



Arm[®] Corstone[™] Reference Systems

Revision: r0p0

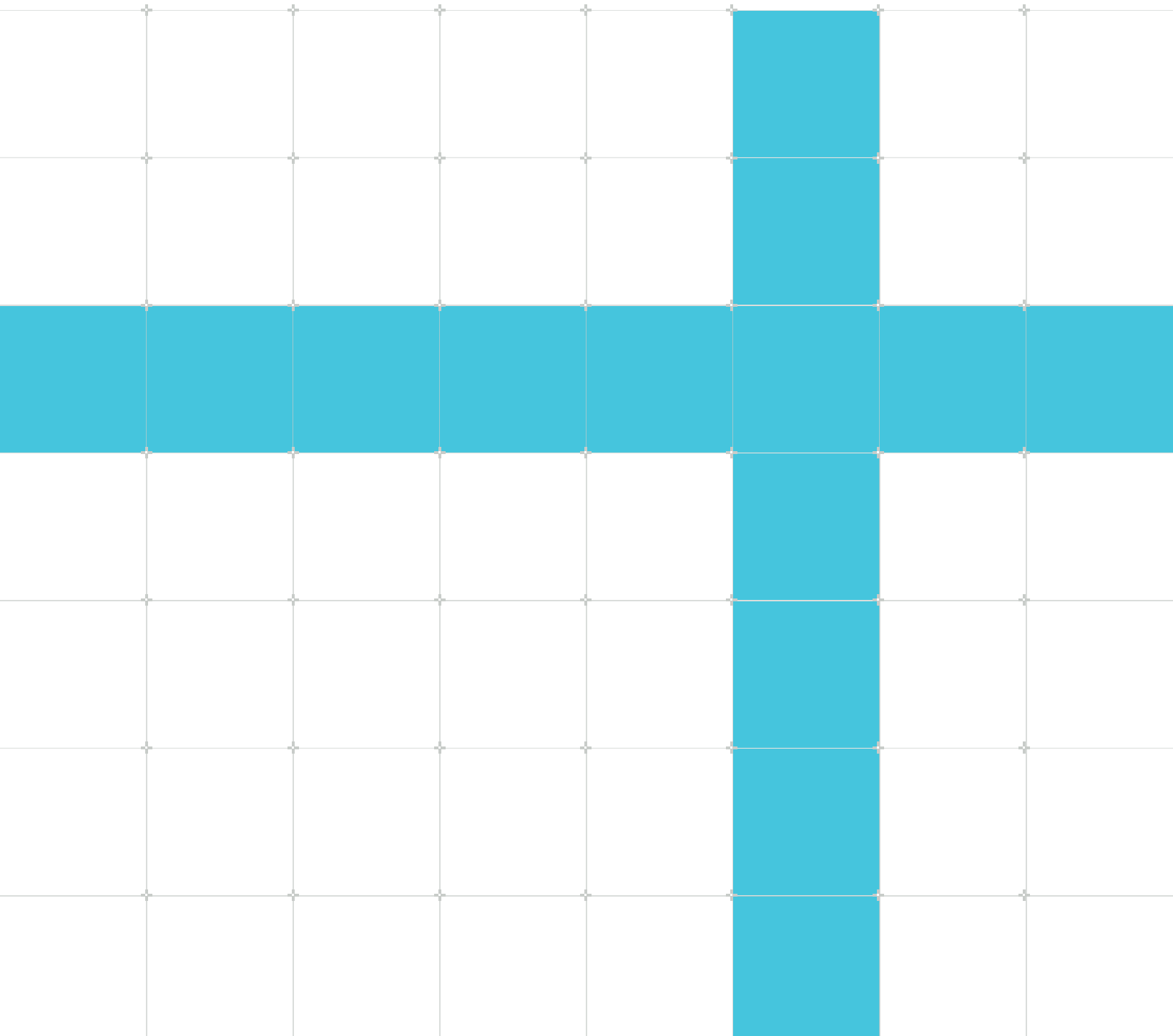
Architecture Specification Ma1

Non-Confidential

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

Issue 01

102803_0000_01_en



Arm® Corstone™ Reference Systems Architecture Specification Ma1

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

Release information

Document history

Issue	Date	Confidentiality	Change
0000-01	19 May 2022	Non-Confidential	Initial 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, 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.

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 © 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.

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.

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Contents

1 Introduction.....	10
1.1 Product revision status.....	10
1.2 Intended audience.....	10
1.3 Conventions.....	10
1.4 Additional reading.....	12
2 Overview.....	14
2.1 Document-specific conventions.....	15
2.2 Compliance.....	18
2.2.1 Trusted Base System Architecture for Armv8-M.....	18
2.3 Topology.....	18
2.3.1 System block diagram.....	19
3 Configuration options.....	21
4 Interfaces.....	27
4.1 Input and output clocks.....	28
4.2 Functional reset inputs and outputs.....	30
4.3 P-Channel and Q-Channel Device Interfaces.....	32
4.3.1 Clock control Q-Channel device interfaces.....	34
4.3.2 Power Control Q-Channel and P-Channel Expansion Device interfaces.....	34
4.3.3 Power Control P-Channel Expansion Device Interfaces.....	35
4.3.4 Power Control Wakeup Q-Channel Device interfaces.....	36
4.4 Clock Control Q-Channel Control interfaces.....	37
4.5 Expansion Power Control Dependency interface.....	37
4.6 Power Domain ON Status Signals.....	39
4.7 System timestamp interface.....	40
4.8 Main Interconnect Expansion interfaces.....	40
4.9 Peripheral Interconnect Expansion interfaces.....	42
4.10 Interrupt interfaces.....	43
4.11 DMA interfaces.....	44
4.12 CPU Coprocessor Interface.....	44
4.13 TCM subordinate interface.....	45

4.14 Debug and Trace Related interfaces.....	47
4.14.1 Debug Access interface.....	47
4.14.2 Debug Timestamp interface.....	48
4.14.3 Cross Trigger Channel interface.....	48
4.14.4 Cross Trigger Interface.....	49
4.14.5 Debug APB Expansion interface.....	49
4.14.6 CPU<n> External Peripheral interface EPPB.....	50
4.14.7 ATB Trace interfaces.....	51
4.14.8 Debug Authentication interface.....	51
4.15 CryptoCell Related Expansion interfaces.....	52
4.15.1 CryptoCell Lifecycle Indication interface.....	52
4.15.2 CryptoCell Debug Control Enable interface.....	53
4.15.3 CryptoCell Non-Volatile Memory interface.....	53
4.16 Security Control Expansion signals.....	54
4.16.1 Memory Protection Controller Expansion.....	54
4.16.2 Peripheral Interconnect Peripheral Protection Controller Expansion.....	55
4.16.3 Main Interconnect Peripheral Protection Controller Expansion.....	56
4.16.4 Manager Security Controller Expansion.....	57
4.16.5 Bridge Buffer Error Expansion.....	57
4.16.6 Other Security Expansion signals.....	58
4.17 Clock configuration interface.....	58
4.18 Miscellaneous signals.....	60
5 Functional Description.....	62
5.1 Clocking infrastructure.....	62
5.2 Reset infrastructure.....	66
5.2.1 Power-on and Cold reset handling.....	70
5.2.2 CPU Reset Handling.....	71
5.2.3 NPU Reset handling.....	72
5.2.4 Warm Reset Generation.....	72
5.3 CPU.....	73
5.3.1 EVENT Interfaces.....	75
5.3.2 Interrupts.....	75
5.3.3 CPU Custom Datapath Extension interface.....	77
5.4 System interconnect infrastructure.....	77
5.4.1 ACC_WAIT Control.....	79

5.5 Volatile Memory.....	79
5.6 Timers and Watchdogs.....	80
5.6.1 Timestamp-based timers.....	80
5.6.2 SLOWCLK AON Timers.....	81
5.7 Message Handling Unit.....	82
5.8 Power Policy Units.....	82
5.9 Peripheral Protection Controllers.....	84
5.10 Memory Protection Controllers.....	85
5.11 Manager Security Controllers.....	85
5.12 CryptoCell.....	86
5.12.1 No-Crypto Configuration.....	86
5.12.2 Has-Crypto configuration.....	86
5.13 Debug Infrastructure.....	87
5.13.1 Basic Debug Configuration.....	87
5.13.2 Full Debug Configuration.....	88
5.14 System and Security Control.....	93
5.15 Power Control Infrastructure.....	93
5.15.1 Advanced level power infrastructure.....	95
5.15.2 Intermediate level power infrastructure.....	110
5.15.3 Basic level power infrastructure.....	122
5.16 DMA.....	132
5.17 NPU.....	134
5.17.1 NPU security mapping.....	134
6 Programmers model.....	138
6.1 System Memory Map overview.....	138
6.1.1 High Level System Address Map.....	139
6.2 CPU TCM memories.....	145
6.3 Volatile Memory Region.....	146
6.4 Peripheral Region.....	147
6.4.1 Message Handling Unit register map.....	151
6.4.2 Secure Access Configuration Register Block.....	154
6.4.3 Non-secure Access Configuration Register Block.....	183
6.4.4 Timestamp-based timers registers.....	191
6.4.5 Timestamp-based Watchdogs registers.....	191
6.4.6 DMA registers.....	191

6.4.7 NPU<m> registers.....	191
6.5 CPU Private Region.....	192
6.5.1 CPU<n>_PWRCTRL register block.....	193
6.5.2 CPU<n>_IDENTITY register block.....	195
6.5.3 CPU<n>_SECCTRL register block.....	196
6.6 System Control Peripheral Region.....	198
6.6.1 SYSINFO Register Block.....	200
6.6.2 System Control Register Block.....	207
6.7 CPU Private Peripheral Bus region.....	239
6.7.1 EWIC.....	241
6.8 Debug System Access Region.....	241
6.8.1 CoreSight SoC-600 based Debug System not implemented.....	241
6.8.2 CoreSight SoC-600 based Debug System implemented.....	242
A System timer components.....	249
A.1 System Counter overview.....	249
A.2 Counter operation flows.....	249
A.3 System counter Programmers model.....	250
A.3.1 CNTControlBase registers summary.....	250
A.3.2 CNTReadBase registers summary.....	251
A.3.3 Register descriptions.....	252
A.3.4 CNTCR, Counter Control register.....	252
A.3.5 CNTSR, Counter Status register.....	253
A.3.6 CNTCV, Counter Count Value register.....	253
A.3.7 CNTSCR, Counter Scale register.....	254
A.3.8 CNTID, Counter ID register.....	254
A.3.9 CNTSCR0, CNTSCR1, Counter Scaling registers.....	255
A.4 System Timer overview.....	256
A.4.1 System Timer operation.....	257
A.5 System timer programmers model.....	258
A.5.1 CNTBase Registers Summary.....	258
A.5.2 Register descriptions.....	259
A.5.3 CNTPCT, Counter-timer Physical Count register.....	260
A.5.4 CNTFRQ, Counter-timer Frequency register.....	260
A.5.5 CNTP_CVAL, Counter-timer Physical Timer CompareValue register.....	260
A.5.6 CNTP_TVAL, Counter-timer Physical Timer TimerValue register.....	261

A.5.7 CNTP_CTL, Counter-timer Physical Timer Control register.....	261
A.5.8 CNTP_AIVAL, AutoIncrValue register.....	262
A.5.9 CNTP_AIVAL_RELOAD, AutoIncrValue Reload register.....	262
A.5.10 CNTP_AIVAL_CTL, AutoIncrValue Control register.....	262
A.5.11 CNTP_CFG, Configuration register.....	263
A.6 System Watchdog overview.....	264
A.6.1 Watchdog operation.....	264
A.6.2 System watchdog programmers model.....	264
A.6.3 Control Frame registers summary.....	265
A.6.4 Refresh Frame registers Summary.....	266
A.6.5 Register descriptions.....	266
A.6.6 WCS, Watchdog Control and Status register.....	266
A.6.7 WOR, Watchdog Offset register.....	267
A.6.8 WCV, Watchdog Compare Value register.....	267
A.6.9 W_IIDR, Watchdog Interface Identification register.....	267
A.6.10 WRR, Watchdog Refresh register.....	268
B Revisions.....	269

1 Introduction

1.1 Product revision status

The r_xp_y identifier indicates the revision status of the product described in this manual, for example, $r1p2$, where:

r_x	Identifies the major revision of the product, for example, $r1$.
p_y	Identifies the minor revision or modification status of the product, for example, $p2$.

1.2 Intended audience

This book is written for system designers, system integrators, and programmers who are designing or programming a System-on-Chip (SoC) that uses the Corstone Reference Systems Architecture Specification Ma1.

1.3 Conventions

The following subsections describe conventions used in Arm documents.

Glossary







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

See the Arm® Glossary for more information: developer.arm.com/glossary.

Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
<i>italic</i>	Citations.
bold	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.

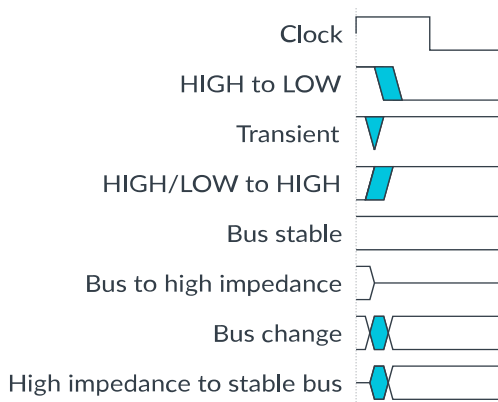
Convention	Use
monospace bold	Language keywords when used outside example code.
monospace <u>underline</u>	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: <pre>MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2></pre>
SMALL CAPITALS	Terms that have specific technical meanings as defined in the <i>Arm® Glossary</i> . For example, IMPLEMENTATION DEFINED , IMPLEMENTATION SPECIFIC , UNKNOWN , and UNPREDICTABLE .
 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® AXI and ACE Protocol Specification	IHI 0022	No
Arm® AMBA® 5 AHB Protocol Specification	IHI 0033	No
Arm® CoreLink™ DMA-350 Controller Technical Reference Manual	102482	No
Arm® CoreLink™ NIC-400 Network Interconnect Technical Overview	DDI 0475	No
Arm® CoreLink™ NIC-450 Network Interconnect Technical Overview	100459	No
Arm® CoreLink™ PCK-600 Power Control Kit Technical Reference Manual	101150	No

Document Name	Document ID	Licensee only
Arm® CoreLink™ SIE-200 System IP for Embedded Configuration and Integration Manual	DIT 0067	Yes
Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual	DDI 0571	No
Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual	101526	No
Arm® CoreSight™ Architecture Specification v3.0	IHI 0029	No
Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual	100806	No
Arm® Cortex®-M55 Processor Integration and Implementation Manual	101052	Yes
Arm® Cortex®-M55 Processor Technical Reference Manual	101051	No
Arm® Cortex®-M85 Processor Integration and Implementation Manual	101925	Yes
Arm® Cortex®-M85 Processor Technical Reference Manual	101924	No
Arm® Cortex®-M85 Processor User Guide Reference Material	101927	No
Arm® Cortex®-M System Design Kit Technical Reference Manual	DDI 0479	No
Arm® CryptoCell™-312 Technical Reference Manual	100774	Yes
Arm® Debug Interface Architecture Specification ADIV6.0	IHI 0074	No
Arm® Ethos™-U55 NPU Technical Reference Manual	102420	No
Arm® Platform Security Architecture, Trusted Base System Architecture for Arm®v6-M, Arm®v7-M and Arm®v8-M 2.0	DEN 0083	Yes
Arm® Power Policy Unit Architecture Specification	DEN 0051	No
Arm® Server Base System Architecture 5.0 Platform Design Document	DEN 0029	No
Arm® SSE-123 Example Subsystem Technical Reference Manual	101370	No
Arm®v8-M Architecture Reference Manual	DDI 0553	No
Low Power Interface Specification - Arm® Q-Channel and P-Channel Interfaces	IHI 0068	No



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

The Corstone Reference Systems Architecture Specification Ma1 specifies the architecture of a subsystem that integrates key components available from Arm that can be integrated into a larger system. This document describes a subsystem for Cortex-M processors that support MVE extensions. The subsystem forms the core of a full system and, to form a full system, extra system peripherals must be added as expansion to the subsystem.

- Cortex-M CPU Core with *M-Profile Vector Extension* (MVE), FPU, DSP extensions, Caches, TCMs and ETM. This revision of CRSAS Ma1 supports Cortex-M85 or Cortex-M55 processors
- Ethos NPU core currently supports Ethos-U55 processor
- Multiple banks of System Volatile Memory, for example SRAMs
- Memory Protection Controllers (MPC)
- Manager Security Controller (MSC)
- Exclusive Access Monitor (EAM)
- System interconnect
- Implementation Defined Attribution Unit (IDAU)
- CMSDK Timers and Watchdog timers
- Timestamp-based System Timers and Watchdog timers
- Subsystem Controllers for security and general system control
- CoreSight SoC-600M components
- Power Policy Units, Clock Controller and Low Power Interface interconnect components (PCK-600)

This specification does not include descriptions of other components that are not directly architecturally visible but are necessary to implement the system.

These components are integrated to implement CRSAS Ma1 with the following features:

- TrustZone® aware system, with the system segregated into Secure and Non-secure worlds.
- Support for flexible banked volatile memory. This provides flexibility for the integrator and implementor to trade off between cost, complexity, memory throughput, and power consumption.
- Configurability to allow several features within the system to be included or removed.
- Power Control infrastructure, with several pre-defined voltage and power domains.
- Each switchable power domain has local power policy control, and coordinates with other power domains through a centralized dependency control and/or power interfaces. This provides the system with an autonomous dynamic power control infrastructure that, while being software configurable, aims to minimize software interaction.
- Clock control infrastructure that supports high-level clock control including dynamic clock gating and provides clock request handshakes to clock generators.

- Comprehensive reset generation and control.
- A CoreSight SoC based debug infrastructure that supports a shared Debug Access Port and Trace Port, with support also for cross triggering.

This specification is implemented by the following Arm products:

- Corstone-310

2.1 Document-specific conventions

In addition to the Typographic Conventions section, there are document-specific conventions that apply.

Text

Text between < and > is a label to be replaced with the actual name or value of the item. For example, CPU<n> where “n” is an instance number of either 0 or 1.

- Mathematical expressions are allowed within <> to offset the variable. For example, <NUMCPU+1>.
- <n> always denotes {0-<NUMCPU>}
- <m> always denotes {1-<NUMNPU-1>}

Text between { and } indicates a range to be replaced with the actual legal values. The allowed values can be:

- A range indicate by a start and end with a “-” or in-between. For example, {0-3} allows the any value between 0 and 3. One of the numbers can also be a variable. For example, {0-<NUMCPU>} where NUMCPU is a configuration value between 0 to 3.
- A list of discrete values indicated by a comma separate list. For example, {0,1,2,3} allows the value to be any one of those values. One of the discrete values can also be a range. For example, {0, 3-6, 9} is the same as writing {0, 3, 4, 5, 6, 9}.
- A variable which has constraints. For example, <x> where x is {0-3} or <x> where x is between 0 and 3. Allows x to take any value between 0 and 3.

In this specification the terms subsystem implementor and integrator are used. The definition of these terms are:

- **Subsystem Implementer**

Person implementing a subsystem according to this specification. This does not include the physical implementation of an existing frontend implementation of a subsystem implementation.

- **Subsystem Integrator**

Person integrating this implementation of the subsystem, into a wider SoC, for example, by adding expansion components to the expansion ports.

Numbers

Numbers are normally written in decimal. Binary numbers are preceded by 0b, and hexadecimal numbers by 0x.

For both binary and hexadecimal numbers, where a bit is represented by the letter “x”, the value is irrelevant. For example, a value expressed as 0b1x can be either 0b11 or 0b10. To improve readability, long numbers can be written with an underscore separator between every four characters, for example, 0xFFFF_0000_0000_0000.

Signals

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

- HIGH for active-HIGH
- LOW for active-LOW

A lowercase ‘n’ at the start or end of a signal name denotes an active-LOW signal.

The value of a multi-bit signal is defined using hexadecimal numbers.

When referring to bits within of a multi-bit signal, the following custom is used:

- **<SIGNAL_NAME>[BIT_RANGE]**

Where:

- SIGNAL_NAME is the name of the signal being referred to.
- BIT_RANGE is the bit range being referred to. If BIT_RANGE is omitted, then the text is referring to the whole signal.

For example, when referring to the bit range 31:2 of Debug Access Interface address signal, the text would read:

- **DEBUGPADDR[31:2]**

Registers

When referring to registers and fields, the following convention is used:

- **<REGISTER_NAME>.<FIELD_NAME>[BIT_RANGE]**

Where:

- REGISTER_NAME is the short name of the register being referred to.
- FIELD_NAME is the name of the field within the register being referred to. If the FIELD_NAME is omitted, then the text is referring to the whole register.
- BIT_RANGE is the bit range, within the field, being referred to. If the BIT_RANGE is omitted, then the text is referring to the whole field or to the whole register if FIELD_NAME is omitted as well. When referring to the bit ranges of a field, the field starts at 0.

For example, when referring to the DESIGNER_ID field of the SYSVERSION register, bits 1:0, the text would read:

- SYSVERSION.DESIGNER_ID[1:0]

In register bit field description tables, the following abbreviations are used to describe the accessibility of each bit field:

Table 2-1: Register bit field abbreviations

Abbreviation	Accessibility	Description
R	Read Accessible	Unless stated, these bit fields retain the value written to them until reset or powering down.
RW	Read and Write Accessible	Unless stated, these bit fields retain the value written to them until reset or powering down.
RO	Read only	These can only be read. Unless stated, any writes to these bit fields are ignored. This is similar to WI.
RAZ	Read as Zero	These always return Zeros for reads. Unless stated, writes proceed as normal.
RAO	Read as One	These always return Ones for reads. Unless stated, writes proceed as normal.
RAZ/WI	Read as Zero Write Ignores	These always return Zeros for read and ignores writes.
RAZ/WO	Read as Zero Write Only	These can only be written. These always return Zeros for reads.
W	Write Accessible	Unless stated, these bit fields retain the value written to them until reset or powering down.
WO	Write only	These can only be written. Unless stated, any read from these bit fields returns zeros.
WI	Write Ignore	These ignore writes. Unless stated, reads proceed as normal. This is similar to RO.
W1S	Write 1 to Set	Each bit when written with one is set to one. Writing zeros to these are ignored.
W1C	Write 1 to Clear	Each bit when written with one is cleared to zero. Writing zeros to these are ignored.
W0S	Write 0 to Set	Each bit when written with zero is set to one. Writing ones to these are ignored.
W0C	Write 0 to Clear	Each bit when written with zero is cleared to zero. Writing ones to these are ignored.

Unless stated otherwise, after a register bit field is modified by a register access, it retains its values until a reset is applied or power is lost.

In this specification areas of the design, register values or behaviours, that are described as IMPLEMENTATION DEFINED (IMPL_DEF). It is up to the Subsystem Implementer what the value or behaviour is, but in certain situations the specification requires an implementation to select from a list of options. Each implementation must specify the behaviour or register values as part of its own specification.

In this specification areas of the design, register values or behaviours, are described as CONFIGURATION DEFINED (CFG_DEF). The value or behaviour is dependent upon the configuration of the component.

Register values are defined using actual values that are written in or read from the registers. They are expressed as numbers, either as binary numbers or hexadecimal numbers. Alternatively, for single-bit values, they can be expressed as either 1 or 0, representing 0b1 and 0b0 respectively.

2.2 Compliance

The system architecture described in this document is designed to meet the following requirement:

- [Trusted Base System Architecture for Armv8-M](#)

2.2.1 Trusted Base System Architecture for Armv8-M

The *Arm® Platform Security Architecture, Trusted Base System Architecture for Arm®v6-M, Arm®v7-M and Arm®v8-M 2.0* (TBSA-M) is part of Arm Platform Security Architecture (PSA).

TBSA-M implements best practice security principles when designing systems around Armv8-M processing elements (PEs). These principles support the design and integration of the following features rooted in hardware:

- Root of Trust (RoT)
- A protected key store
- Isolation between Secure and Non-secure software components
- A Secure firmware update mechanism
- A lifecycle management mechanism, for Secure control of debug, test, and access to provisioned secrets
- A high-entropy random number generator, for reliable cryptography
- Cryptographic acceleration

CRSAS Ma1 specifies a system architecture that only partly fulfills the requirements specified within the TBSA-M. However, it specifies many features that help form the core of a system that complies.

For a system that integrates an CRSAS Ma1 based subsystem to comply with TBSA-M, the integrator, when expanding the system is required to continue to complying with TBSA-M requirements. For example:

- Suitable fuses are required to be added to CryptoCell. While CryptoCell handles error detection, you must ensure the fuses cannot be unprogrammed and they are reliable and confidential.
- When adding more managers and subordinates to the system, the memory space continues to obey Secure and Non-secure world partitioning.

Adherence to TBSA-M requirements helps to create a secure system, but it does not create a system that mitigates Denial of Service (DoS) attacks. As such, DoS is out of scope of CRSAS Ma1.

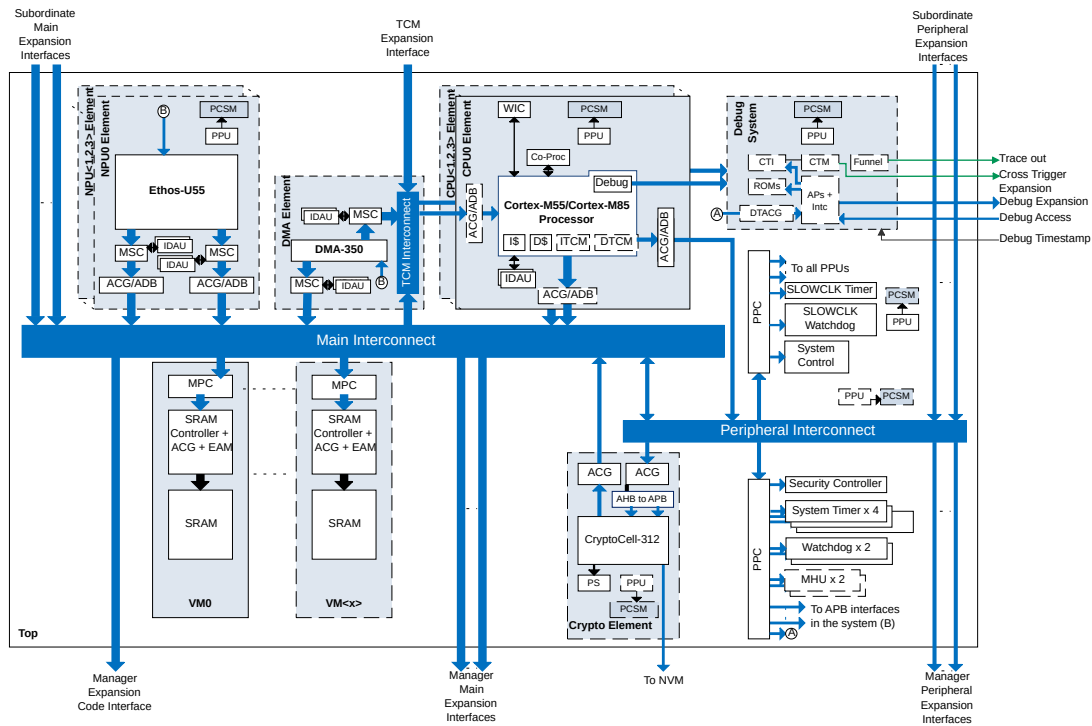
2.3 Topology

This section describes the topology of CRSAS Ma1.

2.3.1 System block diagram

The following figure shows a representative system block diagram of a fully featured CRSAS Ma1 based IoT Subsystem.

Figure 2-1: Representative CRSAS Ma1 based system topology



The subsystem can be divided up into the following key groups of functionalities, which in this document is also referred to as elements.



Elements are used simply in this document to group functionalities that are closely related to each other or due to their common configurability.

NPU element

Each NPU element contains an Ethos-U55 NPU and its associated private infrastructure to integrate the NPU into the system. CRSAS Ma1 supports zero to four NPU elements within the system.

CPU element

Each CPU element contains an Armv8.1-M processor and its associated private infrastructure to integrate the core into the system. CRSAS Ma1 supports one to four CPU elements within the system.

Main interconnect element

Connects all parts of the system to each other.

Peripheral interconnect element

Provides access to lower performance and often device type peripherals within the system.

DMA element

Provides DMA functionality and system and expansion access to the CPU TCMs. The TCM interconnect shown in this element can be merged with the Main interconnect as long as the connectivity shown is maintained.

Volatile memory bank element

Volatile memory banks collectively implement the main volatile storage within the subsystem and implement the functionality to manage the volatile storage.

Peripherals element

Defines a common set of peripherals expected in the system.

Security and system control element

Defines the infrastructure required to implement all configure, control, and monitor system states. These include security, clock, reset, and power control.

Debug system element

Defines the debug infrastructures that allow each CPU and the system to be debugged securely.

CryptoCell element

Contains a CryptoCell core and its associated infrastructure to integrate the core into the system.

3 Configuration options

The CRSAS Ma1 specification is configurable, which allow systems based on this specification to scale across the performance, power, and area requirement of the market.

An implementation of CRSAS Ma1 can support a subset of the configuration options defined here. An implementation of the subsystem, however, cannot support additional configuration options unless they are limited to micro-architectural features and are features that are orthogonal to existing configuration. Because of this, an implementation cannot add additional legal values to existing configuration options. The method by which the configurations are supported is

IMPLEMENTATION DEFINED.

Table 3-1: CRSAS Ma1 configurations

Configuration option name	Legal values	Description
NUMCPU	0-3	It describes the number of Cortex-M CPU cores in the subsystem. The number of cores is equal to NUMCPU + 1. When PILEVEL = 0, NUMCPU must be set to '0'.
CPU<n>TYPE	0, 3, 4	Describes the type of CPU that is integrated: <ul style="list-style-type: none"> 0: Not implemented 3: Cortex-M55 4: Cortex-M85 Others: Reserved CPU0TYPE must not be '0' and sparse CPUs are not supported. Therefore CPU0TYPE to CPU<NUMCPU>TYPE must not be '0's.
NUMNPU	0-4	Describes the number of Ethos NPU cores in the subsystem. The number of cores is equal to NUMNPU.
NPU<m>TYPE	0-1	Describes the type of NPU that is integrated, if any: <ul style="list-style-type: none"> 0: Not implemented 1: Ethos-U55 Others: Reserved This configuration option must exist when NUMNPU != 0. Sparse NPUs are not supported. Therefore, NPU0TYPE must not be '0's.
NPU<m>PORSLRST	0-1	Describes the default security level that each NPU resets to. <ul style="list-style-type: none"> 0: Secure State 1: Non-secure State Others: Reserved This configuration option must exist when NUMNPU != 0.

Configuration option name	Legal values	Description
NPU<m>PORPLRST	0-1	<p>Describes the default privilege level that each NPU resets to.</p> <ul style="list-style-type: none"> 0: Unprivileged State 1: Privileged State Others: Reserved <p>This configuration option only exists when NUMNPU != 0.</p> <p>We recommend that this is set to 1.</p>
NUMDMA	0-1	Describes the number of DMA cores present in the system.
DMATYPE	0-1	<p>Describes the type of DMA that is integrated:</p> <ul style="list-style-type: none"> 0: Not implemented 1: DMA-350 Others: Reserved <p>This configuration option must exist when NUMDMA != 0.</p>
PILEVEL	0, 1, 2	<p>Power Infrastructure Level. Defines the implemented power structure of the system:</p> <ul style="list-style-type: none"> 0: Basic Power Structure 1: Intermediate Power Structure 2: Advance Power infrastructure Others: Reserved
NUMVMBANK	0-4	Selects the number of Volatile Memory Banks.
VMADDRWIDTH	14 to (24- ceil(log ₂ (NUMVMBANK)))	Defines the address width for all Volatile Memory Banks when NUMVMBANK > 0. This then defines the size of each bank as 2 ^{VMADDRWIDTH} bytes.
VMMPCBLKSIZE	3-15	Defines the Block size of the MPC associated with all Volatile Memory Banks. Volatile Memory Block size = 2 ^(VMMPCBLKSIZE + 5) bytes.
HASCRYPTO	0, 1	<p>Defines whether CryptoCell-312 is included.</p> <ul style="list-style-type: none"> 0: No 1: Yes
HASCSS	0, 1	<p>Defines whether the CoreSight SoC-600 based Debug infrastructure is included.</p> <ul style="list-style-type: none"> 0: No 1: Yes <p>HASCSS must be to 1 when NUMCPU > 0.</p>
HASCPU<n>IWIC	0, 1	<p>Defines if each CPU has IWIC.</p> <ul style="list-style-type: none"> 0: No 1: Yes
PDCMQCHWIDTH	0-4	<p>It selects the width of Power Dependency Control Matrix Q-Channel interface that the system supports.</p> <p>When set to '0', the Power Dependency Control Matrix Q-Channel interface does not exist.</p>

Configuration option name	Legal values	Description
INITSVTOR<n>RST[31:7]	Any address values that resides in Secure world and is executable.	The value of CPU <n> Secure Vector table offset address register in the System Control Register.
DBGENSELDIS	When HASCRYPTO = 1, the only legal value is 1. Else, both 0 and 1 are legal.	The DBGEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force DBGEN to use DBGENIN
NIDENSELDIS	When HASCRYPTO = 1, the only legal value is 1. Else, both 0 and 1 are legal.	The NIDEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force NIDEN to use NIDENIN
SPIDENSELDIS	When HASCRYPTO = 1, the only legal value is 1. Else, both 0 and 1 are legal.	The SPIDEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force SPIDEN to use SPIDENIN
SPNIDENSELDIS	When HASCRYPTO = 1, the only legal value is 1. Else, both 0 and 1 are legal.	The SPNIDEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force SPNIDEN to use SPNIDENIN
DAPACCENSELDIS	When HASCRYPTO = 1, the only legal value is 1. Else, both 0 and 1 are legal.	The DAPACCEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force DAPACCEN to use DAPACCENIN
DAPDSSACCENSELDIS	When HASCRYPTO = 1, the only legal value is 1 only. Else, both 0 and 1 are legal.	The DAPDSSACCEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force DAPDSSACCEN to use DAPDSSACCENIN
SYSDSSACCENSELDIS	When HASCRYPTO = 1, the only legal value is 1. Else, both 0 and 1 are legal.	The SYSDSSACCEN Selector Disable. <ul style="list-style-type: none"> 0: No 1: Yes, force SYSDSSACCEN<n> and SYSDSSACCENX to use SYSDSSACCENIN<n> and SYSDSSACCENXIN respectively. <p>This configuration option must exist when HASCSS = 1.</p>
NSMSCEXPST[15:0]	0, 1 for each bit	The reset value for NSMSCEXP.NS_MSCEXP[15:0]. This value defines the security world of each expansion MSC when the PD_SYS power domain is powered up or is reset. It defines up to 16 MSCs, where each bit is: <ul style="list-style-type: none"> 0: Secure 1: Non-secure

Configuration option name	Legal values	Description
ACWAITNRST	0, 1	The reset value of Bus access wait at reset. This defines whether the system blocks access from expansion managers that implement access gating into the Main and Peripheral Interconnect when the PD_SYS power domain is powered up or is reset. <ul style="list-style-type: none"> 0: Blocks access 1: Allows access
CPU<n>EXPNUMIRQ	Zero to Maximum number of interrupts the CPU can support, minus 32.	It specifies the number of expansion interrupts for each CPU.
CPU<n>EXPIRQDIS	One bit per expansion interrupt, with each set to '0' or '1'.	It specifies for each CPU whether each expansion interrupt bit is implemented or disabled.
CPU<n>INTNMIENABLERST	0, 1	It specifies the Warm reset value of NMI_ENABLE.CPU<n>_INTNMI_ENABLE. This determines whether the internally generated interrupt sources can raise the NMI interrupt on each CPU: <ul style="list-style-type: none"> 0: Internal NMI sources are masked from driving NMI 1: Interrupt NMI sources can drive NMI
CPU<n>EXPNMIENABLERST	0, 1	It specifies the Warm reset value of NMI_ENABLE.CPU<n>_EXPNMI_ENABLE. This determines whether the CPU<n>EXPNMI top level pin is able to raise an NMI interrupt on each CPU: <ul style="list-style-type: none"> 0: CPU<n>EXPNMI is masked from driving NMI. 1: CPU<n>EXPNMI is allowed to drive NMI.
CPU<n>WAITRST	0, 1	The (Primary) wait of the CPU at boot control register CPU<n>WAIT reset value. <ul style="list-style-type: none"> 0: Boot normally 1: Wait at boot <p>We recommend that this is set to '0', unless there are other reasons in the system or SoC to initially stop a processor from booting post reset. For example, there may be another higher security entity in the SoC that wants access to the system prior to allowing the CPUs to boot.</p>
CPU<n>CPUIDRST	It is defined by the SoC integrator.	A unique identity value defined for each CPU in the system. Defines the values read at each CPU <n> local's CPU<n>_IDENTITY.CPUID register. Legal values are within 0 to 15, inclusive.
LOCKDCAIC	0, 1	Disable access to all processor's instruction cache direct cache access registers DCAICLR and DCAICRR. Asserting this signal prevents direct access to the instruction cache Tag or Data RAM content. This is required when using eXecutable Only Memory (XOM).

Configuration option name	Legal values	Description
COLDRESET_MODE	0, 1	<p>Cold Reset Mode. It defines if the watchdog timeouts, RESETREQ signal, and writes to the SWRESET register can be used to trigger a system Cold reset and therefore drive nCOLDRESETAON:</p> <ul style="list-style-type: none"> 0: Watchdogs, RESETREQ signal, and the SWRESETREQ register value contributes to Cold Reset. 1: Watchdogs, RESETREQ signal, and the SWRESETREQ register value does not contribute to Cold Reset. <p>When set to '1', an entity outside the subsystem is expected to observe the following signals to decide when to drive HOSTRESETREQ:</p> <ul style="list-style-type: none"> NSWDRSTREQSTATUS SWDRSTREQSTATUS SSWDRSTREQSTATUS RESETREQSTATUS SWRSTREQSTATUS
DEBUGLEVEL	0, 1, 2	<p>It selects the debug level of the subsystem:</p> <ul style="list-style-type: none"> 0: Debug System does not exist. There is no Debug Access and no Trace support. 1: Debug system exists without Trace support. Debug Access Interface(s) exists, but Trace is not supported. 2: Debug system exists with Trace support. Both Debug Access Interface(s) and Trace Interface exist.
MPCEXPDIS[15:0]	0, 1 for each bit	It disables support for individual bits on the SMPCEXPSTATUS bus. If MPCEXPDIS[i] = 1'b1, then either SMPCEXPSTATUS[i] does not exist, or if it does exist, is not used.
MSCEXPDIS[15:0]	0, 1 for each bit	It disables support for individual bits on the SMSCEXPSTATUS , SMSCEXPCLR and NSMSCEXP buses. If MSCEXPDIS[i] = 1'b1, then either SMSCEXPSTATUS[i] , SMSCEXPCLR[i] and NSMSCEXP[i] does not exist, or if they do exist, SMSCEXPSTATUS[i] is not used, SMSCEXPCLR[i] are tied LOW and NSMSCEXP[i] are tied HIGH.
BRGEXPDIS[15:0]	0, 1 for each bit	It disables support for individual bits on the BRGEXPSTATUS and BRGEXPCLR buses. If BRGEXPDIS[i] = 1'b1, then either BRGEXPSTATUS[i] and BRGEXPCLR[i] does not exist, or if they do exist, BRGEXPSTATUS[i] is not used and BRGEXPCLR[i] are tied LOW.
PERIPHPPCEXP{0-3}DIS[15:0]	0, 1 for each bit	It disables support for individual bits on the PERIPHNSPPCEXP{0-3} and PERIPHPPCEXP{0-3} buses. If PERIPHPPCEXP{0-3}DIS[i] = 1'b1, then either PERIPHNSPPCEXP{0-3}[i] and PERIPHPPCEXP{0-3}[i] does not exist, or if they do exist, PERIPHNSPPCEXP{0-3}[i] and PERIPHPPCEXP{0-3}[i] are not used and tied LOW.
MAINPPCEXP{0-3}DIS[15:0]	0, 1 for each bit	It disables support for individual bits on the MAINNSPPCEXP{0-3} and MAINPPCEXP{0-3} buses. If MAINPPCEXP{0-3}DIS[i] = 1'b1, then either MAINNSPPCEXP{0-3}[i] and MAINPPCEXP{0-3}[i] does not exist, or if they do, MAINNSPPCEXP{0-3}[i] and MAINPPCEXP{0-3}[i] are not used and tied LOW.
CPU<n>CLKCFGRST[3:0]	0x0 – 0xF	CLK_CFG0.CPU<n>CLKCFG reset value. CPU<n>CLKCFGRST defines the CPU<n>CLKCFG output expected to be used for clock divider or generation configuration of CPU<n>CLK .
SYSCLKCFGRST[3:0]	0x0 – 0xF	CLK_CFG1.SYSCLKCFG reset value. SYSCLKCFGRST defines the SYSCLKCFG output expected to be used for clock divider or generation configuration of SYSCLK .

Configuration option name	Legal values	Description
AONCLKCFGRST[3:0]	0x0 – 0xF	CLK_CFG1.AONCLKCFG reset value. AONCLKCFGRST defines the AONCLKCFG output expected to be used for clock divider or generation configuration of AONCLK .
CPU<n>RSTREQENRST	0, 1	CPU<n>RSTREQEN reset value. CPU<n>RSTREQENRST defines the reset value of RESET_MASK.CPU<n>RSTREQEN that is used to mask the system reset request signal, often with the signal name SYSRESETREQ , from CPU <n>. When set to '0' at reset, CPU<n> is not able to cause a system warm reset when setting its local AIRCR.SYSRESETREQ control in its Application Interrupt and Reset Control Register (AIRCR). Setting this to '1' allows the reset to be requested. This control must be set to '0' when HASCRYPTO = 1.
HASCPU<n>CPIF	0, 1	It specifies whether the coprocessor interface is included for CPU<n>: <ul style="list-style-type: none"> 1: Coprocessor interface is not included 2: Coprocessor interface is included
SOCIMPLID[11:0]	It is defined by the SoC integrator.	The SoC integrator JEP106 code.
SOCREV[3:0]	It is defined by the SoC integrator.	The SoC minor revision code.
SOCVAR[3:0]	It is defined by the SoC integrator.	The SoC variant or major revision code.
SOCPTID[11:0]	It is defined by the SoC integrator.	The SoC product identity code.
IMPLID[11:0]	It is defined by the implementor.	The subsystem implementor JEP106 code.
IMPLREV[3:0]	It is defined by the implementor.	The subsystem minor revision code.
IMPLVAR[3:0]	It is defined by the implementor.	The subsystem variant or major revision code.
IMPLPTID[11:0]	It is defined by the implementor.	The subsystem product identity code.
CPU<n>MCUROMADDR[31:12]	It is defined by the SoC integrator.	The address pointer to MCU ROM table private to each CPU core. Only the 20 most significant bits are configurable. All lower address bits are zeros. This configuration point must exist when HASCSS = 1.
CPU<n>MCUROMVALID	It is defined by the SoC integrator.	The address pointer to MCU ROM table private to each CPU core is valid. This configuration point must exist when HASCSS = 1.

4 Interfaces

The subsystem has several interfaces. This section provides the associated properties of each interface such as address and data width, along with the clock, power and reset domain that each belongs to. Some interfaces are only required under certain configurations. It is **IMPLEMENTATION DEFINED** whether an interface which is not required is either:

- Not implemented
- Implemented but the signals are tied off

In this section, the following conventions are used:

AMBA Manager Interface

An AMBA interface that is described as a manager interface is one where the subsystem is the manager and must be connected to a subordinate interface.

For an AXI manager interface the **ARVALID** signal is an output and **ARREADY** is an input.

For an AHB manager interface the **HADDR** signal is an output and **HRDATA** is an input.

For an APB manager interface, the **PADDR** signal is an output.

AMBA Subordinate Interface

An AMBA interface that is described as a subordinate interface is one where the subsystem is the subordinate and must be connected to a manager interface.

For an AXI subordinate interface the **ARVALID** signal is an input and **ARREADY** is an output.

For an AHB subordinate interface the **HADDR** signal is an input and **HRDATA** is an output.

For an APB subordinate interface, the **PADDR** signal is an input.

LPI Control Interface

An LPI interface that is described as a control interface, is one where the subsystem is the device and must be connected to a control interface.

For a Q-Channel control interface the **QREQn** signal is an input and **QACCEPTn**, **QDENY**, and **QACTIVE** are outputs.

For a P-Channel control interface the **PREQn** signal is an input and **PACCEPTn**, **PDENY**, and **PACTIVE** are outputs.

LPI Device Interface

An LPI interface described as a device interface, is one where the subsystem is the controller and must be connected to a device interface.

For a Q-Channel device interface the **QREQn** signal is an output and **QACCEPTn**, **QDENY** and **QACTIVE** are inputs.

For a P-Channel device interface the **PREQn** signal is an output and **PACCEPTn**, **PDENY** and **PACTIVE** are inputs.

An implementation of CRSAS Ma1 can add additional interface as long as it does not affect the behavior of these interfaces. For example, the processor core used in this system often provides more processor state outputs. These can always be made available for expansion but their existence is **IMPLEMENTATION DEFINED**.

4.1 Input and output clocks

The following table lists all the clock inputs into the subsystem.

Table 4-1: CRSAS Ma1 input clocks

Clock Name	Target Power ¹ Domain PILEVEL= 2	Description
SLOWCLK	PD_AON	Slow Clock. An always active slow clock input that is completely asynchronous to the other clocks in the system. This is one of the only two clocks expected to be active in the lowest power state of the system, HIBERNATE{0,1}, and is used primarily by timers that reside in the PD_AON domain.
AONCLK	PD_AON	Always ON Clock Input. This clock is used for logic in the PD_AON domain that is not running on SLOWCLK . This allows the rest of the PD_AON domain to run on a faster clock and yet be independent from the rest of the system.
CNTCLK	PD_AON	System Counter Timestamp Clock associated with the CNTVALUE<G/B> System Counter Timestamp input.
SYSCLK	PD_MGMT ²	Main System Clock Input. This clock is the main clock used to drive the main system that resides in PD_SYS. This clock is also used for logic in the PD_MGMT domain if it exists, or logic that is merged from PD_MGMT to PD_AON if PILEVEL < 2.
DEBUGCLK	PD_DEBUG	Debug System Clock Input. This clock is used to drive all logic in the debug System. This input clock must exist when HASCSS = 1.
CPU<n>CLK	PD_CPU<n>	CPU clock. This is the clock used to drive each CPU.
NPU<m>CLK	PD_NPU<m>	NPU clock. This is the clock used to drive each NPU. This clock only exists if NUMNPU > 0.

¹ When PILEVEL = 0, power domains are merged as follows:

- PD_MGMT is merged with PD_AON and all reference to PD_MGMT is replaced by PD_AON.
- PD_CPU<n> is merged with PD_SYS and all reference to PD_CPU<n> is replaced by PD_SYS.

² When PILEVEL = 1, power domains are merged as follows:

- PD_MGMT is merged with PD_AON and all reference to PD_MGMT is replaced by PD_AON.

The relation between these clocks is **IMPLEMENTATION DEFINED** with the architecture able to support all clocks being completely asynchronous to each other. However, during implementation, to reduce the number of clock sources, the implementor, or the SoC Integrator can drive several clocks using the same clock sources, so long as the implementor takes clock availability, power and response time into consideration.

In typical use, we recommend that **SLOWCLK** is driven using a 32kHz clock source. If reducing standby power is an important consideration for a product, **AONCLK** can be driven at a lower clock rate (around 1MHz to 10MHz) compared to **SYSCLK** to improve the transition time entering and leaving the lowest power state of the system, but still support the use of very low leakage implementation library cells. Alternatively, **AONCLK** can be driven using the same clock source as **SYSCLK**. If there is no requirement to run the processors and System Timestamp Counter at a different speed to the system, then **CPU<n>CLK** and **CNTCLK** can also be driven using the same clock source as **SYSCLK**. **DEBUGCLK** can also be driven using the same clock as **CPU<n>CLK**. **NPU<m>CLK** may be driven from the same clock source as **SYSCLK**.

At a minimum, only two clock sources are needed. For example, one at 32KHz for **SLOWCLK** and another at 200MHz for the rest. If for example, very large SRAMs with higher access time are needed in the system, **SYSCLK** could be clocked synchronously to and at a fixed fraction of **CPU<n>CLK**.

The following table also shows for each clock the target power domain that clock is expected to be used.



All clocks always first enter either through the PD_AON domain before being used in their respective power domain.

The following table lists all the clock outputs from the subsystem.

Table 4-2: CRSAS Ma1 output clocks

Clock Name	Power Domain expected to be used in ¹ PILEVEL = 2	Description
MGMTSYSCLK	PD_MGMT ¹	Gated version of SYSCLK expected to be used to drive expansion logic that resides in the PD_MGMT power domain when PILEVEL = 2.
SYSSYSCLK	PD_SYS	Gated version of SYSCLK expected to be used to drive expansion logic that resides in the PD_SYS power domain.
DEBUGDEBUGCLK	PD_DEBUG	Gated version of DEBUGCLK expected to be used to drive debug expansion logic that resides in the PD_DEBUG power domain. This output clock must exist when HASCSS = 1.
CPUCPU<n>CLK	PD_CPU<n> ¹	Gated version of CPU<n>CLK expected to be used to drive expansion logic that resides in the PD_CPU<n> powerdomain.
DEBUGCPU<n>CLK	PD_DEBUG	Gated version of CPU<n>CLK expected to be used to drive debug expansion logic that resides in the PD_DEBUG power domain. This output clock must exist when HASCSS = 0.
CRYPTOSYSCLK	PD_CRYPTO ¹	Gated version of SYSCLK expected to be used to drive expansion logic that resides in the PD_CRYPTO power domain. This clock must exist when HASCRYPTO = 1.

¹ When PILEVEL = 0, power domains are merged as follows:

- PD_MGMT is merged with PD_AON and all reference to PD_MGMT is replaced by PD_AON.
- PD_CPU<n> is merged with PD_SYS and all reference to PD_CPU<n> is replaced by PD_SYS.

- PD_CRYPTO is merged with PD_SYS and all reference to PD_CRYPTO, if exist, is replaced by PD_SYS.

When PILEVEL = 1, power domains are merged as follows:

- PD_MGMT is merged with PD_AON and all reference to PD_MGMT is replaced by PD_AON,
- PD_CRYPTO is merged with PD_SYS and all reference to PD_CRYPTO, and all references to PD_CRYPTO are replaced by PD_SYS.

CRSAS Ma1 provides a Q-Channel interface for each of the output clocks to allow expansion logic to control the availability of each clock output, see [Clock control Q-Channel device interfaces](#).

4.2 Functional reset inputs and outputs

Reset inputs are used to drive or to request for resets while reset outputs are used to reset expansion logic that resides in specific power domains. For more information, see [Reset infrastructure](#).

Unless explicitly stated, it is **IMPLEMENTATION DEFINED** if these resets are asynchronous or synchronous. We recommend that all resets are asynchronously asserted and synchronously deasserted in relation to the reset domain's clock, except for reset requests that end with the "REQ" suffix.

The following table lists all the reset interfaces on the subsystem.

Table 4-3: Reset signals

Signal name	Power Domain where it is used when ¹ PILEVEL = 2	Direction	Description
nPORESET	PD_AON	Input	Active LOW Power-On Reset Input.
nSRST	PD_AON	Input	Active LOW Cold Reset request Input from Debugger.
HOSTRESETREQ	PD_AON	Input	Higher Authority request to perform a Cold reset, for example from a hosting system. This is an asynchronous input. This reset request always results in a system Cold reset regardless of the COLDRESET_MODE configuration. After it is asserted, the signal must be held high until the reset occurs on nCOLDRESETAON. Holding this input HIGH also results in the system being held in Cold reset.
RESETREQ	PD_AON	Input	Active-HIGH Hardware Request Input to perform a Cold reset. This is an asynchronous input. After it is asserted, the signal must be held HIGH until the reset occurs on nCOLDRESETAON. The signal must be cleared because of the assertion of nCOLDRESETAON. When COLDRESET_MODE = 1, this input has no effect on Cold reset directly.
EXPWARMRESETREQ	PD_AON	Output	Active-HIGH Request to expansion logic to prepare for a Warm reset. This input is associated with EXPWARMRESETACK. This signal is an asynchronous output. The existence of this signal and the associated acknowledge signal is IMPLEMENTATION DEFINED .

Signal name	Power Domain where it is used when ¹ PILEVEL = 2	Direction	Description
EXPWARMRESETACK	PD_AON	Input	Active-HIGH Acknowledge for expansion logic to indicate that it is ready for Warm reset. This input is associated with EXPWARMRESETREQ . This is an asynchronous input. The existence of this signal and the associated request signal is IMPLEMENTATION DEFINED .
nCOLDRESETAON	PD_AON	Output	Active LOW Cold Reset for the Expansion System. This Cold reset merges other reset sources within the system with nPORESET to generate this reset in the PD_AON domain. This reset is expected to be used in the PD_AON domain.
nWARMRESETAON	PD_AON	Output	Active LOW Warm reset for the Expansion System. This Warm reset is generated by combining Warm reset requests from the system along with Cold reset in the PD_AON domain. This reset is expected to be used in the PD_AON domain.
nCOLDRESETMGMT	PD_MGMT ¹	Output	Active LOW Cold reset for the PD_MGMT power domain. This reset is expected to be used in the PD_MGMT domain.
nWARMRESETMGMT	PD_MGMT ¹	Output	Active LOW Warm reset for the PD_MGMT power domain. This reset is expected to be used in the PD_MGMT domain.
nCOLDRESETSYS	PD_SYS	Output	Active LOW Cold reset for the PD_SYS power domain. This reset is expected to be used in the PD_SYS domain.
nWARMRESETSYS	PD_SYS	Output	Active LOW Warm reset for the PD_SYS power domain. This reset is expected to be used in the PD_SYS domain.
nCOLDRESETDEBUG	PD_DEBUG	Output	Active LOW Cold reset for the PD_DEBUG power domain. This reset is expected to be used in the PD_DEBUG domain. This output must exist when HASCSS = 1.
nWARMRESETCRYPTO	PD_CRYPTO ¹	Output	Active LOW Cold reset for the PD_CRYPTO power domain. This reset is expected to be used in the PD_CRYPTO domain. This reset must exist when HASCRYPTO = 1.
nWARMRESETCPU<n>	PD_CPU<n> ¹	Output	Active LOW Warm reset for each of the PD_CPU<n> power domain. These resets are expected to be used in the PD_CPU<n> domains.
nCOLDRESETCPU<n>	PD_CPU<n> ¹	Output	Active LOW Cold reset for each of the PD_CPU<n> power domain. These resets are expected to be used in the PD_CPU<n> domains.
nCOLDRESETDEBUGCPU<n>	PD_DEBUG	Output	Active LOW Cold reset for PD_DEBUG associated with the debug logic of each CPU. These are expected to be used in the PD_DEBUG domain.

¹ When PILEVEL = 0, power domains are merged as follows:

- PD_MGMT is merged with PD_AON and all reference to PD_MGMT is replaced by PD_AON.
- PD_CPU<n> is merged with PD_SYS and all reference to PD_CPU<n> is replaced by PD_SYS.
- PD_CRYPTO is merged with PD_SYS and all reference to PD_CRYPTO, if exist, is replaced by PD_SYS.

When PILEVEL = 1, power domains are merged as follows:

- PD_MGMT is merged with PD_AON and all reference to PD_MGMT is replaced by PD_AON.
- PD_CRYPTO is merged with PD_SYS and all reference to PD_CRYPTO, if exist, is replaced by PD_SYS.

4.3 P-Channel and Q-Channel Device Interfaces

Each P-Channel or Q-Channel Device Interface independently allows external expansion logic to handshake with the system to ensure that:

For warm reset control

All external managers and subordinate interfaces in a Warm reset domain are in safe state and isolated from the cold reset domain before asserting Warm reset. It also allows external manager to request for a power domain to wake.

For clock control

All external dependent logic can prepare itself before the hierarchical gating of clocks.

During system integration, expansion logic that resides within a power or clock domain associated with each P-Channel or Q-Channel normally merges all P-Channel or Q-Channel interfaces within the expansion domain to drive each interface.

When merging, the following rules must be obeyed to prevent system deadlocks:

- All bus managers in the expansion system must, either individually or collectively have a full Q-Channel interface, or a P-Channel interface.

Each Q-Channel interface must be able to deny a quiescence request if the logic that it controls has outstanding operations on the bus or is unable to enter quiescent state for any other reason. Similarly, each P-Channel interface must be able to deny a request to enter a different PSTATE if the logic is unable to enter the requested state. The exception is the WARM_RST power state where managers are expected to suspend their activity cleanly, activate internal reset domain crossing protections and always accept the request.

Each P-Channel interface for power control must be able to support all power states that the domain implements.

- All bus subordinates in the expansion system must, either individually or collectively have a Q-Channel or P-Channel interface that must be able to delay the acceptance of a quiescence request or a Power state request if the current bus operation is about to complete.

The LPI interface must also be able to deny a quiescence request or a power state request if the logic that it controls has other outstanding operations that prevent it from entering quiescent or the requested state. The exception is the WARM_RST power state where subordinates are expected to suspend their activity cleanly, activate internal reset domain crossing protections and always accept the request.

- You must sequence the Q-Channels associated with these external bus interfaces in such a way to ensure that all bus managers are in quiescent state before any bus subordinates are requested to enter quiescent state.

Similarly, with P-Channels, you must ensure that all bus managers and subordinate of these interfaces can enter the new requested state in the right order depending on their dependencies before the P-Channel accepts entering that state.

An implementation of the system can depend on external sequencing to be added by a system integrator to do this for their expansion logic, or alternatively, implement multiple Q-Channels or P-Channel interfaces for each domain to separately handshake managers, subordinates, and even intermediate components in the domain in sequence.

If a Q-Channel Device interface is not used, its associated **QACTIVE** and **QDENY** signals must be tied LOW and the **QREQn** output looped back into its **QACCEPTn** input. Similarly, if a P-Channel Device interface is not used, then its associated **PACTIVE** and **PDENY** must be tied LOW, and the **PREQn** output looped back into its **PACCEPTn** input.



Not used means there is no consumer of the corresponding clock or power, this not to be confused with the case where the consumer of the corresponding clock or power has no LPI interface.

Individual signals of the Q-Channel Device interfaces are sometimes described as a vector. For example, a Q-Channel Device interface could have x bits for each signal. In this case, bit i of all signals within 0 to $x-1$ forms a single bit Q-Channel interface. For example, **QACTIVE[1]**, **QREQn[1]**, **QACCEPTn[1]** and **QDENY[1]** form a Q-Channel interface and there are x Q-Channel interfaces. When a Q-Channel device interface is defined as a vector in this document, unless otherwise stated, it means the following:

- **QACTIVE**[$x-1:0$] are considered as ORed together at the controller,
- Collectively all Q-Channel i has to enter Q_STOPPED state before the entire interface is considered to be in the Q_STOPPED state. And similarly, all Q-Channel i has to enter Q_RUN state before the entire interface is considered to be in the Q_RUN state.
- When transitioning the entire interface between Q_RUN state to Q_STOPPED state, if any of the individual bit i Q-Channel denies the transition, all other Q-Channel must also return to the original Q-Channel state.

When P-Channel Device interface is used, the P-Channel encoding replicates the PCK-600 PPU's Device P-Channel bit assignment, with each DEVACTIVE bit used to request entry to a Power mode and Operating mode, and DEVSTATE vector representing the power mode being requested.

For more information, see *Arm® CoreLink™ PCK-600 Power Control Kit Technical Reference Manual* and *Arm® Power Policy Unit Architecture Specification*.

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

4.3.1 Clock control Q-Channel device interfaces

CRSAS Ma1 defines a Q-Channel Device interface for each of the output clocks to allow expansion logic to control the availability of each clock output. These are used to support high-level clock gating. Each interface can either be single bit or a vector, and is **IMPLEMENTATION DEFINED**.

The following Q-Channels are provided:

- MGMTSYSCLK Q-Channel Device interface for **MGMTSYSCLK**. This interface resides in the PD_MGMT power domain when PILEVEL = 2, or in the PD_AON domain when PILEVEL < 2.
- SYSSYSCLK Q-Channel Device interface for **SYSSYSCLK**. This interface resides in the PD_SYS power domain.
- DEBUGDEBUGCLK Q-Channel Device interface for **DEBUGDEBUGCLK**. This interface must exist when HASCSS = 1. This interface resides in the PD_DEBUG power domain.
- CPU<n>CLK Q-Channel Device interface for **CPU<n>CLK**. This interface resides in the PD_CPU<n> power domain when PILEVEL > 0, or in the PD_SYS power domain when PILEVEL = 0.
- DEBUGCPU<n>CLK Q-Channel Device interface for **DEBUGCPU<n>CLK**. This interface must exist when HASCSS = 0. This interface resides in the PD_DEBUG power domain.
- CRYPTOSYSCLK Q-Channel Device interface for **CRYPTOSYSCLK**. This interface must exist when HASCRYPTO = 1. This interface resides in the PD_CRYPTO power domain.

All clock Q-Channel Device interfaces are for clock control only and do not support waking the system from hibernation.

Each of the Clock Control Device Q-Channel interfaces can be multi-bit and are either an asynchronous interface or synchronous to the clock that each of the Q-Channel interfaces control. The choice is **IMPLEMENTATION DEFINED**.

4.3.2 Power Control Q-Channel and P-Channel Expansion Device interfaces

CRSAS Ma1 provides power control Q-Channel or P-Channel Device interfaces to allow expansion logic to handshake and coordinate the expansion logic power state.

Each power domain that supports expansion is provided with either Q-Channel or P-Channel interfaces as follows:

- MGMTPWR Q-Channel or P-Channel Device interface for PD_MGMT. This interface resides in the PD_MGMT power domain and must exist when PILEVEL = 2. When PILEVEL < 2, this interface is not required, but if it exists, it resides in PD_AON domain and can be tied or used to handshake Warm reset entry for expansion logic residing in the PD_AON domain.
- SYSPWR Q-Channel or P-Channel Device interface for PD_SYS.
- DEBUGPWR Q-Channel or P-Channel Device interface for PD_DEBUG.
- CPU<n>PWR Q-Channel or P-Channel Device interface for PD_CPU<n>. This interface must exist when PILEVEL > 0. When PILEVEL = 0, this interface is not required, but if it exists, it resides in PD_SYS domain and must be loop backed and tied.

- CRYPTOPWR Q-Channel or P-Channel Device interface for PD_CRYPTO. This interface must exist when both PILEVEL = 2 and HASCRYPTO = 1.

These Q-Channel or P-Channel Device interfaces are driven by expansion logic that resides within their respective power domain that they control and are used to do the following:

- Used by the expansion logic to indicate through the **QACTIVE** or **PACTIVE** signal that the expansion logic is IDLE or hint that it wants to enter a different power state.
- For a power controller to request the expansion logic to enter a different power state.
- Allow the expansion logic to accept for deny the request to enter a different power state.

The choice between Q-Channel or P-Channel interface is **IMPLEMENTATION DEFINED**. The Q-Channel or P-Channel interfaces do not support the waking of the power domain, since they reside within the power domain that is being controlled. Instead, associated Power Control Wakeup Q-Channel Device interfaces described in [Power Control Wakeup Q-Channel Device interfaces](#) should be used to request for specific power domains to power up.

Currently, each domain is described as having at least one Q-Channel or P-Channel interface. However, an implementation can provide multiple Q-Channel, multi-bit Q-Channel or multiple P-Channel interfaces per domain in order, for example, to sequence expansion logic when entering lower power states. The number of Power Control Q-Channel or P-Channel per domain is therefore **IMPLEMENTATION DEFINED**.

These Power Control Device Q-Channel or P-Channel interfaces are either asynchronous interfaces or are synchronous to the clock used in each domain that each of the Q-Channel or P-Channel interfaces control. The choice is **IMPLEMENTATION DEFINED**.

4.3.3 Power Control P-Channel Expansion Device Interfaces

CRSAS Ma1 provides power control P-Channel Device interfaces to allow expansion of existing power domain hierarchy with new power domains by coordinating allowed power states of external power domain in the power domain hierarchy.

Each power domain that supports power domain hierarchy extension is provided with a P-Channel interface as follows:

MGMPDHCPWR P-Channel Device interface for PD_MGMT

This interface resides in the PD_MGMT power domain when PILEVEL = 2. When PILEVEL < 2, this interface resides in PD_AON power domain. The interface must only be used for handshaking and coordinating allowed power states of external power domains which are meant to be at the same level as PD_SYS and PD_DEBUG in the power domain hierarchy. The interface is running on **MGMTSYSCLK** and reset on **nCOLDRESETMGMT**.

