Arm® CoreLink™ SSE-200 Subsystem for Embedded

Revision: r1p0

Technical Reference Manual

arm

101104_0100_00_en

Arm® CoreLink™ SSE-200 Subsystem for Embedded

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
A	20 December 2016	Confidential	First release for r0p0 Beta (ARM DDI0574).
0100-00	26 September 2017	Non-Confidential	First release for r1p0 EAC (ARM 101104).

Non-Confidential Proprietary Notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

Product Status

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

Web Address

http://www.arm.com

Contents
Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual

Preface

This preface introduces the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual*.

It contains the following:

About this book

This book is for the Arm® CoreLink™SSE-200 Subsystem for Embedded (SSE-200). It provides a high-level overview of the SSE-200. It describes architectural information, and as such, facilitates the creation of software or a SoC targeted at an Internet of Things (IoT) application.

Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this book, for example, r*1*p*2*, where:

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

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

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 SSE-200.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction
This chapter introduces the SSE-200.

Chapter 2 Functional description
This chapter describes the functionality of the SSE-200.

Chapter 3 Programmers Model
This chapter describes the SSE-200 memory regions and registers, and provides information on how to program a SoC that contains an implementation of the SSE-200.

Appendix A Signal Descriptions
This appendix summarizes the interface signals present in the SSE-200 elements.

Appendix B Revisions
This appendix describes the technical changes between released issues of this book.

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.

Typographic conventions

italic
Introduces special terminology, denotes cross-references, and citations.

bold
Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

`monospace`
Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`mono`space
Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

`monospace italic`
Denotes arguments to monospace text where the argument is to be replaced by a specific value.

`monospace bold`
: Denotes language keywords when used outside example code.

`<and>`
: Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS
: Used in body text for a few terms that have specific technical meanings, that are defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Timing diagrams

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

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

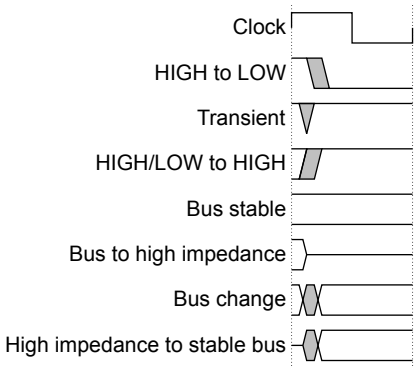

Figure 1 Key to timing diagram conventions

Signals

The signal conventions are:

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

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

Additional reading

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

Arm publications

- *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual* (Arm 101104)
- *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* (Arm DDI 0571).
- *Arm® Cortex®-M System Design Kit Technical Reference Manual* (Arm DDI 0479).
- *Arm® Cortex®-M33 Processor Technical Reference Manual* (Arm 100230).
- *Arm® Cortex®-M33 Processor User Guide Reference Material* (Arm 100234).

The following confidential books are only available to licensees or require registration with ARM:

- *Arm® CoreLink™ SSE-200 Subsystem for Embedded Configuration and Integration Manual* (Arm 100224).
- *Arm® CoreLink™ SSE-200 Subsystem for Embedded Release Note* (Arm EPM-135617).
- *Arm®v7-M Architecture Reference Manual* (Arm DDI 0403).
- *Arm®v8-M Architecture Reference Manual* (Arm DDI 0553).
- *Arm® Cortex®-M33 Processor Integration and Implementation Manual* (Arm 100323).
- *Arm® CoreSight™ Components Technical Reference Manual* (Arm DDI 0314).
- *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* (Arm IHI 0068).
- *Arm® Power Policy Unit Architecture Specification, version 1.1* (Arm DEN 0051).
- *Arm® Debug Interface Architecture Specification ADIv5.0 to ADIv5.2* (Arm IHI 0031).
- *Arm® Embedded Trace Macrocell (ETMv4) Architecture Specification* (Arm IHI 0064).
- *Arm® AMBA® 5 AHB Protocol Specification* (Arm IHI 0033).
- *Arm® AMBA® APB Protocol Specification Version 2.0* (Arm IHI 0024).
- *Arm® TrustZone® CryptoCell-312 Technical Reference Manual* (ARM 100774).

Other publications

None.

Feedback

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to *errata@arm.com*. Give:

- The title *Arm CoreLink SSE-200 Subsystem for Embedded Technical Reference Manual*.
- The number 101104_0100_00_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note

Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

Chapter 1
Introduction

This chapter introduces the SSE-200.

It contains the following sections:

1.1 About the SSE-200

The Arm CoreLink SSE-200 Subsystem for Embedded is a collection of pre-assembled elements to use as the basis of an *Internet of Things* (IoT) *System on Chip* (SoC).

It is complemented by software libraries that are integrated with the Mbed™ operating system. The SSE-200 is part of the CoreLink SDK-200 System Design Kit (SDK-200) which also contains the CoreLink SIE-200 System IP for Embedded product and other components. The SDK-200 provides components to quickly create systems that are based on Cortex-M processors.

The following figure shows the major blocks present in the SSE-200.

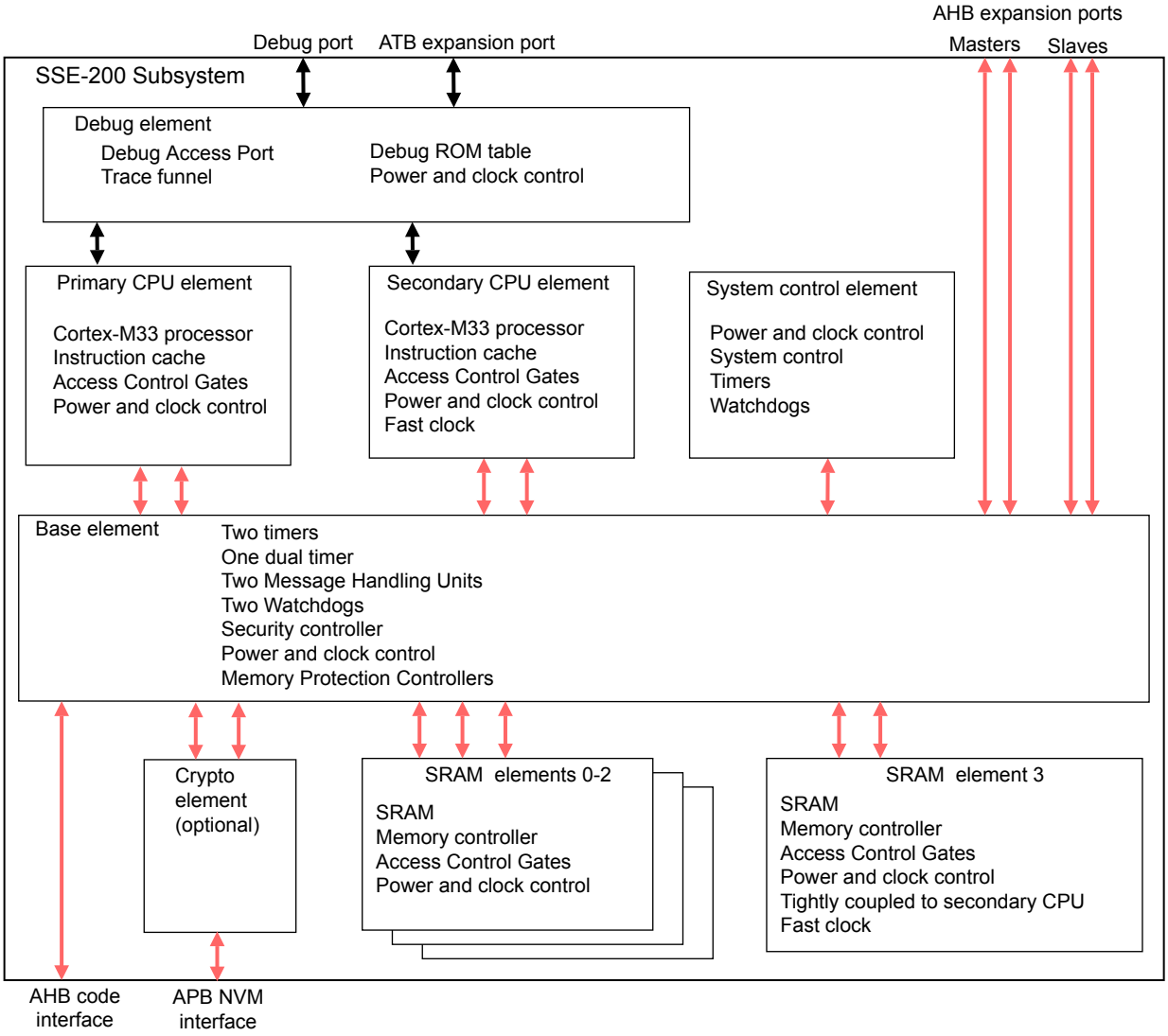

Figure 1-1 SSE-200 subsystem elements

1.1.1 About IoT System on Chip implementations

The SSE-200 subsystem must be extended to create an IoT SoC. A complete system typically contains the following components:

Compute subsystem
The compute subsystem consists of two Cortex-M33 processors and associated bus, debug, controller, peripherals, and interface logic supplied by Arm.

Reference system memory and peripherals
SRAM is part of the SSE-200, but a SoC requires extra memory, control, and peripheral components beyond the minimum subsystem components. Flash memory, for example, is not provided with the SSE-200.

Communication interface
The endpoint must have some way of communicating with other nodes or masters in the system. This interface could be WiFi, Bluetooth, or a wired connection.

Sensor or control component
To be useful as an endpoint, the reference design is typically extended by adding sensors or control logic such as temperature input or motor control output.

Software development environment
Arm provides a complete software development environment which includes the Mbed operating system, Arm or GNU (GCC) compilers and debuggers, and firmware.

Custom peripherals typically require corresponding third-party firmware that can be integrated into the software stack.

IoT hardware and software

The following figure shows a block diagram of the hardware and software in an IoT system.

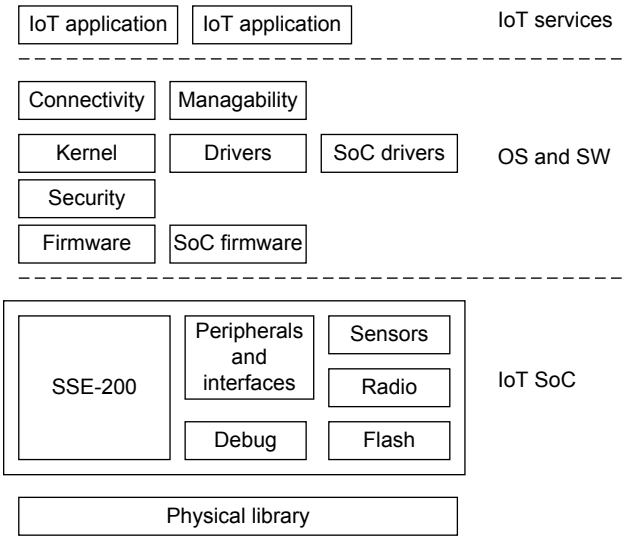

Figure 1-2 Hardware and software solution

1.2 Features of the SSE-200

The SSE-200 contains the following components:

- Two Cortex-M33 processors:
  - Optional *Floating-Point Unit* (FPU) and *Digital Signal Processor* (DSP) extensions (configurable).
  - *Embedded Trace Macrocell* (ETM).

  For more information, see the *Arm® Cortex®-M33 Processor Technical Reference Manual*.
- CoreSight debug system with configurable Secure Debug and Trace.
- Secure AMBA interconnect:
  - *Advanced High Performance Bus* (AHB5) Bus Matrix.
  - AHB5 TrustZone *Memory Protection Controller* (MPC).
  - AHB5 TrustZone *Peripheral Protection Controller* (PPC).
  - AHB5 *Exclusive Access Monitor* (EAM).
  - AHB5 *Access Control Gates* (ACG).
  - AHB5 to *Advanced Peripheral Bus* (APB) Bridges.
  - Expansion AHB5 master and slave buses (two each).
- Memory system:
  - AHB5 master bus to external code memory.
  - Static memory controllers.
  - Multiple banks of SRAM.

    One bank of SRAM functions as *Tightly Coupled Memory* (TCM).
  - Instruction caches.
- Security components:
  - TrustZone CryptoCell-312 (optional).
  - *Implementation Defined Attribution Unit* (IDAU).
  - Secure expansion ports.
  - System Security Controller.
  - System Controller.
- APB peripherals with security support:
  - Three general-purpose timers with configurable security. One timer is on the 32KHz domain and two are on the SYSCLK PD_SYS domain.
  - A *Cortex-M System Design Kit* (CMSDK) dual timer with configurable security.
  - Three Watchdog timers with fixed security. One Secure watchdog is on the 32KHz domain and one Secure and one Non-Secure is on the SYSCLK PD_SYS domain.
  - Two *Message Handling Units* (MHUs) allow software to raise interrupts.
- Power-control components:
  - *Power Dependency Control Matrix* (PDCM).
  - *Power Policy Units* (PPU).
  - CoreLink LPD-500 Low Power Distributor.
  - Wakeup on interrupt from *External Wakeup Controllers* (EWC) and *Wakeup Interrupt Controllers* (WIC).

1.3 CoreLink SIE-200 components

The SSE-200 uses the following components from the CoreLink SIE-200 System IP for Embedded product:

- AHB5 to AHB5 Sync-down Bridge.
- AHB5 to AHB5 Sync-up Bridge.
- AHB5 Slave Multiplexer.
- AHB5 Master Multiplexer.
- AHB5 Default Slave.
- AHB5 Bus Matrix.
- AHB5 to APB4 Synchronous Bridge.
- AHB5 to APB4 Asynchronous Bridge.
- AHB5 Access Control Gate.
- AHB5 Exclusive Access Monitor.
- AHB5 to SRAM Interface.
- AHB5 TrustZone Memory Protection Controller.
- APB4 TrustZone Peripheral Protection Controller.
- Power Dependency Control Matrix.

