request to dynamically clock gate it. The clock source of the INTCLK can also be programmed through this register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

System Power Integration Kit registers

###### Address offset

0x820

###### Type

RW

**Reset value**

0x0000\_0101

**Bit descriptions****Table 7-217: INTCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - INTPLLCLK</p> <p>Other values are RESERVED</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - INTPLLCLK</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

**7.4.9.2 INTCLK\_DIV1, Cache Coherent Interconnect Clock Divider Control register**

This register provides the ability to request a new clock divider value on the source clock (INTPLLCLK) to generate the required output clock (INTCLK). The current divider value can also be read out from this register.

**Configurations**

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

### Address offset

0x824

### Type

RW

### Reset value

0x000F\_000F

## Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-218: INTCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

## 7.4.9.3 GICCLK\_CTRL, GIC Clock Control register

This register provides the ability to program the number of clock cycles between the GICCLK not being required and the request to dynamically clock gate it. The clock source of the GICCLK can also be programmed through this register.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

**Address offset**

0x850

**Type**

RW

**Reset value**

0x0000\_0101

**Bit descriptions****Table 7-219: GICCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This field is not used if the clock gating is not implemented for respective clock.</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

#### 7.4.9.4 GICCLK\_DIV1, GIC Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate the required output clock (GICCLK). The current divider value can also be read out from this register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[System Power Integration Kit registers](#)

###### Address offset

0x854

###### Type

RW

###### Reset value

0x000F\_000F

##### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-220: GICCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1, e.g. setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

#### 7.4.9.5 SCPPIKCLK\_CTRL, SCP APB Clock Control register

This register provides the ability to program the number of clock cycles between the SCPPIKCLK not being required and the request to dynamically clock gate it. The clock source of the SCPPIKCLK can also be programmed through this register.

##### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

### Address offset

0x860

### Type

RW

### Reset value

0x0000\_0101

## Bit descriptions

**Table 7-221: SCPPIKCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This field is not used if the clock gating is not implemented for respective clock.</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED</p>	RW	-

Bits	Name	Description	Type	Reset
[7:0]	CLKSELECT	Selects the clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - SYSPLLCLK  Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.	RW	-

#### 7.4.9.6 SCPPIKCLK\_DIV1, SCP APB Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate the required output clock (SCPPIKCLK). The current divider value can also be read out from this register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[System Power Integration Kit registers](#)

###### Address offset

0x864

###### Type

RW

###### Reset value

0x000F\_000F

##### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-222: SCPPIKCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F

Bits	Name	Description	Type	Reset
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.9.7 SYSPERCLK\_CTRL, System Peripheral Clock Control register

This register provides the ability to program the number of clock cycles between the SYSPERCLK not being required and the request to dynamically clock gate it. The clock source of the SYSPERCLK can also be programmed through this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x870

##### Type

RW

##### Reset value

0x0000\_0101

#### Bit descriptions

**Table 7-223: SYSPERCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	Number of clock cycles between the clock not being required and the request to dynamically clock gate it.  0x0 - No cycles 0x1 - 1 cycle ... 0xFF - 255 cycles  This field is not used if the clock gating is not implemented for respective clock.  This is reserved if dynamic clock gating is not implemented.	RW	-

Bits	Name	Description	Type	Reset
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - APSYSPLLCLK  Other values are RESERVED	RW	-
[7:0]	CLKSELECT	Selects the clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - APSYSPLLCLK  Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.	RW	-

### 7.4.9.8 SYSPERCLK\_DIV1, System Peripheral Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate required output clock (SYSPERCLK). The current divider value can also be read out from this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x874

##### Type

RW

##### Reset value

0x000F\_000F

#### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-224: SYSPERCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.9.9 DMCCLK\_CTRL, DMC Clock Control register

DMC Clock Control.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x880

##### Type

RW

##### Reset value

0x0000\_0001

## Bit descriptions

**Table 7-225: DMCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:17]	RESERVED	Reserved	RAZ/WI	-
16	dmcclk_1xclkbypassdiv2	<p>CLKCTRL_DMCLK_1XCLKBYPASSDIV2 : control bit for ClkMux on 1x and 2x clock after the divider to bypass the divider, in the DMC clock selection. Default set to 1'b0.</p> <p>0 - div2 applied. DMCLK1x is <math>\frac{1}{2}</math> the frequency of DMCLK2x. (Default)</p> <p>1 - div2 bypassed. DMCLK1x is the same frequency as DMCLK2x.</p>	RW	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - DDRPLL</p> <p>Other values are RESERVED</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - DDRPLL</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

### 7.4.9.10 DMCLK\_DIV1, DMC Clock Divider Control register

DMC Clock Divider Control.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

### Address offset

0x884

### Type

RW

### Reset value

0x0000\_001F

## Bit descriptions

**Table 7-226: DMCCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RW	-
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the DMCCLK  The divider value is the value of CLKDIV + 1, for example, setting a value of 0 indicates a divider value of 1.	RW	-

## 7.4.9.11 APUARTCLK\_CTRL, UART Clock Control register

This register provides the ability to program the number of clock cycles between the APUARTCLK not being required and the request to dynamically clock gate it. The clock source of the APUARTCLK can also be programmed through this register.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

**Address offset**

0x8A0

**Type**

RW

**Reset value**

0x0000\_0101

**Bit descriptions****Table 7-227: APUARTCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This field is not used if the clock gating is not implemented for respective clock.</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - APSYSPLLCLK</p> <p>Other values are RESERVED</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - APSYSPLLCLK</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

### 7.4.9.12 APUARTCLK\_DIV1, UART Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (APSYSPLLCLK) to generate the required output clock (APUARTCLK). The current divider value can also be read out from this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x8A4

##### Type

RW

##### Reset value

0x001F\_001F

#### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-228: APUARTCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x1F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x1F

### 7.4.9.13 IONCICLK\_CTRL, I/O NCI Network Clock Control register

This register provides the ability to program the number of clock cycles between the IONCICLK not being required and the request to dynamically clock gate it. The clock source of the IONCICLK can also be programmed through this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x8B0

##### Type

RW

##### Reset value

0x0000\_0101

#### Bit descriptions

**Table 7-229: IONCICLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This field is not used if the clock gating is not implemented for respective clock.</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - SYSPLLCLK  Other values are RESERVED	RW	-
[7:0]	CLKSELECT	Selects the clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - SYSPLLCLK  Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.	RW	-

#### 7.4.9.14 IONCICLK\_DIV1, I/O NCI Network Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate the required output clock (IONCICLK). The current divider value can also be read out from this register.

##### Configurations

This register is available in all configurations.

##### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x8B4

##### Type

RW

##### Reset value

0x000F\_000F

##### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-230: IONCICLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.9.15 TCU<x>CLK\_CTRL, TCU Clock Control register

This register provides the ability to program the number of clock cycles between the TCU<x>CLK not being required and the request to dynamically clock gate it. The clock source of the TCU<x>CLK can also be programmed through this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0x900

##### Type

RW

##### Reset value

0x0000\_0101

## Bit descriptions

**Table 7-231: TCU<x>CLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This field is not used if the clock gating is not implemented for respective clock.</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

### 7.4.9.16 TCU<x>CLK\_DIV1, TCU Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate required output clock (TCU<x>CLK). The current divider value can also be read out from this register.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[System Power Integration Kit registers](#)

### Address offset

0x904

### Type

RW

### Reset value

0x000F\_000F

## Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-232: TCU<x>CLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

## 7.4.9.17 TCUx\_CLK\_ENABLE, TCU Clock Enable register

This control register provides the ability to disable or clock gate the respective TCU clocks.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[System Power Integration Kit registers](#)

### Address offset

0x940

**Type**

RW

**Reset value**

0x0000\_00FF

**Bit descriptions****Table 7-233: TCUx\_CLK\_ENABLE bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	0x0
[7:0]	CLK_ENABLE	<ul style="list-style-type: none"> <li>0 - Disable clock</li> <li>1 - Enable clock</li> </ul>	RW	0xFF

**7.4.9.18 NCIx\_CLK\_ENABLE, NCI Clock Enable register**

This control register provides the ability to disable or clock gate the respective NCI clocks.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System Power Integration Kit registers](#)**Address offset**

0x944

**Type**

RW

**Reset value**

0x0000\_00FF

**Bit descriptions****Table 7-234: NCI\_CLK\_ENABLE bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	CLK_ENABLE	<ul style="list-style-type: none"> <li>0 - Disable clock</li> <li>1 - Enable clock</li> </ul>	RW	0xFF

## 7.4.9.19 CLKFORCE\_STATUS, System PIK Clock Force Status register

This register captures the status (enabled/disabled) of the dynamic clock gating on GICCLK, SCPPIKCLK, SYSPERCLK, IONCICLK, and TCUCLK.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[System Power Integration Kit registers](#)

#### Address offset

0xA00

#### Type

RO

### Bit descriptions

If a bit reads back as 1 then the associated dynamic clock gating associated with the clock is disabled otherwise it is enabled.

If dynamic clock gating is not implemented for a particular clock, the corresponding clkforce bit is reserved.

**Table 7-235: CLKFORCE\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:14]	RESERVED	Reserved	RAZ/WI	-
[13]	TCUCLKFORCE	Clock force status for (TCUCLK)	RO	-
[12]	GICCLKFORCE	Clock force status for (GICCLK)	RO	-
[11:6]	RESERVED	Reserved	RAZ/WI	-
[5]	SYSPERCLKFORCE	Clock force status for (SYSPERCLK)	RO	-
[4]	SCPPIKCLKFORCE	Clock force status for (SCPPIKCLK)	RO	-
[3]	RESERVED	Reserved	RAZ/WI	-
[2]	IONCICLKFORCE	Clock force status for (IONCICLK)	RO	-
[1:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.9.20 CLKFORCE\_SET, System PIK Clock Force Set register

This register is used to set force these dynamic gated clocks to be free-running: GICCLK, SCPPIKCLK, SYSPERCLK, IONCICLK, and TCUCLK.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[System Power Integration Kit registers](#)

#### Address offset

0xA04

#### Type

WO

### Bit descriptions

Writing a one to a bit within the CLKFORCE\_SET register disables any dynamic hardware clock gating, whilst writing zero to a bit is ignored.

**Table 7-236: CLKFORCE\_SET bit descriptions**

Bits	Name	Description	Type	Reset
[31:14]	RESERVED	Reserved	RAZ/WI	-
[13]	TCUCLKFORCE	Clock force status for (TCUCLK)	RO	-
[12]	GICCLKFORCE	Clock force status for (GICCLK)	RO	-
[11:6]	RESERVED	Reserved	RAZ/WI	-
[5]	SYSPERCLKFORCE	Clock force status for (SYSPERCLK)	RO	-
[4]	SCPPIKCLKFORCE	Clock force status for (SCPPIKCLK)	RO	-
[3]	RESERVED	Reserved	RAZ/WI	-
[2]	IONCICLKFORCE	Clock force status for (IONCICLK)	RO	-
[1:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.9.21 CLKFORCE\_CLR, System PIK Clock Force Clear register

This register is used to clear force these dynamic gated clocks to be free-running: GICCLK, SCPPIKCLK, SYSPERCLK, IONCICLK, and TCUCLK.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[System Power Integration Kit registers](#)

### Address offset

0xA08

### Type

WO

### Reset value

0x0

## Bit descriptions

Writing a one to a bit within the CLKFORCE\_CLR register enables the dynamic hardware clocking gating, whilst writing zero to a bit is ignored. The bit allocation is the same as the [CLKFORCE\\_STATUS](#) register.

**Table 7-237: CLKFORCE\_CLR bit descriptions**

Bits	Name	Description	Type	Reset
[31:14]	RESERVED	Reserved	RAZ/WI	-
[13]	TCUCLKFORCE	Clock force status for (TCUCLK)	RO	-
[12]	GICCLKFORCE	Clock force status for (GICCLK)	RO	-
[11:6]	RESERVED	Reserved	RAZ/WI	-
[5]	SYSUPERCLKFORCE	Clock force status for (SYSUPERCLK)	RO	-
[4]	SCPPIKCLKFORCE	Clock force status for (SCPPIKCLK)	RO	-
[3]	RESERVED	Reserved	RAZ/WI	-
[2]	IONCICLKFORCE	Clock force status for (IONCICLK)	RO	-
[1:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.9.22 IOMACRO\_OVERRIDE, IOMacro Override register

This register provides SW override capability for the TBU's utlb\_roundrobin and sec\_roundrobin inputs.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

**Functional group**[System Power Integration Kit registers](#)**Address offset**

0xB10

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-238: IOMACRO\_OVERRIDE bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	utlb_roundrobin	Software override signal for TBUs utlb_roundrobin input tie-off. Software needs to be set this bit before releasing the reset to the TBU as this sample coming out of reset.	RW	0x0
[0]	Sec_override	Software override signal for TCU sec_override input tie-off. Software needs to be set this bit before releasing the reset to the TBU as this sample coming out of reset.	RW	0x0

### 7.4.9.23 PIK\_CONFIG, System PIK Configuration register

This is a Power Control Logic related configuration register.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System Power Integration Kit registers](#)**Address offset**

0xFC0

**Type**

RO

**Reset value**

0x0026\_0001

## Bit descriptions

**Table 7-239: PIK\_CONFIG bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	PCL_ID	Power Control Logical ID. This field is set to 0x0026.	RO	0x0026
[15:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	no_of_ppu	Defines the number of PPUs in the power control logic.  This value is set to indicate number of PPUs. The values is dependent on the number PPUs implemented in the subsystem. This reads back as 1.	RO	0x1

## 7.4.9.24 PID4, System PIK Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[System Power Integration Kit registers](#)

#### Address offset

0xFD0

#### Type

RO

#### Reset value

0x0000\_0044

## Bit descriptions

**Table 7-240: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	4KB_count	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x4
[3:0]	jep106_c_code	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code.	RO	0x4

### 7.4.9.25 PID5, System PIK Peripheral ID 5 register

The PID5 register is Reserved.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0xFD4

##### Type

RO

##### Reset value

0x0000\_0000

#### Bit descriptions

**Table 7-241: PID5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.9.26 PID6, System PIK Peripheral ID 6 register

The PID6 register is Reserved.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0xFD8

##### Type

RO

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-242: PID6 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

[7.4.9.27 PID7, System PIK Peripheral ID 7 register](#)

The PID7 register is Reserved.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System Power Integration Kit registers](#)**Address offset**

0xFDC

**Type**

RO

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-243: PID7 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

[7.4.9.28 PID0, System PIK Peripheral ID 0 register](#)

The PID0 register contains the first eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

### Address offset

0xFE0

### Type

RO

### Reset value

0x0000\_00B8

## Bit descriptions

**Table 7-244: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	part_number_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xB8

## 7.4.9.29 PID1, System PIK Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

System Power Integration Kit registers

### Address offset

0xFE4

### Type

RO

### Reset value

0x0000\_00B0

## Bit descriptions

**Table 7-245: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	jep106_id_3_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	part_number_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

## 7.4.9.30 PID2, System PIK Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[System Power Integration Kit registers](#)

#### Address offset

0xFE8

#### Type

RO

#### Reset value

0x0000\_000B

## Bit descriptions

**Table 7-246: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	Revision	Specifies the major revision number for the block. For r0p0, the value is defined by the implementation.	RO	0x0
[3]	jedec_used	Specifies whether the JEDEC JEP106 identification scheme is in use. Always reads as 0x1.	RO	0b1
[2:0]	jep106_id_6_4	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

### 7.4.9.31 PID3, System PIK Peripheral ID 3 register

The PID3 register provides information about any modifications to the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0xFEC

##### Type

RO

##### Reset value

0x00000000

#### Bit descriptions

**Table 7-247: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.9.32 ID0, System PIK Component ID 0 register

The ID0 register contains segment 0 of the system PIK component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0xFF0

**Type**

RO

**Reset value**

0x0D

**Bit descriptions****Table 7-248: ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_0	Specifies segment 0 of the code that identifies the system PIK component class. Reads as 0x0D.	RO	-

**7.4.9.33 ID1, System PIK Component ID 1 register**

The ID1 register contains segment 1 of the system PIK component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[System Power Integration Kit registers](#)**Address offset**

0xFF4

**Type**

RO

**Reset value**

0xF0

**Bit descriptions****Table 7-249: ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_1	Specifies segment 1 of the code that identifies the system PIK component class. Reads as 0xF0.	RO	-

### 7.4.9.34 ID2, System PIK Component ID 2 register

The ID2 register contains segment 2 of the system PIK component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0xFF8

##### Type

RO

##### Reset value

0x05

#### Bit descriptions

**Table 7-250: ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_2	Specifies segment 2 of the code that identifies the system PIK component class. Reads as 0x05.	RO	-

### 7.4.9.35 ID3, System PIK Component ID 3 register

The ID3 register contains segment 3 of the system PIK component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[System Power Integration Kit registers](#)

##### Address offset

0xFFC

##### Type

RO

**Reset value**

0xB1

**Bit descriptions****Table 7-251: ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_3	Specifies segment 3 of the code that identifies the system PIK component class. Reads as 0xB1.	RO	-

## 7.4.10 Debug Power Integration Kit registers

The Debug PIK registers enable configuration of various settings for the debug subsystem.