SYSPDHCPWR P-Channel Device interface for PD_SYS

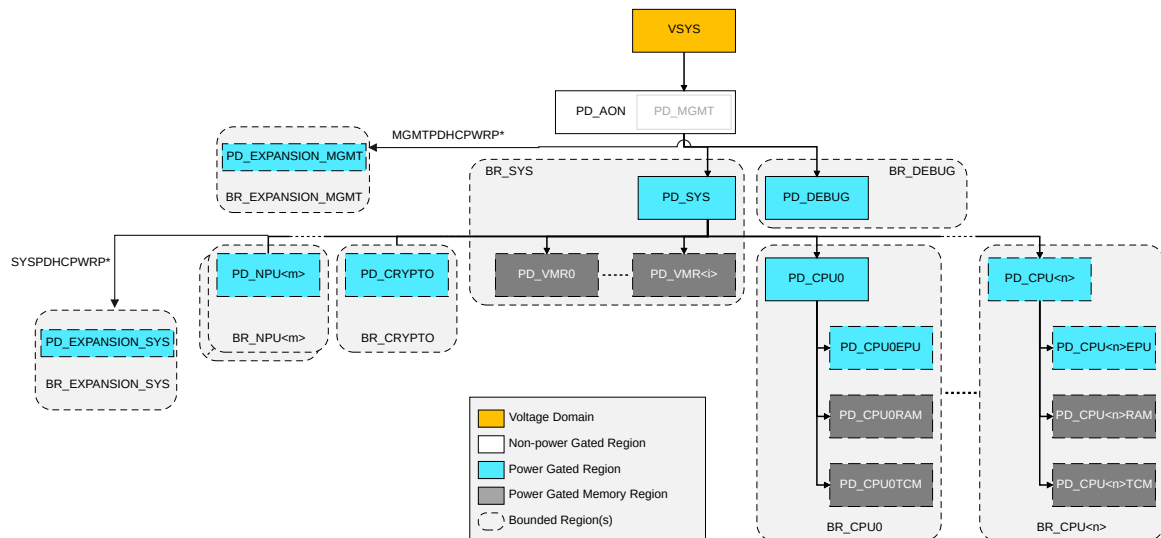
This interface resides in the PD_MGMT power domain when PILEVEL = 2. When PILEVEL < 2, this interface resides in PD_AON power domain. The interface must only be used for handshaking and coordinating allowed power states of external power domains which are meant to be at the same level as PD_NPU and PD_CPU in the power domain hierarchy. The interface is running on **MGMTSYSCLK** and reset on **nCOLDRESETMGMT**.

Currently, each domain is described as having a minimum of a single P-Channel interface. However, an implementation can provide multiple P-Channel interfaces per domain. For example, to enable multiple power domains to be added and controlled. The number of Power Control P-Channel interfaces per domain is therefore **IMPLEMENTATION DEFINED**.

These Device P-Channel interfaces are either an asynchronous interface or are synchronous to the clock used in each domain that each of the P-Channel interface controls. The choice is **IMPLEMENTATION DEFINED**.

For more information on power and voltage domains, please see the following figure and [Power Control Infrastructure](#).

Figure 4-1: Power control P-Channel expansion device interfaces with PILEVEL = 1



4.3.4 Power Control Wakeup Q-Channel Device interfaces

CRSAS Ma1 provides power control Wakeup Q-Channel Device interfaces to allow expansion logic to request for specific power domains to power up. The Q-Channel interfaces and the domains they control are:

- PWRMGMTWAKE Q-Channel Device Interface for PD_MGMT. This interface must exist when PILEVEL = 2.
- PWRSYSWAKE Q-Channel Device Interface for PD_SYS.
- PWRDEBUGWAKE Q-Channel Device Interface for PD_DEBUG.
- PWRCPU<n>WAKE Q-Channel PD_CPU<n>. This interface must exist when PILEVEL > 0.

These interfaces all reside in the PD_AON domain. Because they are power up requests, also referred to as wake up request, we recommend that at minimum, only the **QACTIVE** signal of each interface is implemented. Therefore, when using these to wake a domain, you must drive and hold the appropriate **QACTIVE** signal to request to turn on the domain until the domain is ON. For example, to wake PD_SYS, you must set **PWRSYSWAKEQACTIVE** Q-Channel signal to request to turn ON until the SYSPWR Q-Channel or P-Channel Device Interface for PD_SYS indicates that the power domain is ON.

These Wakeup Device Q-Channel interfaces are either asynchronous or are synchronous to the clock used in each domain that each of the Q-Channel interfaces control. The choice is **IMPLEMENTATION DEFINED**.

4.4 Clock Control Q-Channel Control interfaces

Each Q-Channel Control interface in this section is single bit per signal on the Q-Channel interface and independently allows the system to request for the availability of a clock source. Each Q-Channel Control interface allows an external clock controller to handshake with the system to safely turn the clock source OFF.

Each of the clock control Q-Channel interfaces can be either asynchronous or synchronous to the clock that each of the Q-Channel interfaces control. The choice is **IMPLEMENTATION DEFINED**.

CRSAS Ma1 has the following clock control Q-Channel Control interfaces:

- AONCLK Q-Channel Control interface for **AONCLK**.
- SYSCLK Q-Channel Control interface for **SYSCLK**.
- CPU<n>CLK Q-Channel Control interface for **CPU<n>CLK**.
- NPU<m>CLK Q-Channel Control interface for **NPU<m>CLK**.
- DEBUGCLK Q-Channel Control interface for **DEBUGCLK**. This interface must exist if HASCSS = 1.

If an input clock source is always running, and there is no clock controller associated with this clock input, then you can tie its associated clock control Q-Channel Control interface by tying **QREQn** input to HIGH. These interfaces are in the PD_AON domain.

4.5 Expansion Power Control Dependency interface

CRSAS Ma1 provides an optional set of 2, up to 4-bit width Q-Channel interfaces that allow external power domains to use the Power Dependency Control Matrix to keep power domains within the subsystem from entering a lower power state.

These signals are:

PDCMONQREQn[<PDCMQCHWIDTH-1>:0]

Power Dependency Control Matrix **QREQn** inputs.

When each bit is set to '1', it indicates that the external domain that drives it is in functional power mode.

PDCMONQACCEPTn[<PDCMQCHWIDTH-1>:0]

Power Dependency Control Matrix **QACCEPTn** outputs.

Each bit acknowledges an associated bit on **PDCMONQREQn** by returning the request input value. This acts as a four-phase handshake so that the driver of each request bit can determine that the request has been seen by receiving unit.

PDCMRETQREQn[<PDCMQCHWIDTH-1>:0]

Power Dependency Control Matrix **QREQn** inputs.

When each bit is set to '1', it indicates that the external domain that drives it is in functional power mode.

PDCMRETQACCEPTn[<PDCMQCHWIDTH-1>:0]

Power Dependency Control Matrix **QACCEPTn** outputs.

Each bit acknowledges an associated bit on **PDCMRETQREQn** by returning the request input value. This acts as a four-phase handshake so that the driver of each request bit can determine that the request has been seen by receiving unit.

These signals provides a way for an external power domain to request for a domain within the system to remain powered so that the external domain can access the internal power domain or at least request that the internal domain retain its register context. For example, the external domain may want to access the Main interconnect in the PD_SYS domain.

1. External domain raises an interrupt with the host processor, which if not already ON will wake up PD_SYS.
2. Host processor sets the relevant bit in PDCM_PD_SYS_SENSE.S_PDCMONQREQ{0-<PDCMQCHWIDTH-1>}.

The external system can now keep PD_SYS powered using one of the **PDCMONQREQn** signals.

Similarly, if the external domain required PD_SYS to maintain state, but did not currently require access to it:

1. External domain raises an interrupt with the host processor, which if not already ON will wake up PD_SYS.
2. Host processor sets the relevant bit in PDCM_PD_SYS_SENSE.S_PDCMRETQREQ{0-<PDCMQCHWIDTH-1>}.

The external system can now ensure PD_SYS retains state using one of the **PDCMRETQREQn** signals.

The four-phase handshake must be used to ensure that there is no race condition between ending a bus access that wakes-up the domain and activating the keep-up of the domain.

These are asynchronous signals that either reside in the PD_AON power domain or in the PD_MGMT domain, and the choice is **IMPLEMENTATION DEFINED**. For more information on registers

that use these signals and description of related functionality, see [System Control Register Block](#) and [Power Control Infrastructure](#).

4.6 Power Domain ON Status Signals

CRSAS Ma1 provides a set of output signals that indicates if the power domain each is associated with is in the ON Power Mode.

These signals are:

PDMGMTON

- HIGH indicates that the PPU of the PD_MGMT power domain presumes that the domain is ON
- LOW indicates that the PPU presumes that the power domain is in a lower power state
- This signal must exist when PILEVEL = 2.

PDSYSON

- HIGH indicates that the PPU of the PD_SYS power domain presumes that the domain is ON
- LOW indicates that the PPU of the power domain presumes that the domain is in a lower power state.

PDCPU<n>ON

- HIGH indicates that the PPU of the PD_CPU<n> power domain presumes that the domain is ON
- LOW indicates that the PPU of the power domain presumes that the domain is in a lower power state
- This signal must exist when PILEVEL > 0.

PDNPU<m>ON

- HIGH indicates that the PPU of the PD_NPU<m> power domain presumes that the domain is ON
- LOW indicates that the PPU of the power domain presumes that the domain is in a lower power state
- This signal only exists if NUMNPU > 0.

PDDEBUGON

- HIGH indicates that the PPU of the PD_DEBUG power domain presumes that the domain is ON
- LOW indicates that the PPU of the power domain presumes that the domain is in a lower power state

PDCRYPTOON

- HIGH indicates that the PPU of the PD_CRYPTON power domain presumes that the domain is ON

- LOW indicates that the PPU of the power domain presumes that the domain is in a lower power state
- This signal must exist when PILEVEL = 2 and HASCRYPTO = 1.



These are primarily status signals and are typically driven using **PPUHWSTAT** signals. We recommend that these are not used as power control signals directly.

4.7 System timestamp interface

CRSAS Ma1 provides a system timestamp input from an expansion timestamp counter. This timestamp is expected to be driven by a timestamp generator in the subsystem expansion. This resides in the PD_AON power domain and **nWARMRESETAON** reset domain.

The system timestamp interface has the following properties:

- Name: **CNTVALUE<G/B>[{23-63}:0]**
- Description: Timestamp input value. This value can either be:
 - Gray coded. In this case, the signal name ends with 'G' and is asynchronous.
 - Binary coded. In this case, the signal name ends with 'B' and is synchronous to **CNTCLK**.
- Width: 24-64 bit, IMPL_DEF
- Direction: Input
- Clock domain: **CNTCLK** or is asynchronous

When HASCSS = 1, we recommend that the expansion system uses the CTI triggers to implement timestamp halting. See sections [Cross Trigger Interface](#) and [Cross Trigger](#) for more information on the CTI interface.

When HASCSS = 0, a Debug System does not exist to provide these control signals. Therefore, the system integrator has to depend on the **CPU<n>HALTED** signals to halt the system timestamp generator.

4.8 Main Interconnect Expansion interfaces

CRSAS Ma1 provides a configurable number of Manager and Subordinate Expansion interfaces of the Main Interconnect. These interfaces allow the system integrator to add additional bus managers and bus subordinates to the system.

For more information, see [System interconnect infrastructure](#).

The AMBA protocol used for this interface can either be AHB5 or AXI5 and must at least support the following properties:

- 32-bit address
- 32-bit, 64-bit, or 128-bit data
- Synchronous to **SYSSYSCLK**
- On the **nWARMRESETSYS** reset
- TrustZone Support enabled

We recommend that these interfaces are 64-bit or 128-bit wide and use the AXI5 protocol.

The types of Manager and Subordinate Expansion interfaces on the Main Interconnect are as follows:

Manager Code Main Expansion interface

This interface provides access to Code Memory and is mapped to the following address range:

- 0x0100_0000 to 0x09FF_FFFF
- 0x1100_0000 to 0x19FF_FFFF

There must be at least one such interface on the subsystem. If there are more than one such interfaces, it is **IMPLEMENTATION DEFINED** how the preceding address range is divided across the interfaces. If any implemented interface is not used or any of the preceding region is not implemented, a default subordinate must be used to respond with bus error. This interface must export information to allow a debug access to be distinguished from an access by another manager in the system. We recommend that one such interface is provided in an implementation of the subsystem.

Manager Main Expansion interface

This interface provides access to other subordinates in the system and is mapped to the following address range:

- 0x2800_0000 to 0x2FFF_FFFF
- 0x3800_0000 to 0x3FFF_FFFF
- 0x6000_0000 to 0xDFFF_FFFF

There can be zero or more such interfaces on the subsystem. If there are no such interfaces, all access to the above address range results in decode error. Additionally, if any implemented interface is not used or any of the preceding regions is not implemented, a default subordinate must be used to respond to the access targeting the interface and any address region not implemented with decode error. If there are more than one such interfaces, it is **IMPLEMENTATION DEFINED** how the preceding address range is divided across the interfaces. If implemented, all these interfaces must export information to allow a debug access to be distinguished from an access by another manager in the system. We recommend that one such interface is provided in an implementation of the subsystem.

Subordinate Main Expansion interface

This interface provides access to system from expansion managers. This interface can be used to access all memory mapped regions in the system except for regions private to the

processors. We recommend that one such interface is provided in an implementation of the subsystem. If any implemented interface is not used, the interface must be tied to idle.

4.9 Peripheral Interconnect Expansion interfaces

CRSAS Ma1 provides a configurable number of Manager and Subordinate Expansion interfaces from the Peripheral Interconnect. These interfaces allow the system integrator to add additional bus managers and bus subordinates to the system that is expected to require lower latency access to peripherals.

For more information, see [System interconnect infrastructure](#).

The AMBA protocol used for this interface can either be AHB5 or AXI5 and must at least support the following properties:

- 32-bit address
- 32-bit or 64-bit data
- Synchronous to **SYSSYSCLK**
- On the **nWARMRESETSYS** reset
- TrustZone Support enabled

We recommend that these interfaces are 32-bit wide and use the AHB5 protocol.

The types of Manager and Subordinate Expansion interfaces on the Peripheral Interconnect are as follows:

Manager Peripheral Expansion interface

This interface provides access to other subordinates in the system and is mapped to the following address range:

- 0x4010_0000 to 0x47FF_FFFF
- 0x4810_0000 to 0x4FFF_FFFF
- 0x5010_0000 to 0x57FF_FFFF
- 0x5810_0000 to 0x5FFF_FFFF
- 0xE020_0000 to 0xEFFF_FFFF
- 0xF020_0000 to 0xFFFF_FFFF

There can be zero or more such interfaces on the subsystem. If there are no such interfaces, all access to the preceding address range it targets responds with bus error. Additionally, if any implemented interface is not used or any of the preceding region is not implemented, a default subordinate must be used to respond with decode error. If there are more than one such interfaces, it is **IMPLEMENTATION DEFINED** how the preceding address range is divided across the interfaces. If implemented all these interfaces must export information to allow a debug access to be distinguished from an access by another manager in the system. We recommend that one such interface is provided in an implementation of the subsystem.

Subordinate Peripheral Expansion interface

This interface provides access to the Peripheral bus from expansion managers. This interface can be used to access all memory mapped regions in the system except to regions private to the processors. However, we recommend that this interface is not used to access areas outside the following memory mapped regions because of potential memory throughput limitations due to bus protocol and bus width conversion incurred at implementation:

- 0x4000_0000 to 0x4FFF_FFFF
- 0x5000_0000 to 0x5FFF_FFFF
- 0xE010_0000 to 0xEFFF_FFFF
- 0xF010_0000 to 0xFFFF_FFFF

We recommend that one such interface is provided in an implementation of the subsystem. If any implemented interface is not used, the interface must be tied to idle.

4.10 Interrupt interfaces

CRSAS Ma1 includes interrupt signals for use by the subsystem expansion. These connect to the interrupt controller of each CPU within the system and optionally to an External Wakeup Controller (EWIC) associated with the CPU or the Internal Wakeup Interrupt Controller (IWIC) of the CPU.

Signal name

CPU<n>EXPIRQ[CPU<n>EXPNUMIRQ-1:0]

Width

CPU<n>EXPNUMIRQ

Direction

Input

Description

These are Interrupt inputs from the subsystem expansion to the CPU<n> interrupt controller within the subsystem.

Each CPU in the subsystem implements a configurable number of external interrupt lines and of these, 32 are reserved for internal use and the rest are made available here.

CPU<n>EXPNUMIRQ defines the number of interrupts made available as expansion interrupts for CPU<n>.



Each bit **CPU<n>EXPIRQ[i]** is ultimately connected to IRQ[32+i] of the CPU<n>'s NVIC.

Signal name

CPU<n>EXPNMI

Width

1

Direction

Input

Description

This provides a non-maskable interrupt input from the subsystem expansion to the interrupt controller of CPU<n> within the subsystem.

This input is merged with other non-maskable interrupt sources within the subsystem before it is seen by the NVIC of the CPU core.

4.11 DMA interfaces

CRSAS Ma1 supports a DMA in the system and when DMA exists, namely NUMDMA>1, the following interfaces can exist depending on the configuration of the DMA:

- DMA Trigger Interfaces
- DMA General Purpose Output (GPO) Interfaces
- DMA Stream Interfaces

For more information, see *Arm® CoreLink™ DMA-350 Controller Technical Reference Manual*.

If any DMA interfaces exist, they all reside in the PD_SYS power domain, run on **SYSSYSCLK**, and they are on the **nWARMRESETSYS** reset domain.

4.12 CPU Coprocessor Interface

Each CPU core of the subsystem can be configured to have a coprocessor interface. If a CPU<n> coprocessor interface exists, then HASCPU<n>CPIF = 1.

These interfaces reside in their respective CPU's PD_CPU<n> power domain, **CPUCPU<n>CLK** clock domain and **nWARMRESETCPU<n>** reset domain. Note that because of the actual CPU implementation, the reset can be a special output that is dependent at least on **nWARMRESETCPU<n>**.

For more information on the coprocessor and related interfaces, see *Arm® Cortex®-M55 Processor Integration and Implementation Manual* or *Arm® Cortex®-M85 Processor Integration and Implementation Manual*.

4.13 TCM subordinate interface

A 64-bit subordinate TCM interface provides system access only to Tightly Coupled Memories (TCM) internal to each CPU. The protocol of this interface is **IMPLEMENTATION DEFINED**.

This expansion interface is typically used together with DMA controllers to transfer data to and from the processors for compute applications and each provides access to the TCM DMA interface. For example, for a Cortex M55 or Cortex-M85 core this TCM DMA interface is called the S-AHB DMA interface.

For more information on M55 S-AHB DMA interface, see *Arm® Cortex®-M55 Processor Integration and Implementation Manual*.

For more information on Cortex-M85 S-AHB DMA interface, see *Arm® Cortex®-M85 Processor Integration and Implementation Manual*.

This interface supports the following properties:

- 32-bit address
- 64-bit data
- Normal Memory access only
- Additional byte lane strobe signals to support non-contiguous write-data in a beat
- TrustZone Support enabled

The memory map presented on each of these interfaces and the underlying functionality for TCM access is defined by the *Arm® Cortex®-M55 Processor Technical Reference Manual* or *Arm® Cortex®-M85 Processor Technical Reference Manual*.

The memory map is shown in the following table. Instruction TCM (ITCM) accesses are mapped into a single address space, while each 64-bit Data TCM (DTCM) access is mapped to two 32-bit wide DTCMs.

Each of these interfaces resides either in the PD_SYS power domain or in their respective processor's PD_CPU<n> power domain. Depending on where they reside, the following conditions are true:

- If in the PD_CPU<n> power domain, the interface is on **CPUCPU<n>CLK** clock domain and **nWARMRESETCPU<n>** reset domain.
- If in the PD_SYS power domain, the interface is on **SYSSYSCLK** clock domain and **nWARMRESETSYS** reset domain.

Table 4-4: TCM DMA interface memory map

Start address	End address	xADDRS[3:2]	TCM accessed
0x0A00_0000	0x0A00_0000 + {ITCM size}	-	Non-secure CPU0 ITCM
0x0B00_0000	0x0B00_0000 + {ITCM size}	-	Non-secure CPU1 ITCM
0x0C00_0000	0x0C00_0000 + {ITCM size}	-	Non-secure CPU2 ITCM

Start address	End address	xADDRS[3:2]	TCM accessed
0x0D00_0000	0x0D00_0000 + {ITCM size}	-	Non-secure CPU3 ITCM
0x1A00_0000	0x1A00_0000 + {ITCM size}	-	Secure CPU0 ITCM
0x1B00_0000	0x1B00_0000 + {ITCM size}	-	Secure CPU1 ITCM
0x1C00_0000	0x1C00_0000 + {ITCM size}	-	Secure CPU2 ITCM
0x1D00_0000	0x1D00_0000 + {ITCM size}	-	Secure CPU3 ITCM
0x2400_0000	0x2400_0000 + {DTCM size}	2b00	Non-secure CPU0 D0TCM
0x2400_0000	0x2400_0000 + {DTCM size}	2b01	Non-secure CPU0 D1TCM
0x2400_0000	0x2400_0000 + {DTCM size}	2b10	Non-secure CPU0 D2TCM
0x2400_0000	0x2400_0000 + {DTCM size}	2b11	Non-secure CPU0 D3TCM
0x2500_0000	0x2500_0000 + {DTCM size}	2b00	Non-secure CPU1 D0TCM
0x2500_0000	0x2500_0000 + {DTCM size}	2b01	Non-secure CPU1 D1TCM
0x2500_0000	0x2500_0000 + {DTCM size}	2b10	Non-secure CPU1 D2TCM
0x2500_0000	0x2500_0000 + {DTCM size}	2b11	Non-secure CPU1 D3TCM
0x2600_0000	0x2600_0000 + {DTCM size}	2b00	Non-secure CPU2 D0TCM
0x2600_0000	0x2600_0000 + {DTCM size}	2b01	Non-secure CPU2 D1TCM
0x2600_0000	0x2600_0000 + {DTCM size}	2b10	Non-secure CPU2 D2TCM
0x2600_0000	0x2600_0000 + {DTCM size}	2b11	Non-secure CPU2 D3TCM
0x2700_0000	0x2700_0000 + {DTCM size}	2b00	Non-secure CPU3 D0TCM
0x2700_0000	0x2700_0000 + {DTCM size}	2b01	Non-secure CPU3 D1TCM
0x2700_0000	0x2700_0000 + {DTCM size}	2b10	Non-secure CPU3 D2TCM
0x2700_0000	0x2700_0000 + {DTCM size}	2b11	Non-secure CPU3 D3TCM
0x3400_0000	0x2400_0000 + {DTCM size}	2b00	Secure CPU0 D0TCM
0x3400_0000	0x3400_0000 + {DTCM size}	2b01	Secure CPU0 D1TCM
0x3400_0000	0x3400_0000 + {DTCM size}	2b10	Secure CPU0 D2TCM
0x3400_0000	0x3400_0000 + {DTCM size}	2b11	Secure CPU0 D3TCM
0x3500_0000	0x3500_0000 + {DTCM size}	2b00	Secure CPU1 D0TCM
0x3500_0000	0x3500_0000 + {DTCM size}	2b01	Secure CPU1 D1TCM
0x3500_0000	0x3500_0000 + {DTCM size}	2b10	Secure CPU1 D2TCM
0x3500_0000	0x3500_0000 + {DTCM size}	2b11	Secure CPU1 D3TCM
0x3600_0000	0x3600_0000 + {DTCM size}	2b00	Secure CPU2 D0TCM
0x3600_0000	0x3600_0000 + {DTCM size}	2b01	Secure CPU2 D1TCM
0x3600_0000	0x3600_0000 + {DTCM size}	2b10	Secure CPU2 D2TCM
0x3600_0000	0x3600_0000 + {DTCM size}	2b11	Secure CPU2 D3TCM
0x3700_0000	0x3700_0000 + {DTCM size}	2b00	Secure CPU3 D0TCM
0x3700_0000	0x3700_0000 + {DTCM size}	2b01	Secure CPU3 D1TCM
0x3700_0000	0x2700_0000 + {DTCM size}	2b10	Secure CPU3 D2TCM
0x3700_0000	0x2700_0000 + {DTCM size}	2b11	Secure CPU3 D3TCM



Note

These addresses in the previous table are specific only for the TCM DMA subordinate interface. These TCMs reside at address offsets in the main memory map and at private address offsets to each processor's local TCM. For more information, see [CPU TCM memories](#). To make these TCMs visible to other managers outside of the subsystem, this interface must be mapped to expansion regions of the memory map defined in the preceding table. This is **IMPLEMENTATION DEFINED**.

Global exclusive access monitors are not supported on TCM Subordinate interface. Ensure that no coherency issues arise when you are using this interface to access the TCMs. It is valid for processors in the system to cache TCM instruction and data values. Therefore, care must be taken to ensure consistency of the TCM and Cache data.

4.14 Debug and Trace Related interfaces

This section describes the debug and trace related interfaces of CRSAS Ma1.

4.14.1 Debug Access interface

When `DEBUGLEVEL > 0`, CRSAS Ma1 provides interfaces for debug access from an external Debug Access Port or an external debug infrastructure. Depending on the HASCSS configuration, the interface or interfaces provide the following:

- When `HASCSS = 0`, where there can only be one CPU, the CPU Debug D-AHB access is provided as an expansion interface. This allows the SoC integrator to drive the interface using a suitable CoreSight MEM-AP and provides debug access to the processor. For more information on the processor's D-AHB interface, see *Arm® Cortex®-M55 Processor Technical Reference Manual* and *Arm® Cortex®-M85 Processor Technical Reference Manual*.

The interface is in `PD_CPU0` power domain and resides in the **CPUCPU0CLK** clock domain and **nCOLDRESETCPU0** reset domain for Cortex-M55 or **nCOLDRESETDEBUGCPU0** and **DEBUGCPU0CLK** for Cortex-M85.

- When `HASCSS = 1`, a single Debug Access Interface is provided as an expansion subordinate interface. This interface is an APB4 subordinate interface and provides an additional **DPABORT** input signal to allow an external DAP to signal a transaction abort. This interface resides in the **DEBUGDEBUGCLK** clock domain and **nCOLDRESETDEBUG** reset domain.

When `DEBUGLEVEL = 0`, these interfaces might not exist. If they do exist, they are tied or unused.

4.14.2 Debug Timestamp interface

When **DEBUGLEVEL** = 2, CRSAS Ma1 provides one or more 64-bit timestamp input or inputs. This timestamp is expected to be driven by a timestamp generator in the subsystem expansion. Depending on **HASCSS** configuration, the interface or interfaces are as follows:

- When **HASCSS** = 0 where there can only be one CPU, the processor's debug global timestamp input, **TSVALUEB[63:0]**, is provided as an expansion interface, **CPU0TSVALUEB[63:0]**. This is expected to be driven by a global timestamp generator. For more information on these interfaces, see *Arm® Cortex®-M55 Processor Technical Reference Manual* or *Arm® Cortex®-M85 Processor Technical Reference Manual*.

This interface resides in the PD_DEBUG power domain and resides in the **DEBUGCPU0CLK** clock domain and **nCOLDRESETDEBUGCPU0** reset domain.

- When **HASCSS** = 1, a single **TSVALUE<B/G>** is provided and is used to generate all debug timestamps input to all processors within the system. It resides in the PD_DEBUG power domain. The Timestamp input value **TSVALUE<B/G>** width is **IMPLEMENTATION DEFINED**, and it can either be:
 - Gray coded. In this case, the signal name ends with 'G' and is asynchronous.
 - Binary coded. In this case, the signal name ends with 'B' and is synchronous to **DEBUGDEBUGCLK** and is on the **nCOLDRESETDEBUG** reset domain.

For both configurations of **HASCSS**, each processor's **TSCLKCHANGE** input is provided as an expansion interface as **CPU<n>TSCLKCHANGE** to allow the CPUs to be notified of a change in timestamp clock ratio. Each interface that is associated with CPU<n> resides in their respective **DEBUGCPU<n>CLK** clock domain and **nCOLDRESETDEBUGCPU<n>** reset domain.

When **DEBUGLEVEL** < 2, this timestamp interface might not exist. If they do exist, they are tied or unused.

4.14.3 Cross Trigger Channel interface

When **DEBUGLEVEL** > 0, CRSAS Ma1 includes one or more sets of Cross Trigger Channel inputs and Cross Trigger Channel outputs to allow partners to expand the cross trigger infrastructure as follows.

When **HASCSS** = 1, CRSAS Ma1 includes a shared Cross Trigger Matrix (CTM) and provides a single Cross Trigger Channel input and a single Cross Trigger Channel output to allow partners to expand the cross trigger infrastructure. See the table below.

For more information of the CTM, see *Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual*. This interface is synchronous to **DEBUGDEBUGCLK**, resides in the PD_DEBUG power domain and **nCOLDRESETDEBUG** reset domain.

Table 4-5: Cross Trigger Channel interface

Signal name	Width	Direction	Description
CTMCHANNELIN[3:0]	4	Input	Channel in port

Signal name	Width	Direction	Description
CTMCHANNELOUT[3:0]	4	Output	Channel out port

When HASCSS = 0 where there can only be one CPU, the Cross Trigger Channel interfaces of the processor are provided as expansion interfaces, **CPU0CTICHIN[3:0]** and **CPU0CTICHOUT[3:0]**. For more information of these interfaces, see *Arm® Cortex®-M55 Processor Technical Reference Manual* and *Arm® Cortex®-M85 Processor Technical Reference Manual*.

- For Cortex-M55, these interfaces are synchronous to **CPUCPU0CLK**, and reside in the PD_CPU0 power domain and in the **nCOLDRESETCPU0** reset domain.
- For Cortex-M85, these interfaces are synchronous to **DEBUGCPU0CLK**, and reside in the PD_DEBUG power domain and in the **nCOLDRESETDEBUGCPU0** reset domain.

One or more of these interfaces must exist when DEBUGLEVEL > 0. Otherwise, this interface might not exist. If it does exist, it is tied or unused.

4.14.4 Cross Trigger Interface

When DEBUGLEVEL > 0 and HASCSS > 0, CRSAS Ma1 includes a shared Cross Trigger Interface (CTI) module. Some trigger signals to and from the CTI are used within the system, while a number is made available to system expansion. The following table lists the trigger signals that are available for system expansion.

For more information on the CTI, see *Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual*.

Table 4-6: CTI triggers

Signal name	Width	Direction	Description
CTIEVENTIN[7:4]	4	Input	CTI Trigger inputs
CTIEVENTOUT[7:6]	2	Output	CTI Trigger outputs

This interface is synchronous to **DEBUGDEBUGCLK**, resides in the PD_DEBUG power domain and **nCOLDRESETDEBUG** reset domain.

This interface must exist when DEBUGLEVEL > 0 and HASCSS = 1. Otherwise, this interface might not exist and if it does, it is tied or unused.

4.14.5 Debug APB Expansion interface

When DEBUGLEVEL > 0 and HASCSS = 1, CRSAS Ma1 provides a Debug APB expansion interface so that partners can add more debug functionality to the Debug System. This interface is only accessible through:

- The debug interface using an external DAP when **SECDBGSTAT.SYSDSSACCEN<n>_STATUS** = 1.
- The system interconnect for CPU<n> when **SECDBGSTAT.SYSDSSACCEN<n>_STATUS** = 1, or for **IMPLEMENTATION DEFINED** manager on expansion interfaces when **SECDBGSTAT.SYSDSSACCENX_STATUS** = 1.

For more information of the address mapping of this interface, see [HASCSS = 1](#).

This interface is synchronous to **DEBUGDEBUGCLK**, is in the **nCOLDRESETDEBUG** reset domain and resides in the PD_DEBUG power domain.

Table 4-7: Debug APB Expansion interface

Signal name	Width	Direction	Description
DEBUGPRDATA[31:0]	32	Input	APB read data. Drives this bus during read cycles
DEBUGPREADY	1	Input	APB ready. Uses this signal to extend an APB transfer.
DEBUGPSLVERR	1	Input	Indicates a transfer failure. The APB peripherals are not required to support the PSLVERR pin.
DEBUGPADDR[31:2]	30	Output	The APB address bus for manager interface
DEBUGPSEL	1	Output	APB select. Indicates that the subordinate device is selected, and a data transfer is required.
DEBUGPENABLE	1	Output	APB enable. Indicates the second and subsequent cycles of an APB transfer.
DEBUGPWRITE	1	Output	APB RW transfer. Indicates an APB write access when HIGH, and an APB read access when LOW.
DEBUGPWDATA[31:0]	32	Output	Write data.

When **DEBUGLEVEL = 0** or **HASCSS = 0**, this interface might not exist. If it exists, it is tied or unused.

4.14.6 CPU<n> External Peripheral interface EPPB

Each CPU<n> in the system provides an interface that allows users to add peripherals to the External PPB region that are private to each processor.

This is either one 32-bit AMBA 4 APB interface for a Cortex-M55, or two 32-bit AMBA 4 APB interfaces for a Cortex-M85. These are typically used for integration with additional CoreSight debug and trace components if necessary. Only data accesses are allowed on each of these interfaces privately from each processor at address **0xE0004_0000** to **0xE00F_FFFF**. Some of these regions are already used by peripherals like EWIC or reserved and others are available for integration of additional components.

Cortex-M55

If ;n>TYPE is Cortex-M55, each interface that is associated to CPU<n> resides in the PD_CPU<n> power domain, the **CPUCPU<n>CLK** clock domain and the **nCOLDRESETCPU<n>** reset domain.

Cortex-M85

If ;n>TYPE is Cortex-M85, each interface that is associated with CPU<n> core PPB resides in the PD_CPU<n> power domain, the **CPUCPU<n>CLK** clock domain and the **nCOLDRESETCPU<n>** reset domain. Each interface that is associated with CPU<n> Debug PPB resides in the PD_DEBUG power domain, the **DEBUGCPU<n>CLK** clock domain and the **nCOLDRESETDEBUGCPU<n>** reset domain.

For more information and for the address mapping, see [CPU Private Peripheral Bus region](#), Arm® Cortex®-M55 Processor Technical Reference Manual or Arm® Cortex®-M85 Processor Technical Reference Manual.

4.14.7 ATB Trace interfaces

When **DEBUGLEVEL** = 2, CRSAS Ma1 provides interfaces to output trace data to an expansion Trace Port Interface Unit (TPIU). The number and types of interfaces vary depending on the HASCSS configuration as follows:

- When **HASCSS** = 0 and there can only be one CPU, the processor provides its ITM ATB Trace interface and an ETM ATB Trace interface directly for expansion. The processor's Trace synchronization and Trigger Interface are also provided for expansion. For more information on these interfaces, see *Arm® Cortex®-M55 Processor Technical Reference Manual* or the relevant sections of *Arm® Cortex®-M85 Processor Technical Reference Manual*.

All interfaces reside in the PD_DEBUG power domain and reside in the **DEBUGCPU0CLK** clock domain and **nCOLDRESETDEBUGCPU0** reset domain.

- When **HASCSS** = 1, CRSAS Ma1 provides a single ATB bus interface that is normally expected to connect to an external TPIU. This trace bus is synchronous to **DEBUGDEBUGCLK**, is reset using **nCOLDRESETDEBUG** and resides in the PD_DEBUG power domain.

Table 4-8: ATB Trace interface

Signal name	Width	Direction	Description
ATVALID	1	Output	A transfer is valid during this cycle. If LOW, all the other ATB signals must be ignored in this cycle.
ATID[6:0]	7	Output	An ID that uniquely identifies the source of the trace.
ATBYTE	IMPLEMENTATION DEFINED	Output	The number of bytes on ATDATA to be captured, minus 1. Note that as ATDATA is only eight bits wide, this signal may not exist.
ATDATA	IMPLEMENTATION DEFINED	Output	Trace data bus.
ATREADY	1	Input	Subordinate is ready to accept data.
AFVALID	1	Input	This is the flush signal. All buffers must be flushed because trace capture is about to stop.
AFREADY	1	Output	This is a flush acknowledge. Asserted when buffers are flushed.
SYNCREQ	1	Input	Trace synchronization request trace sinks.

When **DEBUGLEVEL** < 2 these interfaces might not exist. If they do exist, they are tied or unused.

4.14.8 Debug Authentication interface

The debug authentication signals of the subsystem reside in the PD_MGMT power domain, when **PILEVEL** = 2 its states are saved and restored when entering and then leaving the **HIBERNATION1** lower power state, respectively. The state retention is **IMPLEMENTATION DEFINED** and may be performed using shadow registers in PD_AON. Throughout **HIBERNATION1** low power state these signals must be driven to PD_AON if used in PD_AON and remain static.

The input signals in the following table define the debug authentication signal values, when they are not overridden by the internal Secure debug configuration registers. The merged debug authentication signals are then made available as outputs to the rest of the system. These are

defined as the output signals in the following table. For more information on how the output is generated, see [Secure Debug Configuration Registers](#).

Table 4-9: Debug authentication interface

Signal name	Width	Direction	Description
DBGENIN	1	Input	Debug Enable Input
NIDENIN	1	Input	Non-Invasive Debug Enable Input
SPIDENIN	1	Input	Secure Privileged Invasive Debug Enable Input
SPNIDENIN	1	Input	Secure Privileged Non-Invasive Debug Enable Input
DAPACCENIN	1	Input	External Debug Access Enable Input.
DAPDSSACCENIN	1	Input	Debug Access Port to Debug System Access Enable Input.
SYSDSSACCEN<n>IN	1	Input	CPU to Debug System through System Interconnect Access Enable Input. This signal must exist when HASCSS = 1. There is one bit per CPU<n> to indicate which CPU is allowed access.
SYSDSSACCENXIN	1	Input	Implementation Defined Manager to Debug System through System Interconnect Access Enable Input. This signal must exist when HASCSS = 1. The Manager selected for access control is IMPLEMENTATION DEFINED
DBGEN	1	Output	Merged Debug Enable Output
NIDEN	1	Output	Merged Non-Invasive Debug Enable Output
SPIDEN	1	Output	Merged Secure Privileged Invasive Debug Enable Output
SPNIDEN	1	Output	Merged Secure Privileged Non-Invasive Debug Enable Output
DAPACCEN	1	Output	Merged External Debug Access Enable Output
DAPDSSACCEN	1	Output	Merged Debug Access Port to Debug System Access Enable Output.
SYSDSSACCENX	1	Output	Merged Defined Manager to Debug System through System Interconnect Access Enable Output. This signal must exist when HASCSS = 1. The Manager selected for access control is IMPLEMENTATION DEFINED .

4.15 CryptoCell Related Expansion interfaces

The following section details interfaces that must exist when HASCRYPTO = 1.

4.15.1 CryptoCell Lifecycle Indication interface

The CryptoCell Lifecycle Indication interface indicates the lifecycle state of the system through different stages of manufacture and deployment of the final product.

All signals are synchronous to **CRYPTOSYSCLK** and are on the Always ON power domain.

The following table shows the output signals on the CryptoCell Lifecycle Indication interface.

Table 4-10: CryptoCell Lifecycle Indication interface

Signal name	Width	Direction	Description
CRYPTOLCS[2:0]	3	Output	Lifecycle State values: <ul style="list-style-type: none"> 3'b000 Chip Manufacture (CM) 3'b001 Device Manufacture (DM) 3'b101 Secure Enable (SE) 3'b111 Return to Manufacturer (RMA)
CRYPTOLCSVALID	1	Output	Lifecycle State values on CRYPTOLCS is valid, where: <ul style="list-style-type: none"> '1' CRYPTOLCS[2:0] is valid. '0' CRYPTOLCS[2:0] is not valid.

4.15.2 CryptoCell Debug Control Enable interface

The CryptoCell Debug Control Enable interface provides signals that can be used to control the debug authentication signals at the top level of CRSAS Ma1 and other tests, debug and security-related functionality in SoC.

All signals are synchronous to **CRYPTOSYSCLK** and are on the Always ON power domain.

The following table shows the output signals on the CryptoCell Debug Control Enable Interface.

Table 4-11: CryptoCell debug control enable interface

Signal name	Width	Direction	Description
CRYPTODCUEN[127:1]	127	Output	Debug Control Enable Values

4.15.3 CryptoCell Non-Volatile Memory interface

The CryptoCell Non-Volatile Memory interface is the CryptoCell interface to the *One-Time Programmable* memory (OTP).

All signals are synchronous to **CRYPTOSYSCLK** and are on the **nWARMRESETCRYPTO** reset domain.

Table 4-12: CryptoCell Non-Volatile Memory interface

Signal name	Width	Direction	Description
CRYPTONVMPADDR	13	Output	APB Address
CRYPTONVMPENABLE	1	Output	APB Enable
CRYPTONVMPSEL	1	Output	APB Select
CRYPTONVMPSTRB	4	Output	APB Byte Lane Strobe
CRYPTONVMPWDATA	32	Output	APB Write Data
CRYPTONVMPWRITE	1	Output	APB Write Enable
CRYPTONVMPRDATA	32	Input	APB Read Data
CRYPTONVMPREADY	1	Input	APB Response Ready

Signal name	Width	Direction	Description
CRYPTONVMPSLVERR	1	Input	APB Subordinate Error
CRYPTONVMPROGCOMPLETED	1	Input	Program Completed status. This signal indicates that any programming of the OTP-NVM memory is completed.

For more information on this interface, see *Arm® CryptoCell™-312 Technical Reference Manual*.

4.16 Security Control Expansion signals

CRSAS Ma1 provides additional status and control signals to handle additional Manager Security Controllers (MSC), Memory Protection Controllers (MPC), Peripheral Protection Controllers (PPC) and Bridges with write buffers in the expansion system. These signals allow all the components to be controlled using the same set of security control registers already implemented within the subsystem.

All signals in this section are synchronous to **SYSSYSCLK**, and the SYSSYSCLK Q-Channel Device Interface is needed to control the availability of **SYSSYSCLK**. These signals reside in the PD_SYS power domain and in the **nWARMRESETSYS** reset domain.



Note

While CRSAS Ma1 defines a full set of signals in this document, many of these interfaces and some of their individual bits can be unimplemented or disabled. These are defined as configuration options in [Configuration options](#). How each configuration option is implemented is **IMPLEMENTATION DEFINED**.

4.16.1 Memory Protection Controller Expansion

CRSAS Ma1 supports up to 16 MPCs to be added to the expansion system.

The **SMPCEXPSTATUS** signal allows the interrupts of the MPCs to be internally merged to the single MPC Combined interrupt.

Table 4-13: MPC Expansion Interrupt Status input

Signal name	Width	Direction	Description
SMPCEXPSTATUS	16	Input	<p>Interrupt Status inputs from all Expansion Memory Protection Controller. These are programmed through the SECMPCCINTSTAT.SMPCEXP_STATUS register fields in the Secure Access Configuration Register Block and are used to raise an interrupt using the MPC Combined Interrupt.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>

4.16.2 Peripheral Interconnect Peripheral Protection Controller Expansion

CRSAS Ma1 supports up to four additional PPCs to be added to the Peripheral Interconnect in the expansion system. The following signals are provided to control the PPC <i> where i is {0-3}.

Table 4-14: Peripheral Interconnect PPC Expansion interface

Signal Name	Width	Direction	Description
SPERIPHPPCEXPSTATUS	4	Input	<p>Peripheral Interconnect PPC Interrupt Status Input. Each bit SPERIPHPPCEXPSTATUS[i] is to be connected to a single PPC <i>.</p> <p>The SPERIPHPPCEXPSTATUS[i] bit is associated to the SECPPCINTSTAT.SPERIPHPPCEXP_STATUS[i] register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
SPERIPHPPCEXPCLR	4	Output	<p>Peripheral Interconnect PPC Interrupt Clear Output. Each bit SPERIPHPPCEXPCLR[i] is to be connected to a single PPC <i>.</p> <p>The SPERIPHPPCEXPCLR[i] bit is associated to the SECPPCINTCLR.SPERIPHPPCEXP_CLR[i] register field.</p> <p>Individual bits of this interface can be unimplemented or disabled which results in the associated register bit field being RAZ/WI.</p>
PERIPHNSPPCEXP<i>	16	Output	<p>Peripheral Interconnect PPC Non-secure Gating Control. These are a set of multiple bit interfaces, and each interface connects to PPC <i>. When each bit 'j' of an interface is HIGH, it defines a specific <j> interface that the target PPC controls as Non-secure access only.</p> <p>Each bit PERIPHNSPPCEXP<i>[j] is driven by the PERIPHNSPPCEXP<i>[j] register.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
PERIPHPPPEXP<i>	16	Output	<p>Peripheral Interconnect PPC Privilege Gating Control. These are a set of multiple bit interfaces and each interface connects to PPC <i>. When each bit PERIPHPPPEXP<i>[j] of an interface is HIGH, it defines the <j> interface that the target PPC <i> controls as both privileged and unprivileged access. Else, it is privileged access only.</p> <p>Each bit PERIPHPPPEXP<i>[j] is selected from either PERIPHPPPEXP<i>[j] if PERIPHNSPPCEXP<i>[j] is '0' or PERIPHNSPPPEXP<i>[j] otherwise.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit fields that contributes to this control signal being RAZ/WI.</p>

4.16.3 Main Interconnect Peripheral Protection Controller Expansion

CRSAS Ma1 supports up to four additional PPCs to be added to the Main Interconnect in the expansion system. The following signals are provided to control each PPC<i> where i is {0-3}.

Table 4-15: Main Interconnect PPC Expansion interface

Signal name	Width	Direction	Description
SMAINPPCEXPSTATUS	4	Input	<p>Main Interconnect PPC Interrupt Status Input. Each bit SMAINPPCEXPSTATUS[i] is to be connected to a single PPC<i></p> <p>The SMAINPPCEXPSTATUS[i] bit is associated to the SECPPCINTSTAT.SMAINPPCEXP_STATUS[i] register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
SMAINPPCEXPCLR	4	Output	<p>Main Interconnect PPC Interrupt Clear Output. Each bit SMAINPPCEXPCLR[i] is to be connected to a single PPC<i>.</p> <p>The SMAINPPCEXPCLR[i] bit is associated to the SECPPCINTCLR.SMAINPPCEXP_CLR[i] register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
MAINNSPPCEXP0<i>	16	Output	<p>Main Interconnect PPC Non-secure Gating Control. These are a set of multiple bit interfaces, and each interface connects to PPC<i>. When each bit MAINNSPPCEXP<i>[j] of an interface is HIGH, it defines the <j> interface that the target PPC<i> controls as Non-secure access only.</p> <p>Each bit MAINNSPPCEXP<i>[j] is driven by the MAINNSPPCEXP<i>[j] register.</p> <p>Individual bits of this interface can be unimplemented or disabled which results in the associated register bit field being RAZ/WI.</p>
MAINPPPEXP<i>	16	Output	<p>Main Interconnect PPC Privilege Gating Control. These are a set of multiple interfaces and each interface connects to PPC<i>. When each bit MAINPPPEXP<i>[j] of an interface is HIGH it defines the <j> interface that the target PPC<i> controls as both privileged and unprivileged access. Else, it is privileged access only.</p> <p>Each bit MAINPPPEXP<i>[j] is selected from either MAINPPPEXP<i>[j] if MAINNSPPCEXP<i>[j] is '0' or MAINNSPPPEXP<i>[j] otherwise.</p> <p>Individual bits of this interface can be unimplemented or disabled which results in the associated register bit fields that contributes to this control signal being RAZ/WI.</p>

4.16.4 Manager Security Controller Expansion

CRSAS Ma1 supports up to 16 additional Manager Security Controllers (MSC) to be added to the expansion system. The following signals are provided to control each MSC<i> where i is {0-15}.

Table 4-16: MSC Expansion interface

Signal name	Width	Direction	Description
SMSCEXPSTATUS	16	Input	<p>MSC Interrupt Status Input. Each bit SMSCEXPSTATUS[i] is to be connected to a single MSC<i>.</p> <p>These are associated with the SECMSCINTSTAT.SMSCEXP_STATUS register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
SMSCEXPCLR	16	Output	<p>MSC Interrupt Clear Output. Each bit SMSCEXPCLR[i] is to be connected to a single MSC<i>.</p> <p>These are associated with the SECMSCINTCLR.SMSCEXP_CLR register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
NSMSCEXP	16	Output	<p>MSC Non-secure Configuration. Each bit NSMSCEXP[i] is to be connected to a single MSC<i>. Set HIGH to configure a manager as Non-secure.</p> <p>These are associated with the NSMSCEXP.NS_MSCEXP register field.</p> <p>Individual bits of this interface can be unimplemented or disabled. Any disabled bit of this interface that still exist must be tied HIGH.</p>

4.16.5 Bridge Buffer Error Expansion

CRSAS Ma1 supports up to 16 additional bridges with buffer error signalling to be added to the expansion system.

Table 4-17: Bridge Error Interrupt Expansion interface

Signal name	Width	Direction	Description
BRGEXPSTATUS	16	Input	<p>Bridge Error Interrupt Status Input. Each bit BRGEXPSTATUS[i] is to be connected to a single bridge <i> where i is 0 to 15.</p> <p>These are associated with the BRGINTSTAT.BRGEXP_STATUS register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>
BRGEXPCLR	16	Output	<p>Bridge Error Interrupt Clear Output. Each bit BRGEXPCLR[i] is to be connected to a single bridge <i> where i is 0 to 15.</p> <p>These are associated with the BRGINTCLR.BRGEXP_CLR register field.</p> <p>Individual bits of this interface can be unimplemented or disabled, which results in the associated register bit field being RAZ/WI.</p>

4.16.6 Other Security Expansion signals

The following table lists other signals that are related to Security that is needed by PPCs and MSCs in the expansion system.

Table 4-18: Other Security Expansion signals

Signal name	Width	Direction	Description
SECRESPCFG	1	Output	<p>This output configures how to respond to an access when a security violation occurs.</p> <ul style="list-style-type: none"> 0: Read-Zero Write Ignore 1: Bus error <p>This output is controlled by the SECRESPCFG register.</p>
ACCWAITn	1	Output	<p>This request output is used to control any external gating unit that may be required to block accesses to the system through the Main and Peripheral Interconnect Expansion interfaces.</p> <ul style="list-style-type: none"> 1: No gating 0: Access gated <p>This output is controlled by the BUSWAIT register.</p>
ACCWAITNSTATUS	1	Input	<p>This status output is used to indicate the current state of any external gating unit that may be used to block access to the system through the Main and Peripheral Interconnect Expansion interfaces</p> <ul style="list-style-type: none"> 1: No gating 0: Access gated <p>This input can be read using the BUSWAIT register.</p>

4.17 Clock configuration interface

CRSAS Ma1 provides a set of control and status signals for some of the clocks to configure generators or dividers that might exist in the expansion system.

Each set of control and status signal and even individual bits of this interface can be unimplemented or disabled. This, and how they are used are IMPLEMENTATION DEFINED, but if a set exists, then the default value of each signal at reset must be set to a value to allow the input clock to run at a default clock rate to allow the system to at least boot without software configuring these sets of registers. The reset value can be configured.

All signals in this interface reside in the PD_AON power domain, are synchronous to **AONCLK** and are on the **nCOLDRESETAON** reset domain.

Table 4-19: Clock configuration interface signals

Signal name	Width	Direction	Description
CPU<n>CLKCFG	4	Output	<p>These control outputs provide a set of four-bit outputs that allows the system to configure an external clock generation logic that drives CPU<n>CLK. These output signals are driven by the register fields CLK_CFG0.CPU<n>CLKCFG.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
CPU<n>CLKCFGSTATUS	4	Input	<p>These sets of four-bit input signals are used to read the status of any external clock generation logic that drives CPU<n>CLK. The values on this interface can be read by the register fields CLK_CFG0.CPU<n>CLKCFGSTATUS.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
SYSCLKCFG	4	Output	<p>This control output provides four-bit outputs that allows the system to configure an external clock generation logic that drives SYSCLK. This output is driven by the register fields CLK_CFG1.SYSCLKCFG.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
SYSCLKCFGSTATUS	4	Input	<p>This four-bit input signal is used to read the status of any external clock generation logic that drives SYSCLK. The values on this interface can be read by the register fields CLK_CFG1.SYSCLKCFGSTATUS.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
AONCLKCFG	4	Output	<p>This control output provides a four-bit output that allows the system to configure an external clock generation logic that drives AONCLK. This output is driven by the register fields CLK_CFG1.AONCLKCFG.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
AONCLKCFGSTATUS	4	Input	<p>This four-bit status output is used to read the status of any external clock generation logic that drives AONCLK. The values on this interface can be read by the register fields CLK_CFG1.AONCLKCFGSTATUS.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
NPU<m>CLKCFG	4	Output	<p>These control outputs provide a set of four-bit outputs that allows the system to configure an external clock generation logic that drives NPU<m>CLK. These outputs are driven by the register fields CLK_CFG2.NPU<m>CLKCFG.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>
NPU<m>CLKCFGSTATUS	4	Input	<p>These sets of four-bit status inputs are used to read the status of any external clock generation logic that drives NPU<m>CLK. The values on this interface can be read by the register fields CLK_CFG2.NPU<m>CLKCFGSTATUS.</p> <p>Any unimplemented bits result in the associated register bit field being RAZ/WI.</p>

4.18 Miscellaneous signals

The following are other signals available for CRSAS Ma1.

Table 4-20: Other miscellaneous top-level signals

Signal name	Width	Direction	Sync to	Power domain	Description
LOCKNSVTOR<n>	1	Input	CPU<n>CLK	PD_CPU<n>	<p>Disables writes to the CPU<n> VTOR_NS register. For more information on this register, see <i>Arm®v8-M Architecture Reference Manual</i>.</p> <p>When HIGH, this input prevents changes to the Non-secure vector table base address register of CPU<n>.</p> <p>If not used, tie to LOW, tying this signal HIGH causes loss of vector table control.</p>
LOCKNSMPU<n>	1	Input	CPU<n>CLK	PD_CPU<n>	<p>This input disables writes to CPU<n> registers that are associated with the Secure Memory Protection Unit (MPU) region from software or from a debug agent connected to the processor.</p> <ul style="list-style-type: none"> MPU_CTRL MPU_RNR MPU_RBAR MPU_RLAR MPU_RBAR_An MPU_RLAR_An <p>For more information on these registers, see <i>Arm®v8-M Architecture Reference Manual</i>.</p> <p>When HIGH, this input prevents changes to the memory regions which have been programmed in the secure MPU. All writes to these registers are ignored.</p> <p>If not used, tie to LOW, tying this signal HIGH causes loss of Secure Memory Protection Unit (MPU) control.</p>
CPU<n>WAITCLR	1	Input	SYSCLK	PD_MGMT	<p>When HIGH, clears the register fields CPUWAIT.CPU<n>WAIT. This allows an external entity to release a processor that is already waited by CPUWAIT to start execution. Once set to '1', this signal must be held at '1' until CPU<n>WAITCLRRESP is '1'.</p> <p>Note while this is clocked using SYSCLK this input can optionally be implemented as an asynchronous input.</p>
CPU<n>WAITCLRRESP	1	Output	SYSCLK	PD_MGMT	<p>This signal provides a response to the CPU<n>WAITCLR request. When CPU<n>WAITCLR is '1' and CPUWAIT.CPU<n>WAIT is '0', this signal is set to '1' until CPU<n>WAITCLR request goes '0'.</p>

Signal name	Width	Direction	Sync to	Power domain	Description
NSWDRSTREQSTATUS	1	Output	SYSCLK	PD_AON	Non-secure watchdog reset request status. This output is '1' when the Non-secure Watchdog is raising a reset request and RESET_MASK.NSWDRSTREQEN is '1'. Once set to HIGH, it does not return to low unless a reset occurs clearing this status. This output is optional, but must exist when COLDRESET_MODE = 1.
SWDRSTREQSTATUS	1	Output	SYSCLK	PD_AON	Secure watchdog reset request status. This output is '1' when the Secure Watchdog is raising a reset request. Once set to HIGH, it does not return to low unless a reset occurs clearing this status. This output is optional, but must exist when COLDRESET_MODE = 1.
SSWDRSTREQSTATUS	1	Output	SYSCLK	PD_AON	Secure Privileged SLOWCLK watchdog reset request status. This output is '1' when the SLOWCLK Watchdog is raising a reset request. Once set to HIGH, it does not return to low unless a reset occurs clearing this status. This output is optional, but must exist when COLDRESET_MODE = 1.
RESETREQSTATUS	1	Output	SYSCLK	PD_AON	Hardware Reset Request status. This output is set to '1' if RESETREQ input is '1'. Once set to HIGH, it must not be cleared unless the system is reset. This output is optional, but must exist when COLDRESET_MODE = 1.
SWRSTREQSTATUS	1	Output	SYSCLK	PD_AON	Software Reset Request Status. This output is '1' when SWRESET.SWRESETREQ is set to '1'. Once set to HIGH, it must not be cleared unless the system is reset while restores the register field to '0'. This output is optional, but must exist when COLDRESET_MODE = 1.
CPU<n>LOCKUP	1	Output	AONCLK	PD_AON	Processor Lockup Status. There is one bit per Processor. Each bit indicates if the associated CPU<n> has lockup. This signal is an output directly from CPU<n>.
CPU<n>HALTED	1	Output	CPU<n>CLK	PD_CPU<n>	Processor Halted Status. There is one bit per Processor. Each bit indicates if the associated CPU<n> has halted. This signal is an output directly from CPU<n>.
CPU<n>EDBGRQ	1	Input	CPU<n>CLK	PD_AON, PD_CPU<n>	External request for CPU<n> to enter halt mode. This signal is an input directly to CPU<n> and also to EWIC<n>.
CPU<n>DBGRESTART	1	Input	CPU<n>CLK	PD_CPU<n>	Request for CPU<n> to perform synchronized exit from halt mode. Forms a handshake with CPU<n>DBGRESTARTED . This signal is an input directly to CPU<n>.
CPU<n>DBGRESTARTED	1	Output	CPU<n>CLK	PD_CPU<n>	Acknowledges CPU<n>DBGRESTART . This signal is an output directly from CPU<n>.

5 Functional Description

The following sections provides more details on the functionality of the system.

5.1 Clocking infrastructure

CRSAS Ma1 provides several input clocks into the system. These are as follows:

SLOWCLK

An always running clock that is expected also to be the first available when the system first powers up.

This clock is also required to run while others can be turned off when the system is in its lowest power state other than OFF.

This clock minimally drives the following logic that resides in the PD_AON power domain:

- A 32-bit Timer. This timer running at **SLOWCLK** allows the user to setup a wake event.
- A 32-bit Watchdog Timer. This watchdog timer provides protection against an unresponsive system, particularly during the lowest power state.

AONCLK

This clock drives all other logic that resides in the PD_AON domain.

This does not include logic in the PD_MGMT power domain that is merged into PD_AON when $PILEVEL < 2$. The Q-Channel Control interface, AONCLK Q-Channel Control Interface, allows the subsystem to request and handshake the availability of the **AONCLK**.

This clock drives the following:

- External Wakeup Interrupt Controller<n> (EWIC<n>) that allows a processor to be woken through an interrupt.
- The PD_MGMT PPU.
- All other logic in the PD_AON domain that is not running on **SLOWCLK** or **SYSCLK** when $PILEVEL < 2$.

SYSCLK

This is the main system clock that drives most of the system that resides in the PD_SYS power domain.

This clock also drives all logic that resides in the PD_MGMT domain. The Q-Channel Control interface, SYSCLK Q-Channel Control Interface, allows the subsystem to request and handshake the availability of the **SYSCLK**.

This clock is requested to run in any of the following cases:

- System is not in HIBERNATION{0-1} or in SYS_RET power state.

- It is forced to run through the CLOCK_FORCE register.
- Activity based clock request from the logic associated with this clock.

This clock drives the following:

- The Main Interconnect and Peripheral Interconnect and other related functionality like expansion interfaces.
- System and Security Control related registers and logic, except those that resides in PD_AON
- Power Control logic that resides in PD_MGMT power domain that controls the PD_SYS, PD_CPU<n>, PD_DEBUG, PD_NPU<m> and PD_CRYPTO power domains.
- All Volatile Memory interfaces and peripherals in the PD_SYS domain, along with the interfaces to Timers and Watchdog timers.

CPU<n>CLK

Each clock input drives a single processor core and its associated expansion interfaces and integration logic. The Q-Channel Control interface, CPU<n>CLK Q-Channel Control Interface, allows the subsystem to request and handshake the availability of the **CPU<n>CLK**.

Each clock is expected to be requested to run in any of the following cases:

- Its associated CPU core domain, PD_CPU<n>, or the CPU core's debug domain in the Debug System, PD_DEBUG, are not in OFF, MEM_RET* or FULL_RET* state.
- It is forced to run through the CLOCK_FORCE register.
- Clock request from the logic associated with this clock.

DEBUGCLK

This clock must exist when HASCSS = 1.

This clock drives the Debug System. The Q-Channel Control interface, DEBUGCLK Q-Channel Control Interface, allows the subsystem to request and handshake the availability of the **DEBUGCLK**.

This clock is expected to be requested to run in any of the following cases:

- Its associated CPU core's debug domain in the Debug System, PD_DEBUG, is not in OFF state.
- It is forced to run through the CLOCK_FORCE register.
- Clock request from the logic associated with this clock.

CNTCLK

This clock is associated with the System Timestamp input, CNTVALUE<G/B>.

This clock is used to drive timestamp related logic in all timestamp-based Timers and Watchdogs.

NPU<m>CLK

These clocks must be present when NUMNPU > 0. Each clock input drives a single NPU core and its associated interfaces and integration logic. The Q-Channel Control interface,

NPU<m>CLK Q-Channel Control Interface, allows the subsystem to request and handshake the availability of the **NPU<m>CLK**.

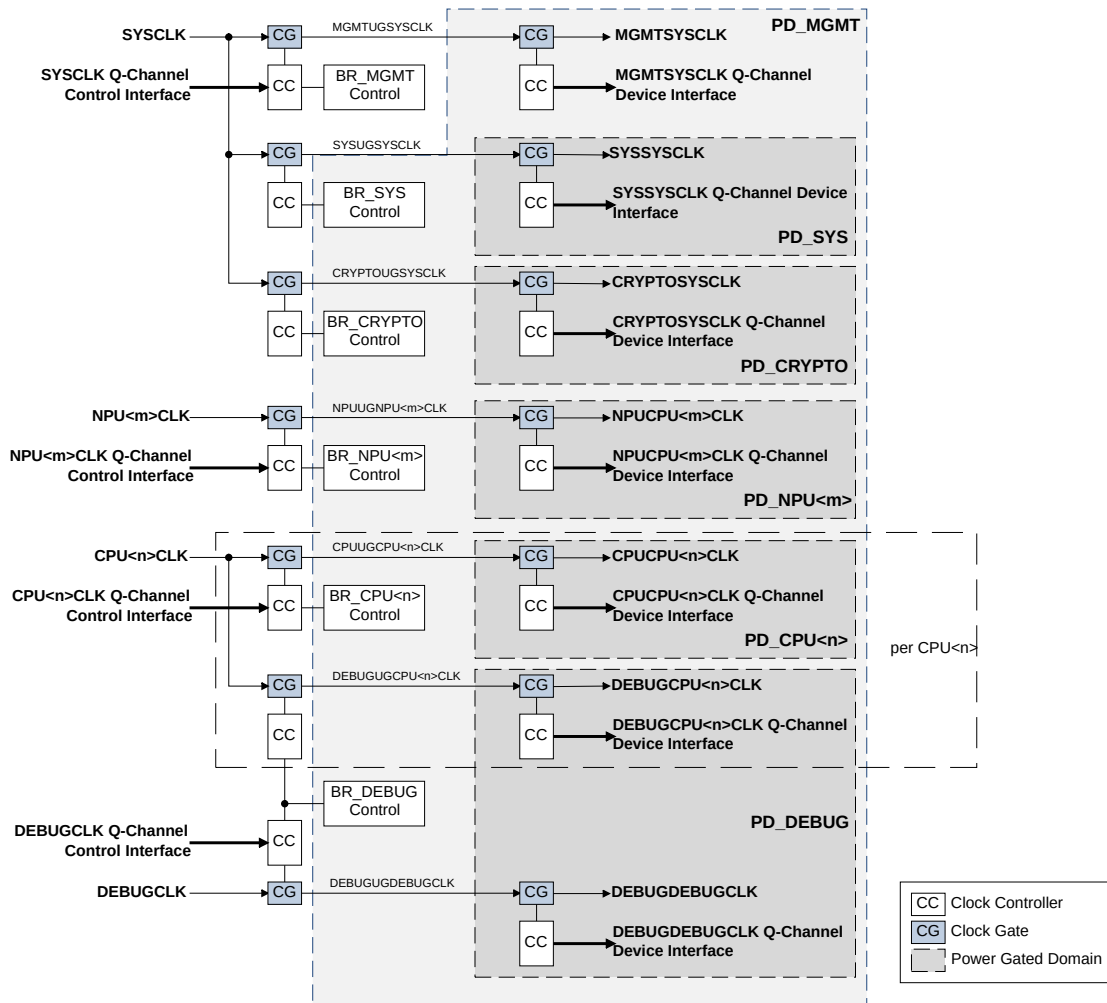
Each clock is expected to be requested to run in any of the following cases:

- Its associated NPU domain, PD_NPU<m>, is not in OFF state.
- It is forced to run through the CLOCK_FORCE register.
- Clock request from logic associated with this clock.

The granularity of the switching of each clock as controlled by each clock Q-Channel control interface is **IMPLEMENTATION DEFINED**. We recommend that all Q-Channel Control interfaces defined here are used in relation to the system power state, where each clock is requested to turn on or off depending on the power state of the domains that use them. The expectation is that there is a higher cycle cost in starting and stopping these clock sources. We do not recommend that these clock Q-Channel control interfaces are used to perform activity based high-level clock gating. Instead, an implementation of CRSAS Ma1 should perform activity based high-level clock gating internally.

The following figure shows an example clock structure showing how the output clocks are related to the input clocks and their power domain relationships.

Figure 5-1: Clock tree example



The diagram does not show internal Q-Channel and P-Channel infrastructure that is required for each clock controller.



Note

All blocks not in a Power Gated Domain are in the always on PD_AON domain. For more on power domains, see sections [Advanced level power infrastructure](#), [Intermediate level power infrastructure](#) and [Basic level power infrastructure](#). Note that in this example, clocks are often gated twice, once in relation to the state of a power domain that the clock is driving and gated again internally within a domain depending on the activity within the domain. An implementation of CRSAS Ma1 can have varying levels of gating and this is **IMPLEMENTATION DEFINED**.

CRSAS Ma1 provides clock source force registers that request the clock sources to continue clock generation regardless of the state of the system. These clock source forces do not affect the power or high-level clock gating provided within the system. For more information, see [CLOCK_FORCE](#).

5.2 Reset infrastructure

CRSAS Ma1 defines the following system level reset scopes, namely:

- Power-on reset
- Cold reset
- Warm reset

It is not possible to have a Power-on reset without also having a Cold reset. It is not possible to have a Cold reset without also having a Warm reset.