The following CoreLink SIE-200 components can be added to the SSE-200 by the system integrator:

- AHB5 GPIO.
- AHB5 Example Slave.
- AHB5 to External SRAM Interface.
- AHB5 to Flash Interface Modules.
- AHB5 to AHB5 and APB4 Asynchronous Bridge.
- AHB5 Downsizer.
- AHB5 Upsizer.

1.4 Compliance

The SSE-200 complies with, or includes components that comply with, the following specifications:

- *1.4.1 Arm® Architecture* on page 1-16.
- *1.4.2 Debug* on page 1-16.
- *1.4.3 Interrupt controller architecture* on page 1-16.
- *1.4.4 Advanced Microcontroller Bus Architecture* on page 1-16.

This Technical Reference Manual complements the TRMs for included components, architecture reference manuals, architecture specifications, protocol specifications, and relevant external standards. It does not duplicate information from these sources.

1.4.1 Arm® Architecture

The Cortex-M33 processor in the SSE-200 implements the Armv8-M architecture which executes the Arm-v8M T32 instruction set.

See the *Arm®v8-M Architecture Reference Manual* for more information.

Security

The Arm TrustZone technology in the Armv8-M architecture enables memory and peripheral spaces to be partitioned into Secure and Non-secure regions. No access to Secure assets is possible from the Non-secure world.

1.4.2 Debug

The SSE-200 implements the Arm CoreSight debug interface.

See the *Arm® CoreSight™ Components Technical Reference Manual* for more information.

1.4.3 Interrupt controller architecture

The SSE-200 implements the following features:

- Arm *Nested Vectored Interrupt Controller* (NVIC).
- Arm *Wakeup Interrupt Controller* (WIC).

See the *Arm® Cortex®-M33 Processor Technical Reference Manual* for more information on the NVIC.
See the *Arm® Cortex®-M33 Processor User Guide Reference Material* for more information on the WIC.

1.4.4 Advanced Microcontroller Bus Architecture

The SSE-200 complies with the:

- *Advanced High Performance Bus* (AHB5) protocol.

  See the *Arm® AMBA® 5 AHB Protocol Specification*.
- *Advanced Peripheral Bus* (APB4) protocol.

  See the *Arm® AMBA® APB Protocol Specification Version 2.0*.

The SSE-200 contains components that use Arm TrustZone technology that supports the ARMv8-M Security Extension for Secure and Non-secure states.

Related references

1.4.1 Arm® Architecture on page 1-16.
1.4.2 Debug on page 1-16.
1.4.3 Interrupt controller architecture on page 1-16.
1.4.4 Advanced Microcontroller Bus Architecture on page 1-16.

1.5 Product revisions

This section describes the differences in functionality between product revisions:

r0p0

First release.

r1p0

Technical updates.

Chapter 2
Functional description

This chapter describes the functionality of the SSE-200.

It contains the following sections:

2.1 Top-level system partitioning

This section describes the top-level partitioning of the SSE-200 subsystem.

This section contains the following subsections:

2.1.1 Overview

The SSE-200 components are organized into the following blocks or elements:

- Base element.
- CPU elements.
- Debug element.
- System control element.
- SRAM elements.
- Crypto element.

Note

The provided system components only form part of the finished SoC. Arm expects the system designer to extend and customize the subsystem for their application requirements.

The top-level view of the subsystem elements and the AHB5 and APB bus interconnections is shown in the following figure. The following abbreviations are used in the figure:

ACG
: AHB5/APB Access Control Gate.

EAM
: AHB5 Exclusive Access Monitor.

MHU
: Message Handling Unit.

MPC
: AHB5 Memory Protection Controller.

MSC
: Master Security Controller.

PCSM
: Power Control State Machine.

PIK
: Power Integration Kit.

PPC
: AHB5/APB Peripheral Protection Controller.

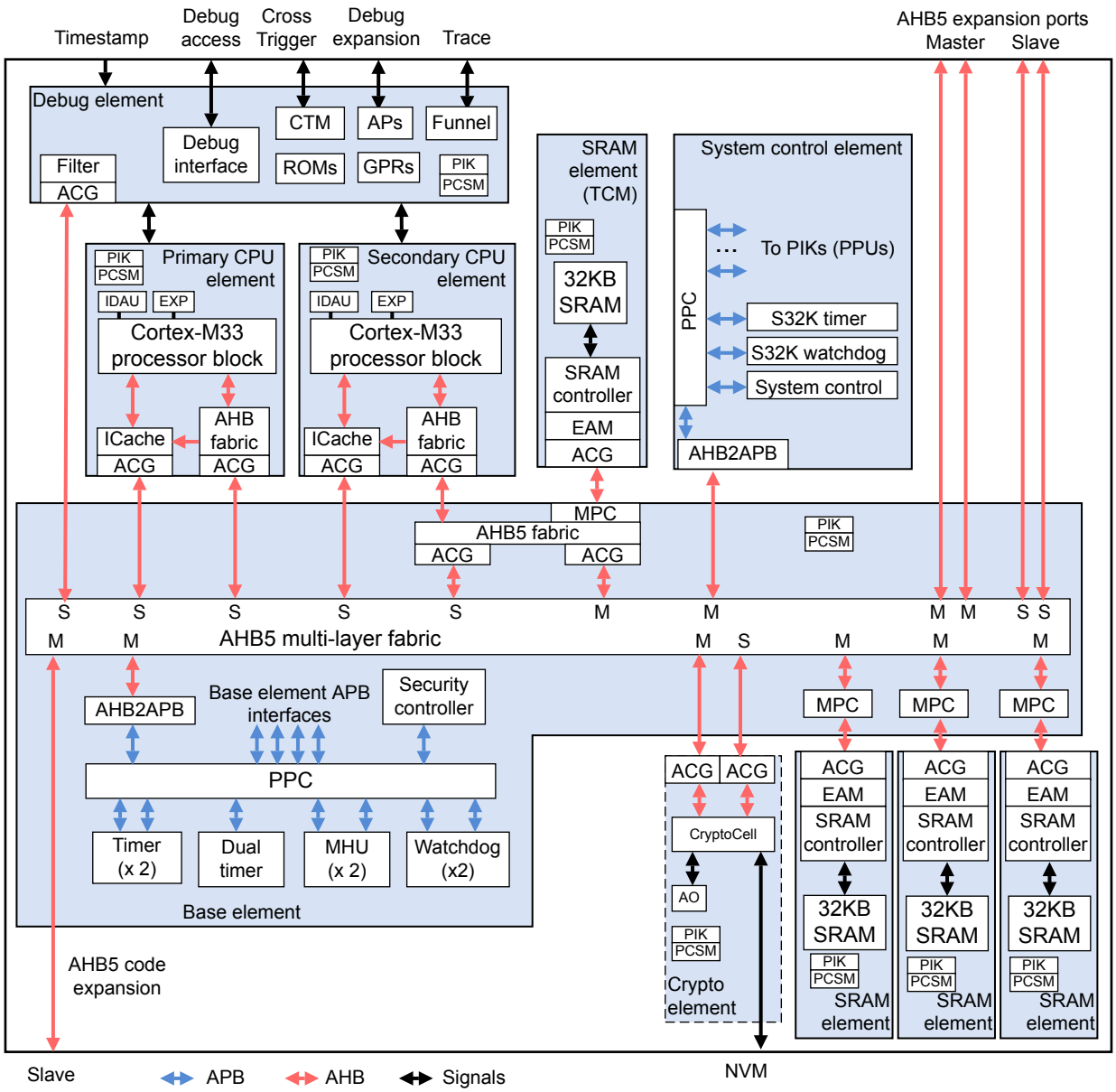

Figure 2-1 Top-level element interconnections

2.1.2 Configuration options

Configuration parameters set some of the processor and system options, for example:

- Reset value for the vector table offset addresses for both processors.
- CPU IDs.
- Number of expansion interrupts for the processors and the wakeup controllers.
- Interrupt latencies.
- Presence of FPUs and support for DSP extension instructions.
- Debug resources.
- Clock divider values.

Note

- See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Configuration and Integration Manual* for more configuration details.
- This manual is only available to customers who have licensed the SSE-200 product.

Related references

A.1 Clock, reset, and power control signals on page Appx-A-146.
A.2 Interrupt signals on page Appx-A-152.
A.3 AHB expansion bus signals on page Appx-A-154.
A.4 Debug and Trace signals on page Appx-A-157.
A.5 Security component interfaces on page Appx-A-161.
A.6 Miscellaneous top-level signals on page Appx-A-165.
A.7 CryptoCell-312 signals on page Appx-A-168.

2.1.3 Interface signals

The SSE-200 has the following interfaces at the boundary of subsystem to allow customers to customize the subsystem:

- Clock and reset.
- Processor-related signals:
  - Processor control.
  - Interrupts.
  - Configuration signals.
- Base element:
  - AHB expansion.
- System control:
  - Static configuration signals.
  - Power control expansion interfaces.
  - External Wakeup Controller interrupt inputs.
- Debug and Trace:
  - Debug access.
  - Timestamp.
  - Cross Trigger Channel.
  - Debug APB expansion.
  - ATB Trace.
  - Debug authentication.
- Crypto: This element integrates the CryptoCell-312 into the system to provide cryptographic acceleration. This element is optional, and it includes:
  - NVM APB interface.
  - *Debug Control Unit* (DCU) signals.
  - *Life Cycle State* (LCS) signals.
- Security.
- Top-level static configuration signals.

Related references

A.1 Clock, reset, and power control signals on page Appx-A-146.
A.2 Interrupt signals on page Appx-A-152.
A.3 AHB expansion bus signals on page Appx-A-154.
A.4 Debug and Trace signals on page Appx-A-157.
A.5 Security component interfaces on page Appx-A-161.
A.6 Miscellaneous top-level signals on page Appx-A-165.
A.7 CryptoCell-312 signals on page Appx-A-168.

2.2 Clocks

This section describes the clocks in the SSE-200 subsystem.

This section contains the following subsections:

2.2.1 Overview

The following clocks are present in the SSE-200:

- The subsystem uses the clock **MAINCLK** input to derive:
  - **SYSCLK** which is the clock for the primary processor, bus matrix, most of the SRAMs, and peripherals.
  - **FCLK** is used by the debug system, secondary processor, and the last SRAM.

    SRAM element 3 is also used for *Tightly Coupled Memory* (TCM) to provide higher performance for data accesses.
- A slow clock is used by some modules within the system control element.

  The **S32KCLK** clock is an asynchronous clock input that is used primarily to drive the S32KWATCHDOG. **S32KCLK** also drives the S32KTIMER, and other logic that must be clocked at the lowest system power mode (hibernation).

All derived clocks are synchronous to the main input clock and a range of primary to secondary ratios are supported.

The generated clocks are locally clock gated within each element depending on which reset domain it is on, and the activity and power state of the dependent logic. The power state of dependent logic is controlled by the related *Power Policy Unit* (PPU). Some of the gated clocks are output for expansion logic in the same clock, reset, and power domains.

The following diagram shows the interconnections for the clock signals:

In the diagram:

- The CC blocks are clock controllers for hierarchical clock gating.
- The CG blocks are clock gates.
- The PPU blocks are Power Policy Units.

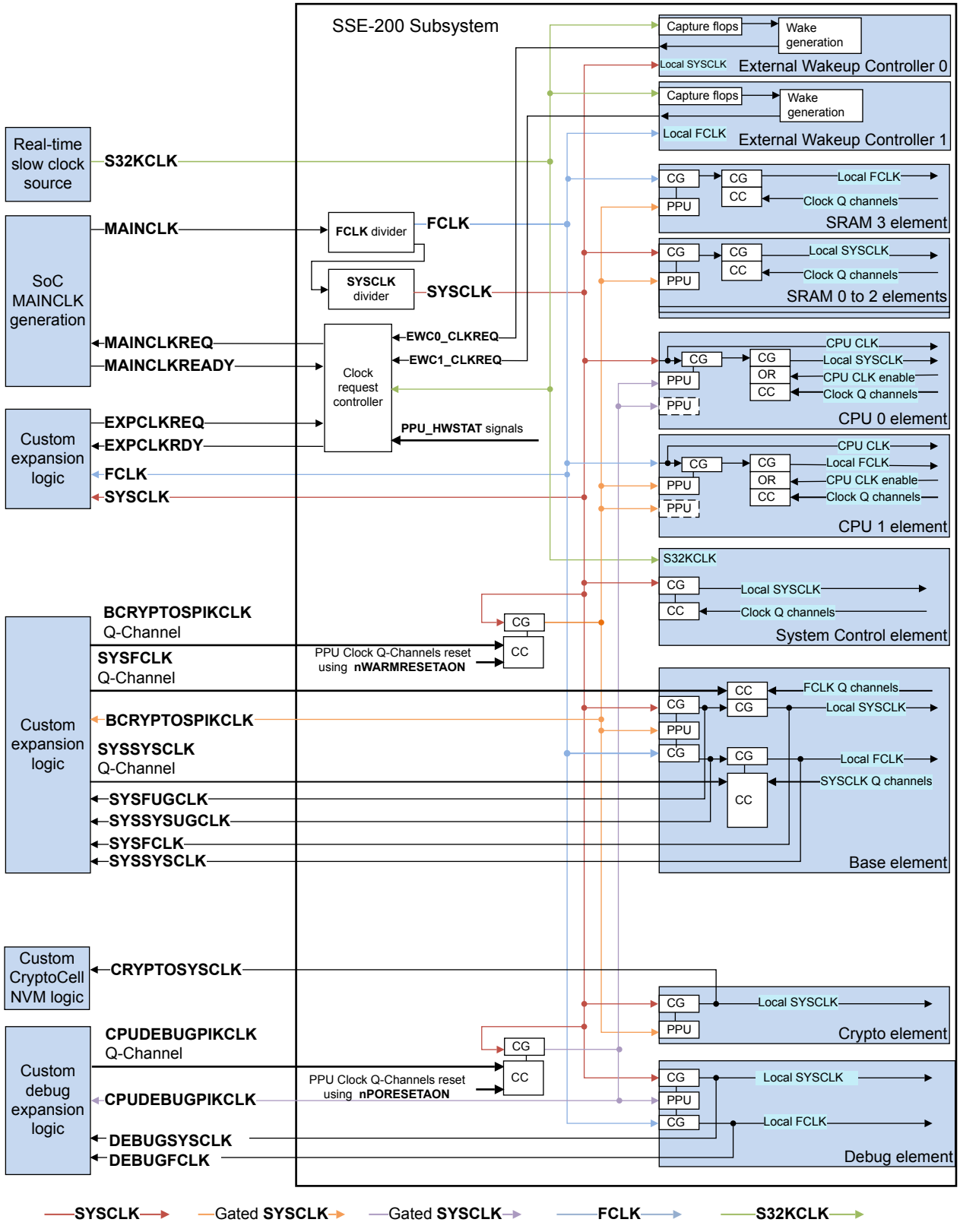

Figure 2-2 Clock interconnection

The Clock Request Controller handshakes with the external clock generator to turn off the main clock when it is not required.