**Table 7-252: Debug PIK register summary**

Offset	Name	Description	Type	Reset	Width
0x000	DBG_PWR_REQ_ST	Debug Power Request Status.	RO	0x0000_0000	32-bit
0x04	DBG_PWR_ACK	Debug Power Request Acknowledge.	RW	0x0000_0000	32-bit
0x08 – 0x1C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x020	DBG_RST_REQ_ST	Debug Reset Request Status.	RO	0x0000_0000	32-bit
0x024	DBG_RST_ACK	Debug Reset Request Acknowledge.	RW	0x0000_0000	32-bit
0x028 – 0x02C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x030	SYS_PWR_REQ_ST	System Power-up Request Status.	RO	0x0000_0000	32-bit
0x034	SYS_PWR_ACK	System Power-up Request Acknowledge.	RW	0x0000_0000	32-bit
0x038 – 0x04C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x050	SYS_RST_REQ_ST	System Reset Request Status.	RO	0x0000_0000	32-bit
0x054	SYS_RST_ACK	System Reset Request Acknowledge.	RW	0x0000_0000	32-bit
0x054 – 0x7FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x810	TRACECLK_CTRL	TPIU Clock Control. Only if TPIU is implemented.	RW	0x0000_0101	32-bit
0x814	TRACECLK_DIV1	TPIU Clock Divider Control. Only if TPIU is implemented.	RW	0x001F_001F	32-bit
0x818 – 0x81C	RESERVED	Reserved	RAZ/WI	--	32-bit
0x820	SYSDBGCLK_CTRL	Debug APB Clock Control.	RW	0x0000_0101	32-bit
0x824	SYSDBGCLK_DIV1	Debug APB Clock Divider Control.	RW	0x000F_000F	32-bit
0x828 – 0x82C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x830	DBGCLK_CTRL	Debug Clock (DBGCLK) Control	RW	0x0000_0101	32-bit
0x834	DBGCLK_DIV1	Debug Clock (DBGCLK) Divider	RW	0x000F_000F	32-bit
0x838 – 0x83C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x840 – 0xA0C	RESERVED	Reserved	RAZ/WI	-	32-bit
0xA00	CLKFORCE_STATUS	Debug PIK Clock Force Status.	RO	-	32-bit
0xA04	CLKFORCE_SET	Debug PIK Clock Force Set.	WO	-	32-bit
0xA08	CLKFORCE_CLR	Debug PIK Clock Force Clear.	WO	-	32-bit
0xB00	DBG_PWR_REQ_INT_ST	Debug Power Request Interrupt Status.	RW1C	0x0000_0000	32-bit

Offset	Name	Description	Type	Reset	Width
0xB04	DBG_RST_REQ_INT_ST	Debug Reset Request Interrupt Status.	RW1C	0x0000_0000	32-bit
0xB08 – 0xB1C	RESERVED	Reserved	RAZ/WI	-	32-bit
0xB20	SYS_PWR_REQ_INT_ST	System Power Request Interrupt Status.	RW1C	0x0000_0000	32-bit
0xB24	SYS_RST_REQ_INT_ST	System Reset Request Interrupt Status.	RW1C	0x0000_0000	32-bit
0xB28 – 0xBEC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xBF0	DAP_TARGET_ID	Debug Access Port Target ID	RO	CFG_DEF	32-bit
0xBF4	DAP_INSTANCE_ID	Debug Access Port Instance ID	RO	CFG_DEF	32-bit
0xBF8 – 0xFB8	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFBC	CAPO	Debug PIK Capability 0.	RO	0x0000_8421	32-bit
0xFC0	PCL_CFG	Power Control Logic Configuration.	RO	0x0034_0000	32-bit
0xFD0	PID4	Debug PIK Peripheral ID 4.	RO	0x0000_0044	32-bit
0xFD4	PID5	Debug PIK Peripheral ID 5.	RO	0x0000_0000	32-bit
0xFD8	PID6	Debug PIK Peripheral ID 6.	RO	0x0000_0000	32-bit
0xFDC	PID7	Debug PIK Peripheral ID 7.	RO	0x0000_0000	32-bit
0xFE0	PID0	Debug PIK Peripheral ID 0.	RO	0x0000_00B8	32-bit
0xFE4	PID1	Debug PIK Peripheral ID 1.	RO	0x0000_00B0	32-bit
0xFE8	PID2	Debug PIK Peripheral ID 2.	RO	0x0000_000B	32-bit
0xFEC	PID3	Debug PIK Peripheral ID 3.	RO	0x0000_0000	32-bit
0xFF0	ID0	Debug PIK Component ID 0.	RO	0x0000_000D	32-bit
0xFF4	ID1	Debug PIK Component ID 1.	RO	0x0000_00F0	32-bit
0xFF8	ID2	Debug PIK Component ID 2.	RO	0x0000_0005	32-bit
0xFFC	ID3	Debug PIK Component ID 3.	RO	0x0000_00B1	32-bit
0x1000 – 0x8FFC	RESERVED	Reserved	RAZ/WI	-	32-bit

#### 7.4.10.1 DBG\_PWR\_REQ\_ST, Debug Power Request Status register

This register captures current status of the CDBGPWRUPREQx.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

Debug Power Integration Kit registers

###### Address offset

0x000

###### Type

RO

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-253: DBG\_PWR\_REQ\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	-	Reserved	RO	0x0
[0]	DBGWRREQ_ST	Current status of the CDBGWRUPREQ	RO	0x0

**7.4.10.2 DBG\_PWR\_ACK, Debug Power Request Acknowledge register**

The value of the CDBGWRUPACKx can be set using this register.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Debug Power Integration Kit registers

**Address offset**

0x04

**Type**

RW

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-254: DBG\_PWR\_ACK bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	DBGWRACK	Set the value of the CDBGWRUPACK	RW	0x0

**7.4.10.3 DBG\_RST\_REQ\_ST, Debug Reset Request Status register**

This register captures current status of the CDBGRESETREQ.

**Configurations**

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[Debug Power Integration Kit registers](#)

### Address offset

0x020

### Type

RO

### Reset value

0x0000\_0000

## Bit descriptions

**Table 7-255: DBG\_RST\_REQ\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	DBGRESETREQ_ST	Status of the CDBGRESETREQ.	RO	0x0

## 7.4.10.4 DBG\_RST\_ACK, Debug Reset Request Acknowledge register

The value of the CDBGRESETACKx can be set using this register.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[Debug Power Integration Kit registers](#)

### Address offset

0x024

### Type

RW

### Reset value

0x0000\_0000

## Bit descriptions

**Table 7-256: DBG\_RST\_ACK bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	DBGRESETACK	Set the value of the CDBGRESETACKx.	RW	0x0

## 7.4.10.5 SYS\_PWR\_REQ\_ST, System Power-up Request Status register

This register captures current status of CSYSPWRUPREQ.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Debug Power Integration Kit registers](#)

#### Address offset

0x030

#### Type

RO

#### Reset value

0x0

## Bit descriptions

**Table 7-257: SYS\_PWR\_REQ\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	SYSPWRREQ_ST	Status of the CSYSPWRUPREQx.	RO	0x0

## 7.4.10.6 SYS\_PWR\_ACK, System Power-up Request Acknowledge register

The value of the CSYSPWRUPACKx can be set using this register.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0x034

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-258: SYS\_PWR\_ACK bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	SYSPWRACK	Set the value of the CSYSPWRUPACK	RW	0x0

## 7.4.10.7 SYS\_RST\_REQ\_ST, System Reset Request Status register

This register captures current status of CSYSRSTREQ.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0x050

### Type

RO

### Reset value

0x0

## Bit descriptions

**Table 7-259: SYS\_RST\_REQ\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	SYSRSTREQ_ST	Status of the CSYSRSTREQ	RO	0x0

## 7.4.10.8 SYS\_RST\_ACK, System Reset Request Acknowledge register

The value of the CSYSRSTACKx can be set using this register.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Debug Power Integration Kit registers](#)

#### Address offset

0x054

#### Type

RW

#### Reset value

0x0

## Bit descriptions

**Table 7-260: SYS\_RST\_ACK bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	SYSRSTACK	Set the value of the CSYSRSTACK.	RW	0x0

## 7.4.10.9 TRACECLK\_CTRL, TPIU Clock Control register

This register provides the ability to program the number of clock cycles between the TRACECLK not being required and the request to dynamically clock gate it. The clock source of the TRACECLK can also be programmed through this register.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[Debug Power Integration Kit registers](#)

### Address offset

0x810

### Type

RW

### Reset value

0x0000\_0101

## Bit descriptions

**Table 7-261: TTRACECLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	<p>Number of clock cycles between the clock not being required and the request to dynamically clock gate it.</p> <p>0x0 - No cycles</p> <p>0x1 - 1 cycle</p> <p>...</p> <p>0xFF - 255 cycles</p> <p>This is reserved if dynamic clock gating is not implemented.</p>	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	<p>Acknowledges the currently selected clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED</p>	RW	-
[7:0]	CLKSELECT	<p>Selects the clock source</p> <p>0000_0000 - Clock Gated</p> <p>0000_0001 - REFCLK</p> <p>0000_0010 - SYSPLLCLK</p> <p>Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.</p>	RW	-

### 7.4.10.10 TRACECLK\_DIV1, TPIU Clock Divider Control register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate required output clock (TRACECLK). The current divider value can also be read out from this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Power Integration Kit registers

##### Address offset

0x814

##### Type

RW

##### Reset value

0x000F\_000F

#### Bit descriptions

**Table 7-262: TRACECLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.10.11 SYSDBGCLK\_CTRL, Debug APB Clock Control register

This register provides the ability to program the number of clock cycles between the (SYSDBGCLK) not being required and the request to dynamically clock gate it. The clock source of (SYSDBGCLK) can also be programmed through this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

##### Address offset

0x820

##### Type

RW

##### Reset value

0x0000\_0101

#### Bit descriptions

**Table 7-263: SYSDBGCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - DBGSYSPLLCLK  Other values are RESERVED	RW	0x01
[7:0]	CLKSELECT	Selects the clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - DBGSYSPLLCLK  Other values are RESERVED. The result of writing one of the RESERVED values into this field is UNPREDICTABLE.	RW	0x01

## 7.4.10.12 SYSDBGCLK\_DIV1, Debug APB Clock Divider register

This register provides the ability to request a new clock divider value on the source clock (DBGSYSPPLLCLK) to generate the required output clock (SYSDBGCLK). The current divider value can also be read out from this register.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Debug Power Integration Kit registers

#### Address offset

0x824

#### Type

RW

#### Reset value

0x000F\_000F

### Bit descriptions

**Table 7-264: SYSDBGCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

## 7.4.10.13 DBGCLK\_CTRL, CoreSight ATB Input Expansion Clock Control register

This register provides the ability to program the number of clock cycles between the DBGCLK not being required and the request to dynamically clock gate it. The clock source of DBGCLK can also be programmed through this register.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Debug Power Integration Kit registers](#)

#### Address offset

0x830

#### Type

RW

#### Reset value

0x0000\_0101

### Bit descriptions

**Table 7-265: DBGCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	RESERVED	RESERVED	RW	-
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - SYSPLLCLK  Other values are RESERVED	RW	0x01
[7:0]	CLKSELECT	Selects the clock source  0000_0000 - Clock Gated  0000_0001 - REFCLK  0000_0010 - SYSPLLCLK  Other values are RESERVED. The result of writing one of the RESERVED values into this field is <b>UNPREDICTABLE</b> .	RW	0x01

### 7.4.10.14 DBGCLK\_DIV1, CoreSight ATB Input Expansion Clock Divider register

This register provides the ability to request a new clock divider value on the source clock (DBGSYSPLLCLK) to generate required output clock (DBGCLK). The current divider value can also be read out from this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

##### Address offset

0x834

##### Type

RW

##### Reset value

0x000F\_000F

#### Bit descriptions

**Table 7-266: DBGCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.10.15 CLKFORCE\_STATUS, Debug PIK Clock Force Status register

This register captures the status (enabled/disabled) of the dynamic clock gating on DBGCLK and SYSDBGCLK.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0xA00

### Type

RO

### Bit descriptions

If a bit reads back as 1 then the associated dynamic clock gating associated with the clock is disabled otherwise it is enabled.

If dynamic clock gating is not implemented for a particular clock, the corresponding clkforce bit is reserved.

**Table 7-267: CLKFORCE\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	DBGCLKFORCE	Clock force for DBGCLK	RO	-
[1]	SYSDBGCLKFORCE	Clock force for PCLKDBG	RO	-
[0]	TRACECLKFORCE	Clock force for TRACECLK	RO	-

## 7.4.10.16 CLKFORCE\_SET, Debug PIK Clock Force Set register

This register provides the ability to disable dynamic clock gating on DBGCLK and SYSDBGCLK.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0xA04

### Type

WO

## Bit descriptions

Writing a one to a bit within the CLKFORCE\_SET register disables any dynamic hardware clock gating, whilst writing zero to a bit is ignored. The bit allocation is the same as the CLKFORCE\_STATUS register.

If dynamic clock gating is not implemented for a particular clock, the corresponding clkforce bit is reserved.

**Table 7-268: CLKFORCE\_SET bit descriptions**

Bits	Name	Description	Type	Reset
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	DBGCLKFORCE	Clock force for DBGCLK	WO	-
[1]	SYSDBGPCLKFORCE	Clock force for PCLKDBG	WO	-
[0]	TRACECLKFORCE	Clock force for TRACECLK	WO	-

## 7.4.10.17 CLKFORCE\_CLR, Debug PIK Clock Force Clear register

This register is used to clear force DBGCLK and SYSDBGPCLK to be free-running.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

Debug Power Integration Kit registers

#### Address offset

0xA08

#### Type

WO

## Bit descriptions

Writing a one to a bit within the CLKFORCE\_CLR register enables the dynamic hardware clocking gating, whilst writing zero to a bit is ignored. The bit allocation is the same as the CLKFORCE\_STATUS register

If dynamic clock gating is not implemented for a particular clock, the corresponding clkforce bit is reserved.

**Table 7-269: CLKFORCE\_CLR bit descriptions**

Bits	Name	Description	Type	Reset
[31:3]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[2]	DBGCLKFORCE	Clock force for DBGCLK	RO	-
[1]	SYSDBGCLKFORCE	Clock force for PCLKDBG	RO	-
[0]	TRACECLKFORCE	Clock force for TRACECLK	RO	-

#### 7.4.10.18 DBG\_PWR\_REQ\_INT\_ST, Debug Power Request Interrupt Status register

This register indicates the status of CDBGPWRUPREQ interrupt. The interrupt is cleared by writing 1 to this register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

Debug Power Integration Kit registers

###### Address offset

0xB00

###### Type

RW1C

###### Reset value

0x0

##### Bit descriptions

**Table 7-270: DBG\_PWR\_REQ\_INT\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	CDBGPWRUPREQ_INT	This bit is set on any edge of CDBGPWRUPREQ signal. Writing a 1 to this bit clears it.	RW1C	0x0

#### 7.4.10.19 DBG\_RST\_REQ\_INT\_ST, Debug Reset Request Interrupt Status register

This register indicates the status of CDBGRESETREQ interrupt. The interrupt is cleared by writing 1 to this register.

##### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0xB04

### Type

RW1C

### Reset value

0x0

## Bit descriptions

**Table 7-271: DBG\_RST\_REQ\_INT\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	CDBGRESETREQ_INT	This bit is set on any edge of CDBGRESETREQx signal. Writing a 1 to this bit clears it.	RW1C	0x0

## 7.4.10.20 SYS\_PWR\_REQ\_INT\_ST, System Power Request Interrupt Status register

This register indicates the status of SYSPWRREQ interrupt. The interrupt is cleared by writing 1 to this register.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0xB20

### Type

RW1C

### Reset value

0x0

## Bit descriptions

**Table 7-272: SYS\_PWR\_REQ\_INT\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	CDBG_PWRUPREQ_INT	This bit is set on any edge of CDBG_PWRUPREQx signal. Writing a 1 to this bit clears it.	RW1C	0x0

### 7.4.10.21 SYS\_RST\_REQ\_INT\_ST, System Reset Request Interrupt Status register

This register indicates the status of SYSRSTREQ interrupt. The interrupt is cleared by writing 1 to this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Power Integration Kit registers

##### Address offset

0xB24

##### Type

RW1C

##### Reset value

0x0

## Bit descriptions

**Table 7-273: SYS\_RST\_REQ\_INT\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	CDBG_RESETREQ_INT	This bit is set on any edge of CDBG_RESETREQx signal. Writing a 1 to this bit clears it.	RW1C	0x0

### 7.4.10.22 DAP\_TARGET\_ID, Debug Access Port Target ID register

This register contains the values of DAP Target ID, Target part number and Designer ID. The value of this register depends on the SOCTARGETID signal.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0xBF0

### Type

RO

### Reset value

CFG\_DEF

## Bit descriptions

**Table 7-274: DAP\_TARGET\_ID bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	TREVISION	Target revision. The value comes from the tie-off signal DAP_TREVISION.	RO	CFG_DEF
[27:12]	TPARTNO	Target part number. The value comes from the tie-off signal DAP_TPARTNO	RO	CFG_DEF
[11:1]	TDESIGNER	Designer ID, based on 11-bit JEDEC JEP106 continuation and identity code. The value comes from the tie-off signal DAP_DESIGNER	RO	CFG_DEF
[0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.10.23 DAP\_INSTANCE\_ID, Debug Access Port Instance ID register

This register contains the value of DAP Instance ID tie-off. The value of this register depends on the SOCTARGETID signal.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Power Integration Kit registers

### Address offset

0xBF4

### Type

RO

**Reset value**

CFG\_DEF

**Bit descriptions****Table 7-275: DAP\_INSTANCE\_ID bit descriptions**

Bits	Name	Description	Type	Reset
[31:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	INSTANCEID	Controls the DAP instance ID tie-off for multi drop support. This value is IMPLEMENTATION DEFINED, but must conform to Arm® Debug Interface Architecture Specification ADIv6.0	RO	-

[7.4.10.24 CAPO, Debug PIK Capability 0 register](#)

Debug PIK Capability 0.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Debug Power Integration Kit registers

**Address offset**

0xFBC

**Type**

RO

**Reset value**

0x0000\_8421

**Bit descriptions****Table 7-276: CAPO bit descriptions**

Bit	Name	Description	Type	Reset
[31:20]	RESERVED	Reserved	RAZ/WI	-
[19:15]	NUM_CSYSRST	Number of systems reset requests	RO	0b00001
[14:10]	NUM_CDBGRST	Number of debug domain reset requests	RO	0b00001
[9:5]	NUM_CSYSPWR	Number of system power domain power up requests	RO	0b00001
[4:0]	NUM_CDBGPWR	Number of debug power domain power up requests	RO	0b00001

### 7.4.10.25 PCL\_CFG, Power Control Logic Configuration register

This register provides the Logical ID for Debug PIK to distinguish from different PIKs. It also provides information about number of PPUs within this PIK.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

##### Address offset

0xFC0

##### Type

RO

##### Reset value

0x00330000

#### Bit descriptions

**Table 7-277: PCL\_CFG bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	PCL_ID	Power Control Logical ID. This reads 0x0034	RO	0x0033
[15:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	NUM_OF_PPUs	Number of PPUs.	RO	0x0

### 7.4.10.26 PID4, Debug PIK Peripheral ID 4 register

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

**Address offset**

0xFD0

**Type**

RO

**Reset value**

0x0000\_0044

**Bit descriptions****Table 7-278: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	4KB_count	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.	RO	0x4
[3:0]	jep106_c_code	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code. Reads as 0x4.	RO	0x4

**7.4.10.27 PID5, Debug PIK Peripheral ID 5 register**

The PID5 register is Reserved.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Debug Power Integration Kit registers](#)**Address offset**

0xFD4

**Type**

RO

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-279: PID5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.10.28 PID6, Debug PIK Peripheral ID 6 register

The PID6 register is Reserved.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

##### Address offset

0xFD8

##### Type

RO

##### Reset value

0x0000\_0000

#### Bit descriptions

**Table 7-280: PID6 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.10.29 PID7, Debug PIK Peripheral ID 7 register

The PID7 register is Reserved.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