On Power-on reset, registers that have a defined reset value contain that value. On Cold reset, same as Power-on reset except the [RESET_SYNDROME](#) remain unchanged. On Warm reset, same as Cold reset except, debug related registers, parts of the power clock management infrastructure as specified in this document remain unchanged. Local derivative of these resets may exits per power or clock domain.

Reset distribution defines how the output resets are related to the input resets and reset requests, and which power domains the reset outputs primarily drive.

CRSAS Ma1 has the following reset input signal to reset the subsystem:

nPORESET

This is the Active LOW Cold reset input for the system. We recommend that the reset is asserted for one or more **SLOWCLK** cycles.

CRSAS Ma1 also has the following reset-related request and handshake signals:

nSRST

This input, typically from an external debugger, is an Active LOW system-wide reset request.

This reset request, when initially asserted, results in a cold reset being applied to the subsystem and then subsequently removed. However, while **nSRST** is asserted, the CPUWAIT inputs of all CPUs are held HIGH, preventing all CPU cores from booting, until **nSRST** is de-asserted. This is regardless of the register setting in the [CPUWAIT](#) register. For more information on **nSRST**, see *Arm® Debug Interface Architecture Specification ADIv6.0*.

While the minimum duration of the assertion of **nSRST** is **IMPLEMENTATION DEFINED**, we strongly recommend that **nSRST** input be at least three **SLOWCLK** cycles long, especially if **nSRST** is asynchronous to **SLOWCLK**. This is because **nSRST** is likely to be treated as a reset request signal that needs to be resynchronized before use, rather than an actual reset signal. It can then be held LOW for as long as is required to hold off the CPU from execution.

RESETREQ

This reset request input allows external expansion logic to request for a Cold reset.

After it is asserted, the signal must be held HIGH until the reset occurs on **nCOLDRESETAON** which must clear this input. This reset is expected to be driven by expansion logic that is hosted by the subsystem.

HOSTRESETREQ

This reset request input allows a higher-level entity to reset the system.

A higher-level entity can for example be a host system that this subsystem is subservient to, which needs to have control over the reset of this system. Asserting this request causes a Cold reset. Holding this request HIGH maintains the subsystem in Cold reset.

Since this input, like **nSRST**, is a request rather than a direct reset input, we strongly recommend that it is asserted for at least three **SLOWCLK** cycles long.

EXPWARMRESETREQ

This expansion Warm reset request output signal, when set to HIGH, indicates that the reset logic is about to assert Warm reset to the system, and waits for the **EXPWARMRESETACK** signal to be HIGH before asserting the reset.

This allows any external logic to delay the assertion of Warm reset to be able to complete any critical operations. If this signal is not used, you must connect **EXPWARMRESETREQ** to **EXPWARMRESETACK**. Once **EXPWARMRESETREQ** is set to HIGH, **EXPWARMRESETACK** must also transition to HIGH in a reasonable time. Similarly, once **EXPWARMRESETREQ** has returned to LOW, **EXPWARMRESETACK** must also transition to LOW in a reasonable time. Failing to do so can cause system deadlock.

The existence of this signal and the associated acknowledge signal is **IMPLEMENTATION DEFINED**.

EXPWARMRESETACK

This expansion Warm reset acknowledge input signal works with **EXPWARMRESETREQ** to allow any external logic to delay the assertion of Warm reset.

The existence of this signal and the associated request signal is **IMPLEMENTATION DEFINED**.

The following outputs are provided for expansion logic hosted by the subsystem:

nCOLDRESETAON

This PD_AON domain reset output is the Cold reset signal intended for use by the subsystem expansion logic. This reset merges other reset sources that are required to cause Cold reset. See [Power-on and Cold reset handling](#).

nWARMRESETAON

The subsystem generates this signal to perform a system Warm reset. The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when the system is in WARM_RST state. It is expected to reset all non-debug related expansion logic that resides in the PD_AON power domain.

nCOLDRESETMGMT

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when PD_MGMT power domain is in lower power (OFF) states. When PILEVEL = 2, it is expected to be used in the PD_MGMT power domain, or when PILEVEL < 2, it is expected to be used to reset logic that was in the PD_MGMT that is merged into PD_AON.

nWARMRESETMGMT

The scope for this reset is a subset of **nCOLDRESETMGMT** and is also asserted when system is in WARM_RST state. When PILEVEL = 2, this reset is used to reset all non-debug related expansion logic in the PD_MGMT power domain. When PILEVEL < 2, this reset is used to reset all non-debug related expansion logic in the PD_AON power domain that was merged from PD_MGMT.

nCOLDRESETSYS

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when PD_SYS power domain is in lower power (MEM_RET/OFF) states. It is expected to reset all expansion logic in the PD_SYS power domain.

nWARMRESETSYS

The scope for this reset is a subset of **nCOLDRESETSYS** and is also asserted when system is in WARM_RST state. It is expected to reset all non-debug related expansion logic in the PD_SYS power domain.

nCOLDRESETDEBUG

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when PD_DEBUG power domain is in lower power (OFF) state. It is expected to reset all expansion logic in the PD_DEBUG power domain.

nWARMRESETCRYPTO

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when system is in WARM_RST state and when PD_CRYPTPO is in lower power (OFF) state. It is expected to reset all non-debug related expansion logic in the PD_CRYPTPO power domain. This reset only exists when HASCRYPTO = 1 and PILEVEL = 2.

nCOLDRESETCPU<n>

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when PD_CPU<n> power domain is in lower power (MEM_RET/OFF) states. It is expected to reset all expansion logic in the PD_CPU<n> power domain.

nWARMRESETCPU<n>

The scope for this reset is a subset of **nCOLDRESETCPU<n>** and is also asserted when system is in WARM_RST state. It is expected to reset all non-debug related expansion logic in the PD_CPU<n> power domain.

nCOLDRESETDEBUGCPU<n>

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when PD_DEBUG system is in lower power (OFF) state. It is expected to reset all debug related expansion logic associated with debug logic of each CPU core that resides in the PD_DEBUG power domain.

The following signal is provided for internal use and defined for reference and clarity:

nWARMRESETNPU<m>

The scope for this reset is a subset of **nCOLDRESETAON** and is also asserted when system is in WARM_RST state and when PD_NPU<m> is in lower power (OFF) state. It is expected to reset all logic in the PD_NPU<m> power domain. This reset must exist when NUMNPU > 0.

In addition to the above, four status outputs are provided to indicate the status of four events that can be used to generate a Cold reset of the system. These are as follows:

NSWDRSTREQSTATUS

Non-secure watchdog reset request status. This output is '1' when the Non-secure Watchdog is raising a reset request and RESET_MASK.NSWDRSTREQEN is '1'. Once set to HIGH, it does not return to LOW unless a reset occurs clearing this status.

SWDRSTREQSTATUS

Secure watchdog reset request status. This output is '1' when the Secure Watchdog is raising a reset request. Once set to HIGH it does not return to LOW unless a reset occurs on **nCOLDRESETAON**, clearing this status.

RESETREQSTATUS

Hardware Reset Request status. This output is set to '1' when the **RESETREQ** input is '1'. Once set to HIGH, it must not be cleared unless the system is reset on **nCOLDRESETAON**, which restores the register field to '0'.

SWRSTREQSTATUS

Software Reset Request Status. This output is '1' when SWRESET.SWRESETREQ is set to '1'. Once set to HIGH, it must not be cleared unless the system is reset on **nCOLDRESETAON**, which restores the register field to '0'.

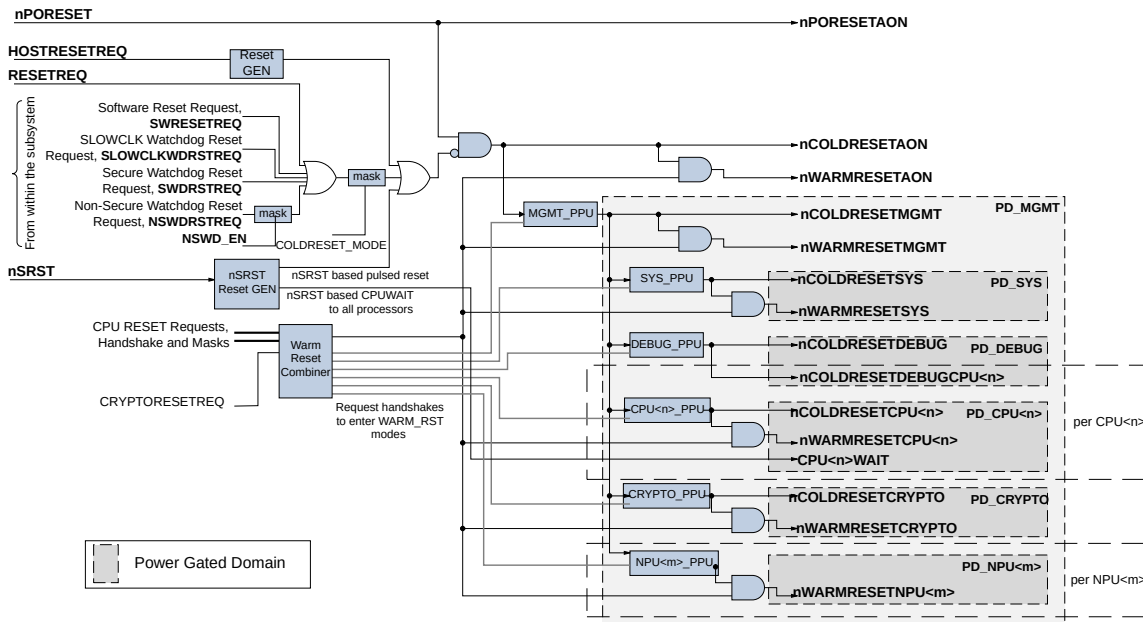
SSWDRSTREQSTATUS

Secure Privileged **SLOWCLK** Watchdog reset request status. This output is '1' when the Secure Privileged **SLOWCLK** Watchdog is raising a reset request. Once set to HIGH, it does not return to LOW unless a reset occurs on **nCOLDRESETAON**, clearing this status.

The following figure shows an example reset structure showing how the output resets are related to the input resets and reset request, and which power domain the reset output primarily drives. The diagram does not show infrastructure for reset resynchronization and how the resets are used within each domain. It shows a reset signal **nPORESETAON** which is not an output of the subsystem and is only used within the system to reset the RESET_SYNDROME register. See [RESET_SYNDROME](#).

An implementation of CRSAS Ma1 can have a different reset tree implementation but the relationships as specified for the reset signals must remain the same.

Figure 5-2: Reset structure example



5.2.1 Power-on and Cold reset handling

The input **nPORESET** is the power-on reset input that resets all registers in the design. This includes resetting the Reset Syndrome, see [RESET_SYNDROME](#).

The **nPORESET** is combined with the masked and combined reset requests to generate the internal combined Cold reset, which is available for use by expansion through the **nCOLDRESETAON** output. These requests are from the following:

- All watchdog timers' reset requests, through a relevant mask if it exists for each, and only if **COLDRESET_MODE** = 0.
- Reset request input **RESETREQ** if **COLDRESET_MODE** = 0.
- Reset request through the **SWRESET.SWRESETREQ** register value, if **COLDRESET_MODE** = 0.
- Reset request input **HOSTRESETREQ**.
- And reset generated from **nSRST** request by using negative-edge detection first and then stretching the result.

The **nCOLDRESETAON** signal resets almost all logic within the system except the [RESET_SYNDROME](#) registers.



In each power domain that is directly controlled by a PPU that resides in the PD_AON or PD_MGMT domain, the PPU is reset using **nCOLDRESETAON** or **nCOLDRESETMGMT** respectively. Each PPU then in turn generates the Cold reset that is used within the power domain it controls. For example, the PPU for PD_SYS is responsible for generating **nCOLDRESETSYS**.

When COLDRESET_MODE = 1, it allows the system to ignore watchdog timers, **RESETREQ** and SWRESET.SWRESETREQ based reset when generating Cold reset. This is essential if the Cold reset control is owned by a different system. When COLDRESET_MODE = 1, if any of the five statuses, **NSWDRSTREQSTATUS**, **SWDRSTREQSTATUS**, **SSWDRSTREQSTATUS**, **RESETREQSTATUS**, **SWRSTREQSTATUS** is '1', the external entity must, in a short time, the duration of which is **IMPLEMENTATION DEFINED**, cause a reset by using the **HOSTRESETREQ** input.



Failing to reset the system can potentially cause system deadlock or even compromise security.

5.2.2 CPU Reset Handling

Each CPU<n>'s **nPORESET** reset input is driven by the PPU that controls each CPU Core power domain, PD_CPU<n>. This PPU itself is reset using **nCOLDRESETMGMT**.

A Warm reset mode is driven from Warm Reset Generation logic that resides in the PD_AON domain. This, along with all PPUs in the system, is used to force the system to idle before driving Warm reset which includes the CPU<n>'s **nSYSRESET** input. The **nSYSRESET** signal of each CPU is generated when the system enters WARM_RST state, ensuring the logic in PD_CPU<n> power domain is quiescent when that occurs.

If the reset is the result of a Cold reset request from the **nSRST** input, after a momentary Cold reset, CPUWAIT input of all CPUs is forced HIGH as long as **nSRST** is held LOW to stop the processor from starting execution until **nSRST** is released. This allows a debugger to hold the CPU core from execution after reset while it uses the Debug Access Port to perform debug operations.

5.2.2.1 Boot after reset

After resets, including power-on reset, all processors boot using the values defined in the Initial Secure Reset Vector Registers **INITSVTOR<n>** in the System Control Registers as the boot address.

The default address is **IMPLEMENTATION DEFINED**, but we recommend that the default is set to 0x0100_0000 which is mapped to code memory through the Manager Code Main Expansion Interface. These addresses can be modified by software before subsequent warm reboots of the processors.

The TrustZone for Armv8-M states that boot must start from a Secure memory space. At boot, all Volatile Memory is Secure only. Software must change or restore the settings in the MPC to release memory for Non-secure world use.

The **CPUWAIT** input to each core can force each processor to wait before executing the instruction. Each CPU in the system has an associated CPU<n>WAIT register that controls if it starts running its boot code when it wakes. There is also a **CPU<n>WAITCLR** input for each CPU<n> to allow an external entity to clear the associated CPU<n>WAIT register bit.

For more information, see [CPUWAIT](#).

5.2.3 NPU Reset handling

If NUMNPU > 0, each **nRESET** input of the NPUs is driven by the PPU that controls the NPU power domain PD_NPU<m> and the Warm reset is driven from Warm Reset Generation logic that resides in the PD_AON domain. This, along with all PPUs in the system, is used to force the system to idle before driving Warm reset. The PPU itself is reset using **nCOLDRESETMGMT**.

5.2.3.1 NPU security and privilege from reset

When the NPU reset is released by the PPU controlling the NPU power domain PD_NPU<m>, the NPU<m> sample the signals driven by NPUSPPORSL.SP_NPU<m>PORSL to determine its default security level. And NPUSPPORPL.SP_NPU<m>PORPL or NPUNSPORPL.NS_NPU<m>PORPL to determine its default privilege level.

For more details, see [NPUSPPORPL](#) and [NPUNSPORPL](#).

For information on how the NPU is expected to transition between security and privilege levels, see [NPU security mapping](#).

5.2.4 Warm Reset Generation

The System provides a Warm reset for each power domain, which is available for the expansion system on the **nWARMRESETAON** signal. This reset is generated by first merging System reset requests from all CPU<n> in the system, and from CryptoCell if it exists after masking each with the values in the [RESET_MASK](#) register. The merged reset request is then used to request all PPUs in the system to enter the Warm reset state. Once all PPUs have entered the Warm reset, which is the WARM_RST system power state, then **nWARMRESETAON** is asserted. Each lower hierarchical domain has its local version of Warm reset signal to perform the Warm reset.



The PPUs are not reset by Warm reset.

The **nCOLDRESETAON** signal also drives **nWARMRESETAON**, but this request is immediate without requesting for all PPUs to enter the Warm reset state. Asserting **nCOLDRESETAON** also resets all PPUs in the system.

5.3 CPU

CRSAS Ma1 supports one to four ARMv8-M MVE processor cores. Each processor can support different static configurations except that the following that must be adhered to:

Table 5-1: CPU configurations that must be supported

Configuration	Value for CPU<n>	Description
SECEXT	1	It specifies whether the Armv8-M Security Extension is included: <ul style="list-style-type: none"> 0: Security Extension not included. 1: Security Extension included.
MPU_NS	> 0	It specifies the number of Non-secure MPU regions included.
MPU_S	> 0	It specifies the number of Secure MPU regions included.
SAU	> 0	It specifies the number of SAU regions included.
NUMIRQ	CPU<n>EXPNUMIRQ + 32	It specifies the number of external interrupts included for each CPU.
IWIC	HASCPU<n>IWIC	It specifies whether the Internal Wake-up Interrupt Controller (IWIC) is included for each CPU: <ul style="list-style-type: none"> 0: IWIC not included 1: IWIC included
WICLINES	CPU<n>EXPNUMIRQ + 35	It specifies the number of IRQ lines supported by the IWIC and EWIC. The value always includes the three internal events (NMI, RXEV, and Debug monitor event) and at least one IRQ.
CTI	If DEBUGLEVEL = 0 then 0, else 1	It specifies whether the Cross Trigger Interface (CTI) unit is included: <ul style="list-style-type: none"> 0: CTI not included 1: CTI included
BUSPROT	0	It specifies whether interface protection is supported on the M-AXI, P-AHB, EPPB, and S-AHB interfaces of the processor: <ul style="list-style-type: none"> 0: Interface protection not included 1: Interface protection included
LOCKSTEP	0	Specifies whether the processor is configured for dual-redundant lockstep operation. The options are: <ul style="list-style-type: none"> 0: Regular processor operation 1: Dual-redundant lockstep operation <p>When LOCKSTEP is set to 1, we recommend that RAR must also be set to 1. Otherwise, software must initialize all registers to prevent accidental triggering of Dual-Core Lock-Step (DCLS) mismatch when the UNKNOWN state gets propagated to the top level. However, such software requirements might not be practical in certain software system environments. Therefore, the RAR option is strongly recommended for DCLS designs.</p>
ITGU	1	It specifies whether the processor is configured with ITCM Security gate unit: <ul style="list-style-type: none"> 0: TCM Gate Unit not included 1: TCM Gate Unit included

Configuration	Value for CPU<n>	Description
DTGU	1	It specifies whether the processor is configured with DTCM Security gate unit: <ul style="list-style-type: none"> 0: TCM Gate Unit not included 1: TCM Gate Unit included
CFGMEMALIAS	0b10000	Memory address alias bit for the ITCM, DTCM, and P-AHB regions. The address bits used for the memory alias are: <p>0b00001: Alias bit = 24.</p> <p>0b00010: Alias bit = 25.</p> <p>0b00100: Alias bit = 26.</p> <p>0b01000: Alias bit = 27.</p> <p>0b10000: Alias bit = 28.</p> <p>0b00000: No Alias. TCM security gating disabled.</p>
CFGBIGEND	0	Data endian format: <ul style="list-style-type: none"> 0: Little-endian (LE) 1: Byte-invariant big-endian (BE8)
CFGITCMSZ	Must not be set to 0b0000	Size of the instruction TCM region, encoded as: <ul style="list-style-type: none"> = 0b0000: No ITCM implemented, which is not supported by CRSAS Ma1. ≥ 0b0011: $2^{\text{CFGITCMSZ}-1}$ KB. Others: Reserved.
CFGDTCMSZ	Must not be set to 0b0000	Size of the data TCM region, encoded as: <ul style="list-style-type: none"> = 0b0000: No DTCM implemented, which is not supported by CRSAS Ma1. ≥ 0b0011: $2^{\text{CFGDTCMSZ}-1}$ KB. Others: Reserved.
CFGPAHBSZ	Must be set to 0b010 - 128MB to complement low and high latency memory map	Size of the Peripheral AHB (P-AHB) port memory region, encoded as: <ul style="list-style-type: none"> = 0b000: P-AHB disabled. = 0b001: 64MB = 0b010: 128MB = 0b011: 256MB = 0b100: 512MB

In the preceding table, the CPU configurations ITGUBLKSZ and DTGUBLKSZ are IMPLEMENTATION DEFINED. However, we recommend that ITGUBLKSZ and DTGUBLKSZ are equal to VMMPCBLKSIZE.

Each CPU also has several miscellaneous interfaces input signals that are tied as shown in the following table.

Table 5-2: CPU interfaces ties that are required

Interface signals	Value for CPU<n>	Description
LOCKDCAIC	LOCKDCAIC	It disables access to the instruction cache direct cache access registers DCAICLR and DCAICRR in the processor.
INITTCMEN[1:0]	0b11	TCM enable initialization out of reset. Set to all ones to enable both TCMs.
WICCONTROL[0]	1	Setting to '1' causes the CPU to use WIC when in WFI DEEPSLEEP.
INITPAHBEN	1	Set to '1' to always enable P-AHB interface. This signal controls the reset value of PAHBCR.EN.

WICCONTROL[0] is connected to bit [0] of **WICCONTROL[3:0]** input signal of the Cortex-M55 or Cortex-M85 CPU.

5.3.1 EVENT Interfaces

Each CPU has event interfaces, RXEV and TXEV. In CRSAS Ma1, all **TXEVs** are logically or together and used to drive all processors' **RXEV**.



In this system, events do not wake a processor from its EWIC based low power state, or wake the system from Hibernation{0/1} state.

CRSAS Ma1 does not support the use of WFE for the CPU to enter DeepSleep state.

5.3.2 Interrupts

CRSAS Ma1 provides the following events that can generate interrupts within the system:

- PPU Interrupts
- Message Handling Units
- Security-based interrupts
- Timers and Watchdogs
- Cross Trigger interrupts
- NPUs
- DMA

In addition, depending on the configuration options, CPU<n>EXPNUMIRQ and CPU<n>EXPIRQDIS, interrupts of each CPU<n> are made available to be driven through expansion logic.

The table below lists the interrupt map of each CPU<n>. For the first 32 interrupts, unless otherwise specified, all CPU cores receive the same interrupt signals. Each CPU<n> only sees its own local CTIIRQ interrupts. Software must ensure that all PPU interrupts must be handled as Secure interrupts. If an interrupt source does not exist because of the chosen configuration of the system, the unused interrupt pin is not used, and the interrupt is disabled and reserved.

The table also indicates those interrupts that also act as wakeup interrupts at each CPU<n>'s associated External Wakeup Interrupt Controller (EWIC) or Internal Wakeup Interrupt Controller (IWIC). The EWIC and IWIC act primarily as entities that take over the masking and holding of an interrupt, on behalf of the CPU<n>'s NVIC, when the CPU<n> is in its OFF or low-power state and is unable on its own to handle interrupts.

Using the EWIC:

1. The system can enter a lower power state that switches off each processor along with most of the system except the EWIC itself.
2. The system can run the EWIC on a much lower clock frequency to lower power consumption and attain a very low standby operating power.

An EWIC always exists with each CPU<n>, but the IWIC only exists for CPU<n> when HASCPU<n> IWIC = 1.

Table 5-3: CPU <n> interrupt map

Interrupt input for CPU <n>	Interrupt Source for CPU <n>	WIC support
NMI	Combined Secure System Watchdog, SLOWCLK Watchdog and CPU<n>EXPNMI	Yes
IRQ[0]	Non-secure Watchdog Reset Request	Yes
IRQ[1]	Non-secure Watchdog Interrupt	Yes
IRQ[2]	SLOWCLK Timer	Yes
IRQ[3]	Timer 0	Yes
IRQ[4]	Timer 1	Yes
IRQ[5]	Timer 2	Yes
IRQ[6]	MHU 0 CPU<n> Interrupt. MHU interrupts are not shared, and each CPU only sees its own interrupt from the MHU. (Reserved if NUMCPU = 0). See Message Handling Unit .	Yes
IRQ[7]	MHU 1 CPU<n> Interrupt. MHU interrupts are not shared, and each CPU only sees its own interrupt from the MHU. (Reserved if NUMCPU = 0). See Message Handling Unit .	Yes
IRQ[8]	CryptoCell Interrupt. (Reserved if HASCRYPTO = 0)	Yes
IRQ[9]	MPC Combined (Secure)	Yes
IRQ[10]	PPC Combined (Secure)	Yes
IRQ[11]	MSC Combined (Secure)	Yes
IRQ[12]	Bridge Error Combined Interrupt (Secure)	Yes
IRQ[13]	Reserved	-
IRQ[14]	PPU_Combined	Yes
IRQ[15]	Reserved	-
IRQ[16]	NPU0	Yes
IRQ[17]	NPU1	Yes
IRQ[18]	NPU2	Yes
IRQ[19]	NPU3	Yes
IRQ[23:20]	Reserved	-
IRQ[24]	DMA (Combined Secure)	Yes
IRQ[25]	DMA (Combined Non-secure)	Yes

Interrupt input for CPU <n>	Interrupt Source for CPU <n>	WIC support
IRQ[26]	DMA (Security violation)	Yes
IRQ[27]	Timer 3 AON	Yes
IRQ[28]	CPU<n>CTIIRQ0 (local CPU CTI only)	No
IRQ[29]	CPU<n>CTIIRQ1 (local CPU CTI only)	No
IRQ[31:30]	Reserved	No
IRQ[<CPU<n>EXPNUMIRQ+31>:32]	CPU<n>EXPIRQ[CPU<n>EXPNUMIRQ-1:0]. See Interrupt Interfaces .	Yes

5.3.3 CPU Custom Datapath Extension interface

If supported by the CPU core, each CPU core of the subsystem can be configured to have a Custom Datapath Extension interface. The method to determine if this interface is present is **IMPLEMENTATION DEFINED**.

If a CPU<n> Custom Datapath Extension interface exists, then the protocol of this interface is **IMPLEMENTATION DEFINED**.

These interfaces reside in their respective processors' PD_CPU<n> power domain, CPUCPU<n>CLK clock domain and nWARMRESETCPU<n> reset domain.



Depending on the actual CPU implementation, the reset can be a special output that is dependent at least on nWARMRESETCPU<n>.

For more information on the Custom Datapath Extensions and related interfaces, see *Arm® Cortex®-M55 Processor Integration and Implementation Manual* or *Arm® Cortex®-M85 Processor Integration and Implementation Manual*.

5.4 System interconnect infrastructure

The system interconnect infrastructure provides a bus infrastructure that transfers memory mapped access from bus managers to subordinates in the system. CRSAS Ma1 defines two key interconnects that form the System Interconnect and they are:

Main Interconnect

This interconnect is expected to provide the highest amount of bus throughput, and is primarily, but not exclusively, for code and data accesses that targets memories or high throughput interfaces. For example:

- In-subsystem volatile memories
- Flash, DRAM controllers or ROM that reside outside the system
- Other high throughput devices.

While the performance and protocol of the interconnect is **IMPLEMENTATION DEFINED**, we recommend that this interconnect is implemented to match the throughput capability of the processor. The Main Interconnect provides access support for all memory regions that are not private to the processors. However, access to the following regions are forwarded to the Peripheral Interconnect:

- 0x4000_0000 to 0x5FFF_FFFF.
- 0xE010_0000 to 0xFFFF_FFFF.

Peripheral Interconnect

This interconnect is expected to provide access to lower performance peripherals. For example:

- System and Watchdog Timers,
- System and other security Configuration Registers
- Power control logic

A Cortex-M MVE based processor has direct interface to access this Interconnect. Any access that does not reside in the following region must be routed to the Main interconnect:

- 0x4000_0000 to 0x5FFF_FFFF.
- 0xE010_0000 to 0xFFFF_FFFF.

While the performance and protocol of the interconnect is **IMPLEMENTATION DEFINED**, we recommend that this interconnect is implemented to have a lower throughput performance compared to Main Interconnect and support a bus protocol that match those needed by the peripherals. The latency of the interface is also expected to be reduced.

The interconnect must be able to achieve the following:

- Transfer the following key properties of the access, from a manager to a subordinate, which is required to ensure that any security-related infrastructure can have all the necessary information to make decisions on access rights.
 - Security attribute indicating an access is marked as Secure or Non-secure.
 - The privilege level of the access, which is either privileged or unprivileged, or of a similar type.
 - Bus access address, and read/write attribute, and optionally, manager identity.
- Complete an access that has been issued. At completion of an access, minimally communicate as a response that the access is successful or has failed.
- Support the ability to atomically read and modify a memory location. For example, through the implementation of read lock and write conditional type exclusive memory access.

The Interconnect resides across PD_SYS and potentially across PD_AON and even PD_MGMT if it exists. The interconnect runs primarily on **SYSCLK**.

TCM Interconnect

This interconnect provides access to the Tightly Coupled Memories (TCM) that are internal to CPU<n> from

- TCM subordinate interface
- Main interconnect
- The DMA

The TCM interconnects follows the mapping and properties of the [TCM subordinate interface](#).

5.4.1 ACC_WAIT Control

CRSAS Ma1 provides a set of controls to let you add access control gates or block access to the system expansion interfaces.

These controls let you:

- Add access control gates, driven using the **ACCWAITn** signal
- Block access to the system, when the system:
 - Leaves HIBERNATION{0-1} state
 - Resets
 - Performs first power up
 - When software wants to reconfigure security settings in the system

These controls prevent access to the system until all security-related features of the system have been set up correctly.

After software is ready, it can release the gates by writing to the **BUSWAIT.ACC_WAITN** register. Then software can check the current status of all external gating units by reading the **ACCWAITNSTATUS** signal value through the **BUSWAIT.ACC_WAITN_STATUS** register.

5.5 Volatile Memory

CRSAS Ma1 can support 0 to 4 volatile memory banks, VM<x>. The number of memory banks is defined by the NUMVMBANK configuration, and therefore x is 0 to NUMVMBANK-1, except when NUMVMBANK = 0 where there is no volatile memory bank.

Each volatile memory (VM) can support a configurable amount of SRAM memory, but must obey the following:

- The size of each is powers of two.
- The size of each bank is defined using the VMADDRWIDTH configuration option and is equal to $2^{\text{VMADDRWIDTH}}$ bytes.
- The total memory size of all volatile memory banks that exist within the system combined is less than 16Mbytes.
- All volatile memory forms a contiguous area of memory and is mapped to a starting address of 0x2100_0000, which is also aliased to a starting address of 0x3100_0000.

All VM<x> must support Exclusive Accesses from the processors and external managers.

Each VM<x> has a Memory Protection Controller (MPC) associated with it that provides the ability to map segments of each memory to Secure world or Non-secure world. For more information on MPC, see *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* and *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual*. CRSAS Ma1 requires that all VMs use the same MPC block size as defined by the configuration VMMPCBLKSIZE. In addition, the **cfg_init_value** input pin of each MPC must be set to 0b0 so that out of reset, all memory locations are mapped to the Secure world by default.

CRSAS Ma1 supports a power domain for each bank, PD_VMR<x>. Each power domain only contains the actual volatile memory of the memory bank with all other logic of the memory banks, including the MPC, residing in PD_SYS. Therefore any power gating or retention only refers to the memory itself. For more information on power control of PD_VMR<x> see [BR_SYS power modes](#).

All VM<x>s run on **SYSSYSCLK**.

5.6 Timers and Watchdogs

CRSAS Ma1 supports two main classes of timers and watchdogs: Timestamp based Timers and **SLOWCLK** AON based timers.

5.6.1 Timestamp-based timers

The first class of timer and watchdogs are timestamp-based. They use the timestamp value provided on the System Timestamp Interface.

For more information on the System Timestamp Interface, see [System Timestamp Interface](#).

The timestamp-based Timers provided in CRSAS Ma1 have the following features:

- Memory-mapped System Timer with register access through the Peripheral Interconnect
- 64-bit timestamp input, generating events through comparison with a timer value
- An 'auto-increment' feature to support regular event generation
- Down counter emulation
- Maskable level interrupt generation

Timestamp-based Watchdog timers are simplified timers that support the following:

- Memory-mapped System Timer with register access through the Peripheral Interconnect
- 64-bit timestamp input, generating events through comparison with a timer value
- An 'auto-increment' feature to support refreshing the watchdog
- Watchdog reset request generation on double watchdog timeout
- Separate refresh register access frame to support refreshing from a different security or privilege level

See [Timestamp-based timers registers](#) and [Timestamp-based Watchdogs registers](#) for details on the timer registers.

A total of four timestamp-based timers are provided along with two timestamp-based watchdog timers. These are mapped to the following addresses:

- Timer 0 at address 0x4800_0000 and aliased to 0x5800_0000.
- Timer 1 at address 0x4800_1000 and aliased to 0x5800_1000.
- Timer 2 at address 0x4800_2000 and aliased to 0x5800_2000.
- Timer 3 at address 0x4800_3000 and aliased to 0x5800_3000.
- Secure Watchdog Timer control frame at address 0x5804_0000 and refresh frame at 0x5804_1000.
- Non-secure Watchdog Timer control frame at address 0x4804_0000 and refresh frame at 0x4804_1000.

Timer 0, Timer 1, Timer 2, and Timer 3 can be configured by software to be a Secure or a Non-secure timer through the Peripheral Protection Controller (PPC), and the software can control that PPC through the Secure Access Configuration Register Block. See [Secure Access Configuration Register Block](#). Security mapping of both watchdog timers are permanently assigned, with one as Non-secure and another as Secure. The control frames are permanently restricted to only privileged accesses while the refresh frames are configurable through PPC to be privileged only or both privileged and unprivileged accessible. All timers and watchdog timers generate interrupts, and the Non-secure watchdog can generate an additional interrupt on a second timeout event for notifying the Secure world. This allows the Secure world to handle the Non-Secure watchdog double timeout instead of directly applying reset to the system immediately. The Secure Watchdog can request a reset of the system if double timeout occurs. The Non-secure Watchdog can be configured by software to do the same if necessary, but by default this is not allowed.

All timestamp-based timers and watchdogs, except for Timer 3, reside in PD_SYS power domain and are reset by **nWARMRESETSYS**, while the Timer 3 resides in the PD_AON power domain and is reset by **nWARMRESETAON**. Therefore, Timer 3 can generate interrupts to wake the system even if the system is in the Hibernation state where PD_SYS is turned off or in retention.

5.6.2 SLOWCLK AON Timers

The second class of timers and watchdogs are simple CMSDK based 32-bit timers that run on **SLOWCLK**. They reside in PD_AON power domain and are reset by **nWARMRESETAON**. A single timer and a single Secure privileged Watchdog are provided and are expected to be used when the system is in HIBERNATION{0-1} when potentially only **SLOWCLK** is available and running and all other clocks are off.

These timers are mapped to the following addresses:

- **SLOWCLK** Timer at address 0x4802_F000 and aliased to 0x5802_F000.
- Secure Privileged **SLOWCLK** Watchdog Timer at address 0x5802_E000.

The **SLOWCLK** Timer can be configured by software to be Secure or Non-secure access only, and privileged or unprivileged access through the Peripheral Protection Controller that is controlled through registers in the Secure Access Configuration Register Block and the Non-secure Access Configuration Register Block. For more information, see [Secure Access Configuration Register Block](#) and [Non-secure Access Configuration register block](#). The watchdog is Secure privilege access only. All CMSDK timers and watchdog timer generate interrupts, but the Secure Watchdog can request a Cold reset of the system if double timeout occurs.

For more information on CMSDK Timers and watchdog, see *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

5.7 Message Handling Unit

When NUMCPU > 0, CRSAS Ma1 implements two Message Handling Units (MHUs) to allow processors to interrupt each other to pass message. Two MHUs are provided so that it is possible for software to place one MHU in the Secure world and another in the Non-secure world. Both MHUs reside in the PD_SYS power domain and are reset using the **nWARMRESETSYS**.

See [Message Handling Unit register map](#) for details on the MHU registers. The MHUs in the system are expected to conform to the MHUv1 specification.

When NUMCPU = 0, there are no MHUs in the system.

5.8 Power Policy Units

CRSAS Ma1 leverages Power Policy Units (PPUs) for power control for each Bounded Region (BR) in the system. Each BR is a collection of power domains where their power states are controlled collectively, usually by a PPU. The following table lists the mandatory configurations of all PPU in the system, which BR each controls, and what power domain each resides in. Other PPU configurations that are not listed here are **IMPLEMENTATION DEFINED**. For more information on Power Policy Units, see *Arm® Power Policy Unit Architecture Specification* and *Arm® CoreLink™ PCK-600 Power Control Kit Technical Reference Manual*.

Bounded Region Controlled by PPU

Table 5-4: Power Policy Unit Associations and Mandated Configurations

PPU configuration	MGMT_PPU	DEBUG_PPU	SYS_PPU ⁴	CPU<n>_PPU ³	NPU<m>_PPU ⁶	CRYPTO_PPU ¹
Bounded Region Controlled by PPU	BR_MGMT	BR_DEBUG	BR_SYS	BR_CPU<n>	BR_NPU<m>	BR_CRYPTO
Power domain the PPU resides in	PD_AON	PD_MGMT ²				
Reset signal used by the PPU	nCOLDRESETAON	nCOLDRESETMGMT				
Device interface type	P-Channel	P-Channel				
Default power policy (DEF_PWR_POLICY)	ON/OFF ⁵	OFF				
Default power mode dynamic transition enable (DEF_PWR_DYN_EN)	1	1				

PPU configuration	MGMT_PPU	DEBUG_PPU	SYS_PPU ⁴	CPU<n>_PPU ³	NPU<m>_PPU ⁶	CRYPTO_PPU ¹
Dynamic support	ON WARM_RST OFF	ON WARM_RST OFF	ON FULL_RET MEM_RST WARM_RST OFF	ON FULL_RET MEM_RST MEM_OFF FUNC_RST LOGIC_RST WARM_RST OFF	ON WARM_RST OFF	ON WARM_RST OFF
Default Operating Policy (DEF_OP_POLICY)	0	0	0	0	0	0
Number of Operating Modes (NUM_OPMODE_CFG)	0	0	2 ^{NUMVMBANK} -1	3	0	0
Operating Mode Active configuration (OP_ACTIVE_CFG)	0	0	1 if NUMVMBANK > 0, else 0	1	0	0
Default Operating Mode Dynamic Transition Enable (DEF_OP_DYN_EN)	0	0	1 if NUMVMBANK > 0, else 0	1	0	0
Default Perform Device Interface Handshake When Transition from ON to WARM_RST mode Enable (WARM_RST_DEVREQEN_CFG)	1	1	1	1	1	1

¹ This PPU only exist when HASCRYPTO = 1 and PILEVEL > 0

² If PILEVEL < 2 PD_MGMT is merged into PD_AON.

³ If PILEVEL = 0, then CPU0_PPU is renamed as SYS_PPU

⁴ If PILEVEL = 0, this SYS_PPU column does not exist, while CPU0_PPU is renamed as SYS_PPU.

⁵ OFF when PILEVEL = 2, else ON

⁶ This PPU only exist when NUMNPU > 0

The PPUs are mapped to secure address space as defined in [System Control Peripheral Region](#). The write accessibility of PPU registers is controlled through the PWRCTRL.PPU_ACCESS_FILTER. When it is set to '1', the system blocks all write accesses to the PPUs, by ignoring the writes, except for the following registers if they exist for each PPU:

- Interrupt Mask Register, at address offset 0x030
- Additional Interrupt Mask Register, at address offset 0x034
- Interrupt Status Register, at address 0x038

- Additional Interrupt Status Register, at address 0x03C

Access to the PPU is normally not required for normal operation because CRSAS Ma1 is architected to use the PPU primarily in dynamic mode, where the request to enter or leave a power state is managed and handshake using the PPU's Device interface. Hence the only time access to PPUs might be required is for debug purposes.

When PWRCTRL.PPU_ACCESS_FILTER is set to '0', all PPU registers are freely accessible to the secure world.

5.9 Peripheral Protection Controllers

Peripheral Protection Controllers in the system enable the software to control whether a peripheral is accessible to the Secure or Non-secure world, and to control privileged access or unprivileged access. PPC protected peripherals can raise interrupt on security violation and response with bus error or **RAZ/WI** depending on the global [SECRESPCFG](#) configuration.

For more information, see *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* and *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual*. In CRSAS Ma1, peripherals that are aliased to two memory areas, one Secure, and another Non-secure, are protected by PPCs. The PPC defines which region the peripheral resides in.

Within CRSAS Ma1, subordinate peripheral interfaces are protected using PPCs. These are controlled by the Secure Access Configuration Register Block and Non-secure Access Configuration Register Block. For more information on these register blocks, see [Secure Access Configuration Register Block](#) and [Non-secure Access Configuration register block](#). These registers also control Security Control Expansion Signals to drive external PPCs.

Two groups of peripherals protected behind Peripheral Protection Controllers are currently defined in CRSAS Ma1 and these are as follows:

- Peripheral Interconnect Peripheral Protection Controller 0 (PPC0). This group includes the following peripherals:
 - Memory Protection Controller configuration register block
 - Secure Access Configuration Register Block
 - Non-secure Access Configuration Register Block
 - Message Handling Units
 - Timestamp-based Watchdog Control Frames
 - Timestamp-based Watchdog Refresh Frames
 - All Timestamp based Timers
 - Message Handling Units (MHUs)
 - Debug System Access
 - Watchdog Refresh Frames. Secure or Non-secure mapping of Watchdog Refresh Frames is fixed, only their privilege levels are configurable.

- All NPUs
- Peripheral Interconnect Peripheral Protection Controller 1 (PPC1). This group includes the following peripherals:
 - System Control Register Block
 - SLOWCLK Watchdog Timer
 - PPUs
 - SLOWCLK Timers



Care should be taken if unprivileged access to bus manager in the system is permitted, for example, DMA or NPUs. If unprivileged access is permitted, then unprivileged code has the potential to use these managers to access and modify privileged memory or peripherals. We recommend that only privileged programming access is permitted to these managers.

For more information, see [Secure Access Configuration Register Block](#).

5.10 Memory Protection Controllers

Memory Protection Controllers (MPCs) in the system partitions memory into pages and allows software to define if each region is Secure or Non-secure.

For more information, see *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* and *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual*. In CRSAS Ma1, each memory page that is protected by the MPC is aliased to two memory areas, one Secure, and another Non-secure. Depending on the security attribute defined for that page in the MPC by software, the page either only exists in the Secure region or the Non-secure region.

MPC protected memory pages can raise interrupt on security violation and response with bus error or **RAZ/WI** depending on the MPC or the [SECRESPCFG](#) register settings. The combined interrupt status of internal and external MPCs is available in the [SECMPCINTSTAT](#) register.

A single MPC is provided for each [Volatile Memory](#) bank and each is mapped into main memory as defined in [Peripheral Region](#).

5.11 Manager Security Controllers

Manager Security Controllers (MSC) in the system transform memory transactions issued by non-CPU managers, that does not support Arm TrustZone for Armv8-M, into memory transactions suitable to Arm TrustZone for Armv8-M systems.

Within CRSAS Ma1, Manager peripheral interfaces are protected using MSCs. These are controlled by the [Secure Access Configuration Register Block](#). MSC protected Managers can raise interrupt on security violation and response with bus error or **RAZ/WI** depending on the global [SECRESPCFG](#) configuration.

For more information, see *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* and *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual*.

5.12 CryptoCell

CRSAS Ma1 supports two possible cryptographic configurations:

- HASCRYPTO = 0: In this No-Crypto configuration, the CryptoCell-312 IP and its associated integration logic do not exist within the system.
- HASCRYPTO = 1: In this Has-Crypto configuration, the CryptoCell-312 IP and its associated integration logic exist within the system.

5.12.1 No-Crypto Configuration

When HASCRYPTO = 0, CryptoCell-312 does not exist in the system. Therefore, all associated interfaces and configuration that are associated with CryptoCell-312 do not need to exist.

In such a system, the root of trust can still be within the system, but if TBSA-M compliance is still required, the system integrator must ensure that the integrated system contains all the necessary resources that a root of trust requires.

5.12.2 Has-Crypto configuration

When HASCRYPTO = 1, CryptoCell-312 exists in the system and therefore, any interfaces and configuration that are associated with CryptoCell-312 also exist.

CryptoCell-312 provides the following:

- Cryptographic acceleration for the protection of data-in-transit (communication protocols) and data-at-rest.
- Protection of various assets belonging to the IC or device manufacturer, service operators providing services over the target device and also for the end user. These asset protection features include:
 - Image verification at boot or during runtime
 - Authenticated debug
 - True random number generation
 - Lifecycle management
 - Provisioning of assets

When CryptoCell-312 exists, the following is provided:

- A configuration interface for CryptoCell that is mapped at aliased address regions 0x4009_0000 to 0x4009_3FFF and 0x5009_0000 to 0x5009_3FFF. This interface provides access to programmer visible registers within CryptoCell-312 and parts of the *Non-Volatile Memory* (NVM) memory with word address of 0x00A0 to 0x1FFC, and CryptoCell-312 itself handles security checking of accesses to its registers on its own.

- Access to the same NVM memory at aliased system addresses 0x0E00_0000 to 0x0E00_1FFF and 0x1E00_0000 to 0x1E00_1FFF. This interface provides access to the NVM, with word address of 0x00A0 to 0x1FFC. Note the address offset of 0xA0 being applied to the system address when accessing the NVM memory.
- CryptoCell can access the system as a manager only to the following address space:
 - Manager Code Main Expansion Interface, at addresses 0x0100_0000 to 0x0DFF_FFFF, and 0x1100_0000 to 0x1DFF_FFFF.
 - All implemented VM areas 0x2100_0000 to 0x21FF_FFF, and 0x3100_0000 to 0x31FF_FFFF.
 - Manager Main Expansion Interface at the following address range:
 - 0x2800_0000 to 0x2FFF_FFFF.
 - 0x3800_0000 to 0x3FFF_FFFF.
 - 0x6000_0000 to 0xDFFF_FFFF.
- Persistent State storage to store key states that must be preserved.
- An NVM interface, which connects to the top level of CRSAS Ma1.

For more information on CryptoCell-312, see *Arm® CryptoCell™-312 Technical Reference Manual*.

5.13 Debug Infrastructure

CRSAS Ma1 supports two possible configurations that define the extent of its debug system infrastructure. These are as follows:

- HASCSS = 0: In this Basic Debug configuration, a CoreSight SoC-600 based common debug infrastructure is not defined and does not exist. Instead, all debug interfaces are brought out from the processor as expansion interfaces. When HASCSS = 0, NUMCPU must be 0.
- HASCSS = 1: In this Full Debug configuration, a CoreSight Soc-600 based common debug infrastructure exists and is called the Debug System.

A debug access to any of the systems obeys normal memory map and decode rules, but must not cause a security violation interrupt to be generated, if the debugger performs an operation that would normally cause one.

5.13.1 Basic Debug Configuration

When HASCSS = 0, there can only be one processor and the system does not include a common shared CoreSight SoC-600 based debug infrastructure. Instead, the following applies:

- The processor's separate debug power domain, if it exists, is merged into a single power domain PD_DEBUG, which is controlled using a single PPU, DEBUG_PPU.
- The processor's debug interfaces, if they exist depending on the DEBUGLEVEL configuration, are brought out as expansion interfaces. See [Debug and trace related interfaces](#).

5.13.2 Full Debug Configuration

When $HASCSS = 1$ and $DEBUGLEVEL > 0$, the system contains a CoreSight SoC-600 based debug infrastructure. This infrastructure provides the following functionality:

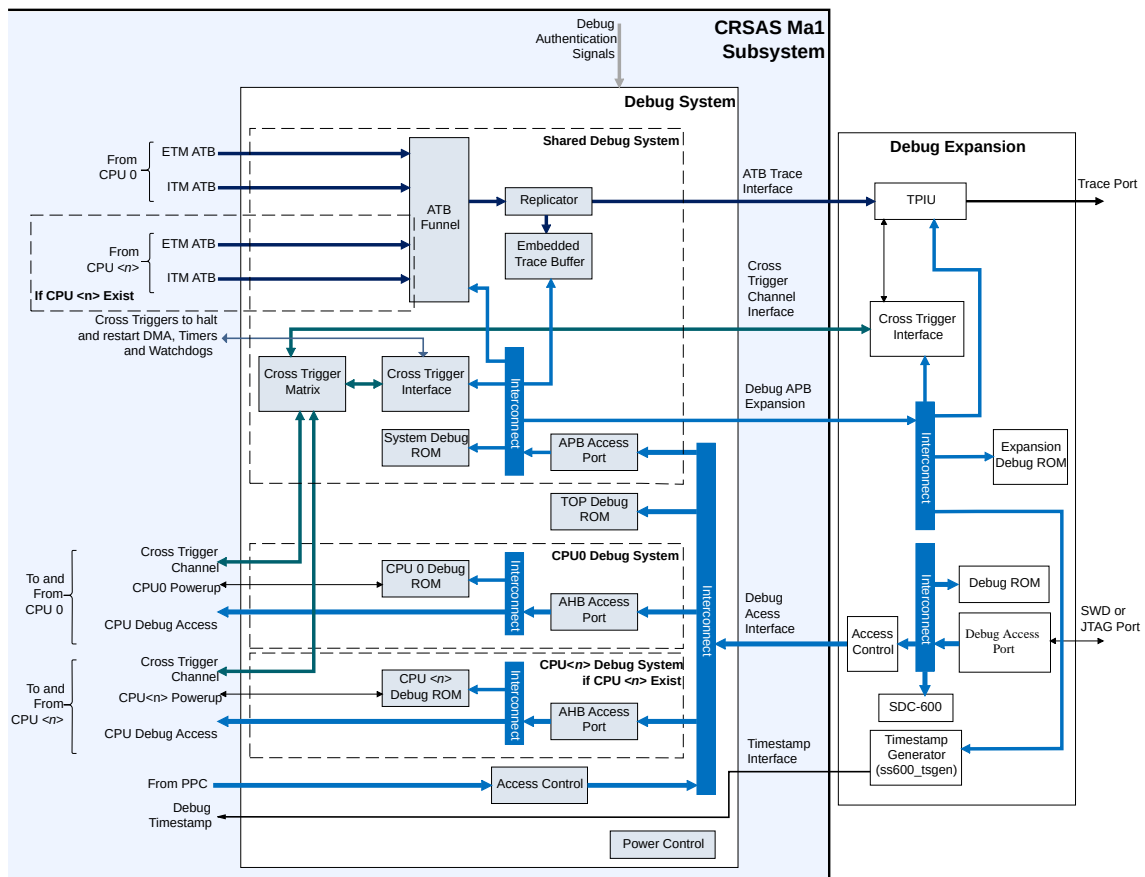
- A single CoreSight Debug Access Port to access an number of Coresight Access Ports (APs). Each of these next level Coresight Access Ports then provides debug access to each individual CPU core and to the Shared Debug System that they all share.
- When $DEBUGLEVEL > 1$, a shared trace infrastructure where all trace data are funneled onto a single trace data output, or to an Embedded Trace Buffer (ETB).
- A System Subordinate interface to the main interconnect to provide System access to the CoreSight Access Ports
- Cross Trigger Matrix and Interfaces to distribute triggers between cores and the Shared Debug System.
- Expansion interfaces which allow additional CoreSight components to be added according to the specific needs of an SoC.

The following figure shows a representative example architecture block diagram of the Debug System.



This example does not show **IMPLEMENTATION DEFINED** components or blocks that may be required due to clock, reset, and power domain crossing, nor does it include bus width conversion or buffering that is not directly visible to software.

Figure 5-3: Example debug subsystem structure



In the above figure, the block “Debug System” is shown as composed of a number of lower level systems blocks that it provides access to. These include:

- CPU<n> Debug Systems which are each associated to a processor in the system,
- A Shared Debug System, which all processors share to provide the ability to merge trace data, support cross triggering, and to support debug expansion.

5.13.2.1 Debug Access

The CoreSight SoC-600 based debug infrastructure provides a single APB4 Debug Access Interface for an expansion Debug Access Port (DAP) to connect to minimum of two and up to five Memory Access Ports (MEM-AP), along with a Debug ROM table with the following purpose:

- One MEM-AP is used for accessing the Shared Debug System infrastructure components, including debug expansion logic on the Debug APB Expansion Interface.
- One MEM-AP for each processor in the system that provides access to each CPU<n> Debug System's Access Interface.

- A Debug System ROM table provides information about the preceding MEM-APs.

System interconnect access to these is also provided, but access is gated using the **SYSDSSACCEN<n>** and **SYSDSSACCENX** Debug Access Control signals to control which of the processors in the system or an **IMPLEMENTATION DEFINED** manager or group of managers are allowed access to the preceding components. These are mapped to address **0xE010_0000** to **0xE01F_FFFF** in the Non-secure world, and to **0xF010_0000** to **0xF01F_FFFF** in the Secure world. The **PERIPHNSPPCO.NS_SYSDSS** register then determines which of the two regions is accessible and the **PERIPHSPPPCO.SP_SYSDSS** determines the privilege level.

CPU Debug Access

To provide Debug access to each CPU core, CRSAS Ma1 provides a MEM-AP accessible through the Debug Access Interface. From this MEM-AP, a Class 0x9 Debug ROM table is provided to do the following:

- Points to the CPU<n> ROM Table that is private to each CPU at PPB address region at **0xE00F_F000**. This ROM table provides Granular Power Requestor (GPR) function as well to allow the debugger to wake CPU<n>.
- Optionally, it also points to another ROM table CPU<n> MCU ROM table added by the system integrator that resides on the PPB address region and on the EPPB expansion bus at address CPU<n>MCUROMADDR. The existence of the pointer to the CPU<n> MCU ROM is **CFG_DEF**.

All access from the MEM-AP that does not access the ROM table is be forwarded to the CPU's Debug Access Interface, and it is **IMPLEMENTATION DEFINED** if those access goes through power and/or clock crossing bridge.



Accessibility through the processor's Debug Access Interface depends on Debug Authentication signals, **DBGEN** and **SPIDEN**, and **DAUTHCTRL.UIDEN** register in the CPU core. For more information, see *Arm® Cortex®-M55 Processor Technical Reference Manual* or *Arm® Cortex®-M85 Processor Technical Reference Manual*.

Shared Debug System Access

CRSAS Ma1 provides a MEM-AP that is accessible through the Debug Access Interface to provide access to the Shared Debug System. This Shared Debug System provides the following:

- Shared Debug System CoreSight ROM that describes all debug components in the Shared Debug System accessible through this MEM-AP. This ROM table includes an entry pointing to an external CoreSight ROM table at address **0x0008_0000** through the Debug APB Expansion Interface that defines what a system integrator has added as expansion.
- A Trace Funnel to funnel all trace data sources together.
- A Replicator and an Embedded Trace Buffer (ETB) to allow trace data to be either forwarded to a TPIU in the expansion logic, or to be temporarily stored in the ETB so that software or the debugger through the Debug Access Interface can read them.
- A Cross Trigger Matrix and a Cross Trigger Interface that support triggers to and from:
 - DMA, Timers, and Watchdogs in the system,

- Triggers from the processor cores,
- Triggers from the expansion system.
- The Debug APB Expansion Interface to add Debug components to the system. This allows a system integrator to extend the Shared Debug System in accordance with the needs of the SoC.

For example, in

- In [Full Debug Configuration](#) the Debug Expansion adds the following to complete the debug solution as a standalone microcontroller:
 - A Debug Access Port (DAP) to allow an external debugger to access the platform.
 - An Access Control Gate, controlled using **DAPDSSACCEN** Debug Authentication Access Control signal.
 - A Cross Trigger Interface (CTI) to provide triggers to and from the TPIU.
 - A Trace Port Interface Unit (TPIU), to output trace data.
 - A Secure Debug Channel APBCOM, to provide an interface for Secure communications between the debugger and the Secure firmware in the system through which Secure debug certificates can be injected to the platform.
 - Debug Timestamp generator.

5.13.2.2 Timestamps

CRSAS Ma1 provides a debug timestamp input, **TSVALUE<B/G>[63:0]**, that is used by all debug logic that requires timestamp within the subsystem. These are primarily the processor cores in the system.

Since each processor core can be running on different clock rates, the timestamp input updates might occur at a rate that is slower than some of these clocks. CRSAS Ma1 does not specify the including of timestamp interpolators to be deployed in the system for faster cores and as a result, when processor clock to the timestamp update rate ratio increases, especially beyond 10:1, it degrades the timestamp granularity, reducing the ability for you to deduce traced events timings accurately between the two clock domains.

5.13.2.3 Cross Trigger

The Shared Debug System implements a Cross Trigger Matrix (CTM) and a single Cross Trigger Interface (CTI). These together allow DMA, timers, and watchdog timers in the CRSAS Ma1 subsystem to be halted and restarted using trigger sources from any of the CPU cores and also from trigger sources external to the subsystem through the Cross Trigger Channel Interface.

The first four Cross Trigger inputs and the first six outputs are reserved for internal use to support the following:

- **SLOWCLK** watchdog and **SLOWCLK** timer halting as follows:
 - **CTIEVENTOUT[0]** of the CTI is used internally to halt the **SLOWCLK** Timer and Watchdog.

- **CTIEVENTOUT[1]** of the CTI is used internally to restart the **SLOWCLK** Timer and Watchdog.
- Triggers to and from the ETB
 - **CTIEVENTOUT[2]** of the CTI is used to drive TRIGIN input of the ETB to request ETB to insert a trigger in a trace stream
 - **CTIEVENTOUT[3]** of the CTI is used to drive FLUSHIN input of the ETB to initiate a flush.
 - **CTIEVENTIN[0]** of the CTI takes the event output FLUSHCOMP of the ETB to indicate that a flush operation is completed.
 - **CTIEVENTIN[1]** of the CTI takes the event output ACQCOMP of the ETB to indicate that trace acquisition is completed.
 - **CTIEVENTIN[2]** of the CTI takes the event output FULL of the ETB to indicate that either the trace memory is full or the write pointer wrapped around.
 - **CTIEVENTIN[3]** of the CTI is reserved and tied 0.
- If NUMDMA > 0, triggers a halt and restart the DMA, reserved if NUMDMA=0
 - **CTIEVENTOUT[4]** of the CTI is used internally to halt the DMA.
 - **CTIEVENTOUT[5]** of the CTI is used internally to restart the DMA.

All other Cross Trigger Interface inputs and outputs of the CTI Block are made available as expansion signals through **CTIEVENTIN[7:4]** and **CTIEVENTOUT[7:6]**.

When integrating a CRSAS Ma1 based subsystem with HASCSS=1, we recommend that the triggers signals are used to also halt the system timestamp counter in the following way

- Use **CTIEVENTOUT[6]** to halt the system timestamp counter.
- Use **CTIEVENTOUT[7]** to restart the system timestamp counter.



It is the responsibility of debug software to halt and restart NPUs in the system, using their programming interface.

5.13.2.4 Trace Infrastructure

When DEBUGLEVEL = 2, CRSAS Ma1 provides a trace infrastructure that funnels all trace source from all processor cores to a single trace data stream. This stream is then replicated and one drives the ATB Trace interface that is expected to be picked up by a TPIU. The other stream is taken up by the ETB, allowing the trace data to be read by software or by an external debugger through the Debug Access Interface.

When DEBUGLEVEL < 2, CRSAS Ma1 does not provide any trace infrastructure.

5.14 System and Security Control

CRSAS Ma1 provides several registers in the system to allow various features of the system to be discovered, configured and controlled. These registers are grouped into four register blocks:

Secure Access Configuration Register Block

The Secure Access Configuration Register Block provides registers to configure Peripheral Protection Controllers (PPC) and Manager Security Controllers (MSC) that reside in the system and in the expansion system through the Security Control Expansion interface. These are Secure access-only registers. For more information on the registers implemented in this block, see [Secure Access Configuration Register Block](#).

Non-secure Access Configuration Register Block

The Non-secure Access Configuration Register Block provides registers to configure Peripheral Protection Controllers (PPC) that resides in the system and in the expansion system through the Security Control Expansion interface. These are Non-secure access-only registers. For more information on the registers implemented in this block, see [Non-secure Access Configuration Register Block](#).

System Information Register Block

The System Information Register Block provides information on the system configuration and identity. For more information on the registers implemented in this block, see [SYSINFO Register Block](#).

System Control Register Block

The System Control Register Block implements registers for power, clocks, resets and other general system control. For more information on the registers implemented in this block, see [System Control Register Block](#).

5.15 Power Control Infrastructure

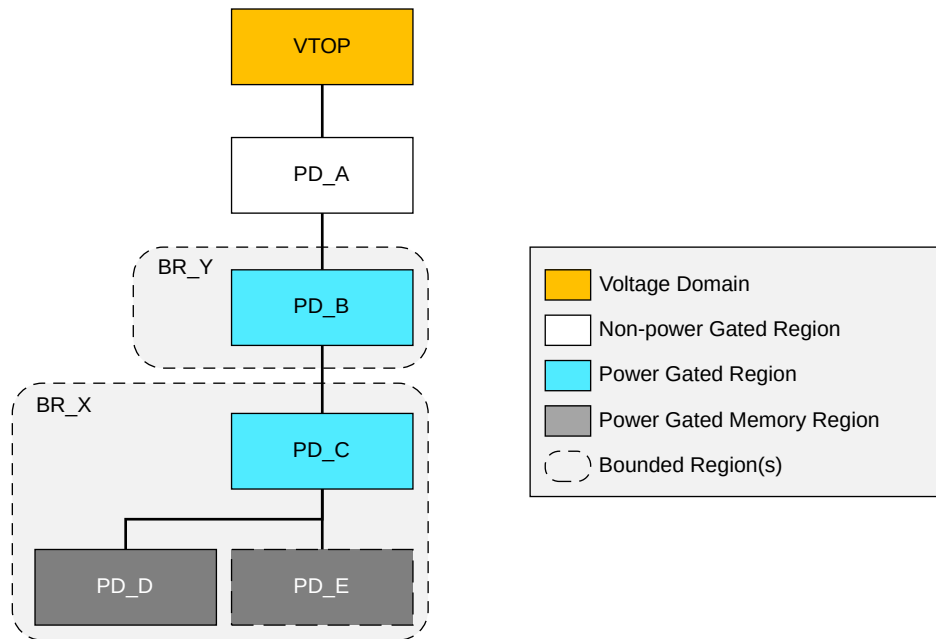
CRSAS Ma1 supports three possible power infrastructure levels:

- PILEVEL = 0: This is Basic Level power infrastructure.
- PILEVEL = 1: This is Intermediate Level power infrastructure.
- PILEVEL = 2: This is Advanced Level power infrastructure.

This section provides a description of the power domain hierarchy, power states of each power domain, and the power control dependencies between the power domains, and, finally the overall system power states.

The following figure shows a simple power hierarchy diagram that is used to describe the power region relationships.

Figure 5-4: Example power hierarchy



In the preceding diagram, each rectangular block represents either a voltage domain or a power domain. If a line connects two blocks where one is above another, it indicates that the domain below is within the hierarchy of the domain above. For example, PD_A region is within the VTOP voltage domain, PD_B is within PD_A, PD_C is within PD_B, and PD_D and PD_E is within PD_C. A block with dotted line indicates that the existence of the region is configuration-dependent.

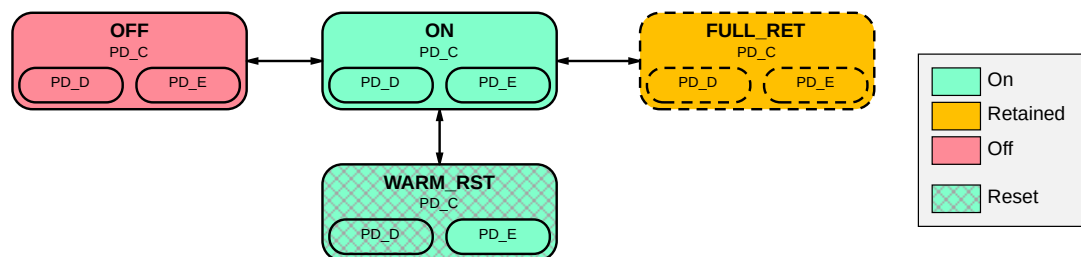
Typically, when a region in the higher-level hierarchy has several regions below it, before that region can enter a lower power state, the lower hierarchy level regions also must already be in a lower power state, or to transition to the lower power state before or at the same time.

A rounded dotted bounding box over one or several regions indicates that their power states are controlled collectively. These are called Bounded Regions (BR). For example, PD_C, PD_D and PD_E regions are in a bounded region BR_X. A bounded region is often controlled using a single Power Policy Unit. PPUs are complemented by LPI infrastructure components to bring together the quiescence status and control of IP blocks primarily within the power domains that are controlled by each PPU.

The power modes of a Bounded Region are represented using a state transition diagram that shows all the modes and the supported transitions between them. For example, the following shows a simple four modes transition diagram, for a Bounded Region with three power regions, PD_C, PD_D and PD_E. Each state is represented by a box that represents the higher hierarchy region power state, with boxes internally that represents lower hierarchy region power states. A name is

given to each power mode in bold. For example, the diagram below only has four BR power modes, OFF, ON, WARM_RST and FULL_RET. The different colors indicate the power state of each region itself. Patterned fill of a region indicates a Bounded Region mode specifically entered to prepare for the application of reset for all registers residing in the corresponding Warm reset domain. PD_E is never reset because it is a memory power domain. Dashed lined block (FULL_RET in the example) indicates that the mode is optional.

Figure 5-5: Example Power Mode Transition Diagram of three bounded power regions, PD_C, PD_D and PD_E



Note

These mode diagrams are very similar to PPU power mode transitions. However, the diagrams here and their modes do not indicate that a power mode in the PPU with the same name is used, although it is likely that they match and it is **IMPLEMENTATION DEFINED** how these are mapped to PPU modes.

A Power Dependency Control Matrix (PDCM) is also defined as a table for each PILEVEL. The PDCM is used to describe and control how each power domain affects another and the lowest power state each domain is allowed to enter.

For each PILEVEL, a System Level Power State table is provided that defines and describes several high-level power states of the system in relation of the power domains states. Each row of table describes a defined System Level Power state, and the supported power states of the power domains and other system states like clock availability and voltage supply, of that state. Combinations of power domain states that are not supported in the table are not supported by the system.