System control registers control the clock divider.

Q-Channel based clock control logic provides the following features:

- Access Control Gates can isolate clock domains and provide Q-Channel handshake interfaces.
- A clock controller in each domain connects to the domains and gates the clock when the domain is idle.

2.2.2 Clock generation and control

This section describes clock generation and control for **MAINCLK**, **FCLK**, and **SYSCLK** and clocks derived from them.

MAINCLK

Most clocks within the subsystem are derived from **MAINCLK**, the subsystem main input clock. The following handshake signals control the clock state:

- **MAINCLKREQ** is the system request for **MAINCLK** to become active.
- **MAINCLKREADY** is the notification that **MAINCLK** is ready to be used.

MAINCLK is required in the following conditions:

- Any PPU in the system is in the ON state.
- Any External Wakeup Controller (EWC) is requesting that a core wakes up.
- Expansion logic uses the **EXP_CLK_REQ** and **EXP_CLK_ACK** handshake logic to request that ungated clocks are available.

Therefore, **MAINCLK** only turns off when the system goes into the Hibernate state and no other parts of the system, including from external expansion logic, requests the clocks to be available.

FCLK and SYSCLK

FCLK and **SYSCLK** are generated from **MAINCLK**:

- **FCLK** is generated by dividing down **MAINCLK**.
- **SYSCLK** is then generated by dividing down **FCLK**.
- The divider ratio can be set from configuration parameters and also by software.

Note

When generating **SYSCLK**, ensure that the **MAINCLK** frequency, and the selected clock divider ratios for **FCLK** and **SYSCLK**, result in a **SYSCLK** frequency of more than double the **S32KCLK** frequency.

The subsystem uses the **EXPCLKREQ** and **EXPCLKREADY** handshake signals to provide **SYSCLK** and **FCLK** on request from external hardware. The handshake follows the same protocol as the **MAINCLKREQ** and **MAINCLKREADY** handshake signals. When using the **EXPCLKREQ** and **EXPCLKREADY** handshake signals to request active clocks, the SSE-200 also requests that the **MAINCLK** clock is active.

Note

All signals described previously connect to and from the always ON (AON) part of the system.

Clock and reset outputs are provided for expansion of other power domains within the subsystem as follows:

- For the PD_SYS power domain, **SYSFCLK** and **SYSSYSCLK** are power-gated and hierarchical clock-gated versions of **FCLK** and **SYSCLK**. These are used for expansion logic that resides in the

PD_SYS power domain. In additional **SYSSYSUGCLK** and **SYSFUGCLK** are the equivalent power-gated but not hierarchical clock-gated versions of **SYSCLK** and **FCLK**.

- For the PD_DEBUG power domain, **DEBUGFCLK** and **DEBUGSYSCLK** are hierarchical clock gated versions of **FCLK** and **SYSCLK**. These are used for debug expansion logic that resides in the PD_DEBUG power domain.
- The **HINTSYSCLKENCLK** and **DEBUGHINTSYSCLKENCLK** signals are clock pulses that are generated by the divider to indicate when, relative to the divided clock **FCLK** or **DEBUGFCLK**, an enable must be generated. These are used by expansion logic external to the subsystem to generate the enables locally so that they can be used with bridges or other equivalent logic for crossing between **FCLK** and **SYSCLK** clock domains. The enables would normally occur at the last **FCLK** cycle prior and up to the rising edge of **SYSCLK**. **HINTSYSCLKENCLK** is used to generate the enable in the Always ON domain while **DEBUGHINTSYSCLKENCLK** is used in the debug power domain to do the same.

Note

All fast clock outputs, **FCLK**, **SYSFCLK** and **DEBUGFCLK** can be gated by **HINTSYSCLKENCLK** when CPU 1 in the system is turned off. To enable gating, set the **FCLKHINTGATE_ENABLE** bit in the CLOCK_FORCE register to HIGH.

This allows all logic that runs on the faster clock to run at the same clock speed as the rest of the system and reduces power consumption, but at the cost of slightly higher latency when accessing SRAM3.

2.2.3 External wakeup controller clocks

The capture flops in the EWC typically run on **S32KCLK**, and therefore **S32CLK** is expected to be always on. If however, the system designer replaces the interrupt capture flops in the EWC with latches, the EWC does not require **S32KCLK**..

The wakeup request must be able to be generated from the capture flops without any other clock running, other than **S32KCLK**, if flops are used instead of latches. All other flops in the EWCs are expected to run on SYSCLK or FCLK depending on which core it is associated with.

2.2.4 Power control expansion

Two clocks allow the system designer to add extra power control logic in the Always-On power domain:

- **CPUDEBUGPIKCLK** is a hierarchically gated version of **SYSCLK** clock that is related to the **nPORESETAON** reset domain.

  The resynchronized reset signal for this clock is **nCPUDEBUGPIKRESET**.

  A Q-Channel interface is also provided to allow expansion logic on this clock domain to influence the hierarchical gating of the clock.

  This clock is typically used for power control logic that supports expansion of the CPU and Debug elements.
- **BCRYPTOSPIKCLK** is a hierarchically gated version of **SYSCLK** that is related to the **nWARMRESETAON** reset domain.

  The resynchronized reset signal for this clock is **nBCRYPTOSPIKRESET**.

  A Q-Channel interface is provided to allow expansion logic on this clock domain to influence the hierarchical gating of the clock.

  This clock is typically used for power control logic that supports expansion of the main system that is reset directly or indirectly by **nWARMRESETAON**.

2.2.5 Component clocks

The elements use the system clocks as follows:

SRAM element

Each SRAM runs on one clock. Most run on **SYSCLK**. Optionally one SRAM can run on **FCLK** instead of **SYSCLK**.

The PPU gates the clock that is used in the element for power management.

The SRAM element can locally implement dynamic clock control to further reduce the amount of time the clock is active when the element is idle.

CPU element

Each CPU element runs on a single clock:

- The primary core runs on **SYSCLK**.
- The secondary core runs on **FCLK**.

Because the Cortex-M33 processor has its own clock gating control, the Cortex-M33 cores require an ungated clock to be supplied to each core.

The PPU can gate the clock to the rest of the CPU element, including the instruction cache and the ACG, for power management. Additional dynamic clock control is combined with Cortex-M33 processor clock enable output to further reduce the amount of time the clock is active when the element is idle.

Base element

The base element requires both **FCLK** and **SYSCLK**.

The PPU gates each clock for power management.

The base element locally implements dynamic clock control to further reduce the amount of time each clock is active when the element is idle. These clocks are available for expansion use as the **SYSFCLK**, **SYSSYSCLK**, and **SYSSYSUGCLK** clock outputs.

Debug element

The debug element requires both **FCLK** and **SYSCLK**.

The PPU gates each clock for power management.

The debug element does not implement dynamic clock control. The local clocks are running whenever the element is powered ON. These clocks are available for expansion use as the **DEBUGFCLK** and **DEBUGSYSCLK** clock outputs.

System control element

The system control element requires **S32KCLK**, **FCLK**, and **SYSCLK**.

Because the control element is in the always-on power domain, the element does not have a PPU to gate clocks.

The element implements dynamic clock control for **SYSCLK** to reduce the amount of time **SYSCLK** clock is active when the element is idle.

Crypto element

The CryptoCell, if present, runs on **SYSCLK**.

For power control, the PPU gates this clock. The **CRYPTOSYSCLK** is the power-gated and hierarchical clock-gated version of **SYSCLK**. This clock is available to use for CryptoCell NVM Interface expansion.

The element also implements dynamic clock control for **SYSCLK**, to reduce the amount of time **SYSCLK** is active when the element is idle.

Related references

A.1 Clock, reset, and power control signals on page Appx-A-146.

2.3 Resets

This section describes the resets in the SSE-200 subsystem.

This section contains the following subsections:

2.3.1 Overview

The SSE-200 reset infrastructure takes reset inputs and requests from external sources and from within the subsystem, and generates the resets that are distributed through the system.

The following figure shows the reset infrastructure:

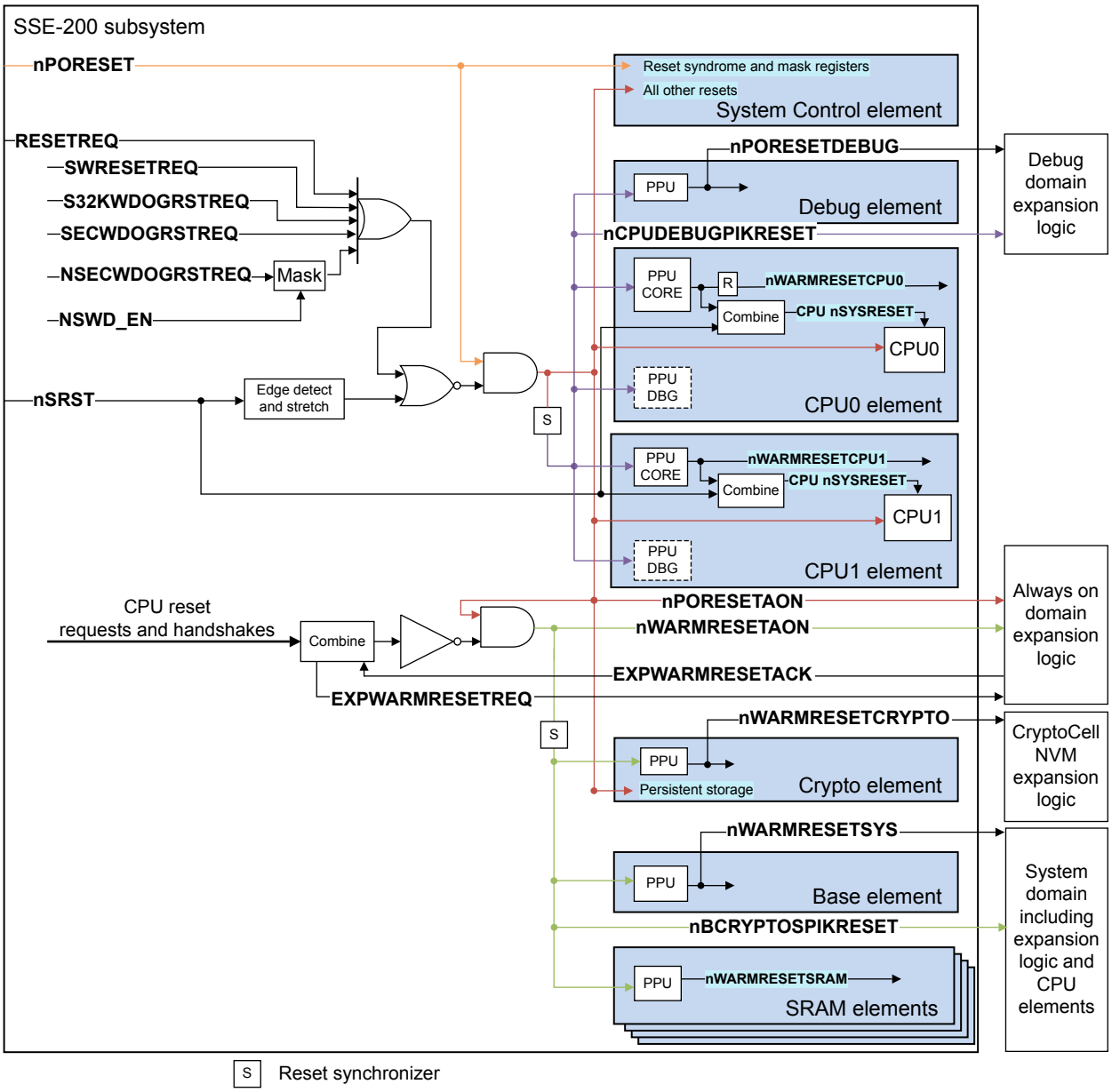

Figure 2-3 Reset interconnection

2.3.2 Reset inputs and outputs

The following reset-related signals in the subsystem are from or to the Always On part of the subsystem:

nPORESET

This is the Power-on reset input for the SSE-200. Arm recommends that this signal is one or more **S32CLK** cycles long.

nSRST

This signal, typically from an external debugger, is the system-wide reset request:

- This reset is applied to the subsystem on the falling edge of the **nSRST** signal and then removed, except for the **nSYSRESET** pin of each processor core.
- If **nSRST** is held low, all processor cores are also stopped from booting.

The **nSRST** input must be a minimum of three **S32KCLK** cycles long. **nSRST** can be held LOW while performing other debug related tasks to indefinitely prevent the processor from execution. This might be done, for example, when inserting a debug certificate into the SRAM. After **nSRST** is deasserted after its assertion, it must be held inactive for at least three **S32KCLK** cycles before being asserting again.

nWARMRESETAON

This signal performs a Warm reset of the system. This signal:

- Must not be used to reset any debug-related logic.
- Is also asserted if **nPORESETAON** is asserted.
- Merges other reset sources that are required to cause system reset.
- Is asynchronous and must be resynchronized before use.

nPORESETAON

The **nPORESETAON** reset output is the Power-on reset signal intended for use by the expansion subsystem logic. This reset output merges other reset sources that are required to cause power-on reset. This reset output is asynchronous and must be resynchronized before use.