##### Address offset

0xFDC

##### Type

RO

**Reset value**

0x0000\_0000

**Bit descriptions****Table 7-281: PID7 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

**7.4.10.30 PID0, Debug PIK Peripheral ID 0 register**

The PID0 register contains the first eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Debug Power Integration Kit registers](#)**Address offset**

0xFE0

**Type**

RO

**Reset value**

0x0000\_00B8

**Bit descriptions****Table 7-282: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	part_number_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xB8

### 7.4.10.31 PID1, Debug PIK Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Power Integration Kit registers

##### Address offset

0xFE4

##### Type

RO

##### Reset value

0x0000\_00B0

#### Bit descriptions

**Table 7-283: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	jep106_id_3_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	0xB
[3:0]	part_number_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

### 7.4.10.32 PID2, Debug PIK Peripheral ID 2 register

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Power Integration Kit registers

**Address offset**

0xFE8

**Type**

RO

**Reset value**

0x0000\_000B

**Bit descriptions****Table 7-284: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	Revision	Specifies the major revision number for the block. The value is defined by the implementation.	RO	0x0
[3]	jedec_used	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0b1
[2:0]	jep106_id_6_4	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

**7.4.10.33 PID3, Debug PIK Peripheral ID 3 register**

The PID3 register provides information about any modifications to the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Debug Power Integration Kit registers

**Address offset**

0xFEC

**Type**

RO

**Reset value**

0x00000000

**Bit descriptions****Table 7-285: PID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0

Bits	Name	Description	Type	Reset
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.10.34 ID0, Debug PIK Component ID 0 register

The ID0 register contains segment 0 of the debug PIK component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Power Integration Kit registers

##### Address offset

0xFFE0

##### Type

RO

##### Reset value

0x0D

#### Bit descriptions

**Table 7-286: ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_0	Specifies segment 0 of the code that identifies the debug PIK component class. Reads as 0x0D.	RO	0x0D

### 7.4.10.35 ID1, Debug PIK Component ID 1 register

The ID1 register contains segment 1 of the debug PIK component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Power Integration Kit registers

**Address offset**

0xFF4

**Type**

RO

**Reset value**

0xF0

**Bit descriptions****Table 7-287: ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_1	Specifies segment 1 of the code that identifies the debug PIK component class. Reads as 0xF0.	RO	0xF0

[7.4.10.36 ID2, Debug PIK Component ID 2 register](#)

The ID2 register contains segment 2 of the debug PIK component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Debug Power Integration Kit registers](#)**Address offset**

0xFF8

**Type**

RO

**Reset value**

0x05

**Bit descriptions****Table 7-288: ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_2	Specifies segment 2 of the code that identifies the debug PIK component class. Reads as 0x05.	RO	0x05

### 7.4.10.37 ID3, Debug PIK Component ID 3 register

The ID3 register contains segment 3 of the debug PIK component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Power Integration Kit registers](#)

##### Address offset

0xFFC

##### Type

RO

##### Reset value

0xB1

#### Bit descriptions

**Table 7-289: ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_3	Specifies segment 3 of the code that identifies the debug PIK component class.	RO	0xB1

### 7.4.11 Debug Chain Power Control Logic registers

The Debug Chain Power Control Logic registers provide information about the power settings for debug chains in a compute subsystem. These registers are available only when separate debug chain power or reset domains are implemented.

This register set must be implemented only when power control is supported for each debug chain in a compute subsystem. RD-N2 does not support debug chain power control.

If there are more than eight debug chains, the DBGCHn\_PPU registers are shared by groups of chains.

**Table 7-290: Debug Chain Power Control Logic register summary**

Offset	Name	Description	Type	Reset	Width
0x000	DBG_CHN_PWR_REQ_ST	Debug Chain Power Request Status	RW	0x0	32-bit
0x004	DBG_CHN_PWR_ACK	Debug Chain Power Request Acknowledge	RO	0x0	32-bit

Offset	Name	Description	Type	Reset	Width
0x008 - 0xAFC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xB00	DBG_CHN_PPU_INT_ST	Debug Chain PPU Interrupt Status	RO	0x0	32-bit
0xB04	DBG_CHN_PWRUP_INT_ST	Debug Chain Power-up Interrupt Status	RW1C	0x0	32-bit
0xB08 - 0xFBC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFC0	PCL_CFG	Power Control Logic Configuration	RO	IMPLEMENTATION DEFINED	32-bit
0xFD0	PID4	Debug Chain Power Control Peripheral ID 4	RO	IMPLEMENTATION DEFINED	32-bit
0xFD4	PID5	Debug Chain Power Control Peripheral ID 5	RO	IMPLEMENTATION DEFINED	32-bit
0xFD8	PID6	Debug Chain Power Control Peripheral ID 6	RO	IMPLEMENTATION DEFINED	32-bit
0xFDC	PID7	Debug Chain Power Control Peripheral ID 7	RO	IMPLEMENTATION DEFINED	32-bit
0xFE0	PID0	Debug Chain Power Control Peripheral ID 0	RO	IMPLEMENTATION DEFINED	32-bit
0xFE4	PID1	Debug Chain Power Control Peripheral ID 1	RO	IMPLEMENTATION DEFINED	32-bit
0xFE8	PID2	Debug Chain Power Control Peripheral ID 2	RO	IMPLEMENTATION DEFINED	32-bit
0xFEC	PID3	Debug Chain Power Control Peripheral ID 3	RO	0x00	32-bit
0xFF0	ID0	Debug Chain Power Control Component ID 0	RO	0x0D	32-bit
0xFF4	ID1	Debug Chain Power Control Component ID 1	RO	0xF0	32-bit
0xFF8	ID2	Debug Chain Power Control Component ID 2	RO	0x05	32-bit
0xFFC	ID3	Debug Chain Power Control Component ID 3	RO	0xB1	32-bit

### 7.4.11.1 DBG\_CHN\_PWR\_REQ\_ST, Debug Chain Power Request Status register

Debug Chain Power Request Status.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Chain Power Control Logic registers

**Address offset**

0x000

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-291: DBG\_CHN\_PWR\_REQ\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:N]	RESERVED	Reserved	RAZ/WI	-
[N-1:0]	CDBGPWRUPREQ	Status of CDBGPWRUPREQ[x] signal from the GPR in the applications processor debug logic. N is equal to the number of PPUs in the Debug Chain PIK	RW	-

**7.4.11.2 DBG\_CHN\_PWR\_ACK, Debug Chain Power Request Acknowledge register**

Debug Chain Power Request Acknowledge.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

[Debug Chain Power Control Logic registers](#)

**Address offset**

0x004

**Type**

RO

**Reset value**

0x0

## Bit descriptions

**Table 7-292: DBG\_CHN\_PWR\_ACK bit descriptions**

Bits	Name	Description	Type	Reset
[31:N]	RESERVED	Reserved	RAZ/WI	-
[N-1:0]	CDBGWRUPACK	Sets the acknowledge (CDBGWRUPACK[x]) to a debug chain power up request (CDBGWRUPREQ[x]) from the GPR in the applications processor debug logic.  N is equal to the number of PPUs in the Debug Chain PIK	RO	-

## 7.4.11.3 DBG\_CHN\_PPU\_INT\_ST, Debug Chain PPU Interrupt Status register

Debug Chain PPU Interrupt Status.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Debug Chain Power Control Logic registers](#)

#### Address offset

0xB00

#### Type

RO

#### Reset value

0x0

## Bit descriptions

**Table 7-293: DBG\_CHN\_PPU\_INT\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:N]	RESERVED	Reserved	RAZ/WI	-
[N-1:0]	PPU_INT	Status of interrupt from PPU for Debug Chain N.  N is equal to the number of PPUs in the Debug Chain PIK	RO	-

### 7.4.11.4 DBG\_CHN\_PWRUP\_INT\_ST, Debug Chain Power-up Interrupt Status register

Debug Chain Power-up Interrupt Status.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Chain Power Control Logic registers](#)

##### Address offset

0xB04

##### Type

RW1C

##### Reset value

0x0

#### Bit descriptions

**Table 7-294: DBG\_CHN\_PWR\_INT\_ST bit descriptions**

Bits	Name	Description	Type	Reset
[31:N]	RESERVED	Reserved	RAZ/WI	-
[N-1:0]	CDBGPWRUPREQ_INT	This bit is set on any edge of CDBGPWRUPREQ signal for any debug chain. Writing a 1 to this bit clears it.  N is equal to the number of PPUs in the Debug Chain PIK	RW1C	-

### 7.4.11.5 PCL\_CFG, Power Control Logic Configuration register

Power Control Logic Configuration.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Chain Power Control Logic registers](#)

**Address offset**

0xFC0

**Type**

RO

**Reset value**

IMPLEMENTATION DEFINED

**Bit descriptions****Table 7-295: PCL\_CFG bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	PCL_ID	POWER CONTROL LOGIC_ID. This field is set to 0x0035.	RO	-
[15:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	NO_OF_PPUs	Defines the number of PPUs in the POWER CONTROL LOGIC.  This value is set to the number of debug chains implemented.	RO	-

**7.4.11.6 PID4, Debug Chain Power Control Peripheral ID 4 register**

The PID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Debug Chain Power Control Logic registers

**Address offset**

0xFD0

**Type**

RO

**Reset value**

IMPLEMENTATION DEFINED

**Bit descriptions****Table 7-296: PID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[7:4]	4KB_count	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2. Reads as 0x6, which means that the power control logic occupies a 256KB address block.	RO	-
[3:0]	jep106_c_code	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code. Reads as 0x4.	RO	-

#### 7.4.11.7 PID5, Debug Chain Power Control Peripheral ID 5 register

The PID5 register is Reserved.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[Debug Chain Power Control Logic registers](#)

###### Address offset

0xFD4

###### Type

RO

###### Reset value

IMPLEMENTATION DEFINED

##### Bit descriptions

**Table 7-297: PID5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

#### 7.4.11.8 PID6, Debug Chain Power Control Peripheral ID 6 register

The PID6 register is Reserved.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

**Functional group**

Debug Chain Power Control Logic registers

**Address offset**

0xFD8

**Type**

RO

**Reset value**

IMPLEMENTATION DEFINED

**Bit descriptions****Table 7-298: PID6 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

**7.4.11.9 PID7, Debug Chain Power Control Peripheral ID 7 register**

The PID7 register is Reserved.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

Debug Chain Power Control Logic registers

**Address offset**

0xFDC

**Type**

RO

**Reset value**

IMPLEMENTATION DEFINED

**Bit descriptions****Table 7-299: PID7 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

### 7.4.11.10 PID0, Debug Chain Power Control Peripheral ID 0 register

The PID0 register contains the first eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Chain Power Control Logic registers](#)

##### Address offset

0xFE0

##### Type

RO

##### Reset value

IMPLEMENTATION DEFINED

#### Bit descriptions

**Table 7-300: PID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	part_number_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	-

### 7.4.11.11 PID1, Debug Chain Power Control Peripheral ID 1 register

The PID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[Debug Chain Power Control Logic registers](#)

##### Address offset

0xFE4

**Type**

RO

**Reset value**

IMPLEMENTATION DEFINED

**Bit descriptions****Table 7-301: PID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	jep106_id_3_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral.	RO	-
[3:0]	part_number_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	-

**7.4.11.12 PID2, Debug Chain Power Control Peripheral ID 2 register**

The PID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[Debug Chain Power Control Logic registers](#)**Address offset**

0xFE8

**Type**

RO

**Reset value**

IMPLEMENTATION DEFINED

**Bit descriptions****Table 7-302: PID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	Revision	Specifies the major revision number for the block. For r0p0, the value is defined by the implementation.	RO	-
[3]	jedec_used	Specifies whether the JEDEC JEP106 identification scheme is in use. Always reads as 0x1.	RO	-

Bits	Name	Description	Type	Reset
[2:0]	jep106_id_6_4	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	-

### 7.4.11.13 PID3, Debug Chain Power Control Peripheral ID 3 register

The PID3 register contains the manufacturer silicon revision number and a value that is incremented with each authorized customer modification.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

Debug Chain Power Control Logic registers

##### Address offset

0xFEC

##### Type

RO

##### Reset value

0x00

#### Bit descriptions

Table 7-303: PID3 bit descriptions

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	RevAnd	The top-level RTL provides a 4-bit input, <b>ECOREVNUM</b> , that is normally tied LOW and provides a read value of 0x0. When silicon is available and if metal fixes are necessary, the manufacturer can modify the tie-offs to indicate a revision of the silicon.	RO	-
[3:0]	mod_number	Specifies the authorized customer modification increment. Set to 0x0.	RO	-

### 7.4.11.14 ID0, Debug Chain Power Control Component ID 0 register

The ID0 register contains segment 0 of the debug chain power control component class identifier.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Chain Power Control Logic registers

### Address offset

0xFF0

### Type

RO

### Reset value

0x0D

## Bit descriptions

**Table 7-304: ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_0	Specifies segment 0 of the code that identifies the debug chain power control component class. Reads as 0x0D.	RO	-

## 7.4.11.15 ID1, Debug Chain Power Control Component ID 1 register

The ID1 register contains segment 1 of the debug chain power control component class identifier.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Chain Power Control Logic registers

### Address offset

0xFF4

### Type

RO

### Reset value

0xF0

## Bit descriptions

**Table 7-305: ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_1	Specifies segment 1 of the code that identifies the debug chain power control component class. Reads as 0xF0.	RO	-

## 7.4.11.16 ID2, Debug Chain Power Control Component ID 2 register

The ID2 register contains segment 2 of the debug chain power control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[Debug Chain Power Control Logic registers](#)

#### Address offset

0xFF8

#### Type

RO

#### Reset value

0x05

## Bit descriptions

**Table 7-306: ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_2	Specifies segment 2 of the code that identifies the debug chain power control component class. Reads as 0x05.	RO	-

## 7.4.11.17 ID3, Debug Chain Power Control Component ID 3 register

The ID3 register contains segment 3 of the debug chain power control component class identifier.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

Debug Chain Power Control Logic registers

### Address offset

0xFFC

### Type

RO

### Reset value

0xB1

## Bit descriptions

**Table 7-307: ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	comp_id_3	Specifies segment 3 of the code that identifies the debug chain power control component class. Reads as 0xB1.	RO	-

## 7.4.12 MSCP Power Control registers

The MSCP Power Control registers provide access to various parameters for both the SCP and MCP. The MCP registers are a subset of the SCP registers.

The following table lists all the registers in the SCP Power Control module.

The SCP registers are a superset of the MCP registers. Registers that are only in the SCP are marked as 'SCP only'.

**Table 7-308: MSCP Power Control register summary**

Offset	Name	Description	Type	Reset	Width
0x000 – 0x00C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x010	RESET_SYNDROME	Reset Syndrome	RW	0x1	32-bit
0x014 – 0x01C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x020	SURVIVAL_RESET_STATUS	Survival Reset Status	RW	UNKNOWN	32-bit
0x024 – 0x030	RESERVED	Reserved	RAZ/WI	-	32-bit
0x034	ADDR_TRANS	MSCP Address Translation	RW	0x0	32-bit
0x038	DBG_ADDR_TRANS	MSCP AP Debug Address Translation	RW	0x0	32-bit

Offset	Name	Description	Type	Reset	Width
0x3C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x40	WS1_TIMER_MATCH	WS1 Timer Value	RW	0x0	32-bit
0x44	WS1_TIMER_EN	WS1 Timer Enable	RW	0x1	32-bit
0x48 - 0x1FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x200	SS_RESET_SET	Subsystem Reset Request Set	WO	0x0	32-bit
0x204 - 0x80C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x810	CORECLK_CTRL	Core Clock Control	RW	0x0000_0001	32-bit
0x814	CORECLK_DIV1	Core Clock Divider	RW	0x0000_001F	32-bit
0x818 - 0x81C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x820	ACLK_CTRL	AXI Clock Control	RW	0x0000_0101	32-bit
0x824	ACLK_DIV1	AXI Clock Divider Control	RW	0x001F_001F	32-bit
0x828 - 0x82C	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0x830	GTSYNCCLK_CTRL	Generic Timer Synchronization Clock Control. SCP only	RW	-	32-bit
0x834	GTSYNCCLK_DIV1	Generic Timer Synchronization Clock Divider Control. SCP only	RW	-	32-bit
0x838 - 0x9FC	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xA00	CLKFORCE_STATUS	MSCP Clock Force Status.	RO	0x0	32-bit
0xA04	CLKFORCE_SET	MSCP Force Set.	WO	0x0	32-bit
0xA08	CLKFORCE_CLR	MSCP Force Clear.	WO	0x0	32-bit
0xA0C - 0xA10	RESERVED	Reserved	RAZ/WI	-	32-bit
0xA60	CONS_MMUTCU_INT_STATUS	Consolidated MMU TCU Interrupt Status	RO	0x0	32-bit
0xA64	CONS_MMUTCU_INT_CLR	Consolidated MMU TCU Interrupt Clear	WO	0x0	32-bit
0xA68	CONS_MMUTBU_INT_STATUS0	Consolidated MMU TBU RAS FHI Interrupt Status	RO	0x0	32-bit
0xA6C	CONS_MMUTBU_INT_CLR0	Consolidated MMU TBU RAS FHI Interrupt Clear	WO	0x0	32-bit
0xA70	CONS_MMUTBU_INT_STATUS1	Consolidated MMU TBU RAS FHI Interrupt Status	RO	0x0	32-bit
0xA74	CONS_MMUTBU_INT_CLR1	Consolidated MMU TBU RAS FHI Interrupt Clear	WO	0x0	32-bit
0xA78	CONS_MMUTBU_INT_STATUS2	Consolidated MMU TBU RAS ERI Interrupt Status	RO	0x0	32-bit
0xA7C	CONS_MMUTBU_INT_CLR2	Consolidated MMU TBU RAS ERI Interrupt Clear	WO	0x0	32-bit
0xA80	CONS_MMUTBU_INT_STATUS3	Consolidated MMU TBU RAS ERI Interrupt Status	RO	0x0	32-bit
0xA84	CONS_MMUTBU_INT_CLR3	Consolidated MMU TBU RAS ERI Interrupt Clear	WO	0x0	32-bit
0xA88	CONS_MMUTBU_INT_STATUS4	Consolidated MMU TBU RAS CRI Interrupt Status	RO	0x0	32-bit
0xA84	CONS_MMUTBU_INT_CLR4	Consolidated MMU TBU RAS CRI Interrupt Clear	WO	0x0	32-bit
0xA90	CONS_MMUTBU_INT_STATUS5	Consolidated MMU TBU RAS CRI Interrupt Status	RO	0x0	32-bit
0xA94	CONS_MMUTBU_INT_CLR5	Consolidated MMU TBU RAS CRI Interrupt Clear	WO	0x0	32-bit