5.15.1 Advanced level power infrastructure

The Advanced level power infrastructure specifies a power control infrastructure that provides additional functionality to help an implementation achieve very low power at the lowest System Power State. This is achieved by providing more voltage and power domains.

The system at PILEVEL = 2 supports two voltage domains:

VAON

The always on voltage domain.

This voltage domain is intended only to drive Always ON logic and therefore only includes the PD_AON domain. As its name suggest, the domain must always be always ON.

This domain is separated from the VSYS to allow always on logic to be implemented using a different target process technology to reduce always on power.

VSYS

The system voltage domain.

This voltage domain implements all other power domains in the system. VSYS, if required, can be powered OFF in the lowest power state as long as all retention requirement of any host logic host within it during the lowest power state are met.

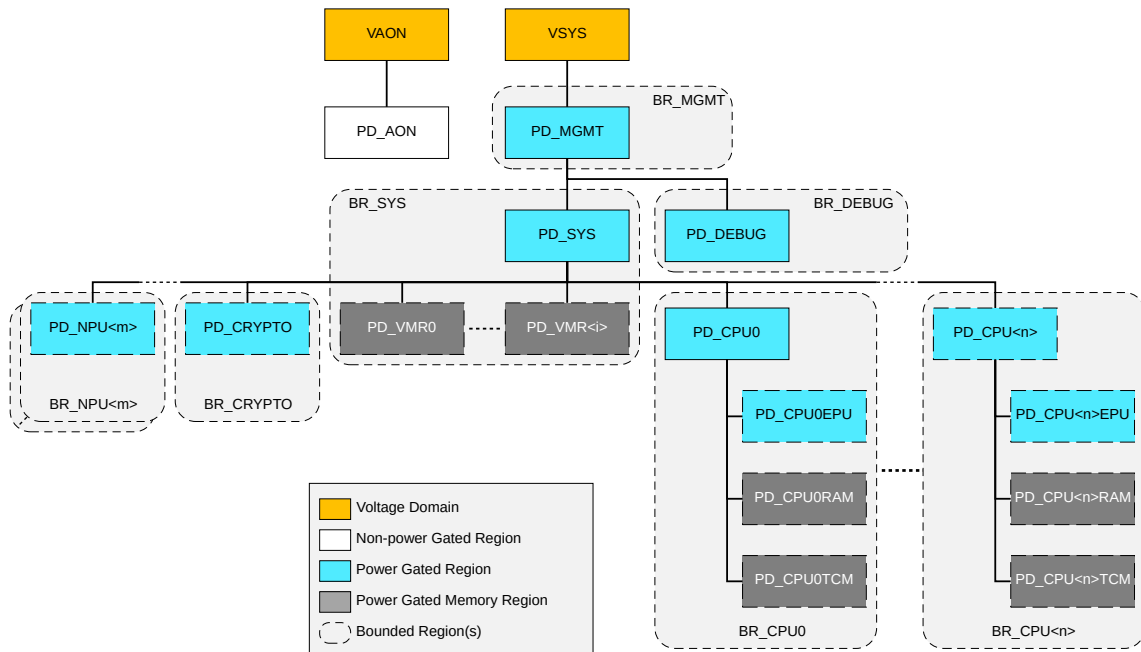
5.15.1.1 Power hierarchy

The system at PILEVEL = 2 supports the following power domains:

- PA_AON
- PD_DEBUG
- PD_MGMT
- PD_SYS
- PD_VMR<*i*>, where *i* is 0 to NUMVMBANK-1, and does not exist if NUMVMBANK = 0
- PD_CRYPTO
- PD_NPU<*m*>, does not exist if NUMNPU = 0
- PD_CPU<*n*>
- PD_CPU<*n*>EPU
- PD_CPU<*n*>RAM
- PD_CPU<*n*>TCM

These power domains are hierarchical and the following figure shows the voltage and power domain hierarchy.

Figure 5-6: Voltage and Power domain hierarchy of a subsystem with PILEVEL = 2



The figure also shows several Bounded Regions. Each of these Bounded Regions is controlled by a single PPU, and these are:

BR_MGMT

This is controlled by a single PPU, MGMT_PPU.

The power domain in BR_MGMT is PD_MGMT

BR_DEBUG

This is controlled by a single PPU, DEBUG_PPU.

The power domain in BR_DEBUG is PD_DEBUG: This is a combined power gated logic and memory domain for all debug logic in the system.

BR_SYS

This is controlled by a single PPU, SYS_PPU.

The power domains in BR_SYS are:

- PD_SYS
- PD_VMR<i> one for each VM<i>, where i is 0 to NUMVMBANK – 1. This power domain does not exist when NUMVMBANK = 0.

BR_CPU<n>

Each is controlled by a single PPU, CPU<n>_PPU.

The power domains in BR_CPU<n> are:

- PD_CPU<n>
- PD_CPU<n>EPU
- PD_CPU<n>RAM
- PD_CPU<n>TCM

BR_CRYPTO

This is controlled by a single PPU, CRYPTO_PPU.

The power domain in BR_CRYPTO is PD_CRYPTO

BR_NPU<m>

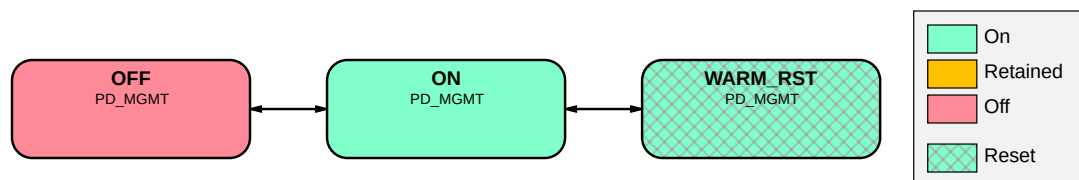
Each Bounded Region is controlled by a single PPU, NPU<m>_PPU.

The power domain in BR_NPU<m> is PD_NPU<m>

5.15.1.2 BR_MGMT power modes

The following figure shows the power modes that BR_MGMT supports.

Figure 5-7: BR_MGMT power mode transition diagram



On Cold reset the PD_MGMT automatically transitions to ON state and enters OFF state eventually. When a Warm reset is requested, the bounded region power state enters the WARM_RST once the domain is idle and ready for reset. The domain dynamically enters the OFF state. It always either enters or stays in the ON state on the following conditions:

- Any of the other power domains that lie within PD_MGMT is entering or is already in the ON state.
- MGMTWAKEUP Q-Channel Device Interface for PD_MGMT is driven to request to leave the OFF state.

BR_MGMT power modes do not specifically support the FUL_RET state but some registers that might reside in the PD_MGMT domain may need to be retained when PD_MGMT is OFF and it is **IMPLEMENTATION DEFINED** how this is achieved.

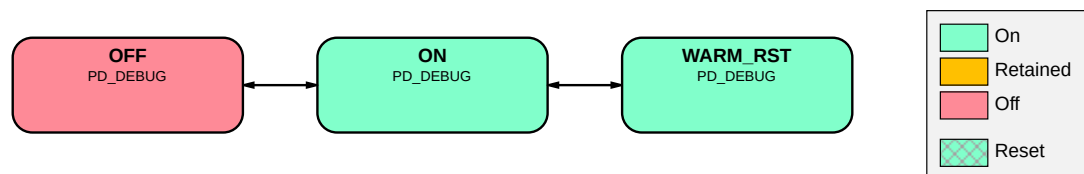
5.15.1.3 BR_DEBUG power modes

On Cold reset, PD_DEBUG automatically transitions to ON state and enters OFF state eventually, if the debug system is not required to stay ON. It is woken either through the PD_DEBUG's Power Control Wakeup Q-Channel Device interface or through a bus access targeting its shared debug domain.

The WARM_RST state is entered to bring the debug domain into an idle state so that the main system can be Warm reset cleanly. The WARM_RST state is entered even though the debug domain is not itself being Warm reset at the same time.

The following figure shows the power modes that BR_DEBUG supports.

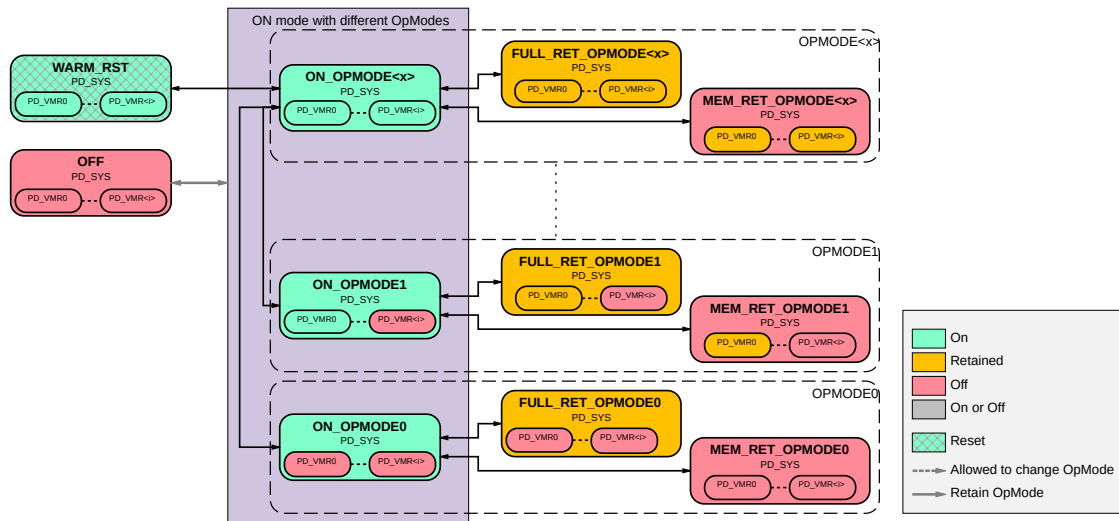
Figure 5-8: BR_DEBUG power mode transition diagram



5.15.1.4 BR_SYS power modes

The following figure shows the power modes that BR_SYS supports.

Figure 5-9: BR_SYS power mode transition diagram



BR_SYS power modes have $2^{\text{NUMVMBANK}}$ different operating modes, which can be 1, 2, 4, 8, or 16. This is encoded as a binary value of up to four bits with each bit for a PD_VMR<i></i>. Each bit therefore indicates a specific PD_VMR<i></i> to be enabled or disabled.

When NUMVMBANK = 0, the BR_SYS power modes only have one operating mode which means that a PPU supporting the BR_SYS does not need to support operating modes. When a PD_VMR<i></i> is disabled, its lower power mode is always the OFF state. Other than OFF or WARM_RST states, it is only possible to move between the operating modes while in one of the ON modes.

For example, for a system with NUMVMBANK = 2

- OPMODE0: PD_VMR0 and PD_VMR1 are both disabled.
- OPMODE1: PD_VMR0 is enabled and PD_VMR1 is disabled.
- OPMODE2: PD_VMR0 is disabled and PD_VMR1 is enabled.
- OPMODE3: PD_VMR0 and PD_VMR1 are both enabled.

On Cold reset, the system enters the ON_OPMODE<NUMVMBANK-1>. In this power mode, all PD_VMR<i></i> are enabled.

When a Warm reset is requested, the bounded region transitions to the WARM_RST through other power modes once the domain is idle and ready for reset.



The DMA includes a minimum power register in its programmers model. If the DMA is included in the system (NUMDMA > 0), it can prevent PD_SYS from transitioning to a power state that is lower than its internal minimum power register allows.

5.15.1.4.1 Controlling PD_VMR<i> power mode

To configure the required low-power state of each PD_VMR<i>, software must configure the register fields PDCM_PD_VMR<i>_SENSE.MIN_PWR_STATE as follows:

- Set to 0b00 to set the low-power state of the PD_VMR<i> to OFF. Therefore PD_VMR<i> only ever transitions between ON and OFF state, and is never in retention, meaning that in low-power state, all state in VM<i> is lost.

With this setting, when PD_SYS is ON and the BR_SYS is in one of the ON_OPMODE modes that currently has PD_VMR<i> ON, it transitions to another ON_OPMODE where PD_VMR<i> is OFF automatically when the PD_VMR<i> domain is idle. Therefore, expect to see PD_VMR<i> turn off quickly once PDCM_PD_VMR<i>_SENSE.MIN_PWR_STATE is set to 0b00.

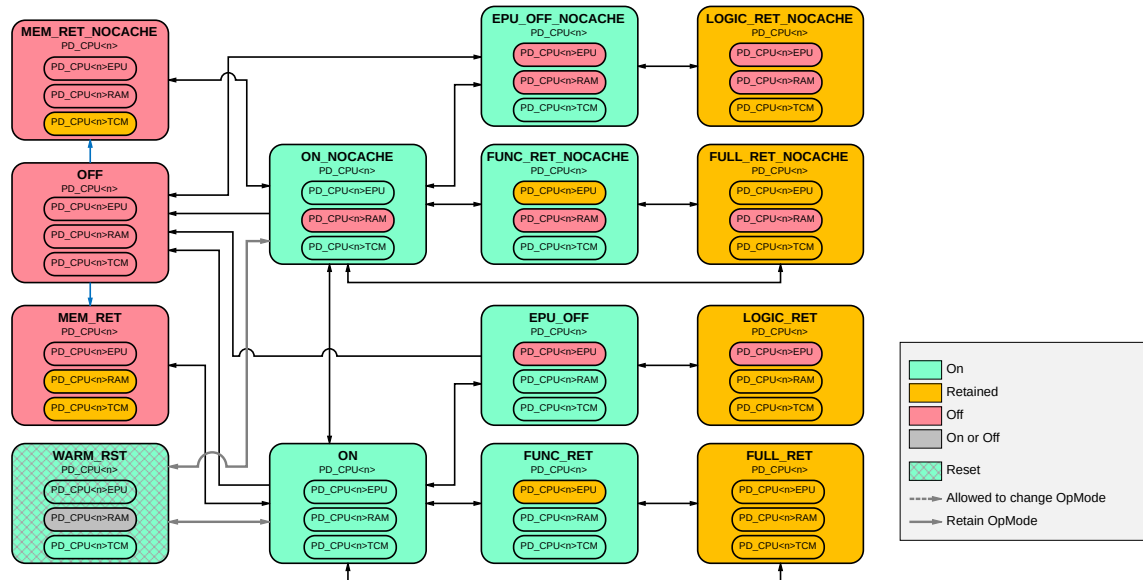
Once PD_VMR<i> is OFF, it can only be returned to ON by an access on the bus targeting PD_VMR<i>, but following that, when PD_VMR<i> is idle again, it turns off again. To avoid this, configure PDCM_PD_VMR<i>_SENSE.MIN_PWR_STATE to a nonzero value before accessing the VM<i>.

- Set to 0b01 to set the low-power state of the PD_VMR<i> to retention. PD_VMR<i> only ever transitions between ON and retention state, and never enters the OFF state, meaning that in low-power state, all register states in VM<i> is retained. Therefore, for BR_SYS, it never transitions to a power mode that has PD_VMR<i> turned off. To place the PD_VMR<i> into retention, the BR_SYS power modes have to enter one of the FULL_RET_OPMOE modes or MEM_RET_OPMODE mode where PD_VMR<i> is in retention. This means that it is not possible to place a PD_VMR into retention while keeping PD_SYS ON.

5.15.1.5 BR_CPU<n> power modes

The following figure shows the power modes that each of the BR_CPU<n> supports.

Figure 5-10: BR_CPU<n> power mode transition diagram



On Cold reset, the bounded domains region enters the EPU_OFF_NOCACHE power mode.

This power mode diagram is like that supported by Cortex-M55 and Cortex-M85 cores, except that an additional PD_CPU<n>TCM power region is added.

The PD_CPU<n>TCM power state always matches that of PD_CPU<n>, except when in MEM_RET or MEM_RET_NOCACHE states where it is to be retained. The choice from ON_NOCACHE to either MEM_RET_NOCACHE or OFF is determined by the CPU's local TCM minimum power state register configuration, CPUPWRCFG.TCM_MIN_PWR_STATE.

When a Warm reset is requested, the bounded region transitions to the WARM_RST through other power mode once the domain is idle and ready for reset.

5.15.1.5.1 Controlling PD_CPU<n>RAM power states

The BR_CPU<n> bounded region uses operating modes to support the ability to turn on or off the cache RAMs in modes other than the OFF mode. Power modes with cache RAMs disabled, called the NOCACHE operating modes, are suffixed with NOCACHE. Power modes with cache RAMs enabled, called the CACHE operating modes, are the modes without the NOCACHE suffix. To select the use of the NONCACHE operating modes, the following registers must be configured:

- Set CPDLPSTATE.RLPSTATE register in the CPU to OFF which is 0b11,

- Set MSCR.DCACTIVE register in the CPU to 0b0 to disable the Data Cache,
- Set MSCR.ICACTIVE register in the CPU to 0b0 to disable the Instruction Cache.

Transitions between the two operating modes can only occur between the ON and ON_NOCACHE power modes.

When in the NOCACHE operating modes, the PD_CPU<n>RAM is always turned off. When operating in the CACHE operating modes, PD_CPU<n>RAM enters RAM retention when entering MEM_RET, LOGIC_RET or FULL_RET power modes.

5.15.1.5.2 Controlling PD_CPU<n>EPU power states

Other than WARM_RST state, the BR_CPU<n> bounded region provides the ability for the PD_CPU<n>EPU to enter a lower power state independently as long as the PD_CPU<n> is ON. To control if the PD_CPU<n>EPU can remain ON or be allowed to enter the Retention (RET) or OFF state, software can use the register CPDLPSTATE.ELPSTATE in conjunction with the CPPWR.SU10 in the CPU as follows:

RET

To allow the PD_CPU<n>EPU to enter Retention (RET) state where BR_CPU<n> never enters EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE and OFF states, set either:

- CPDLPSTATE.ELPSTATE to RET, 0b10, or
- CPDLPSTATE.ELPSTATE to OFF, 0b11 and CPPWR.SU10 to 0b0.

Retention is only entered when CPU<n> is sleeping using WFI or WFE.

OFF

To allow the PD_CPU<n>EPU to enter OFF state where BR_CPU<n> never requests FUNC_RET, FUNC_RET_NOCACHE, FULL_RET_NOCACHE, and FULL_RET states set:

- CPDLPSTATE.ELPSTATE to OFF, 0b11, CPPWR.SU10 to 0b1.

ON

To keep the PD_CPU<n>EPU in ON or context retained state where BR_CPU<n> never requests EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE, FUNC_RET, FUNC_RET_NOCACHE and OFF states, set:

- CPDLPSTATE.ELPSTATE to ON, which is 0b00 or 0b01.



Depending on the processor core and the system implementation, other conditions may be required to reach the desired state. For more information on these conditions see the power control of the processors in, *Arm® Cortex®-M55 Processor Technical Reference Manual*, *Arm® Cortex®-M55 Processor Integration and Implementation Manual*, and *Arm® Cortex®-M85 Processor Integration and Implementation Manual*. For more information on CPPWR, see *Arm®v8-M Architecture Reference Manual*.

5.15.1.5.3 Controlling PD_CPU<n> power states

For the PD_CPU<n> to enter a lower power state, software on the CPU<n> must first configure its CPDLPSTATE.CLPSTATE register. This register defines what power state CPU<n> can enter when in a sleep state as follows:

- Set CPDLPSTATE.CLPSTATE to RET, to allow the PD_CPU<n> to enter retention state only when in a sleep state. When set to RET, BR_CPU<n> never enters OFF, MEM_RET_NOCACHE, or MEM_RET. Other modes can be entered depending on PD_CPU<n>EPU and PD_CPU<n>RAM domain power states.
- Set CPDLPSTATE.CLPSTATE to OFF, to allow the PD_CPU<n> to enter OFF state only when in a sleep state. When set to OFF, BR_CPU<n> never requests LOGIC_RET_NOCACHE and LOGIC_RET modes. Other modes can be entered depending on PD_CPU<n>EPU and PD_CPU<n>RAM domain power states, and if a coprocessor CP<i> that is included in the system is indicating that the state cannot be lost CPPWR.SU<i>=0b0.
- Set CPDLPSTATE.CLPSTATE to ON, that is to 0b00 or 0b01 to keep PD_CPU<n> ON even when it is in a sleep mode, which in turn means that BR_CPU<n> never enters LOGIC_RET_NOCACHE, LOGIC_RET, FULL_RET_NOCACHE, FULL_RET, OFF, MEM_RET_NOCACHE, and MEM_RET modes. Other modes can be entered depending on PD_CPU<n>EPU and PD_CPU<n>RAM domain power states



Note

Depending on the processor core and the system implementation, other conditions may be required to reach the desired state. For more information on these conditions see the power control of the processors in, *Arm® Cortex®-M55 Processor Technical Reference Manual*, *Arm® Cortex®-M55 Processor Integration and Implementation Manual*, and *Arm® Cortex®-M85 Processor Integration and Implementation Manual*. For more information on CPPWR, see *Arm®v8-M Architecture Reference Manual*.

For each CPU to then enter a lower power state, the CPU must enter WFI DeepSleep state. In CRSAS Ma1, DeepSleep always uses a WIC. External WIC for each CPU always exists, and all External WICs reside in the PD_AON domain. Depending on HASCPU<n>IWIC:

- If HASCPU<n>IWIC is set to 0b0 for CPU<n>, then Internal WIC does not exist for CPU<n> and the External WIC is always used for CPU<n>.
- If HASCPU<n>IWIC is set to 0b1 for CPU<n>, then Internal WIC also exists, and the choice of which WIC to use is a software choice through CPUPWRCFG.USEIWIC. See [CPUPWRCFG](#).

Any Internal WIC that exists resides in associated PD_CPU<n> power domain.

The following are the types of power states that each processor in CRSAS Ma1 supports from a programmers point of view, and how to enter each:

OFF - DeepSleep state

Allows the CPU to turn off but utilizes interrupts through the external WIC to wake the CPU. To enter this state, the CPU must select to use the External WIC through the CPUPWRCFG.USEIWIC register, set the CPU<n>'s CPDLPSTATE.CLPSTATE to OFF and

enable DeepSleep, before entering WFI. If Internal WIC is selected PD_CPU<n> is not able to turn off and remains ON.

RET - DeepSleep state

Allows the CPU to enter retention state and utilizes interrupts through the external WIC to wake the CPU. To enter this state, the CPU must select to use the External WIC through the CPUPWRCFG.USEIWIC register, set CPDLPSTATE.CLPSTATE to RET and enable DeepSleep, before entering WFI. Note that if Internal WIC is selected instead, the PD_CPU<n> is not able to enter retention and remains ON.

ON - DeepSleep state

Allows the CPU to enter a sleep state that only supports stopping the CPU clock internally, including to the NVIC. It can use either the External or Internal WIC to wake the CPU. To enter this state, the CPU must set CPDLPSTATE.CLPSTATE to 0b01, ON, enable DeepSleep before entering WFI. Selection of External or Internal WIC can be performed using the CPUPWRCFG.USEIWIC register prior to entering WFI.

ON - Sleep state

Allows the CPU to enter a sleep state that still is ON, keeping its NVIC clocking and running with the rest of the core clock turned off. To enter this state, the CPU must not enable DeepSleep or set CPDLPSTATE.CLPSTATE to ON, 0b00, before entering WFI or WFE.

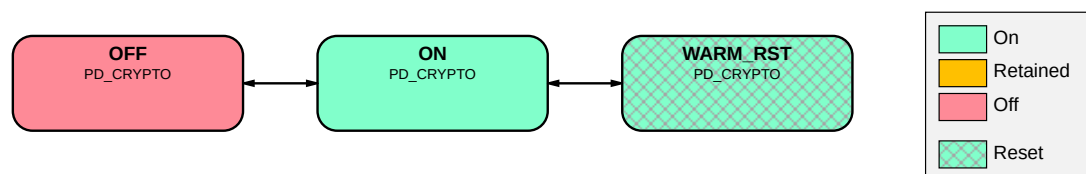
ON state

The CPU's normal running state. In this state, the EPU and the RAMs in the CPU have a degree of separate control as detailed in [Controlling PD_CPU<n>RAM power states](#) and [Controlling PD_CPU<n>EPU power states](#).

5.15.1.6 BR_CRYPTO power modes

The following figure shows the power modes that BR_CRYPTO supports.

Figure 5-11: BR_CRYPTO power mode transition diagram



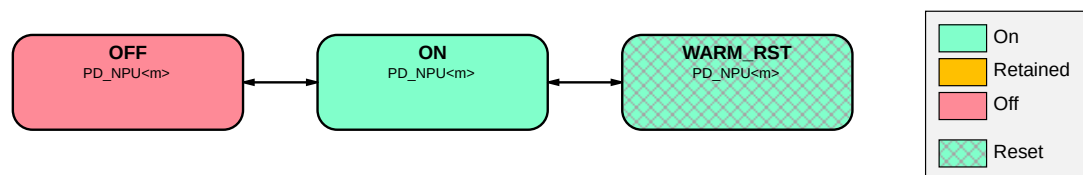
On Cold reset, the bounded domain enters the ON state.

When a Warm reset is requested, the bounded region transitions to the WARM_RST once all dependent domain is idle and ready for reset.

5.15.1.7 BR_NPU<m> power modes

The following figure shows the power modes supported by BR_NPU<m>.

Figure 5-12: BR_NPU power mode transition diagram



On Cold reset, the bounded domain enters the ON state.

When a Warm reset is requested, the bounded region transitions to the WARM_RST once all dependent domain is idle and ready for reset. The NPU has a power control register that can be programmed to prevent the NPU powering off. For more information of this register, see *Arm® Ethos™-U55 NPU Technical Reference Manual*. After reset the NPU does not block power down, once it enters an idle state.

5.15.1.8 Advanced level power dependency control

The following table shows an example, with PDCMQCHWIDTH of 4, and four CPU cores, how each of the power domains are affected by the power state of the other domains.

The heading row of the table lists the Power Domains that are being controlled while the left column lists the Power dependency inputs. Power dependency inputs are either:

- The “ON” state of each power domain in the system. For example, PD_MGMT_ON means PD_MGMT is ON when asserted.
- Expansion Power Dependency Control Matrix Q-Channel Signals, **PDCMONQREQn** and **PDCMRETQREQn** that are driven by expansion logic from outside the subsystem indicating a keep-up request from external power domains.

“Conf” indicates that the power domain is software configurable while “Y” indicates that it is always sensitive to the respective dependency input. For example, PD_SYS can be software configured to be sensitive to the ON state of PD_SYS and all Expansion Power Control Dependency inputs, and it is always sensitive to the ON state of all PD_CPU<n>, PD_NPU<m> and PD_CRYPTO. If a power domain is sensitive to an ON dependency input, it means that once the power domain being controlled is already ON, if any of the dependency inputs is ON or true, then the power domain remains ON. The exception is with the **PDCMRETQREQn** inputs, where, if a power domain is configured to be sensitive to these, the power domain maintains at least in a retention state. The PDCM is used primarily to define when a power domain should not enter a lower power state. It is not designed to support powering up of any power domain.

Table 5-5: Power Dependency Matrix when PILEVEL = 2

Power Dependency Inputs/ Power Domain	PD_MGMT	PD_SYS	PD_VMR0 ¹	PD_VMR1 ¹	PD_VMR2 ¹	PD_VMR3 ¹
PD_MGMT_ON	Conf	-	-	-	-	-
PD_SYS_ON	Y	Conf	-	-	-	-
PD_CPU0_ON ²	Y	Y	Conf	Conf	Conf	Conf
PD_CPU1_ON ²	Y	Y	Conf	Conf	Conf	Conf
PD_CPU2_ON ²	Y	Y	Conf	Conf	Conf	Conf
PD_CPU3_ON ²	Y	Y	Conf	Conf	Conf	Conf
PD_NPU0_ON ⁴	Y	Y	Conf	Conf	Conf	Conf
PD_NPU1_ON ⁴	Y	Y	Conf	Conf	Conf	Conf
PD_NPU2_ON ⁴	Y	Y	Conf	Conf	Conf	Conf
PD_NPU3_ON ⁴	Y	Y	Conf	Conf	Conf	Conf
PD_CRYPTON ³	Y	Y	-	-	-	-
PD_DEBUG_ON	Y	-	-	-	-	-
PDCMONQREQn[k] ⁵	Y	Conf	Conf	Conf	Conf	Conf
PDCMRETQREQn[k] ⁵	Conf	Conf	Conf	Conf	Conf	Conf

¹ PD_VMR<i> columns do not exist if $i > (\text{NUMVMBANK} - 1)$

² PD_CPU<n>_ON rows do not exist if $n > \text{NUMCPU}$

³ PD_CRYPTON row does not exist if $\text{HASCRYPTO} = 0$

⁴ PD_NPU<m>_ON rows do not exist if $\text{NUMNPU} = 0$

⁵ **PDCMONQREQn[k]** and **PDCMRETQREQn[k]**, where k is $\{0 - <\text{PDCMQCHWIDTH}-1\}$

PD_VMR<i> can also be configured to be sensitive to a power domain. For example, if PD_VMR0 sensitivity is configured by software to be sensitive to PD_CPU0_ON, and if PD_CPU0 is ON, the memory also remains ON. However, PD_VMR0 is controlled using the same PPU as PD_SYS and as a result of the power state of the PPU shown in [BR_SYS power modes](#), whenever PD_VMR0 is ON, PD_SYS also remains in one of the ON_OPMODE states where PD_VMR0 is ON.



PD_SYS and PD_MGMT can be configured to be sensitive to themselves. When set up by software to do so, each domain remains ON once it is ON.

The intention of the PDCM and all other sensitivity defined for each power domain is to allow, as much as possible for the power control of the System to be performed primarily using dynamic power transitions. This reduces the number of software interactions needed for system management and therefore improves its responsiveness and contributes to further power reduction.

CRSAS Ma1 also provides a programmable register for the following power domains shown in the following table, which defines the lowest power mode that each domain can enter. PD_DEBUG, PD_NPU<m> and PD_CRYPTO, if they exist, defaults to a minimum power state of OFF.

Table 5-6: Power Domain's Minimum Power State, for PILEVEL = 2

Power Domain	Supported MIN_PWR_STATE for each domain
PD_MGMT	ON, OFF.
PD_SYS	ON, OFF, Retention.
PD_VMR<i>	ON, OFF, Retention.

The MIN_PWR_STATES of PD_MGMT, PD_SYS and PD_VMR therefore help to determine which power modes of the BR_SYS PPU can be used next when performing dynamic power transition.

For example, if the system is idle and all dependent domains are not ON, and then the BR_SYS PPU tries to enter the bounded domain collectively to a low-power state, and if MIN_PWR_STATE of PD_SYS is set to Retention, then BR_SYS is never allowed to enter the OFF state nor any of the MEM_RET_OPMODE<x> states because PS_SYS is not allowed to turn off. Then depending on current PD_VMR<i> state, it then tries to enter one of associated FULL_RET_OPMODE<x> states.

In another example, If PDCM_PD_VMR<x>_SENSE.MIN_PWR_STATE of PD_VMR0 is ON and all MIN_PWR_STATE of other PD_VMRs are OFF, then, if currently all PD_VMR<i> are ON when BR_SYS PPU is entering a low-power state, it transitions to ON_OPMODE1 state to turn off all PD_VMR<i> except PD_VMR0. It never enters FULL_RET_OPMODE1 nor MEM_RET_OPMODE1.

5.15.1.9 System Level Power States

Through the relationship defined by the Power dependency control matrix, the MIN_PWR_STATES of each power domain, and the CPU minimum power states, we can define several System Power States that the system supports.

SYS_OFF

All voltage and power domains are OFF.

HIBERNATION1

The VSYS voltage domain maybe ON or OFF, and the system is in the lowest power state that can still be woken from sleep. At wake, the system has to reboot.

HIBERNATION0

The VSYS voltage domain is ON, and the system is in the second lowest power state where PD_MGMT is still ON. The system can be woken from sleep, potentially quicker than from HIBERNATION1. At wake, the system has to reboot.

SYS_RET

The system is in retention, and at wake, the system can continue to execute since no system state is lost.

SYS_ON

The system is ON.

The following table lists the power states of each power domain that each of the System Power State supports or requires. See [Controlling PD_CPU<n> power states in BR_CPU<n> power modes](#).

Table 5-7: System Level Power States

Power Domains	System Level Power States				
	SYS_OFF	HIBERNATION1	HIBERNATION0	SYS_RET	SYS_ON
VAON (PD_AON)	OFF	ON	ON	ON	ON
PD_MGMT	OFF	OFF	ON	ON	ON
PD_SYS	OFF	OFF	OFF	RET	ON
PD_CPU<n> ²	OFF	OFF - DeepSleep	OFF - DeepSleep	OFF - DeepSleep RET - DeepSleep	OFF - DeepSleep RET - DeepSleep ON - DeepSleep ON - Sleep ON
PD_NPU<m> ³	OFF	OFF	OFF	OFF	OFF/ON
PD_DEBUG	OFF	OFF	OFF/ON	OFF/ON	OFF/ON
PD_CRYPT0 ¹	OFF	OFF	OFF	OFF	OFF/ON
PD_VMR<i> ⁴	OFF	OFF/RET	OFF/RET	OFF/RET	OFF/ON

¹ This power domain only exists when HASCRYPTO = 1.

² When CPU<n> is in OFF-DeepSleep state, the TCM associated with that CPU can either be OFF or be in retention and is defined by CPUPWRCFG.TCM_MIN_PWR_STATE. In this state, the EPU must be OFF. The PD_CPU<n>RAM can be OFF permanently and is defined by CPDLPSTATE.RLPSTATE.

³ These power domains only exist when NUMNPU > 0.

⁴ This power domain only exists if NUMVMBANK > 0.

The following points can be made based on the preceding table:

- When PD_CRYPT0 is ON, PD_SYS must also be ON and therefore PD_MGMT is also ON.
- When PD_NPU<m> is ON, PD_SYS must also be ON and therefore PD_MGMT is also ON.
- When waking up any of the PD_CPU<n> or PD_CRYPT0 power domains, if the PD_SYS is OFF or RET, it always also automatically wakes the PD_SYS domain to ON since PD_CPU<n> ON and PD_CRYPT0 ON states are only supported in SYS_ON System Power State. This means that PD_MGMT also automatically turns ON.
- In the HIBERNATION0, PD_MGMT is ON and PD_DEBUG and can be woken independently.
- In the HIBERNATION1, PD_MGMT is OFF. Anything that wakes the system must also wake PD_MGMT.

- PD_VMR<i> is tied to PD_SYS because they belong to the bounded region BR_SYS. You are not able to wake PD_VMR<i> independently from PD_SYS.

An additional transient WARM_RST state exists for the system when a Warm reset is being performed. This state can only be entered from SYS_ON state. In this state, all power domains, except PD_AON, temporarily enter WARM_RST mode and exit it when Warm reset is completed.



When PD_MGMT is OFF, VSYS can also turn OFF. However, in some scenarios depending on implementation or integration, VSYS can remain ON if VAON is also ON.

5.15.2 Intermediate level power infrastructure

The Intermediate Level power infrastructure makes some simplifications to reduce the complexity of the infrastructure at the cost of reducing the support provided to reduce always-on leakage power and power during HIBERNATE1 state.

The system at PILEVEL = 1 supports only one voltage domain:

VSYS

The system voltage domain.

The previous VAON voltage domain is now merged into the system voltage domain and therefore VSYS is now also considered as an always ON voltage domain.

In addition, PD_MGMT is merged into the PD_AON domain.

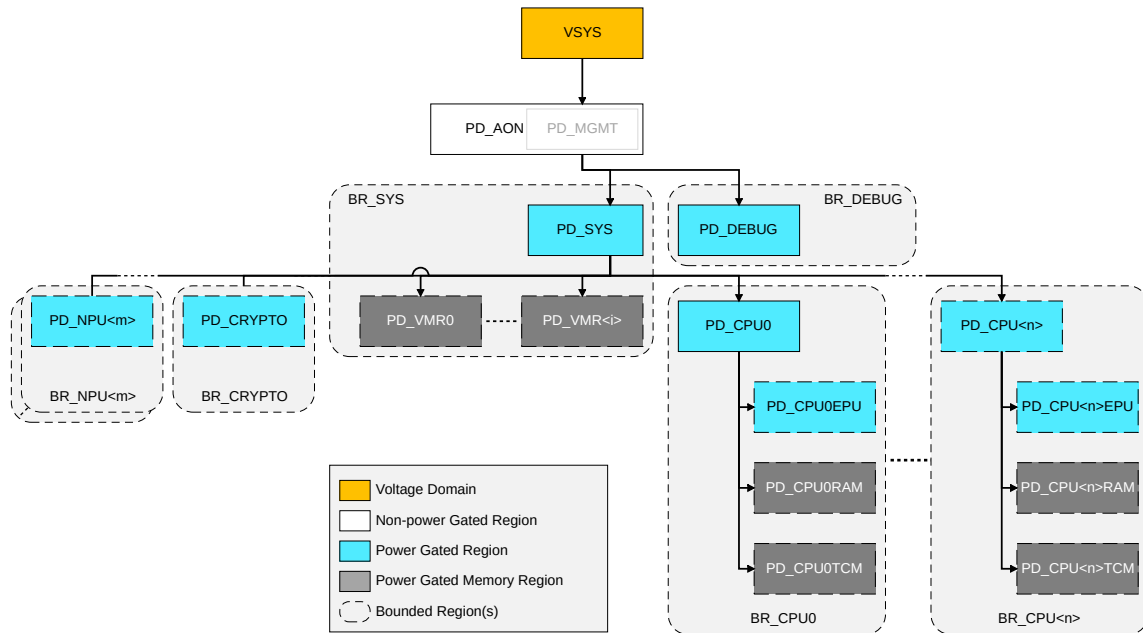
5.15.2.1 Power hierarchy

The system at PILEVEL = 1 supports several power domains:

- PD_AON
- PD_DEBUG
- PD_SYS
- PD_VMR<i>, does not exist if NUMVMBANK = 0
- PD_CRYPTO
- PD_NPU<m>, does not exist if NUMNPU = 0
- PD_CPU<n>
- PD_CPU<n>EPU
- PD_CPU<n>RAM
- PD_CPU<n>TCM

These power domains are hierarchical and the following figure shows the voltage and power domain hierarchy. In the diagram, PD_MGMT is now merged and part of PD_AON.

Figure 5-13: Voltage and Power domain hierarchy of a subsystem with PILEVEL = 1



The figure also shows several bounded power regions:

BR_DEBUG

This is controlled by a single PPU, DEBUG_PPU.

The power domain in BR_DEBUG is:

- PD_DEBUG. This is a combined power gated logic and memory domain for all debug logic in the system.

BR_SYS

This is controlled by a single PPU, SYS_PPU.

The power domains in BR_SYS are:

- PD_SYS
- PD_VMR<i>, one for each VM<i>, where *i* is 0 to NUMVMBANK-1. PD_VMR<i> does not exist when NUMVMBANK = 0.

BR_CPU<n>

Each is controlled by a single PPU, CPU<n>_PPU.

The power domains in BR_CPU<n> are:

- PD_CPU<n>
- PD_CPU<n>EPU
- PD_CPU<n>RAM
- PD_CPU<n>TCM

BR_CRYPTO

This is controlled by a single PPU, CRYPTO_PPU.

The power domain in BR_CRYPTO is:

- PD_CRYPTO

BR_NPU<m>

Each NPU bounded region is controlled by a single PPU, NPU<m>_PPU.

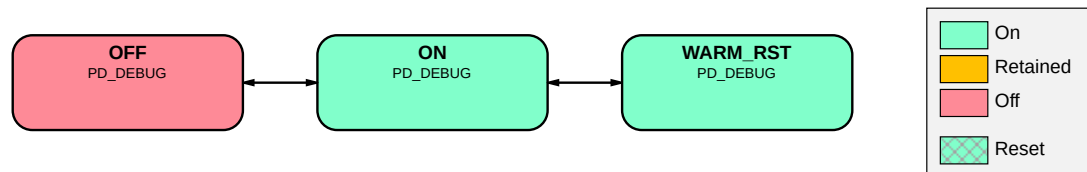
The power domain in BR_NPU<m> is:

- PD_NPU<m>

5.15.2.2 BR_DEBUG power modes

The following figure shows the power modes that BR_DEBUG supports.

Figure 5-14: BR_DEBUG power mode transition diagram

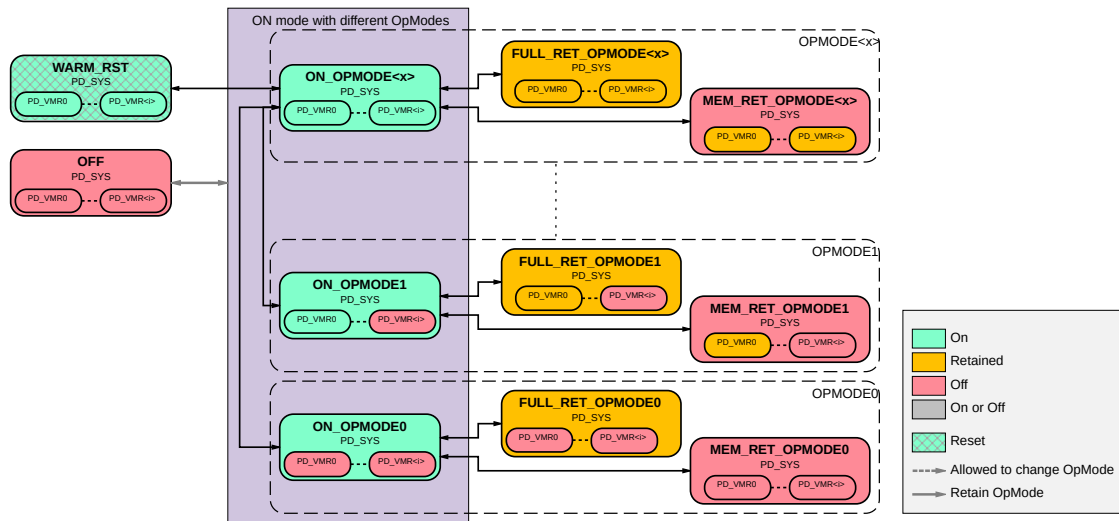


On Cold reset, PD_DEBUG automatically transitions to ON state, and enters OFF state eventually if the debug system is not required to stay ON. It is woken either through the PD_DEBUG's Power Control Wakeup Q-Channel Device Interface interface or through a bus access targeting its shared debug domain. The WARM_RST state is entered to bring the debug domain into an idle state so that the main system can be Warm reset cleanly. The WARM_RST state is entered even though the debug domain is not itself being Warm reset at the same time.

5.15.2.3 BR_SYS power modes

The following figure shows the power modes that BR_SYS supports.

Figure 5-15: BR_SYS power mode transition diagram



BR_SYS power modes have $2^{\text{NUMVMBANK}}$ different operating modes, which are 1, 2, 4, 8, or 16. This is encoded as a binary value of up to four bits with each bit representing a power domain, PD_VMR*i*. Each bit, *i*, therefore indicates a specific PD_VMR*i* to be enabled or disabled. When NUMVMBANK = 0, the BR_SYS power modes only have one operating mode, which means that a PPU supporting the BR_SYS does not need to support operating modes. When a specific PD_VMR*i* is disabled, its lower power mode is always the OFF state. Other than OFF or WARM_RST states, it is only possible to move between the operating modes while in one of the ON modes.

For example, for a system with NUMVMBANK = 2

- OPMODE0: PD_VMR0 and PD_VMR1 are both disabled.
- OPMODE1: PD_VMR0 is enabled and PD_VMR1 is disabled.
- OPMODE2: PD_VMR0 is disabled and PD_VMR1 is enabled.
- OPMODE3: PD_VMR0 and PD_VMR1 are both enabled.

On Cold reset, the system transitions to the ON_OPMODE<x> where *x* is $2^{\text{NUMVMBANK}} - 1$. In this power mode, all PD_VMR*i* are enabled if they exist.

When a Warm reset is requested, the bounded region transitions to the WARM_RST through other power modes once the domain is idle and ready for reset.



The DMA includes a minimum power register in its programmers model. If the DMA is included in the system (NUMDMA > 0), it can prevent PD_SYS from transitioning to a power state that is lower than its internal minimum power register allows.

5.15.2.3.1 Controlling PD_VMR<i> power mode

To configure the required low-power state of each PD_VMR<i>, software must configure the register fields PDCM_PD_VMR<i>_SENSE.MIN_PWR_STATE as follows:

- Set to 0b00 to set the low-power state of the PD_VMR<i> to OFF. Therefore PD_VMR<i> only ever transitions between ON and OFF state, and is never in retention, meaning that in low-power state, all state in VM<i> is lost.

With this setting, when PD_SYS is ON and the BR_SYS is in one of the ON_OPMODE modes that currently has PD_VMR<i> ON, it transitions to another ON_OPMODE where PD_VMR<i> is OFF automatically when the PD_VMR<i> domain is idle. Therefore, expect to see PD_VMR<i> turn off quickly once PDCM_PD_VMR<i>_SENSE.MIN_PWR_STATE is set to 0b00.

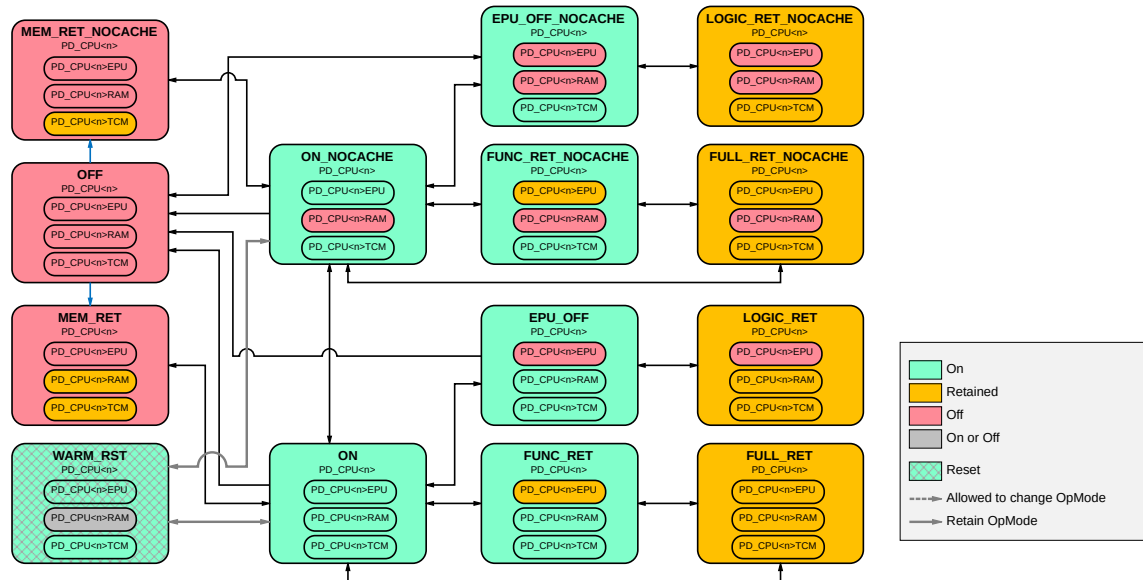
Once PD_VMR<i> is OFF, it can only be returned to ON by an access on the bus targeting PD_VMR<i>, but following that, when PD_VMR<i> is idle again, it turns off again. To avoid this, configure PDCM_PD_VMR<i>_SENSE.MIN_PWR_STATE to a nonzero value before accessing the VM<i>.

- Set to 0b01 to set the low-power state of the PD_VMR<i> to Retention. PD_VMR<i> only ever transitions between ON and Retention state, and is never OFF, meaning that in low-power state, all registers states in VM<i> are retained. Therefore, for BR_SYS, it never transitions to a power mode that has PD_VMR<i> turned off. To place the PD_VMR<i> into Retention, the BR_SYS power modes have to enter one of the FULL_RET_OPMOE modes or MEM_RET_OPMODE mode where PD_VMR<i> is in Retention. This means that it is not possible to place a PD_VMR into retention while keeping PD_SYS ON.

5.15.2.4 BR_CPU<n> power modes

The following figure shows the power modes that BR_CPU<n> supports.

Figure 5-16: BR_CPU<n> power mode transition diagram



On Cold reset, the bounded region enters the EPU_OFF_NOCACHE power mode.

The PD_CPU<n>TCM power state always matches that of PD_CPU<n>, except when in MEM_RET or MEM_RET_NOCACHE states where it is retained. The choice from ON_NOCACHE to either MEM_RET_NOCACHE or OFF is determined by the CPU's local TCM minimum power state register configuration, CPUPWRCFG.TCM_MIN_PWR_STATE.

When a Warm reset is requested, the bounded region transitions to the WARM_RST through other power modes once the domains are idle and ready for reset.

5.15.2.4.1 Controlling PD_CPU<n>RAM power states

The BR_CPU<n> bounded region uses operating modes to support the ability to turn on or off the cache RAMs in modes other than the OFF mode. Power modes with cache RAMs disabled, called the NOCACHE operating modes, are suffixed with NOCACHE. Power modes with cache RAMs enabled, called the CACHE operating modes, are the modes without the NOCACHE suffix. To select the use of the NONCACHE operating modes, the following registers must be configured:

- Set CPDLPSTATE.RLPSTATE register in the CPU to OFF which is 0b11,
- Set MSCR.DCACTIVE register in the CPU to 0b0 to disable the Data Cache,
- Set MSCR.IACTIVE register in the CPU to 0b0 to disable the Instruction Cache.

Transitions between the two operating modes can only occur between the ON and ON_NOCACHE power modes.

When in the NOCACHE operating modes, the PD_CPU<n>RAM is always turned off. When operating in the CACHE operating modes, PD_CPU<n>RAM enters RAM retention when entering MEM_RET, LOGIC_RET or FULL_RET power modes.

5.15.2.4.2 Controlling PD_CPU<n>EPU power states

Other than WARM_RST state, the BR_CPU<n> bounded region provides the ability for the PD_CPU<n>EPU to enter a lower power state independently as long as the PD_CPU<n> is ON. To control if the PD_CPU<n>EPU can remain ON or be allowed to enter the Retention (RET) or OFF state, software can use the register CPDLPSTATE.ELPSTATE in conjunction with the CPPWR.SU10 in the CPU as follows:

RET

To allow the PD_CPU<n>EPU to enter Retention (RET) state where BR_CPU<n> never enters EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE and OFF states, set either:

- CPDLPSTATE.ELPSTATE to RET, 0b10, or
- CPDLPSTATE.ELPSTATE to OFF, 0b11 and CPPWR.SU10 to 0b0.

Retention is only entered when CPU<n> is sleeping using WFI or WFE.

OFF

To allow the PD_CPU<n>EPU to enter OFF state where BR_CPU<n> never requests FUNC_RET, FUNC_RET_NOCACHE, FULL_RET_NOCACHE, and FULL_RET states set:

- CPDLPSTATE.ELPSTATE to OFF, 0b11, CPPWR.SU10 to 0b1.

ON

To keep the PD_CPU<n>EPU in ON or context retained state where BR_CPU<n> never requests EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE, FUNC_RET, FUNC_RET_NOCACHE and OFF states, set:

- CPDLPSTATE.ELPSTATE to ON, which is 0b00 or 0b01.



Note

Depending on the processor core and the system implementation, other conditions may be required to reach the desired state. For more information on these conditions see the power control of the processors in, *Arm® Cortex®-M55 Processor Technical Reference Manual*, *Arm® Cortex®-M55 Processor Integration and Implementation Manual*, and *Arm® Cortex®-M85 Processor Integration and Implementation Manual*. For more information on CPPWR, see *Arm®v8-M Architecture Reference Manual*.

5.15.2.4.3 Controlling PD_CPU<n> power states

For the PD_CPU<n> to enter a lower power state, software on the CPU<n> must first configure its CPDLPSTATE.CLPSTATE register to define what power state CPU<n> can enter when in a sleep state as follows:

- Set CPDLPSTATE.CLPSTATE to RET, to allow the PD_CPU<n> to enter retention state only when in sleep state. When set to RET, BR_CPU<n> never enters OFF, MEM_RET_NOCACHE, or MEM_RET. Other modes can be entered depending on PD_CPU<n>EPU and PD_CPU<n>RAM domain power states.
- Set CPDLPSTATE.CLPSTATE to OFF, to allow the PD_CPU<n> to enter OFF state only when in sleep state. When set to OFF, BR_CPU<n> never requests LOGIC_RET_NOCACHE and LOGIC_RET modes. Other modes can be entered depending on PD_CPU<n>EPU and PD_CPU<n>RAM domain power states, and if a coprocessor CP<i> that is included in the system is indicating that the state cannot be lost CPPWR.SU<i>=0b0.
- Set CPDLPSTATE.CLPSTATE to ON, that is, to 0b00 or 0b01 to keep PD_CPU<n> ON even when it is in sleep mode, which in turn means that BR_CPU<n> never enters LOGIC_RET_NOCACHE, LOGIC_RET, FULL_RET_NOCACHE, FULL_RET, OFF, MEM_RET_NOCACHE and MEM_RET modes. Other modes can be entered depending on PD_CPU<n>EPU and PD_CPU<n>RAM domain power states.



Note

Depending on the processor core and the system implementation, other conditions may be required to reach the desired state. For more information on these conditions, see the power control of the processors in, *Arm® Cortex®-M55 Processor Technical Reference Manual*, *Arm® Cortex®-M55 Processor Integration and Implementation Manual*, and *Arm® Cortex®-M85 Processor Integration and Implementation Manual*. For more information on CPPWR, see *Arm®v8-M Architecture Reference Manual*.

For each CPU to then enter a lower power state, the CPU must enter WFI DeepSleep state. In CRSAS Ma1, DeepSleep always uses a WIC. External WIC for each CPU always exists, and all External WICs reside in the PD_AON domain. Depending on HASCPU<n>IWIC:

- If HASCPU<n>IWIC is set to 0b0 for CPU<n>, then Internal WIC does not exist for CPU<n> and the External WIC is always used for CPU<n>.
- If HASCPU<n>IWIC is set to 0b1 for CPU<n>, then Internal WIC also exists, and the choice of which WIC to use is a software choice through CPUPWRCFG.USEIWIC. See [CPUPWRCFG](#).

Any Internal WIC that exist resides in associated PD_CPU<n> power domain.

The following shows the types of power states that each CPU in CRSAS Ma1 supports from a programmers point of view, and how to enter each.

OFF - DeepSleep state

Allows the CPU to turn off but utilizes interrupts through the external WIC to wake the CPU. To enter this state, the CPU must select to use the External WIC through the CPUPWRCFG.USEIWIC register, set the CPU<n>'s CPDLPSTATE.CLPSTATE to OFF and

enable DeepSleep, before entering WFI. If Internal WIC is selected PD_CPU<n> is not able to turn off and remains ON.

RET - DeepSleep state

Allows the CPU to enter retention state and utilizes interrupts through the external WIC to wake the CPU. To enter this state, the CPU must select to use the External WIC through the CPUPWRCFG.USEIWIC register, set CPDLPSTATE.CLPSTATE to RET and enable DeepSleep, before entering WFI. If Internal WIC is selected instead, the PD_CPU<n> is not able to enter retention and remains ON.

ON - DeepSleep state

Allows the CPU to enter a sleep state that only supports stopping the CPU clock internally, including to the NVIC. It can use either the External or Internal WIC to wake the CPU. To enter this state, the CPU must set CPDLPSTATE.CLPSTATE to 0b01, ON, enable DeepSleep before entering WFI. Selection of External or Internal WIC can be performed using the CPUPWRCFG.USEIWIC register prior to entering WFI.

ON - Sleep state

Allows the CPU to enter a sleep state that still is ON, keeping its NVIC clocking and running with the rest of the core clock turned off. To enter this state, the CPU must not enable DeepSleep, or set CPDLPSTATE.CLPSTATE to ON, 0b00, before entering WFI or WFE.

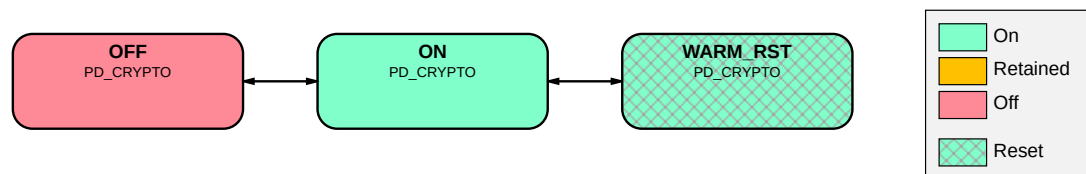
ON state

The CPU normal running state. Note that in this state, the EPU, and the RAMs in the CPU have a degree of separate control as detailed in [Controlling PD_CPU<n>RAM power states](#) and [Controlling PD_CPU<n>EPU power states](#).

5.15.2.5 BR_CRYPTO power modes

The following figure shows the power modes that BR_CRYPTO supports.

Figure 5-17: BR_CRYPTO power mode transition diagram



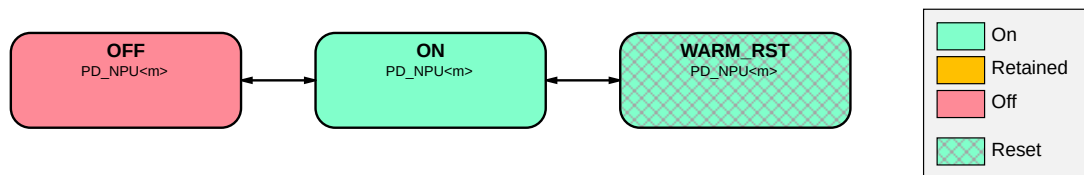
On Cold reset, the bounded domain enters the ON state.

When a Warm reset is requested, the bounded region transitions to the WARM_RST once all dependent domain is idle and ready for reset.

5.15.2.6 BR_NPU<m> power modes

The following figure shows the power modes that BR_NPU<m>.

Figure 5-18: BR_NPU power mode transition diagram



On Cold reset, the bounded domain enters the ON state.

When a Warm reset is requested, the bounded region transitions to the WARM_RST once all dependent domain is idle and ready for reset. The NPU has a power control register that can be programmed to prevent the NPU powering off. For more information of this register, see *Arm® Ethos™-U55 NPU Technical Reference Manual*. After reset, the NPU does not block power down, once it enters an idle state.

5.15.2.7 Intermediate power dependency control

The following table shows the PDCM table and it shows how the PD_SYS and PD_VMR<i> are affected by the power state of the other domains.

The heading row of the table lists the Power Domains that are being controlled, while the left column lists the Power dependency inputs. Power dependency inputs are either:

- The “ON” state of power domains in the system. For example, PD_SYS_ON means PD_SYS is ON when asserted.
- Expansion Power Dependency Control Matrix Q-Channel Signals, **PDCMONQREQn** and **PDCMRETQREQn**, that are driven by expansion logic from outside the subsystem indicating a keep-up request from external power domains.

“Conf” indicates that the power domain is software configurable while “Y” indicates that it is always sensitive to the respective dependency input. For example, PD_SYS can be software configured to be sensitive to the ON state of PD_SYS and all Expansion Power Control Dependency inputs, and it is always sensitive to the ON state of all PD_CPU<n>, PD_NPU<m> and PD_CRYPTO. If a power domain is sensitive to an ON dependency input, it means that once the power domain being controlled is already ON, if any of the dependency inputs is ON or true, then the power domain remains ON. The exception is with the **PDCMRETQREQn** inputs, where, if a power domain is configured to be sensitive to these, the power domain maintains at least in a retention state. The PDCM is used primarily to define when a power domain should not enter a lower power state. It is not designed to support powering up of any power domain.

Table 5-8: Power Dependency Matrix for PILEVEL = 1

Power Dependency Inputs/ Power Domain	PD_SYS	PD_VMR0 ¹	PD_VMR1 ¹	PD_VMR2 ¹	PD_VMR3 ¹
PD_SYS_ON	Conf	-	-	-	-
PD_CPU0_ON ²	Y	Conf	Conf	Conf	Conf
PD_CPU1_ON ²	Y	Conf	Conf	Conf	Conf
PD_CPU2_ON ²	Y	Conf	Conf	Conf	Conf
PD_CPU3_ON ²	Y	Conf	Conf	Conf	Conf
PD_NPU0_ON ³	Y	Conf	Conf	Conf	Conf
PD_NPU1_ON ³	Y	Conf	Conf	Conf	Conf
PD_NPU2_ON ³	Y	Conf	Conf	Conf	Conf
PD_NPU3_ON ³	Y	Conf	Conf	Conf	Conf
PD_CRYPTON ⁴	Y	-	-	-	-
PDCMONQREQn[k] ⁵	Conf	Conf	Conf	Conf	Conf
PDCMRETQREQn[k] ⁵	Conf	Conf	Conf	Conf	Conf

¹ PD_VMR<i> columns do not exist if $i > (\text{NUMVMBANK} - 1)$

² PD_CPU<n>_ON row does not exist if $n > \text{NUMCPU}$

³ PD_NPU<m>_ON row does not exist if $\text{NUMNPU} = 0$

⁴ PD_CRYPTON<n>_ON row does not exist if $\text{HASCRYPTO} = 0$

⁵ **PDCMONQREQn[k]** and **PDCMRETQREQn[k]**, where k is $\{0 - <\text{PDCMQCHWIDTH}-1\}$

PD_VMR<i> can also be configured to be sensitive to a power domain. For example, if PD_VMR0 sensitivity is configured by software to be sensitive to PD_CPU0_ON, and if PD_CPU0 is ON, the memory also remains ON. However, PD_VMR0 is controlled using the same PPU as PD_SYS and as a result of the power state of the PPU shown in the figure in [BR_SYS power modes](#), whenever PD_VMR0 is ON, PD_SYS also remains in one of the ON_OPMODE states where PD_VMR0 is ON.



PD_SYS can be configured to be sensitive to itself. When software configures a domain to be sensitive to self, the domain remains ON once it is ON.

The intention of the PDCM and all other sensitivity defined for each power domain is to allow, as much as possible for the power control of the System to be performed primarily using dynamic power transitions. This reduces the number of software interactions needed for system management and therefore improves its responsiveness and contributes to further power reduction.

CRSAS Ma1 also provides a programmable register for the power domains shown in the following table, which defines the lowest power mode that each domain can enter. PD_NPU<m> and PD_DEBUG defaults to a minimum power state of OFF.

Table 5-9: Power Domain's Minimum Power State for PILEVEL = 1

Power Domain	Supported MIN_PWR_STATE for each domain
PD_SYS	ON, OFF, Retention,
PD_VMR<i>	ON, OFF, Retention

The MIN_PWR_STATES of both PD_SYS and PD_VMR<i> therefore help to determine which power modes of the BR_SYS PPU can be used next when performing dynamic power transition.

For example, if the system is idle and all dependent domains are not ON, and then the BR_SYS PPU try to enter the bounded domain collectively to a low-power state, if MIN_PWR_STATE of PD_SYS is set to Retention, then BR_SYS is never allowed to enter the OFF state nor any of the MEM_RET_OPMODE<x> states because PS_SYS is not allowed to turn off. Then, depending on the current PD_VMR<i> states, it tries to enter one of the associated FULL_RET_OPMODE<x> states.

In another example, if PDCM_PD_VMR<x>_SENSE.MIN_PWR_STATE of PD_VMR0 is "ON" and all other PD_VMRs are OFF, then if currently all PD_VMR<i> are ON, when BR_SYS PPU is entering a low-power state, it transitions to ON_OPMODE1 state to turn off all PD_VMR<i> except PD_VMR0. It never enters FULL_RET_OPMODE1 nor MEM_RET_OPMODE1.

5.15.2.8 Intermediate System level Power States

Through the relationship defined by Power dependency control matrix, and the CPU minimum power states, we can define several System Power States that the system supports.

SYS_OFF

All voltage and power domains are OFF.

HIBERNATION0

The VSYS voltage domain is ON, and the system is in the lowest power state that can still be woken from sleep. At wake, the system has to reboot.

SYS_RET

The system is in retention, and at wake, the system can continue to execute since no system state is lost.

SYS_ON

The system is ON.

The following table lists the power states of each power domain that each of the System Power State supports or requires. See [Controlling PD_CPU<n> power states](#).

Table 5-10: System Level Power States when PILEVEL = 1

Power Domains	System Level Power States			
	SYS_OFF	HIBERNATION0	SYS_RET	SYS_ON
VSYS (PD_AON)	OFF	ON	ON	ON
PD_SYS	OFF	OFF	RET	ON

Power Domains	System Level Power States			
	SYS_OFF	HIBERNATIONO	SYS_RET	SYS_ON
PD_CPU<n> ¹	OFF	OFF – DeepSleep	OFF – DeepSleep RET – DeepSleep	OFF – DeepSleep RET – DeepSleep ON – DeepSleep ON – Sleep ON
PD_NPU<m> ³	OFF	OFF	OFF	OFF/ON
PD_DEBUG	OFF	OFF/ON	OFF/ON	OFF/ON
PD_CRYPTO ²	OFF	OFF	OFF	OFF/ON
PD_VMR<i> ⁴	OFF	OFF/RET	OFF/RET	OFF/RET/ON

¹ When the CPU<n> and therefore PD_SYS is in OFF-DeepSleep state, PD_VMR<i> can either be OFF or be in retention and is defined by CPU<n>PWRCFG.TCM_MIN_PWR_STATE. In this state, the EPU must be OFF. The PD_CPU<n>RAM can be OFF permanently and is defined by CPUPWRCFG.TCM_MIN_PWR_STATE.

² This power domain only exists when HASCRYPTO = 1.

³ These power domains only exist when NUMNPU > 0.

⁴ This power domain only exists if NUMVMBANK > 0.

The following points can be made based on the preceding table:

- The System Level Power States are closely associated with the PD_SYS power states, PD_CPU<n>EPU and PD_CPU<n>RAM, and PD_VMR<i> supported states are associated with PD_SYS states. See the figure in [BR_SYS power modes](#).
- Generally, PD_DEBUG can be woken independently except in the SYS_OFF state.
- Memory power states, PD_VMR, are defined as part of BR_SYS and cannot be controlled independently from PD_SYS.

An additional transient WARM_RST state also exists for the system when a Warm reset is being performed. This state can only be entered from SYS_ON state. In this state, all power domains, except PD_AON, temporarily enter WARM_RST mode and exit it when Warm reset is completed.