RESETREQ

This allows external expansion logic to request a system reset:

- After it is asserted, the signal must be held HIGH until the reset occurs on **nPORESETAON**.
- The signal must be cleared because of the assertion of **nPORESETAON**.

EXPWARMRESETREQ

The **EXPWARMRESETREQ** expansion Warm reset output signal, when set to HIGH, indicates that the reset logic is about to assert a Warm reset to the system. This waits for the **EXPWARMRESETACK** signal to be HIGH before asserting the reset, allowing any external logic to complete critical operations by delaying the assertion of Warm reset.

——— **Note** ———

If **EXPWARMRESETREQ** is not used, you must connect **EXPWARMRESETREQ** to **EXPWARMRESETACK**.

———————————

EXPWARMRESETACK

The **EXPWARMRESETACK** expansion Warm reset input signal works along with **EXPWARMRESETREQ**, allowing external logic to delay the assertion of Warm reset.

The following signals are provided for expansion of other power domains in the subsystem:

nWARMRESETSYS

This is the system reset signal for expansion logic in the PD_SYS power domain.

It supports logic retention and is not asserted if the PD_SYS power domain is entering retention state.

nPORESETDEBUG

This is the Power-on reset signal for debug expansion logic in the PD_DEBUG power domain.

Related references

A.1 Clock, reset, and power control signals on page Appx-A-146.

2.3.3 nPORESET handling

nPORESET directly resets the Reset Syndrome registers and Mask register.

nPORESET is combined with the masked and combined reset requests from watchdog timers, the **RESETREQ** input, the SWRESETREQ register value, and the negative-edge detect and stretched signal of **nSRST**. This generates the internal combined Power-on reset signal **nPORESETAON**, which resets almost all logic within the system.

Note

nPORESETAON resets the PPU, in each PPU-controlled power domain. The PPU then resets the other logic in the domain using its reset pins:

- **DEVWARMRESETn** if the domain does not support retention.
- **DEVRETRESETn** if the domain supports logic or full retention.

2.3.4 Processor reset handling

Because the Cortex-M33 core normally handles reset on its own, **nPORESETAON** is fed directly to the Cortex-M33 core. **nSYSRESET** of the Cortex-M33 core is generated from the local PPU as the PPU can handle Warm reset of the core.

The reset from the PPU, **DEVRETRESETn**, is combined with **nSRST** to hold the core in reset until **nSRST** is deasserted. This allows a debugger to hold the processor core in reset while it uses the Debug Access Port to perform debug operations

Boot after reset

After reset, both processors boot the instruction at the boot address from the vector offset address register in the System Control Registers.

Note

- Static top-level configuration parameters determine the contents of the vector offset address register in the System Control Registers. The default address is `0x00000000` and is statically mapped to code memory of the AHB Master Expansion Code interface.
- Software can modify the boot addresses before a warm reboot of the processors.
- See *A.6.2 Top-level static configuration signals* on page Appx-A-166 and the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Configuration and Integration Manual*.

The TrustZone-M requirement for boot up is that execution starts from a Secure memory space, and optionally automatically executes Non-secure firmware after Secure world initialization. At boot, the SRAM is Secure only. Software must change or restore the settings in the MPC to release memory for Non-secure world use.

The **CPUWAIT** input can force the Cortex-M33 processor to wait before executing the instruction. Each processor in the system has an associated CPU_WAIT register that controls it, if it starts running boot code when it wakes.

See the *Arm® Cortex®-M33 Processor Technical Reference Manual*.

Related references

A.6.2 Top-level static configuration signals on page Appx-A-166.

Certificate Access on page 2-54.

2.3.5 nWARMRESETAON

The **nWARMRESETAON** signal performs a System reset. **nWARMRESETAON** is generated by merging reset requests from the following sources, possibly after masking with the value in the RESET_MASK Register:

- **nPORESETAON**.
- The system reset request from the Cortex-M33 core.

nWARMRESETAON is provided to system designers for external components.

——— **Note** ———

nWARMRESETAON resets each PPU-controlled power domain. The PPU generates the reset in each power domain including the **nWARMRESETAON** reset and the local power management reset. The PPU reset pin is either:

- **DEVWARMRESETn** if the element does not support retention.
- **DEVRETRESETn** if the element supports logic or full retention.

———————————

2.3.6 Power control reset

The *Power Policy Units* (PPUs) in the system are hierarchically clock gated and divided into two clock and reset groups. Each set of clocks and resets are made available for use by expansion power control logic:

- The first group of PPUs run on the **CPUDEBUGPIKCLK** clock. These PPUs are reset using a resynchronized version of **nPORESETAON**, called **nCPUDEBUGPIKRESET**.
- The second group of PPUs run on the **BCRYPTOSPIKCLK** clock. These PPUs are reset using a resynchronized version of **nWARMRESETAON**, called **nBCRYPTOSPIKRESET**.

2.4 CPU elements

This section describes the CPU elements.

This section contains the following subsections:

2.4.1 Overview

There are two Cortex-M33 cores in the SSE-200:

- The primary core in the CPU0 element is synchronous to main interconnect and runs the operating system.
- The secondary core in the CPU1 element contains an FPU and DSP. It is synchronous to the main clock, but runs `N` times faster.

The Cortex-M33 processor has the following features:

- Three-stage pipeline.
- ARMv8-M Mainline profile.
- TrustZone-M Security.
- Up to eight SAU entries each (configurable).
- Up to 16 MPU regions with eight Secure and eight Non-secure (configurable).
- IDAU defining high-level security memory mapping.

Each processor has configuration parameters that can be set in the design stage to specify the processor features including:

- If the FPU is present.
- If the Digital Signal Processing extension instructions are included.
- If the coprocessor interface is present.
- The number of Non-secure and Secure MPU regions.
- The number of security attribution unit regions.
- The number of user interrupts.
- The interrupt priority and interrupt latency that is implemented in the NVIC.
- Debug resources and trace support.

See *2.4.2 Cortex-M33 configurations* on page 2-33 and the SSE-200 Configuration and Integration Manual for more details on system configuration options.

The following figure shows a block diagram of the Cortex-M33 processor logic and CoreSight SoC interface:

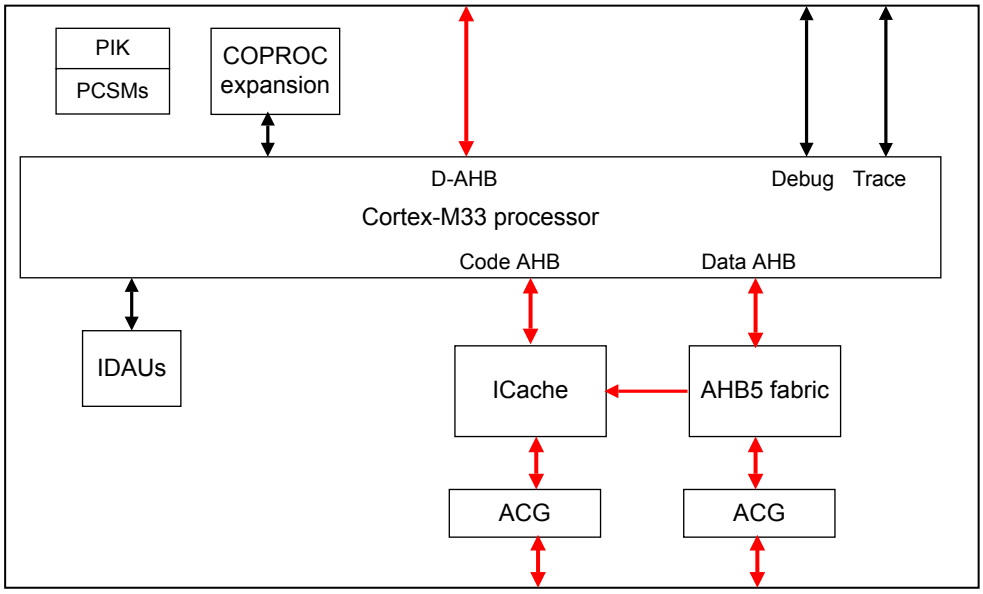

Figure 2-4 CPU element

Related references

2.4.2 Cortex-M33 configurations on page 2-33.

2.4.2 Cortex-M33 configurations

Each processor supports the following configuration with the following features.

Table 2-1 Core configurations

Parameter	CPU 0 option	CPU 1 option	CPU 0 default	CPU 1 default	Description
FPU	CPU0_FPU	CPU1_FPU	0	HAS_FPU	Define if *Floating Point Unit* (FPU) is present: 0: Not Present. 1: Present.
DSP	CPU0_DSP	CPU1_DSP	0	1	*Digital Signal Processing* (DSP) extension instructions are included. 0: Not included. 1: Included.
SECEXT	1	1	-	-	ARMv8-M Security Extensions are always included.

Table 2-1 Core configurations (continued)

Parameter	CPU 0 option	CPU 1 option	CPU 0 default	CPU 1 default	Description
CPIF	CPU0_CPIF	CPU1_CPIF	0	0	Coprocessor interface is included. 0: Not included. This is the only supported value. 1: Included. These are driven by top-level parameters.
MPU_NS	CPU0_MPU_NS	CPU1_MPU_NS	8	8	Defines the number of Non-secure *Memory Protection Unit* (MPU) regions included. Options are: 0, 4, 8, 12, and 16.
MPU_S	CPU0_MPU_S	CPU1_MPU_S	8	8	Defines the number of Secure MPU regions included. Options are: 0, 4, 8, 12, and 16.
SAU	CPU0_SAU	CPU1_SAU	8	8	Defines the number of *Security Attribution Unit* (SAU) regions included. Options are: 0, 4, and 8.
NUMIRQ	CPU0_EXP_NUMIRQ + 32	CPU1_EXP_NUMIRQ + 32	64 + 32 = 96	64 + 32 = 96	Number of user interrupts implemented.
IRQLVL	CPU0_IRQ_LVL	CPU1_IRQ_LVL	4	4	Specifies the number of bits of interrupt priority that is implemented in the NVIC. Supports a range of 3-8. For example, a value of 3 results in eight levels of priority.
IRQLATENCY	IRQLATENCY[31:0] = CPU0_INT_IRQLATENCY[31:0] IRQLATENCY[CPU0_EXP_NUMIRQ+32:32] = CPU0_EXP_IRQLATENCY	IRQLATENCY[31:0] = CPU1_INT_IRQLATENCY[31:0] IRQLATENCY[CPU1_EXP_NUMIRQ+32:32] = CPU1_EXP_IRQLATENCY	-	-	Set interrupt latency.

Table 2-1 Core configurations (continued)

Parameter	CPU 0 option	CPU 1 option	CPU 0 default	CPU 1 default	Description
IRQDIS	IRQDIS[7:0] = 8h00, IRQDIS[17:9] = 9h20, IRQDIS[31:21] = 11h641, If HAS_CRYPTO is True: IRQDIS[8] == 0, IRQDIS[20] == 0, else: IRQDIS[8] == 1, IRQDIS[20] == 1. If SEPARATE_CPUDBG is True: IRQDIS[18] == 0, IRQDIS[19] == 0. else: IRQDIS[18] == 1, IRQDIS[19] == 1.	IRQDIS[7:0] = 8h00, IRQDIS[17:9] = 9h20, IRQDIS[31:21] = 11h641, If HAS_CRYPTO is True present: IRQDIS[8] == 0, IRQDIS[20] == 0. else: IRQDIS[8] == 1, IRQDIS[20] == 1. If SEPARATE_CPUDBG is True: IRQDIS[18] == 0, IRQDIS[19] == 0. else: IRQDIS[18] == 1, IRQDIS[19] == 1.	IRQDIS[NUM_IRQ-32:32] = 448hAAAA ,For IRQDIS[31:0], see cells to the left.	IRQDIS[NUM_IRQ-32:32] = 448hFF00 ,For IRQDIS[31:0], see cells to the left.	Disable support for interrupt. Each bit in IRQDIS corresponds to an interrupt. If the value of a bit in IRQDIS is 1, the corresponding IRQ is not present.
DBGLVL	CPU0_DBGLVL	CPU1_DBGLVL	2	2	Specifies the number of debug resources included. The options are: 0: minimal debug. Not supported. 1: reduced set. Two watchpoint and four breakpoint comparators. 2: full set. Four watchpoint and eight breakpoint comparators. Debug monitor mode is always supported.
ITM	1	1	-	-	Specifies the level of instrumentation trace supported. The options are: 0: No *Instrumentation Trace Macrocell* (ITM) trace included. DWT triggers and counters are not included. 1: Include DWT and ITM trace.
ETM	1	1	-	-	Specifies support for ETM trace. The options are: 0: No ETM trace included. 1: ETM is included.
MTB	0	0	-	-	Specifies support for MTB trace. The options are: 0: No MTB trace included. 1: MTB included.
MTBWIDTH	12	12	-	-	12-bit MTB RAM interface address width. Not used.
WIC	1	1	-	-	WIC included.

Table 2-1 Core configurations (continued)

Parameter	CPU 0 option	CPU 1 option	CPU 0 default	CPU 1 default	Description
WICLINES	CPU0_EXP_NUMIRQ + 35	CPU1_EXP_NUMIRQ + 35	-	-	All interrupts are sensitive to WIC.
CTI	1	1	-	-	CTI included.
RAR	1	1	-	-	Only reset the architecturally required state.

Configuration signals from the SSE-200 determine some of the Cortex-M33 processor options.These configuration signals are listed in the following table.