Offset	Name	Description	Type	Reset	Width
0xA98 - 0xB1C	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xB20 - 0xB2C	CPU_PPU_INT_STATUS<x>	CPU PPU Interrupt Status; <x> = 0 to 3. SCP only	RO	0x0	32-bit
0xB30 - 0xB3C	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xB40 - 0xB4C	CLUS_PPU_INT_STATUS<x>	Cluster PPU Interrupt Status; <x> = 0 to 3. SCP only	RO	0x0	32-bit
0xB30 - 0xB7C	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xB80 - 0xB8C	CPU_PLL_LOCK_STATUS<x>	CPU PLL Lock Status; <x> = 0 to 3. SCP only	RW	0x0	32-bit
0xB90 - 0xBBB	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xBC0 - 0xBCC	CPU_PLL_UNLOCK_STATUS<x>	CPU PLL Unlock Status; <x> = 0 to 3. SCP only	RW	0x0	32-bit
0xBD0 - 0xBFC	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xC00 - 0xC0C	CONS_CLUS_SCF_INT_STATUS<x>	Cluster SCF Interrupt Status; <x> = 0 to 3. SCP only	RW	0x0	32-bit
0xC10 - 0xCFC	RESERVED	Reserved. SCP only	RAZ/WI	-	32-bit
0xD00	TCMECC_ERRSTATUS	TCM ECC Error Status	RW	0x0	32-bit
0xD04	TCMECC_ERRCTRL	TCM ECC Error Control	RW	0x0	32-bit
0xD08	TCMECC_ERRCODE	TCM ECC Error Code	RW	0x0	32-bit
0xD0C	TCMECC_ERRADDR	TCM ECC Error Address	RO	0x0	32-bit
0xD10 - 0xFBC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFC0	PWR_CTRL_CONFIG	MSCP Power Control Configuration	RO	0x0074_0000	32-bit
0xFC4 - 0xFCC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFD0	PERIPHERAL_ID4	MSCP Power Control Peripheral ID 4	RO	0x44	32-bit
0xFD4	PERIPHERAL_ID5	MSCP Power Control Peripheral ID 5	RO	0x00	32-bit
0xFD8	PERIPHERAL_ID6	MSCP Power Control Peripheral ID 6	RO	0x00	32-bit
0xFDC	PERIPHERAL_ID7	MSCP Power Control Peripheral ID 7	RO	0x00	32-bit
0xFE0	PERIPHERAL_ID0	MSCP Power Control Peripheral ID 0	RO	0x0000_00B8	32-bit
0xFE4	PERIPHERAL_ID1	MSCP Power Control Peripheral ID 1	RO	0x0000_00B0	32-bit
0xFE8	PERIPHERAL_ID2	MSCP Power Control Peripheral ID 2	RO	0x0000_000B	32-bit
0xFEC	PERIPHERAL_ID3	MSCP Power Control Peripheral ID 3	RO	0x0000_0000	32-bit
0xFF0	COMPONENT_ID0	MSCP Power Control Component ID 0	RO	0x0D	32-bit
0xFF4	COMPONENT_ID1	MSCP Power Control Component ID 1	RO	0xF0	32-bit
0xFF8	COMPONENT_ID2	MSCP Power Control Component ID 2	RO	0x05	32-bit
0xFFC	COMPONENT_ID3	MSCP Power Control Component ID 3	RO	0xB1	32-bit

The following table lists all the registers in MCP Power Control module.

The MCP registers are a subset of the SCP registers. Locations of SCP registers that are not in the MCP are Reserved, and are marked as 'MCP only'.

**Table 7-309: MCP Power Control register summary**

Offset	Name	Description	Type	Reset	Width
0x000 - 0x00C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x010	RESET_SYNDROME	Reset Syndrome	RW	0x1	32-bit
0x014 - 0x01C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x020	SURVIVAL_RESET_STATUS	Survival Reset Status	RW	UNKNOWN	32-bit
0x024 - 0x030	RESERVED	Reserved	RAZ/WI	-	32-bit
0x034	ADDR_TRANS	MSCP Address Translation	RW	0x0	32-bit
0x038	DBG_ADDR_TRANS	MSCP AP Debug Address Translation	RW	0x0	32-bit
0x3C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x040	WS1_TIMER_MATCH	WS1 Timer Value	RW	0x0	32-bit
0x044	WS1_TIMER_EN	WS1 Timer Enable	RW	0x1	32-bit
0x048 - 0x1FC	RESERVED	Reserved	RAZ/WI	-	32-bit
0x200	SS_RESET_SET	Subsystem Reset Request Set	WO	0x0	32-bit
0x204 - 0x80C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x810	CORECLK_CTRL	Core Clock Control	RW	0x0000_0001	32-bit
0x814	CORECLK_DIV1	Core Clock Divider	RW	0x0000_001F	32-bit
0x818 - 0x81C	RESERVED	Reserved	RAZ/WI	-	32-bit
0x820	ACLK_CTRL	AXI Clock Control	RW	0x0000_0101	32-bit
0x824	ACLK_DIV1	AXI Clock Divider Control	RW	0x001F_001F	32-bit
0x828 - 0x9FC	RESERVED	Reserved. MCP only	RAZ/WI	-	32-bit
0xA00	CLKFORCE_STATUS	MSCP Clock Force Status.	RO	0x0	32-bit
0xA04	CLKFORCE_SET	MSCP Force Set.	WO	0x0	32-bit
0xA08	CLKFORCE_CLR	MSCP Force Clear.	WO	0x0	32-bit
0xA0C - 0xA10	RESERVED	Reserved	RAZ/WI	-	32-bit
0xA60	CONS_MMUTCU_INT_STATUS	Consolidated MMU TCU Interrupt Status	RO	0x0	32-bit
0xA64	CONS_MMUTCU_INT_CLR	Consolidated MMU TCU Interrupt Clear	WO	0x0	32-bit
0xA68	CONS_MMUTBU_INT_STATUS0	Consolidated MMU TBU RAS FHI Interrupt Status	RO	0x0	32-bit
0xA6C	CONS_MMUTBU_INT_CLR0	Consolidated MMU TBU RAS FHI Interrupt Clear	WO	0x0	32-bit
0xA70	CONS_MMUTBU_INT_STATUS1	Consolidated MMU TBU RAS FHI Interrupt Status	RO	0x0	32-bit
0xA74	CONS_MMUTBU_INT_CLR1	Consolidated MMU TBU RAS FHI Interrupt Clear	WO	0x0	32-bit
0xA78	CONS_MMUTBU_INT_STATUS2	Consolidated MMU TBU RAS ERI Interrupt Status	RO	0x0	32-bit
0xA7C	CONS_MMUTBU_INT_CLR2	Consolidated MMU TBU RAS ERI Interrupt Clear	WO	0x0	32-bit
0xA80	CONS_MMUTBU_INT_STATUS3	Consolidated MMU TBU RAS ERI Interrupt Status	RO	0x0	32-bit
0xA84	CONS_MMUTBU_INT_CLR3	Consolidated MMU TBU RAS ERI Interrupt Clear	WO	0x0	32-bit
0xA88	CONS_MMUTBU_INT_STATUS4	Consolidated MMU TBU RAS CRI Interrupt Status	RO	0x0	32-bit
0xA8C	CONS_MMUTBU_INT_CLR4	Consolidated MMU TBU RAS CRI Interrupt Clear	WO	0x0	32-bit
0xA90	CONS_MMUTBU_INT_STATUS5	Consolidated MMU TBU RAS CRI Interrupt Status	RO	0x0	32-bit
0xA94	CONS_MMUTBU_INT_CLR5	Consolidated MMU TBU RAS CRI Interrupt Clear	WO	0x0	32-bit

Offset	Name	Description	Type	Reset	Width
0xA98 – 0xCFC	RESERVED	Reserved. MCP only.	RAZ/WI	-	32-bit
0xD00	TCMECC_ERRSTATUS	TCM ECC Error Status	RW	0x0	32-bit
0xD04	TCMECC_ERRCTRL	TCM ECC Error Control	RW	0x0	32-bit
0xD08	TCMECC_ERRCODE	TCM ECC Error Code	RW	0x0	32-bit
0xD0C	TCMECC_ERRADDR	TCM ECC Error Address	RO	0x0	32-bit
0xD10 – 0xFBC	RESERVED	RAZ/WI	RW	-	32-bit
0xFC0	PWR_CTRL_CONFIG	MSCP Power Control Configuration	RO	0x0074_0000	32-bit
0xFC4 – 0xFCC	RESERVED	Reserved	RAZ/WI	-	32-bit
0xFD0	PERIPHERAL_ID4	MSCP Power Control Peripheral ID 4	RO	0x44	32-bit
0xFD4	PERIPHERAL_ID5	MSCP Power Control Peripheral ID 5	RO	0x00	32-bit
0xFD8	PERIPHERAL_ID6	MSCP Power Control Peripheral ID 6	RO	0x00	32-bit
0xFDC	PERIPHERAL_ID7	MSCP Power Control Peripheral ID 7	RO	0x00	32-bit
0xFE0	PERIPHERAL_ID0	MSCP Power Control Peripheral ID 0	RO	0x0000_00B8	32-bit
0xFE4	PERIPHERAL_ID1	MSCP Power Control Peripheral ID 1	RO	0x0000_00B0	32-bit
0xFE8	PERIPHERAL_ID2	MSCP Power Control Peripheral ID 2	RO	0x0000_000B	32-bit
0xFEC	PERIPHERAL_ID3	MSCP Power Control Peripheral ID 3	RO	0x0000_0000	32-bit
0xFF0	COMPONENT_ID0	MSCP Power Control Component ID 0	RO	0x0D	32-bit
0xFF4	COMPONENT_ID1	MSCP Power Control Component ID 1	RO	0xF0	32-bit
0xFF8	COMPONENT_ID2	MSCP Power Control Component ID 2	RO	0x05	32-bit
0xFFC	COMPONENT_ID3	MSCP Power Control Component ID 3	RO	0xB1	32-bit

#### 7.4.12.1 RESET\_SYNDROME, Reset Syndrome register

This register captures the cause for the last reset.



- To clear a bit, software must write a 0 to the field. All other writes are ignored.
- For a software requested core-only reset, PORESETn bit[0] must be cleared before the register bank can assert the CORE\_RST bit[5].

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0x010

**Type**

RW

**Reset value**

0x1

**Bit descriptions****Table 7-310: RESET\_SYNDROME bit descriptions**

Bits	Name	Description	Type	Reset
[31:6]	RESERVED	Reserved	RAZ/WI	-
[5]	CORE_RST	Last reset caused by a software request for a Core only reset.	RW	0x0
[4:1]	RESERVED	Reserved	RAZ/WI	-
[0]	PORESETn	Last reset caused by the PORESETn input.	RW	0x1

**7.4.12.2 SURVIVAL\_RESET\_STATUS, Survival Reset Status register**

This register does not reset during the SoC reset. Software can use the register to store a message that the reset cannot alter.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0x020

**Type**

RW

**Reset value**

UNKNOWN

**Bit descriptions****Table 7-311: SURVIVAL\_RESET\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	SURVIVAL	This register has no defined reset value. The value is retained over a full SoC reset.  When m7.LOCKUP signal is 1 and watchdog reset occurs this register set to DEADCAFE. When m7.LOCKUP signal is 0 and watchdog reset occurs this register is set to DEADBEEF.	RW	UNKNOWN

### 7.4.12.3 ADDR\_TRANS, MSCP Address Translation register

The fields in this register can be configured to enable address traslation from SCP/MCP to AP memory map.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0x034

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-312: ADDR\_TRANS bit descriptions**

Bits	Name	Description	Type	Reset
[31]	CMN_ATRANS_EN	If this bit is set, translate the Cortex-M7 access to this memory range:  0x6000_0000 - 0x9FFF_FFFF to 4TB*CHIPID + (0x01_4000_0000 - 0x01_7FFF_FFFF)  Once software writes to this set, software to poll this register to make sure the bit is set.	RW	0x0
[30:29]	RESERVED	Reserved	RAZ/WI	
[28:1]	ADDR_47_20	Set the value of address bits [47:20] for SCP/MCP to access all of the AP memory map.  This bit field can be modified while 'ADDR_TRANS_EN' is disabled or is being disabled.	RW	0x0
[0]	ADDR_TRANS_EN	Selects whether the address translation is enabled.  0 - Address translation is disable  1 - Address translation is enabled  Once software writes to this set, software to poll this register to make sure the bit is set.	RW	0x0

#### 7.4.12.4 DBG\_ADDR\_TRANS, MSCP AP Debug Address Translation register

Debug Address translation can be enabled through this register to debug memory region in AP memory map.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[MSCP Power Control registers](#), SCP and MCP

###### Address offset

0x038

###### Type

RW

###### Reset value

0x0

##### Bit descriptions

**Table 7-313: DBG\_ADDR\_TRANS bit descriptions**

Bits	Name	Description	Type	Reset
[31:12]	REMAP_DBGADDR	Remap address for SCP/MCP to access all DBG memory map. See <a href="#">Address translation</a>  This bit field can be modified while 'DBG_ADDR_TRANS.EN' is disabled or is being disabled.	RW	0x0
[11:1]	RESERVED	Reserved	RAZ/WI	-
[0]	EN	Selects whether the address translation is enabled.  0 - Debug Address translation is disable  1 - debug Address translation is enabled to higher address space  See <a href="#">Address translation</a> section for details.  Once software writes to set this register bit, software to poll this register to make sure the bit is set.	RW	0x0

### 7.4.12.5 WS1\_TIMER\_MATCH, WS1 Timer Value register

The delay value for resetting the subsystem on SCP or MCP watchdog WS1 interrupt can be programmed in this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0x040

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-314: WS1\_TIMER\_MATCH bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	WS1_TIMER_MATCH_VALUE	Programs the delay value for resetting the subsystem on SCP or MCP watchdog WS1 interrupt. When the delay expires SURVIVAL register is set to DEADBEEF and a compute subsystem reset is requested.	RW	0x0

### 7.4.12.6 WS1\_TIMER\_EN, WS1 Timer Enable register

Countdown of WS1 Timer can be enabled through this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0x044

**Type**

RW

**Reset value**

0x1

**Bit descriptions**

The WS1 Timer is loaded when the WS1\_TIMER\_MATCH register is written to. If the write to the register occurs when the timer is counting down the count will start counting from the new value. It counts down when both WS1\_TIMER\_EN and WDOGRESETREQ are high. If either goes low the count stops counting and will reset to the value in WS1\_TIMER\_MATCH.

**Table 7-315: WS1\_TIMER\_EN bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	WS1_TIMER_EN	Enables the WS1 Timer.  [0] - WS1 Timer disable. The counter will not count down the WS1 request  [1] - WS1 Timer enable. The counter will count down the WS1 request	RW	0x1

### 7.4.12.7 SS\_RESET\_SET, Subsystem Reset Request Set register

This register provides the ability for software to request a reset of the subsystem.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0x200

**Type**

RW

**Reset value**

0x0

**Bit descriptions**

In the status register the current value of the SS\_RST can be read back. To set the bit, software must write a 1 to the SS\_RST field in the SET register. To clear the bit, software must write a 1 to the SS\_RST field in the CLR register.

**Table 7-316: SS\_RESET\_SET bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	SS_RST	Software write to this register field to request a reset of the Subsystem.	RW	0x0

### 7.4.12.8 CORECLK\_CTRL, Core Clock Control register

This register provides the ability to select the clock source of the SCP/MCP CORECLK.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0x810

##### Type

RW

##### Reset value

0x0000\_0101

#### Bit descriptions

**Table 7-317: CORECLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source:  0x0 - Reserved  0x1 - REFCLK  0x2 - SYSPLL  Other values are Reserved	RW	-

Bits	Name	Description	Type	Reset
[7:0]	CLKSELECT	Select current clock source:  0x0 - Reserved  0x1 - REFCLK  0x2 - SYSPLL  Other values are Reserved	RW	0x1

#### 7.4.12.9 CORECLK\_DIV1, Core Clock Divider register

This register provides the ability to request a new clock divider value on the source clock (SYSPLLCLK) to generate the required respective CORECLK output. The current divider value can also be read out from this register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[MSCP Power Control registers](#), SCP and MCP

###### Address offset

0x814

###### Type

RW

###### Reset value

0x000F\_000F

##### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-318: CORECLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.12.10 ACLK\_CTRL, AXI Clock Control register

This register provides the ability to program the number of clock cycles between the SCP/MCP ACLK not being required and the request to dynamically clock gate it. The clock source of the ACLK can also be programmed through this register.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0x820

##### Type

RW

##### Reset value

0x0000\_0101

#### Bit descriptions

**Table 7-319: ACLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	Reserved	Number of clock cycles between the clock not being required and the request to dynamically clock gate it.  0x0 – No cycles  0x1 – 1 cycle  ...  0xFF – 255 cycles  This field is not used if the clock gating is not implemented for respective clock.	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source:  0x0 - Reserved  0x1 - REFCLK  0x2 - MSCPSYSPLLCLK  Other values are Reserved	RW	01
[7:0]	CLKSELECT	Select current clock source:  0x0 - Reserved  0x1 - REFCLK  0x2 - MSCPSYSPLLCLK  Other values are Reserved	RW	0x1

#### 7.4.12.11 ACLK\_DIV1, AXI Clock Divider register

This register provides the ability to request a new clock divider value on the source clock (MSCPSYSPLLCLK) to generate the respective ACLK output. The current divider value can also be read out from this register.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[MSCP Power Control registers](#), SCP and MCP

###### Address offset

0x824

###### Type

RW

###### Reset value

0x000F\_000F

##### Bit descriptions

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-320: ACLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	-	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1.  E.g. Setting of 0 indicates divider value of 1.	RO	0x0F
[15:5]	RESERVED	Reserved	RAZ/RI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1 e.g. setting a value of 0 indicates a divider value of 1	RW	0x0F

### 7.4.12.12 GTSYNCCLK\_CTRL, Generic Timer Synchronization Clock Control register

This register provides the ability to program the number of clock cycles between the GTSYNCCLK not being required and the request to dynamically clock gate it. The clock source of the GTSYNCCLK can also be programmed through this register.

#### Configurations

This register is available only in the SCP configuration.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP

##### Address offset

0x830

##### Type

RW

##### Reset value

0x0000\_0101

## Bit descriptions

**Table 7-321: GTSYNCCLK\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	ENTRY_DLY	Number of clock cycles between the clock not being required and the request to dynamically clock gate it.  0x0 - No cycles 0x1 - 1 cycle ... 0xFF - 255 cycles  This field is not used if the clock gating is not implemented for respective clock.	RW	-
[23:16]	RESERVED	Reserved	RAZ/WI	-
[15:8]	CLKSELECT_CUR	Acknowledges the currently selected clock source:  0x0 - Reserved 0x1 - REFCLK 0x2 - SYSPLL  Other values are Reserved	RW	-
[7:0]	CLKSELECT	Select current clock source:  0x0 - Reserved 0x1 - REFCLK 0x2 - SYSPLL  Other values are Reserved	RW	0x1

### 7.4.12.13 GTSYNCCLK\_DIV1, Generic Timer Synchronization Clock Divider Control register

This register provides the ability to request a new clock divider value on source clock (SYSPLLCLK) to generate the required output clock (GTSYNCCLK). The current divider value can also be read out from this register.