5.15.3 Basic level power infrastructure

The Basic Level power infrastructure, when PILEVEL = 0, locks the CPU top-level power domain to the main system to reduce the complexity of the infrastructure to reduce the complexity of the

infrastructure by locking the CPU top-level power domain to the main system. This reduces the number of power domains and hence the number of PPUUs in the system.

Therefore, this simplification removes the ability to separately control the power state of the processor, the system, and volatile memories in the system. In addition, such a system can only support a single processor and therefore NUMCPU must be '0' when PILEVEL = 0.

5.15.3.1 Power hierarchy

The system at PILEVEL = 0 supports only one voltage domain:

VSYS

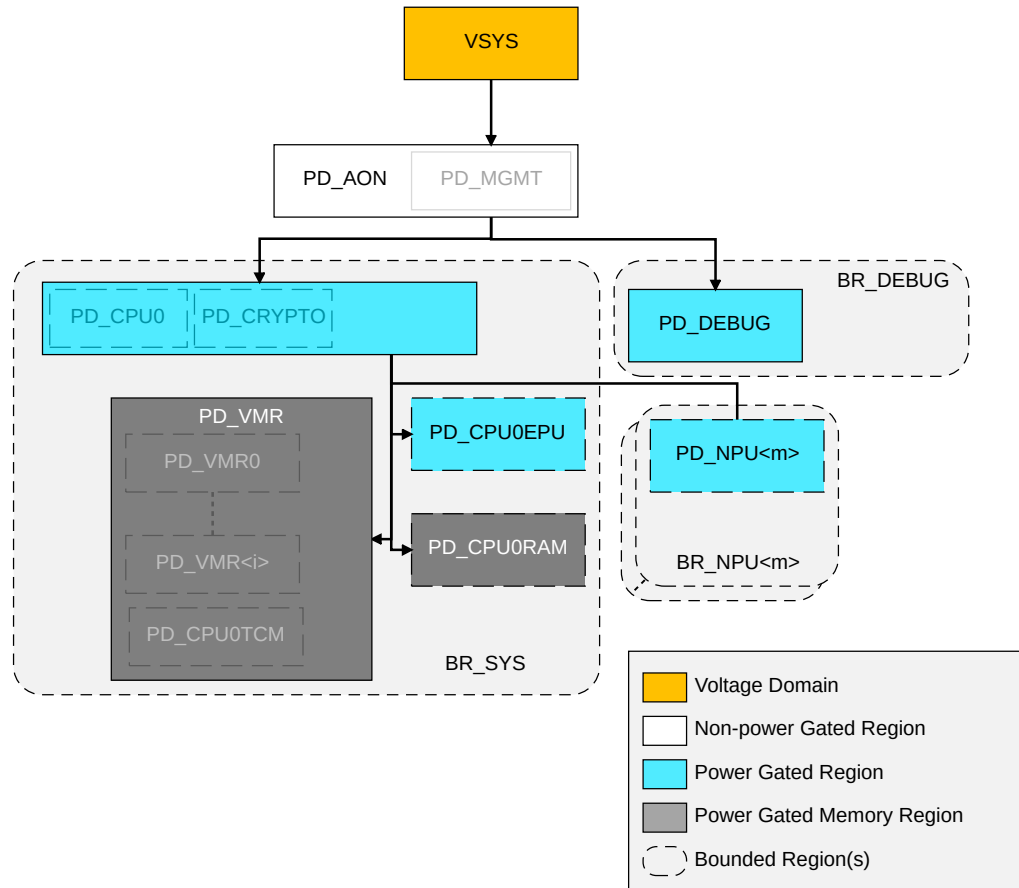
The system voltage domain.

The system at PILEVEL = 0 supports several power domains:

- PA_AON
- PD_DEBUG
- PD_SYS
- PD_NPU<*m*>, does not exist if NUMNPU = 0
- PD_VMR<*i*>, where *i* is 0 to NUMVMBANK-1, and does not exist if NUMVMBANK = 0
- PD_CPU0EPU
- PD_CPU0RAM

These power domains are hierarchical and the figure shows the voltage and power domain hierarchy.

Figure 5-19: Voltage and Power domain hierarchy of a subsystem with PILEVEL = 0



The figure also shows several Bounded Regions. Each of these Bounded Regions is controlled by a single PPU, and these are:

BR_DEBUG

This is controlled by a single PPU, DEBUG_PPU.

The power domain in BR_DEBUG is:

- PD_DEBUG. This is a combined power gated logic and memory domain for all debug logic in the system.

BR_NPU<m>

Each BR_NPU bounded region is controlled by a single PPU, NPU<m>_PPU.

The power domain in BR_NPU<m> is:

- PD_NPU<m>

BR_SYS

This is controlled by a single PPU, SYS_PPU.

The power domains in BR_SYS are:

- PD_SYS. Note that what was PD_CRYPT0 is now merged into PD_SYS.
- PD_CPU0EPU
- PD_CPU0RAM
- PD_VMR which contains all the volatile memory instances.

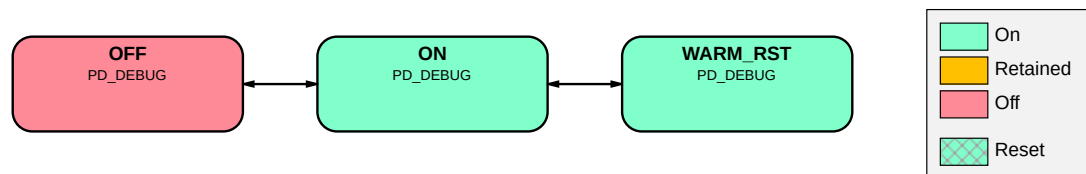
What was PD_CPU in PILEVEL = 1 is now merged with PD_SYS and the PD_SYS domain is now controlled by the PPU that used to control PD_CPU power domain. In addition, what was PD_CPU0TCM power domain is merged with all PD_VMR<*i*>, where *i* is 0 to NUMVMBANK – 1, to form a single PD_VMR domain, controlled using the same PPU as if it were the TCM memory in PILEVEL = 1.

Because of this, other than PD_DEBUG, all other power control depends on a single PPU to reduce complexity and area, but at a cost to flexibility.

5.15.3.2 BR_DEBUG power modes

The following figure shows the power modes that BR_DEBUG supports.

Figure 5-20: BR_DEBUG power mode transition diagram

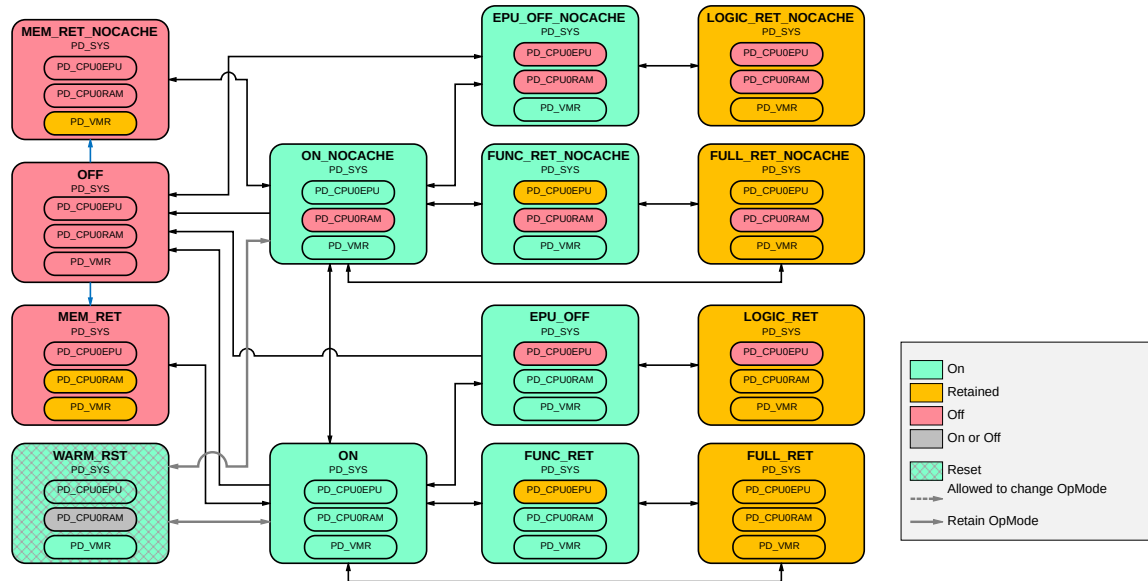


On Cold reset, PD_DEBUG automatically transition to ON state, and enters OFF state eventually if the debug system is not required to stay ON. It is woken either through the PD_DEBUG's Power Control Wakeup Q-Channel Device Interface interface or through a bus access targeting its shared debug domain. The WARM_RST state is entered to bring the debug domain into an idle state so that the main system can be Warm reset cleanly. The WARM_RST state is entered even though the debug domain is not itself being Warm reset at the same time.

5.15.3.3 BR_SYS power modes

The following figure shows the power modes that BR_SYS supports. Because PD_SYS is merged with PD_CPU0 when PILEVEL = 0, BR_SYS has a power mode transition diagram like BR_CPU<n> when PILEVEL = 1.

Figure 5-21: BR_SYS power mode transition diagram



On Cold reset, the bounded region enters EPU_OFF_NOCACHE power mode.

The PD_VMR power state always matches that of PD_SYS, except when in MEM_RET or MEM_RET_NOCACHE state where it is retained. The choice from ON_NOCACHE to either MEM_RET_NOCACHE or OFF is determined by the CPU 0 local TCM minimum power state register configuration CPUPWRCFG.TCM_MIN_PWR_STATE. Therefore at PILEVEL = 0, Volatile memory bank are treated like TCMs for power control.

When a Warm reset is requested, the bounded region transitions to the WARM_RST through other power modes once the domains are idle and ready for reset. The DMA includes a minimum power register in its programmers' model. If the DMA is included in the system (NUMDMA > 0) then it can prevent PD_SYS from transitioning to a power state that is lower than its internal minimum power register allows.

5.15.3.3.1 Controlling PD_CPU0RAM power states

The BR_SYS bounded region uses operating modes to support the ability to turn on or off the cache RAMs in modes other than the OFF mode.

- Power modes with cache RAMs disabled, called the NOCACHE operating modes, are suffixed with NOCACHE.
- Power modes with cache RAMs enabled, called the CACHE operating modes, are the modes without the NOCACHE suffix.

To select the use of the NONCACHE operating modes, the following registers must be configured:

- Set CPDLPSTATE.RLPSTATE register in the CPU to OFF which is 0b11,
- Set MSCR.DCACTIVE register in the CPU to 0b0 to disable Data Cache,
- Set MSCR.ICACTIVE register in the CPU set to 0b0 to disable Instruction Cache.

Transitions between the two operating modes can only occur between the ON and ON_NOCACHE power modes.

When in the NOCACHE operating modes, the PD_CPU0RAM is always turned off. When operating in the CACHE operating modes, PD_CPU0RAM enters RAM retention when entering MEM_RET, LOGIC_RET or FULL_RET power modes.

5.15.3.3.2 Controlling PD_CPU0EPU power states

Other than WARM_RST state, the BR_SYS bounded region provides the ability for the PD_CPU0EPU to enter a lower power state independently as long as the PD_SYS is ON. To control if the PD_CPU0EPU can remain ON or be allowed to enter the Retention (RET) or OFF state, software can use the register CPDLPSTATE.ELPSTATE in conjunction with the CPPWR.SU10 in the CPU as follows:

RET

To allow the PD_CPU0EPU to enter Retention (RET) state where BR_SYS never enters EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE and OFF states, set either:

- CPDLPSTATE.ELPSTATE to RET, 0b10, or
- CPDLPSTATE.ELPSTATE to OFF, 0b11 and CPPWR.SU10 to 0b0.

Retention is only entered when CPU<n> is sleeping using WFI or WFE.

OFF

To allow the PD_CPU0EPU to enter OFF state where BR_SYS never requests FUNC_RET, FUNC_RET_NOCACHE, FULL_RET_NOCACHE, and FULL_RET states set:

- CPDLPSTATE.ELPSTATE to OFF, 0b11, CPPWR.SU10 to 0b1.

ON

To keep the PD_CPU0EPU in ON or context retained state where BR_SYS never requests EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE, FUNC_RET, FUNC_RET_NOCACHE and OFF states, set:

- CPDLPSTATE.ELPSTATE to ON, which is 0b00 or 0b01.



Depending on the processor core and the system implementation, other conditions may be required to reach the desired state. For more information on these conditions see the power control of the processors in, *Arm® Cortex®-M55 Processor Technical Reference Manual*, *Arm® Cortex®-M55 Processor Integration and Implementation Manual*, and *Arm® Cortex®-M85 Processor Integration and Implementation Manual*. For more information on CPPWR, see *Arm®v8-M Architecture Reference Manual*.

5.15.3.3.3 Controlling PD_SYS Power States

Other than WARM_RST state, the BR_SYS bounded region provides the ability for the PD_CPU0EPU to enter a lower power state independently as long as the PD_SYS is ON. To control if the PD_CPU0EPU can remain ON or be allowed to enter the Retention (RET) or OFF state, software can use the register CPDLPSTATE.ELPSTATE in conjunction with the CPPWR.SU10 in the CPU as follows:

RET

To allow the PD_CPU0EPU to enter Retention (RET) state where BR_SYS never enters EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE and OFF states, set either:

- CPDLPSTATE.ELPSTATE to RET, 0b10, or
- CPDLPSTATE.ELPSTATE to OFF, 0b11 and CPPWR.SU10 to 0b0.

Retention is only entered when CPU<n> is sleeping using WFI or WFE.

OFF

To allow the PD_CPU0EPU to enter OFF state where BR_SYS never requests FUNC_RET, FUNC_RET_NOCACHE, FULL_RET_NOCACHE, and FULL_RET states set:

- CPDLPSTATE.ELPSTATE to OFF, 0b11, CPPWR.SU10 to 0b1.

ON

To keep the PD_CPU0EPU in ON or context retained state where BR_SYS never requests EPU_OFF, LOGIC_RET EPU_OFF_NOCACHE, LOGIC_RET_NOCACHE, MEM_RET, MEM_RET_NOCACHE, FUNC_RET, FUNC_RET_NOCACHE and OFF states, set:

- CPDLPSTATE.ELPSTATE to ON, which is 0b00 or 0b01.



Note

Depending on the processor core and the system implementation, other conditions may be required to reach the desired state. For more information on these conditions see the power control of the processors in, *Arm® Cortex®-M55 Processor Technical Reference Manual*, *Arm® Cortex®-M55 Processor Integration and Implementation Manual*, and *Arm® Cortex®-M85 Processor Integration and Implementation Manual*. For more information on CPPWR, see *Arm®v8-M Architecture Reference Manual*.

For the system to then enter a lower power state, the CPU must enter WFI DeepSleep state. In CRSAS Ma1, DeepSleep always uses a WIC. EWIC of the CPU always exists and resides in the PD_AON domain. Depending on HASCPU0IWIC:

- If HASCPU0IWIC is set to 0b0, then Internal WIC does not exist and the EWIC is always used for CPU<n>.
- If HASCPU0IWIC is set to 0b1, then Internal WIC also exists, and the choice of which WIC to use is a software choice through CPUPWRCFG.USEIWIC. See [CPUPWRCFG](#).

Any Internal WIC that exists resides in the PD_SYS power domain.

The following shows the types of power states that the system and therefore the CPU in CRSAS Ma1 supports from a programmer's point of view, and how to enter each.

OFF - DeepSleep state

Allows PD_SYS to turn off but utilizes interrupts through the external WIC to wake the system. To enter this state, the CPU must select to use the External WIC through the CPUPWRCFG.USEIWIC register, set the CPU0's CPDLPSTATE.CLPSTATE to OFF and enable DeepSleep, before entering WFI. Note that if Internal WIC is selected PD_CPU<n> is not able to turn off and remains ON.

RET - DeepSleep state

Allows PD_SYS to enter retention state and utilizes interrupts through the external WIC to wake the system. To enter this state, the CPU must select to use the External WIC through the CPUPWRCFG.USEIWIC register, set CPDLPSTATE.CLPSTATE to RET and enable DeepSleep, before entering WFI. Note that if Internal WIC is selected instead, the PD_SYS is not able to enter retention and remains ON.

ON - DeepSleep state

Allows the PD_SYS to stay ON, with the CPU in a sleep that is only stopping the CPU clock internally, including to the NVIC. It can use either the External or Internal WIC to wake the CPU. To enter this state, the CPU must set CPDLPSTATE.CLPSTATE to 0b01, ON, enable DeepSleep before entering WFI. Selection of External or Internal WIC can be performed using the CPUPWRCFG.USEIWIC register prior to entering WFI.

ON - Sleep state

Allows the PD_SYS to stay ON, with the CPU in a sleep state that keeps its NVIC clocking and running, but with the rest of the CPU core clock turned off. To enter this state, the CPU must not enable DeepSleep or set CPDLPSTATE.CLPSTATE to ON, 0b00 before entering WFI or WFE.

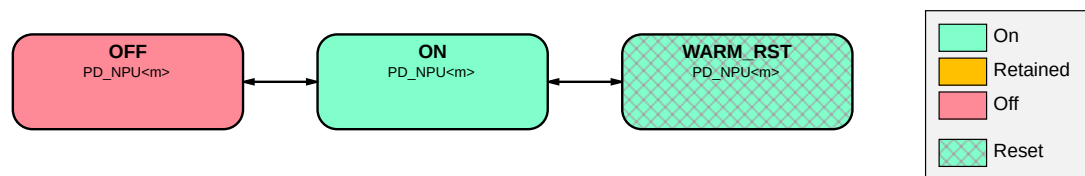
ON state

The PD_SYS and the CPU's normal running state. In this state, the EPU and the RAMs in the CPU have a degree of separate control as detailed in the previous sections.

5.15.3.4 BR_NPU<m> power modes

The following figure shows the power modes that BR_NPU<m>.

Figure 5-22: BR_NPU power mode transition diagram



On Cold reset, the bounded domain enters the ON state.

When a Warm reset is requested, the bounded region transitions to the WARM_RST once all dependent domain is idle and ready for reset. The NPU has a power control register that can be programmed to prevent the NPU powering off. For more information of this register, see *Arm® Ethos™-U55 NPU Technical Reference Manual*. After reset, the NPU does not block power down, once it enters an idle state.

5.15.3.5 Basic power dependency control

The following table shows the PDCM table and it shows how the PD_SYS is affected by the power state of the other domains.

The table only includes dependency controls for only a single power domain, PD_SYS, in the top row. The left column lists the Power dependency inputs. Power dependency inputs are either:

- The “ON” state of power domains in the system. For example, PD_SYS_ON means PD_SYS is ON when asserted.
- Expansion Power Dependency Control Matrix Q-Channel signals, **PDCMONQREQn** and **PDCMRETQREQn**, that are driven by expansion logic from outside the subsystem indicating a keep-up request from external power domains.

“Conf” indicates that it is software configurable to be sensitive to the respective dependency input. For example, PD_SYS can be software configured to be sensitive to the ON state of itself and all Expansion Power Control Dependency inputs. If a power domain is sensitive to an ON dependency input, it means that once the power domain being controlled is already ON, if any of the dependency inputs is ON or true, then the power domain remains ON. The exception is with

the **PDCMRETQREQn** inputs, where, if a power domain is configured to be sensitive to these, the power domain maintains at least in a retention state. The PDCM is used primarily to define when a power domain should not enter a lower power state. It is not designed to support powering up of any power domain.

Table 5-11: Power Dependency Matrix, for PILEVEL = 0

Power Dependency Inputs/ Power Domain	PD_SYS
PD_SYS_ON	Conf
PD_NPU<m> ¹	Y
PDCMONQREQn[k] ²	Conf
PDCMRETQREQn[k] ²	Conf

¹ PD_NPU<m>_ON row does not exist if NUMNPU = 0

² **PDCMONQREQn[k]** and **PDCMRETQREQn[k]**, where k is {0- <PDCMQCHWIDTH-1>}

Since PD_SYS can be configured to be sensitive to itself, when set up by software to do so, PD_SYS remains ON once it is ON.

The intention of the PDCM and all other sensitivity defined for each power domain is to allow, as much as possible for the power control of the System to be performed primarily using dynamic power transitions. This reduces the number of software interactions needed for system management and therefore improves its responsiveness and contributes to further power reduction.

At PILEVEL = 0, the desired power state of the system is defined in part of the CPU internal controls and states, and the choice between IWIC or EWIC. See [Controlling PD_SYS power states](#).

5.15.3.6 Basic System Level Power States

Through the relationship defined by Power dependency control matrix, and the CPU minimum power states, we can define several System Power States that the system supports, and these are:

SYS_OFF

All voltage and power domains are OFF.

HIBERNATION0

The VSYS voltage domain is ON, and the system is in the lowest power state that can still be woken from sleep. At wake, the system has to reboot.

SYS_RET

In this state, the system is in retention, and at wake, the system can continue to execute since no system state is lost.

SYS_ON

In this state, the system is ON.

The following lists the power states of each power domain that each of the System Power State supports or requires. See [Controlling PD_CPU<n> power states](#).

Table 5-12: System Level Power States when PILEVEL = 0

Power Domains	System Level Power States			
	SYS_OFF	HIBERNATION0	SYS_RET	SYS_ON
VSYS (PD_AON)	OFF	ON	ON	ON
PD_SYS ¹	OFF	OFF – DeepSleep	RET – DeepSleep	ON – DeepSleep ON – Sleep ON
PD_NPU<m> ²	OFF	OFF	OFF	OFF/ON
PD_DEBUG	OFF	OFF/ON	OFF/ON	OFF/ON

¹ When the CPU and therefore PD_SYS is in OFF-DeepSleep state, PD_VMR can either be OFF or be in retention and is defined by CPUPWRCFG.TCM_MIN_PWR_STATE. In this state, the EPU must be OFF. The PD_CPU0RAM can be OFF permanently and is defined by CPDLPSTATE.RLPSTATE.

² These power domains only exist when NUMNPU > 0.

The following points can be made based on the preceding table:

- The System Level Power States are closely associated with the PD_SYS power states. PD_CPU0EPU and PD_CPU0RAM and PD_VMR supported states are associated with PD_SYS states. See [BR_SYS power modes](#).
- Generally, PD_DEBUG can be woken independently except in the SYS_OFF state.
- Memory power states, PD_VMR are defined as part of BR_SYS and cannot be controlled independently from PD_SYS.

An additional transient WARM_RST state also exists for the system when a Warm reset is being performed. This state can only be entered from SYS_ON state. In this state, all power domains, except PD_AON temporarily enter WARM_RST mode and exit it when Warm reset is completed.

5.16 DMA

The CRSAS Ma1 supports an optional DMA-350.

- **NUMDMA = 0**, the DMA IP and its associated integration logic does not exist within the system.
- **NUMDMA = 1** the DMA IP and its associated integration logic does exist within the system.

When NUMDMA = 1 the DMA can support any static configuration except that the following that must be adhered to:

Table 5-13: Non-supported static configurations

Configuration	Value for DMA	Description
AXI5_M1_PRESENT	1	Specifies whether an additional manager port is present. When set the m1 manager port is present <ul style="list-style-type: none"> 0: Interface not included 1: Interface included
SECEXT_PRESENT	1	Specifies where TrustZone security is supported <ul style="list-style-type: none"> 0: Security Extensions are not implemented 1: Security Extensions are implemented
NUM_CHANNELS	$\geq (\text{NUMCPU} + 1) * 2$	Specifies the number of DMA channels

We recommend that DMA channels are not shared between security and privilege levels and that each security and privilege level that requires DMA support has a dedicated channel configured in the DMA.

The DMA can access the system memory map as follows:

- M0 Interface: All of the system memory except for the CPU<n> S-AHB Instruction TCM and CPU<n> S-AHB Data TCM areas
- M1 Interface: Only CPU<n> S-AHB Instruction TCM and CPU<n> S-AHB Data TCM areas

This memory mapping is enforced using Manager Security Controllers (MSC), IDAUs and MPCs for Secure and Non-secure world separation of memories. And MSC, IDAU and Peripheral Protection Controllers (PPC) for privileged and unprivileged separation and Secure and Non-secure world separation of peripherals.



There is no protection provided for privileged or unprivileged memory access. Therefore, we recommend that only privileged software is allowed to program the DMA.

The DMA can be configured with 0-32 trigger in and trigger out interfaces. For more information on the DMA configuration, see *Arm® CoreLink™ DMA-350 Controller Technical Reference Manual*. Other configuration parameters may also create DMA Interfaces, such as GPO or Stream interfaces, and so on. These other interfaces are routed directly to the top level of the subsystem and are in the same power and clock domain as the DMA.

5.17 NPU

The CRSAS Ma1 supports a configurable number of Ethos-U55 NPU cores. If implemented, each NPU can support a different static configuration except that the following that must be adhered to:

Configuration	Value for NPU<m>	Description
CUSTOM_DMA_PRESENT	0	It specifies whether the NPU includes an interface for a custom DMA: <ul style="list-style-type: none"> 0: Interface not included 1: Interface included

The Ethos-U55 NPU does not support HW debug control. We recommend that the debug SW triggers a routine that controls the NPU when using HW flow control of the CPU and rest of system for debugging.

The NPU can access the system memory map as follows:

- M0 Interface : All of the system memory except for the CPU<n> S-AHB Instruction TCM and CPU<n> S-AHB Data TCM areas
- M1 Interface : All of the system memory except for the CPU<n> S-AHB Instruction TCM and CPU<n> S-AHB Data TCM areas

This memory mapping is enforced using Manager Security Controllers (MSC), IDAUs and MPCs for Secure and Non-secure world separation of memories. And MSC, IDAU and Peripheral Protection Controllers (PPC) for privileged and unprivileged separation and Secure and Non-secure world separation of peripherals.



There is no protection provided for privileged or unprivileged memory access. Therefore, we recommend that the NPU should only be made available to non-privilege software, if the risk of non-privilege software accessing privilege memory is acceptable.

For more information on Ethos-U55, see *Arm® Ethos™-U55 NPU Technical Reference Manual*.

5.17.1 NPU security mapping

The NPU is capable of operating in different security modes. To change from operating in one security world to another, a reset is required.

When the NPU reset is released by the PPU controlling the NPU power domain PD_NPU<m>, it samples the signals driven by:

- **NPUSPPORSL.SP_NPU<m>PORSL** to determine its default security level.
- **NPUSPPORPL.SP_NPU<m>PORPL** or **NPUNSPORPL.NS_NPU<m>PORPL** to determine the NPU's default privilege level.

See [NPUSPPORPL](#) and [NPUNSPORPL](#). These registers are also used to control the PPC, which provides access protection to the NPU's configuration interface.



The PPC protection setting takes effect as soon as the security and privileged registers are updated.

The following sequences are recommended when transitioning the NPU to a new security or privilege level.

Changing security level from Secure (S) to Non-secure (NS)

The following sequence details the steps for the Secure privilege software (SW) to change security level:

1. Read the current security from the corresponding System register: `initial_security=NPUSPPORSL.SP_NPU<m>PORSL`
2. If `initial_security=S` then SW reads the privilege to be retained in the target Security state from the corresponding System register: `initial_privilege=NPUNSPORPL.NS_NPU<m>PORPL` else skip remaining sequence.
3. Write on the current security alias of the NPU (S) the configuration interface, targeting the `CMD.power_q_enable` register field of the NPU to prevent the NPU powering OFF (0 = Power OFF denied) while preserving the rest of the CMD register
4. Write on the current security alias of the NPU (S) the configuration interface, targeting the `RESET.pending_CSL` register field of the NPU to set the new security level (1= Non-secure) while preserving `RESET.pending_CPL=initial_privilege`
5. Read repeatedly (poll) on the current security alias of the NPU (S) the configuration interface, targeting the `STATUS` register of the NPU until the field `reset_status` no longer returns the value 1
6. Write the new security level into `NPUSPPORSL.SP_NPU<m>PORSL` register field (1 = Non-secure)
7. Write on the current security alias of the NPU (NS) the configuration interface, targeting the `CMD.power_q_enable` register field of the NPU to allow the NPU to speculatively power OFF (1 = Power OFF allowed) while preserving the rest of the CMD register

Changing security level from Non-secure (NS) to Secure (S)

The following sequence details the steps for the Non-secure privilege software to change security level to Secure:

1. Read the current security from the corresponding System register: `initial_security=NPUSPPORSL.SP_NPU<m>PORSL`
2. If `initial_security=NS` then SW reads the privilege to be retained in the target Security state from the corresponding System register: `initial_privilege=NPUSPPORPL.SP_NPU<m>PORPL` else skip remaining sequence.
3. Write on the current security alias of the NPU (NS) the configuration interface, targeting the `CMD.power_q_enable` register field of the NPU to prevent the NPU powering OFF (0 = Power OFF denied) while preserving the rest of the CMD register
4. Write the new security level into `NPUSPPORSL.SP_NPU<m>PORSL` register field (0 = Secure)

5. Write on the current security alias of the NPU (S) the configuration interface, targeting the RESET.pending_CSL register field of the NPU to set the new security level (0=Secure) while preserving RESET.pending_CPL=initial_privilege
6. Read repeatedly (poll) on the current security alias of the NPU (S) the configuration interface, targeting the STATUS register of the NPU until the field reset_status no longer returns the value 1
7. Write on the current security alias of the NPU (S) the configuration interface, targeting the CMD.power_q_enable register field of the NPU to the allow the NPU to speculatively power OFF (1 = Power OFF allowed) while preserving the rest of the CMD register

Changing privilege level

Secure or Non-secure privilege software can change the privilege state of the NPU, as long as the privilege state being moved to is equal or lower than the software enacting the change and the security level of the software is at least matching the current security level (initial_security) of the NPU.

The following sequence details the steps for the privilege software to change privilege level:

1. Read the current privilege level from the corresponding System register:

Current security level = Non-secure

initial_privilege=NPUNSPORPL.NS_NPU<m>PORPL

Current security level = Secure

initial_privilege=NPUSPPORPL.SP_NPU<m>PORPL

2. If initial_privilege is the desired one, then skip remaining sequence.
3. Write on the current security alias of the NPU (initial_security) the configuration interface, targeting the CMD.power_q_enable register field of the NPU to the prevent the NPU powering OFF (0 = Power OFF denied) while preserving the rest of the CMD register
4. Write to the current security alias of the NPU (initial_security) the configuration interface, targeting the RESET.pending_CPL register field of the NPU to set the new privilege level (0=User; 1=privileged) while preserving RESET.pending_CSL=initial_security
5. Read repeatedly (poll) from the current security alias of the NPU (initial_security) the configuration interface, targeting the STATUS register of the NPU until the field reset_status no longer returns the value 1
6. Write to the alias corresponding to the current security level of the NPU (initial_security) the new privilege level into

Current security level = Non-secure

NPUNSPORPL.NS_NPU<m>PORPL

Current security level = Secure

NPUSPPORPL.SP_NPU<m>PORPL

7. Write on the current security alias of the NPU (initial_security) the configuration interface, targeting the CMD.power_q_enable register field of the NPU to allow the NPU to speculatively power OFF (1 = Power OFF allowed) while preserving the rest of the CMD register

6 Programmers model

This section describes the functions and programmers model of CRSAS Ma1.

6.1 System Memory Map overview

The High Level System Address Map table shows the high-level view of the memory map defined by CRSAS Ma1. This memory map is divided into Secure and Non-secure regions. The memory alternates between Secure and Non-secure regions on 256Mbyte regions, with only a few address areas exempted from security mapping because they are related to debug functionality.

To provide memory blocks and peripherals that can be mapped either as Secure or Non-secure using software, several address regions are aliased as shown in the table. Software can then choose to allocate each memory block or peripheral as Secure or Non-secure using protection controllers. The Implementation Defined Attribution Unit (IDAU) Region Values columns in the table specifies each area's security along with its ID and each region's Non-secure Callable (NSC) settings.

The following occur except when specifically stated:

- All accesses to unmapped regions of the memory result in bus-error response.
- When accessing unmapped address space within a mapped region taken by a peripheral, the access results in Read-As-Zero and Write-Ignored (**RAZ/WI**) except when specifically stated otherwise.
- Any accesses that result in security violations are either **RAZ/WI** or return a bus error response as defined by the **SECRESPCFG** register setting.

Some regions of memory map are reserved to maintain compatibility with past and future subsystems. Other areas are mapped to Expansion interfaces.

All accesses targeting populated Volatile Memory regions within 0x2100_0000 to 0x21FF_FFFF and 0x3100_0000 to 0x31FF_FFFF support exclusive access since they implement exclusive access monitoring, provided the accesses are from:

- The CPUs,
- Expansion managers through the Subordinate Main Expansion Interfaces and Subordinate Peripheral Expansion Interfaces.

Exclusive access is not supported for other regions implemented within the subsystem. For regions that reside in user expansion areas, exclusive access support is defined by the user expansion logic. If an exclusive access tries to access a region that does not support exclusive accesses, these accesses are not monitored for exclusive access and might still update their target memory locations regardless of their associated exclusive responses.

6.1.1 High Level System Address Map

Security values do not define privileged or unprivileged accessibility. These are defined by the PPC, or by the register blocks that is mapped to each area. See lower-level details of each area for details.

The System Address Map defines the following values for accessibility:

S

Secure Access

NS

Non-secure

NSC

Non-Secure Callable



1. The NSC values are defined through registers in the Secure Access Configuration registers.
2. IF HASCRYPTO = 0, this region is reserved and responds with bus error.

Table 6-1: High Level System Address Map

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
1	0x00000000 - 0x00FFFFFF	16MB	ITCM	5	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	CPU Instruction TCM. See CPU TCM memories .
2	0x01000000 - 0x09FFFFFF	192MB	Code Expansion	6	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	Manager Code Main Expansion Interface. See Main Interconnect Expansion Interfaces .
2.1	0x0A000000 - 0x0AFFFFFF	16MB	CPU0 ITCM	6.1	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	See TCM subordinate interface .
2.2	0x0B000000 - 0x0BFFFFFF	16MB	CPU1 ITCM	6.2	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	See TCM subordinate interface .
2.3	0x0C000000 - 0x0CFFFFFF	16MB	CPU2 ITCM	6.3	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	See TCM subordinate interface .

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
2.4	0x0D000000 - 0x0DFFFFFF	16MB	CPU3 ITCM	6.4	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	See TCM subordinate interface .
3	0x0E000000 - 0x0E001FFF	8KB	CryptoCell NVM Code	7	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	CryptoCell Code Access to NVM. See Has-Crypto configuration .
4	0x0E002000 - 0x0FFFFFFF	-	Reserved	-	<ul style="list-style-type: none"> Security: NS IDAUID: 0 NSC: 0 	Reserved
5	0x10000000 - 0x10FFFFFF	16MB	ITCM	1	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	CPU Instruction TCM. See CPU TCM memories .
6	0x11000000 - 0x19FFFFFF	192MB	Code Expansion	2	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	Manager Code Main Expansion Interface. See Main Interconnect Expansion Interfaces .
6.1	0x1A000000 - 0x1AFFFFFF	16MB	CPU0 ITCM	2.1	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	See TCM subordinate interface .
6.2	0x1B000000 - 0x1BFFFFFF	16MB	CPU1 ITCM	2.2	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	See TCM subordinate interface .
6.3	0x1C000000 - 0x1CFFFFFF	16MB	CPU2 ITCM	2.3	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	See TCM subordinate interface .
6.4	0x1D000000 - 0x1DFFFFFF	16MB	CPU3 ITCM	2.4	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	See TCM subordinate interface .
7	0x1E000000 - 0x1E001FFF	8KB	CryptoCell NVM Code	3	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	CryptoCell Code Access to NVM. See Has-Crypto configuration .

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
8	0x1E002000 - 0x1FFFFFFF	-	Reserved	-	<ul style="list-style-type: none"> Security: S IDAUID: 1 NSC: CODENSC 	Reserved
9	0x20000000 - 0x20FFFFFF	16MB	DTCM	13	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	CPU Data TCM. See CPU TCM memories .
10	0x21000000 - 0x21FFFFFF	16MB	Volatile Memory	14	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	Volatile Memory See Volatile Memory Region .
11	0x22000000 - 0x23FFFFFF	32MB	Reserved	-	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	Reserved
11.1	0x24000000 - 0x24FFFFFF	16MB	CPU0 DTCM	15.1	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	See TCM subordinate interface .
11.2	0x25000000 - 0x25FFFFFF	16MB	CPU1 DTCM	15.2	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	See TCM subordinate interface .
11.3	0x26000000 - 0x26FFFFFF	16MB	CPU2 DTCM	15.3	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	See TCM subordinate interface .
11.4	0x27000000 - 0x27FFFFFF	16MB	CPU3 DTCM	15.4	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	See TCM subordinate interface .
12	0x28000000 - 0x2FFFFFFF	128MB	Main Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: 2 NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
13	0x30000000 - 0x30FFFFFF	16MB	DTCM	9	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	CPU Data TCM. See CPU TCM memories .

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
14	0x31000000 - 0x31FFFFFF	16MB	Volatile memory	10	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	Internal Multi-bank Volatile Memory. See Volatile Memory Region .
15	0x32000000 - 0x33FFFFFF	32MB	Reserved	-	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	Reserved
15.1	0x34000000 - 0x34FFFFFF	16MB	CPU0 DTCM	11.1	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	See TCM subordinate interface .
15.2	0x35000000 - 0x35FFFFFF	16MB	CPU1 DTCM	11.2	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	See TCM subordinate interface .
15.3	0x36000000 - 0x36FFFFFF	16MB	CPU2 DTCM	11.3	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	See TCM subordinate interface .
15.4	0x37000000 - 0x37FFFFFF	16MB	CPU3 DTCM	11.4	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	See TCM subordinate interface .
16	0x38000000 - 0x3FFFFFFF	128MB	Main Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: 3 NSC: RAMNSC 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
17	0x40000000 - 0x4000FFFF	64KB	Peripherals	27	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	Peripheral Region. See Peripheral Region .
18	0x40010000 - 0x4001FFFF	64KB	Private CPU	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	CPU Private Peripheral Region. See CPU Private Region .
19	0x40020000 - 0x4003FFFF	128KB	System Control Peripheral Region	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	System Control Peripheral Region. See System Control Peripheral Region .

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
20	0x40040000 - 0x400FFFFFF	768KB	Peripherals	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	Peripheral Region. See Peripheral Region .
21	0x40100000 - 0x47FFFFFF	127MB	Peripheral Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	Manager Peripheral Expansion Interface. See Main Interconnect Expansion Interfaces .
22	0x48000000 - 0x4800FFFF	64KB	Peripherals	32	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	Peripheral Region. See Peripheral Region .
23	0x48010000 - 0x4801FFFF	64KB	Private CPU	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	CPU Private Peripheral Region. See CPU Private Region .
24	0x48020000 - 0x4803FFFF	128KB	System Control	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	System Control Peripheral Region. See System Control Peripheral Region .
25	0x48040000 - 0x480FFFFFF	768KB	Peripherals	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	Peripheral Region. See Peripheral Region .
26	0x48100000 - 0x4FFFFFFF	127MB	Peripheral Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: 4 NSC: 0 	Manager Peripheral Expansion Interface. See Peripheral Interconnect Expansion Interfaces .
27	0x50000000 - 0x5000FFFF	64KB	Peripherals	17	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Peripheral Region. See Peripheral Region .
28	0x50010000 - 0x5001FFFF	64KB	Private CPU	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	CPU Private Peripheral Region. See CPU Private Region .
29	0x50020000 - 0x5003FFFF	128KB	System Control	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	System Control Peripheral Region. See System Control Peripheral Region .
30	0x50040000 - 0x500FFFFFF	786KB	Peripherals	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Peripheral Region. See Peripheral Region .

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
31	0x50100000 - 0x57FFFFFF	127MB	Peripheral Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Manager Peripheral Expansion Interface. See Peripheral Interconnect Expansion Interfaces
32	0x58000000 - 0x5800FFFF	64KB	Peripherals	22	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Peripheral Region. See Peripheral Region .
33	0x58010000 - 0x5801FFFF	64KB	Private CPU	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Processor Private Peripheral Region. See CPU Private Region .
34	0x58020000 - 0x5803FFFF	128KB	System Control	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	System Control Peripheral Region. See System Control Peripheral Region .
35	0x58040000 - 0x580FFFFF	768KB	Peripherals	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Peripheral Region. See Peripheral Region .
36	0x58100000 - 0x5FFFFFFF	127MB	Peripheral Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: 5 NSC: 0 	Manager Peripheral Expansion Interface. See Peripheral Interconnect Expansion Interfaces .
37	0x60000000 - 0x6FFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: 6 NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
38	0x70000000 - 0x7FFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: 7 NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
39	0x80000000 - 0x8FFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: 8 NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
40	0x90000000 - 0x9FFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: 9 NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
41	0xA0000000 - 0xAFFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: A NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
42	0xB0000000 - 0xBFFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: B NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .

Row ID	Address	Size	Region	Alias	IDAU Region Values	Description
43	0xC0000000 - 0xCFFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: C NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
44	0xD0000000 - 0xDFFFFFFF	256MB	Main Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: D NSC: 0 	Manager Main Expansion Interface. See Main Interconnect Expansion Interfaces .
45	0xE0000000 - 0xE00FFFFF	1MB	PPB	-	Exempt	CPU Private Peripheral Bus Region. Local to the CPU. See CPU Private Peripheral Bus Region .
46	0xE0100000 - 0xE01FFFFF	1MB	Debug System	49	<ul style="list-style-type: none"> Security: NS IDAUID: E NSC: 0 	Debug System Access Region. See Debug System Access Region .
47	0xE0200000 - 0xEFFFFFFF	254MB	Peripheral Expansion	-	<ul style="list-style-type: none"> Security: NS IDAUID: E NSC: 0 	Manager Peripheral Expansion Interface. See Peripheral Interconnect Expansion Interfaces .
48	0xF0000000 - 0xF00FFFFF	1MB	Reserved	-	<ul style="list-style-type: none"> Security: S IDAUID: F NSC: 0 	Reserved
49	0xF0100000 - 0xF01FFFFF	1MB	Debug System	46	<ul style="list-style-type: none"> Security: S IDAUID: F NSC: 0 	Debug System Access Region. See Debug System Access Region .
50	0xF0200000 - 0xFFFFFFFF	254MB	Peripheral Expansion	-	<ul style="list-style-type: none"> Security: S IDAUID: F NSC: 0 	Manager Peripheral Expansion Interface. See Peripheral Interconnect Expansion Interfaces .

6.2 CPU TCM memories

The processors in CRSAS Ma1 are configured to implement Tightly Coupled Memories (TCM) for Instruction and Data. These memories reside in the following location from the perspective of each CPU core:

- 0x0000_0000 to 0x00FF_FFFF and 0x1000_0000 to 0x10FF_FFFF for Instruction TCM. Both regions are aliased, and each provides up to 16MB of TCM space.
- 0x2000_0000 to 0x20FF_FFFF and 0x3000_0000 to 0x30FF_FFFF for Data TCM. Both regions are aliased, and each provides up to 16MB of TCM space.

Each CPU has access to its own local TCMs through its private address and other CPUs TCMs through the Main Interconnect. A CPU does not have access to its own TCMs through the Main

Interconnect. Other managers on the Main interconnect have access to the TCMs. A TCM DMA subordinate interface to allow expansion managers to access the TCMs is provided. For more information, see [TCM subordinate interface](#).

All TCMs always start at the base address of each region. Any memory areas in that region that are not used are reserved and return a bus error response if accessed.

6.3 Volatile Memory Region

CRSAS Ma1 supports up to four internal Volatile Memory (VM) Banks. While these are typically implemented as SRAMs, actual memories used are **IMPLEMENTATION DEFINED**.

All VM banks in the system are of the same size and collectively they form a contiguous area of memory of up to 16MB. These are then aliased onto both the Secure and Non-secure memory regions. A Memory protection controller per VM then divides the VM into pages and determines where each page resides in either the Secure or Non-secure regions. Any unused areas within that 16MB region are reserved.

The following table shows an example where two memory banks are configured and each is 512kB in size.

The Volatile Memory Region example address map defines the following values for security:

NS_MPC

Non-secure access only gated by an MPC.

S_MPC

Secure access only gated by an MPC.

Table 6-2: Volatile Memory Region example address map

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
1	0x2100_0000	0x2107_FFFF	512KB	VM0	4	NS_MPC	Maps to Internal Volatile Memory Bank 1
2	0x2108_0000	0x210F_FFFF	512KB	VM1	5	NS_MPC	Maps to Internal Volatile Memory Bank 2
3	0x2110_2000	0x21FF_FFFF	15MB	Reserved	-	-	Reserved
4	0x3100_0000	0x3107_FFFF	512KB	VM0	1	S_MPC	Maps to Internal Volatile Memory Bank 1
5	0x3108_0000	0x310F_FFFF	512KB	VM1	2	S_MPC	Maps to Internal Volatile Memory Bank 2
8	0x3110_2000	0x31FF_FFFF	15MB	Reserved	-	-	Reserved

6.4 Peripheral Region

The Peripheral Region are memory regions where the peripherals of the system reside. There are eight regions in total:

0x4000_0000 to 0x4000_FFFF

Non-secure region for low latency peripherals that are expected to be aliased in its associated Secure region, 0x5000_0000 to 0x5000_FFFF.

0x4004_0000 to 0x400F_FFFF

Non-secure region for low latency peripherals that are expected to be not aliased.

0x4800_0000 to 0x4800_FFFF

Non-secure region for high latency peripherals that are expected to be aliased in its associated Secure region, 0x5800_0000 to 0x5800_FFFF.

0x4804_0000 to 0x480F_FFFF

Non-secure region for high latency peripherals that are expected to be not aliased.

0x5000_0000 to 0x5000_FFFF

Secure region for low latency peripherals that are expected to be aliased in its associated Non-secure region, 0x4000_0000 to 0x4000_FFFF.

0x5004_0000 to 0x500F_FFFF

Secure region for low latency peripherals that are expected to be not aliased.

0x5800_0000 to 0x5800_FFFF

Secure region for high latency peripherals that are expected to be aliased in its associated Non-secure region, 0x4800_0000 to 0x4800_FFFF.

0x5804_0000 to 0x580F_FFFF

Secure region for high latency peripherals that are expected to be not aliased.

For regions that are aliased to both a Secure and a Non-secure region, the final mapping of a peripheral in these regions to either a Secure or a Non-secure region is either determined by Peripheral Protection Controller (PPC) that is programmed using Secure Access Configuration registers, or determined by the peripheral that intrinsically supports TrustZone.

The following table shows the memory map of the Peripheral Regions.

The Peripheral Region address map defines the following values for security:

NS_PPC

Non-secure access only, gated by a PPC.

S_PPC

Secure access only, gated by a PPC.

S

Secure access only.

NS

Non-secure access only.

P

Privileged access only

UP

Unprivileged and privileged access allowed

P_PPC

Unprivileged access controlled by PPC

When PPC protects a peripheral, the configurability of the security or privileged attributes for a given peripheral is defined by [PERIPHSPPPC0](#), [PERIPHSPPPC1](#), [PERIPHNSPPPC0](#), [PERIPHNSPPPC1](#), [NPUSPPORSL](#), [NPUNSPORPL](#), [NPUSPPORPL](#), [PERIPHNSPPC0](#), and [PERIPHNSPPC1](#) registers.

Table 6-3: Peripheral Region address map

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
1	0x4000_0000	0x4000_0FFF	4KB	MHU0 ²	23	NS_PPC, P_PPC	Message Handling Unit 0. See Message Handling Unit .
2	0x4000_1000	0x4000_1FFF	4KB	MHU1 ²	24	NS_PPC, P_PPC	Message Handling Unit 1. See Message Handling Unit .
3	0x4000_2000	0x4000_3FFF	8KB	DMA ⁴	25	NS, UP	DMA Configuration interface See DMA registers .
4	0x4000_4000	0x4000_4FFF	4KB	NPU0 ³	26	NS_PPC, P_PPC	NPU0 Configuration interface See NPU<m> registers .
5	0x4000_5000	0x4000_5FFF	4KB	NPU1 ³	27	NS_PPC, P_PPC	NPU1 Configuration interface See NPU<m> registers .
6	0x4000_6000	0x4000_6FFF	4KB	NPU2 ³	28	NS_PPC, P_PPC	NPU2 Configuration interface See NPU<m> registers .
7	0x4000_7000	0x4000_7FFF	4KB	NPU3 ³	29	NS_PPC, P_PPC	NPU3 Configuration interface See NPU<m> registers .
8	0x4000_8000	0x4000_FFFF	-	Reserved	-	-	Reserved. (RAZ/WI)
9	0x4004_0000	0x4007_FFFF	-	Reserved	-	-	Reserved
10	0x4008_0000	0x4008_0FFF	4KB	NSACFG	-	NS_PPC, P_PPC	Non-secure Access Configuration Register Block. See Non-secure access configuration register block .
11	0x4008_1000	0x4008_FFFF		Reserved	-	-	Reserved (RAZ/WI)

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
12	0x4009_0000	0x4009_3FFF	16KB	CryptoCell312	-	NS, UP	CryptoCell 312. See Has-Crypto configuration .
13	0x4009_4000	0x400F_FFFF	-	Reserved	-	-	Reserved
14	0x4800_0000	0x4800_0FFF	4KB	TIMER0	41	NS_PPC P_PPC	Timer 0. See Timestamp-based timers registers .
15	0x4800_1000	0x4800_1FFF	4KB	TIMER1	42	NS_PPC P_PPC	Timer 1. See Timestamp-based timers registers .
16	0x4800_2000	0x4800_2FFF	4KB	TIMER2	43	NS_PPC P_PPC	Timer 2. See Timestamp-based timers registers .
17	0x4800_3000	0x4800_3FFF	4KB	TIMER3	44	NS_PPC P_PPC	Timer 3. See Timestamp-based timers registers .
18	0x4800_4000	0x4800_FFFF	-	Reserved	-	-	Reserved (RAZ/WI)
19	0x4804_0000	0x4804_0FFF	4KB	NSWDCTRL	-	NS_PPC, P_PPC, P	Non-secure Watchdog Control Frame. See Timestamp-based Watchdogs registers .
20	0x4804_1000	0x4804_1FFF	4KB	NSWDREF	-	NS_PPC, P_PPC	Non-secure Watchdog Refresh Frame. See Timestamp-based Watchdogs registers .
21	0x4804_2000	0x4804_FFFF	-	Reserved	-	-	Reserved (RAZ/WI)
22	0x4805_0000	0x480F_FFFF	-	Reserved	-	-	Reserved
23	0x5000_0000	0x5000_0FFF	4KB	MHU0 ²	1	S_PPC, P_PPC	Message Handling Unit 0. See Message Handling Unit .
24	0x5000_1000	0x5000_1FFF	4KB	MHU1 ²	2	S_PPC, P_PPC	Message Handling Unit 1. See Message Handling Unit .
25	0x5000_2000	0x5000_3FFF	8KB	DMA ⁴	3	S, UP	DMA Configuration interface See DMA registers .
26	0x5000_4000	0x5000_4FFF	4KB	NPU0 ³	4	S_PPC, P_PPC	NPU0 Configuration interface See NPU<m> registers .
27	0x5000_5000	0x5000_5FFF	4KB	NPU1 ³	5	S_PPC, P_PPC	NPU1 Configuration interface See NPU<m> registers .
28	0x5000_6000	0x5000_6FFF	4KB	NPU2 ³	6	S_PPC, P_PPC	NPU2 Configuration interface See NPU<m> registers .
29	0x5000_7000	0x5000_7FFF	4KB	NPU3 ³	7	S_PPC, P_PPC	NPU3 Configuration interface See NPU<m> registers .
30	0x5000_8000	0x5000_FFFF	-	Reserved	-	-	Reserved (RAZ/WI)
31	0x5004_0000	0x5007_FFFF	-	Reserved	-	-	Reserved

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
32	0x5008_0000	0x5008_0FFF	4KB	SACFG	-	S_PPC, P_PPC, P	Secure Access Configuration Register Block. Secure access configuration register block.
33	0x5008_1000	0x5008_2FFF	-	Reserved	-	-	Reserved (RAZ/WI)
34	0x5008_3000	0x5008_3FFF	4KB	VM0MPC ¹	-	S_PPC, P_PPC, P	VM0 Memory Protection Controller. See Volatile memory .
35	0x5008_4000	0x5008_4FFF	4KB	VM1MPC ¹	-	S_PPC, P_PPC, P	VM1 Memory Protection Controller. See Volatile memory .
36	0x5008_5000	0x5008_5FFF	4KB	VM2MPC ¹	-	S_PPC, P_PPC, P	VM2 Memory Protection Controller. See Volatile memory .
37	0x5008_6000	0x5008_6FFF	4KB	VM3MPC ¹	-	S_PPC, P_PPC, P	VM3 Memory Protection Controller. See Volatile memory .
38	0x5008_7000	0x5008_FFFF	-	Reserved	-	-	Reserved (RAZ/WI)
39	0x5009_0000	0x5009_3FFF	16KB	CryptoCell312	-	S, UP	CryptoCell 312. See Has-Crypto configuration .
40	0x5009_4000	0x500F_FFFF	-	Reserved	-	-	Reserved
41	0x5800_0000	0x5800_0FFF	4KB	TIMER0	14	S_PPC, P_PPC	Timer 0. See Timestamp-based timers registers .
42	0x5800_1000	0x5800_1FFF	4KB	TIMER1	15	S_PPC, P_PPC	Timer 1. See Timestamp-based timers registers .
43	0x5800_2000	0x5800_2FFF	4KB	TIMER2	16	S_PPC, P_PPC	Timer 2. See Timestamp-based timers registers .
44	0x5800_3000	0x5800_3FFF	4KB	TIMER3	17	S_PPC, P_PPC	Timer 3. See Timestamp-based timers registers .
45	0x5800_4000	0x5800_FFFF	-	Reserved	-	-	Reserved (RAZ/WI)
46	0x5804_0000	0x5804_0FFF	4KB	SWDCTRL	-	S_PPC, P_PPC, P	Secure Watchdog Control Frame. See Timestamp-based Watchdogs registers .
47	0x5804_1000	0x5804_1FFF	4KB	SWDREF	-	S_PPC, P_PPC	Secure Watchdog Refresh Frame. See Timestamp-based Watchdogs registers .
48	0x5804_2000	0x5804_FFFF	-	Reserved	-	-	Reserved (RAZ/WI)
49	0x5805_0000	0x580F_FFFF	-	Reserved	-	-	Reserved

¹ The number of VM<i>MPC regions depends on NUMVMBANK. If VM<i> does not exist, then the VM<i>MPC region is Reserved.

² MHU0 and MHU1 only exist if NUMCPU > 0.

³ NPUn only exists if NUMNPU > 0.

⁴ DMA only exists if NUMDMA = 1

6.4.1 Message Handling Unit register map

The CRSAS Ma1 implements up to two Message Handling Units (MHUs). These allow software to raise interrupts to the CPU cores. Both MHUs are mapped twice into both Secure and Non-secure regions as follows, and a PPC then controls which area each MHU resides:

- MHU0 in Non-secure region at 0x4000_0000 and Secure region at 0x5000_0000
- MHU1 in Non-secure region at 0x4000_1000 and Secure region at 0x5000_1000

If there is only one CPU core in the system, neither MHU exists and the two regions are reserved and any accesses to an MHU results in **RAZ/WI**. For write access to these registers, only 32-bit writes are supported. Any byte and halfword writes result in its write data ignored.



There are two MHUs in the system so that software can map one for Secure use and another for Non-secure use. However, software can choose to map both to Secure, or Non-secure if needed.

Each MHU has the following register map,. The security of each MHU register area is defined by PPCs' drive by registers in [Secure Access Configuration Register Block](#) and [Non-secure Access Configuration Register Block](#). See [PERIPHNSPPCO](#), [PERIPHNSPPC1](#), [PERIPHSPPPCO](#), [PERIPHSPPPC1](#), [PERIPHNSPPPCO](#), and [PERIPHNSPPPC1](#). All registers reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**. The CRSAS Ma1 defines up to two Message Handling Units (MHUs).

Table 6-4: Message Handling Unit register map

Offset	Name	Type	Reset	Width	Description
0x000 + n(0x010)	CPU<n>INTR_STAT ¹	RO	0x0000_0000	32-bit	CPU <n> Interrupt Status Register. See CPU<n>INTR_STAT .
0x004 + n(0x010)	CPU<n>INTR_SET ¹	WO	0x0000_0000	32-bit	CPU <n> Interrupt Set Register. See CPU<n>INTR_SET .
0x008 + n(0x010)	CPU<n>INTR_CLR ¹	WO	0x0000_0000	32-bit	CPU <n> Interrupt Clear Register. See CPU<n>INTR_CLR .
0x00C + n(0x010)	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
(NUMCPU+1) (0x010) – 0xFC8	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0056	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_001B	32-bit	Peripheral ID 2

Offset	Name	Type	Reset	Width	Description
0xFE0C	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

¹ The following set of registers per CPU exists:

- CPU<n>INTR_STAT
- CPU<n>INTR_SET
- CPU<n>INTR_CLR

6.4.1.1 CPU <n> Interrupt registers

The CPU <n> Interrupt registers, CPU<n>INTR_STAT, CPU<n>INTR_SET, and CPU<n>INTR_CLR, allow software to raise an interrupt, clear an interrupt and check the value written that is used to raise the interrupt to CPU <n>. Separate Set and Clear registers allow individual bit of the interrupt status to be set and cleared, therefore it supports SW using each bit to represent an event that can be independently set and cleared. One set of these registers exist per CPU<n>.

6.4.1.1.1 CPU<n>INTR_STAT

CPU <n> Interrupt Status register.

Configurations

Present only when NUMCPU > 0.

This register is **RAZ/WI** if NUMCPU = 0.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

For write access to this register, only 32-bit writes are supported. Any byte and halfword writes result in its write data ignored.

Bit descriptions

Table 6-5: CPU<n>INTR_STAT bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	ROWI	0x000_0000
3:0	CPU<n> INTR_STAT	CPU <n> Interrupt Status. When any bit is set to HIGH, the MHU interrupt signal to CPU <n> is set to HIGH.	ROWI	0x0

6.4.1.1.2 CPU<n>INTR_SET

The CPU <n> Interrupt Set register.

Configurations

Present only when NUMCPU > 0.

This register is **RAZ/WI** if NUMCPU = 0.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

For write access to this register, only 32-bit writes are supported. Any byte and halfword writes result in its write data ignored.

Bit descriptions

Table 6-6: CPU<n>INTR_SET bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	RAZ/WI	0x000_0000
3:0	CPU<n> INTR_SET	CPU <n> Interrupt Set. When a value '1' is written to CPU<n>INTR_SET[i], the corresponding CPU<n>INTR_STAT[i] is set to high.	RAZW1S	0x0

6.4.1.1.3 CPU<n>INTR_CLR

The CPU <n> Interrupt Clear register.

Configurations

Present only when NUMCPU > 0. This register is **RAZ/WI** if NUMCPU = 0.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

For write access to this register, only 32-bit writes are supported. Any byte and halfword writes result in its write data ignored.

Bit descriptions

Table 6-7: CPU<n>INTR_CLR bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	RAZ/WI	0x000_0000
3:0	CPU<n> INTR_CLR	CPU <n> a value '1' is written to CPU<n>INTR_CLR[i], the corresponding CPU<n>INTR_STAT[i] is cleared.	RAZ/ W1C	0x0

6.4.2 Secure Access Configuration Register Block

The Secure Access Configuration Register Block implements program visible states that allow software to control security gating units within the design. The register block base address is 0x5008_0000. These registers are Secure privileged access only and support 32-bit RW accesses. For write access to these registers, only 32-bit writes are supported. Any byte and halfword writes are ignored.

All registers reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**.

The following table list the registers within this block. Details of each register are described in the following subsections.

Table 6-8: Secure Access Configuration Register Block Register Map

Offset	Name	Type	Reset	Width	Description
0x000	SPCSECCTRL	RW	0x0000_0000	32-bit	Secure Privileged Controller Secure Configuration Control register, see SPCSECCTRL .
0x004	BUSWAIT	RW	CFG_DEF	32-bit	Bus Access wait control after reset, BUSWAIT .
0x008	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0x010	SECRESPCFG	RW	0x0000_0000	32-bit	Security Violation Response Configuration Register, see SECRESPCFG .
0x014	NSCCFG	RW	0x0000_0000	32-bit	Non-secure Callable Configuration for IDAU, see NSCCFG .
0x018	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved

Offset	Name	Type	Reset	Width	Description
0x01C	SECMPCINTSTAT	RO	0x0000_0000	32-bit	Secure MPC Interrupt Status, see SECMPCINTSTAT .
0x020	SECPPCINTSTAT	RO	0x0000_0000	32-bit	Secure PPC Interrupt Status, see SECPPCINTSTAT .
0x024	SECPPCINTCLR	RW	0x0000_0000	32-bit	Secure PPC Interrupt Clear, see SECPPCINTCLR .
0x028	SECPPCINTEN	RW	0x0000_0000	32-bit	Secure PPC Interrupt Enable, see SECPPCINTEN .
0x02C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0x030	SECMSCINTSTAT	RO	0x0000_0000	32-bit	Secure MSC Interrupt Status, see SECMSCINTSTAT .
0x034	SECMSCINTCLR	RW	0x0000_0000	32-bit	Secure MSC Interrupt Clear, see SECMSCINTCLR .
0x038	SECMSCINTEN	RW	0x0000_0000	32-bit	Secure MSC Interrupt Enable, see SECMSCINTEN .
0x03C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0x040	BRGINTSTAT	RO	0x0000_0000	32-bit	Bridge Buffer Error Interrupt Status, see BRGINTSTAT .
0x044	BRGINTCLR	RW	0x0000_0000	32-bit	Bridge Buffer Error Interrupt Clear, see BRGINTCLR .
0x048	BRGINTEN	RW	0x0000_0000	32-bit	Bridge Buffer Error Interrupt Enable, see BRGINTEN .
0x04C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0x050	MAINNSPPCO	RW	0x0000_0000	32-bit	Non-secure Access Peripheral Protection Control 0 on the Main Interconnect, see MAINNSPPCO .
0x054-0x05C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0x060	MAINNSPPCEXP0	RW	0x0000_0000	32-bit	Expansion 0 Non-secure Access Peripheral Protection Control on the Main Interconnect, see MAINNSPPCEXP{0-3} .
0x064	MAINNSPPCEXP1	RW	0x0000_0000	32-bit	Expansion 1 Non-secure Access Peripheral Protection Control on the Main Interconnect, see MAINNSPPCEXP{0-3} .
0x068	MAINNSPPCEXP2	RW	0x0000_0000	32-bit	Expansion 2 Non-secure Access Peripheral Protection Control on the Main Interconnect, see MAINNSPPCEXP{0-3} .
0x06C	MAINNSPPCEXP3	RW	0x0000_0000	32-bit	Expansion 3 Non-secure Access Peripheral Protection Control on the Main Interconnect, see MAINNSPPCEXP{0-3} .
0x070	PERIPHNSPPCO	RW	0x0000_0000	32-bit	Non-secure Access Peripheral Protection Control 0 on Peripheral Interconnect. See PERIPHNSPPCO .
0x074	PERIPHNSPPC1	RW	0x0000_0000	32-bit	Non-secure Access Peripheral Protection Control 1 on Peripheral Interconnect, see PERIPHNSPPC1 .
0x078	NPUSPPORSL	RW	NPU<m> PORSLRST	32-bit	Secure Access NPU security level reset state control: Each Field Controls the PORSL inputs for an associated NPU, see NPUSPPORSL .
0x07C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0x080	PERIPHNSPPCEXP0	RW	0x0000_0000	32-bit	Expansion 0 Non-secure Access Peripheral Protection Control on Peripheral Bus, see PERIPHNSPPCEXP{0-3} .
0x084	PERIPHNSPPCEXP1	RW	0x0000_0000	32-bit	Expansion 1 Non-secure Access Peripheral Protection Control on Peripheral Bus, see PERIPHNSPPCEXP{0-3} .
0x088	PERIPHNSPPCEXP2	RW	0x0000_0000	32-bit	Expansion 2 Non-secure Access Peripheral Protection Control on Peripheral Bus, see PERIPHNSPPCEXP{0-3} .
0x08C	PERIPHNSPPCEXP3	RW	0x0000_0000	32-bit	Expansion 3 Non-secure Access Peripheral Protection Control on Peripheral Bus, see PERIPHNSPPCEXP{0-3} .