Table 2-2 Static configuration signals for the Cortex-M33 processor

Signal Name	Tie Value	Description
CFGBIGEND	0	Little-endian data endianness.
CFGSSTCALIB[25:0]	0x200_0000	Secure SysTick calibration configuration. No alternative reference clock is provided, and the frequency of clock arriving at the processor is not computable in hardware. **CFGSSTCALIB[25]** NOREF = HIGH. **CFGSSTCALIB[24]** SKEW = LOW. **CFGSSTCALIB[23:0]** TENMS = 0x00_0000.
CFGNSSTCALIB[25:0]	0x200_0000	Non-secure SysTick calibration configuration indicating that no alternative reference clock is provided, and the frequency of clock arriving at the processor is not computable in hardware. **CFGNSSTCALIB[25]** NOREF = HIGH. **CFGNSSTCALIB[24]** SKEW = LOW. **CFGNSSTCALIB[23:0]** TENMS = 0x00_0000.
CFGFPU	1	FPU hardware support enabled.
CFGDSP	1	DSP hardware support enabled.
CFGSECEXT	1	ARMv8-M security support enabled.
MPUNSDISABLE	0	Disables support for the Non-secure MPU. Set to LOW to not disable.
MPUSDISABLE	0	Disables support for the Secure MPU. Set to LOW to not disable.
SAUDISABLE	0	Disables support for the SAU. Set to LOW to not disable.

Both of the Cortex-M33 processors have a **LOCKSMPU** static configuration signal that has the following functions:

- When HIGH, the signal disables writes to the MPU_CTRL, MPU_RNR, MPU_RBAR, MPU_RLAR, MPU_RBAR_An and MPU_RLAR_An from software or from a debug agent connected to the processor.
- When asserted, this signal prevents changes to programmed secure MPU memory regions and all writes to the registers are ignored.
- Locking of the Secure MPU is not supported.

When both CPU0 and CPU1 exist in the system, the event interfaces of both processors cross connect, so that one processor raises an event with the other. Events can be used with WFE instructions, which also allow the processor to be placed into a lower power state if necessary. If only one processor is configured, the event interface input is tied LOW and the output is not used.

Note

Arm expects the event interfaces are used to communicate near term events between the cores. The architecture does not have the provision to be able to use events to wake a processor core that is in a powered off state. This architecture, however, does support waking the core from retention or clock-off lower power state. In addition, event interfaces on their own, like interrupts, cannot differentiate events that occur close together. It is difficult to count the high frequency events that occur close together. For example, when the difference in clock speed between both processors is large, or the software handler that is required to deal with events is slow.

2.4.3 Instruction cache

Each CPU element contains a single L1 instruction cache that is on the code AHB interface of the Cortex-M33 core.

The cache:

- Reduces the code access fetches targeting the flash memory and therefore reduces activity on the flash and power.
- Reduces the overall latency of code access. This permits increasing the latency of the instruction fetch from memory which might be necessary to deal with long timing paths.

The cache has the following features:

- 2-way set associative.
- 16-Byte cache lines.
- Configurable size. The top-level parameters `CPU0_ICACHESIZE` and `CPU1_ICACHESIZE` set the size.
- Configuration interface local to each processor.
- Supports uncached bypass operation.

Note

- The instruction code fetches in the region from `0x0000_0000` to `0x1FFF_FFFF` are cached. All accesses to other memory regions are not cached.
- Each cache includes a configuration interface that is only accessible to the processor that is connected to it and it resides at address `0x5001_0000`. This configuration interface is not accessible to other masters in the system.

The cache supports Secure and Non-secure segregation of contents by retaining the security attributes of the cached contents. If the SAU or the IDAU configurations change, the cache might contain contents that do not match the new security settings. To maintain security, you must therefore disable and invalidate the cache before modifying the SAU or IDAU.

Because this is an instruction cache, only read accesses are subject to caching. All write accesses bypass the cache. If the INVMAT option of the cache is enabled, when a cacheable write access occurs to a line that exists in the cache, that cache line is invalidated.

Note

If a cache line exists in the cache that matches a non-cacheable write access, the invalidation does not occur. Therefore, when changing the attribute of an address region from cacheable to non-cacheable that the instruction cache accesses, you must invalidate the cache.

The following table lists the instruction cache configuration parameters.

Table 2-3 Instruction cache configurations

Parameter	Processor 0 configuration	Processor 1 configuration	Processor 0 default value	Processor 1 default value	Description
CSIZE	CPU0_ICACHESIZE	CPU1_ICACHESIZE	2KB	2KB	Define the cache size. Supported cache sizes: • 9 = 512B. • 10 = 1KB. • 11 = 2KB. • 12 = 4KB. • 13 = 8KB. • 14 = 16KB. • Other = Reserved.
DMA	CPU0_ICACHEDMA	CPU1_ICACHEDMA	0	0	Defines the existence of micro DMA capability and line locking capability. When set to 1, the instruction cache provides cache line prefetch and locking capability.
INVMAT	CPU0_ICACHEINVMAT	CPU1_ICACHEINVMAT	0	0	Invalidate on Write Match. When set to 1, any cacheable writes to a line that also exists in the instruction cache results in the cache line being invalidated.
COFFSET	0	0	0	0	Cacheable Region Offset Upper bits to compare against. This parameter is fixed.
COFFSIZE	3	3	3	3	Cacheable Region Size. Value defines the number of upper address bits to compare. Therefore, the addressable space is $2^{(32\text{-COFFSIZE})}$.
STATS	CPU0_ICACHESTATS	CPU1_ICACHESTATS	1	1	1 = include statistics functionality.

Table 2-3 Instruction cache configurations (continued)

Parameter	Processor 0 configuration	Processor 1 configuration	Processor 0 default value	Processor 1 default value	Description
XOM	CPU0_XOM	CPU1_XOM	0	0	Enable Execute Only Memory support. When set to 1, the **HRUSER[0]** signal on the AHB5 Master Expansion Code Interface is used to indicate if current data being returned is Execute Only. If the data type access arriving at the instruction cache targets a XOM location, the instruction cache masks the data. When set to 0, this **HRUSER[0]** is ignored by the associated instruction cache.
REDUCE_READS	ICACHERRDS		1	1	Reduce instruction cache Tag Reads. When set to 1, it masks off an access to the TAG RAM if this set has been previously accessed and the RAM data is valid and does not need re-accessing.

2.4.4 CPU_WAIT control

Each core in the system has an associated CPU_WAIT register. The register controls the core if it starts running boot code when it wakes.

Caution

The only way for a core to enter a lower-power state is to at least enter into WFI:

- If after powering up, **CPU_WAIT** is held HIGH, the core will not be able to run code to enter WFI, and the core remains powered up.

  To prevent the core from being on indefinitely, if **CPU_WAIT** is HIGH, ensure that there is at least another agent in the system that can clear this register to allow the core to start up boot.

2.4.5 Interrupts

The CPU element provides the following events that can generate interrupts in the system:

- Instruction cache IRQ.
- PPU0/PPU1 IRQ.
- *Cross Trigger Interrupt* CTI IRQ [1:0].

In addition to the interrupts, the following events can be used to control instruction execution:

- RXEV (incoming event) is the event that is received by the Cortex-M33 processor and is connected to the TXEV of the other processor in the system.
- TXEV (outgoing event) is the event that is transmitted by Cortex-M33 processor and is connected to the RXEV of the other processor in the system.

The following table lists the interrupt sources for CPU0 and CPU1.

Table 2-4 Interrupt sources

Interrupt Input	CPU 0 and CPU 1 interrupt source
NMI	Combined Secure Watchdog, S32kwatchdog, and NMI_Expansion
IRQ[0]	Non-secure Watchdog Reset Request
IRQ[1]	Non-secure Watchdog Interrupt
IRQ[2]	S32K Timer
IRQ[3]	Timer 0
IRQ[4]	Timer 1
IRQ[5]	Dual Timer
IRQ[6]	Message Handling Unit 0 CPU*n* Interrupt
IRQ[7]	Message Handling Unit 1 CPU*n* Interrupt
IRQ[8]	CryptoCell-312 (if CryptoCell is present)
IRQ[9]	MPC Combined (Secure)
IRQ[10]	PPC Combined (Secure)
IRQ[11]	MSC Combined (Secure)
IRQ[12]	Bridge Error Combined Interrupt (Secure)
IRQ[13]	CPU*n* instruction cache Interrupt
IRQ[14]	Reserved
IRQ[15]	SYS_PPU
IRQ[16]	CPU0_PPU
IRQ[17]	CPU1_PPU
IRQ[18]	CPU0DBG_PPU (if SEPARATE_CPUDBG_PD configuration is True)
IRQ[19]	CPU1CBG_PPU (if SEPARATE_CPUDBG_PD configuration is True)
IRQ[20]	Crypto PPU (if CryptoCell is present)
IRQ[21]	Reserved
IRQ[22]	RAM0_PPU
IRQ[23]	RAM1_PPU
IRQ[24]	RAM2_PPU
IRQ[25]	RAM3_PPU
IRQ[26]	DEBUG_PPU
IRQ[27]	Reserved
IRQ[28]	CPU*n*CTIIRQ0
IRQ[29]	CPU*n*CTIIRQ1
IRQ[30]	Reserved
IRQ[31]	Reserved
IRQ[95:32]	Expansion Interrupt Inputs.

Note

- Unless specified, both cores receive the same interrupt signals.
- Each processor only sees its own local instruction cache interrupt and CTIIRQ interrupts. Instruction cache and all PPUs interrupts must be handled as Secure interrupts.
- If, because of the system configuration, an interrupt source does not exist, the unused interrupt pin is not used, and the interrupt is disabled and reserved.

The Non-secure watchdog interrupt signal and its reset request signal are both used to generate interrupts to the processor. The reset request interrupt must be handled as a Secure interrupt by the *Trusted Execution Environment* (TEE) so that it does not directly reset the system. To enable the Non-secure watchdog reset of the system, set the NSWD_EN field in the RESET_MASK register to HIGH.

The Secure watchdog interrupt request and the S32K Watchdog interrupt request are merged to generate an internal Non-Maskable Interrupt. This interrupt can be used to raise an NMI interrupt on CPU0 or CPU1.

The settings in the NMI_ENABLE register can be modified to allow software to route the watchdog interrupts to a single core. That core can then own the interrupt handling for the watchdog interrupt.

The NMI_ENABLE register has masks for the expansion external NMI interrupt inputs to allow software to determine if the external NMI must be raised. These Interrupts must all be handled as Secure interrupts.

There are two *Message Handling Units* (MHUs) in the system. If the system supports a *Trusted Execution Environment* (TEE), one MHU must be configured as a Secure MHU and the other as a Non-secure MHU.

2.4.6 Power domains

See *2.9 Power control infrastructure* on page 2-59.

2.4.7 Clock domains

See *2.2 Clocks* on page 2-22.

2.4.8 Security

The CPU element is TrustZone-M security aware. The Cortex-M33 core Security Extension is always enabled.

The software-controlled *Secure Attribution Unit* (SAU) and the *Implementation Defined Attribution Unit* (IDAU) define the memory map secuirty attribues, for a core.

Implementation Defined Attribution Unit (IDAU) security values are defined as follows.

Table 2-5 CPU element IDAU definition

Address		Output					Description
From	To	IDAUNS	IDAUNSC	IDAUID	IDAUIDV	IDAUNCHK	
0x0000_0000	0x0FFF_FFFF	1	0	0	1	0	Non-secure
0x1000_0000	0x1FFF_FFFF	0	NSCCFG[0]	1	1	0	Secure (might be callable)
0x2000_0000	0x2FFF_FFFF	1	0	2	1	0	Non-secure
0x3000_0000	0x3FFF_FFFF	0	NSCCFG[1]	3	1	0	Secure (might be callable)

Table 2-5 CPU element IDAU definition (continued)

Address		Output					Description
From	To	IDAUNS	IDAUNSC	IDAUID	IDAUIDV	IDAUNCHK	
0x4000_0000	0x4FFF_FFFF	1	0	4	1	0	Non-secure
0x5000_0000	0x5FFF_FFFF	0	0	5	1	0	Secure
0x6000_0000	0x6FFF_FFFF	1	0	6	1	0	Non-secure
0x7000_0000	0x7FFF_FFFF	0	0	7	1	0	Secure
0x8000_0000	0x8FFF_FFFF	1	0	8	1	0	Non-secure
0x9000_0000	0x9FFF_FFFF	0	0	9	1	0	Secure
0xA000_0000	0xAFFF_FFFF	1	0	0xA	1	0	Non-secure
0xB000_0000	0xBFFF_FFFF	0	0	0xB	1	0	Secure
0xC000_0000	0xCFFF_FFFF	1	0	0xC	1	0	Non-secure
0xD000_0000	0xDFFF_FFFF	0	0	0xD	1	0	Secure
0xE000_0000	0xE00F_FFFF	1	0	0xE	1	1	Exempted
0xE010_0000	0xEFFF_FFFF	1	0	0xE	1	0	Non-secure
0xF000_0000	0xF00F_FFFF	0	0	0xF	1	1	Exempted
0xF010_0000	0xFFFF_FFFF	0	0	0xF	1	0	Secure

For the mapping of these regions against the main system memory map, see *3.2 Memory map* on page 3-74.

2.4.9 External wakeup

See *2.9.5 External wakeup controllers* on page 2-63.

2.5 Base element

This section describes the Base element.

This section contains the following subsections:

2.5.1 Overview

The base element provides the following features:

- A multilayer AHB5 interconnect for all the subsystem elements and expansion buses.
- A Memory Protection Controller for each SRAM element.
- AHB to APB bus converters and TrustZone Peripheral Protection Controllers for:
  - Two CMSDK Timers.
  - One CMSDK Dual Timer.
  - One CMSDK Watchdog timers.
  - Message Handling Units that can send messaging interrupts to each core.
- Two AHB5 slave expansion ports and three master expansion ports.
- A security controller with expansion support.
- A single voltage domain and power-gated region.
- Two synchronous clock domains.

The following figure shows a block diagram of the base element.