#### Configurations

This register is available only in the SCP configuration.

#### Attributes

##### Width

32-bit

**Functional group**

MSCP Power Control registers, SCP

**Address offset**

0x834

**Type**

RW

**Reset value**

0x001F\_001F

**Bit descriptions**

The divider is only on the faster clock. The REFCLK is not divided when it is selected.

**Table 7-322: GTSYNCCLK\_DIV1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:21]	RESERVED	Reserved	RAZ/WI	-
[20:16]	CLKDIV_CUR	Acknowledges the currently active clock divider value.  Divider value is the value + 1, e.g. setting of 0 indicates divider value of 1.	RW	1F
[15:5]	RESERVED	Reserved	RAZ/WI	-
[4:0]	CLKDIV	Requests a new clock divider value for the respective clock  The divider value is the value of CLKDIV + 1, e.g. setting a value of 0 indicates a divider value of 1.	RW	0x1F

**7.4.12.14 CLKFORCE\_STATUS, Clock Force Status register**

This register captures the status (enabled/disabled) of the dynamic clock gating on SCPACLK, MCPACLK and SCPGTSYNCCLK.

If dynamic clock gating is not implemented for a particular clock, the corresponding CLKFORCE bit is reserved.

If a bit reads back as 1 then the associated dynamic clock gating associated with the clock is disabled otherwise it is enabled.

If dynamic clock gating is not implemented for a particular clock, the corresponding CLKFORCE bit is reserved.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xA00

**Type**

RO

**Reset value**

0x0000\_0000

**Bit descriptions**

If a bit reads back as 1 then the associated dynamic clock gating associated with the clock is disabled otherwise it is enabled. Table: CLKFORCE\_STATUS bit descriptions

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	GTSYNCCLKCLKFORCE	Clock force status for SCPGTSYNCCLK. This is only for SCP. This bit is reserved for MCP.	RO	-
[0]	ACLKFORCE	Clock force status for SCP and MCP ACLK	RO	-

[7.4.12.15 CLKFORCE\\_SET, Clock Force Set register](#)

This register provides the ability to disable dynamic clock gating on the respective clock.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xA04

**Type**

WO

**Reset value**

0x0000\_0000

**Bit descriptions**

Writing a 1 to a bit within the CLKFORCE\_SET register disables any dynamic hardware clock gating for the respective clock, whilst writing 0 (zero) to a bit is ignored. The bit allocation is the same as the [CLKFORCE\\_STATUS](#) register.

If dynamic clock gating is not implemented for a particular clock, the corresponding clkforce bit is reserved.

**Table 7-324: CLKFORCE\_SET bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	GTSYNCCLKCLKFORCE	Clock force status for GTSYNCCLK. This is only for SCP. This bit is reserved for MCP.	RO	-
[0]	ACLKFORCE	Clock force status for SCP and MCP ACLK	RO	-

### 7.4.12.16 CLKFORCE\_CLR, MSCP PWR CTRL Clock Force Clear register

This register is used to clear force SCPACLK and MCPACLK to be free-running.

If dynamic clock gating is not implemented for a particular clock, the corresponding clkforce bit is reserved.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xA00

##### Type

WO

##### Reset value

0x0000\_0000

#### Bit descriptions

Writing a 1 to a bit within the CLKFORCE\_CLR register enables the dynamic hardware clock gating for the respective clock, whilst writing 0 (zero) to a bit is ignored. The bit allocation is the same as the [CLKFORCE\\_STATUS](#) register.

**Table 7-325: CLKFORCE\_CLR bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	GTSYNCCLKCLKFORCE	Clock force status for GTSYNCCLK. This is only for SCP. This bit is reserved for MCP.	RO	-
[0]	ACLKFORCE	Clock force status for SCP and MCP ACLK	RO	-

### 7.4.12.17 CONS\_MMUTCU\_INT\_STATUS, Consolidated MMU TCU Interrupt Status register

Different bits in this register show the current status of the RAS CRI/FHI/ERI interrupts from the associated TCUs.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTCU\_INT\_CLR.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xA60

##### Type

RO

##### Reset value

0x0

#### Bit descriptions

**Table 7-326: CONS\_MMUTCU\_INT\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:24]	RESERVED	Reserved	RAZ	-

Bits	Name	Description	Type	Reset
[23:0]	CONS_TCU_INT_STATUS	<p>Each bit shows the current status of the RAS MMU_TCU_RASCRi interrupt from the associated TCUs.</p> <p>Bit 0 - I/O Virtualization block 0 TCU's MMU_TCU_RASCRi</p> <p>Bit 1 - I/O Virtualization block 1 TCU's MMU_TCU_RASCRi</p> <p>...</p> <p>Bit 5 - I/O Virtualization block 5 TCU's MMU_TCU_RASCRi</p> <p>Bit 7:6 - Reserved</p> <p>Bit 8 - I/O Virtualization block 0 TCU's MMU_TCU_RASFHI</p> <p>Bit 9 - I/O Virtualization block 1 TCU's MMU_TCU_RASFHI</p> <p>...</p> <p>Bit 13 - I/O Virtualization block 5 TCU's MMU_TCU_RASFHI</p> <p>Bit 15:14 - Reserved</p> <p>Bit 16 - I/O Virtualization block 0 TCU's MMU_TCU_RASERI</p> <p>Bit 17 - I/O Virtualization block 1 TCU's MMU_TCU_RASERI</p> <p>...</p> <p>Bit 21 - I/O Virtualization block 5 TCU's MMU_TCU_RASERI</p> <p>Bit 23:22 - Reserved</p>	-	0x0

#### 7.4.12.18 CONS\_MMUTCU\_INT\_CLR, Consolidated MMU TCU Interrupt Clear register

This register clears the status of the interrupts captured in CONS\_MMUTCU\_INT\_STATUS.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[MSCP Power Control registers](#), SCP and MCP

###### Address offset

0xA64

**Type**

WO

**Reset value**

0x0

**Bit descriptions****Table 7-327: CONS\_MMUTCU\_INT\_CLR bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit CONS_MMUTCU_INT_STATUS register bit.	RO	0x0

**7.4.12.19 CONS\_MMUTBU\_INT\_STATUS0, Consolidated MMU TBU RAS FHI Interrupt Status register**

This register captures the status of MMU TBUs RAS FHI Interrupts from the 6 TBU integration modules.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTBU\_INT\_CLR0.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xA68

**Type**

RO

**Reset value**

0x0

**Bit descriptions****Table 7-328: CONS\_MMUTBU\_INT\_STATUS0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:6]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[5:0]	CONS_TBU_INT_STATUS[31:0]	<p>Each bit shows the current status of the RAS MMU_TBU_RASFHI interrupt from the associated TBUs&lt;31:0&gt;</p> <p>Bit 0 - I/O Virtualization block 0 TBU's MMU_TBU_RASFHI</p> <p>Bit 1 - I/O Virtualization block 1 TBU's MMU_TBU_RASFHI</p> <p>...</p> <p>Bit 5 - I/O Virtualization block 0 TBU's MMU_TBU_RASFHI</p>	RO	0x0

### 7.4.12.20 CONS\_MMUTBU\_INT\_CLR0, Consolidated MMU TBU RAS FHI Interrupt Clear register

This register clears the status of the interrupts captured in [CONS\\_MMUTBU\\_INT\\_STATUS0](#) register.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xA6C

##### Type

WO

##### Reset value

0x0

#### Bit descriptions

**Table 7-329: CONS\_MMUTBU\_INT\_CLR0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit <a href="#">CONS_MMUTBU_INT_STATUS0</a> register bit.	RO	0x0

## 7.4.12.21 CONS\_MMUTBU\_INT\_STATUS1, Consolidated MMU TBU32-TBU63 RAS FHI Interrupt Status register

This register captures the status of MMU TBUs RAS FHI Interrupts from TBU32-TBU63.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTBU\_INT\_CLR1.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xA70

#### Type

RO

#### Reset value

0x0

### Bit descriptions

**Table 7-330: CONS\_MMUTBU\_INT\_STATUS1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_TBU_INT_STATUS	<p>Each bit shows the current status of the RAS MMU_TBU_RASEFHI interrupt from the associated TBUs&lt;63:32&gt;.</p> <p>Bit 0 - TBU 32 MMU_TBU_RASFHI</p> <p>Bit 1 - TBU 33 MMU_TBU_RASFHI</p> <p>...</p> <p>Bit 31 - TBU 63 MMU_TBU_RASFHI</p> <p>This register is implemented only when No. of TBUS in the system &gt;32</p>	-	0x0

### 7.4.12.22 CONS\_MMUTBU\_INT\_CLR1, Consolidated MMU TBU32-TBU63 RAS FHI Interrupt Clear register

This register clears the status of the interrupts captured in CONS\_MMUTBU\_INT\_STATUS1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xA74

##### Type

WO

##### Reset value

0x0

#### Bit descriptions

**Table 7-331: CONS\_MMUTBU\_INT\_CLR1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit CONS_MMUTBU_INT_STATUS<0-5> register bit.	RO	0x0

### 7.4.12.23 CONS\_MMUTBU\_INT\_STATUS2, Consolidated MMU TBU RAS ERI Interrupt Status register

This register captures the status of MMU TBUs RAS ERI Interrupts from the 6 TBU integration modules.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTBU\_INT\_CLR2.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xA78

**Type**

RO

**Reset value**

0x0

**Bit descriptions****Table 7-332: CONS\_MMUTBU\_INT\_STATUS2 bit descriptions**

<b>Bits</b>	<b>Name</b>	<b>Description</b>	<b>Type</b>	<b>Reset</b>
[31:6]	RESERVED	Reserved	RAZ/WI	-
[5:0]	CONS_TBU_INT_STATUS[31:0]	<p>Each bit shows the current status of the RAS MMU_TBU_RASERI interrupt from the associated TBUs&lt;31:0&gt;</p> <p>Bit 0 - I/O Virtualization block 0 TBU's MMU_TBU_RASERI</p> <p>Bit 1 - I/O Virtualization block 1 TBU's MMU_TBU_RASERI</p> <p>...</p> <p>Bit 5 - I/O Virtualization block 0 TBU's MMU_TBU_RASERI</p>	RO	0x0

**7.4.12.24 CONS\_MMUTBU\_INT\_CLR2, Consolidated MMU TBU RAS ERI Interrupt Clear register**

This register clears the status of the interrupts captured in [CONS\\_MMUTBU\\_INT\\_STATUS2](#) register.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xA7C

**Type**

WO

**Reset value**

0x0

**Bit descriptions****Table 7-333: CONS\_MMUTBU\_INT\_CLR2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit <a href="#">CONS_MMUTBU_INT_STATUS2</a> register bit.	RO	0x0

**7.4.12.25 CONS\_MMUTBU\_INT\_STATUS3, Consolidated MMU TBU32-TBU63 RAS ERI Interrupt Status register**

This register captures the status of MMU TBUs RAS ERI Interrupts from TBU32-TBU63. This register is implemented only when the number of TBUs in the system is >32.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTBU\_INT\_CLR3.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xA80

**Type**

RO

**Reset value**

0x0

## Bit descriptions

**Table 7-334: CONS\_MMUTBU\_INT\_STATUS3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_TBU_INT_STATUS	<p>Each bit shows the current status of the MMU_TBU_RASERI interrupt from the associated TBUs&lt;63:32&gt;.</p> <p>Bit 0 - TBU 32 MMU_TBU_RASERI</p> <p>Bit 1 - TBU 33 MMU_TBU_RASERI</p> <p>...</p> <p>Bit 31 - TBU 63 MMU_TBU_RASERI</p>	-	0x0

## 7.4.12.26 CONS\_MMUTBU\_INT\_CLR3, Consolidated MMU TBU32-TBU63 RAS ERI Interrupt Clear register

This register clears the status of the interrupts captured in CONS\_MMUTBU\_INT\_STATUS3.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xA84

#### Type

WO

#### Reset value

0x0

## Bit descriptions

**Table 7-335: CONS\_MMUTBU\_INT\_CLR3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit CONS_MMUTBU_INT_STATUS<0-5> register bit.	RO	0x0

### 7.4.12.27 CONS\_MMUTBU\_INT\_STATUS4, Consolidated MMU TBU RAS CRI Interrupt Status register

This register captures the status of MMU TBUs RAS CRI Interrupts from the 6 TBU integration modules.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTBU\_INT\_CLR4

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xA88

##### Type

RO

##### Reset value

0x0

#### Bit descriptions

**Table 7-336: CONS\_MMUTBU\_INT\_STATUS4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:6]	RESERVED	Reserved	RAZ/WI	-
[5:0]	CONS_TBU_INT_STATUS[31:0]	<p>Each bit shows the current status of the RAS MMU_TBU_RASCRI interrupt from the associated TBUs&lt;31:0&gt;</p> <p>Bit 0 - I/O Virtualization block 0 TBU's MMU_TBU_RASCRI</p> <p>Bit 1 - I/O Virtualization block 1 TBU's MMU_TBU_RASCRI</p> <p>...</p> <p>Bit 5 - I/O Virtualization block 0 TBU's MMU_TBU_RASCRI</p>	RO	0x0

### 7.4.12.28 CONS\_MMUTBU\_INT\_CLR4, Consolidated MMU TBU RAS CRI Interrupt Clear register

This register clears the status of the interrupts captured in [CONS\\_MMUTBU\\_INT\\_STATUS4](#) register.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xA8C

##### Type

WO

##### Reset value

0x0

#### Bit descriptions

**Table 7-337: CONS\_MMUTBU\_INT\_CLR4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit <a href="#">CONS_MMUTBU_INT_STATUS4</a> register bit.	RO	0x0

### 7.4.12.29 CONS\_MMUTBU\_INT\_STATUS5, Consolidated MMU TBU32-TBU63 RAS CRI Interrupt Status register

This register captures the status of MMU TBUs RAS CRI Interrupts from TBU32-TBU63. This register is implemented only when the number of TBUs in the system is >32.

This register is set by hardware on the rising edge of respective interrupt signals and cleared by software writing to CONS\_MMUTBU\_INT\_CLR5.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MSCP Power Control registers](#), SCP and MCP

### Address offset

0xA90

### Type

RO

### Reset value

0x0

## Bit descriptions

**Table 7-338: CONS\_MMUTBU\_INT\_STATUS5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_TBU_INT_STATUS	Each bit shows the current status of the RAS MMU_TBU_RASECRI interrupt from the associated TBUs<63:32>.  Bit 0 - TBU 32 MMU_TBU_RASCR Bit 1 - TBU 33 MMU_TBU_RASCR ... Bit 31 - TBU 63 MMU_TBU_RASCR	-	0x0

## 7.4.12.30 CONS\_MMUTBU\_INT\_CLR5, Consolidated MMU TBU32-TBU63 RAS CRI Interrupt Clear register

Register to clear the status of the interrupts captured in CONS\_MMUTBU\_INT\_STATUS5.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MSCP Power Control registers](#), SCP and MCP

### Address offset

0xA94

**Type**

WO

**Reset value**

0x0

**Bit descriptions****Table 7-339: CONS\_MMUTBU\_INT\_CLR5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_INT_STATUS	Writing '1' to this register bit will clear corresponding bit CONS_MMUTBU_INT_STATUS<0-5> register bit.	RO	0x0

**7.4.12.31 CPU\_PPU\_INT\_STATUS<x>, CPU PPU Interrupt Status register, x = 0-3**

This register captures the associated status of CPU PPU interrupts.

- CPU\_PLL\_INT\_STATUS0 holds the PPU interrupt status for cores 0–31.
- CPU\_PLL\_INT\_STATUS1 holds the PPU interrupt status for cores 32–63.
- CPU\_PLL\_INT\_STATUS2 holds the PPU interrupt status for cores 64–95.
- CPU\_PLL\_INT\_STATUS3 holds the PPU interrupt status for cores 96–127.
- CPU\_PLL\_INT\_STATUS4 holds the PPU interrupt status for cores 128–159.
- CPU\_PLL\_INT\_STATUS5 holds the PPU interrupt status for cores 160–191.
- CPU\_PLL\_INT\_STATUS6 holds the PPU interrupt status for cores 192–223.
- CPU\_PLL\_INT\_STATUS7 holds the PPU interrupt status for cores 224–256.

**Configurations**

This register is available only in the SCP configuration.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers, SCP](#)**Address offset**

0xB20 – 0xB2C

**Type**

RO

**Reset value**

0x0

## Bit descriptions

**Table 7-340: CPU\_PPU\_INT\_STATUS<x> bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CPU_PPU_INT_STATUS	<p>Each bit shows the current status of the associated CPU PPU Interrupt</p> <p>Bit 0 - CPU {x*32+0} PPU Interrupt</p> <p>Bit 1 - CPU {x*32+1} PPU Interrupt</p> <p>....</p> <p>Bit 11 - CPU {x*32+11} PPU Interrupt</p> <p>Bit 12- 31 - CPU {x*32+12} to CPU {x*32+31} PPU Interrupt</p> <p>Where x = 0 to 3</p>	RO	0x0

### 7.4.12.32 CLUS\_PPU\_INT\_STATUS<x>, Cluster PPU Interrupt Status register, x = 0-3

This register captures the associated status of cluster PPU interrupts.

- CONS\_CLUS\_PPU\_INT\_STATUS0 holds the PPU interrupt status for clusters 0–31.
- CONS\_CLUS\_PPU\_INT\_STATUS1 holds the PPU interrupt status for clusters 32–63.
- CONS\_CLUS\_PPU\_INT\_STATUS2 holds the PPU interrupt status for clusters 64–95.
- CONS\_CLUS\_PPU\_INT\_STATUS3 holds the PPU interrupt status for clusters 96–127.
- CONS\_CLUS\_PPU\_INT\_STATUS4 holds the PPU interrupt status for clusters 128–159.
- CONS\_CLUS\_PPU\_INT\_STATUS5 holds the PPU interrupt status for clusters 160–191.
- CONS\_CLUS\_PPU\_INT\_STATUS6 holds the PPU interrupt status for clusters 192–223.
- CONS\_CLUS\_PPU\_INT\_STATUS7 holds the PPU interrupt status for clusters 224–256.