Offset	Name	Type	Reset	Width	Description
0x090	MAINSPPPC0	RW	0x0000_0000	32-bit	Secure Unprivileged Access Peripheral Protection Control 0 on Main Interconnect, see MAINSPPPC0 .
0x094-0x09C	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved.
0x0A0	MAINSPPPCEXP0	RW	0x0000_0000	32-bit	Expansion 0 Secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINSPPPCEXP{0-3} .
0x0A4	MAINSPPPCEXP1	RW	0x0000_0000	32-bit	Expansion 1 Secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINSPPPCEXP{0-3} .
0x0A8	MAINSPPPCEXP2	RW	0x0000_0000	32-bit	Expansion 2 Secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINSPPPCEXP{0-3} .
0x0AC	MAINSPPPCEXP3	RW	0x0000_0000	32-bit	Expansion 3 Secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINSPPPCEXP{0-3} .
0x0B0	PERIPHSPPC0	RW	0x0000_0000	32-bit	Secure Unprivileged Access Peripheral Protection Control 0 on Peripheral Interconnect, see PERIPHSPPC0 .
0x0B4	PERIPHSPPC1	RW	0x0000_0000	32-bit	Secure Unprivileged Access Peripheral Protection Control 1 on Peripheral Interconnect, see PERIPHSPPC1 .
0x0B8	NPUSPPORPL	RW	NPU<m>PORPLRST	32-bit	Secure Access NPU privilege level reset state control see NPUSPPORPL .
0x0BC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved.
0x0C0	PERIPHSPPCEXP0	RW	0x0000_0000	32-bit	Expansion 0 Secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHSPPCEXP{0-3} .
0x0C4	PERIPHSPPCEXP1	RW	0x0000_0000	32-bit	Expansion 1 Secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHSPPCEXP{0-3} .
0x0C8	PERIPHSPPCEXP2	RW	0x0000_0000	32-bit	Expansion 2 Secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHSPPCEXP{0-3} .
0x0CC	PERIPHSPPCEXP3	RW	0x0000_0000	32-bit	Expansion 3 Secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHSPPCEXP{0-3} .
0x0D0	NSMSCEXP	RW	CFG_DEF	32-bit	Expansion MSC Non-secure Configuration, see NSMSCEXP .
0x0D4-0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4-0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0052	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_003B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.4.2.1 SPCSECCTRL

The Security Privilege Controller Security Configuration Control Register implements the security lock register.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-9: SPCSECCTRL bit descriptions

Bits	Name	Description	Type	Default
31:1	-	Reserved.	RAZ/WI	0x0000_0000
0	SPCSECCFGLOCK	Active-HIGH Control to Disable writes to Security related control registers in the Secure Access Configuration Register Block. Once set to HIGH, it can no longer be cleared to zero except through reset or the PD_SYS turning OFF. Registers that can no longer be modified when SPCSECCFGLOCK is set to HIGH are: <ul style="list-style-type: none"> • NSCCFG • MAINNSPPCO • MAINNSPPCEXP<i> • PERIPHNSPPCO • PERIPHNSPPC1 • PERIPHNSPPCEXP<i> • MAINSPPPCO • MAINSPPPCEXP<i> • PERIPHSPPPCO • PERIPHSPPPC1 • PERIPHSPPPCEXP<i> • NSMSCEXP • NPUSPPORSL • NPUSPPORPL 	RW1S	0x0

6.4.2.2 BUSWAIT

The Bus Access Wait Register allows software to gate access entering the interconnect from specific managers in the system, causing them to stall so that the CPU can complete the configuration of the MPCs or other Security registers in the system before the stalled accesses commencing.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-10: BUSWAIT bit descriptions

Bits	Name	Description	Type	Default
31:17	-	Reserved.	RAZ/ WI	0x0000
16	ACC_WAITN_STATUS	<p>This status register indicates the status of any gating units that are used to block bus access to the system:</p> <ul style="list-style-type: none"> • '1': allow access. • '0': block access. <p>This register reflects the combined status of all gating units in the system, including status on the input signal, ACCWAITNSTATUS expected to be driven from external gating units.</p> <p>Both ACC_WAITN_STATUS and ACC_WAITN together, ensuring that software can determine that all gates have reached the state that is requested.</p>	RO	0x0
15:1	-	Reserved.	RAZ/ WI	0x0000

Bits	Name	Description	Type	Default
0	ACC_WAITN	<p>Request gating units to block bus access to system:</p> <ul style="list-style-type: none"> • '1': allow access. • '0': block access. <p>This control only affects the Access Control Gates (ACG) in the system that feeds into the interconnect, and it excludes access from CPU cores. This register also drives the output signal ACCWAITn.</p> <p>Both ACC_WAITN_STATUS and ACC_WAITN together, ensuring that software can determine that all gates have reached the state that is requested.</p>	RW	ACCWAITNRST

6.4.2.3 SECRESPCFG

The Security Violation Response Configuration Register is used to define the response to an access that causes security violation on the Bus Fabric.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-11: SECRESPCFG bit descriptions

Bits	Name	Description	Type	Default
31:1	-	Reserved.	RAZ/WI	0x0000_0000
0	SECRESPCFG	<p>This field configures the response in case of a security violation:</p> <ul style="list-style-type: none"> • '0': Read-Zero Write Ignore • '1': Bus error <p>Some components, for example, the AHB Memory Protection Controllers (MPC), provide their own control registers to configure their own response. Components that have their own register to control functionality do not have to use this register.</p>	RW	0x0

6.4.2.4 NSCCFG

The Non-secure Callable Configuration register allows software to define if the region 0x1000_0000 to 0x1FFF_FFFF that normally hosts Secure code, and the region 0x3000_0000 to 0x3FFF_FFFF that normally implements Secure Volatile Memories, are Non-secure Callable regions of memory.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-12: NSCCFG bit descriptions

Bits	Name	Description	Type	Default
31:2	-	Reserved.	RAZ/WI	0x0000_0000
1	RAMNSC	Configures if the region 0x3000_0000 to 0x3FFF_FFFF is Non-secure Callable: <ul style="list-style-type: none"> '0': Not Non-secure Callable '1': Non-secure Callable. 	RW	0x0
0	CODENSC	Configures if the CODE region 0x1000_0000 to 0x1FFF_FFFF is Non-secure Callable: <ul style="list-style-type: none"> '0': Not Non-secure Callable '1': Non-secure Callable. 	RW	0x0

6.4.2.5 SECMPICINTSTAT

The interrupt signals from all Memory Protection Controllers (MPC), both within the CRSAS Ma1 and in the Expansion logic, are merged and sent to the CPUs on a single Interrupt signal. The Secure MPC Interrupt Status Register therefore provides Secure software with the ability to check

which one of the MPCs is causing the interrupt. Once the source of the interrupt is identified, you must use the MPC register interface to clear the interrupt.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-13: SECMPCINTSTAT bit descriptions

Bits	Name	Description	Type	Default
31:16	SMPCEXP_STATUS	Interrupt Status for Expansion Memory Protection Controller. Each bit <i>i</i> shows the status of input signal SMPCEXPSTATUS[i] . The MPCEXPDIS configuration point defines if each bit within this register is actually implemented such that if MPCEXPDIS[i] = 1'b1, then SMPCEXP_STATUS[i] is disabled and is RAZ/WI .	RO	0x0000_0000
15:4	-	Reserved.	RAZ/WI	0x0000_0000
3	SMPCVM3_STATUS	Interrupt Status for Memory Protection Controller of Volatile Memory Bank 3. This register is not used and RAZ/WI if NUMVMBANK < 4	RO	0x0
2	SMPCVM2_STATUS	Interrupt Status for Memory Protection Controller of Volatile Memory Bank 2. This register is not used and RAZ/WI if NUMVMBANK < 3	RO	0x0
1	SMPCVM1_STATUS	Interrupt Status for Memory Protection Controller of Volatile Memory Bank 1. This register is not used and RAZ/WI if NUMVMBANK < 2	RO	0x0
0	SMPCVM0_STATUS	Interrupt Status for Memory Protection Controller of Volatile Memory Bank 0. This register is not used and RAZ/WI if NUMVMBANK < 1	RO	0x0

6.4.2.6 SECPPCINTSTAT

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised through a combined interrupt that is then sent to the CPUs. The PPC Secure PPC Interrupt Status Register allows software to determine the source of the interrupt.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-14: SECPPCINTSTAT bit descriptions

Bits	Name	Description	Type	Default
31:24	-	Reserved.	RAZ/WI	0x000
23:20	SMAINPPCEXP_STATUS	Interrupt Status of Expansion Peripheral Protection Controller on the Main Interconnect. Each bit <i>i</i> shows the status of PPC connected to the input signal SMAINPPCEXPSTATUS[i] .	RO	0x0
19:17	-	Reserved.	RAZ/WI	0x0
16	SMAINPPCO_STATUS	Interrupt Status of Peripheral Protection Controller 0 on the Main Interconnect. Since there are no PPCs on the main interconnect, the field is not used and is reserved.	RO	0x0
15:8	-	Reserved.	RAZ/WI	0x000
7:4	SPERIPHPPCEXP_STATUS	Interrupt Status of Expansion Peripheral Protection Controller on the Peripheral Interconnect. Each bit <i>i</i> shows the status of PPC connected to the input signal SPERIPHPPCEXPSTATUS[i] .	RO	0x0
3:2	-	Reserved.	RAZ/WI	0x0
1	SPERIPHPPC1_STATUS	Interrupt Status of Peripheral Protection Controller 1 on the Peripheral Interconnect within the System.	RO	0x0
0	SPERIPHPPC0_STATUS	Interrupt Status of Peripheral Protection Controller 0 on the Peripheral Interconnect within the System.	RO	0x0

6.4.2.7 SECPPCINTCLR

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised through a combined interrupt that is then sent to the CPUs. The PPC Secure PPC Interrupt Clear Register allows software to clear the interrupt.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-15: SECPPCINTCLR bit descriptions

Bits	Name	Description	Type	Default
31:24	-	Reserved.	RAZ/ WI	0x000
23:20	SMAINPPCEXP_CLR	Interrupt Clear of Expansion Peripheral Protection Controller on the Main Interconnect. Each bit <i>i</i> , when set to HIGH clears the interrupt status of the PPC connected to SMAINPPCEXPSTATUS[i] and SMAINPPCEXPCLR[i] .	RAZ/ W1C	0x0
19:17	-	Reserved.	RAZ/ WI	0x0
16	SMAINPPCO_CLR	Interrupt Clear of Peripheral Protection Controller 0 on the Main Interconnect. Write '1' to clear SMAINPPCO_STATUS. Since there are no PPCs on the main interconnect, the field is not used and is reserved.	RAZ/ WI	0x0
15:8	-	Reserved.	RAZ/ WI	0x000
7:4	SPERIPHPPCEXP_CLR	Interrupt Clear of Expansion Peripheral Protection Controller on the Peripheral Interconnect. Each bit SPERIPHPPCEXP_CLR[i], when set to HIGH clears the interrupt status of the PPC connected to SPERIPHPPCEXPSTATUS[i] and SPERIPHPPCEXPCLR[i] .	RAZ/ W1C	0x0
3:2	-	Reserved.	RAZ/ WI	0x0
1	SPERIPHPPC1_CLR	Interrupt Clear of Peripheral Protection Controller 1 on the Peripheral Interconnect within the System.	RAZ/ W1C	0x0
0	SPERIPHPPC0_CLR	Interrupt Clear of Peripheral Protection Controller 0 on the Peripheral Interconnect within the System.	RAZ/ W1C	0x0

6.4.2.8 SECPPCINTEN

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised through a combined interrupt that is then sent to the CPUs. The PPC Secure PPC Interrupt Status Enable Register allows software to enable or disable (Mask) the interrupt.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-16: SECPPCINTEN bit descriptions

Bits	Name	Description	Type	Default
31:24	-	Reserved.	RAZ/WI	0x000
23:20	SMAINPPCEXP_EN	Interrupt Enable of Expansion Peripheral Protection Controller on the Main Interconnect. Write '1' to bit <i>i</i> enable interrupt from SMAINPPCEXP_STATUS[i].	RW	0x0
19:17	-	Reserved.	RAZ/WI	0x0
16	SMAINPPCO_EN	Interrupt Enable of Peripheral Protection Controller 0 on the Main Interconnect. Write '1' to enable interrupt from SMAINPPCO_STATUS. Since there are no PPCs on the main interconnect, the field is not used and is reserved.	RO	0x0
15:8	-	Reserved.	RAZ/WI	0x000
7:4	SPERIPHPPCEXP_EN	Interrupt Enable of Expansion Peripheral Protection Controller on the Peripheral Interconnect. Write '1' to bit <i>i</i> to enable interrupt from SPERIPHPPCEXP_STATUS [i].	RW	0x0
3:2	-	Reserved.	RAZ/WI	0x0
1	SPERIPHPPC1_EN	Interrupt Enable of Peripheral Protection Controller 1 on the Peripheral Interconnect within the System. Write '1' to enable interrupt from SPERIPHPPC1_STATUS.	RW	0x0
0	SPERIPHPPCO_EN	Interrupt Enable of Peripheral Protection Controller 0 on the Peripheral Interconnect within the System. Write '1' to enable interrupt from SPERIPHPPCO_STATUS.	RW	0x0

6.4.2.9 SECMSCINTSTAT

When a security violation occurs at any Manager Security Controller (MSC) in the subsystem and in the expansion logic, an interrupt is raised through a combined interrupt to the CPUs. The Secure MSC Interrupt Status Register allows software to determine the source of the interrupt.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-17: SECMSCINTSTAT bit descriptions

Bits	Name	Description	Type	Default
31:16	SMSCEXP_STATUS	Interrupt Status for Expansion MSC. Each bit <i>i</i> shows the interrupt status of MSC connected to the input signal SMSCEXPSTATUS[i] . The configuration option MSCEXPDIS defines if each bit within this register is actually implemented such that if MSCEXPDIS[i] = 1'b1 then SMSCEXP_STATUS[i] is disabled and RAZ/WI .	RO	0x0000
15:12	-	Reserved.	RAZ/WI	0x0
11	SMSCNPU3M1_STATUS	Interrupt status for NPU3 M1 MSC RAZ/WI If NUMNPU < 4	RO	0x0
10	SMSCNPU3M0_STATUS	Interrupt status for NPU3 M0 MSC RAZ/WI If NUMNPU < 4	RO	0x0
9	SMSCNPU2M1_STATUS	Interrupt status for NPU2 M1 MSC RAZ/WI If NUMNPU < 3	RO	0x0
8	SMSCNPU2M0_STATUS	Interrupt status for NPU2 M0 MSC RAZ/WI If NUMNPU < 3	RO	0x0
7	SMSCNPU1M1_STATUS	Interrupt status for NPU1 M1 MSC RAZ/WI If NUMNPU < 2	RO	0x0

Bits	Name	Description	Type	Default
6	SMSCNPU1M0_STATUS	Interrupt status for NPU1 M0 MSC RAZ/WI If NUMNPU < 2	RO	0x0
5	SMSCNPU0M1_STATUS	Interrupt status for NPU0 M1 MSC RAZ/WI If NUMNPU < 1	RO	0x0
4	SMSCNPU0M0_STATUS	Interrupt status for NPU0 M0 MSC RAZ/WI If NUMNPU < 1	RO	0x0
3	SMSCDMAM1_STATUS	Interrupt status for DMA M1 MSC RAZ/WI If NUMDMA < 1	RO	0x0
2	SMSCDMAM0_STATUS	Interrupt status for DMA M0 MSC RAZ/WI If NUMDMA < 1	RO	0x0
1:0	-	Reserved.	RAZ/WI	0x0

6.4.2.10 SECMSCINTCLR

When a security violation occurs at any Manager Security Controller (MSC) in the subsystem and in the expansion logic, an interrupt is raised through a combined interrupt to the CPUs. The Secure MSC Interrupt Clear Register allows software to clear the interrupt.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-18: SECMSCINTCLR bit descriptions

Bits	Name	Description	Type	Default
31:16	SMSCEXP_CLR	Interrupt Clear for Expansion MSC. Each bit <i>i</i> , when set to HIGH clears the interrupt status of the MSC connected to SMSCEXPSTATUS[i] and SMSCEXPCLR[i] . The configuration option MSCEXPDIS defines whether each bit within this register is actually implemented such that if MSCEXPDIS[i] = 1'b1 then SMSCEXP_CLR[i] is disabled and RAZ/WI .	RAZ/ W1C	0x0000
15:12	-	Reserved.	RAZ/ WI	0x0
11	SMSCNPU3M1_CLR	Interrupt clear for NPU3 M1 MSC RAZ/WI If NUMNPU < 4	RAZ/ W1C	0x0
10	SMSCNPU3M0_CLR	Interrupt clear for NPU3 M0 MSC RAZ/WI If NUMNPU < 4	RAZ/ W1C	0x0
9	SMSCNPU2M1_CLR	Interrupt clear for NPU2 M1 MSC RAZ/WI If NUMNPU < 3	RAZ/ W1C	0x0
8	SMSCNPU2M0_CLR	Interrupt clear for NPU2 M0 MSC RAZ/WI If NUMNPU < 3	RAZ/ W1C	0x0
7	SMSCNPU1M1_CLR	Interrupt clear for NPU1 M1 MSC RAZ/WI If NUMNPU < 2	RAZ/ W1C	0x0
6	SMSCNPU1M0_CLR	Interrupt clear for NPU1 M0 MSC RAZ/WI If NUMNPU < 2	RAZ/ W1C	0x0
5	SMSCNPU0M1_CLR	Interrupt clear for NPU0 M1 MSC RAZ/WI If NUMNPU < 1	RAZ/ W1C	0x0
4	SMSCNPU0M0_CLR	Interrupt clear for NPU0 M0 MSC RAZ/WI If NUMNPU < 1	RAZ/ W1C	0x0
3	SMSCDMAM1_CLR	Interrupt clear for DMA M1 MSC RAZ/WI If NUMDMA < 1	RAZ/ W1C	0x0
2	SMSCDMAM0_CLR	Interrupt clear for DMA M0 MSC RAZ/WI If NUMDMA < 1	RAZ/ W1C	0x0
1:0	-	Reserved.	RAZ/ WI	0x0

6.4.2.11 SECMSCINTEN

When a security violation occurs at any Manager Security Controller (MSC) in the subsystem and in the expansion logic, an interrupt is raised through a combined interrupt to the CPUs. The Secure MSC Interrupt Enable Register allows software to enable or disable (mask) the interrupt.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-19: SECMSCINTEN bit descriptions

Bits	Name	Description	Type	Default
31:16	SMSCEXP_EN	Interrupt Enable for Expansion MSC. Each bit <i>i</i> enables or disables the input interrupt signal SMSCEXPSTATUS[i] . The parameter MSCEXPDIS defines if each bit within this register is actually implement such that if MSCEXPDIS[i] = 1'b1, then SMSCEXP_EN[i] is disabled and RAZ/WI .	RW	0x0000
15:12	-	Reserved.	RAZ/WI	0x0
11	SMSCNPU3M1_EN	Interrupt enable for NPU3 M1 MSC RAZ/WI If NUMNPU < 4	RW	0x0
10	SMSCNPU3M0_EN	Interrupt enable for NPU3 M0 MSC RAZ/WI If NUMNPU < 4	RW	0x0
9	SMSCNPU2M1_EN	Interrupt enable for NPU2 M1 MSC RAZ/WI If NUMNPU < 3	RW	0x0
8	SMSCNPU2M0_EN	Interrupt enable for NPU2 M0 MSC RAZ/WI If NUMNPU < 3	RW	0x0
7	SMSCNPU1M1_EN	Interrupt enable for NPU1 M1 MSC RAZ/WI If NUMNPU < 2	RW	0x0

Bits	Name	Description	Type	Default
6	SMSCNPU1M0_EN	Interrupt enable for NPU1 M0 MSC RAZ/WI If NUMNPU < 2	RW	0x0
5	SMSCNPU0M1_EN	Interrupt enable for NPU0 M1 MSC RAZ/WI If NUMNPU < 1	RW	0x0
4	SMSCNPU0M0_EN	Interrupt enable for NPU0 M0 MSC RAZ/WI If NUMNPU < 1	RW	0x0
3	SMSCDMAM1_EN	Interrupt enable for DMA M1 MSC RAZ/WI If NUMDMA < 1	RW	0x0
2	SMSCDMAM0_EN	Interrupt enable for DMA M0 MSC RAZ/WI If NUMDMA < 1	RW	0x0
1:0	-	Reserved.	RAZ/WI	0x0

6.4.2.12 BRGINTSTAT

The CRSAS Ma1 and its expansion logic can contain bus bridges which are necessary to handle clock domain crossing. To improve system performance, some of these bridges can buffer write data and complete a write access on their subordinate interfaces before any potential error response is received for the write access on their manager interfaces. When this occurs, these bridges can raise a combined interrupt. The Bridge Buffer Error Interrupt Status Register allows software to determine source of the interrupt.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-20: BRGINTSTAT bit descriptions

Bits	Name	Description	Type	Default
31:16	BRGEXP_STATUS	Interrupt Status for Expansion Bridge Buffer Error Interrupts. Each bit <i>i</i> shows the interrupt status of the bridge connected to the input signal BRGEXPSTATUS[i] . The configuration option BRGEXPDIS defines if each bit within this register is actually implemented such that if BRGEXPDIS[i] = 1'b1, then BRGEXP_STATUS[i] is disabled and RAZ/WI .	RO	0x0000
15:0	-	Reserved.	RAZ/WI	0x0000

6.4.2.13 BRGINTCLR

The CRSAS Ma1 and its expansion logic can contain bus bridges which are necessary to handle clock domain crossing. To improve system performance, some of these bridges can buffer write data and complete a write access on their subordinate interfaces before any potential error response is received for the write access on their manager interfaces. When this occurs, these bridges can raise a combined interrupt. The Bridge Buffer Error Interrupt Clear Register allows software to clear the interrupt.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-21: BRGINTCLR bit descriptions

Bits	Name	Description	Type	Default
31:16	BRGEXP_CLR	Interrupt Clear of Expansion Bridge Buffer Error Interrupts. Each bit <i>i</i> when set to HIGH clears the interrupt status of the bridge connected to BRGEXPSTATUS[i] and BRGEXPCLEAR[i] . The configuration option BRGEXPDIS defines if each bit within this register is actually implemented such that if BRGEXPDIS[i] = 1'b1 then BRGEXP_CLR[i] is disabled and RAZ/WI .	RAZ/W1C	0x0000
15:0	-	Reserved.	RAZ/WI	0x0000

6.4.2.14 BRGINTEN

The CRSAS Ma1 and its expansion logic can contain bus bridges which are necessary to handle clock domain crossing. To improve system performance, some of these bridges can buffer write data and complete a write access on their subordinate interfaces before any potential error response is received for the write access on their manager interfaces. When this occurs, these bridges can raise a combined interrupt. The Bridge Buffer Error Interrupt Enable Register allows software to enable or disable (mask) the interrupt.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-22: BRGINTEN bit descriptions

Bits	Name	Description	Type	Default
31:16	BRGEXP_EN	Interrupt Enable of Expansion Bridge Buffer Error Interrupts. Each bit <i>i</i> enables the interrupt of the bridge connected to BRGEXPSTATUS[i]. The configuration option BRGEXPDIS defines if each bit within this register is actually implemented such that if BRGEXPDIS[i] = 1'b1, then BRGEXP_EN[i] is disabled and RAZ/WI .	RW	0x0000
15:0	-	Reserved.	RAZ/WI	0x0000

6.4.2.15 MAINNSPPCO

The Main Interconnect Non-secure Access Peripheral Protection Controller Register allows software to configure if each peripheral on the Main Interconnect that it controls through a PPC is Secure access only or is Non-secure access only.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

- '1': Allow Non-secure access only
- '0': Allow Secure access only

CRSAS Ma1 currently do not have any interfaces on the Main Interconnect that need security configuration support of the PPC. Therefore, this register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-23: MAINNSPPC0 bit descriptions

Bits	Name	Description	Type	Default
31:0	-	Reserved.	RAZ/WI	0x0000_0000

6.4.2.16 MAINNSPPCEXP{0-3}

The Main Interconnect Non-secure Access Subordinate Peripheral Protection Controller Expansion register 0, 1, 2, and 3 allow software to configure the security level of each Main Interconnect peripheral that resides in the subsystem expansion outside the subsystem. These registers can be used to control the PPCs - outside the subsystem - that are protecting the Main Interconnect peripherals connected to the Main Interconnect through the Subordinate Main Expansion interfaces.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

- '1': Allow Non-secure access only
- '0': Allow Secure access only

These controls directly control the expansion signals on the Security Control Expansion interface. All four registers are similar and each register x, where x is from 0 to 3, is defined as seen in the Bit descriptions table.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-24: MAINNSPPCEXP<x> bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/WI	0x0000
15:0	MAINNSPPCEXP<x>	Expansion <x> Non-Secure Access Main Interconnect Subordinate Peripheral Protection Control. Each bit <i>i</i> drives the output signal MAINNSPPCEXP<x>[i] . The configuration option MAINPPCEXP<x>DIS defines if each bit within this register is actually implemented such that if MAINPPCEXP<x>DIS[i] = 1'b1, then MAINNSPPCEXP<x>[i] is disabled and RAZ/WI .	RW	0x0000

6.4.2.17 PERIPHSPPCO

The Peripheral Interconnect Non-secure Access Peripheral Protection Controller Registers allows software to configure if each peripheral on the Peripheral Interconnect that it controls through a PPC is Secure access only or is Non-secure access only.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

- '1': Allow Non-secure access only
- '0': Allow Secure access only

See also [PERIPHSPPC1](#).

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-25: PERIPHNSPPC0 bit descriptions

Bits	Name	Description	Type	Default
31:8	-	Reserved.	RAZ/WI	0x000_000
7	NS_SYSDSS	Access Security for interconnect access to Debug System. When HASCSS = 0, this field is reserved and RAZ/WI .	RW	0x0
6	-	Reserved.	RAZ/WI	0x0
5	NS_TIMER3	Access Security for TIMER3	RW	0x0
4	NS_MHU1	Access Security for MHU 1. When NUMCPU = 0, this field is reserved and RAZ/WI .	RW	0x0
3	NS_MHU0	Access Security for MHU 0. When NUMCPU = 0, this field is reserved and RAZ/WI .	RW	0x0
2	NS_TIMER2	Access Security for TIMER2	RW	0x0
1	NS_TIMER1	Access Security for TIMER1	RW	0x0
0	NS_TIMER0	Access Security for TIMER0	RW	0x0



Note

Access to several peripherals filtered by PPC0 though its security setting is fixed and is not represented in the preceding table. Because of this, if such peripheral is accessed using the wrong security it also result in interrupts being raised for PPC0. Full list of peripherals protected by PPC0 are listed in [Peripheral Protection Controllers](#).

6.4.2.18 PERIPHNSPPC1

The Peripheral Interconnect Non-secure Access Peripheral Protection Controller Registers allow software to configure if each peripheral on the Peripheral Interconnect that it controls through a PPC is Secure access only or is Non-secure access only.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

- '1': Allow Non-secure access only
- '0': Allow Secure access only

See also [PERIPHNSPPC0](#).

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-26: PERIPHNSPPC1 bit descriptions

Bits	Name	Description	Type	Default
31:1	-	Reserved.	RAZ/WI	0x0000_0000
0	NS_SLOWCLK_TIMER	Access Security for SLOWCLK_TIMER	RW	0x0



Access to several peripherals filtered by PPC1 though its security setting is fixed and is not represented in the preceding table. Because of this, if such peripheral is accessed using the wrong security it also result in interrupts being raised for PPC1. Full list of peripherals protected by PPC1 are listed in [Peripheral Protection Controllers](#).

6.4.2.19 NPUSPPORSL

The NPU Secure Access Security Level Reset Control Register allows software to configure if each NPU resets to Secure or Non-secure state. The value of this register is only sampled by the NPU, when the NPU is released from reset. The default reset value of this register is controlled by **NPU<m>PORSLRST**.

- '1': Reset to Non-secure state
- '0': Reset to Secure state

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-27: NPUSPPORSL bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	RAZ/WI	0x000_000
3	SP_NPU3PORSL	NPU3 security level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 4.	RW	NPU3PORSLRST
2	SP_NPU2PORSL	NPU2 security level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 3.	RW	NPU2PORSLRST
1	SP_NPU1PORSL	NPU1 security level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 2.	RW	NPU1PORSLRST
0	SP_NPU0PORSL	NPU0 security level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 1.	RW	NPU0PORSLRST

6.4.2.20 PERIPHNSPPCEXP{0-3}

The Peripheral Interconnect Non-secure Access Subordinate Peripheral Protection Controller Expansion registers 0, 1, 2, and 3 allow software to configure the security level of each Peripheral Interconnect peripheral that resides in the expansion logic outside the subsystem.

These registers can be used to control the PPCs - outside the subsystem- that are protecting the Peripheral Interconnect peripherals connected to the Peripheral Interconnect through the Subordinate Peripheral Expansion interfaces.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

- '1': Allow Non-secure access only
- '0': Allow Secure access only

These controls directly control the expansion signals on the Security Control Expansion interface. All four registers are similar and each register x, where x is from 0 to 3, is defined as seen in the Bit descriptions table.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-28: PERIPHSPPCEXP<x> bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/ WI	0x0000
15:0	PERIPHSPPCEXP <x>	Expansion <x> Non-Secure Access Peripheral Interconnect Subordinate Peripheral Protection Control. Each bit x drives the output signal PERIPHSPPCEXP<x>[i] . The configuration option PERIPHPPCEXP<x>DIS defines if each bit within this register is actually implemented such that if PERIPHPPCEXP<x>DIS[i] = 1'b1, then PERIPHSPPCEXP<x>[i] is disabled, reads as zeros and any writes to it is ignored.	RW	0x0000

6.4.2.21 MAINSPPPC0

The Secure Unprivileged Access Main Interconnect Subordinate Peripheral Protection Controller Register allows software to configure if each peripheral on the Main Interconnect that it controls through a PPC is only allowed Secure Privileged Access or is allowed Secure Unprivileged access as well. Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Secure unprivileged and privileged access
- '0': Allow Secure privileged access only

CRSAS Ma1 currently does not have any Main Interconnect subordinate interfaces that need Secure Unprivileged Access configuration support of the PPC. Therefore, this register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-29: MAINSPPPC0 bit descriptions

Bits	Name	Description	Type	Default
31:0	-	Reserved.	RAZ/WI	0x0000_0000

6.4.2.22 MAINSPPPCEXP{0-3}

The Expansion Secure Privileged Access Main Interconnect Subordinate Peripheral Protection Controller Registers 0, 1, 2 and 3 allow software to configure each Main Interconnect peripheral that it controls through each PPC, that resides in the expansion logic outside the subsystem, is Secure privileged Access only or is allowed Secure Unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Secure unprivileged and privileged access
- '0': Allow Secure privileged access only

These settings directly control the expansion signals on the Security Control Expansion interface. All four registers are similar and each register x, where x is from 0 to 3, is defined as seen in the Bit descriptions table.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-30: MAINSPPCEXP<x> bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/ WI	0x0000
15:0	MAINSPPCEXP <x>	Expansion <x> Secure Privileged Access Main Interconnect Subordinate Peripheral Protection Control. Each bit <i>n</i> drives the output signal MAINPPCEXP<x>[i] if MAINSPPCEXP<x>. MAINSPPCEXP<x>[i] is also LOW, where <i>x</i> is 0 to 3. The configuration point MAINPPCEXP<x>DIS defines if each bit within this register is actually implemented such that if MAINPPCEXP<x>DIS[i] = 1'b1, then MAINSPPCEXP<x>[i] is disabled, RAZ/WI and any writes to it is ignored.	RW	0x0000

6.4.2.23 PERIPHSPPPCO

Secure Unprivileged Access Peripheral Interconnect Subordinate Peripheral Protection Controller Register allows software to configure if each Peripheral Interconnect peripheral that it controls through a PPC is only allowed Secure privileged access or is allowed Secure unprivileged access as well. Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Secure Unprivileged and privileged access.
- '0': Allow Secure privileged access only.

CRSAS Ma1 has two such registers, see also [PERIPHSPPPC1](#).

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-31: PERIPHSPPPC0 bit descriptions

Bits	Name	Description	Type	Default
31:8	-	Reserved.	RAZ/WI	0x000_000
7	SP_SYSDSS	Secure privileged setting for interconnect access to Debug System. When HASCSS = 0, this field is reserved and RAZ/WI .	RW	0x0
6	SP_WATCHDOG_REF	Secure System Watchdog Refresh Frame.	RW	0x0
5	SP_TIMER3	Secure privileged setting for TIMER3.	RW	0x0
4	SP_MHU1	Secure privileged setting for MHU 1. When NUMCPU = 0, this field is reserved and RAZ/WI .	RW	0x0
3	SP_MHU0	Secure privileged setting for MHU 0. When NUMCPU = 0, this field is reserved and RAZ/WI .	RW	0x0
2	SP_TIMER2	Secure privileged setting for TIMER2.	RW	0x0
1	SP_TIMER1	Secure privileged setting for TIMER1.	RW	0x0
0	SP_TIMER0	Secure privileged setting for TIMER0.	RW	0x0

6.4.2.24 PERIPHSPPPC1

Secure Unprivileged Access Peripheral Interconnect Subordinate Peripheral Protection Controller Register allows software to configure if each Peripheral Interconnect peripheral that it controls through a PPC is only allowed Secure privileged access or is allowed Secure unprivileged access as well. Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Secure Unprivileged and privileged access.
- '0': Allow Secure privileged access only.

CRSAS Ma1 has two such registers, see also [PERIPHSPPPC0](#).

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-32: PERIPHSPPPC1 bit descriptions

Bits	Name	Description	Type	Default
31:2	-	Reserved.	RAZ/WI	0x0000_0000
0	SP_SLOWCLK_TIMER	Secure privileged setting for SLOWCLK_TIMER.	RW	0x0

6.4.2.25 NPUSPPORPL

The NPU Secure Access Privileged Level Reset Control Register allows software to configure if each NPU resets to privileged or unprivileged state. The value of this register is only sampled by the NPU, when the NPU is released from reset and NPUSPPORSL.SP_NPU<m>PORSL is Secure State. The default reset value of this register is controlled by **NPU<m>PORPLRST**:

- '1': Reset to privileged state
- '0': Reset to unprivileged state

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-33: NPUSPPORPL bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	RAZ/WI	0x0000_000
3	SP_NPU3PORPL	NPU3 privilege level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 4.	RW	NPU3PORPLRST
2	SP_NPU2PORPL	NPU2 privilege level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 3.	RW	NPU2PORPLRST
1	SP_NPU1PORPL	NPU1 privilege level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 2.	RW	NPU1PORPLRST

Bits	Name	Description	Type	Default
0	SP_NPUOPORPL	NPU0 privilege level reset control configuration. This field does not exist, is reserved and RAZ/WI when NUMNPU < 1.	RW	NPU0PORPLRST

6.4.2.26 PERIPHSPPPCEXP{0-3}

The Expansion Secure Privileged Access Peripheral Interconnect Subordinate Peripheral Protection Controller registers 0, 1, 2 and 3 allow software to configure each Peripheral Interconnect peripheral that it controls through each PPC, that resides in the expansion logic outside the subsystem, is allowed Secure privileged Access only or is allowed Secure Unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1' Allow Secure unprivileged and privileged access.
- '0' Allow Secure privileged access only.

These directly control the expansion signals on the Security Control Expansion interface. All four registers are similar and each register x, where x is from 0 to 3, is defined as seen in the Bit descriptions table.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-34: PERIPHSPPPCEXP<x> bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/WI	0x0000

Bits	Name	Description	Type	Default
15:0	PERIPHSPPCEXP<x>	Expansion <x> Secure Privileged Access Peripheral Interconnect Subordinate Peripheral Protection Control. Each bit <i>i</i> drives the output signal PERIPHSPPCEXP<x>[i] if PERIPHNSPPCEXP<x>. PERIPHNSPPCEXP<x>[i] is also LOW, where <i>x</i> is 0 to 3. The configuration option PERIPHPPCEXP<x>DIS defines if each bit within this register is actually implemented such that if PERIPHPPCEXP<x>DIS[i] = 1'b1, then PERIPHSPPCEXP<x>[i] is disabled and RAZ/WI .	RW	0x0000

6.4.2.27 NSMSCEXP

The Non-secure Expansion Manager Security Controller Register allows software to configure if each manager that is located behind each MSC in the subsystem expansion is a Secure or Non-secure device.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Secure Privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-35: NSMSCEXP bit descriptions

Bits	Name	Description	Type	Default
31:16	NS_MSCEXP	Expansion MSC Non-secure Configuration. Each bit <i>i</i> controls the Non-secure configuration of each MSC and drives the signal NSMSCEXP[i] , set HIGH to define a Manager as Non-secure, or LOW for Secure. The parameter MSCEXPDIS defines if each bit within this register is actually implemented such that if MSCEXPDIS[i] = 1'b1 then NS_MSCEXP[i] is disabled and RAOWI . Resets to NSMSCEXPST.	RW	NSMSCEXPST
15:0	-	Reserved.	RAZ/WI	0x0000

6.4.3 Non-secure Access Configuration Register Block

The Non-secure Access Configuration Register Block implements program visible states that allow software to control various security gating units within the design. This register block base address

is 0x4008_0000. These registers are Non-secure privileged access only and support 32-bit RW accesses. For write access to these registers, only 32-bit writes are supported. Any byte and halfword writes are ignored.

All registers reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**.

The following table lists the registers within this unit. Details of each register are described in the following subsections.

Table 6-36: Non-secure Access Configuration Register Block register map

Offset	Name	Type	Reset	Width	Description
0x000-0x08C	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved.
0x090	MAINNSPPPC0	RW	0x0000_0000	32-bit	Non-secure Unprivileged Access Peripheral Protection Control 0 on Main Interconnect, see MAINNSPPPC0 .
0x094-0x09C	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved.
0x0A0	MAINNSPPPCEXP0	RW	0x0000_0000	32-bit	Expansion 0 Non-secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINNSPPPCEXP{0-3} .
0x0A4	MAINNSPPPCEXP1	RW	0x0000_0000	32-bit	Expansion 1 Non-secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINNSPPPCEXP{0-3} .
0x0A8	MAINNSPPPCEXP2	RW	0x0000_0000	32-bit	Expansion 2 Non-secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINNSPPPCEXP{0-3} .
0x0AC	MAINNSPPPCEXP3	RW	0x0000_0000	32-bit	Expansion 3 Non-secure Unprivileged Access Peripheral Protection Control on Main Interconnect, see MAINNSPPPCEXP{0-3} .
0x0B0	PERIPHNSPPPC0	RW	0x0000_0000	32-bit	Non-secure Unprivileged Access Peripheral Protection Control 0 on Peripheral Interconnect, see PERIPHNSPPPC0 .
0x0B4	PERIPHNSPPPC1	RW	0x0000_0000	32-bit	Non-secure Unprivileged Access Peripheral Protection Control 1 on Peripheral Interconnect, see PERIPHNSPPPC1 .
0x0B8	NPUNSPORPL	RW	NPU<m>PORPLRST	32-bit	Non-secure Access NPU Privilege level reset state control, see NPUNSPORPL .
0x0BC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved.
0x0C0	PERIPHNSPPPCEXP0	RW	0x0000_0000	32-bit	Expansion 0 Non-secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHNSPPPCEXP{0-3} .
0x0C4	PERIPHNSPPPCEXP1	RW	0x0000_0000	32-bit	Expansion 1 Non-secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHNSPPPCEXP{0-3} .
0x0C8	PERIPHNSPPPCEXP2	RW	0x0000_0000	32-bit	Expansion 2 Non-secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHNSPPPCEXP{0-3} .
0x0CC	PERIPHNSPPPCEXP3	RW	0x0000_0000	32-bit	Expansion 3 Non-secure Unprivileged Access Peripheral Protection Control on Peripheral Interconnect, see PERIPHNSPPPCEXP{0-3} .
0x0D0-0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4-0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0053	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1

Offset	Name	Type	Reset	Width	Description
0xFE8	PIDR2	RO	0x0000_003B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.4.3.1 MAINNSPPPC0

Non-secure Unprivileged Access Main Interconnect Subordinate Peripheral Protection Controller Register allows software to configure if each peripheral on the Main Interconnect that it controls through a PPC is only allowed Non-secure Privileged Access or is allowed Non-secure Unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Non-secure unprivileged and privileged access
- '0': Allow Non-secure privileged access only

CRSAS Ma1 currently does not have any Main Interconnect subordinate interfaces that need Non-secure Unprivileged Access configuration support of the PPC. Therefore, this register is reserved.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Non-secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-37: MAINNSPPPC0 bit descriptions

Bits	Name	Description	Type	Default
31:0	-	Reserved.	RAZ/WI	0x0000_0000

6.4.3.2 MAINNSPPPCEXP{0-3}

The Expansion Non-secure Unprivileged Access Main Interconnect Subordinate Peripheral Protection Controller registers 0, 1, 2 and 3 allow software to configure each Main Interconnect peripheral that it controls through each PPC, that resides in the expansion logic outside the subsystem, is only Non-secure privileged Access or is allowed Non-secure Unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Non-secure unprivileged and privileged access
- '0': Allow Non-secure privileged access only

These settings directly control the expansion signals on the Security Control Expansion interface. All four register are similar and each register, x where x is from 0 to 3 is as seen in the Bit descriptions table.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Non-secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-38: MAINNSPPPCEXP<x> bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/WI	0x0000
15:0	MAINNSPPPCEXP<x>	Expansion <x> Non-secure Privileged Access Main Interconnect Subordinate Peripheral Protection Control. Each bit <i>i</i> drives the output signal MAINPPPCEXP<x>[i] if MAINPPPCEXP<x>. MAINNSPPPCEXP<x>[i] is also HIGH, where x is 0 to 3. The configuration point MAINPPPCEXP<x>DIS defines if each bit within this register is actually implemented such that if MAINPPPCEXP<x>DIS[i] = 1'b1 then MAINNSPPPCEXP<x>[i] is disabled and RAZ/WI .	RW	0x0000

6.4.3.3 PERIPHNSPPPC0

Non-secure Unprivileged Access Peripheral Interconnect Subordinate Peripheral Protection Controller Register allows software to configure if each Peripheral Interconnect peripheral that it controls through a PPC is only Non-secure privileged access or is allowed Non-secure unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Non-secure unprivileged and privileged access
- '0': Allow Non-secure privileged access only

CRSAS Ma1 has two such registers, see also [PERIPHNSPPPC1](#).

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Non-secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-39: PERIPHNSPPPC0 bit descriptions

Bits	Name	Description	Type	Default
31:8	-	Reserved.	RAZ/WI	0x0000_000
7	NSP_SYSDSS	Non-secure privileged setting for Debug System. When HASCSS = 0, this field is reserved and RAZ/WI .	RW	0x0
6	NSP_WATCHDOG_REF	Secure System Watchdog Refresh Frame.	RW	0x0
5	NSP_TIMER3	Non-secure privileged setting for TIMER3.	RW	0x0
4	NSP_MHU1	Non-secure privileged setting for MHU 1. When NUMCPU = 0, this field is reserved and RAZ/WI .	RW	0x0
3	NSP_MHU0	Non-secure privileged setting for MHU 0. When NUMCPU = 0, this field is reserved and RAZ/WI .	RW	0x0
2	NSP_TIMER2	Non-secure privileged setting for TIMER2.	RW	0x0
1	NSP_TIMER1	Non-secure privileged setting for TIMER1.	RW	0x0
0	NSP_TIMER0	Non-secure privileged setting for TIMER0.	RW	0x0

6.4.3.4 PERIPHNSPPPC1

Non-secure Unprivileged Access Peripheral Interconnect Subordinate Peripheral Protection Controller Register allows software to configure if each Peripheral Interconnect peripheral that it controls through a PPC is only allowed Non-secure privileged access or is allowed Non-secure unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Non-secure unprivileged and privileged access
- '0': Allow Non-secure privileged access only

CRSAS Ma1 has two such registers, see also [PERIPHNSPPPC0](#).

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Non-secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-40: PERIPHNSPPPC1 bit descriptions

Bits	Name	Description	Type	Default
31:1	-	Reserved.	RAZ/WI	0x0000_0000
0	NSP_SLOWCLK_TIMER	Non-secure privileged setting for SLOWCLK_TIMER.	RW	0x0

6.4.3.5 NPUNSPORPL

The NPU power on reset Non-secure access privileged level reset control registers allow software to configure if each NPU resets to privileged or unprivileged state.

The value of this register is only sampled by the NPU, when the NPU is released from reset and **NPUSPPORSL.SP_NPU<m>PORSL** is Non-secure State.

The default reset value of this register is controlled by **NPU<m>PORPLRST**, and it can take the following values:

- '1': Reset to privileged state
- '0': Reset to unprivileged state

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Non-secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-41: NPUNSPORPL bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	RAZ/WI	0x0000_000
3	NS_NPU3PORPL	Configures the Non-secure access privilege level strap value for NPU3. When NUMNPU < 4, This field is reserved and RAZ/WI	RW	NPU3PORPLRST
2	NS_NPU2PORPL	Configures the Non-secure access privilege level strap value for NPU2. When NUMNPU < 3, This field is reserved and RAZ/WI	RW	NPU2PORPLRST
1	NS_NPU1PORPL	Configures the Non-secure access privilege level strap value for NPU1. When NUMNPU < 2, This field is reserved and RAZ/WI	RW	NPU1PORPLRST
0	NS_NPU0PORPL	Configures the Non-secure access privilege level strap value for NPU2. When NUMNPU < 1, This field is reserved and RAZ/WI	RW	NPU0PORPLRST

6.4.3.6 PERIPHNSPPPCEXP{0-3}

The Expansion Non-secure Unprivileged Access Peripheral Interconnect Subordinate Peripheral Protection Controller registers 0, 1, 2, and 3 allow software to configure each Peripheral Interconnect peripheral that it controls through each PPC, that resides in the expansion logic

outside the subsystem, is only allowed Non-secure privileged Access or is allowed Non-secure unprivileged access as well.

Each field defines this for an associated peripheral, by the following settings:

- '1': Allow Non-secure unprivileged and privileged access.
- '0': Allow Non-secure privileged access only.

These directly controls the expansion signals on the Security Control Expansion interface. All four register are similar and each register x, where x is from 0 to 3, is defined as seen in the Bit descriptions table.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

PD_SYS

Reset

This register is reset by **nWARMRESETSYS**.

Usage constraints

This register is Non-secure privileged access only and supports 32-bit RW accesses. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-42: PERIPHNSPPPCEXP<x> bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/ WI	0x0000
15:0	PERIPHNSPPPCEXP<x>	Expansion <x> Non-secure Privileged Access Peripheral Interconnect Subordinate Peripheral Protection Control. Each bit <i>i</i> drives the output signal PERIPHPPPCEXP<x>[i] if PERIPHNSPPPCEXP<x>. PERIPHNSPPPCEXP<x>[i] is also HIGH, where x is 0 to 3 and <i>i</i> is 0 to 15. The configuration option PERIPHPPPCEXP<x>DIS defines if each bit within this register is actually implemented such that if PERIPHPPPCEXP<x>DIS[i] = 1'b1, where x is 0 to 3 and <i>i</i> is 0 to 15, then PERIPHNSPPPCEXP<x>[i] is disabled and RAZ/WI .	RW	0x0000

6.4.4 Timestamp-based timers registers

CRSAS Ma1 implements four timestamp-based timers in the system, **TIMER<x>** where x is 0 to 3. All timers are mapped to the Secure or Non-secure world through PPC0, which also controls accessibility of unprivileged accesses. See [Secure access configuration register block](#).

All timestamp timers, except for Timer3, reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**, while the Timer 3 resides in the PD_AON power domain and is reset by **nWARMRESETAON**.

For more information, see [System timer components](#) appendix.

6.4.5 Timestamp-based Watchdogs registers

CRSAS Ma1 implements two timestamp-based watchdogs in the system. All reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**. One watchdog timer is Secure access only, while another is Non-secure. Each Watchdog Timer implements two register frames, a Control Frame and a Refresh Frame. The Control Frame is always fixed privileged while the Refresh Frame accessibility to unprivileged access is configurable and controlled by PPC0. See [PERIPHSPPPCO](#) and [PERIPHNSPPPCO](#).

For more information, see [System Watchdog overview](#) appendix.

6.4.6 DMA registers

CRSAS Ma1 implements up to one DMA-350. See *Arm® CoreLink™ DMA-350 Controller Technical Reference Manual* for full details of the DMA software interface. Security and privilege checking of accesses to DMA registers are handled by the DMA.

The DMA resides in the PD_SYS power domain and is reset by **nWARMRESETSYS**.

6.4.7 NPU<m> registers

CRSAS Ma1 implements up to four Ethos-U55 NPUs. See *Arm® Ethos™-U55 NPU Technical Reference Manual* for full details of the NPU software interface. Security and privilege checking of accesses to NPU registers are handled by PPC0.

All NPUs reside in the PD_NPU<m> power domain and are reset by **nWARMRESETNPU<m>**.

6.5 CPU Private Region

Each processor in the system has its own copy of the CPU Private Region which is only accessible to itself. Each CPU Private Region consists of four subregions as follows:

0x4001_0000 to 0x4001_FFFF

Implements a Non-secure Low Access Latency Region

0x4801_0000 to 0x4801_FFFF

Implements a Non-secure High Access Latency Region

0x5001_0000 to 0x5001_FFFF

Implements a Secure Low Access Latency Region

0x5801_0000 to 0x5801_FFFF

Implements a Secure High Access Latency Region

Each of these regions is not accessible from any other manager in the system, including from the expansion subordinate interfaces on the Main and Peripheral Interconnect, except through the external debugger through the local CPUs. Of the four preceding regions only 0x4001_0000 to 0x4001_FFFF and 0x5001_0000 to 0x5001_FFFF implements any registers.

The memory map of the CPU Private Region is detailed in the following table.

The CPU Private Region address map defines the following values for security:

S

Secure access only

NS

Non-secure access only

P

Privileged access only

UP

Unprivileged and privileged access allowed

Table 6-43: CPU Private Region address map

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
0	0x4001_0000	0x4001_1FFF	-	Reserved	-	-	Reserved
1	0x4001_2000	0x4001_2FFF	4KB	CPU<n>_PWRCTRL	7	NS, P	CPU <n> Power Control Block. See CPU<n>_PWRCTRL register block .
2	0x4001_3000	0x4001_EFFF		Reserved			Reserved
3	0x4001_F000	0x4001_FFFF	4KB	CPU<n>_IDENTITY	9	NS, UP	CPU <n> Identity Block. See CPU<n>_IDENTITY register block .

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
4	0x4801_0000	0x4801_FFFF	-	Reserved	-	-	Reserved
5	0x5001_0000	0x5001_0FFF	-	Reserved	-	-	Reserved
6	0x5001_1000	0x5001_1FFF	4KB	CPU<n>_SECCTRL	-	S, P	CPU <n> Local Security Control Block. See CPU<n>_SECCTRL register block .
7	0x5001_2000	0x5001_2FFF	4KB	CPU<n>_PWRCTRL	1	S, P	CPU <n> Power Control Block. See CPU<n>_PWRCTRL register block .
8	0x5001_3000	0x5001_EFFF	-	Reserved	-	-	Reserved
9	0x5001_F000	0x5001_FFFF	4KB	CPU<n>_IDENTITY	3	S, UP	CPU <n> Identity Block. See CPU<n>_IDENTITY register block .
10	0x5801_0000	0x5801_FFFF	-	Reserved	-	-	Reserved

6.5.1 CPU<n>_PWRCTRL register block

CRSAS Ma1 implements a CPU<n>_PWRCTRL register block for each CPU <n> in the subsystem. All blocks reside at address 0x4001_2000 in a Non-secure region and are also aliased to 0x5001_2000 in the Secure region. Each CPU <n> can only see its own CPU<n>_PWRCTRL registers. These are read-only registers when accessed from the Non-secure region starting at address 0x4001_2000 and any writes access to it in that region are ignored.

The following table lists the registers in each CPU<n>_PWRCTRL register block.

Table 6-44: CPU<n>_PWRCTRL register map

Offset	Name	Type	Reset	Width	Description
0x000	CPUPWRCFG	RW	0x0000_0000	32-bit	CPU <n> Local Power Configuration register. See CPUPWRCFG .
0x004 – 0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_005A	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_000B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.5.1.1 CPUPWRCFG

The CPUPWRCFG register provides the local CPU software control registers for power control.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read only if accessed from the Non-secure region starting at address 0x4001_2000 and any write accesses to it in that region are ignored.

Power domain

This register resides in the same power domain, PD_CPU<n>, as its associated CPU core so that when the CPU is powered down, the register is also powered down and is cleared when powered back up.

Reset

This register resides in the same reset domain, nWARMRESETCPU<n>, as its associated CPU core so that when the CPU is powered down, the register is also powered down and is cleared when powered back up.

Usage constraints

Each CPU <n> can only see its own CPU<n>_PWRCTRL registers.

Bit descriptions

Table 6-45: CPUPWRCFG bit descriptions

Bits	Name	Description	Type	Default
31:5	-	Reserved	RAZ/ WI	0x000_0000
4	TCM_MIN_PWR_STATE	<p>Defines the minimum power state of the TCM for CPU<n>.</p> <ul style="list-style-type: none"> '0': OFF, '1': Retention. <p>This bit is read access only from the Non-secure world.</p> <p>When PD_CPU<n> returns from MEM_RET or MEM_RET_NOCACHE state to one of the ON states, this bit is set to 1'b1.</p>	RW	0x0
3:1	-	Reserved	RAZ/ WI	0x0

Bits	Name	Description	Type	Default
0	USEIWIC	When HIGH selects the use of IWIC for CPU <n> when in DeepSleep. Else selects the use of EWIC. If HASCPU<n>HASIWIC for this CPU <n> is 0, this field is reserved and RAZ/WI . This bit is read access only from the Non-secure world.	RW	0x0

6.5.2 CPU<n>_IDENTITY register block

CRSAS Ma1 implements a CPU<n>_IDENTITY register block for each CPU <n> in the subsystem. All blocks reside at address 0x4001_F000 in a Non-secure region and are also aliased to 0x5001_F000 in the Secure region. Each CPU <n> can only see its own CPU<n>_IDENTITY registers. These are read-only registers and any write accesses to the region are ignored.

The following table lists the registers in each CPU<n>_IDENTITY block.

Table 6-46: CPU<n>_IDENTITY register map

Offset	Name	Type	Reset	Width	Description
0x000	CPUID	RO	CPU<n>CPUIDRST	32-bit	Unique CPU Identity Number, where <n> is used for the view that CPU <n> sees. See CPUID .
0x004 – 0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0055	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_000B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.5.2.1 CPUID

The CPUID Register is a read only register that, when read by CPU <n>, provides an identity code to the CPU that is unique to that CPU.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read only and any write accesses to it are ignored.

Power domain

This register resides in the same power domain, PD_CPU<n>, as its associated CPU core so that when the CPU is powered down, the register is also powered down and is cleared when powered back up.

Reset

This register resides in the same reset domain, **nWARMRESETCPU<n>**, as its associated CPU core so that when the CPU is powered down, the register is also powered down and is cleared when powered back up.

Usage constraints

Each CPU <n> can only see its own CPU<n>_IDENTITY registers.

Bit descriptions

Table 6-47: CPUID bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved	RAZ/WI	0x000_0000
3:0	CPUID	CPU Identity. Defined by configuration CPU<n>CPUIDRST. The identity value for each CPU <n> must be unique.	RO	CPU<n>CPUIDRST

6.5.3 CPU<n>_SECCTRL register block

Each CPU <n> in the system has associated with it a CPU<n>_SECCTRL register block that allows the security locks of each CPU to be configured. Each register block resides in the same reset domain, **nWARMRESETCPU<n>**, and power domain as its associated CPU core so that when a CPU is powered down, they are also powered down and are cleared when powered back up. These registers are Secure access only and reside at address 0x5001_1000.

The following table lists the registers in each CPU<n>_SECCTRL Register block.

Table 6-48: CPU<n>_SECCTRL Register Map

Offset	Name	Type	Reset	Width	Description
0x000	CPUSECCFG	RW	0x0000_0000	32-bit	CPU Local Security Configuration. See CPUSECCFG .
0x004 – 0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0059	32-bit	Peripheral ID 0

Offset	Name	Type	Reset	Width	Description
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_001B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.5.3.1 CPUSECCFG

The CPU Local Security Configuration Register allows software to set security lock bits at the interface of each associated CPU<n>.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is secure access only.

Power domain

This register resides in the same power domain as its associated CPU core so that when the CPU is powered down, the register is also powered down and is cleared when powered back up.

Reset

This register resides in the same reset domain, **nWARMRESETCPU<n>**, as its associated CPU core so that when the CPU is powered down, the register is also powered down and is cleared when powered back up.

Bit descriptions

Table 6-49: CPUSECCFG bit descriptions

Bits	Name	Description	Type	Default
31:6	-	Reserved.	RAZ/WI	0x0000_0000
5	LOCKDTGU	When HIGH, disables writes to the CPU <n> DTGU_CTRL and DTGU_LUTn registers from software or from a debug agent connected to the processor. Once set to HIGH, it cannot be cleared until Reset.	RW1S	0x0
4	LOCKITGU	When HIGH, disables writes to the CPU <n> ITGU_CTRL and ITGU_LUTn from software or from a debug agent connected to the processor. Once set to HIGH, it cannot be cleared until Reset.	RW1S	0x0

Bits	Name	Description	Type	Default
3	LOCKTCM	When HIGH, disables writes to the CPU <n> ITCMCR, DTCMCR from software or from a debug agent connected to the processor. Once set to HIGH, it cannot be cleared until Reset.	RW1S	0x0
2	LOCKSMPU	When HIGH, disables write to the CPU <n> MPU_CTRL, MPU_RNR, MPU_RBAR, MPU_RLAR, MPU_RBAR_An, MPU_RLAR_An registers associated with the Secure MPU from software or from a debug agent connected to the processor. Once set to HIGH, it cannot be cleared until Reset.	RW1S	0x0
1	LOCKSAU	When HIGH, disables writes to the CPU <n> SAU_CTRL, SAU_RNR, SAU_RBAR and SAU_RLAR registers from software or from a debug agent connected to the processor. Once set to HIGH, it cannot be cleared until Reset.	RW1S	0x0
0	LOCKSVTAIRCR	When HIGH, disables writes to the CPU <n> VTOR_S, AIRCR.PRIS, and AIRCR.BFHFNMINS registers. Once set to HIGH, it cannot be cleared until Reset.	RW1S	0x0

6.6 System Control Peripheral Region

The System Control Peripheral Regions are a collection of memory regions where system control related peripherals are mapped. These peripherals reside either in the PD_AON domain or in the PD_MGMT domain if PILEVEL = 2. There are four regions in total as follows:

0x4002_0000 to 0x4003_FFFF

Non-secure region for low latency system control peripherals. Some peripherals may be expected to be aliased in its associated Secure region, 0x5002_0000 to 0x5003_FFFF.

0x4802_0000 to 0x4803_FFFF

Non-secure region for high latency system control peripherals. Some peripherals may be expected to be aliased in its associated Secure region, 0x5802_0000 to 0x5803_FFFF.

0x5002_0000 to 0x5003_FFFF

Secure region for low latency system control peripherals. Some peripherals may be expected to be aliased in its associated Non-secure region, 0x4002_0000 to 0x4003_FFFF.

0x5802_0000 to 0x5803_FFFF

Secure region for low latency system control peripherals. Some peripherals may be expected to be aliased in its associated Non-secure region, 0x4802_0000 to 0x4803_FFFF.

For an aliased peripheral in these regions, mapping of each peripheral to either Secure or Non-secure region is determined by Peripheral Protection Controllers (PPC) that are controlled using the Secure Access Configuration register block. These PPCs also define privileged or unprivileged accessibility. For more information, [Secure Access Configuration Register Block](#).

The following table shows the System Control Peripheral Region address map.

The System Control Peripheral Region address map defines the following values for security:

NS_PPC

Non-secure access only, gated by a PPC

S_PPC

Secure access only, gated by a PPC

S

Secure access only

NS

Non-secure access only

P

Privileged access only

UP

Unprivileged and privileged access allowed

P_PPC

Unprivileged access controlled by PPC



Note

When PPC protects a peripheral the configurability of the security or privileged attributes for a given peripheral is defined by [PERIPHSPPPC0](#), [PERIPHSPPPC1](#), [PERIPHNSPPPC0](#), [PERIPHNSPPPC1](#), [PERIPHNSPPC0](#), and [PERIPHNSPPC1](#) registers.

Table 6-50: System Control Peripheral Region address map

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
1	0x4002_0000	0x4003_FFFF	128KB	Reserved	-	-	Reserved
2	0x4802_0000	0x4802_0FFF	4KB	SYSINFO	7	NS, UP	System Information Register Block. See SYSINFO register block .
3	0x4802_1000	0x4802_EFFF	56KB	Reserved	-	NS	Reserved. When accessed, results in RAZ/WI .
4	0x4802_F000	0x4802_FFFF	4KB	SLOWCLK Timer	31	NS_PPC, P_PPC	Timer running on SLOWCLK . See SLOWCLK AON timers .
5	0x4803_0000	0x4803_FFFF	64KB	Reserved	-	-	Reserved
6	0x5002_0000	0x5003_FFFF	128KB	Reserved	-	-	Reserved
7	0x5802_0000	0x5802_0FFF	4KB	SYSINFO	2	S, UP	System Information Register Block. See SYSINFO register block .
8	0x5802_1000	0x5802_1FFF	4KB	SYSCONTROL	-	S_PPC, P_PPC	System Control Register Block. See System control register block .
18	0x5802_2000	0x5802_2FFF	4KB	SYS_PPU	-	S_PPC, P_PPC	PPU for BR_SYS. See Power Policy Units .
19	0x5802_3000	0x5802_3FFF	4KB	CPU0_PPU	-	S_PPC, P_PPC	PPU for BR_CPU0. See Power Policy Units .
20	0x5802_4000	0x5802_4FFF	4KB	CPU1_PPU ¹	-	S_PPC, P_PPC	PPU for BR_CPU1. See Power Policy Units .
21	0x5802_5000	0x5802_5FFF	4KB	CPU2_PPU ²	-	S_PPC, P_PPC	PPU for BR_CPU2. See Power Policy Units .
22	0x5802_6000	0x5802_6FFF	4KB	CPU3_PPU ³	-	S_PPC, P_PPC	PPU for BR_CPU3. See Power Policy Units .

Row ID	From address	To address	Size	Region name	Alias with row ID	Security	Description
23	0x5802_7000	0x5802_7FFF	4KB	CRYPTO_PPU ⁴	-	S_PPC, P_PPC	PPU for BR_CRYPT. See Power Policy Units .
24	0x5802_8000	0x5802_8FFF	4KB	MGMT_PPU	-	S_PPC, P_PPC	PPU for BR_MGMT. See Power Policy Units .
25	0x5802_9000	0x5802_9FFF	4KB	DEBUG_PPU	-	S_PPC, P_PPC	PPU for BR_DEBUG. See Power Policy Units .
26	0x5802_A000	0x5802_AFFF	4KB	NPU0_PPU ⁵	-	S_PPC, P_PPC	PPU for BR_NPU0. See Power Policy Units .
27	0x5802_B000	0x5802_BFFF	4KB	NPU1_PPU ⁶	-	S_PPC, P_PPC	PPU for BR_NPU1. See Power Policy Units .
28	0x5802_C000	0x5802_CFFF	4KB	NPU2_PPU ⁷	-	S_PPC, P_PPC	PPU for BR_NPU2. See Power Policy Units .
29	0x5802_D000	0x5802_DFFF	4KB	NPU3_PPU ⁸	-	S_PPC, P_PPC	PPU for BR_NPU3. See Power Policy Units .
30	0x5802_E000	0x5802_EFFF	4KB	SLOWCLK Watchdog	-	S_PPC, P_PPC	Watchdog Timer running on SLOWCLK . See SLOWCLK AON timers .
31	0x5802_F000	0x5802_FFFF	4KB	SLOWCLK Timer	4	S_PPC, P_PPC	Timer running on SLOWCLK . See SLOWCLK AON timers .
32	0x5803_0000	0x5803_FFFF	64KB	Reserved	-	-	Reserved

¹ When NUMCPU < 1, the CPU1_PPU region is reserved and **RAZ/WI**.

² When NUMCPU < 2, the CPU2_PPU region is reserved and **RAZ/WI**.

³ When NUMCPU < 3, the CPU3_PPU region is reserved and **RAZ/WI**.

⁴ When HASCRYPTO < 1, the CRYPTO_PPU region is reserved and **RAZ/WI**.

⁵ When NUMNPU < 1, the NPU0_PPU region is reserved and **RAZ/WI**.

⁶ When NUMNPU < 2, the NPU1_PPU region is reserved and **RAZ/WI**.

⁷ When NUMNPU < 3, the NPU2_PPU region is reserved and **RAZ/WI**.

⁸ When NUMNPU < 4, the NPU3_PPU region is reserved and **RAZ/WI**.

6.6.1 SYSINFO Register Block

The System Information Register Block provides information on the system configuration and identity. This register block is read-only and is accessible by accesses of any security attributes. This module resides at base address 0x5802_0000 in the Secure region, and 0x4802_0000 in the Non-secure region.

Details of each register are described in the following subsections:

Table 6-51: System Information register map

Offset	Name	Type	Reset	Width	Description
0x000	SOC_IDENTITY	RO	CFG_DEF	32-bit	See SOC_IDENTITY .
0x004	SYS_CONFIG0	RO	CFG_DEF	32-bit	System Hardware Configuration 0 Register. See SYS_CONFIG0 .
0x008	SYS_CONFIG1	RO	CFG_DEF	32-bit	System Hardware Configuration 1 Register. See SYS_CONFIG1 .
0x00C	SYS_CONFIG2	RO	CFG_DEF	32-bit	System Hardware Configuration 2 Register. See section SYS_CONFIG2 .
0x010 – 0xFC4	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFC8	IIDR	RO	CFG_DEF	32-bit	Subsystem Implementation Identity Register. See IIDR .
0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0058	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_002B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.6.1.1 SOC_IDENTITY

The System-On-Chip (SoC) Identity Register provides an area where software can find out about the SoC's part number, its implementor and revision number. These are defined by configuration options that are expected to be set by a SoC integrator to identify the SoC.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read-only and is accessible by accesses of any security attributes.

Bit descriptions

Table 6-52: SOC_IDENTITY bit descriptions

Bits	Name	Description	Type	Default
31:20	SOC_PRODUCT_ID	IMPL_DEF value identifying the SoC.	RO	SOCPRID
19:16	SOC_VARIANT	IMPL_DEF value variant or major revision of the SoC.	RO	SOCVAR
15:12	SOC_REVISION	IMPL_DEF value used to distinguish minor revisions of the SoC.	RO	SOCREV
11:0	SOC_IMPLEMENTATOR	Contains the JEP106 code of the company that implemented the SoC: <ul style="list-style-type: none"> [11:8] JEP106 continuation code of implementer. [7] Always 0. [6:0] JEP106 identity code of implementer. 	RO	SOCIMPLID

We recommend that the SoC Identity values are also used to identify the SoC or the complete subsystem to the debugger in one or both of the following ways:

1. If the system is a standalone system, the SoC Identity values can be used to generate the TARGETID of the SoC Debug Port in the expansion system, as follows:
 - TARGETID[31:28] uses SOCVAR,
 - TARGETID[27:16] uses SOCPRID,
 - TARGETID[15:12] tied to 0x00
 - TARGETID[11:1] uses {SOCIMPLID[11:8], SOCIMPLID[6:0]}.
 - TARGETID[0] tied to 0b1.