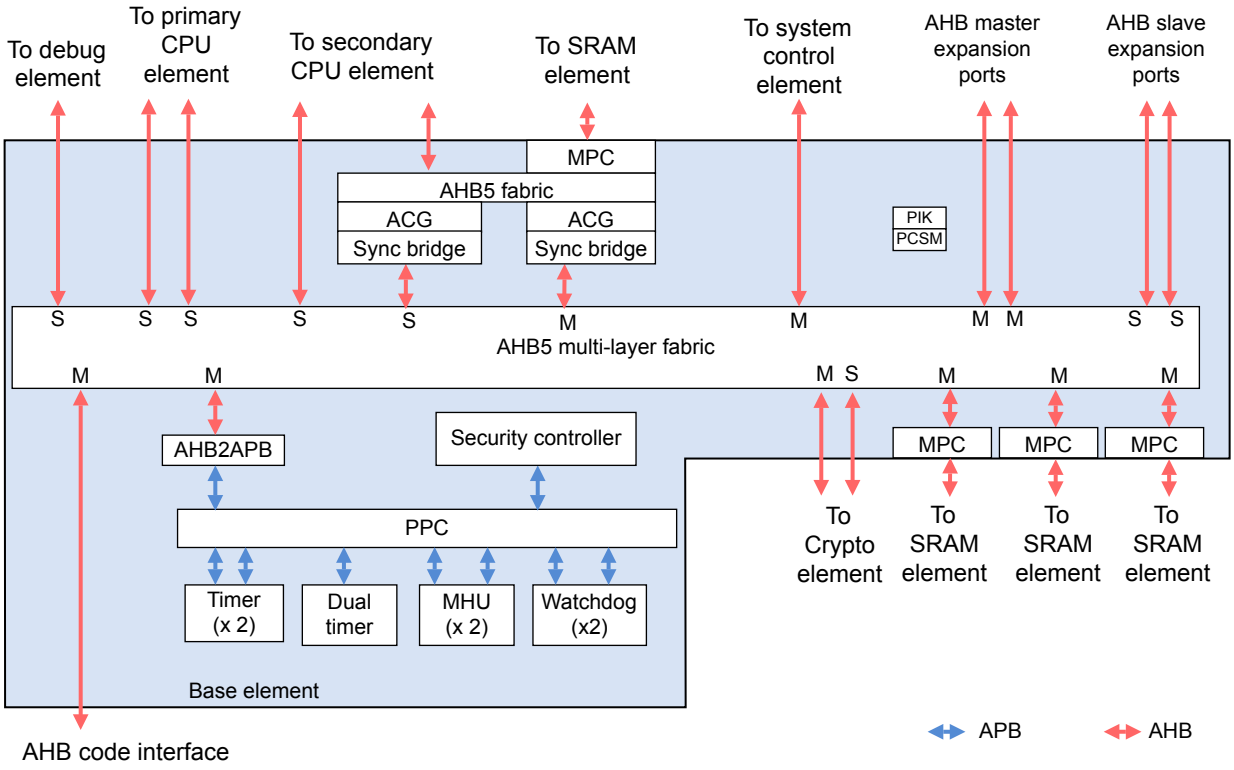

Figure 2-5 Base element block diagram

2.5.2 Component sources

The following components are from the SIE-200 product:

- AHB5 interconnect.
- *Memory Protection Controllers* (MPC).
- AHB5 to APB bus converter.
- APB *Peripheral Protection Controller* (PPC).
- AHB5 slave mux.
- AHB5 master mux.
- AHB5 sync-up bridge.
- AHB5 sync-down bridge.
- *Access Control Gates* (ACG).

The following components are from the *Cortex-M System Design Kit* (CMSDK):

- Timers and dual timers.
- Watchdog timers.

The following components are unique to the subsystem:

- The security controller which implements both the Secure Privilege Control Register Block and the Non-secure Privilege Register Block.
- Message handling units that allow messaging interrupts to be sent to each processor core.

2.5.3 AHB5 bus matrix

The bus matrix connects AHB5 components through its slave and master ports.

The following table lists the bus matrix slave ports.

Table 2-6 AHB5 bus matrix slave ports

Port number	Slave port
S0	CPU0 Code AHB Interface
S1	CPU0 System AHB Interface
S2	CPU1 Code AHB Interface
S3	CPU1 System AHB Interface
S4	CryptoCell AHB interface
S5	Debug certificate AHB Interface
S6	Slave Expansion0 AHB Interface
S7	Slave Expansion1 AHB Interface

The following table lists the bus matrix master ports and their connections to the slave ports.

Table 2-7 AHB5 bus matrix master ports

Master port	S0	S1	S2	S3	S4	S5	S6	S7
AHB Master Expansion Code Interface	Y	-	Y	-	Y	-	Y	
AHB Master Expansion 0 Interface	-	Y	-	Y	Y	-	Y	Y
AHB Master Expansion 1 Interface	-	Y	-	Y	-	-	Y	Y
SRAM0 AHB Interface	-	Y	-	Y	Y	Y	Y	Y
SRAM1 AHB Interface	-	Y	-	Y	Y	-	Y	Y
SRAM2 AHB Interface	-	Y	-	Y	Y	-	Y	Y
SRAM3 AHB Interface	-	Y	-	Y	Y	-	Y	Y
CryptoCell code APB	Y	-	Y	-	-	-	Y	
CryptoCell configuration APB	-	Y	-	Y	-	-	Y	Y
All other system control and peripherals	-	Y	-	Y	-	-	Y	Y

Note

Paths to CryptoCell only exist if the associated Crypto element exists.

2.5.4 SRAM

This section describes the SRAM-related features of the base element.

Memory protection controller

The MPCs in the base element control the security for the SRAM regions.

One *Memory Protection Controller* (MPC) is included on the path to each SRAM block so that accesses can be blocked when a security violation occurs.

Each SRAM block is implemented within an SRAM element. Each MPC APB configuration interface is mapped to the following base addresses:

- `0x5008_3000` for SRAM Bank 0.
- `0x5008_4000` for SRAM Bank 1.

- 0x5008_5000 for SRAM Bank 2.
- 0x5008_6000 for SRAM Bank 3.

The cfg_init_value of each MPC is tied to 0 so that at boot, the SRAM is Secure only. Software must change or restore the settings in the MPC to release memory for Non-secure world use.

The BLK_SIZE configuration of each MPC, which defines the MPC block size is defined by the top-level parameter SRAM_MPC_BLK_SIZE. This is set at a default value of 3 to select 256 byte blocks.

The GATE_PRESENT configuration parameter of each MPC is set to 0 to disable the MPC gating feature.

All SRAM MPCs reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**.

Related references

3.5 SRAM element on page 3-121.
2.6 SRAM elements on page 2-49.

Tightly-Coupled Memory

The AHB5 fabric is designed to keep an SRAM element, SRAM Bank 3, close to the secondary core, and for both to run at a higher clock speed. This allows the SRAM to function as *Tightly-Coupled Memory* (TCM) on the secondary processor data bus.

2.5.5 AHB5 TrustZone peripheral protection controllers

The peripheral protection controller gates transactions to peripherals based on whether there is a security violation.

The Base element contains a single APB *Peripheral Protection Controller* (PPC) and a single AHB PPC. These PPCs do not have their own software accessible programming interface. Instead, control of this PPC resides in the Secure Privilege Control Block and Non-secure Privilege Control Block. See *3.4.6 Security Privilege Control Block* on page 3-101 and *3.4.7 Non-secure Privilege Control Block* on page 3-116.

See the *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual*.

2.5.6 Message handling unit

Two *Message Handling Units* (MHU) allow software to raise interrupts to the processor cores.

Related references

3.4.5 Message handling unit on page 3-99.

2.5.7 Timers and watchdogs

This sections describes the timers and watchdogs in the Base element.

CMSDK APB Timers

The SSE-200 includes two instances of the CMSDK APB timers:

- CMSDK TIMER 0 in Non-secure region at 0x4000_0000 and in Secure region at 0x5000_0000.
- CMSDK TIMER 1 in Non-secure region at 0x4000_1000 and in Secure region at 0x5000_1000.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

Dual Timer

The CMSDK APB dual-input timer consists of two programmable 32-bit down-counters that can generate interrupts when they reach zero. The operation of each timer module is identical.

The Base element contains one APB dual-input timer which is located in Non-secure region at 0x4000_2000 and in Secure region at 0x5000_2000.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

Watchdog timers

The Base element instantiates two APB watchdog timers that are connected to the internal APB bus. Each watchdog is permanently mapped to either a Secure or a Non-secure region of address space:

- Non-secure CMSDK Watchdog in the Non-secure region at `0x4008_1000`.

  The Non-secure Watchdog can raise an interrupt to both processors. On a watchdog reset request event, a separate interrupt is raised but software can choose to allow it to directly reset the system.
- Secure CMSDK Watchdog in the Secure region at `0x5008_1000`.

  The Secure watchdog can raise a Non-Maskable Interrupt (NMI) to both processors . In this case, a watchdog reset event resets the entire system.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

2.5.8 Expansion ports

The following sections describe the different types of AHB5 ports.

AHB slave expansion interfaces

Two expansion AMBA AHB5 slave interfaces are provided to allow the system integrator to add extra bus masters to the system. These interfaces are:

- AHB5 slave expansion 0 interface.
- AHB5 slave expansion 1 interface.

The Base element also provides an AHB5 master expansion code interface. This master interface is provided primarily to provide access to code memory.

Each of these interfaces supports the following features:
- 32-bit address.
- 32-bit data width.
- TrustZone-Armv8-M security support, with **HNONSEC** signal.
- **HPROT** signal indicates the access property, including if the access is privileged.
- Exclusive access support to SRAM memory.

AHB5 slave expansion 0 interface can access the entire system memory map. The AHB5 slave expansion 1 interface can access the entire system memory map but it cannot access the AHB5 master expansion code interface.

AHB master expansion interfaces

Two expansion AMBA AHB5 master interfaces are provided to allow the system integrator to add extra slave peripherals to the system. These interfaces are:

- AHB5 master expansion 0 interface.
- AHB5 master expansion 1 interface.

The SSE-200 also provides an AHB5 master expansion code interface. This master interface is provided primarily to provide access to code memory.

Each of these interfaces supports the following features:
- 32-bit address bus for both AHB5 Master Expansion 0 and AHB5 Master Expansion 1 interfaces, with each access providing the full 32-bit address.

  29-bit address bus for the AHB5 Master Expansion Code interface.
- 32-bit data width.

- TrustZone Security support, with the inclusion of **HNONSEC** and **HPROT** signals.
- Exclusive access support. However, for exclusive accesses to function correctly, the slave memory device in the expansion system that supports exclusive accesses must implement exclusive access monitoring.

Note

In the expansion system, Arm expects the system integrator to insert a *Memory Protection Controller* (MPC) on the path to code memory, on the AHB5 Master Expansion Code Interface. The MPC provides security access gating, for the aliased memory region that this interface supports.

Related references

A.3 AHB expansion bus signals on page Appx-A-154.

2.5.9 Security controller

The Security Controller contains the Secure Privilege Control Register Block and the Non-secure Privilege Register Block. These implement program-visible states that allow software to control security gating units within the design. The block provides extra expansion security control signals to support extra security gating units in the expansion logic.

Related references

3.4.6 Security Privilege Control Block on page 3-101.

2.5.10 Power control

See *2.9 Power control infrastructure* on page 2-59.

2.6 SRAM elements

This section describes the SRAM elements.

This section contains the following subsections:

2.6.1 Overview

The SSE-200 supports four SRAM elements. Each SRAM element has the following features:

- One bank of single port SRAM.
- Zero clock cycle latency.
- ON/OFF/MEM_RET power policy support.
- Exclusive access support.

A Memory Protection Controller in the Base element manages Secure access.

Related references

Memory protection controller on page 2-45.

2.6.2 SRAM banks

There are four banks of contiguous SRAM. Each SRAM has the following features:

- Configurable 8KB, 16KB, 32KB, or 64KB.
- *Exclusive Access Monitor* (EAM).
- Independent memory power control, each residing in PD_SRAM<N> power domain.

The last bank of SRAM is the *Data Tightly Coupled Memory* (DTCM) that provides high throughput because it runs at the same speed as secondary core.

Note

Each SRAM element has a Memory Protection Controller that are associated with it and is implemented within the Base element.

2.6.3 AHB5 Exclusive Access Monitor

The EAM monitors access to slaves downstream of it. The component implements the EAM feature that is described in the AHB5 specification.

See the *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual*.

2.7 System control element

The section describes the System control element.

This section contains the following subsections:

2.7.1 System Information Register Block

The System Information Register Block provides information on the system configuration and identity.

See *3.6.2 System information registers* on page 3-123 for information about the registers that this block implements.

2.7.2 System Control Register Block

The System Control Register Block implements registers for power, clocks, resets, and other general system control.

See *3.6.1 System control registers* on page 3-122 for information about the registers that this block implements.

2.7.3 Timers and watchdogs

The SSE-200 implements a single CMSDK Timer and a single CMSDK Watchdog that run on**S32KCLK**.

The timeris aliased onto Secure and Non-secure regions. The watchdog is permanentley mapped to the Secure region. The System Control element APB Peripherals Protection Controller controls the region that the timer resides in. The timer can raise an interrupt to both processor cores and the watchdog timer can raise a Non-Maskable Interrupt to both processor cores. When the watchdog reset event occurs, it resets the entire system. See *2.3 Resets* on page 2-27.

The timer and watchdog can be halted using CTI triggers from the debug subsystem. See *2.8 Debug element* on page 2-52.

2.7.4 Peripheral Protection Controller

The System Control element contains a single APB *Peripheral Protection Controller* (PPC).

The PPC does not have its own software accessible programming interface. Instead control of this PPC resides in the Secure Privilege Control Block and Non-secure Privilege Control Block. See *3.4.6 Security Privilege Control Block* on page 3-101 and *3.4.7 Non-secure Privilege Control Block* on page 3-116.