#### Configurations

This register is available only in the SCP configuration.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP

##### Address offset

0xB40 – 0xB4C

**Type**

RO

**Reset value**

0x0

**Bit descriptions****Table 7-341: CLUS\_PPU\_INT\_STATUS<x> bit descriptions**

<b>Bits</b>	<b>Name</b>	<b>Description</b>	<b>Type</b>	<b>Reset</b>
[31:0]	CLUS_PPU_INT_STATUS	Each bit shows the current status of the associated Cluster PPU Interrupt  Bit 0 - Cluster {x*32+0} PPU Interrupt  Bit 1 - Cluster {x*32+1} PPU Interrupt  ....  Bit 11 - Cluster {x*32+11} PPU Interrupts  Bit 12- 31 - Cluster {x*32+12} to Cluster {x*32+31} PPU Interrupt  Where x = 0 to 3	RO	0x0

**7.4.12.33 CPU\_PLL\_LOCK\_STATUS<x>, CPU PLL Lock Status register, x = 0-7**

This register indicates the lock status of Core PLL.

- CPU\_PLL\_LOCK\_STATUS0 holds the PLL lock status for cores 0–31.
- CPU\_PLL\_LOCK\_STATUS1 holds the PLL lock status for cores 32–63.
- CPU\_PLL\_LOCK\_STATUS2 holds the PLL lock status for cores 64–95.
- CPU\_PLL\_LOCK\_STATUS3 holds the PLL lock status for cores 96–127.
- CPU\_PLL\_LOCK\_STATUS4 holds the PLL lock status for cores 128–159.
- CPU\_PLL\_LOCK\_STATUS5 holds the PLL lock status for cores 160–191.
- CPU\_PLL\_LOCK\_STATUS6 holds the PLL lock status for cores 192–223.
- CPU\_PLL\_LOCK\_STATUS7 holds the PLL lock status for cores 224–256.

**Configurations**

This register is available only in the SCP configuration.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP

**Address offset**

0xB80 – 0xB8C

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-342: CPU\_PLL\_LOCK\_STATUS<x> bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CPU_PLL_LOCK_STATUS	Each bit is set to one when the associated Core PLL lock status goes high.  Software must write 1 to clear the bit	RW	0x0

**7.4.12.34 CPU\_PLL\_UNLOCK\_STATUS<x>, CPU PLL Unlock Status register, x = 0-7**

This register indicates the unlock status of Core PLL.

- CPU\_PLL\_UNLOCK\_STATUS0 holds the PLL unlock status for cores 0–31.
- CPU\_PLL\_UNLOCK\_STATUS1 holds the PLL unlock status for cores 32–63.
- CPU\_PLL\_UNLOCK\_STATUS2 holds the PLL unlock status for cores 64–95.
- CPU\_PLL\_UNLOCK\_STATUS3 holds the PLL unlock status for cores 96–127.
- CPU\_PLL\_UNLOCK\_STATUS4 holds the PLL unlock status for cores 128–159.
- CPU\_PLL\_UNLOCK\_STATUS5 holds the PLL unlock status for cores 160–191.
- CPU\_PLL\_UNLOCK\_STATUS6 holds the PLL unlock status for cores 192–223.
- CPU\_PLL\_UNLOCK\_STATUS7 holds the PLL unlock status for cores 224–256.

**Configurations**

This register is available only in the SCP configuration.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers, SCP](#)**Address offset**

0xBC0 – 0xBCC

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-343: CPU\_PLL\_UNLOCK\_STATUS<x> bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CPU_PLL_UNLOCK_STATUS	Each bit is set to one when the associated PLL lock status goes low.  Software must write 1 to clear the bit	RW	0x0

**7.4.12.35 CONS\_CLUS\_SCF\_INT\_STATUS<x>, Cluster SCF Interrupt Status register, x = 0-7**

This register captures the current status of the associated Cluster SCF interrupt.

- CONS\_CLUS\_SCF\_INT\_STATUS0 holds the SCF interrupt status for clusters 0–31.
- CONS\_CLUS\_SCF\_INT\_STATUS1 holds the SCF interrupt status for clusters 32–63.
- CONS\_CLUS\_SCF\_INT\_STATUS2 holds the SCF interrupt status for clusters 64–95.
- CONS\_CLUS\_SCF\_INT\_STATUS3 holds the SCF interrupt status for clusters 96–127.
- CONS\_CLUS\_SCF\_INT\_STATUS4 holds the SCF interrupt status for clusters 128–159.
- CONS\_CLUS\_SCF\_INT\_STATUS5 holds the SCF interrupt status for clusters 160–191.
- CONS\_CLUS\_SCF\_INT\_STATUS6 holds the SCF interrupt status for clusters 192–223.
- CONS\_CLUS\_SCF\_INT\_STATUS7 holds the SCF interrupt status for clusters 224–256.

**Configurations**

This register is available only in the SCP configuration.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers, SCP](#)**Address offset**

0xC00 – 0xC0C

**Type**

RO

**Reset value**

0x0

## Bit descriptions

**Table 7-344: CONS\_CLUS\_SCF\_INT\_STATUS<x> bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	CONS_CLUS_SCF_INT_STATUS	<p>Each bit shows the current status of the associated Cluster SCF interrupt</p> <p>Bit 0 - Cluster {x*32+0} SCF Interrupt</p> <p>Bit 1 - Cluster {x*32+1} SCF Interrupt</p> <p>....</p> <p>Bit 11 - Cluster {x*32+11} SCF Interrupt</p> <p>Bit 12- 31 - Cluster {x*32+12} to Cluster {x*32+31} SCF Interrupt</p> <p>Where x = 0 to 3</p>	RW	0x0-

## 7.4.12.36 TCMECC\_ERRSTATUS, TCM ECC Error Status register

This register captures the error status of TCM RAM.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xD00

#### Type

RW

#### Reset value

0x0

## Bit descriptions

**Table 7-345: TCMECC\_ERRSTATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:3]	RESERVED	Reserved	RAZ/WI	-
[2]	OF	Multibit error occurred. Write a 1 to clear this bit and clear the MULTIBIT TYPE also.	RW	0b0
[1]	UE	Uncorrectable and uncontainable error have occurred. Write a 1 to clear this bit and clear the ERRCODE also.	RW	0b0

Bits	Name	Description	Type	Reset
[0]	CE	Correctable error has occurred. Write a 1 to clear this bit and clear the ERRCODE field also.	RW	0b0

### 7.4.12.37 TCMECC\_ERRCTRL, TCM ECC Error Control register

This control register enables ECC checking and error injection on ITC/DTC RAMs. The error mask bits in the register can be programmed to enable/disable interrupt generation for each type of errors.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xD04

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

Table 7-346: TCMECC\_ERRCTRL bit descriptions

Bits	Name	Description	Type	Reset
[31:9]	RESERVED	Reserved	RAZ/WI	-
[8:7]	TGT_RAM	<ul style="list-style-type: none"> <li>0d00 - Inject error in ITCMRAM path</li> <li>0d01 - Inject error in DTCMORAM path</li> <li>0d10 - Inject error in DTCMRAM path</li> <li>0d11 - Reserved</li> </ul>	RW	0b00

Bits	Name	Description	Type	Reset
[6:5]	INJECT_ERROR	<ul style="list-style-type: none"> <li>0d00 - Do not inject any error</li> <li>0d01 - Inject a correctable error on the RAM(specified by TGT_RAM field) read data of next read transaction that comes after this bit is set.</li> <li>0d10 - Inject a Uncorrectable error on the RAM(specified by TGT_RAM field) read data of next read transaction that comes after this bit is set.</li> <li>0d11 - Reserved.</li> </ul> <p>Follow the same error checking and recording the error. Reset these bits after the injection of the error.</p> <p>If this bit is set before the error record is cleared it will generate a OF error.</p>	RW	0b00
[4]	OF_MASK	<ul style="list-style-type: none"> <li>0 - Generate an interrupt when a OF error is seen</li> <li>1 - Do not generate an interrupt when a OF error is seen</li> </ul>	RW	0b0
[3]	UE_MASK	<ul style="list-style-type: none"> <li>0 - Generate an interrupt when a UE occurs</li> <li>1 - Do not generate an interrupt when a EE error occurs</li> </ul>	RW	0b0
[2]	CE_MASK	<ul style="list-style-type: none"> <li>0 - Generate an interrupt when a CE occurs</li> <li>1 - Do not generate an interrupt when a CE error occurs</li> </ul>	RW	0b0
[1]	ITCMRAM_ECC_EN	<ul style="list-style-type: none"> <li>0 - Disable ECC checking for ITCRAM</li> <li>1 - Enable ECC checking for ITCRAM</li> </ul>	RW	0b0
[0]	DTCMRAM_ECC_EN	<ul style="list-style-type: none"> <li>0 - Disable ECC checking for DTCRams</li> <li>1 - Enable ECC checking for DTCRams</li> </ul>	RW	0b0

### 7.4.12.38 TCMECC\_ERRCODE, TCM ECC Error Code register

This register captures the error code of TCM RAM errors.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xD08

##### Type

RO

##### Reset value

0x0

## Bit descriptions

**Table 7-347: TCMECC\_ERRCODE bit descriptions**

Bits	Name	Description	Type	Reset
[31:5]	RESERVED	Reserved	RAZ/WI	-
[4:3]	MULTIBIT_TYPE	Type of the last multibit error type. • 0x00 – No error • 0x01 – Last multibit error is Correctable error • 0x10 – Last multibit error is Uncorrectable error	RO	0b00
[2:0]	ERRCODE	Error code field. • 000 – No Error • 001 – Error occurred in ITCM RAM • 010 – Error occurred in DTORAM	RO	0b000

## 7.4.12.39 TCMECC\_ERRADDR, TCM ECC Error Address register

This register captures the address of the RAM location where the error occurred.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xD0C

#### Type

RO

#### Reset value

0x0

## Bit descriptions

**Table 7-348: TCMECC\_ERRADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	ERRORADDR	Address of the RAM location that error occurred. This is valid only if the ERRCODE is non-zero.	RO	0x0

## 7.4.12.40 PWR\_CTRL\_CONFIG, MSCP Power Control Configuration register

This register indicates the power controller ID for SCP/MCP.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xFC0

#### Type

RO

#### Reset value

0x0075\_0000

### Bit descriptions

**Table 7-349: PWR\_CTRL\_CONFIG bit descriptions**

Bits	Name	Description	Type	Reset
[31:16]	PWR_CTRL_ID	Indicates the type of PWR CTRL  0x0074 - SCP PWR CTRL  0x0075 - MCP PWR CTRL	RW	CFG
[15:4]	RESERVED	Reserved	RAZ/WI	-
[3:0]	NO_OF_PPU	Defines the number of PPUs in the Power Control Registers  This value is set to 0 to indicate no PPU.	RW	0x0

## 7.4.12.41 PERIPHERAL\_ID4, MSCP Power Control Peripheral ID 4 register

The PERIPHERAL\_ID4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xFD0

**Type**

RO

**Reset value**

0x44

**Bit descriptions****Table 7-350: PERIPHERAL\_ID4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	4KB_COUNT	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2.  Reads as 0x4, which means that the power control logic occupies a 64KB address block.	RO	0x4
[3:0]	JEP106_C_CODE	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code. Reads as 0x4.	RO	0x4

**7.4.12.42 PERIPHERAL\_ID5, MSCP Power Control Peripheral ID 5 register**

The PERIPHERAL\_ID5 register is Reserved.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xFD4

**Type**

RO

**Reset value**

0x00

## Bit descriptions

**Table 7-351: PERIPHERAL\_ID5 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.12.43 PERIPHERAL\_ID6, MSCP Power Control Peripheral ID 6 register

The PERIPHERAL\_ID6 register is Reserved.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xFD8

#### Type

RO

#### Reset value

0x00

## Bit descriptions

**Table 7-352: PERIPHERAL\_ID6 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

## 7.4.12.44 PERIPHERAL\_ID7, MSCP Power Control Peripheral ID 7 register

The PERIPHERAL\_ID7 register is Reserved.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xFDC

**Type**

RO

**Reset value**

0x00

**Bit descriptions****Table 7-353: PERIPHERAL\_ID7 bit descriptions**

Bits	Name	Description	Type	Reset
[31:0]	RESERVED	Reserved	RAZ/WI	-

**7.4.12.45 PERIPHERAL\_ID0, MSCP Power Control Peripheral ID 0 register**

The PERIPHERAL\_ID0 register contains the first eight bits of the identifier for the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xFE0

**Type**

RO

**Reset value**

0x0000\_00B8

**Bit descriptions****Table 7-354: PERIPHERAL\_ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/ WI	-
[7:0]	PART_NUMBER_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xB8

### 7.4.12.46 PERIPHERAL\_ID1, MSCP Power Control Peripheral ID 1 register

The PERIPHERAL\_ID1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[MSCP Power Control registers](#), SCP and MCP

##### Address offset

0xFE4

##### Type

RO

##### Reset value

0x0000\_00B0

#### Bit descriptions

**Table 7-355: PERIPHERAL\_ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	JEP106_ID_3_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral	RO	0xB</code.>
[3:0]	PART_NUMBER_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

### 7.4.12.47 PERIPHERAL\_ID2, MSCP Power Control Peripheral ID 2 register

The PERIPHERAL\_ID2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xFE8

**Type**

RO

**Reset value**

0x0000\_000B

**Bit descriptions****Table 7-356: PERIPHERAL\_ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block. For rOp0, the value is defined by the implementation..	RO	0x0
[3]	JEDEC_USED	Specifies whether the JEDEC JEP106 identification scheme is in use.	RO	0x1
[2:0]	JEP106_ID_6_4	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral.	RO	0b011

**7.4.12.48 PERIPHERAL\_ID3, MSCP Power Control Peripheral ID 3 register**

The PERIPHERAL\_ID3 register provides information about any modifications to the peripheral.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[MSCP Power Control registers](#), SCP and MCP**Address offset**

0xFEC

**Type**

RO

**Reset value**

0x00

## Bit descriptions

**Table 7-357: PERIPHERAL\_ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

## 7.4.12.49 COMPONENT\_ID0, MSCP Power Control Component ID 0 register

The COMPONENT\_ID0 register contains segment 0 of the debug chain power control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xFF0

#### Type

RO

#### Reset value

0x0D

## Bit descriptions

**Table 7-358: COMPONENT\_ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID_0	Specifies segment 0 of the code that identifies the debug chain power control component class	RO	0x0D

## 7.4.12.50 COMPONENT\_ID1, MSCP Power Control Component ID 1 register

The COMPONENT\_ID1 register contains segment 1 of the debug chain power control component class identifier.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MSCP Power Control registers](#), SCP and MCP

### Address offset

0xFF4

### Type

RO

### Reset value

0xF0

## Bit descriptions

**Table 7-359: COMPONENT\_ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID_1	Specifies segment 1 of the code that identifies the debug chain power control component class	RO	0xF0

## 7.4.12.51 COMPONENT\_ID2, MSCP Power Control Component ID 2 register

The COMPONENT\_ID2 register contains segment 2 of the debug chain power control component class identifier.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

[MSCP Power Control registers](#), SCP and MCP

### Address offset

0xFF8

### Type

RO

### Reset value

0x05

## Bit descriptions

**Table 7-360: COMPONENT\_ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID_2	Specifies segment 2 of the code that identifies the debug chain power control component class	RO	0x05

## 7.4.12.52 COMPONENT\_ID3, MSCP Power Control Component ID 3 register

The COMPONENT\_ID3 register contains segment 3 of the debug chain power control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

[MSCP Power Control registers](#), SCP and MCP

#### Address offset

0xFFC

#### Type

RO

#### Reset value

0xB1

### Bit descriptions

**Table 7-361: COMPONENT\_ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID_3	Specifies segment 3 of the code that identifies the debug chain power control component class	RO	0xB1

## 7.4.13 DMC manager registers

The DMC manager registers capture the interrupt status of the raw interrupt signals coming from the dynamic memory controller.

If a third-party DMC is integrated into the RD-N2, these registers must be provided. The DMC interrupts are combined and sent as single interrupt to the application processor or SCP.

**Table 7-362: DMC manager register summary**

Offset	Name	Description	Type	Reset	Width
0x00	DMC_CORE_RESET_CONTROL	DMC Reset Control	RW	0x0	32-bit
0x04	DMC_MISC_CONTROL	DMC TZC Bypass and Address Decode Control	RW	0x0	32-bit
0x08	DMC_RAWERR_INT_STATUS	DMC Error Interrupt Status	RO	0x0	32-bit
0x0C	DMC_RAWTEMP_INT_STATUS	DMC Temperature Interrupt Status	RO	0x0	32-bit

#### 7.4.13.1 DMC\_CORE\_RESET\_CONTROL, DMC Reset Control register

DMC Reset Control.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

DMC manager registers

###### Address offset

0x00

###### Type

RW

###### Reset value

0x0

##### Bit descriptions

**Table 7-363: DMC\_CORE\_RESET\_CONTROL bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	Reset Release control	0 - Hold in reset  1 - Release the rese. The output of this register will be and'ed with systop reset.	RW	-

#### 7.4.13.2 DMC\_MISC\_CONTROL, DMC TZC Bypass and Address Decode Control register

DMC control register for TZC bypass and address decode.

This only applies to an AXI based Memory Controller integrated with TZC-400 as part of the Dynamic Memory block.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

DMC manager registers

### Address offset

0x04

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-364: DMC\_MISC\_CONTROL bit descriptions**

Bits	Name	Description	Type	Reset
[31:1]	RESERVED	Reserved	RAZ/WI	-
[0]	TZC_BYPASS	0 - Do not bypass the TZC 1 - BYPASS the TZC  Applicable only when TZC-400 is implemented in the DMC path.	RW	-

## 7.4.13.3 DMC\_RAWERR\_INT\_STATUS, DMC Error Interrupt Status register

DMC Error Interrupt Status.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

DMC manager registers

### Address offset

0x08

### Type

RO

**Reset value**

0x0

**Bit descriptions****Table 7-365: DMC\_RAWERR\_INT\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:10]	RESERVED	Reserved	RAZ/WI	-
[9]	ecc_corrected_err_intr_fault	ecc_corrected_err_intr_fault	RO	0x0
[8]	ecc_uncorrected_err_intr_fault	ecc_uncorrected_err_intr_fault	RO	0x0
[7]	ecc_corrected_err_intr_fault_dch1	ecc_corrected_err_intr_fault_dch1	RO	0x0
[6]	ecc_uncorrected_err_intr_fault_dch1	ecc_uncorrected_err_intr_fault_dch1	RO	0x0
[5]	dfi_alert_err_fatl_intr	dfi_alert_err_fatl_intr	RO	0x0
[4]	dfi_alert_err_max_reached_intr	dfi_alert_err_max_reached_intr	RO	0x0
[3]	dfi_alter_err_intr	dfi_alter_err_intr	RO	0x0
[2]	dfi_alert_err_fatl_intr_dch1	dfi_alert_err_fatl_intr_dch1	RO	0x0
[1]	dfi_alert_err_max_reached_intr_dch1	dfi_alert_err_max_reached_intr_dch1	RO	0x0
[0]	dfi_alter_err_intr_dch1	dfi_alter_err_intr_dch1	RO	0x0

**7.4.13.4 DMC\_RAWMTEMP\_INT\_STATUS, DMC Temperature Interrupt Status register**

DMC Temperature Interrupt Status.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[DMC manager registers](#)**Address offset**

0x0C

**Type**

RO

**Reset value**

0x0

## Bit descriptions

**Table 7-366: DMC\_RAWTEMP\_INT\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:2]	RESERVED	Reserved	RAZ/WI	-
[1]	drate_temp_limit_intr_fault	drate_temp_limit_intr_fault	RO	0x0
[0]	drate_temp_limit_intr_fault_dch1	drate_temp_limit_intr_fault_dch1	RO	0x0

### 7.4.14 PCIe integration control registers

The PCIe integration control registers allow software to allocate different memory regions for Enhanced Configuration Access Method (ECAM) and Memory-mapped I/O (MMIO) address space on a bifurcated x16 PCIe lane. See [PCIe MMIO and ECAM memory regions](#) for PCIe requirements.