For more information on TARGETID, see *Arm® Debug Interface Architecture Specification ADIV6.0*.

2. The SoC Identity values can also be used in to define the PIDR values of the first Debug ROM in the expansion system that the debugger sees for this system, as follows:
 - REVISION uses SOCVAR,
 - {PART_1, PART_0} uses SOCPRID,
 - {DES_2, DES_1, DES_0} uses SOCIMPLID,
 - REVAND uses SOCREV,
 - CMOD set to 0x0.

For more information on PIDR registers of CoreSight Debug ROM, see *Arm® CoreSight™ Architecture Specification v3.0*.

6.6.1.2 SYS_CONFIG0

The System Hardware Configuration Registers provide several registers that allow software to query the configuration of the CRSAS Ma1 based subsystem. See also [SYS_CONFIG1](#) and [SYS_CONFIG2](#).

In the following table, the fields CPU<n>_TCM_BANK_NUM and CPU<n>_HAS_SYSTCM refer to TCMs that are implemented on the system interconnect close to each associated CPU, rather than the TCMs that are implemented within the CPU core. CRSAS Ma1 currently does not support TCMs being implemented on the system interconnect and hence these fields are reserved and **RAZ/WI**.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read-only and is accessible by accesses of any security attributes.

Bit descriptions

Table 6-53: SYS_CONFIG0 bit descriptions

Bits	Name	Description	Type	Default
31:28	CPU1_TCM_BANK_NUM	The VM Bank that is the TCM memory for CPU 1.	RO	0x0
27	CPU1_HAS_SYSTCM	CPU 1 has System TCM: <ul style="list-style-type: none"> • '0': No • '1': Yes This is not the CPU's local ITCM or DTCM, but instead these are the TCMs that are implemented at system level.	RO	0x0
26:24	CPU1_TYPE	CPU 1 Core Type: <ul style="list-style-type: none"> • '000': Does not exist • '011': Cortex-M55 Processor • '100': Cortex-M85 • Others: Reserved 	RO	CPU1TYPE
23:20	CPU0_TCM_BANK_NUM	The VM Bank that is the TCM memory for CPU 0.	RO	0x0
19	CPU0_HAS_SYSTCM	CPU 0 has System TCM: <ul style="list-style-type: none"> • '0' = No • '1' = Yes. This is not the CPU's local ITCM or DTCM, but instead these are the TCMs that are implemented at system level.	RO	0x0

Bits	Name	Description	Type	Default
18:16	CPU0_TYPE	CPU 0 Core Type: <ul style="list-style-type: none"> '000': Does not exist '011': Cortex-M55 Processor '100': Cortex-M85 Others: Reserved 	RO	CPU0TYPE
15:13	Reserved	Reserved	RO	0x0
12:11	PI_LEVEL	Power Infrastructure Level: <ul style="list-style-type: none"> '00': Basic Level '01': Intermediate Level '10': Advanced Level Others: Reserved. 	RO	PILEVEL
10	HAS_CSS	Includes CoreSight SoC-600 based Debug infrastructure. <ul style="list-style-type: none"> '0': No '1': Yes 	RO	HASCSS
9	HAS_CRYPT0	Includes CryptoCell: <ul style="list-style-type: none"> '0': No '1': Yes 	RO	HASCRIPT0
8:4	VM_ADDR_WIDTH	Volatile Memory Bank Address Width, where the size of each bank is equal to $2^{\text{VM_ADDR_WIDTH}}$ bytes.	RO	VMADDRWIDTH
3:0	NUM_VM_BANK	Number of Volatile Memory Banks.	RO	NUMVMBANK

6.6.1.3 SYS_CONFIG1

The System Hardware Configuration Registers provide several registers to allow software to find out about the configuration of the CRSAS Ma1 based subsystem. See also [SYS_CONFIG0](#) and [SYS_CONFIG2](#).

In the following table, the fields CPU<n>_TCM_BANK_NUM and CPU<n>_HAS_SYSTCM refer to TCMs that are implemented on the system interconnect close to each associated CPU, rather than the TCMs that are implemented within the CPU core. CRSAS Ma1 currently does not support TCMs being implemented on the system interconnect and hence these fields are reserved and **RAZ/WI**.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read-only and is accessible by accesses of any security attributes.

Bit descriptions

Table 6-54: SYS_CONFIG1 bit descriptions

Bits	Name	Description	Type	Default
31:16	Reserved	Reserved	RAZ/ WI	0x0000
15:12	CPU3_TCM_BANK_NUM	The VM Bank that is the TCM memory for CPU 3.	RO	0x0
11	CPU3_HAS_SYSTCM	CPU 3 has System TCM: <ul style="list-style-type: none"> '0': No '1': Yes <p>Note that this is not the CPU's local ITCM or DTCM, but instead these are the TCMs that are implemented at system level.</p>	RO	0x0
10:8	CPU3_TYPE	CPU 3 Core Type: <ul style="list-style-type: none"> '000': Does not exist '011': Cortex-M55 Processor '100': Cortex-M85 Others: Reserved. 	RO	CPU3TYPE
7:4	CPU2_TCM_BANK_NUM	The VM Bank that is the TCM memory for CPU 2.	RO	0x0
3	CPU2_HAS_SYSTCM	CPU 2 has System TCM. <ul style="list-style-type: none"> '0': No '1': Yes <p>Note that this is not the CPU's local ITCM or DTCM, but instead these are the TCMs that are implemented at system level.</p>	RO	0x0
2:0	CPU2_TYPE	CPU 2 Core Type: <ul style="list-style-type: none"> '000': Does not exist '011': Cortex-M55 Processor '100': Cortex-M85 Others: Reserved. 	RO	CPU2TYPE

6.6.1.4 SYS_CONFIG2

The System Hardware Configuration Registers provide several registers to allow software to find out about the configuration of the CRSAS Ma1 based subsystem. See also [SYS_CONFIG0](#) and [SYS_CONFIG1](#).

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read-only and is accessible by accesses of any security attributes.

Bit descriptions

Table 6-55: SYS_CONFIG2 bit descriptions

Bits	Name	Description	Type	Default
31:15	Reserved	Reserved	RAZ/WI	0x0000
14:12	DMA_TYPE	DMA Core Type: <ul style="list-style-type: none"> '000': Does not exist '001': DMA-350 Others: Reserved 	RO	DMATYPE
11:9	NPU3_TYPE	NPU 3 Core Type: <ul style="list-style-type: none"> '000': Does not exist '001': Ethos-U55 Others: Reserved 	RO	NPU3TYPE
8:6	NPU2_TYPE	NPU 2 Core Type: <ul style="list-style-type: none"> '000': Does not exist '001': Ethos-U55 Others: Reserved 	RO	NPU2TYPE
5:3	NPU1_TYPE	NPU 1 Core Type: <ul style="list-style-type: none"> '000': Does not exist '001': Ethos-U55 Others: Reserved 	RO	NPU1TYPE
2:0	NPU0_TYPE	NPU 0 Core Type: <ul style="list-style-type: none"> '000': Does not exist '001': Ethos-U55 Others: Reserved 	RO	NPU0TYPE

6.6.1.5 IIDR

The subsystem Implementation Identity Register provides an area where software can find out about the subsystem Implementation part number, its implementor and revision number. These are defined by configuration options that are expected to be set by the subsystem implementor.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Type

This register is read-only and is accessible by accesses of any security attributes.

Bit descriptions

Table 6-56: IIDR bit descriptions

Bits	Name	Description	Type	Default
31:20	IMP_PRODUCT_ID	IMPL_DEF value identifying the subsystem implementation.	RO	IMPLPRTID
19:16	IMP_VARIANT	IMPL_DEF value variant or major revision of the subsystem implementation.	RO	IMPLVAR
15:12	IMP_REVISION	IMPL_DEF value used to distinguish minor revisions of the subsystem implementation.	RO	IMPLREV
11:0	IMP_IMPLEMENTATOR	Contains the JEP106 code of the company that implemented the subsystem implementation: <ul style="list-style-type: none"> [11:8] JEP106 continuation code of implementer. [7] Always 0. [6:0] JEP106 identity code of implementer. 	RO	IMPLID

6.6.2 System Control Register Block

The System Control Register Block implements registers for power, clocks, resets, and other general system control. This module resides at base address 0x5802_1000 in the Secure region. The System Control Register Block is Secure privileged access only. For write access to these registers, only 32-bit writes are supported. Any byte and halfword writes results in its write data ignored.

Most System Control registers reside in the PD_MGMT power domain when PILEVEL = 2, or is merged into PD_AON when PILEVEL < 2. When entering lower power states, some of the register values must be retained. How retention is achieved is IMPLEMENTATION DEFINED. Other registers not in PD_MGMT reside instead in the PD_AON domain.

The following table shows the details of this register block.

Table 6-57: System Control Register Map

Offset	Name	Type	Reset	Width	Description
0x000	SECDBGSTAT	RO	CFG_DEF	32-bit	Secure Debug Configuration Status Register. See SECDBGSTAT .
0x004	SECDBGSET	RW	0x0000_0000	32-bit	Secure Debug Configuration Set Register. See SECDBGSET .
0x008	SECDBGCLR	WO	0x0000_0000	32-bit	Secure Debug Configuration Clear Register. See SECDBGCLR .
0x00C	SCSECCTRL	RW	0x0000_0000	32-bit	System Control Security Controls Register. See SCSECCTRL .
0x010	CLK_CFG0	RW	CFG_DEF	32-bit	Clock Configuration Register 0. See CLK_CFG0 .
0x014	CLK_CFG1	RW	CFG_DEF	32-bit	Clock Configuration Register 1. See CLK_CFG1 .

Offset	Name	Type	Reset	Width	Description
0x018	CLOCK_FORCE	RW	CFG_DEF	32-bit	Clock Forces. See CLOCK_FORCE .
0x1C	CLK_CFG2	RW	CFG_DEF	32-bit	Clock Configuration Register 1. See CLK_CFG2 .
0x020 – 0x0FF	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0x100	RESET_SYNDROME	RW	0x0000_0001	32-bit	Reset syndrome. See RESET_SYNDROME .
0x104	RESET_MASK	RW	CFG_DEF	32-bit	Reset Mask. See RESET_MASK .
0x108	SWRESET	WO	0x0000_0000	32-bit	Software Reset. See SWRESET .
0x10C	GRETREG	RW	0x0000_0000	32-bit	General Purpose Retention Register. See GRETREG .
0x110	INITSVTOR0 ¹	RW	CFG_DEF	32-bit	CPU 0 Initial Secure Reset Vector Register. See INITSVTOR<n> .
0x114	INITSVTOR1 ¹	RW	CFG_DEF	32-bit	CPU 1 Initial Secure Reset Vector Register. See INITSVTOR<n> .
0x118	INITSVTOR2 ²	RW	CFG_DEF	32-bit	CPU 2 Initial Secure Reset Vector Register. See INITSVTOR<n> .
0x11C	INITSVTOR3 ³	RW	CFG_DEF	32-bit	CPU 3 Initial Secure Reset Vector Register. See INITSVTOR<n> .
0x120	CPUWAIT	RW	CFG_DEF	32-bit	CPU Boot Wait Control. See CPUWAIT .
0x124	NMI_ENABLE	RW	CFG_DEF	32-bit	Enabling and Disabling Non Maskable Interrupts. See NMI_ENABLE .
0x128	PPUINTSTAT	RO	0x0000_0000	32-bit	PPU Interrupt Status. See PPUINTSTAT .
0x12C – 0x1F8	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0x1FC	PWRCTRL	RW	0x0000_0003	32-bit	Power Configuration and Control. See PWRCTRL .
0x200	PDCM_PD_SYS_SENSE	RW	CFG_DEF	32-bit	PDCM PD_SYS Sensitivity. See PDCM_PD_SYS_SENSE .

Offset	Name	Type	Reset	Width	Description
0x204	PDCM_PD_CPU0_SENSE	RO	0x0000_0000	32-bit	PDCM PD_CPU0 Sensitivity. See PDCM_PD_CPU<n>_SENSE .
0x208	PDCM_PD_CPU1_SENSE ¹	RO	0x0000_0000	32-bit	PDCM PD_CPU1 Sensitivity. See PDCM_PD_CPU<n>_SENSE .
0x20C	PDCM_PD_CPU2_SENSE ²	RO	0x0000_0000	32-bit	PDCM PD_CPU2 Sensitivity. See PDCM_PD_CPU<n>_SENSE .
0x210	PDCM_PD_CPU3_SENSE ³	RO	0x0000_0000	32-bit	PDCM PD_CPU3 Sensitivity. See PDCM_PD_CPU<n>_SENSE .
0x214	PDCM_PD_VMR0_SENSE ⁴	RW	0x4000_0000	32-bit	PDCM PD_VMR0 Sensitivity. See PDCM_PD_VMR<x>_SENSE .
0x218	PDCM_PD_VMR1_SENSE ⁵	RW	0x4000_0000	32-bit	PDCM PD_VMR1 Sensitivity. See PDCM_PD_VMR<x>_SENSE .
0x21C	PDCM_PD_VMR2_SENSE ⁶	RW	0x4000_0000	32-bit	PDCM PD_VMR2 Sensitivity. See PDCM_PD_VMR<x>_SENSE .
0x220	PDCM_PD_VMR3_SENSE ⁷	RW	0x4000_0000	32-bit	PDCM PD_VMR3 Sensitivity. See PDCM_PD_VMR<x>_SENSE .
0x224 – 0x248	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0x24C	PDCM_PD_MGMT_SENSE	RW	CFG_DEF	32-bit	PDCM PD_MGMT Sensitivity. See PDCM_PD_MGMT_SENSE .
0x250 – 0xFCC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved Future RSS / system state retention
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_0054	32-bit	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	32-bit	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_004B	32-bit	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

¹ These registers do not exist and are reserved if NUMCPU < 1:

- PDCM_PD_CPU1_SENSE
- INITSVTOR0

- INITSVTOR1

² These registers do not exist and are reserved if NUMCPU < 2:

- INITSVTOR2
- PDCM_PD_CPU2_SENSE

³ These registers do not exist and are reserved if NUMCPU < 3:

- INITSVTOR3
- PDCM_PD_CPU3_SENSE

⁴ These registers do not exist and are reserved if NUMVMBANK < 1:

- PDCM_PD_VMRO_SENSE

⁵ These registers do not exist and are reserved if NUMVMBANK < 2:

- PDCM_PD_VMR1_SENSE

⁶ These registers do not exist and are reserved if NUMVMBANK < 3:

- PDCM_PD_VMR2_SENSE

⁷ These registers do not exist and are reserved if NUMVMBANK < 4:

- PDCM_PD_VMR3_SENSE

6.6.2.1 Secure Debug Configuration Registers

The Secure Debug Configuration Registers are used to select the source value for the Secure Debug Authentication, **DBGEN**, **NIDEN**, **SPIDEN**, **SPNIDEN**, **DAPACCEN**, and Debug Access Controls, **DAPDSSACCEN**, **SYSDSSACCENX**, and **SYSDSSACCEN<n>**. For each signal and just one for all SYSDSSACCEN<n> and SYSDSSACCENX, a selector is provided to select between an internal register value and the value on the boundary of the subsystem.

Secure software can set or clear the internal register and selector values by setting the associated bit in the SECDBGSET register or in the SECDBGCLR register respectively. Secure software can read the output values used system wide by reading the associated SECDBGSTAT register bit. Secure software can read internal register values by reading SECDBGSET, see [SECDBGSET](#).

For example, the source of DBGEN value used in the system is selected by the DBGEN_SEL where:

- If DBGEN_SEL is LOW, the input **DBGENIN** signal is used to define the system wide **DBGEN** value.
- If DBGEN_SEL is HIGH, the internal register value DBGEN_I is used to define the system wide **DBGEN** value.

Write 1 to the SECDBGSET.DBGEN_I_SET register to set DBGEN_I value to HIGH. Write 1 to SECDBGSET.DBGEN_SEL_SET to set DBGEN_SEL value to HIGH.

Write 1 to the SECDDBGCLR.DBGEN_I_CLR register to set DBGEN_I value to LOW. Write 1 to SECDDBGCLR.DBGEN_SEL_CLR to set DBGEN_SEL to LOW.

Read the SECDBGSTAT.DBGEN_STATUS register to read the output value of **DBGEN**. Read SECDBGSET.DBGEN_I_SET register to read the value of DBGEN_I.

The **DBGEN** value is also made available to external expansion logic through the **DBGEN** output signal of the subsystem.

Selector Disable Configuration options are provided to allow each of the selector to be forced to zero, forcing the associated SEL_STATUS field to LOW, forcing each respective debug control output to use its external value:

- DBGENSELDIS for disabling DBGEN_SEL,
- NIDENSELDIS for disabling NIDEN_SEL,
- SPIDENSELDIS for disabling SPIDEN_SEL,
- SPNIDENSELDIS for disabling SPNIDEN_SEL,
- DAPACCENSELDIS for disabling DAPACCEN_SEL,
- DAPDSSACCENSELDIS for disabling DAPDSSACCEN_SEL,
- SYSDSSACCENSELDIS for disabling SYSDSSACCEN_SEL.

These can be used to disable the ability for Secure firmware to modify or override the Debug Authentication and the Debug Access Controls values, especially when CryptoCell exists (HASCRYPTO = '1') in the system and the intention is to use signals derived from **CRYPTODCUEN** to control debug instead.

These registers are reset by **nCOLDRESETAON** only.

These registers reside in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively. This choice is **IMPLEMENTATION DEFINED**.

6.6.2.1.1 SECDBGSTAT

Secure Debug Configuration Status Register.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain, if its states are saved and restored when entering and then leaving the lower power state, respectively. This choice is **IMPLEMENTATION DEFINED**.

Reset

This register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-58: SECDBGSTAT bit descriptions

Bits	Name	Description	Type	Default
31	-	Reserved.	RAZ/WI	0x0
30	SYSDSSACCENSELDIS_STATUS	Returns the SYSDSSACCENSELDIS configuration value when read. Ignores Writes. Reserved and RAZ/WI if HASCSS = 0.	RO	SYSDSSACCENSELDIS
29	DAPDSSACCENSELDIS_STATUS	Returns the DAPDSSACCENSELDIS configuration value when read.	RO	DAPDSSACCENSELDIS
28	DAPACCENSELDIS_STATUS	Returns the DAPACCENSELDIS configuration value when read.	RO	DAPACCENSELDIS
27	SPNIDENSELDIS_STATUS	Returns the SPNIDENSELDIS configuration value when read.	RO	SPNIDENSELDIS
26	SPIDENSELDIS_STATUS	Returns the SPIDENSELDIS configuration value when read.	RO	SPIDENSELDIS
25	NIDENSELDIS_STATUS	Returns the NIDENSELDIS configuration value when read.	RO	NIDENSELDIS
24	DBGENSELDIS_STATUS	Returns the DBGENSELDIS configuration value when read.	RO	DBGENSELDIS
23:18	-	Reserved.	RAZ/WI	0x00
17	SYSDSSACCEN_SEL_STATUS	Active-HIGH System Mapped Debug Access Enable Selector Value. Used as a common selector for all SYSDSSACCEN<n>_STATUS. This bit returns the SYSDSSACCEN_SEL value Forced to Zero if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0.	RO	0x0
16	SYSDSSACCENX_STATUS	Active-HIGH System Mapped Debug Access for Implementation Defined Manager(s) . Reserved and RAZ/WI if HASCSS = 0. This bit reflects the value on the SYSDSSACCENX pin.	RO	SYSDSSACCENX
15	SYSDSSACCEN3_STATUS	Active-HIGH System Mapped Debug Access for CPU 3 Enable Value. This bit reflects the value on the SYSDSSACCEN3 after selection. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 3.	RO	SYSDSSACCEN3
14	SYSDSSACCEN2_STATUS	Active-HIGH System Mapped Debug Access for CPU 2 Enable Value. This bit reflects the value on the SYSDSSACCEN2 after selection. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 2.	RO	SYSDSSACCEN2
13	SYSDSSACCEN1_STATUS	Active-HIGH System Mapped Debug Access for CPU 1 Enable Value. This bit reflects the value on the SYSDSSACCEN1 after selection. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 1.	RO	SYSDSSACCEN1

Bits	Name	Description	Type	Default
12	SYSDSSACCEN0_STATUS	Active-HIGH System Mapped Debug Access for CPU 0 Enable Value. This bit reflects the value on the SYSDSSACCEN0 after selection. Reserved and RAZ/WI if HASCSS = 0.	RO	SYSDSSACCEN0
11	DAPDSSACCEN_SEL_STATUS	Active-HIGH DAP to Debug Subsystem Access Enable Selector Value. This bit returns the DAPDSSACCEN_SEL value. Forced to Zero if DAPDSSACCENSELDIS = 1.	RO	0x0
10	DAPDSSACCEN_STATUS	Active-HIGH DAP to Debug Subsystem Access Enable Value. This bit reflects the value on the DAPDSSACCEN pin.	RO	DAPDSSACCEN
9	DAPACCEN_SEL_STATUS	Active-HIGH DAP Access Enable Selector Value. This bit returns the DAPACCEN_SEL value. Forced to Zero if DAPACCENSELDIS = 1.	RO	0x0
8	DAPACCEN_STATUS	Active-HIGH DAP Access Enable Value. This bit reflects the value on the DAPACCEN pin.	RO	DAPACCEN
7	SPNIDEN_SEL_STATUS	Active-HIGH Secure Privileged Non-Invasive Debug Enable Selector Value. This bit returns the SPNIDEN_SEL value. Forced to Zero if SPNIDENSELDIS = 1.	RO	0x0
6	SPNIDEN_STATUS	Active-HIGH Secure Privileged Non-Invasive Debug Enable Value. This bit reflects the value on the SPNIDEN pin.	RO	SPNIDEN
5	SPIDEN_SEL_STATUS	Active-HIGH Secure Privileged Invasive Debug Enable Selector Value. This bit returns the SPIDEN_SEL value. Forced to Zero if SPIDENSELDIS = 1.	RO	0x0
4	SPIDEN_STATUS	Active-HIGH Secure Privileged Invasive Debug Enable Value. This bit reflects the value on the SPIDEN pin.	RO	SPIDEN
3	NIDEN_SEL_STATUS	Active-HIGH Non-Invasive Debug Enable Selector Value. This bit returns the NIDEN_SEL value. Forced to Zero if NIDENSELDIS = 1.	RO	0x0
2	NIDEN_STATUS	Active-HIGH Non-Invasive Debug Enable Value. This bit reflects the value on the NIDEN pin.	RO	NIDEN
1	DBGEN_SEL_STATUS	Active-HIGH Debug Enable Selector Value. This bit returns the DBGEN_SEL value. Forced to Zero if DBGENSELDIS = 1.	RO	0x0
0	DBGEN_STATUS	Active-HIGH Debug Enable Value. This bit reflects the value on the DBGEN pin.	RO	DBGEN

6.6.2.1.2 SECDBGSET

Secure Debug Configuration Set Register.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively. This choice is **IMPLEMENTATION DEFINED**.

Reset

This register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-59: SECDBGSET bit descriptions

Bits	Name	Description	Type	Default
31:18	-	Reserved.	RAZ/WI	0x0000
17	SYSDSSACCEN_SEL_SET	Set Active-HIGH System Mapped Debug Access Enable Selector Value. Write HIGH to set SYSDSSACCEN_SEL. Reserved and RAZ/WI if HASCSS = 0 or SYSDSSACCENSELDIS = 1.	RAZW1S	0x0
16	SYSDSSACCENX_I_SET	Set internal version of Active-HIGH System Mapped Debug Access for Implementation Defined Manager(s) Enable. Write HIGH to set SYSDSSACCENX_I. When read returns SYSDSSACCENX_I. Reserved and RAZ/WI if HASCSS = 0 or SYSDSSACCENSELDIS = 1.	RW1S	0x0
15	SYSDSSACCEN3_I_SET	Set internal version of Active-HIGH System Mapped Debug Access for CPU 3 Enable Set Register. Write HIGH to set SYSDSSACCEN3_I. When read returns SYSDSSACCEN3_I. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 3 or SYSDSSACCENSELDIS = 1. When read, this returns the internal SYSDSSACCEN3 register value before selection.	RW1S	0x0
14	SYSDSSACCEN2_I_SET	Set internal version of Active-HIGH System Mapped Debug Access for CPU 2 Enable. Write HIGH to set SYSDSSACCEN2_I. When read returns SYSDSSACCEN2_I. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 2 or SYSDSSACCENSELDIS = 1.	RW1S	0x0
13	SYSDSSACCEN1_I_SET	Set internal version of Active-HIGH System Mapped Debug Access for CPU 1 Enable. Write HIGH to set SYSDSSACCEN1_I. When read returns SYSDSSACCEN1_I. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 1 or SYSDSSACCENSELDIS = 1.	RW1S	0x0
12	SYSDSSACCEN0_I_SET	Set internal version of Active-HIGH System Mapped Debug Access for CPU 0 Enable. Write HIGH to set SYSDSSACCEN0_I. When read returns SYSDSSACCEN0_I. Reserved and RAZ/WI if HASCSS = 0 or SYSDSSACCENSELDIS = 1.	RW1S	0x0
11	DAPDSSACCEN_SEL_SET	Set Active-HIGH DAP to Debug Subsystem Access Enable Selector. Write HIGH to set DAPDSSACCEN_SEL. RAZ/WI if DAPDSSACCENSELDIS = 1.	RAZW1S	0x0

Bits	Name	Description	Type	Default
10	DAPDSSACCEN_I_SET	Set internal version of Active-HIGH DAP to Debug Subsystem Access Enable. Write HIGH to set DAPDSSACCEN_I. RAZ/WI if DAPDSSACCENSELDIS = 1.	RW1S	0x0
9	DAPACCEN_SEL_SET	Set Active-HIGH DAP Access Enable Selector. Write HIGH to set DAPACCEN_SEL. RAZ/WI if DAPACCENSELDIS = 1.	RAZW1S	0x0
8	DAPACCEN_I_SET	Set internal version of Active-HIGH DAP Access Enable. Write HIGH to set DAPACCEN_I. When read returns DAPACCEN_I. RAZ/WI if DAPACCENSELDIS = 1.	RW1S	0x0
7	SPNIDEN_SEL_SET	Set Active-HIGH Secure Privileged Non-Invasive Debug Enable Selector. Write HIGH to set SPNIDEN_SEL. RAZ/WI if SPNIDENSELDIS = 1.	RAZW1S	0x0
6	SPNIDEN_I_SET	Set internal version of Active-HIGH Secure Privileged Non-Invasive Debug Enable. Write HIGH to set SPNIDEN_I. When read returns SPNIDEN_I. RAZ/WI if SPNIDENSELDIS = 1.	RW1S	0x0
5	SPIDEN_SEL_SET	Set Active-HIGH Secure Privileged Invasive Debug Enable Selector. Write HIGH to set SPIDEN_SEL. RAZ/WI if SPIDENSELDIS = 1.	RAZW1S	0x0
4	SPIDEN_I_SET	Set internal version of Active-HIGH Secure Privileged Invasive Debug Enable. Write HIGH to set SPIDEN_I. When read returns SPIDEN_I. RAZ/WI if SPIDENSELDIS = 1.	RW1S	0x0
3	NIDEN_SEL_SET	Set Active-HIGH Non-Invasive Debug Enable Selector. Write HIGH to set NIDEN_SEL. RAZ/WI if NIDENSELDIS = 1.	RAZW1S	0x0
2	NIDEN_I_SET	Set internal version of Active-HIGH Non-Invasive Debug Enable. Write HIGH to set NIDEN_I. When read returns NIDEN_I. RAZ/WI if NIDENSELDIS = 1.	RW1S	0x0
1	DBGEN_SEL_SET	Set Active-HIGH Debug Enable Selector. Write HIGH to set DBGEN_SEL. RAZ/WI if DBGENSELDIS = 1.	RAZW1S	0x0
0	DBGEN_I_SET	Set internal version of Active-HIGH Debug Enable. Write HIGH to set DBGEN_I. When read returns DBGEN_I. RAZ/WI if DBGENSELDIS = 1.	RW1S	0x0

6.6.2.1.3 SECDBGCLR

The Secure Debug Configuration Clear Register.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively. This choice is **IMPLEMENTATION DEFINED**.

Reset

This register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-60: SECDDBGCLR bit descriptions

Bits	Name	Description	Type	Default
31:18	-	Reserved.	RAZ/WI	0x0000
17	SYSDSSACCEN_SEL_CLR	Clears Active-HIGH System Mapped Debug Access Enable Selector Value. Write HIGH to clear SYSDSSACCEN_SEL. RAZ/WI if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0.	RAZ/W1C	0x0
16	SYSDSSACCENX_I_CLR	Clears internal version of Active High System Mapped Debug Access for Implementation Defined Manager(s) Enable. Write HIGH to clear SYSDSSACCENX_I. RAZ/WI if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0.	RAZ/W1C	0x0
15	SYSDSSACCEN3_I_CLR	Clears internal version of Active High System Mapped Debug Access for CPU 3 Enable. Write HIGH to clear SYSDSSACCEN3_I. RAZ/WI if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 3.	RAZ/W1C	0x0
14	SYSDSSACCEN2_I_CLR	Clears internal version of Active High System Mapped Debug Access for CPU 2 Enable. Write HIGH to clear SYSDSSACCEN2_I. RAZ/WI if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 2.	RAZ/W1C	0x0
13	SYSDSSACCEN1_I_CLR	Clears internal version of Active High System Mapped Debug Access for CPU 1 Enable. Write HIGH to clear SYSDSSACCEN1_I. RAZ/WI if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0 or NUMCPU < 1.	RAZ/W1C	0x0
12	SYSDSSACCEN0_I_CLR	Clears internal version of Active High System Mapped Debug Access for CPU 0 Enable. Write HIGH to clear SYSDSSACCEN0_I. RAZ/WI if SYSDSSACCENSELDIS = 1. Reserved and RAZ/WI if HASCSS = 0.	RAZ/W1C	0x0
11	DAPDSSACCEN_SEL_CLR	Clears Active-HIGH DAP to Debug Subsystem Access Enable Selector. Write HIGH to clear DAPDSSACCEN_SEL. RAZ/WI if DAPDSSACCENSELDIS = 1.	RAZ/W1C	0x0
10	DAPDSSACCEN_I_CLR	Clears internal version of Active High DAP to Debug Subsystem Access Enable. Write HIGH to clear DAPDSSACCEN_I. RAZ/WI if DAPDSSACCENSELDIS = 1.	RAZ/W1C	0x0
9	DAPACCEN_SEL_CLR	Clears Active-HIGH DAP Access Enable Selector. Write HIGH to clear DAPACCEN_SEL. RAZ/WI if DAPACCENSELDIS = 1.	RAZ/W1C	0x0
8	DAPACCEN_I_CLR	Clears internal version of Active High DAP Access Enable. Write HIGH to clear DAPACCEN_I. RAZ/WI if DAPACCENSELDIS = 1.	RAZ/W1C	0x0
7	SPNIDEN_SEL_CLR	Clears Active-HIGH Secure Privileged Non-Invasive Debug Enable Selector. Write HIGH to clear SPNIDEN_SEL. RAZ/WI if SPNIDENSELDIS = 1.	RAZ/W1C	0x0
6	SPNIDEN_I_CLR	Clears internal version of Active High Secure Privileged Non-Invasive Debug Enable. Write HIGH to clear SPNIDEN_I. RAZ/WI if SPNIDENSELDIS = 1.	RAZ/W1C	0x0
5	SPIDEN_SEL_CLR	Clears Active-HIGH Secure Privileged Invasive Debug Enable Selector. Write HIGH to clear SPIDEN_SEL. RAZ/WI if SPIDENSELDIS = 1.	RAZ/W1C	0x0
4	SPIDEN_I_CLR	Clears internal version of Active High Secure Privileged Invasive Debug Enable. Write HIGH to clear SPIDEN_I. RAZ/WI if SPIDENSELDIS = 1.	RAZ/W1C	0x0
3	NIDEN_SEL_CLR	Clears Active-HIGH Non-Invasive Debug Enable Selector. Write HIGH to clear NIDEN_SEL. RAZ/WI if NIDENSELDIS = 1.	RAZ/W1C	0x0
2	NIDEN_I_CLR	Clears internal version of Active High Non-Invasive Debug Enable. Write HIGH to clear NIDEN_I. RAZ/WI if NIDENSELDIS = 1.	RAZ/W1C	0x0

Bits	Name	Description	Type	Default
1	DBGEN_SEL_CLR	Clears Active-HIGH Debug Enable Selector. Write HIGH to clear DBGEN_SEL. RAZ/WI if DBGENSELDIS = 1.	RAZ/ W1C	0x0
0	DBGEN_I_CLR	Clears internal version of Active High Debug Enable. Write HIGH to clear DBGEN_I. RAZ/WI if DBGENSELDIS = 1.	RAZ/ W1C	0x0

6.6.2.2 SCSECCTRL

The System Control Security Controls provides register bits to set the Secure Configuration lock of this register block.

These registers are reset by **nCOLDRESETAON**.

These registers reside in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-61: SCSECCTRL bit descriptions

Bits	Name	Description	Type	Default
31:3	-	Reserved.	RAZ/ WI	0x0000_0000
2	SCSECCFGLOCK	Active-HIGH control to disable writes to Security related control registers SECDBGSET and SECDBGCLR. Once set to HIGH, it can no longer be cleared to zero except through Cold reset.	RW1S	0x0
1:0	-	Reserved.	RAZ/ WI	0x0

6.6.2.3 CLK_CFG0

The CLK_CFG0 register provides control register fields to drive expansion clock generation logic that drives clock for this subsystem.

Each clock control handshake comprises of a configuration request register, that is CLKCFG, and a status register, that is CLKCFGSTATUS that acts as an acknowledgment. After writing to each CLKCFG field, the software has to poll the associated CLKCFGSTATUS field until the CLKCFGSTATUS is the same as the CLKCFG before doing any other operations. In addition, if it is the first write to the register after boot, we recommend that software polls to check that the targeted CLKCFGSTATUS field matches its associated CLKCFG value before the write occurs. The actual number of bits being implemented in the related CLKCFG and CLKCFGSTATUS signals can be less than that defined below, and is IMPLEMENTATION DEFINED. For any bit that is not implemented, the associated register bit is **RAZ/WI**.

See also [CLK_CFG1](#) and [CLK_CFG2](#).

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-62: CLK_CFG0 bit descriptions

Bits	Name	Description	Type	Default
31:28	CPU3CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for CPU3CLK . RAZ/WI if NUMCPU < 3	RO	CPU3CLKCFGSTATUS
27:24	CPU2CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for CPU2CLK . RAZ/WI if NUMCPU < 2	RO	CPU2CLKCFGSTATUS
23:20	CPU1CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for CPU1CLK . RAZ/WI if NUMCPU < 1	RO	CPU1CLKCFGSTATUS
19:16	CPU0CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for CPU0CLK .	RO	CPU0CLKCFGSTATUS

Bits	Name	Description	Type	Default
15:12	CPU3CLKCFG	Clock Configuration value that drives CPU3CLKCFG signals. RAZ/WI if NUMCPU < 3	RW	CPU3CLKCFG RST
11:8	CPU2CLKCFG	Clock Configuration value that drives CPU2CLKCFG signals. RAZ/WI if NUMCPU < 2	RW	CPU2CLKCFG RST
7:4	CPU1CLKCFG	Clock Configuration value that drives CPU1CLKCFG signals. RAZ/WI if NUMCPU < 1	RW	CPU1CLKCFG RST
3:0	CPU0CLKCFG	Clock Configuration value that drives CPU0CLKCFG signals.	RW	CPU0CLKCFG RST

6.6.2.4 CLK_CFG1

The CLK_CFG1 register provides control register fields to drive expansion clock generation logic that drives clock for this subsystem.

Each clock control handshake comprises of a configuration request register, that is CLKCFG, and a status register, that is CLKCFGSTATUS that acts as an acknowledgment. After writing to each CLKCFG field, the software has to poll the associated CLKCFGSTATUS field until the CLKCFGSTATUS is the same as the CLKCFG before doing any other operations. In addition, if it is the first write to the register after boot, we recommend that software polls to check that the targeted CLKCFGSTATUS field matches its associated CLKCFG value before the write occurs. The actual number of bits being implemented in the related CLKCFG and CLKCFGSTATUS signals can be less than that defined below, and is IMPLEMENTATION DEFINED. For any bit that is not implemented, the associated register bit is **RAZ/WI**.

See also [CLK_CFG0](#) and [CLK_CFG2](#).

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nCOLDRESETAON** only.

Bit descriptions

Table 6-63: CLK_CFG1 bit descriptions

Bits	Name	Description	Type	Default
31:24	-	Reserved.	RAZ/WI	0x00

Bits	Name	Description	Type	Default
23:20	AONCLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for AONCLK .	RO	AONCLKCFGSTATUS
19:16	SYSCLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for SYSCLK .	RO	SYSCLKCFGSTATUS
15:8	-	Reserved.	RAZ/WI	0x00
7:4	AONCLKCFG	Clock Configuration value that drives AONCLKCFG signals.	RW	AONCLKCFG
3:0	SYSCLKCFG	Clock Configuration value that drives SYSCLKCFG signals.	RW	SYSCLKCFG

6.6.2.5 CLK_CFG2

The CLK_CFG2 register provides control register fields to drive expansion clock generation logic that drives clock for this subsystem.

Each clock control handshake comprises of a configuration request register, that is CLKCFG, and a status register, that is CLKCFGSTATUS that acts as an acknowledgment. After writing to each CLKCFG field, the software has to poll the associated CLKCFGSTATUS field until the CLKCFGSTATUS is the same as the CLKCFG before doing any other operations. In addition, if it is the first write to the register after boot, we recommend that software polls to check that the targeted CLKCFGSTATUS field matches its associated CLKCFG value before the write occurs. The actual number of bits being implemented in the related CLKCFG and CLKCFGSTATUS signals can be less than that defined below, and is IMPLEMENTATION DEFINED. For any bit that is not implemented, the associated register bit is **RAZ/WI**.

See also [CLK_CFG0](#) and [CLK_CFG1](#).

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nCOLDRESEAON** only.

Bit descriptions

Table 6-64: CLK_CFG2 bit descriptions

Bits	Name	Description	Type	Default
31:28	NPU3CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for NPU3CLK . RAZ/WI if NUMNPU < 4	RO	NPU3CLKCFGSTATUS
27:24	NPU2CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for NPU2CLK . RAZ/WI if NUMNPU < 3	RO	NPU2CLKCFGSTATUS
23:20	NPU1CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for NPU1CLK . RAZ/WI if NUMNPU < 2	RO	NPU1CLKCFGSTATUS
19:16	NPU0CLKCFGSTATUS	Clock Configuration Status value that reports the status of clock control for NPU0CLK . RAZ/WI if NUMNPU < 1	RO	NPU0CLKCFGSTATUS
15:12	NPU3CLKCFG	Clock Configuration value that drives NPU3CLKCFG signals. RAZ/WI if NUMNPU < 4	RW	NPU3CLKCFGRST
11:8	NPU2CLKCFG	Clock Configuration value that drives NPU2CLKCFG signals. RAZ/WI if NUMNPU < 3	RW	NPU2CLKCFGRST
7:4	NPU1CLKCFG	Clock Configuration value that drives NPU1CLKCFG signals. RAZ/WI if NUMNPU < 2	RW	NPU1CLKCFGRST
3:0	NPU0CLKCFG	Clock Configuration value that drives NPU0CLKCFG signals. RAZ/WI if NUMNPU < 1	RW	NPU0CLKCFGRST

6.6.2.6 CLOCK_FORCE

The Clock Force register allows software to override dynamic clock gating that may be implemented in the system and keep each clock running.

Bits 0 to 11 are clock forces that does not apply to clock gates within the design that are responsible for the gating of clocks when gating power to a power gated region. Instead, it is applied to hierarchical dynamic clock gating within the system, with one bit for each power domain. Forcing a clock ON can reduce the latency that is incurred as a result of dynamic clock control but generally has a reverse side effect of increasing the dynamic power consumption of the system. Note that all clock force default values, are set to HIGH at reset. This allows the system to boot in case any hierarchical dynamic clock control implementation is non-functional. You must clear the associated clock force register bit to enable hierarchical dynamic clock control for each power domain.

Bits 16 to 26 are clock forces that are used to force the external clock generator to continue to generate its clock. This can be useful, if desired, to avoid clock generators like PLLs from turning off, which can result in a long turn on time.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-65: CLOCK_FORCE bit descriptions

Bits	Name	Description	Type	Default
31:27	-	Reserved.	RAZ/WI	0x00
26	NPU3CLK_FORCE	Set HIGH to request the input NPU3CLK source to stay ON. The field is reserved and RAZ/WI if NUMNPU < 4.	RW	0x0
25	NPU2CLK_FORCE	Set HIGH to request the input NPU2CLK source to stay ON. The field is reserved and RAZ/WI if NUMNPU < 3.	RW	0x0
24	NPU1CLK_FORCE	Set HIGH to request the input NPU1CLK source to stay ON. The field is reserved and RAZ/WI if NUMNPU < 2.	RW	0x0
23	NPU0CLK_FORCE	Set HIGH to request the input NPU0CLK source to stay ON. The field is reserved and RAZ/WI if NUMNPU < 1.	RW	0x0
22	CPU3CLK_FORCE	Set HIGH to request the input CPU3CLK source to stay ON. The field is reserved and RAZ/WI if NUMCPU < 3.	RW	0x0
21	CPU2CLK_FORCE	Set HIGH to request the input CPU2CLK source to stay ON. The field is reserved and RAZ/WI if NUMCPU < 2.	RW	0x0
20	CPU1CLK_FORCE	Set HIGH to request the input CPU1CLK source to stay ON. The field is reserved and RAZ/WI if NUMCPU < 1.	RW	0x0
19	CPU0CLK_FORCE	Set HIGH to request the input CPU0CLK source to stay ON.	RW	0x0
18	DEBUGCLK_FORCE	Set HIGH to request the input DEBUGCLK source to stay ON. This field is reserved and RAZ/WI if DEBUGCLK is not implemented.	RW	0x0
17	SYSCLK_FORCE	Set HIGH to request the input SYSCLK source to stay ON.	RW	0x0
16	AONCLK_FORCE	Set HIGH to request the input AONCLK source to stay ON.	RW	0x0
15:12	-	Reserved.	RAZ/WI	0x0
11	NPU3_CLKFORCE	Set HIGH to force PD_NPU3 Local clocks to run. The field is reserved and RAZ/WI if NUMNPU < 4.	RW	0x1
10	NPU2_CLKFORCE	Set HIGH to force PD_NPU2 Local clocks to run. The field is reserved and RAZ/WI if NUMNPU < 3.	RW	0x1
9	NPU1_CLKFORCE	Set HIGH to force PD_NPU1 Local clocks to run. The field is reserved and RAZ/WI if NUMNPU < 2.	RW	0x1
8	NPU0_CLKFORCE	Set HIGH to force PD_NPU0 Local clocks to run. The field is reserved and RAZ/WI if NUMNPU < 1.	RW	0x1

Bits	Name	Description	Type	Default
7	CPU3_CLKFORCE	Set HIGH to force PD_CPU3 Local clocks to run. The field is reserved and RAZ/WI if NUMCPU < 3.	RW	0x1
6	CPU2_CLKFORCE	Set HIGH to force PD_CPU2 Local clocks to run. The field is reserved and RAZ/WI if NUMCPU < 2.	RW	0x1
5	CPU1_CLKFORCE	Set HIGH to force PD_CPU1 Local clocks to run. The field is reserved and RAZ/WI if NUMCPU < 1.	RW	0x1
4	CPU0_CLKFORCE	Set HIGH to force PD_CPU0 Local clocks to run.	RW	0x1
3	CRYPTO_CLKFORCE	Set HIGH to force all clocks in PD_CRYPT to run. The field is reserved and RAZ/WI if HASCRYPTO = 0.	RW	0x1
2	DEBUG_CLKFORCE	Set HIGH to force all clocks in PD_DEBUG to run.	RW	0x1
1	SYS_CLKFORCE	Set HIGH to force all clocks in PD_SYS to run.	RW	0x1
0	MGMT_CLKFORCE	Set HIGH to force all clocks in PD_MGMT domain to run.	RW	0x1

6.6.2.7 RESET_SYNDROME

The RESET_SYNDROME register stores the reason for the last Reset event. Writing HIGH to a bit results in that bit write value to be ignored and the bit maintaining its previous value. RESET_SYNDROME is cleared by software writing zero to each bit to clear. If after starting from a reset event, RESET_SYNDROME is not cleared, on another reset event, the register may no longer accurately reflect the last reset event. CPU<n>LOCKUP does not actually generate reset, but when HIGH, it indicates that a CPU has locked-up and could be a precursor to another reset event, for example, watchdog timer reset request.

The RESET_SYNDROME register always stores Reset events regardless of the COLDRESET_MODE configuration and even if the Cold reset generation is performed externally into the system through the **HOSTRESETREQ** reset request signal. Also note that CPU<n>LOCKUP events are always stored, and only reset requests that are not masked by their respective mask bits are stored in the register.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

The RESET_SYNDROME register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively and if its functionality is maintained while PD_MGMT is in low power state.

Reset

This register is reset by **nPORESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-66: RESET_SYNDROME bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/WI	0x0_0000
15	CPU3LOCKUP	CPU 3 Lockup Status. This bit is reserved and RAZ/WI if NUMCPU < 3.	RWOC	0x0
14	CPU2LOCKUP	CPU 2 Lockup Status. This bit is reserved and RAZ/WI if NUMCPU < 2.	RWOC	0x0
13	CPU1LOCKUP	CPU 1 Lockup Status. This bit is reserved and RAZ/WI if NUMCPU < 1.	RWOC	0x0
12	CPU0LOCKUP	CPU 0 Lockup Status.	RWOC	0x0
11	CPU3RSTREQ	CPU 3 Warm Reset Request. This bit is reserved and RAZ/WI if NUMCPU < 3.	RWOC	0x0
10	CPU2RSTREQ	CPU 2 Warm Reset Request. This bit is reserved and RAZ/WI if NUMCPU < 2.	RWOC	0x0
9	CPU1RSTREQ	CPU 1 Warm Reset Request. This bit is reserved and RAZ/WI if NUMCPU < 1.	RWOC	0x0
8	CPU0RSTREQ	CPU 0 Warm Reset Request.	RWOC	0x0
7	HOSTRESETREQ	Host Level Cold Reset Request Input.	RWOC	0x0
6	CRYPTORSTREQ	CryptoCell Warm Reset Request. This bit is reserved and RAZ/WI if HASCRYPTO = 0.	RWOC	0x0
5	SWRESETREQ	Software Cold Reset Request.	RWOC	0x0
4	RESETREQ	Subsystem Hardware Cold Reset Request Input.	RWOC	0x0
3	SLOWCLKWDRSTREQ	SLOWCLK Watchdog Cold Reset Request.	RWOC	0x0
2	SWDRSTREQ	Secure Watchdog Cold Reset Request.	RWOC	0x0
1	NSWDRSTREQ	Non-secure Watchdog Cold Reset Request.	RWOC	0x0
0	PoR	Power-On-Reset	RWOC	0x01

6.6.2.8 RESET_MASK

The RESET_MASK register allows software to control which reset sources are going to be merged to generate the system wide warm reset, **nWARMRESETAON**, or the **nCOLDRESETAON** signal. Set each bit to HIGH to enable each source. Note that each of these mask bits, if cleared, not only prevents the reset source being used to generate the reset, it also prevents the associated RESET_SYNDROME register bit from recording the event.

Depending on the COLDRESET_MODE configuration, the final reset could be generated within the subsystem, or through a higher level reset generation entity through the **HOSTRESETREQ** signal.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-67: RESET_MASK bit descriptions

Bits	Name	Description	Type	Default
31:12	-	Reserved.	RAZ/WI	0x00_0000
11	CPU3RSTREQEN	CPU 3 Warm Reset Request Enable. This bit is Reserved and RAZ/WI if NUMCPU < 3.	RW	CPU3RSTREQENRST
10	CPU2RSTREQEN	CPU 2 Warm Reset Request Enable. This bit is Reserved and RAZ/WI if NUMCPU < 2.	RW	CPU2RSTREQENRST
9	CPU1RSTREQEN	CPU 1 Warm Reset Request Enable. This bit is Reserved and RAZ/WI if NUMCPU < 1.	RW	CPU1RSTREQENRST
8	CPU0RSTREQEN	CPU 0 Warm Reset Request Enable.	RW	CPU0RSTREQENRST
7:2	-	Reserved.	RAZ/WI	0x00
1	NSWDRSTREQEN	Non-secure Watchdog Reset Enable.	RW	0x0
0	-	Reserved.	RAZ/WI	0x0

6.6.2.9 SWRESET

The SWRESET register allows software to request for a system Cold reset. To request for a Cold reset, write '1' to the register. The register always returns zeros.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON or in the PD_MGMT power domain. Its content is not retained when PD_MGMT is turned off in HIBERNATION1 state when PILEVEL = 2.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-68: SWRESET bit descriptions

Bits	Name	Description	Type	Default
31:10	-	Reserved.	RAZ/WI	0x00_0000
9	SWRESETREQ	Software Reset Request. Write '1' to set to HIGH, to request a Cold reset.	RAZW1S	0x0
8:0	-	Reserved.	RAZ/WI	0x000

6.6.2.10 GRETRREG

The General Purpose Retention Register provides 16 bits of retention register for general storage, through HIBERANTION0 or HIBERNATION1 system power states.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

The GRETRREG resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-69: GRETRREG bit descriptions

Bits	Name	Description	Type	Default
31:16	-	Reserved.	RAZ/WI	0x00000
15:0	GRETRREG	General Purpose Retention Register.	RW	0x0000

6.6.2.11 INITSVTOR<n>

The INITSVTOR<n> register is used to define the CPU <n> Initial Secure Vector table offset (VTOR_S.TBLOFF[31:7]) out of reset.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

The INITSVTOR<n> register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

The INITSVTOR<n> register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-70: INITSVTOR<n> bit descriptions

Bits	Name	Description	Type	Default
31:7	INITSVTOR<n>	Default Secure Vector table offset at reset for CPU <n>.	RW	INITSVTOR<n> RST[31:7]
6:1	-	Reserved.	RAZ/WI	0x00
0	INITSVTOR<n> LOCK	Lock INITSVTOR<n>. When set to '1', it stops any further writes to INITSVTOR<n> and INITSVTOR<n>LOCK fields. Cleared only by warm reset.	RW1S	0x0

6.6.2.12 CPUWAIT

The CPUWAIT register provides controls to force each CPU to wait after reset rather than Boot Immediately. This allows another entity in the expansion system or the debugger to access the system prior to the CPU booting.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

The CPUWAIT register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

The CPUWAIT register is reset by **nCOLDRESETAON** only.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-71: CPUWAIT bit descriptions

Bits	Name	Description	Type	Default
31:4	-	Reserved.	RAZ/WI	0x0000_0000
3	CPU3WAIT	<p>CPU 3 waits at boot.</p> <ul style="list-style-type: none"> ‘0’: boot normally. ‘1’: wait at boot. <p>When CPU3WAITCLR input is 1'b1, this bit is also cleared. This bit is Reserved and RAZ/WI if NUMCPU < 3.</p>	RW	CPU3WAITRST
2	CPU2WAIT	<p>CPU 2 waits at boot.</p> <ul style="list-style-type: none"> ‘0’: boot normally. ‘1’: wait at boot. <p>When CPU2WAITCLR input is 1'b1, this bit is also cleared. This bit is Reserved and RAZ/WI if NUMCPU < 2.</p>	RW	CPU2WAITRST
1	CPU1WAIT	<p>CPU 1 waits at boot.</p> <ul style="list-style-type: none"> ‘0’: boot normally. ‘1’: wait at boot. <p>When CPU1WAITCLR input is 1'b1, this bit is also cleared. This bit is Reserved and RAZ/WI if NUMCPU < 1.</p>	RW	CPU1WAITRST
0	CPU0WAIT	<p>CPU 0 waits at boot.</p> <ul style="list-style-type: none"> ‘0’: boot normally. ‘1’: wait at boot. <p>When CPU0WAITCLR input is 1'b1, this bit is also cleared.</p>	RW	CPU0WAITRST

6.6.2.13 NMI_ENABLE

The NMI_ENABLE register provides controls to enable or disable the internally or externally generated Non-Maskable Interrupt sources from generating an NMI interrupt on each CPU core.

This allows a CPU to take control of all internal NMI interrupt sources or allow all CPUs to see the same NMI interrupts.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

The NMI_ENABLE register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON** and its reset value is defined by the configuration options.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-72: NMI_ENABLE bit descriptions

Bits	Name	Description	Type	Default
31:20	-	Reserved.	RAZ/WI	0x0000
19	CPU3_EXPNMI_ENABLE	<p>CPU 3 Externally Sourced NMI Enable. This determines if the input CPU3EXPNMI can raise NMI interrupt on CPU 3:</p> <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. <p>This bit is reserved and RAZ/WI if NUMCPU < 3.</p>	RW	CPU3EXPNMIENABLERST
18	CPU2_EXPNMI_ENABLE	<p>CPU 2 Externally Sourced NMI Enable. This determines if the input CPU2EXPNMI can raise NMI interrupt on CPU 2:</p> <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. <p>This bit is reserved and RAZ/WI if NUMCPU < 2.</p>	RW	CPU2EXPNMIENABLERST
17	CPU1_EXPNMI_ENABLE	<p>CPU 1 Externally Sourced NMI Enable. This determines if the input CPU1EXPNMI can raise NMI interrupt on CPU 1:</p> <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. <p>This bit is reserved and RAZ/WI if NUMCPU < 1.</p>	RW	CPU1EXPNMIENABLERST

Bits	Name	Description	Type	Default
16	CPU0_EXPNMI_ENABLE	CPU 0 Externally Sourced NMI Enable. This determines if the input, CPU0EXPNMI can raise NMI interrupt on CPU 0: <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. 	RW	CPU0EXPNMIENABLERST
15:4	-	Reserved.	RAZ/WI	0x0000
3	CPU3_INTNMI_ENABLE	CPU 3 Internally Sourced NMI Enable. This determines if the subsystem internally generated NMI interrupt sources can raise NMI interrupt on CPU 3: <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. This bit is reserved and RAZ/WI if NUMCPU < 3.	RW	CPU3INTNMIENABLERST
2	CPU2_INTNMI_ENABLE	CPU 2 Internally Sourced NMI Enable. This determines if the subsystem internally generated NMI interrupt sources can raise NMI interrupt on CPU 2: <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. This bit is reserved and RAZ/WI if NUMCPU < 2.	RW	CPU2INTNMIENABLERST
1	CPU1_INTNMI_ENABLE	CPU 1 Internally Sourced NMI Enable. This determines if the subsystem internally generated NMI interrupt sources can raise NMI interrupt on CPU 1: <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. This bit is reserved and RAZ/WI if NUMCPU < 1.	RW	CPU1INTNMIENABLERST
0	CPU0_INTNMI_ENABLE	CPU 0 Internally Sourced NMI Enable. This determines if the subsystem internally generated NMI interrupt sources can raise NMI interrupt on CPU 0: <ul style="list-style-type: none"> HIGH, allowed. LOW, is masked and not allowed. 	RW	CPU0INTNMIENABLERST

6.6.2.14 PPUINTSTAT

The PPU Interrupt Status Register brings together all PPU interrupt statuses in to a single register.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2.

Reset

This register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-73: PPUINTSTAT bit descriptions

Bits	Name	Description	Type	Default
31:12	-	Reserved.	RAZ/WI	0x0000_0000
11	DEBUG_PPU_INTSTAT	Debug PPU Interrupt	RO	0x0
10	NPU3_PPU_INTSTAT	NPU 3 PPU interrupt RAZ/WI if NUMNPU <4	RO	0x0
9	NPU2_PPU_INTSTAT	NPU 2 PPU interrupt RAZ/WI if NUMNPU <3	RO	0x0
8	NPU1_PPU_INTSTAT	NPU 1 PPU interrupt RAZ/WI if NUMNPU <2	RO	0x0
7	NPU0_PPU_INTSTAT	NPU 0 PPU interrupt RAZ/WI if NUMNPU <1	RO	0x0
6	CRYPTO_PPU_INTSTAT	Crypto PPU interrupt. RAZ/WI if HASCRYPTO = 0	RO	0x0
5	CPU3_PPU_INTSTAT	CPU 3 PPU interrupt RAZ/WI if NUMCPU <3	RO	0x0
4	CPU2_PPU_INTSTAT	CPU 2 PPU interrupt RAZ/WI if NUMCPU <2	RO	0x0
3	CPU1_PPU_INTSTAT	CPU 1 PPU interrupt RAZ/WI if NUMCPU <1	RO	0x0
2	CPU0_PPU_INTSTAT	CPU 0 PPU interrupt	RO	0x0
1	SYS_PPU_INTSTAT	System PPU interrupt	RO	0x0
0	MGMT_PPU_INTSTAT	Management PPU Interrupt	RO	0x0

6.6.2.15 PWRCTRL

The Power Control register configures the power control features in CRSAS Ma1 System.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-74: PWRCTRL bit descriptions

Bits	Name	Description	Type	Default
31:2	-	Reserved.	RAZ/WI	0x0000_0000
1	PPU_ACCESS_UNLOCK	PPU_ACCESS_FILTER write unlock. When '1', both PPU_ACCESS_FILTER and this register bits can be written. When set to '0', the PPU_ACCESS_FILTER and this register bit is no longer writable, and PPU_ACCESS_UNLOCK stays '0'.	RW0C	0x01
0	PPU_ACCESS_FILTER	Filter Access to PPU Registers. When set to '1', only key PPU interrupt handling registers are open to write access, and all other PPU registers are read only. When set to '0', it releases all PPU registers to full access. For more information in PPU registers accessibility, see Power Policy Units .	RW	0x01

6.6.2.16 PDCM_PD_SYS_SENSE

The Power Dependency Control Matrix System Power domain (PD_SYS) Sensitivity register is used to define what keeps the PD_SYS domain awake and the minimum power state to use when the domain is in its low power state.

Configurations

This register implementation depends on the configuration of individual fields.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON**.

Bit descriptions

Table 6-75: PDCM_PD_SYS_SENSE bit descriptions

Bits	Name	Description	Type	Default
31:30	MIN_PWR_STATE	Defines the Minimum Power State, when PD_SYS is trying to enter a lower power state: <ul style="list-style-type: none"> '00': Minimum power state is OFF, '01': Minimum power state is Retention, '10': Minimum power state is ON, Others: Reserved. 	RW	0x0
29:24	-	Reserved.	RAZ/WI	0x000
23	S_PDCMRETQREQ3	Enable sensitivity to PDCMQRETREQn[3] signal. If set to '1', PD_SYS stays in ON or RET if PDCMRETQREQn[3] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RW	0x0
22	S_PDCMRETQREQ2	Enable sensitivity to PDCMQRETREQn[2] signal. If set to '1', PD_SYS stays in ON or RET if PDCMRETQREQn[2] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RW	0x0
21	S_PDCMRETQREQ1	Enable sensitivity to PDCMQRETREQn[1] signal. If set to '1', PD_SYS stays in ON or RET if PDCMRETQREQn[1] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RW	0x0
20	S_PDCMRETQREQ0	Enable sensitivity to PDCMQRETREQn[0] signal. If set to '1', PD_SYS stays in ON or RET if PDCMRETQREQn[0] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RW	0x0
19	S_PDCMONQREQ3	Enable sensitivity to PDCMQONREQn[3] signal. If set to '1', PD_SYS stays ON if PDCMONQREQn[3] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RW	0x0
18	S_PDCMONQREQ2	Enable sensitivity to PDCMQONREQn[2] signal. If set to '1', PD_SYS stays ON if PDCMONQREQn[2] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RW	0x0
17	S_PDCMONQREQ1	Enable sensitivity to PDCMQONREQn[1] signal. If set to '1', PD_SYS stays ON if PDCMONQREQn[1] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RW	0x0
16	S_PDCMONQREQ0	Enable sensitivity to PDCMQONREQn[0] signal. If set to '1', PD_SYS stays ON if PDCMONQREQn[0] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RW	0x0
15	S_PD_MGMT_ON	Tied to LOW. Ignores PD_MGMT power state. This bit is reserved and RAZ/WI if PILEVEL < 2.	RO	0x0
14	-	Reserved.	RAZ/WI	0x0
13	S_PD_DEBUG_ON	Tied to LOW. Ignores PD_DEBUG power state.	RO	0x0
12	S_PD_CRYPTON_ON	Tied to HIGH. PD_SYS stays ON if PD_CRYPTON power domain is ON. This bit is reserved and RAZ/WI if HASCRYPTON = 0 or PILEVEL < 2.	RO	0x1
11:9	-	Reserved.	RAZ/WI	0x0
8	S_PD_NPU3_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_NPU3 is ON. This bit is reserved and RAZ/WI if NUMNPU < 4.	RO	0x1
7	S_PD_NPU2_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_NPU2 is ON. This bit is reserved and RAZ/WI if NUMNPU < 3.	RO	0x1

Bits	Name	Description	Type	Default
6	S_PD_NPU1_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_NPU1 is ON. This bit is reserved and RAZ/WI if NUMNPU < 2.	RO	0x1
5	S_PD_NPU0_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_NPU0 is ON. This bit is reserved and RAZ/WI if NUMNPU < 1.	RO	0x1
4	S_PD_CPU3_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_CPU3 is ON. This bit is reserved and RAZ/WI if NUMCPU < 3 or PILEVEL < 1.	RO	0x1
3	S_PD_CPU2_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_CPU2 is ON. This bit is reserved and RAZ/WI if NUMCPU < 2 or PILEVEL < 1.	RO	0x1
2	S_PD_CPU1_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_CPU1 is ON. This bit is reserved and RAZ/WI if NUMCPU < 1 or PILEVEL < 1.	RO	0x1
1	S_PD_CPU0_ON	Tied to HIGH. PD_SYS always tries to stay ON if PD_CPU0 is ON. This bit is reserved and RAZ/WI if PILEVEL < 1.	RO	0x1
0	S_PD_SYS_ON	Enable PD_SYS ON Sensitivity. Set this to high to keep PD_SYS awake once powered ON.	RW	0x0

6.6.2.17 PDCM_PD_CPU<n>_SENSE

The Power Dependency Control Matrix CPU <n> Power domain (PD_CPU<n>) Sensitivity register is used to define what keeps the PD_CPU<n> domain awake.

Currently, the PD_CPU<n> are not sensitive to any incoming dependencies.

Configurations

This register implementation depends on the configuration of individual fields. When PILEVEL < 1, these registers do not exist and the register areas they occupy are Reserved, and **RAZ/WI**.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-76: PDCM_PD_CPU<n>_SENSE bit descriptions

Bits	Name	Description	Type	Default
31:24	-	Reserved.	RAZ/WI	0x000
23	S_PDCMRETQREQ3	Tied to LOW. Ignores PDCMRETQREQn[3] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RO	0x0
22	S_PDCMRETQREQ2	Tied to LOW. Ignores PDCMRETQREQn[2] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RO	0x0
21	S_PDCMRETQREQ1	Tied to LOW. Ignores PDCMRETQREQn[1] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RO	0x0
20	S_PDCMRETQREQ0	Tied to LOW. Ignores PDCMRETQREQn[0] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RO	0x0
19	S_PDCMONQREQ3	Tied to LOW. Ignores PDCMONQREQn[3] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RO	0x0
18	S_PDCMONQREQ2	Tied to LOW. Ignores PDCMONQREQn[2] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RO	0x0
17	S_PDCMONQREQ1	Tied to LOW. Ignores PDCMONQREQn[1] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RO	0x0
16	S_PDCMONQREQ0	Tied to LOW. Ignores PDCMONQREQn[0] signal. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RO	0x0
15	S_PD_MGMT_ON	Tied to LOW. Ignores PD_MGMT power state. This bit is reserved and RAZ/WI if PILEVEL < 2.	RO	0x0
14	-	Reserved.	RAZ/WI	0x0
13	S_PD_DEBUG_ON	Tied to LOW. Ignores PD_DEBUG power state.	RO	0x0
12	S_PD_CRYPTON_ON	Tied to LOW. Ignores PD_CRYPTON power state. This bit is reserved and RAZ/WI if HASCRYPTO = 0.	RO	0x0
11:5	-	Reserved.	RAZ/WI	0x000
4	S_PD_CPU3_ON	Tied to LOW. Ignores PD_CPU3 power state. This bit is reserved and RAZ/WI if NUMCPU < 3.	RO	0x0
3	S_PD_CPU2_ON	Tied to LOW. Ignores PD_CPU2 power state. This bit is reserved and RAZ/WI if NUMCPU < 2.	RO	0x0
2	S_PD_CPU1_ON	Tied to LOW. Ignores PD_CPU1 power state. This bit is reserved and RAZ/WI if NUMCPU < 1.	RO	0x0
1	S_PD_CPU0_ON	Tied to LOW. Ignores PD_CPU0 power state.	RO	0x0
0	S_PD_SYS_ON	Tied to LOW. Ignores PD_SYS power state.	RO	0x0

6.6.2.18 PDCM_PD_VMR<x>_SENSE

The Power Dependency Control Matrix Volatile Memory Region <x> Power domain (PD_VMR<x>) Sensitivity register is used to define what keeps awake the PD_VMR<x> domain and the minimum power state to use when the domain is in its low power state, where x is 0 to NUMVMBANK-1.