2.7.5 Power Policy Units

The System Control element also provides access to *Power Policy Unit*s (PPUs) that are used to control power domains in the system.

All PPUs reside in the always on power domain, PD_AON, and are reset by **nPORESETAON**.

The following table lists the configuration of all PPUs in the system.

Note

M is 0 for CPU 0 and 1 for CPU 1 if it exists, and N represents number(s) between 0 and the number of SRAM Banks minus 1.

Table 2-8 Power Policy Units configurations

PPU configuration	PD_SYS	PD_CPU <M>CORE	PD_CPU <M>DBG[a]	PD_DEBUG	PD_SRAM<N>	PD_CRYPTO
PPU name	SYS_PPU	CPU<M>_PPU	CPU<M>DBG_PPU[a]	DEBUG_PPU	RAM<N>_PPU	CRYPTO_PPU
Device interface type	P-Channel	P-Channel	Q-Channel[a]	Q-Channel	Q-Channel	Q-Channel
Default Power Policy (DEF_PWR_POLICY)	ON	OFF[b]	OFF[a]	ON	ON	ON
Default Power mode dynamic transition enable (DEF_PWR_DYN_EN)	OFF	ON	ON[a]	OFF	OFF	OFF
Dynamic support	ON, FULL_RET[c], OFF	ON, FULL_RET[c], WARM_RST, OFF	ON[a], OFF[a]	ON, OFF	ON, MEM_RET OFF	-
Static support	ON, FULL_RET[dc], WARM_RST[d], OFF[d]	ON[d], FULL_RET[dc], WARM_RST[d], OFF[d]	ON[a], WARM_RST[ad], OFF[ad]	ON, WARM_RST[d], OFF[d]	ON, MEM_RET[d], WARM_RST[d], OFF[d]	ON, WARM_RST[d], OFF
PWR_MODE_ENTRY_DEL_CFG	0					
SW_DEV_DEL_CFG	0					
LOCK_CFG	0					
MEM_RET_RAM_REG_CFG	0					
FULL_RET_RAM_REG_CFG	0					
FUNC_RET_RAM_REG_CFG	0					
STA_POLICY_PWR_IRQ_CFG	0					
STA_POLICY_OP_IRQ_CFG	0					
Operating Mode support	None					

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

a If separate CPU debug power domain is not present (when SEPARATE_CPUDBG_PD configuration is False), then this column is invalid. Instead for each core the PD_CPU<M>DBG power domain is merged into the PD_CPU<M>CORE power domain and controlled as if it is just a PD_CPU<M>CORE power domain.

b While the default dynamic power mode of these PPUs is set to OFF, at power-up reset, **nSRST** reset, watchdog resets, reset request on **RESETREQ** input or reset caused by a register write to SWRESET, the SSE-200 requests for the processor to powerup depending on the CPUWAIT register values. AIRCR.SYSRESETREQ based reset request does not depend on CPUWAIT. See *2.9.4 System boot when powering up* on page 2-63 for more information.

c Logic retention support for both the PD_SYS and PD_CPU<M>CORE are configuration options through the CPU_SYS_RETENTION render configuration. If Logic retention is not supported (when CPU_SYS_RETENTION is False) then the Dynamic support and Static Support for FULL_RET does not exist.

d The PPU must not be programmed to use static power state because they can cause system deadlocks. And any attempt to write to the PPU_PWPR.OP_DYN_EN register to change to static power mode for each PPU results in a bus error being returned and the write being blocked.

2.8 Debug element

This section describes the Debug element.

This section contains the following subsections:

2.8.1 Overview

The following figure shows a block diagram of the Debug element:

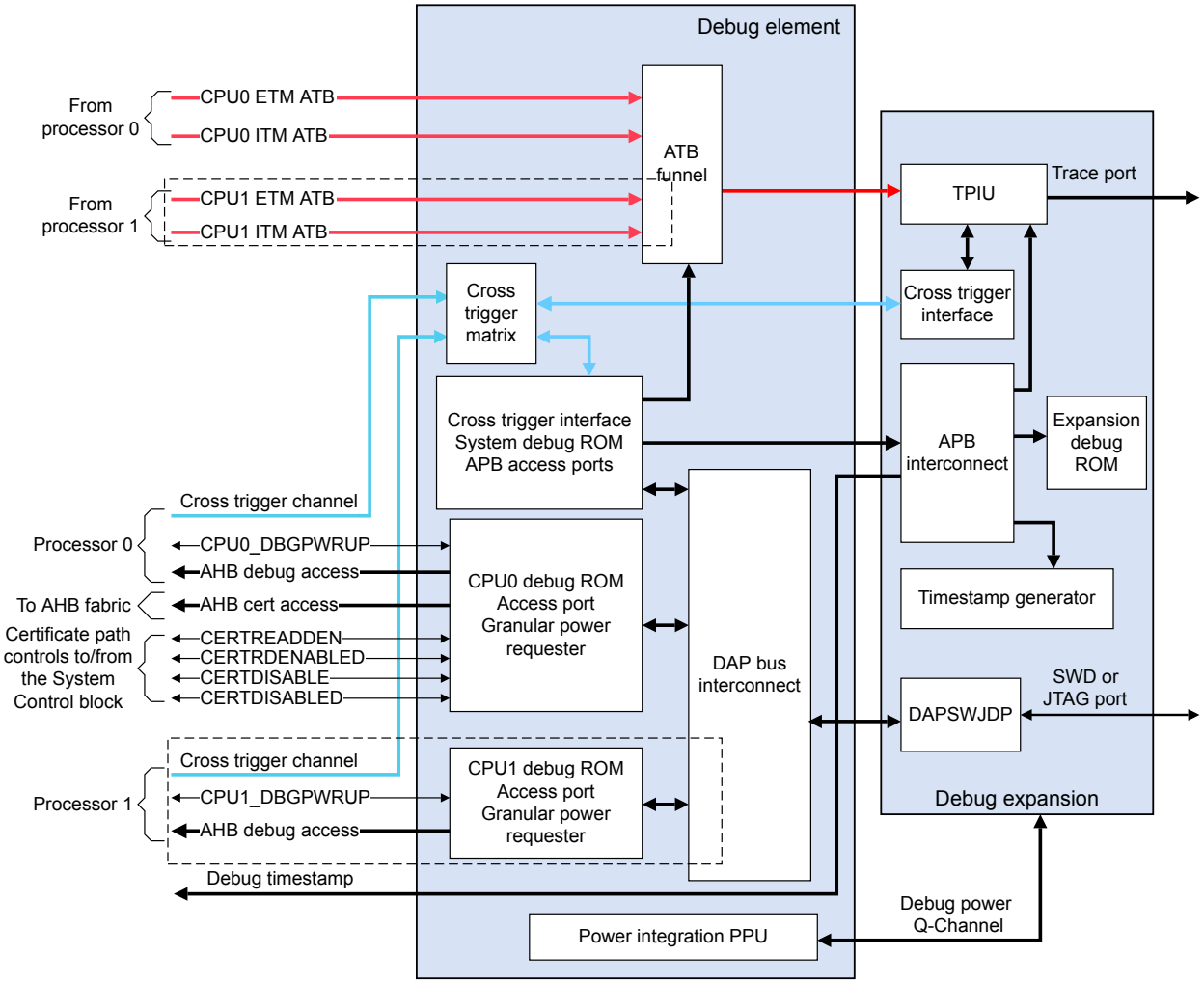

Figure 2-6 Debug element block diagram

The Debug element provides the following features and interfaces:

- Single *Debug Access Port* (DAP) shared between both processors and other system level debug logic.
- Debug certificate access filter.
- Combining all trace sources within the system to a single shared ATB trace output to support a single TPIU trace output.
- CTM and CTI.

- Timestamp distribution from expansion logic to the debug logic of both processors.
- Granular Power Requester to allow the debugger to selectively request parts of the SSE-200 to turn on.

In this figure, the example debug expansion logic shows an SWJ - DP being used. The SWJ - DP is a combined JTAG-DP and SW-DP that enables you to connect either an SWD or JTAG probe to a target. It is a standard CoreSight debug port.

The expansion example also shows the use of a single TPIU to bridge the ATB output to a trace output. The debug logic also provides an APB interface to allow the external debug logic to be controlled through the single DAP interface.

Note

The default example implementation reuses the TPIU and SWJ - DP that is provided as part of the Cortex-M33 package. This configuration is sufficient for basic use.

For more sophisticated multiprocessor debug solution, a full CoreSight SoC IP solution can be licensed and implemented.

Related references

A.4 Debug and Trace signals on page Appx-A-157.

2.8.2 Debug access

The Debug element provides a DAP Bus input that connects to the following independent APs:

- AHB-AP for CPU 0 debug access and certificate access.
- AHB-AP for CPU 1 debug access.
- APB-AP for System debug access.

An external DAP, such as the JTAG DAP, is required in the expansion subsystem to drive the Debug Access Interface.

Processor Debug Access

To provide Debug access to each processor, an AHB-AP is provided along with a single bit *Granular Power Requester* (GPR) to allow the debugger to request the processor to power up. A Debug ROM is also provided to point to both the internal Debug ROM of the processor and the GPR.

The GPR power request handshake signal connects to all PPUs in each CPU element to allow debugger to request the processor to power up. The AHB-AP, for each processor, is always accessible regardless of the debug authentication settings. It is the Cortex-M33 core that performs the task of gating access depending on the debug authentication signals settings.

System Debug Access

All other debug logic in the Debug element is accessed by an APB-AP.The **NIDEN** debug authentication signal controls access to determine if debug is allowed.

Connected to the APB Debug are:

- System Debug ROM that describes all debug components in the Debug element accessible from this APB-AP. This includes a pointer to an external Expansion debug ROM at address location `0xF008_0000`.
- A trace funnel
- A trace FIFO
- A Cross Trigger Interface that support trigger to and from:
  - The trace FIFO
  - Timers and Watchdogs in the system to implement halt on trigger.
- The Debug APN Expansion Interface allows customers to add extra Debug components to the system.

A *Cross Trigger Matrix* (CTM) is included in the Debug element so that all CTIs, including those in the processors and in the expansion subsystem, can trigger each other.

Timestamp must be provided from the expansion subsystem so that it can be used with other trace sources outside the subsystem.

Certificate Access

The Debug element provides an access path from the Debug AHB Access Interface to enable it to write to an 8KB window. The window address location is `0x3000_0000` to `0x3000_1FFF` during initial system boot. This is a write-only path by default after Power-on reset.

Read access to the same 8KB window can be enabled using a control register bit CERTREADEN. The CERTDISABLE control register can disable this access port, and therefore disabling access to this window.

This certification access piggybacks onto the CPU0 AHB-AP. Before using the certificate path, the CPU0 must first be powered up using the CPU0 GPR controls which indirectly also wake the Base element.

This allows a debugger to deposit a debug certificate or a request for data into the memory location at boot while the processor is held off from booting by holding the **nSRST** reset input. If the debugger does not support **nSRST**, the firmware can add debugger wait cycles to allow the debug software to write the SOC ID request to memory before boot. It also provides time to write in the certificate.

When the processor is allowed to boot, the Secure boot code can then check if the content in memory is a certificate or a request:

- If it is a request for information, the Secure boot code can write the requested data into the SRAM window before enabling read access for the debugger to the window by using CERTREADEN.
- If the content is a certificate, after the certificate is checked, the processor can set CERTDISABLE to disable access from the debugger to this window. After CERTDISABLE is set to 1, the processor must check that the corresponding CERTDISABLED = 1, which indicates that the window is now closed, before continuing with the boot process.

For a system that already has debug authentication signals all tied LOW by default, or with fuses, can override the debug authentication settings using the payload in a valid certificate to re-enable debug or Secure debug.

To generate the certificate, it might be necessary to use the read capability of the path to request information about the system. The following rules must be followed if using the certification path:

- When CPU0 wakes, if the content of the SRAM window is a certificate, before it can proceed with any certification check or continue with the boot process, it must close the certification path by writing to the CERTDISABLE register and wait for the CERTDISABLED status to go to 1.
- PD_SYS, PD_CPU0CORE, and PD_SRAM0 must remain in the ON state before CERTDISABLED = 1. This means that CPU0 must not enter WFI and SleepDeep state.
- If a certification check is performed after closing the window, the processor must wipe the certificate 8KB window in SRAM before allowing software to execute from the SRAM window.
- Before enabling read access, the content of the SRAM window must be wiped before it is used.
- When using the certificate window, the Secure firmware or the debugger must not request a system reset from any processor AIRCR register.

When CERTDISABLE is set to 1, it is not possible to later re-enable the certification path except by using Power-up reset or **nSRST**.

The following example sequence list events that a debugger can use to request for identity information from the subsystem:

1. Pull **nSRST** LOW to reset the SoC, and hold **nSRST** LOWto prevent the processors from booting.
2. The debugger writes a pattern into the SRAM window to request for information to be returned, which can be a request for this SoC ID.

Note

Secure write is required.

3. **nSRST** is deasserted, allowing the processor to boot.
4. The processor boot code checks the SRAM window, and discovers the request.
5. The processor wipes the content of SRAM window to all zeros.
6. The processor writes the System or SoC identity data into the window followed by a flag to indicate that the data is ready.
7. The processor writes to the control register (CERTREADEN) to enable SRAM window read access.
8. The processor then enters an infinite loop or WFI with interrupts disabled. This halts the processor.
9. The debugger polls the SRAM window area until the flag is set and then reads the identity data from SRAM.

The following example lists a sequence of events that a debugger can use to provide a certificate to the subsystem that it has generated using the identity data:

1. The SoC is reset by pulling **nSRST** LOW, and **nSRST** is held LOW to prevent the processor cores from booting.
2. The debugger then writes the certificate into the SRAM window.
3. **nSRST** is deasserted, allowing the processor to boot.
4. The processor boot code checks the SRAM window, and discovers the certificate.
5. The processor closes the direct path to the window from debugger by setting the CERTDISABLE register.
6. The processor uses the certificate to perform debug configuration, for example, changing the life cycle state of the system or configuring the debug authentication signal values.
7. The processor then continues its normal boot process.