All the registers in this space are boot Secure. By default these come as Secure and during boot process this register region can be made Non-secure. See NI-700 GPV registers for details.

**Table 7-367: PCIe integration register summary**

Offset	Name	Description	Type	Reset	Width
0x0000	pcie_ctrl_x4_0_ecam1_start_addr	PCIe x4_0 CTRL ECAM Start Address	RW	0x0	32-bit
0x0004	pcie_ctrl_x4_0_ecam1_end_addr	PCIe x4_0 CTRL ECAM End Address	RW	0x0	32-bit
0x0008	PCIe_CTRL_x4_0_MMIO1_START_ADDR	PCIe x4_0 CTRL MMIO1 Start Address	RW	0x0	32-bit
0x000C	PCIe_CTRL_x4_0_MMIO1_END_ADDR	PCIe x4_0 CTRL MMIO1 End Address	RW	0x0	32-bit
0x0010	PCIe_CTRL_x4_0_MMIOH_START_ADDR	PCIe x4_0 CTRL MMIOH Start Address	RW	0x0	32-bit
0x0014	PCIe_CTRL_x4_0_MMIOH_END_ADDR	PCIe x4_0 CTRL MMIOH End Address	RW	0x0	32-bit
0x0018	PCIe_CTRL_x4_0_MMIOH2L_TR_START_ADDR	PCIe x4_0 CTRL MMIOH2L Translated Region Start Address	RW	0x0	32-bit
0x001C	PCIe_CTRL_x4_0_MMIOH2L_TR_END_ADDR	PCIe x4_0 CTRL MMIOH2L Translated Region End Address	RW	0x0	32-bit
0x0020	PCIe_CTRL_x4_0_CFG_START_ADDR	PCIe x4_0 CTRL Configuration space Start Address	RW	0x0	32-bit
0x0024	PCIe_CTRL_x4_0_CFG_END_ADDR	PCIe x4_0 CTRL Configuration space End Address	RW	0x0	32-bit
0x0028	PCIe_CTRL_x4_0_ECAM2_START_ADDR	PCIe x4_0 CTRL ECAM2 Start Address	RW	0x0	32-bit
0x002C	PCIe_CTRL_x4_0_ECAM2_END_ADDR	PCIe x4_0 CTRL ECAM2 End Address	RW	0x0	32-bit
0x0030	PCIe_CTRL_x4_0_ECAM2_ADDR_CTRL	PCIe x4_0 CTRL ECAM2 ADDR Control	RW	0x0	32-bit
0x0034 - 0x00FF	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0100	pcie_ctrl_x4_1_ecam1_start_addr	PCIe x4_1 CTRL ECAM Start Address	RW	0x0	32-bit
0x0104	pcie_ctrl_x4_1_ecam1_end_addr	PCIe x4_1 CTRL ECAM End Address	RW	0x0	32-bit
0x0108	PCIe_CTRL_x4_1_MMIO1_START_ADDR	PCIe x4_1 CTRL MMIO1 Start Address	RW	0x0	32-bit
0x010C	PCIe_CTRL_x4_1_MMIO1_END_ADDR	PCIe x4_1 CTRL MMIO1 End Address	RW	0x0	32-bit

Offset	Name	Description	Type	Reset	Width
0x0110	PCIe_CTRL_x4_1_MMIOH_START_ADDR	PCIe x4_1 CTRL MMIOH Start Address	RW	0x0	32-bit
0x0114	PCIe_CTRL_x4_1_MMIOH_END_ADDR	PCIe x4_1 CTRL MMIOH End Address	RW	0x0	32-bit
0x0118	PCIe_CTRL_x4_1_MMIOH2L_TR_START_ADDR	PCIe x4_1 CTRL MMIOH2L Translated Region Start Address	RW	0x0	32-bit
0x011C	PCIe_CTRL_x4_1_MMIOH2L_TR_END_ADDR	PCIe x4_1 CTRL MMIOH2L Translated Region End Address	RW	0x0	32-bit
0x0120	PCIe_CTRL_x4_1_CFG_START_ADDR	PCIe x4_1 CTRL Configuration space Start Address	RW	0x0	32-bit
0x0124	PCIe_CTRL_x4_1_CFG_END_ADDR	PCIe x4_1 CTRL Configuration space End Address	RW	0x0	32-bit
0x0128	PCIe_CTRL_x4_1_ECAM2_START_ADDR	PCIe x4_1 CTRL ECAM2 Start Address	RW	0x0	32-bit
0x012C	PCIe_CTRL_x4_1_ECAM2_END_ADDR	PCIe x4_1 CTRL ECAM2 End Address	RW	0x0	32-bit
0x0130	PCIe_CTRL_x4_1_ECAM2_ADDR_CTRL	PCIe x4_1 CTRL ECAM2 ADDR Control	RW	0x0	32-bit
0x0134 - 0x01FF	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0200	pcie_ctrl_x8_ecam1_start_addr	PCIe x8 CTRL ECAM Start Address	RW	0x0	32-bit
0x0204	pcie_ctrl_x8_ecam1_end_addr	PCIe x8 CTRL ECAM End Address	RW	0x0	32-bit
0x0208	PCIe_CTRL_x8_MMIOH_START_ADDR	PCIe x8 CTRL MMIOH Start Address	RW	0x0	32-bit
0x020C	PCIe_CTRL_x8_MMIOH_END_ADDR	PCIe x8 CTRL MMIOH End Address	RW	0x0	32-bit
0x0210	PCIe_CTRL_x8_MMIOH2L_TR_START_ADDR	PCIe x8 CTRL MMIOH2L Translated Region Start Address	RW	0x0	32-bit
0x0214	PCIe_CTRL_x8_MMIOH2L_TR_END_ADDR	PCIe x8 CTRL MMIOH2L Translated Region End Address	RW	0x0	32-bit
0x0218	PCIe_CTRL_x8_MMIOH2L_TR_START_ADDR	PCIe x8 CTRL MMIOH2L Translated Region Start Address	RW	0x0	32-bit
0x021C	PCIe_CTRL_x8_MMIOH2L_TR_END_ADDR	PCIe x8 CTRL MMIOH2L Translated Region End Address	RW	0x0	32-bit
0x0220	PCIe_CTRL_x8_CFG_START_ADDR	PCIe x8 CTRL Configuration space Start Address	RW	0x0	32-bit
0x0224	PCIe_CTRL_x8_CFG_END_ADDR	PCIe x8 CTRL Configuration space End Address	RW	0x0	32-bit
0x0228	PCIe_CTRL_x8_ECAM2_START_ADDR	PCIe x8 CTRL ECAM2 Start Address	RW	0x0	32-bit
0x022C	PCIe_CTRL_x8_ECAM2_END_ADDR	PCIe x8 CTRL ECAM2 End Address	RW	0x0	32-bit
0x0230	PCIe_CTRL_x8_ECAM2_ADDR_CTRL	PCIe x8 CTRL ECAM2 ADDR Control	RW	0x0	32-bit
0x0234 - 0x02FF	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0300	pcie_ctrl_x16_ecam1_start_addr	PCIe x16 CTRL ECAM Start Address	RW	0x0	32-bit
0x0304	pcie_ctrl_x16_ecam1_end_addr	PCIe x16 CTRL ECAM End Address	RW	0x0	32-bit
0x0308	PCIe_CTRL_x16_MMIOH_START_ADDR	PCIe x16 CTRL MMIOH Start Address	RW	0x0	32-bit
0x030C	PCIe_CTRL_x16_MMIOH_END_ADDR	PCIe x16 CTRL MMIOH End Address	RW	0x0	32-bit
0x0310	PCIe_CTRL_x16_MMIOH2L_TR_START_ADDR	PCIe x16 CTRL MMIOH2L Translated Region Start Address	RW	0x0	32-bit
0x0314	PCIe_CTRL_x16_MMIOH2L_TR_END_ADDR	PCIe x16 CTRL MMIOH2L Translated Region End Address	RW	0x0	32-bit
0x0318	PCIe_CTRL_x16_MMIOH2L_TR_START_ADDR	PCIe x16 CTRL MMIOH2L Translated Region Start Address	RW	0x0	32-bit
0x031C	PCIe_CTRL_x16_MMIOH2L_TR_END_ADDR	PCIe x16 CTRL MMIOH2L Translated Region End Address	RW	0x0	32-bit

Offset	Name	Description	Type	Reset	Width
0x0320	PCIe_CTRL_x16_CFG_START_ADDR	PCIe x16 CTRL Configuration space Start Address	RW	0x0	32-bit
0x0324	PCIe_CTRL_x16_CFG_END_ADDR	PCIe x16 CTRL Configuration space End Address	RW	0x0	32-bit
0x0328	PCIe_CTRL_x16_ECAM2_START_ADDR	PCIe x16 CTRL ECAM2 Start Address	RW	0x0	32-bit
0x032C	PCIe_CTRL_x16_ECAM2_END_ADDR	PCIe x16 CTRL ECAM2 End Address	RW	0x0	32-bit
0x0330	PCIe_CTRL_x16_ECAM2_ADDR_CTRL	PCIe x16 CTRL ECAM2 ADDR Control	RW	0x0	32-bit
0x334 – 0x3FF	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0400	NCI_PMU_CONS_INT_STATUS	Consolidated PMU interrupts from NI-700	RO	0xFFFFFFFF	32-bit
0x0404 – 0xF0F	RESERVED	Reserved	RAZ/WI	-	32-bit
0x0FD0	PID4	Peripheral ID 4	RO	0x00000004	32-bit
0x0FE0	PID0	Peripheral ID 0	RO	0x000000E7	32-bit
0x0FE4	PID1	Peripheral ID 1	RO	0x000000B0	32-bit
0x0FE8	PID2	Peripheral ID 2	RO	0x0000000B	32-bit
0x0FEC	PID3	Peripheral ID 3	RO	0x00000000	32-bit
0x0FF0	COMPID0	Component ID 0 Register	RO	0x0000000D	32-bit
0x0FF4	COMPID1	Component ID 1 Register	RO	0x000000F0	32-bit
0x0FF8	COMPID2	Component ID 2 Register	RO	0x00000005	32-bit
0x0FFC	COMPID3	Component ID 3 Register	RO	0x000000B1	32-bit



If the transaction address matches multiple {PCIe\_CTRL\_x\*\_START\_ADDR, PCIe\_CTRL\_x\*\_END\_ADDR} regions, or if any PCIe\_CTRL\_x\*\_START\_ADDR is greater than its corresponding PCIe\_CTRL\_x\*\_END\_ADDR, the transaction is terminated and returns a DECERR.

#### 7.4.14.1 PCIe\_CTRL\_x4\_0\_ECAM1\_START\_ADDR, x4 CTRLO ECAM Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM1 region for PCIe x4 CTRLO.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

PCIe integration control registers

**Address offset**

0x0000

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-368: PCIe\_CTRL\_x4\_0\_ECAM1\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.2 PCIe\_CTRL\_x4\_0\_ECAM1\_END\_ADDR, x4 CTRL0 ECAM End Address register**

This register provides the ability to program the end address (bits[47:20]) of ECAM1 region for PCIe x4 CTRL0.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0004

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-369: PCIe\_CTRL\_x4\_0\_ECAM1\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.3 PCIe\_CTRL\_x4\_0\_MMIOL\_START\_ADDR, x4 CTRLO MMIO Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO LOW region for PCIe x4 CTRLO.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0008

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-370: PCIe\_CTRL\_x4\_0\_MMIOL\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

#### 7.4.14.4 PCIe\_CTRL\_x4\_0\_MMIO\_L\_END\_ADDR, x4 CTRL0 MMIO L End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO LOW region for PCIe x4 CTRL0.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

PCIe integration control registers

###### Address offset

0x000C

###### Type

RW

###### Reset value

0x0

##### Bit descriptions

**Table 7-371: PCIe\_CTRL\_x4\_0\_MMIO\_L\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

#### 7.4.14.5 PCIe\_CTRL\_x4\_0\_MMIOH\_START\_ADDR, x4 CTRL0 MMIO H Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH region for PCIe x4 CTRL0.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

PCIe integration control registers

**Address offset**

0x0010

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-372: PCIe\_CTRL\_x4\_0\_MMIOH\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.6 PCIe\_CTRL\_x4\_0\_MMIOH\_END\_ADDR, x4 CTRLO MMIOH End Address register**

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH region for PCIe x4 CTRLO.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0014

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-373: PCIe\_CTRL\_x4\_0\_MMIOH\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.7 PCIe\_CTRL\_x4\_0\_MMIOH2L\_TR\_START\_ADDR, x4 CTRL0 MMIOH2L Translated Region Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x4 CTRL0.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0018

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-374: PCIe\_CTRL\_x4\_0\_MMIOH2L\_TR\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.8 PCIe\_CTRL\_x4\_0\_MMIOH2L\_TR\_END\_ADDR, x4 CTRL0 MMIOH2L Translated Region End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x4 CTRL0.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[PCIe integration control registers](#)

##### Address offset

0x001C

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-375: PCIe\_CTRL\_x4\_0\_MMIOH2L\_TR\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.9 PCIe\_CTRL\_x4\_0\_CFG\_START\_ADDR, x4 CTRL0 Configuration Start Address register

This register provides the ability to program the start address (bits[47:20]) of the configuration region for PCIe x4 CTRL0.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[PCIe integration control registers](#)

**Address offset**

0x0020

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-376: PCIe\_CTRL\_x4\_0\_Cfg\_Start\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.10 PCIe\_CTRL\_x4\_0\_Cfg\_End\_ADDR, x4 CTRL0 Configuration End Address register**

This register provides the ability to program the end address (bits[47:20]) of the configuration region for PCIe x4 CTRL0.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0024

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-377: PCIe\_CTRL\_x4\_0\_CFG\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.11 PCIe\_CTRL\_x4\_0\_ECAM2\_START\_ADDR, x4 CTRLO ECAM2 Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM2 region for PCIe x4 CTRLO.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[PCIe integration control registers](#)

##### Address offset

0x0028

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-378: PCIe\_CTRL\_x4\_0\_ECAM2\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.12 PCIe\_CTRL\_x4\_0\_ECAM2\_END\_ADDR, x4 CTRL0 ECAM2 End Address register

This register provides the ability to program the end address (bits[47:20]) of ECAM2 region for PCIe x4 CTRL0.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x002C

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-379: PCIe\_CTRL\_x4\_0\_ECAM2\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.13 PCIe\_CTRL\_x4\_0\_ECAM2\_ADDR\_CTRL, x4 CTRL0 ECAM2 Address Control register

This register provides the ability to program the start and end address (bits[19:15]) of the ECAM2 region of PCIe x4 CTRL0.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

**Address offset**

0x0030

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-380: PCIe\_CTRL\_x4\_0\_ECAM2\_ADDR\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:10]	RESERVED	Reserved	RAZ/WI	-
[9:5]	START_ADDR	Starting address bits [19:15] of the respective region	RW	0x0
[4:0]	END_ADDR	Ending address bits [19:15] of the respective region	RW	0x0

**7.4.14.14 PCIe\_CTRL\_x4\_1\_ECAM1\_START\_ADDR, x4 CTRL1 ECAM Start Address register**

This register provides the ability to program the start address (bits[47:20]) of ECAM1 region for PCIe x4 CTRL1.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0100

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-381: PCIe\_CTRL\_x4\_1\_ECAM1\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

#### 7.4.14.15 PCIe\_CTRL\_x4\_1\_ECAM1\_END\_ADDR, x4 CTRL1 ECAM End Address register

This register provides the ability to program the end address (bits[47:20]) of ECAM1 region for PCIe x4 CTRL1.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[PCIe integration control registers](#)

###### Address offset

0x0104

###### Type

RW

###### Reset value

0x0

##### Bit descriptions

**Table 7-382: PCIe\_CTRL\_x4\_1\_ECAM1\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

#### 7.4.14.16 PCIe\_CTRL\_x4\_1\_MMIOL\_START\_ADDR, x4 CTRL1 MMIO LOW Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO LOW region for PCIe x4 CTRL1.

##### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0108

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-383: PCIe\_CTRL\_x4\_1\_MMIOL\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

## 7.4.14.17 PCIe\_CTRL\_x4\_1\_MMIOL\_END\_ADDR, x4 CTRL1 MMIO End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO LOW region for PCIe x4 CTRL1.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x010C

### Type

RW

**Reset value**

0x0

**Bit descriptions****Table 7-384: PCIe\_CTRL\_x4\_1\_MMIO\_L\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

**7.4.14.18 PCIe\_CTRL\_x4\_1\_MMIOH\_START\_ADDR, x4 CTRL1 MMIOH Start Address register**

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH region for PCIe x4 CTRL1.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0110

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-385: PCIe\_CTRL\_x4\_1\_MMIOH\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.19 PCIe\_CTRL\_x4\_1\_MMIOH\_END\_ADDR, x4 CTRL1 MMIOH End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH region for PCIe x4 CTRL1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0114

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-386: PCIe\_CTRL\_x4\_1\_MMIOH\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.20 PCIe\_CTRL\_x4\_1\_MMIOH2L\_TR\_START\_ADDR, x4 CTRL1 MMIOH2L Translated Region Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x4 CTRL1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

**Address offset**

0x0118

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-387: PCIe\_CTRL\_x4\_1\_MMIOH2L\_TR\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.21 PCIe\_CTRL\_x4\_1\_MMIOH2L\_TR\_END\_ADDR, x4 CTRL1 MMIOH2L Translated Region End Address register**

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x4 CTRL1.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0011C

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-388: PCIe\_CTRL\_x4\_1\_MMIOH2L\_TR\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.22 PCIe\_CTRL\_x4\_1\_CFG\_START\_ADDR, x4 CTRL1 Configuration Start Address register

This register provides the ability to program the start address (bits[47:20]) of the configuration region for PCIe x4 CTRL1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0120

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-389: PCIe\_CTRL\_x4\_1\_CFG\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.23 PCIe\_CTRL\_x4\_1\_CFG\_END\_ADDR, x4 CTRL1 Configuration End Address register

This register provides the ability to program the end address (bits[47:20]) of the configuration region for PCIe x4 CTRL1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0124

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-390: PCIe\_CTRL\_x4\_1\_CFG\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.24 PCIe\_CTRL\_x4\_1\_ECAM2\_START\_ADDR, x4 CTRL1 ECAM2 Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM2 region for PCIe x4 CTRL1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

**Address offset**

0x0128

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-391: PCIe\_CTRL\_x4\_1\_ECAM2\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.25 PCIe\_CTRL\_x4\_1\_ECAM2\_END\_ADDR, x4 CTRL1 ECAM2 End Address register**

This register provides the ability to program the end address (bits[47:20]) of ECAM2 region for PCIe x4 CTRL1.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x012C

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-392: PCIe\_CTRL\_x4\_1\_ECAM2\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.26 PCIe\_CTRL\_x4\_1\_ECAM2\_ADDR\_CTRL, x4 CTRL1 ECAM2 Address Control register

This register provides the ability to program the start and end address (bits[19:15]) of the ECAM2 region FOR PCIe x4 CTRL1.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0130

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-393: PCIe\_CTRL\_x4\_1\_ECAM2\_ADDR\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:10]	RESERVED	Reserved	RAZ/WI	-
[9:5]	START_ADDR	Starting address bits [19:15] of the respective region	RW	0x0
[4:0]	END_ADDR	Ending address bits [19:15] of the respective region	RW	0x0

### 7.4.14.27 PCIe\_CTRL\_x8\_ECAM1\_START\_ADDR, x8 CTRL ECAM1 Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM1 region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0200

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-394: PCIe\_CTRL\_x8\_ECAM1\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.28 PCIe\_CTRL\_x8\_ECAM1\_END\_ADDR, x8 CTRL ECAM End Address register

This register provides the ability to program the end address (bits[47:20]) of ECAM1 region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0204

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-395: PCIe\_CTRL\_x8\_ECAM1\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

## 7.4.14.29 PCIe\_CTRL\_x8\_MMIO\_START\_ADDR, x8 CTRL MMIO Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO LOW region for PCIe x8 CTRL.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0208

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-396: PCIe\_CTRL\_x8\_MMIO\_L\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

## 7.4.14.30 PCIe\_CTRL\_x8\_MMIO\_L\_END\_ADDR, x8 CTRL MMIO LOW End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO LOW region for PCIe x8 CTRL.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

PCIe integration control registers

#### Address offset

0x020C

#### Type

RW

#### Reset value

0x0

## Bit descriptions

**Table 7-397: PCIe\_CTRL\_x8\_MMIO\_L\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.31 PCIe\_CTRL\_x8\_MMIOH\_START\_ADDR, x8 CTRL MMIOH Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0210

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-398: PCIe\_CTRL\_x8\_MMIOH\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.32 PCIe\_CTRL\_x8\_MMIOH\_END\_ADDR, x8 CTRL MMIOH End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0214

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-399: PCIe\_CTRL\_x8\_MMIOH\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

## 7.4.14.33 PCIe\_CTRL\_x8\_MMIOH2L\_TR\_START\_ADDR, x8 CTRL MMIOH2L Translated Region Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x8 CTRL.

### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0218

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-400: PCIe\_CTRL\_x8\_MMIOH2L\_TR\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

## 7.4.14.34 PCIe\_CTRL\_x8\_MMIOH2L\_TR\_END\_ADDR, x8 CTRL MMIOH2L Translated Region End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x8 CTRL.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

PCIe integration control registers

#### Address offset

0x021C

#### Type

RW

#### Reset value

0x0

### Bit descriptions

**Table 7-401: PCIe\_CTRL\_x8\_MMIOH2L\_TR\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.35 PCIe\_CTRL\_x8\_CFG\_START\_ADDR, x8 CTRL Configuration Start Address register

This register provides the ability to program the start address (bits[47:20]) of the configuration region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0220

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-402: PCIe\_CTRL\_x8\_CFG\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.36 PCIe\_CTRL\_x8\_CFG\_END\_ADDR, x8 CTRL Configuration End Address register

This register provides the ability to program the end address (bits[47:20]) of the configuration region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0224

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-403: PCIe\_CTRL\_x8\_CFG\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

## 7.4.14.37 PCIe\_CTRL\_x8\_ECAM2\_START\_ADDR, x8 CTRL ECAM2 Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM2 region for PCIe x8 CTRL.

## Configurations

This register is available in all configurations.

## Attributes

### Width

32-bit

### Functional group

PCIe integration control registers

### Address offset

0x0228

### Type

RW

### Reset value

0x0

## Bit descriptions

**Table 7-404: PCIe\_CTRL\_x8\_ECAM2\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

## 7.4.14.38 PCIe\_CTRL\_x8\_ECAM2\_END\_ADDR, x8 CTRL ECAM2 End Address register

This register provides the ability to program the end address (bits[47:20]) of ECAM2 region for PCIe x8 CTRL.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

PCIe integration control registers

#### Address offset

0x022C

#### Type

RW

#### Reset value

0x0

## Bit descriptions

**Table 7-405: PCIe\_CTRL\_x8\_ECAM2\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.39 PCIe\_CTRL\_x8\_ECAM2\_ADDR\_CTRL, x8 CTRL ECAM2 Address Control register

This register provides the ability to program the start address (bits[47:20]) of ECAM2 region for PCIe x8 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0230

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-406: PCIe\_CTRL\_x8\_ECAM2\_ADDR\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:10]	RESERVED	Reserved	RAZ/WI	-
[9:5]	START_ADDR	Starting address bits [19:15] of the respective region	RW	0x0
[4:0]	END_ADDR	Ending address bits [19:15] of the respective region	RW	0x0

### 7.4.14.40 PCIe\_CTRL\_x16\_ECAM1\_START\_ADDR, x16 CTRL ECAM1 Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM1 region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

**Functional group**

PCIe integration control registers

**Address offset**

0x0300

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-407: PCIe\_CTRL\_x16\_ECAM1\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.41 PCIe\_CTRL\_x16\_ECAM1\_END\_ADDR, x16 CTRL ECAM End Address register**

This register provides the ability to program the end address (bits[47:20]) of ECAM1 region for PCIe x16 CTRL.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

PCIe integration control registers

**Address offset**

0x0304

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-408: PCIe\_CTRL\_x16\_ECAM1\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.42 PCIe\_CTRL\_x16\_MMIOL\_START\_ADDR, x16 CTRL MMIO Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO LOW region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[PCIe integration control registers](#)

##### Address offset

0x0308

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-409: PCIe\_CTRL\_x16\_MMIOL\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.43 PCIe\_CTRL\_x16\_MMIO\_L\_END\_ADDR, x16 CTRL MMIO L End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO LOW region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[PCIe integration control registers](#)

##### Address offset

0x030C

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-410: PCIe\_CTRL\_x16\_MMIO\_L\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.44 PCIe\_CTRL\_x16\_MMIO\_H\_START\_ADDR, x16 CTRL MMIO H Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

[PCIe integration control registers](#)

**Address offset**

0x0310

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-411: PCIe\_CTRL\_x16\_MMIOH\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.45 PCIe\_CTRL\_x16\_MMIOH\_END\_ADDR, x16 CTRL MMIOH End Address register**

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH region for PCIe x16 CTRL.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0314

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-412: PCIe\_CTRL\_x16\_MMIOH\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.46 PCIe\_CTRL\_x16\_MMIOH2L\_TR\_START\_ADDR, x16 CTRL MMIOH2L Translated Region Start Address register

This register provides the ability to program the start address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0318

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-413: PCIe\_CTRL\_x16\_MMIOH2L\_TR\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.47 PCIe\_CTRL\_x16\_MMIOH2L\_TR\_END\_ADDR, x16 CTRL MMIOH2L Translated Region End Address register

This register provides the ability to program the end address (bits[47:20]) of MMIO HIGH to LOW translated region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x031C

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-414: PCIe\_CTRL\_x16\_MMIOH2L\_TR\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.48 PCIe\_CTRL\_x16\_CFG\_START\_ADDR, x16 CTRL Configuration Start Address register

This register provides the ability to program the start address (bits[47:20]) of the configuration region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

**Address offset**

0x0320

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-415: PCIe\_CTRL\_x16\_CFG\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>1 - Non-secure region, allow all transactions.</li> <li>0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

**7.4.14.49 PCIe\_CTRL\_x16\_CFG\_END\_ADDR, x16 CTRL Configuration End Address register**

This register provides the ability to program the end address (bits[47:20]) of the configuration region for PCIe x16 CTRL.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0324

**Type**

RW

**Reset value**

0x0

## Bit descriptions

**Table 7-416: PCIe\_CTRL\_x16\_CFG\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.50 PCIe\_CTRL\_x16\_ECAM2\_START\_ADDR, x16 CTRL ECAM2 Start Address register

This register provides the ability to program the start address (bits[47:20]) of ECAM2 region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0328

##### Type

RW

##### Reset value

0x0

## Bit descriptions

**Table 7-417: PCIe\_CTRL\_x16\_ECAM2\_START\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:30]	RESERVED	Reserved	RAZ/WI	-
[29]	SEC_ACCCTRL_DIS	<ul style="list-style-type: none"> <li>• 1 - Non-secure region, allow all transactions.</li> <li>• 0 - Secure region, allow only Secure transactions.</li> </ul>	RW	0x0
[28:1]	START_ADDR	Starting address bits [47:20] of the respective region	RW	0x0
[0]	REG_EN	Enable bit to indicate this region is valid	RW	0x0

### 7.4.14.51 PCIe\_CTRL\_x16\_ECAM2\_END\_ADDR, x16 CTRL ECAM2 End Address register

This register provides the ability to program the end address (bits[47:20]) of ECAM2 region for PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x032C

##### Type

RW

##### Reset value

0x0

#### Bit descriptions

**Table 7-418: PCIe\_CTRL\_x16\_ECAM2\_END\_ADDR bit descriptions**

Bits	Name	Description	Type	Reset
[31:28]	RESERVED	Reserved	RAZ/WI	-
[27:0]	END_ADDR	Ending address bits [47:20] of the respective region	RW	0x0

### 7.4.14.52 PCIe\_CTRL\_x8\_ECAM2\_ADDR\_CTRL, x16 CTRL ECAM2 Address Control register

This register provides the ability to program the start and end address (bits[19:15]) of the ECAM2 region of PCIe x16 CTRL.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

**Address offset**

0x0330

**Type**

RW

**Reset value**

0x0

**Bit descriptions****Table 7-419: PCIe\_CTRL\_x16\_ECAM2\_ADDR\_CTRL bit descriptions**

Bits	Name	Description	Type	Reset
[31:10]	RESERVED	Reserved	RAZ/WI	-
[9:5]	START_ADDR	Starting address bits [19:15] of the respective region	RW	0x0
[4:0]	END_ADDR	Ending address bits [19:15] of the respective region	RW	0x0

**7.4.14.53 NCI\_PMU\_CONS\_INT\_STATUS, NCI PMU Interrupt Consolidation register**

This register captures the current status bits for each of the PMU interrupts from the corresponding NCI network. To clear the interrupt, see NI-700 registers.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0400

**Type**

RO

**Reset value**

0xFFFF\_FFFF

**Bit descriptions****Table 7-420: NCI\_PMU\_CONS\_INT\_STATUS bit descriptions**

Bits	Name	Description	Type	Reset
[31:14]	RESERVED	Reserved	RAZ/WI	-

Bits	Name	Description	Type	Reset
[13:0]	NCI_PMU_INT_STATUS	<p>Each bit shows the current status of the PMU interrupt from the corresponding NI-700 network</p> <p>Bit 0 – NCI_CKD0_PMU_INT</p> <p>Bit 1 – NCI_CKD1_PMU_INT</p> <p>Bit 2 – NCI_CKD2_PMU_INT</p> <p>...</p> <p>Bit 13 – NCI_CKD13_PMU_INT</p> <p>Note: See <a href="#">NI-700 Non-coherent Interconnect</a> for clock domains.</p>	RO	-

#### 7.4.14.54 PID\_4, PCIe Integration Control Peripheral ID 4 register

The PID\_4 register contains information about the number of address blocks that the logic occupies and the JEDEC JEP106 configuration code for the peripheral.

##### Configurations

This register is available in all configurations.

##### Attributes

###### Width

32-bit

###### Functional group

[PCIe integration control registers](#)

###### Address offset

0x0FD0

###### Type

RO

###### Reset value

0x00000004

##### Bit descriptions

**Table 7-421: PID\_4 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	SIZE	Specifies the number of 4KB address blocks that are required to access the registers, expressed in powers of 2. Set to 0x0.	RO	0x0
[3:0]	DES_2	Specifies the JEDEC JEP106 continuation code for the peripheral, which indicates the number of 0x7F continuation characters in the manufacturer identity code. Set to 0x4 for Arm.	RO	0x4

### 7.4.14.55 PID\_0, PCIe Integration Control Peripheral ID 0 register

The PID\_0 register contains the first eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0FE0

##### Type

RO

##### Reset value

0x000000E7

#### Bit descriptions

**Table 7-422: PID\_0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	PART_0	Specifies bits[7:0] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0xE7

### 7.4.14.56 PID\_1, PCIe Integration Control Peripheral ID 1 register

The PID\_1 register contains the first four bits of the JEDEC JEP106 identity code and the second eight bits of the identifier for the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0FE4

**Type**

RO

**Reset value**

0x000000B0

**Bit descriptions****Table 7-423: PID\_1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	DES_0	Specifies bits[3:0] of the JEDEC JEP106 identity code for the peripheral. Set to 0xB for Arm.	RO	0xB
[3:0]	PART_1	Specifies bits[11:8] of the part identifier for the peripheral. The value is defined by the implementation.	RO	0x0

**7.4.14.57 PID\_2, PCIe Integration Control Peripheral ID 2 register**

The CNTPIDR2 register specifies whether the JEDEC JEP106 identification scheme is in use, and contains parts of the peripheral JEP106 designer code and block version.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**[PCIe integration control registers](#)**Address offset**

0x0FE8

**Type**

RO

**Reset value**

0x0000000B

**Bit descriptions****Table 7-424: PID\_2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVISION	Specifies the major revision number for the block. The value is defined by the implementation.	RO	0x0
[3]	JEDEC	Specifies whether the JEDEC JEP106 identification scheme is in use. Set to 0x1.	RO	0b1
[2:0]	DES_1	Specifies bits[6:4] of the JEDEC JEP106 designer code for the peripheral. Set to 0x3 for Arm.	RO	0b011

### 7.4.14.58 PID\_3, PCIe Integration Control Peripheral ID 3 register

The PID\_3 register provides information about any modifications to the peripheral.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

##### Address offset

0x0FEC

##### Type

RO

##### Reset value

0x00000000

#### Bit descriptions

**Table 7-425: PID\_3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:4]	REVAND	Manufacturer revision number: This field indicates minor errata fixes specific to this design, for example metal fixes after implementation	RO	0x0
[7:0]	CMOD	Customer modification number: incremented on authorized customer modifications	RO	0x0

### 7.4.14.59 COMP\_ID0, PCIe Integration Control Component ID 0 register

The ID0 register contains segment 0 of the power control component class identifier.

#### Configurations

This register is available in all configurations.

#### Attributes

##### Width

32-bit

##### Functional group

PCIe integration control registers

**Address offset**

0x0FF0

**Type**

RO

**Reset value**

0x0D

**Bit descriptions****Table 7-426: ID0 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID0	Specifies segment 0 of the code that identifies the PCIe integration control component class. Reads as 0x0D.	RO	0x0D

**7.4.14.60 COMP\_ID1, PCIe Integration Control Component ID 1 register**

The ID1 register contains segment 1 of the power control component class identifier.

**Configurations**

This register is available in all configurations.

**Attributes****Width**

32-bit

**Functional group**

PCIe integration control registers

**Address offset**

0x0FF4

**Type**

RO

**Reset value**

0xF0

**Bit descriptions****Table 7-427: ID1 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID1	Specifies segment 1 of the code that identifies the PCIe integration control component class. Reads as 0xF0.	RO	0xF0

## 7.4.14.61 COMP\_ID2, PCIe Integration Control Component ID 2 register

The ID2 register contains segment 2 of the power control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

PCIe integration control registers

#### Address offset

0x0FF8

#### Type

RO

#### Reset value

0x05

### Bit descriptions

**Table 7-428: ID2 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID2	Specifies segment 2 of the code that identifies the PCIe integration control component class.	RO	0x05

## 7.4.14.62 COMP\_ID3, PCIe Integration Control Component ID 3 register

The ID3 register contains segment 3 of the power control component class identifier.

### Configurations

This register is available in all configurations.

### Attributes

#### Width

32-bit

#### Functional group

PCIe integration control registers

#### Address offset

0x0FFC

**Type**

RO

**Reset value**

0xB1

**Bit descriptions****Table 7-429: ID3 bit descriptions**

Bits	Name	Description	Type	Reset
[31:8]	RESERVED	Reserved	RAZ/WI	-
[7:0]	COMP_ID3	Specifies segment 3 of the code that identifies the PCIe integration control component class.	RO	0xB1

# Appendix A Revisions

This appendix describes the technical changes between released issues of this book.

**Table A-1: Issue 01**

Change	Location
First release	-

**Table A-2: Differences between Issue 01 and Issue 02**

Change	Location
Updated STM hardware event support to include hardware event interface assignment	STM hardware event support
Added information about SCP clock, reset, and power control.	System Control Processor block
Updated address translation from SCP and MCP memory map to the AP memory map	Address translation
Updated descriptions of MUH channels	Message communication between processors
Added information about Generic timer synchronization between the chips	Generic timer synchronization between the chips
Added Programmers Model	Programmers Model

**Table A-3: Differences between Issue 02 and Issue 03**

Change	Location
Removed confidential information	Throughout the document
Updated cryptography standards	Processor block
Updated An example SMMU topology with TaaS LTI mode figure	I/O Virtualization block
Updated values in Compute subsystem power and reset states table	Compute subsystem power states
Updated FVP information	About the FVP
Updated FVP information	FVP peripherals
Added new section	Multi-chiplet support
Tables aligned throughout the section	Programmers model
Added new section	About the programmers model

**Table A-4: Differences between Issue 03 and Issue 04**

Change	Location
Document confidentiality updated	Whole document