Configurations

This register implementation depends on the configuration of individual fields. When PILEVEL < 1, these registers do not exist and the register areas they occupy are Reserved, and **RAZ/WI**.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain when PILEVEL = 2 if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-77: PDCM_PD_VMR<x>_SENSE bit descriptions

Bits	Name	Description	Type	Default
31:30	MIN_PWR_STATE	Defines the Minimum Power State, when PD_VMR<x> is trying to enter a lower power state: <ul style="list-style-type: none"> '00': Minimum power state is OFF, '01': Minimum power state is Retention, '10': Minimum power state is ON, Others: Reserved. 	RW	0x01
29:24	-	Reserved.	RAZ/WI	0x0
23	S_PDCMRETQREQ3	Enable sensitivity to PDCMRETQREQn[3] signal. If set to '1', PD_VMR<x> stay in ON or RET if PDCMRETQREQn[3] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RW	0x0
22	S_PDCMRETQREQ2	Enable sensitivity to PDCMRETQREQn[2] signal. If set to '1', PD_VMR<x> stays in ON or RET if PDCMRETQREQn[2] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RW	0x0
21	S_PDCMRETQREQ1	Enable sensitivity to PDCMRETQREQn[1] signal. If set to '1', PD_VMR<x> stays in ON or RET if PDCMRETQREQn[1] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RW	0x0

Bits	Name	Description	Type	Default
20	S_PDCMRETREQ0	Enable sensitivity to PDCMRETREQn[0] signal. If set to '1', PD_VMR<x> stays in ON or RET if [PDCMRETREQn0] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RW	0x0
19	S_PDCMONREQ3	Enable sensitivity to [PDCMONREQn3] signal. If set to '1', D_VMR<x> stays ON if PDCMONREQn[3] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RW	0x0
18	S_PDCMONREQ2	Enable sensitivity to PDCMONREQn[2] signal. If set to '1', PD_VMR<x> stays ON if PDCMONREQn[2] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RW	0x0
17	S_PDCMONREQ1	Enable sensitivity to PDCMONREQn[1] signal. If set to '1', PD_VMR<x> stays ON if PDCMONREQn[1] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RW	0x0
16	S_PDCMONREQ0	Enable sensitivity to PDCMONREQn[0] signal. If set to '1', PD_VMR<x> stays ON if PDCMONREQn[0] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RW	0x0
15	S_PD_MGMT_ON	Tied to LOW. Ignores PD_MGMT power state. This bit is reserved and RAZ/WI if PILEVEL < 2.	RO	0x0
14	-	Reserved.	RAZ/WI	0x0
13	S_PD_DEBUG_ON	Tied to LOW. Ignores PD_DEBUG power state.	RO	0x0
12	S_PD_CRYPTON_ON	Tied to LOW. Ignores PD_CRYPTON power state. This bit is reserved and RAZ/WI if HASCRYPTO = 0.	RO	0x0
11:9	-	Reserved.	RAZ/WI	0x000
8	S_PD_NPU3_ON	Enable PD_NPU3 sensitivity. If set to '1', PD_VMR<x> stays ON if PD_NPU3 is ON. This bit is reserved and RAZ/WI if NUMNPU < 4.	RW	0x0
7	S_PD_NPU2_ON	Enable PD_NPU2 sensitivity. If set to '1', PD_VMR<x> stays ON if PD_NPU2 is ON. This bit is reserved and RAZ/WI if NUMNPU < 3.	RW	0x0
6	S_PD_NPU1_ON	Enable PD_NPU1 sensitivity. If set to '1' PD_VMR<x> stays ON if PD_NPU1 is ON. This bit is reserved and RAZ/WI if NUMNPU < 2.	RW	0x0
5	S_PD_NPU0_ON	Enable PD_NPU0 sensitivity. If set to '1' PD_VMR<x> stays ON if PD_NPU0 is ON. This bit is reserved and RAZ/WI if NUMNPU < 1.	RW	0x0
4	S_PD_CPU3_ON	Enable PD_CPU3 sensitivity. If set to '1', PD_VMR<x> stays ON if PD_CPU3 is ON. This bit is reserved and RAZ/WI if NUMCPU < 3.	RW	0x0
3	S_PD_CPU2_ON	Enable PD_CPU2 sensitivity. If set to '1', PD_VMR<x> stays ON if PD_CPU2 is ON. This bit is reserved and RAZ/WI if NUMCPU < 2.	RW	0x0
2	S_PD_CPU1_ON	Enable PD_CPU1 sensitivity. If set to '1' PD_VMR<x> stays ON if PD_CPU1 is ON. This bit is reserved and RAZ/WI if NUMCPU < 1.	RW	0x0
1	S_PD_CPU0_ON	Enable PD_CPU0 sensitivity. If set to '1' PD_VMR<x> stays ON if PD_CPU0 is ON.	RW	0x0
0	S_PD_SYS_ON	Tied to LOW. Ignores PD_SYS power state.	RO	0x0

6.6.2.19 PDCM_PD_MGMT_SENSE

The Power Dependency Control Matrix PD_MGMT Power Domain Sensitivity register is used to define what keeps the PD_MGMT domains awake.

Configurations

This register implementation depends on the configuration of individual fields. When PILEVEL < 2, this register does not exist and the register area it occupies are Reserved, and **RAZ/WI**.

Attributes

Width

32-bit

Power domain

This register resides in the PD_AON power domain but can also reside in PD_MGMT power domain if its states are saved and restored when entering and then leaving the lower power state, respectively.

Reset

This register is reset by **nWARMRESETAON**.

Usage constraints

This register is Secure privileged access only. For write access to this register, only 32-bit writes are supported. Any byte and halfword writes are ignored.

Bit descriptions

Table 6-78: PDCM_PD_MGMT_SENSE bit descriptions

Bits	Name	Description	Type	Default
31:30	MIN_PWR_STATE	Defines the Minimum Power State, when PD_MGMT is trying to enter a lower power state: <ul style="list-style-type: none"> '00': Minimum power state is OFF, Others: Reserved. 	RO	0x0
29:24	-	Reserved.	RAZ/WI	0x0
23	S_PDCMRETQREQ3	Enable sensitivity to PDCMRETQREQn[3] signal. If set to '1', PD_MGMT stays ON if PDCMRETQREQn[3] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RW	0x0
22	S_PDCMRETQREQ2	Enable sensitivity to PDCMRETQREQn[2] signal. If set to '1', PD_MGMT stays ON if PDCMRETQREQn[2] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RW	0x0
21	S_PDCMRETQREQ1	Enable sensitivity to PDCMRETQREQn[1] signal. If set to '1', PD_MGMT stays ON if PDCMRETQREQn[1] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RW	0x0
20	S_PDCMRETQREQ0	Enable sensitivity to PDCMRETQREQn[0] signal. If set to '1', PD_MGMT stays ON if PDCMRETQREQn[0] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RW	0x0
19	S_PDCMONQREQ3	Enable sensitivity to PDCMONQREQn[3] signal. If set to '1', PD_MGMT stays ON if PDCMONQREQn[3] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 4.	RW	0x0

Bits	Name	Description	Type	Default
18	S_PDCMONQREQ2	Enable sensitivity to PDCMONQREQn[2] signal. If set to '1', PD_MGMT stays ON if PDCMONQREQn[2] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 3.	RW	0x0
17	S_PDCMONQREQ1	Enable sensitivity to PDCMONQREQn[1] signal. If set to '1', PD_MGMT stays ON if PDCMONQREQn[1] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 2.	RW	0x0
16	S_PDCMONQREQ0	Enable sensitivity to PDCMONQREQn[0] signal. If set to '1', PD_MGMT stays ON if PDCMONQREQn[0] signal is HIGH. This bit is reserved and RAZ/WI if PDCMQCHWIDTH < 1.	RW	0x0
15	S_PD_MGMT_ON	Enable sensitivity to PD_MGMT power state. If set to '1' PD_MGMT stays ON if PD_MGMT is already ON.	RW	0x01
14	-	Reserved.	RAZ/WI	0x0
13	S_PD_DEBUG_ON	Tied to HIGH. PD_MGMT stays ON if PD_DEBUG power domain is ON.	RO	0x01
12	S_PD_CRYPTON_ON	Tied to HIGH. PD_MGMT stays ON if PD_CRYPTON power domain is ON. This bit is reserved and RAZ/WI if HASCRYPTO = 0.	RO	0x01
11:9	-	Reserved.	RO	0x000
8	S_PD_NPU3_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_NPU3 is ON. This bit is reserved and RAZ/WI if NUMNPU < 4.	RO	0x01
7	S_PD_NPU2_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_CPU2 is ON. This bit is reserved and RAZ/WI if NUMNPU < 3.	RO	0x01
6	S_PD_NPU1_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_NPU1 is ON. This bit is reserved and RAZ/WI if NUMCPU < 2.	RO	0x01
5	S_PD_NPU0_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_NPU0 is ON. This bit is reserved and RAZ/WI if NUMNPU < 1.	RO	0x01
4	S_PD_CPU3_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_CPU3 is ON. This bit is reserved and RAZ/WI if NUMCPU < 3.	RO	0x01
3	S_PD_CPU2_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_CPU2 is ON. This bit is reserved and RAZ/WI if NUMCPU < 2.	RO	0x01
2	S_PD_CPU1_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_CPU1 is ON. This bit is reserved and RAZ/WI if NUMCPU < 1.	RO	0x01
1	S_PD_CPU0_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_CPU0 is ON.	RO	0x01
0	S_PD_SYS_ON	Tied to HIGH. PD_MGMT always tries to stay ON if PD_SYS is ON.	RO	0x01

6.7 CPU Private Peripheral Bus region

Each CPU, as defined by the ARMv8-M architecture specification, hosts a local Private Peripheral Bus Region (PPB) at address 0xE000_0000 to 0xE00F_FFFF. This region is typically for integration with CoreSight debug and trace components that is normally local to each CPU and is not intended for general peripheral usage.

In CRSAS Ma1, this region has the memory map as shown in the table below. Other than the EWIC, CPU <n> ROM Table, and the peripherals on the External PPB Expansion that is added by an integrator, the existence of all other components depends on the CPU implementation and

configuration. See *Arm® Cortex®-M55 Processor Technical Reference Manual* and *Arm® Cortex®-M85 Processor Technical Reference Manual*.

Not included in the table are TPIU and the ETB, along with a system level debug CoreSight ROM table. This ROM table is also referred to as MCU ROM Table in CPU Technical Reference Manual. Depending on HASCSS:

- If HASCSS = 0 where NUMCPU must be 0, the TPIU, ETB, and the MCU ROM can be integrated using the CPU<n> EPPB Interface, with the MCU ROM pointing to the TPIU, ETB, and the core's internal ROM table. We recommend that in this case, these are integrated at the following addresses:
 - TPIU at 0xE004_0000
 - ETB at 0xE004_5000
 - MCU ROM table at 0xE00F_E000 where an external DAP-Lite or AHB-AP point to.
- If HASCSS = 1, the TPIU and the ETB are be provided

CRSAS Ma1 in the Shared Debug System accessible through a separate MEM-AP. The MCU ROM table can still be added to the EPPB if other debug related components local to the CPU exist so that the MCU ROM table can point to them. Note however that in such a system, the MCU ROM table does not point to the core's internal ROM. Instead, the CPU<n> ROM table provided points to the core's internal ROM. We recommend that the MCU ROM table, if it exists is integrated at 0xE00F_E000.

When HASCSS = 1, Arm does not recommend using this region to expand the debug system, unless the intention is to implement independent debug infrastructure for each CPU. Instead the system integrator should add to the Debug APB Expansion Interface.

For more information on the ETM, CTI, PMC, SBIST and ROM Table, see *Arm® Cortex®-M55 Processor Technical Reference Manual* and *Arm® Cortex®-M85 Processor Technical Reference Manual*.

Table 6-79: CPU <n> Private Peripheral Bus Region

Row ID	From address	To address	Size	Region name	Description
1	0xE004_0000	0xE004_0FFF	4KB	PPB Expansion	CPU <n> Expansion PPB Interface.
2	0xE004_1000	0xE004_1FFF	4KB	ETM ¹	Embedded Trace Module.
3	0xE004_2000	0xE004_2FFF	4KB	CTI ¹	Cross Trigger Interface.
4	0xE004_3000	0xE004_4FFF	8KB	Reserved	Reserved
5	0xE004_5000	0xE004_5FFF	4KB	PPB Expansion	CPU <n> Expansion PPB Interface.
6	0xE004_6000	0xE004_6FFF	4KB	PMC ¹	Programmable MBIST Controller.
7	0xE004_7000	0xE004_7FFF	4KB	EWIC	External Wakeup Interrupt Controller. See EWIC .
8	0xE004_8000	0xE004_8FFF	4KB	PPB Expansion	Reserved for SBIST - Software Based Build In Self Test. Routed to CPU<n> Expansion PPB Interface on PD_CPU<n> power domain.
9	0xE004_9000	0xE00F_EFFF	24KB	PPB Expansion	CPU <n> Expansion PPB Interface.

Row ID	From address	To address	Size	Region name	Description
10	0xE00F_F000	0xE00F_FFFF	4KB	ROM Table	CPU <n> Internal ROM Table.

¹ The existence of the following components is dependent of the configuration and the supported features of the CPU integrated. If they do not exist, these regions are reserved and **RAZ/WI** response when accessed.

- ETM
- CTI
- PMC

6.7.1 EWIC

CRSAS Ma1 is designed to always support an External Wakeup Interrupt Controller (EWIC) for each CPU in the system. This allows the system to support each CPU being switched off independently, and for the EWIC to run at a lower clock rate to help reduce PD_AON dynamic and leakage power consumption.

All EWICs reside in the PD_AON power domain, run on **AONCLK**, and reside in the **nWARMRESETAON** reset domain.

Each EWIC <n> is only accessible through the Private Peripheral Bus Region that is associated with CPU <n> at address 0xE004_7000 to 0xE004_7FFF.

For more information, see *Arm® Cortex®-M55 Processor Technical Reference Manual* and *Arm® Cortex®-M85 Processor Technical Reference Manual*.

6.8 Debug System Access Region

CRSAS Ma1 supports two key configuration options for the debug system:

- HASCSS = 0. CoreSight SoC-600 based Debug System does not exist. See [CoreSight SoC-600 based Debug System not implemented](#).
- HASCSS = 1. CoreSight SoC-600 based Debug System exists. See [CoreSight SoC-600 based Debug System implemented](#).

6.8.1 CoreSight SoC-600 based Debug System not implemented

The HASCSS parameter lets you define if the CoreSight SoC-600 based Debug System is implemented:

- CoreSight SoC-600 based Debug System does not exist, HASCSS = 1.

- CoreSight SoC-600 based Debug System exists, HASCSS = 0.

When the Common CoreSight Debug Infrastructure does not exist, the Debug System Access Region is not used and therefore the following regions are reserved. These regions when accessed return a bus error response.

- 0xE010_0000 to 0xE01F_FFFF
- 0xF010_0000 to 0xF01F_FFFF

Without the CoreSight based Debug System, All Debug related interfaces of each processor core are made available as expansion interfaces. A system integrator has to create a debug infrastructure to provide debug access to each core. This infrastructure has to use the debug authentication and debug access control signals to control accessibility to the cores and the system. For more information on these signals, see [Debug Authentication interface](#).

6.8.2 CoreSight SoC-600 based Debug System implemented

When the CoreSight based System Debug infrastructure exists within an implementation of CRSAS Ma1 (HASCSS=1), the system provides several Memory Access Ports (MEM-APs) to provide access to each CPU and to the shared debug components in the system.

These ports are mapped to the Debug system region with a relative address map as shown in the following table.

Table 6-80: Debug system address map

Row ID	From address	To address	Size	Region name	Description
1	0x0000	0x0FFF	4KB	DSROM	Debug System ROM.
2	0x1000	0x1FFF	4KB	Reserved	Reserved
3	0x2000	0x3FFF	8KB	SYS APB-AP	Shared Debug System Access Port.
4	0x4000	0x5FFF	8KB	CPU0 AHB-AP	CPU 0 Debug System Access Port.
5	0x6000	0x7FFF	8KB	CPU1 AHB-AP	CPU 1 Debug System Access Port. This AP does not exist when NUMCPU < 1, and the region is reserved.
6	0x8000	0x9FFF	8KB	CPU2 AHB-AP	CPU 2 Debug System Access Port. This AP does not exist when NUMCPU < 2, and the region is reserved.
7	0xA000	0xBFFF	8KB	CPU3 AHB-AP	CPU 3 Debug System Access Port. This AP does not exist when NUMCPU < 3, and the region is reserved.
8	0xC000	0xF_FFFF		Reserved	Reserved

This region is accessible through:

- The Debug Access Interface with an address offset of 0x0000. Access is first controlled using the Debug Authentication Access Control signal **DAPDSSACCEN**. When **DAPDSSACCEN** = 0, accesses from the Debug Access Interface are blocked and return error responses. If **DAPDSSACCEN** = 1, accesses are allowed to pass through. Note that a separate **DAPACCEN** is provided to allow the integrator to control a DAP interface directly to stop access even

reaching the Debug Access Interface in the first place. This allows debug components outside the subsystem, with **DAPACCEN** = 1, to still be accessible while **DAPDSSACCEN** = 0 stops accesses arriving at the subsystem Debug Access Interface from accessing the system.

- The Main and Peripheral Interconnect are at the following two aliased address offsets:
 - 0xE010_0000 is the Non-secure alias
 - 0xF010_0000 is the Secure alias

Access from the interconnect is gated by the Debug Authentication Access Control signal, **SYSDSSACCEN<n>**. When **SYSDSSACCEN<n>** = 0, accesses from CPU <n> through the Main Interconnect are blocked and return error responses. If **SYSDSSACCEN<n>** = 1, accesses from CPU <n> are allowed to pass through.

Once access can pass the first gate controlled by **SYSDSSACCEN<n>** into the Debug System, a PPC0 is then used to perform the final mapping of all APs either to the Non-secure region from 0xE010_0000 to 0xE01F_FFFF or to the Secure region from 0xF010_0000 to 0xF01F_FFFF. For more information, see [PERIPHNSPPPC0](#), [PERIPHNSPPC1](#), [PERIPHSPPPC0](#), [PERIPHSPPPC1](#), [PERIPHNSPPPC0](#), and [PERIPHNSPPPC1](#).

- Each Memory Access Port (MEM-AP) is a twin MEM-AP that enables an external debugger to use one logical MEM-AP, and on-chip software to use a separate logical MEM-AP. A twin MEM-AP consists of two sets of 4K registers. The bottom 4K is for the external debugger accesses exclusively while the top 4K is for on-chip software accesses exclusively. It is **IMPLEMENTATION DEFINED** if the external debugger only has access to the top 4K and vice versa if the on-chip software only has access to the bottom 4K.

The following sections list the memory map behind each MEM-AP and the contents of CoreSight Debug ROMs.

6.8.2.1 Shared debug system MEM-AP memory map

The Shared debug system MEM-AP provides access to the Shared debug system that all CPUs in the system shares. It includes a trace funnel for funneling all trace data together, an Embedded Trace Buffer (ETB) that allows trace data to be stored and read by a debugger. In addition, it provides access to debug components that reside in the expansion system through the Debug APB Expansion Interface.

The MEM-AP's base address static configuration is configured to point to Shared debug system CoreSight ROM Table at address 0x0000_0000.

The following table shows the memory map that is visible to the Shared Debug System MEM-AP.

Table 6-81: Shared debug system MEM-AP memory map

Row ID	From address	To address	Size	Region name	Description
1	0x0000_0000	0x0000_0FFF	4KB	SDSROM	Shared Debug System ROM Table.
2	0x0000_1000	0x0000_1FFF	4KB	SDSFUNNEL	Shared Debug System Trace Funnel.
3	0x0000_2000	0x0000_2FFF	4KB	SDSCTI	Shared Debug System Cross Trigger Interface.
4	0x0000_3000	0x0000_3FFF	4KB	SDSETB	Shared Debug System Embedded Trace Buffer.

Row ID	From address	To address	Size	Region name	Description
5	0x0000_4000	0x0000_FFFF	-	Reserved	Reserved
6	0x0001_0000	0xFFFF_FFFF	-	Debug APB Expansion Interface	Debug APB Expansion Interface Region.

6.8.2.2 CPU<n> debug system MEM-AP memory map

The CPU<n> Debug System Memory Access Port provides access to the following;

- CPU<n> Debug CoreSight ROM Table, which the CPU<n> MEM-AP's base address static configuration is configured to point to. This ROM Table includes Granular Power Requester (GPR) functionality to allow software to wake CPU<n>.
- CPU's Debug access interface, which provides access to the processor control and debug logic, and to a view of memory which is consistent with that observed by CPU<n>. This includes the CPU's PPB region, where other CPU<n> private debug components can be hosted. For more information, see [CPU Private Peripheral Bus region](#) and *Arm® Cortex®-M55 Processor Technical Reference Manual* and *Arm® Cortex®-M85 Processor Technical Reference Manual*.

The following table shows the memory map of the memory map as seen by the CPU<n> MEM-AP.

Table 6-82: CPU<n> MEM-AP memory map

Row ID	From address	To address	Size	Region name	Description
1	0x0000_0000	0xF000_7FFF	-	SYSACC	System Memory Access through CPU<n> Debug Access Interface.
2	0xF000_8000	0xF000_8FFF	4KB	CPU<n>ROM	CPU<n> Debug ROM Table.
3	0xF000_9000	0x0000_9FFF	4KB	Reserved	Reserved
4	0xF000_A000	0xFFFF_FFFF	-	SYSACC	System Memory Access through CPU<n> Debug Access Interface.

6.8.2.3 DSROM, Debug System ROM

The Debug System ROM is a CoreSight Class 0x1 ROM table that provides a list of pointers to Access Port within the Debug System.

The following table lists the contents of the Debug System ROM Table.

Table 6-83: Debug System CoreSight ROM Table contents

Offset	Name	Type	Reset	Width	Description
0x000	ROMENTRY0	RO	0x0000_2001	32-bit	ROM entry pointing to the Shared Debug System MEM-AP.
0x004	ROMENTRY1	RO	0x0000_4001	32-bit	ROM entry pointing to Debug System CPU0 MEM-AP.
0x008	ROMENTRY2	RO	0x0000_6001	32-bit	ROM entry pointing to CPU1 Debug System MEM-AP. This register is reserved and RAZ/WI if NUMCPU < 1.
0x00C	ROMENTRY3	RO	0x0000_8001	32-bit	ROM entry pointing to CPU2 Debug System MEM-AP. This register is reserved and RAZ/WI if NUMCPU < 2.
0x010	ROMENTRY4	RO	0x0000_A001	32-bit	ROM entry pointing to CPU3 Debug System MEM-AP. This register is reserved and RAZ/WI if NUMCPU < 3.

Offset	Name	Type	Reset	Width	Description
0x014 – 0xFC8	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFCC	MEMTYPE	RO	0x0000_0000	32-bit	Memory Type Register. MEMTYPE[0] = '0' indicates that the current debug bus does not include system memory.
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_004B	32-bit	Peripheral ID 0: PIDR0[7:0] - Part Number bits [7:0].
0xFE4	PIDR1	RO	0x0000_00B7	32-bit	Peripheral ID 1: PIDR1[3:0] - Part Number [11:8], PIDR1[7:4] - JEP106 Identity Code [3:0].
0xFE8	PIDR2	RO	0x0000_000B	32-bit	Peripheral ID 2: PIDR2[2:0] - JEP106 Identity Code [6:4], PIDR2[3] - JEDEC identifier, PIDR2[7:4] - Revision Code.
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_0010	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.8.2.4 SDSROM, Shared Debug System ROM

The Shared Debug System Debug CoreSight ROM is a CoreSight Class 0x1 ROM table that provides a list of pointers to the following:

- CoreSight components within the Shared Debug System level.
- An external CoreSight ROM through the Debug APB Expansion Interface.

The following table lists the contents of the Debug System CoreSight ROM.

Table 6-84: Debug System ROM Table contents

Offset	Name	Type	Reset	Width	Description
0x000	ROMENTRY0	RO	0x0000_1001	32-bit	ROM entry pointing to the trace funnel.
0x004	ROMENTRY1	RO	0x0000_2001	32-bit	ROM entry pointing to the Cross Trigger Interface.
0x008	ROMENTRY2	RO	0x0000_3001	32-bit	ROM entry pointing to the Embedded Trace Buffer
0x00C	ROMENTRY3	RO	0x0000_4001	32-bit	ROM entry pointing to an external CoreSight ROM table at address 0x0008_0000 through the Debug APB Expansion Interface.

Offset	Name	Type	Reset	Width	Description
0x010 – 0xFC8	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xFCC	MEMTYPE	RO	0x0000_0000	32-bit	Memory Type Register. MEMTYPE[0] = '0' indicates that the current debug bus does not include system memory.
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_004C	32-bit	Peripheral ID 0: PIDR0[7:0] - Part Number bits [7:0].
0xFE4	PIDR1	RO	0x0000_00B7	32-bit	Peripheral ID 1: PIDR1[3:0] - Part Number [11:8], PIDR1[7:4] - JEP106 Identity Code [3:0].
0xFE8	PIDR2	RO	0x0000_000B	32-bit	Peripheral ID 2: PIDR2[2:0] - JEP106 Identity Code [6:4], PIDR2[3] - JEDEC identifier, PIDR2[7:4] - Revision Code.
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_0010	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

6.8.2.5 CPU<n>ROM, CPU<n> Debug System ROM

The CPU<n> Debug System ROM is a CoreSight Class 0x9 ROM table that provides a list of pointers to the following:

- CoreSight ROM in the processor core.
- An optional CoreSight ROM, call the MCUROM residing on the EPPB. It also provides GPR functional to allow a debugger to request the processor to turn on.

The following table lists the contents of the CPU<n> Debug ROM.

Table 6-85: CPU<n> Debug System ROM Table contents

Offset	Name	Type	Reset	Width	Description
0x000	ROMENTRY0	RO	0xE00F_F007	32-bit	ROM entry pointing to the CPU's internal CoreSight ROM table and provides a POWERID value of 0x0.

Offset	Name	Type	Reset	Width	Description
0x004	ROMENTRY1	RO	CFG_DEF	32-bit	ROM entry pointing to the MCU ROM CoreSight Debug ROM table: ROMENTRY1[31:12] = CPU<n> MCUROMADDR[31:12], ROMENTRY1[11:9] = 0x0, ROMENTRY1[8:4] = POWERID = 0x0, ROMENTRY[3] = 0b0, ROMENTRY[2] = 0b1, ROMENTRY[1:0] = {CPU<n>MCUROMVALID, CPU<n>MCU ROMVALID}.
0x010 – 0xAFC	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xA00	DBGPCRO	RW	0x0000_0001	32-bit	Debug Power Control Register 0. For requesting the CPU to turn ON.
0xA04 – 0xA7C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xA80	DBGPSRO	RO	0x0000_0000	32-bit	Debug Power Status Register 0
0xA84 – 0xBFC	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xC00	PRIDRO	RO	0x0000_0001	32-bit	Power Request Identification Register
0xC04 – 0xFB4	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xF00	ITCTRL	RO	0x0000_0000	32-bit	Integration Mode Control Register.
0xF04 – 0xF9C	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xFA0	CLAIMSET	RO	0x0000_0000	32-bit	Claim Tag Set Register.
0xFA4	CLAIMCLR	RO	0x0000_0000	32-bit	Claim Tag Clear Register.
0xFA8	DEVAFF0	RO	0x0000_0000	32-bit	Device Affinity Register 0
0xFAC	DEVAFF1	RO	0x0000_0000	32-bit	Device Affinity Register 1
0xFB0	LAR	RO	0x0000_0000	32-bit	Lock Access Register
0xFB0	LSR	RO	0x0000_0000	32-bit	Lock Status Registers
0xFB8	AUTHSTATUS	RO	CFG_DEF	32-bit	Authentication Status Register. Values are defined by Debug Authentication values.
0xFBC	DEVARCH	RO	0x4770_0AF7	32-bit	Device Architecture Register.
0xFC0	DEVID2	RO	0x0000_0000	32-bit	Device ID Registers 2
0xFC4	DEVID1	RO	0x0000_0000	32-bit	Device ID Registers 1
0xFC8	DEVID	RO	0x0000_0030	32-bit	Device ID Registers
0xFCC	DEVTYPE	RO	0x0000_0000	32-bit	DEVTYPE
0xFD0	PIDR4	RO	0x0000_0004	32-bit	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/ WI	0x0000_0000	32-bit	Reserved
0xFE0	PIDR0	RO	0x0000_004D	32-bit	Peripheral ID 0: PIDR0[7:0] - Part Number bits [7:0].

Offset	Name	Type	Reset	Width	Description
0xFE4	PIDR1	RO	0x0000_00B7	32-bit	Peripheral ID: PIDR1[3:0] - Part Number [11:8], PIDR1[7:4] - JEP106 Identity Code [3:0].
0xFE8	PIDR2	RO	0x0000_000B	32-bit	Peripheral ID 2: PIDR2[2:0] - JEP106 Identity Code [6:4], PIDR2[3] - JEDEC identifier, PIDR2[7:4] - Revision Code.
0xFEC	PIDR3	RO	0x0000_0000	32-bit	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	32-bit	Component ID 0
0xFF4	CIDR1	RO	0x0000_0010	32-bit	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	32-bit	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	32-bit	Component ID 3

Appendix A System timer components

The system timer components are:

- The System Counter generates a timestamp value that can be shared across the System on Chip (SoC).
- The System Timer can raise an interrupt when a period has elapsed.
- System Watchdog provides a mechanism to detect errant system behavior causing reset of the system if a period elapses without intervention.

The components are software-programmable using APB interfaces.

A.1 System Counter overview

This section provides an overview of the System Counter.

The System Counter has the following features:

- Binary encoded, unsigned, 64-bit up-counter, starts counting from 0, with the optional ability to start counting from a different preload value.
- Generates a 64-bit time value that compatible components across the SoC can share.
- Internal 24-bit fractional value to allow fine count resolution control.
- Ability to scale counter clock frequency, therefore allowing operation at lower frequency during low-power mode.
- Hardware can handle dynamic clock switching between different frequencies. Software is required only for initialization.
- Support of software enabled halt-on-debug from external CoreSight Cross Trigger Interface (CTI).

A.2 Counter operation flows

This section describes pseudo-sequences for some of the commonly used Counter programming flows.

Counter initialization

To initialize the counter, perform the following steps:

1. Ensure that CLKSEL is 0x01 and that REFCLK is running.
2. Disable the counter by setting CNTCR.EN = 0.
3. Configure the SoC clock generation to generate the frequencies that are required for the two clock sources.

4. Write to CNTSCR0 to set the scaling value for REFCLK clock source.
5. Write to CNTSCR1 to set the scaling value for FASTCLK clock source.
6. Enable the counter by setting CNTCR.EN = 1.
7. CLKSEL can now be changed at any time to switch the FCLK source between REFCLK and FASTCLK.

Changing scaling registers

To change the scaling registers, perform the following steps:

1. Ensure that REFCLK is running.
2. Disable the counter by setting CNTCR.EN = 0.
3. Write new values to CNTSCR0 and CNTSCR1.
4. Enable the counter by setting CNTCR.EN = 1.

A.3 System counter Programmers model

This section describes programmers model for the System Counter. Registers in the System Counter provide the following functions:

- Enabling and disabling the counter.
- Setting the counter value.
- Changing the operating mode, to change the frequency and increment value.
- Enabling Halt-on-debug, that a debugger can then use to suspend counting.
- Providing status of the Counter value in addition to the operating mode.

These registers are grouped into two 4KB frames:

- A control frame, CNTControlBase.
- A status frame, CNTReadBase.

The base addresses of these frames are **IMPLEMENTATION DEFINED**. Similarly, the security level of each frame is also **IMPLEMENTATION DEFINED**, however, in a system that supports both, Secure and Non-secure memory maps, CNTControlBase is only accessible by Secure memory accesses.

A.3.1 CNTControlBase registers summary

This section provides a summary of the CNTControlBase Frame Registers (System Counter).

The following table shows a summary of the CNTControlBase Frame Registers (System Counter).

Table A-1: CNTControlBase Frame Registers (System Counter)

Offset	Name	Access	Reset value	Description
0x000	CNTCR	RW	0x0000_0000	Counter Control Register. See CNTCR, Counter Control register
0x004	CNTSR	RO	0x0000_000X	Counter Status Register. See CNTSR, Counter Status register
0x008	CNTCV[31:0]	RW	0xFFFF_XXXX	Counter Count Value register. See CNTCV, Counter Count Value register
0x010	CNTSCR	RW	0x0100_0000	Counter Counter Scale register. See CNTSCR, Counter Scale register
0x014-0x018	Reserved	RAZ/WI	0x0000_0000	Reserved
0x01C	CNTID	RO	0x000X_000X	Counter ID register. See CNTID, Counter ID register
0x020-0x0BC	Reserved	RAZ/WI	0x0000_0000	Reserved
0x0C0-0x0CC	Reserved	RAZ/WI	0x0000_0000	Reserved
0x0D0	CNTSRO	RW	0x010_0000	Counter Scale Register 0. See CNTSRO, CNTSCR1, Counter Scaling registers
0x0D4	CNTSR1	RW	0x010_0000	Counter Scale Register 1. See CNTSRO, CNTSCR1, Counter Scaling registers
0x0D8-0xFCC	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFD0	CNTPIDR4	RO	0x0000_0004	Peripheral Identification Register 4.
0xFD4-0xFDC	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFE0	CNTPIDR0	RO	0x0000_00BA	Peripheral Identification Register 0.
0xFE4	CNTPDR1	RO	0x0000_00B0	Peripheral Identification Register 1.
0xFE8	CNTPIDR2	RO	0x0000_000B	Peripheral Identification Register 2.
0xFEC	CNTPIDR3	RO	0x0000_0000	Peripheral Identification Register 3.
0xFF0	CNTCIDR0	RO	0x0000_000D	Component Identification Register 0.
0xFF4	CNTCIDR1	RO	0x0000_00F0	Component Identification Register 1.
0xFF8	CNTCIDR2	RO	0x0000_0005	Component Identification Register 2.
0xFFC	CNTCIDR3	RO	0x0000_00B1	Component Identification Register 3.



CNTSCR is aliased with CNTSRO, meaning that either addresses of CNTSCR and CNTSRO physically access a single register.

A.3.2 CNTReadBase registers summary

This section provides a summary of the CNTReadBase registers.

The following table shows a summary of the CNTReadBase Frame Registers (System Counter).

Table A-2: CNTReadBase Frame Registers (System Counter)

Offset	Name	Access	Reset value	Description
0x000	CNTCV[31:0]	RO	0xFFFF_XXXX	Counter Count Value register. See CNTCV, Counter Count Value register
0x004	CNTCV[63:32]	RO	0xFFFF_XXXX	Counter Count Value register. See CNTCV, Counter Count Value register
0x008-0xFCC	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFD0	CNTPIDR4	RO	0x0000_0004	Peripheral Identification Register 4.

Offset	Name	Access	Reset value	Description
0xFD4-0xFDC	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFE0	CNTPIDR0	RO	0x0000_00BB	Peripheral Identification Register 0.
0xFE4	CNTPIDR1	RO	0x0000_00B0	Peripheral Identification Register 1.
0xFE8	CNTPIDR2	RO	0x0000_000B	Peripheral Identification Register 2.
0xFEC	CNTPIDR3	RO	0x0000_0000	Peripheral Identification Register 3.
0xFF0	CNTCIDR0	RO	0x0000_000D	Component Identification Register 0.
0xFF4	CNTCIDR1	RO	0x0000_00F0	Component Identification Register 1.
0xFF8	CNTCIDR2	RO	0x0000_0005	Component Identification Register 2.
0xFFC	CNTCIDR3	RO	0x0000_00B1	Component Identification Register 3.

A.3.3 Register descriptions

This section describes each System Counter register.

All registers are 32-bit and must be accessed using 32-bit reads and writes.

A.3.4 CNTCR, Counter Control register

The CNTCR register enables the counter, controls the counter frequency setting, and controls counter behavior during debug.

The following table shows the bit assignments.

Table A-3: CNTCR register bit assignments

Bits	Name	Access	Reset value	Function
31:6	-	RAZ/WI	0x0000	Reserved
5	INTRCLR	RW	0x0	Interrupt clear bit, only writes of 0 are permitted, and writes of 1 are ignored. If APB logic is powered off when the interrupt output is asserted, this bit is cleared automatically. Therefore, it is the responsibility of software to ensure that there is no pending interrupt before powering off the APB logic.
4	PSLVERRDIS	RW	0x0	PSLVERR output disable: <ul style="list-style-type: none"> 0: PSLVERR permanently driven to '0'. 1: PSLVERR output that the System Counter generates dynamically.
3	INTRMASK	RW	0x0	Interrupt mask: <ul style="list-style-type: none"> 0: Interrupt output disabled. 1: Interrupt output enabled.

Bits	Name	Access	Reset value	Function
2	SCEN	RW	0x0	<p>Scale enable:</p> <ul style="list-style-type: none"> 0: Scaling is not enabled. The Counter value is incremented by > 0x01.000000 for each counter tick. 1: Scaling is enabled. The counter is incremented by the ScaleVal for > each counter tick. <p>The ScaleVal value that the System Counter uses is from CNTSCR, CNTSCR[0], or CNTSCR[1]. See CNTSCR0, CNTSCR1, Counter Scaling registers</p>
1	HDBG	RW	0x0	<p>Halt On Debug:</p> <ul style="list-style-type: none"> 0: HALTREQ signal into the Counter has no effect. 1: HALTREQ signal into the Counter halts the Count.
0	EN	RW	0x0	<p>Enable Counter:</p> <ul style="list-style-type: none"> 0: Disabled: Count is not incrementing. 1: Enabled: Count is incrementing.

A.3.5 CNTSR, Counter Status register

The CNTSR register provides Counter frequency status information. The following table shows the bit assignments.

Table A-4: CNTSR register bit assignments

Bits	Name	Access	Reset value	Function
31:2	-	RAZ/WI	0x000000	Reserved
1	DBGH	RO	Unknown	<p>Indicates whether the counter is halted because the Halt-on-Debug signal is asserted:</p> <ul style="list-style-type: none"> 0: Counter is not halted. 1: Counter is halted.
0	-	RAZ/WI	0x0	Reserved

A.3.6 CNTCV, Counter Count Value register

The CNTCV register indicates the current count value. The following table shows the bit assignments.

Table A-5: CNTCV register bit assignments

Bits	Name	Access	Function
63:0	CountValue	<ul style="list-style-type: none"> RO from CNT ReadBase RW from CNT ControlBase 	Indicates the count value.

A.3.7 CNTSCR, Counter Scale register

The CNTSCR registers store the Counter Scaling value. The following table shows the bit assignments.

Table A-6: CNTSCR register bit assignments

Bits	Name	Access	Reset value	Function
31:0	ScaleVal	RW	0x0100_0000	<p>When counter scaling is enabled, ScaleVal is the amount added to the Counter Count Value for every period of the counter as determined by 1/Frequency from the current operating frequency of the system counter, the <i>counter tick</i>.</p> <p>ScaleVal is expressed as an unsigned fixed-point number with an 8-bit integer value and a 24-bit fractional value.</p> <p>The CNTSCR register can only be changed when the counter is disabled, that is, CNTCR.EN=0. If the value of CTNSCR changes when CNTCR.EN==1, then the Counter Count Value becomes UNKNOWN and remains UNKNOWN on future ticks of the clock.</p>



If CNTSC=0, CNTSCR is permanently driven to 0x0100_0000.

A.3.8 CNTID, Counter ID register

The CNTID register indicates additional information about Counter Scaling implementation. The following table shows the bit assignments.

Table A-7: CNTID register bit assignments

Bits	Name	Access	Reset value	Function
31:20	-	RAZ/WI	0x0000	Reserved
19	CNTSCR_OVR	RO	Defined by parameter with a default value of 0x0.	<p>Override counter enable condition for writing to CNTSCR* registers:</p> <ul style="list-style-type: none"> 0: CNTSCR* can be written only when CNTCR.EN=- 0: 1: CNTSCR* can be written when CNTCR.EN=0 or - 1:

Bits	Name	Access	Reset value	Function
18:17	CNTSELCLK	RO	0x01	<p>Indicates the clock source that the Counter is using. Based on the settings for Counter scaling, the Counter increment value is chosen either from one of the CNTSCR registers or is fixed to 1.0 when scaling is disabled:</p> <ul style="list-style-type: none"> 0: Invalid status, Counter not incrementing. 1: CLK0 (REFCLK). 10: CLK1 (FASTCLK). 11: Invalid status, counter not incrementing. <p>These bits are only valid when hardware clock switching is implemented (HWCLKSW=1).</p> <p>If HWCLKSW=0, these bits read a constant value of 0x01 regardless of the value of TSCLKSEL input and Counter increment value is used from CNTSCR0 or fixed to 1.0 depending on the status of Counter scaling feature.</p>
16	CNTCS	RO	<p>Defined by parameter with a default value of</p> <p>0x1</p>	<p>Indicates whether Clock Switching is implemented:</p> <ul style="list-style-type: none"> 0: HW-based Counter Clock Switching is not implemented. 1: HW-based Counter Clock Switching is implemented.
15:4	-	RAZ/WI	0x000	Reserved
3:0	CNTSC	RO	0x1	<p>Indicates whether Counter Scaling is implemented:</p> <ul style="list-style-type: none"> 0000: Counter Scaling is not implemented. 0001: Counter Scaling is implemented. All other values are Reserved

A.3.9 CNTSCR0, CNTSCR1, Counter Scaling registers

The CNTSCR0 and CNTSCR1 registers have the same field definitions as the CNTSCR register. These two extra registers are used to preprogram the scaling values so that when hardware-based clock switching is implemented there is no need to program the scaling increment value each time when clock is switched.

When read by software, the value read is the value that software has written. In certain cases, for example when CNTCR.SCEN has disabled scaling, the actual increment value that the Counter uses is 1.0.

However, the value that is read from these registers does not reflect this. Therefore, the actual increment value that the Counter uses depends on the programming of these registers, the value of two HW configuration parameters, CNTSC and HWCLKSW, and the value of CNTCR.SCEN.

This implementation enables software to keep the scaling values in these registers unaffected when increment value changes due to Counter disabling.

These registers can only be written when the Counter is disabled (CNTCR.EN=0). The following table shows the bit assignments.

Table A-8: CNTSCR* register bit assignments

Bits	Name	Access	Reset value	Function
31:0	ScaleVal	RW	0x0100_0000	<p>When counter scaling is enabled, ScaleVal is the amount added to the Counter Count Value for every period of the counter as determined by 1/Frequency from the current operating frequency of the system counter, the <i>counter tick</i>. ScaleVal is expressed as an unsigned fixed-point number with an 8-bit integer value and a 24-bit fractional value.</p> <p>The CNTSCR register can only be changed when the counter is disabled, CNTCR.EN=0. If the value of CTNSCR changes when CNTCR.EN==1, then the Counter Count Value becomes UNKNOWN and remains UNKNOWN on future ticks of the clock.</p> <p>If CNTSC=0, CNTSCR* are permanently driven to 0x0100_0000.</p>



If HWCLKSW= 0, CNTSCR1 is permanently driven to 0x0000_0000.

A.4 System Timer overview

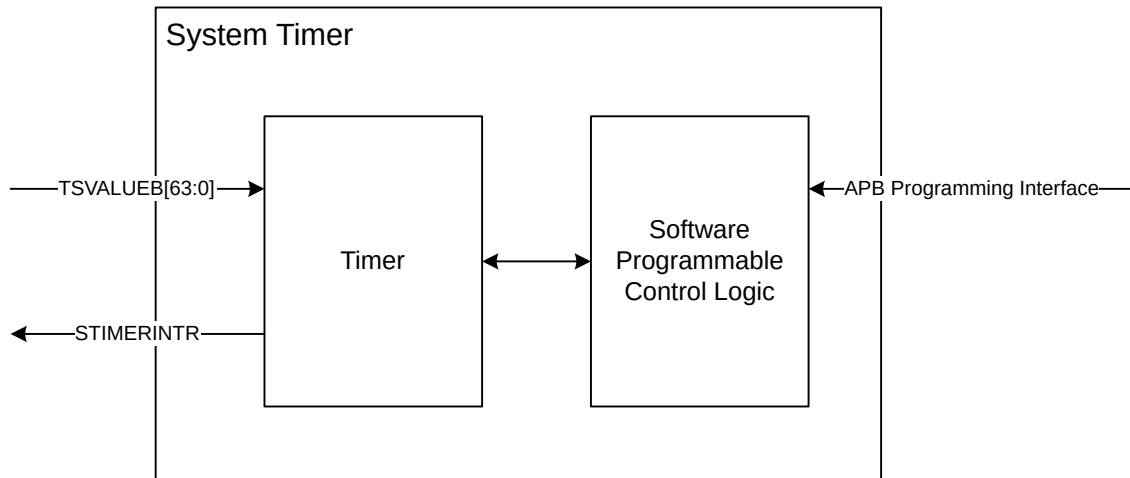
This section provides an overview of the System Timer.

The System Timer has the following features:

- Count up and count down timer functionality with auto-increment feature.
- Generation of a level triggered interrupt after a preconfigured period of time has passed.
- Time-based on shared system time count that the System Counter generates.

The following figure shows a block diagram of the System Timer.

Figure A-1: CRSAS Ma1 System Timer block diagram



A.4.1 System Timer operation

The primary function of the System Timer is to generate an interrupt output that is based on the Timer configuration and interrupt mask setting.

The timers can be programmed to count:

- Up to a threshold, by programming a CompareValue (CNTP_CVAL).
- Down from a programmed value, by programming a TimerValue (CNTP_TVAL).

The basic function of the System Timer is:

$$\text{TimerCondMet} = \text{Counter}[63:0] - \text{CompareValue}[63:0] \geq 0$$

An alternative 32-bit view, called TimerValue, can be used to set the CompareValue as follows:

$$\text{CompareValue} = (\text{Counter}[63:0] + \text{SignExtend}(\text{TimerValue})[63:0])$$

Reading the TimerValue gives the count down value:

$$\text{TimerValue} = (\text{CompareValue}[31:0] - \text{Counter}[31:0])$$

The auto-increment feature allows generation of Timer interrupt at regular intervals without the need for reprogramming the Timer after each interrupt and re-enabling the timer logic.

The Automatic Increment timer view operates as a 64-bit up-counter. An AutoIncrValue Reload register is programmed with a 32-bit offset. If the EN bit of the AutoIncrValue Control register is set, the offset value is zero-extended, summed with the count value, and loaded into the AutoIncrValue register, performed automatically by hardware.

The operation of auto-increment is as follows:

$$\text{AutoIncrValue} = (\text{ZeroExtend}(\text{Reload}[31:0]))[63:0] + \text{Counter}[63:0]$$

where:

`TimerCondMet TRUE` : If the timer condition is met.

`TimerCondMet FALSE` : Otherwise.

When the timer condition is met:

- An interrupt is generated, if the interrupt is not masked in the timer control register, and remains asserted until software clears it by writing to the CLR bit of the AutoIncrValue Control register, `CNTP_AIVAL_CTL`.
- The CLR bit in the AutoIncrValue Control register is set to 1 and it remains 1 until software clears it by writing '0'.
- The AutoIncrValue register is reloaded with the new value.

When the auto-increment feature is enabled, the operation of normal timer, using the compare value or timer value registers, is disabled. This is to ensure that when the interrupt is generated, the cause of interrupt is unambiguous to the user.

The auto-increment timer starts counting only when both the Timer is enabled (`CNTP_CTL.ENABLE=1`) and auto-increment mode is enabled by setting `CNTP_AIVAL_CTL.EN=1`. When both are enabled, the Timer starts counting down until the `AIVAL_RELOAD` period is reached.

A.5 System timer programmers model

The registers of the System Timer are grouped into a single 4KB block that is called the CNTBase frame. The base address of the CNTBase frame is not defined here and is **IMPLEMENTATION DEFINED**.

A.5.1 CNTBase Registers Summary

This section provides a summary of the CNTBase Registers.

The following table shows a summary of the CNTBase Registers.

Table A-9: CNTBase Frame Registers summary

Offset	Name	Access	Reset value	Description
0x000	CNTPCT[31:0]	RO	0xxxxx_xxxx	Physical Count Register. See CNTPCT, Counter-timer Physical Count register
0x004	CNTPCT[63:32]	RO	0xxxxx_xxxx	-
0x008-0x00C	Reserved	RAZ/WI	0x0000_0000	Reserved.

Offset	Name	Access	Reset value	Description
0x010	CNTFRQ	RW	0x0000_0000	Counter Frequency register. See CNTFRQ , Counter-timer Frequency register
0x014-0x01C	Reserved	RAZ/WI	0x0000_0000	Reserved.
0x020	CNTP_CVAL[31:0]	RW	0x0000_0000	Timer CompareValue register. See CNTP_CVAL , Counter-timer Physical Timer CompareValue register
0x024	CNTP_CVAL[63:32]	RW	0x0000_0000	-
0x028	CNTP_TVAL	RW	0xFFFF_FFFF	TimerValue register. See CNTP_TVAL , Counter-timer Physical Timer TimerValue register
0x02C	CNTP_CTL	RW	0x0000_000X	Timer Control register. See CNTP_CTL , Counter-timer Physical Timer Control register
0x030-0x03C	Reserved	RAZ/WI	0x0000_0000	Reserved.
0x040	CNTP_AIVAL[31:0]	RO	0xFFFF_FFFF	AutoIncrValue register. See CNTP_AIVAL , AutoIncrValue register
0x044	CNTP_AIVAL[63:32]	RO	0xFFFF_FFFF	-
0x048	CNTP_AIVAL_RELOAD	RW	0xFFFF_FFFF	AutoIncrValue Reload register. See CNTP_AIVAL_RELOAD , AutoIncrValue Reload register
0x04C	CNTP_AIVAL_CTL	RW	0xFFFF_FFFF	AutoIncrValue Control register. See CNTP_AIVAL_CTL , AutoIncrValue Control register
0x050	CNTP_CFG	RO	0x0000_0001	Timer Configuration register. See CNTP_CFG , Configuration register
0x054-0xFCC	Reserved	RAZ/WI	0x0000_0000	Reserved.
0xFD0	CNTP_PID4	RO	0x0000_0004	Peripheral ID4.
0xFD4-0xFDC	Reserved	RO	0x0000_0000	Reserved.
0xFE0	CNTP_PID0	RO	0x0000_00B7	Peripheral ID0.
0xFE4	CNTP_PID1	RO	0x0000_00B0	Peripheral ID1.
0xFE8	CNTP_PID2	RO	0x0000_000B	Peripheral ID2.
0xFEC	CNTP_PID3	RO	0x0000_0000	Peripheral ID3.
0xFF0	CNTP_CID0	RO	0x0000_000D	Component ID0.
0xFF4	CNTP_CID1	RO	0x0000_00F0	Component ID1.
0xFF8	CNTP_CID2	RO	0x0000_0005	Component ID2.
0xFFC	CNTP_CID3	RO	0x0000_00B1	Component ID3.

A.5.2 Register descriptions

This section describes each of the System Timer registers.

The ARMv8-A defines the CNTFRQ register to enable software to discover the frequency of Timer clock. This register is included for software-compatibility but is not used in this implementation where the Counter can have hardware-switchable clocks.

A.5.3 CNTPCT, Counter-timer Physical Count register

The CNTPCT register holds the 64-bit physical count value. The following table shows the bit assignments.

Table A-10: CNTPCT register bit assignments

Bits	Name	Access	Reset value	Function
63:0	CountValue	RO	0xFFFF_FFFF_FFFF_FFFF	Physical count value.

A.5.4 CNTFRQ, Counter-timer Frequency register

The CNTFRQ register is provided so that software can discover the frequency of the system counter. The instance of this register in the CNTCTLBase frame must be programmed with this value as part of system initialization. Hardware does not interpret the value of the register.

The following table shows the bit assignments.

Table A-11: CNTFRQ register bit assignments

Bits	Name	Access	Reset value	Function
31:0	ClockFrequency	RW	0xFFFF_FFFF_FFFF_FFFF	Clock frequency.

A.5.5 CNTP_CVAL, Counter-timer Physical Timer CompareValue register

The CNTP_CVAL register holds the 64-bit compare value for the timer. The following table shows the bit assignments.

Table A-12: CNTP_CVAL register bit assignments

Bits	Name	Access	Reset value	Function
63:0	CompareValue	RW	0x0000_0000_0000_0000	<p>Holds the 64-bit compare value for the timer.</p> <p>When CNTP_CTL.ENABLE is 1, the timer condition is met when (CNTPCT - CompareValue) is greater than zero. This means that CompareValue acts like a 64-bit counter timer. When the timer condition is met:</p> <ul style="list-style-type: none"> • NTP_CTL.ISTATUS is set to 1. • An interrupt is generated if CNTP_CTL.IMASK is 0. <p>When CNTP_CTL.ENABLE is 0, the timer condition is not met, but CNTPCT continues to count.</p>

A.5.6 CNTP_TVAL, Counter-timer Physical Timer TimerValue register

The CNTP_TVAL register holds the timer value for the timer. The following table shows the bit assignments.

Table A-13: CNTP_TVAL register bit assignments

Bits	Name	Access	Reset value	Function
31:0	TimerValue	RW	0xxxxx_xxxx	<p>The TimerValue view of the physical timer. On a read of this register:</p> <ul style="list-style-type: none"> If CNTP_CTL.ENABLE is 0, the value that is returned is UNKNOWN. If CNTP_CTL.ENABLE is 1, the value that is returned is (CNTP_CVAL - CNTPCT). <p>On a write of this register, CNTP_CVAL is set to (CNTPCT + TimerValue), where TimerValue is treated as a signed 32-bit integer.</p> <p>When CNTP_CTL.ENABLE is 1, the timer condition is met when (CNTPCT - CNTP_CVAL) is greater than zero. This means that TimerValue acts like a 32-bit downcounter timer. When the timer condition is met:</p> <ul style="list-style-type: none"> CNTP_CTL.ISTATUS is set to 1. If CNTP_CTL.IMASK is 0, an interrupt is generated. <p>When CNTP_CTL.ENABLE is 0, the timer condition is not met, but CNTPCT continues to count, so the TimerValue view appears to continue to count down.</p>

A.5.7 CNTP_CTL, Counter-timer Physical Timer Control register

The CNTP_CTL register is a control register for the timer. The following table shows the bit assignments.

Table A-14: CNTP_CTL register bit assignments

Bits	Name	Access	Reset value	Function
31:3	-	RAZ/ WI	-	Reserved.
2	ISTATUS	RO	0xX	<p>The status of the timer. This bit indicates whether the timer condition is met:</p> <ul style="list-style-type: none"> 0: Timer condition is not met. 1: Timer condition is met. <p>When the value of the ENABLE bit is 1, ISTATUS indicates whether the timer condition is met.</p> <p>ISTATUS takes no account of the value of the IMASK bit. If the value of ISTATUS is 1 and the value of IMASK is 0, then the timer interrupt is asserted.</p> <p>When the value of the ENABLE bit is 0, the ISTATUS field is UNKNOWN.</p>

Bits	Name	Access	Reset value	Function
1	IMASK	RW	0xX	<p>Timer interrupt mask bit. Permitted values are:</p> <ul style="list-style-type: none"> 0: The IMASK bit masks the Timer interrupt. 1: The IMASK bit masks the Timer interrupt. <p>For more information, see the description of the ISTATUS bit.</p>
0	ENABLE	RW	0x0	<p>Enables the timer. Permitted values are:</p> <ul style="list-style-type: none"> 0: Timer is disabled. 1: Timer is enabled. <p>Setting this bit to 0 disables the timer output signal, but the timer value accessible from CNTP_TVAL continues to count down.</p> <p>Disabling the output signal might be a power-saving option.</p>

A.5.8 CNTP_AIVAL, AutoIncrValue register

The CNTP_AIVAL register holds the 64-bit Automatic Increment value for the timer. The following table shows the bit assignments.

Table A-15: CNTP_AIVAL register bit assignments

Bits	Name	Access	Width	Reset value	Description
63:0	AutoIncrValue timer value	RO	64	0x0000_0000_0000_0000	Timer AutoIncrValue.

A.5.9 CNTP_AIVAL_RELOAD, AutoIncrValue Reload register

Holds the programmable offset value for the Automatic Increment timer view. The following table shows the bit assignments.

Table A-16: CNTP_AIVAL_RELOAD register bit assignments

Bits	Name	Access	Width	Reset value	Description
31:0	AutoIncrValue timer value	RO	32	0x0000_0000_0000_0000	Timer AutoIncrValue Reload.

A.5.10 CNTP_AIVAL_CTL, AutoIncrValue Control register

Control register for the Automatic Increment timer view. The following table shows the bit assignments.

Table A-17: CNTP_AIVAL_CTL register bit assignments

Bits	Name	Access	Width	Reset value	Description
31:2	-	-	30	-	Reserved.

Bits	Name	Access	Width	Reset value	Description
1	CLR	WO	1	0xX	<p>Timer interrupt clear bit. This bit is only used to clear the interrupt that is generated because of auto-increment mode. For clearing the interrupt generated by normal mode, the Timer should be disabled.</p> <p>Permitted values are:</p> <ul style="list-style-type: none"> 0: Timer interrupt is clear. 1: Timer interrupt is not clear. <p>The CLR bit can only be written to as zero. Writes as one are ignored.</p>
0	EN	WO	1	0xX	<p>Enables the AutoIncrValue register. Permitted values are:</p> <ul style="list-style-type: none"> 0: AutoIncrValue disabled. 1: AutoIncrValue enabled.

A.5.11 CNTP_CFG, Configuration register

Provides timer configuration information.

The following table shows the bit assignments.

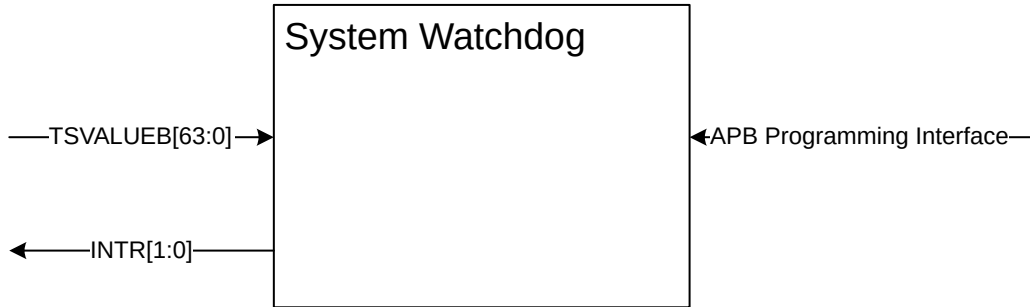
Table A-18: CNTP_CFG register bit assignments

Bits	Name	Access	Width	Reset value	Description
31:4	-	-	28	-	Reserved.
3:0	AIVAL	RO	4	0x1	<p>Indicates whether Automatic Increment is implemented. Permitted values are:</p> <ul style="list-style-type: none"> 0000: Automatic Increment is not implemented. 0001: Automatic Increment is implemented. All other values are Reserved.

A.6 System Watchdog overview

The following figure shows a block diagram of the System Watchdog.

Figure A-2: Block diagram of the System Watchdog



A.6.1 Watchdog operation

The basic function of the Generic Watchdog is to count for a fixed period of time, during which it expects to be refreshed by the system, indicating normal operation.

If a refresh occurs within the watch period, the period is refreshed to the start.

If the refresh does not occur, then the watch period expires, and a signal INTR[0] is raised, and a second watch period begins.

Second watch period behavior

The initial signal is typically wired to an interrupt and alerts the system. The system can attempt to take corrective action that includes refreshing the watchdog within the second watch period.

- If the refresh is successful, the system returns to the previous normal operation.
- If it fails, then the second watch period expires, and a second signal INTR[1] is generated. The signal is fed to higher privileged software as an interrupt or reset for it to take executive action.

The Watchdog uses the input time TSVALUEB generated by the System Counter as the timebase against which the decision to trigger an interrupt is made.

A.6.2 System watchdog programmers model

This section describes the programmers model of the System Watchdog. The Watchdog includes the following two 4KB register frames:

- Control Frame.

- Refresh Frame.

Configurable parameters control the base addresses of Control and Refresh frames. These parameters provide the flexibility of arranging these two frames within an 8KB memory map that the following table shows.

The following table shows the Base addresses of Control and Refresh frames.

Table A-19: Base addresses of Control and Refresh frames

Description	Start address	End address	Size
Control Frame	0x0_0000	0x0_0FFF	4KB
Refresh Frame	0x0_1000	0x0_1FFF	4KB

A.6.3 Control Frame registers summary

This section provides a summary of the Control Frame registers. The following table shows the Control Frame registers summary.

Table A-20: Control Frame registers summary

Offset	Name	Access	Reset value	Description
0x000	WCS	RW	0x0000_0000	WCS, Watchdog Control and Status register
0x004	Reserved	RAZ/WI	0x0000_0000	Reserved
0x008	WOR	RW	0x0000_0000	WOR, Watchdog Offset register
0x00c	-	RAZ/WI	0x0000_0000	Reserved
0x010	WCV[31:0]	RW	0x0000_0000	WCV, Watchdog Compare Value register
0x014	WCV[63:32]	RW	0x0000_0000	-
0x018-0xFC8	-	RAZ/WI	0x0000_0000	Reserved
0xFCC	W_IIDR	RO	0x0000_143B	W_IIDR, Watchdog Interface Identification register
0xFD0	PIDR4	Read-only	0x0000_0004	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFE0	PIDR0	Read-only	0x0000_00B1	Peripheral ID 0
0xFE4	PIDR1	Read-only	0x0000_00B0	Peripheral ID 1
0xFE8	PIDR2	Read-only	0x0000_002B	Peripheral ID 2
0xFEC	PIDR3	Read-only	0x0000_0000	Peripheral ID 3
0xFF0	CIDR0	Read-only	0x0000_000D	Component ID 0
0xFF4	CIDR1	Read-only	0x0000_00F0	Component ID 1
0xFF8	CIDR2	Read-only	0x0000_0005	Component ID 2
0xFFC	CIDR3	Read-only	0x0000_00B1	Component ID 3

A.6.4 Refresh Frame registers Summary

This section provides a summary of the Refresh Frame registers.

The following table shows a summary of the Refresh Frame registers.

Table A-21: Refresh Frame registers summary

Offset	Name	Access	Reset value	Description
0x000	WRR	RW	0x0000_0000	WRR, Watchdog Refresh register
0x004-0xFC8	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFCC	W_IIDR	RO	0x0000_143B	W_IIDR, Watchdog Interface Identification register
0xFD0	PIDR4	Read-only	0x0000_0004	Peripheral ID 4
0xFD4 – 0xFDC	Reserved	RAZ/WI	0x0000_0000	Reserved
0xFE0	PIDR0	Read-only	0x0000_00B0	Peripheral ID 0
0xFE4	PIDR1	Read-only	0x0000_00B0	Peripheral ID 1
0xFE8	PIDR2	Read-only	0x0000_002B	Peripheral ID 2
0xFEC	PIDR3	Read-only	0x0000_0000	Peripheral ID 3
0xFF0	CIDR0	Read-only	0x0000_000D	Component ID 0
0xFF4	CIDR1	Read-only	0x0000_00F0	Component ID 1
0xFF8	CIDR2	Read-only	0x0000_0005	Component ID 2
0xFFC	CIDR3	Read-only	0x0000_00B1	Component ID 3

A.6.5 Register descriptions

This section describes each System Watchdog register.

All registers are 32-bit and must be accessed using 32-bit reads and writes.

A.6.6 WCS, Watchdog Control and Status register

The WCS register is a control and status register for the watchdog. Any write to this register causes an explicit watchdog refresh.

The following table shows the bit assignments.

Table A-22: WCS register bit assignments

Bits	Name	Access	Reset value	Function
31:3	-	RAZ/WI	-	Reserved
2:1	Watchdog Signal Status	RO	0x0	Indicates the current state of the watchdog signals: <ul style="list-style-type: none"> Bit 0 WSO Interrupt INTR[0]. Bit 1 WS1 Interrupt INTR[1].

Bits	Name	Access	Reset value	Function
0	Watchdog Enable	RW	0x0	Watchdog enable: <ul style="list-style-type: none"> 0: A write of 0 disables the Watchdog. 1: A write of 1 to this bit enables the Watchdog. A read of these bits indicates the current state of the Watchdog enable.

A.6.7 WOR, Watchdog Offset register

The WOR register is a countdown timer value for the watchdog. Any write to this register causes an explicit watchdog refresh.

The following table shows the bit assignments.

Table A-23: WOR register bit assignments

Bits	Name	Access	Reset value	Function
31:0	WOR	RW	0x0000_0000	Holds the 32-bit countdown timer value.

A.6.8 WCV, Watchdog Compare Value register

The WCV register holds the compare value of the watchdog. The following table shows the bit assignments.

Table A-24: WCV register bit assignments

Bits	Name	Access	Reset value	Function
63:0	WCV	RW	0x0000_0000	HHolds the current 64-bit compare value.

A.6.9 W_IIDR, Watchdog Interface Identification register

The W_IIDR register is an identification register for the watchdog. The following table shows the bit assignments.

Table A-25: W_IIDR register bit assignments

Bits	Name	Access	Reset value	Function
31:24	ID	RO	0x0	Product identifier. 0x00 = System Watchdog.
23:20	-	RO	0x0	Reserved
19:16	ARCH	RO	0x0	Architecture version, v0.
15:12	REV	RO	0x1	Revision number for the component. 0x1 = r0p1
11:0	JEPCODE	RO	0x43B	Arm JEP106 code.

A.6.10 WRR, Watchdog Refresh register

The WRR register is a refresh register for the Watchdog. Any write to this register causes an explicit watchdog refresh.

The following table shows the bit assignments.

Table A-26: WRR register bit assignments

Bits	Name	Access	Function
31:0	WRR	RW	A write to this location causes the Watchdog to refresh and start a new watch period. A read has no effect and returns 0.

Appendix B Revisions

This appendix describes the technical changes between released issues of this document.

Table B-1: Issue 0000-01

Change	Location
Initial issue of the document	-