Note

This approach is designed primarily for certificate insertion or communication during and right after the assertion of **nSRST**. It is not designed to provide a mechanism for the debugger to do the same after boot or during normal execution.

Debug element clock and power control

The Debug element is primarily running on **DEBUGSYSCLK**, except for the ATB Trace bus, which uses the faster clock, **DEBUGFCLK**. After the Debug element is turned on, **DEBUGSYSCLK** and **DEBUGFCLK** are expected to start running.

The Debug element contains a *Power Policy Unit* (PPU) that oversees the sequencing of clock, resets, and power control. The software can force the PPU to turn ON or OFF the debug element. However, the main intended use is to configure the PPU to support Dynamic transition between ON and OFF, depending on PD_DEBUG Power Q-Channel interface. This interface is expected to be used by the external *Debug Access Port* (DAP) to request for power to turn on. The interface is also for the expansion system to ensure that power is only removed when the expansion subsystem is ready for power down.

After the Debug subsystem is powered up and running, the Granular Power Requesters in the debug power domain can be used to wake each core. Waking a core core indirectly also wakes the Base element power domain.

2.8.3 Timestamps

The SSE-200 provides a single timestamp input, **TSVALUEB[63:0]**, that is used by any debug logic in the subsystem that requires a timestamp. The timestamp input is 64-bits wide and is synchronous to **DEBUGSYSCLK** and therefore synchronous also to **SYSCLK**.

The part of the subsystem on the potentially faster **FCLK** also uses this timestamp for Trace generation without interpolators to increase the granularity of the timestamp. Therefore as **FCLK** to **SYSCLK** ratio

increase, especially beyond 10:1, the timestamp granularity is degraded which reduces the ability to accurately determine traced events timings between the two clock domains.

2.8.4 Cross trigger

The Debug element implements a *Cross Trigger Matrix* (CTM) and a single *Cross Trigger Interface* (CTI) Block.

These allow timers and watchdog timers in the SSE-200 subsystem to be halted and restarted using trigger sources from any of the cores and also from trigger sources external to the subsystem from the Cross trigger channel interface. The following table shows the interconnections between the Cross trigger inputs and outputs of the CTI Block.

Table 2-9 Cross trigger interconnections

Timer	Timer address	Timer halt CTI trigger signal	Timer restart CTI trigger signal
TIMER 0	`0x4000_0000`, `0x5000_0000`	**CTITRIGOUT[0]**	**CTITRIGOUT[1]**
TIMER 1	`0x4000_1000`, `0x5000_1000`	**CTITRIGOUT[0]**	**CTITRIGOUT[1]**
DUAL TIMER	`0x4000_2000`, `0x5000_2000`	**CTITRIGOUT[0]**	**CTITRIGOUT[1]**
NON-SECURE WATCHDOG	`0x4008_1000`	**CTITRIGOUT[0]**	**CTITRIGOUT[1]**
SECURE WATCHDOG	`0x5008_1000`	**CTITRIGOUT[0]**	**CTITRIGOUT[1]**
S32KTIMER	`0x4002_F000`, `0x5002_F000`	**CTITRIGOUT[2]**	**CTITRIGOUT[3]**
S32KWATCHDOG	`0x5002_E000`	**CTITRIGOUT[2]**	**CTITRIGOUT[3]**

The SSE-200 subsystem expects the use of handshake pulse triggers for halting and restarting timers:

- When halting timers, no trigger acknowledgment is returned. Therefore, the timer halt trigger signals **CTITRIGOUT[0]** or **CTITRIGOUT[2]** remain HIGH and must be cleared individually by writing to the CTIINTACK register.
- When restarting timers using **CTITRIGOUT[1]** or **CTITRIGOUT[3]**, trigger acknowledgement is returned on their respective trigger acknowledgement signals **CTITRIGOUTACK[1]** or **CTITRIGOUTACK[3]**.

All **TODBGENSELCTI** signals of the CTI block are tied LOW. Therefore when LOW, the debug authentication signal **DBGEN** masks these triggers. This mask disables the ability of the CTI to halt and restart timers and watchdog timers.

All **TINIDENSELCTI** signals of the CTI blocks are also tied LOW. Therefore when LOW, the debug authentication signal **NIDEN** masks all triggers inputs of the CTI block.

2.8.5 CoreSight debug ROM tables

There are three CoreSight ROM tables that are implemented within the SSE-200 subsystem in addition to the processor core.

Debug system CoreSight ROM table

The debug system CoreSight ROM is only accessible from the debug system Access APB-AP. It is at address `0xF000_0000`.

The following table lists the contents of the ROM table. The PID values depend on the value of the **TARGETIDSYS[31:0]** static configuration signal, which by default is set to `0x0743_0477`.

Table 2-10 Debug system ROM table

Address offset	Value	Description
0x000	0x0000_1003	Entry points to the debug element trace funnel.
0x004	0x0000_2003	Entry points to the debug element CTI.
0x008	0x0008_0003	Entry points to an external ROM on the APB expansion region.
0x00C-0xEFC	0x0000_0000	Unused ROM entries.
0xF00-0xFC8	0x0000_0000	Reserved.
0xFCC	0x0000_0000	MEMTYPE register.
0xFD0	0x0000_0004	Peripheral ID, PIDR4. PIDR4[3:0] is the JEP106 continuation code, which is set by **TARGETIDSYS[11:8]**.
0xFD4	0x0000_0000	Peripheral ID, PIDR5
0xFD8	0x0000_0000	Peripheral ID, PIDR6
0xFDC	0x0000_0000	Peripheral ID, PIDR7
0xFE0	0x0000_0043	Peripheral ID, PIDR0. PIDR0[7:0] is the Part Number bits [7:0], which is set by **TARGETIDSYS[23:16]**.
0xFE4	0x0000_00B7	Peripheral ID, PIDR1. PIDR1[3:0] is the Part Number [11:8], which is set by **TARGETIDSYS[27:24]**. PIDR1[7:4] is the JEP106 Identity Code [3:0], which is set by **TARGETIDSYS[4:1]**.
0xFE8	0x0000_000B	Peripheral ID, PIDR2. PIDR2[2:0], is the JEP106 Identity Code [6:4], which is set by **TARGETIDSYS[7:5]**. PIDR2[3] is the JEDEC identifier. PIDR2[7:4] is the Revision Code, which is set by **TARGETIDSYS[31:28]**.
0xFEC	0x0000_0000	Peripheral ID, PIDR3
0xFF0	0x0000_000D	Component ID, CID0
0xFF4	0x0000_0010	Component ID
0xFF8	0x0000_0005	Component ID, CID2
0xFFC	0x0000_00B1	Component ID, CID3

CPU access CoreSight ROM table

There up to two CPU System CoreSight ROMs, one for each processor. Each ROM is accessible through its associated CPU Access AHB-AP. It is at address 0xF000_8000.

The following table lists the contents of the ROM table.

Table 2-11 CPU access ROM table

Address offset	Value	Description
0x000	0x0000_1003	Entry points to the Granular Power Requester.
0x004	0x0000_2003	Entry points to the internal ROM table for the processor.
0x008-0xEFC	0x0000_0000	Unused ROM entries.
0xF00-0xFC8	0x0000_0000	Reserved.

Table 2-11 CPU access ROM table (continued)

Address offset	Value	Description
0xFCC	0x0000_0000	MEMTYPE register.
0xFD0	0x0000_0004	Peripheral ID, PIDR4.
0xFD4	0x0000_0000	Peripheral ID, PIDR5.
0xFD8	0x0000_0000	Peripheral ID, PIDR6.
0xFDC	0x0000_0000	Peripheral ID, PIDR7.
0xFE0	0x0000_0043	Peripheral ID, PIDR0.
0xFE4	0x0000_00B7	Peripheral ID, PIDR1.
0xFE8	0x0000_000B	Peripheral ID, PIDR2.
0xFEC	0x0000_0000	Peripheral ID, PIDR3.
0xFF0	0x0000_000D	Component ID, CID0.
0xFF4	0x0000_0010	Component ID, CID1.
0xFF8	0x0000_0005	Component ID, CID2.
0xFFC	0x0000_00B1	Component ID, CID3.

2.9 Power control infrastructure

This section describes the power control features.

This section contains the following subsections:

2.9.1 Overview

Low-power operation is essential for IoT endpoint devices which might rely on a battery or on harvested energy. SSE-200 uses two key methods to reduce the overall power of the system:

- Dynamic hierarchal clock control to reduce dynamic power. See *2.2 Clocks* on page 2-22.
- Multiple power-gated regions in the design to reduce leakage power.

Each power domain contains a *Power Integration Kit* (PIK) that performs the following:

- Integrates a *Power Policy Unit* (PPU) that provides technology-independent power control of the domain.
- Integrates Q-Channel and P-Channel infrastructure components to bring together the quiescent status and control of key IP blocks within the power domain to the PPU.

The PIK of each domain is primarily designed to deal with the power control of its associated power domain. However, some power domains are aware of the power state of other power domains, to maintain the correct operation of the system. The *Power Dependency Control Matrix* (PDCM) enables software to configure the relationship between each power domain.

The following figure shows the power control infrastructure of SSE-200, containing multiple power domains, each with its associated PIK and how they are connected to the PDCM.

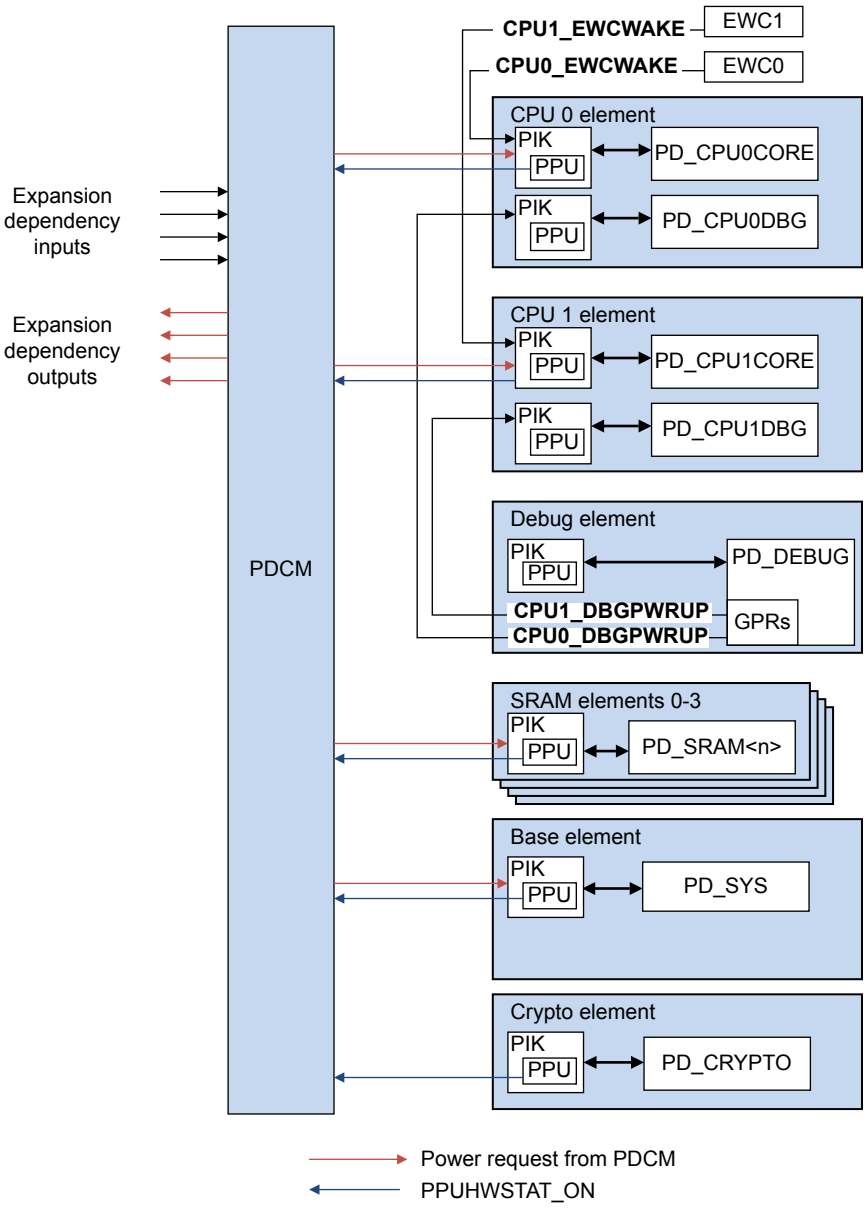

Figure 2-7 Power Dependency Control Matrix

For power-management signals, see *A.1 Clock, reset, and power control signals* on page Appx-A-146.

SSE-200 subsystem defines the following power domains:

- PD_SYS, system power domain.
- PD_DEBUG, debug power domain.
- PD_SRAM<*N*>, SRAM macros power domains.
- PD_CPU0CORE, CPU 0 core power domain.
- PD_CPU1CORE, CPU 1 core power domain (if CPU 1 is present).
- PD_CPU0DBG, CPU 0 debug power domain (if `SEPARATE_CPUDBG_PD` configuration is True).
- PD_CPU1DBG, CPU 1 debug power domain (if CPU 1 is present and if `SEPARATE_CPUDBG_PD` configuration is True).
- PD_CRYPTO, CryptoCell power domain (if CryptoCell is present, `HAS_CRYPTO` is True).
- PD_AON, which is the AON domain.

  Anything that is not in the other domains, is in the PD_AON power domain.

If a separate processor debug power domain is not supported, then PD_CPU0DBG and PD_CPU0CORE power domains are merged into PD_CPU0CORE, and PD_CPU1DBG and PD_CPU1CORE power domain are merged into PD_CPU0CORE.

The following figure shows how the power domains map onto the elements.

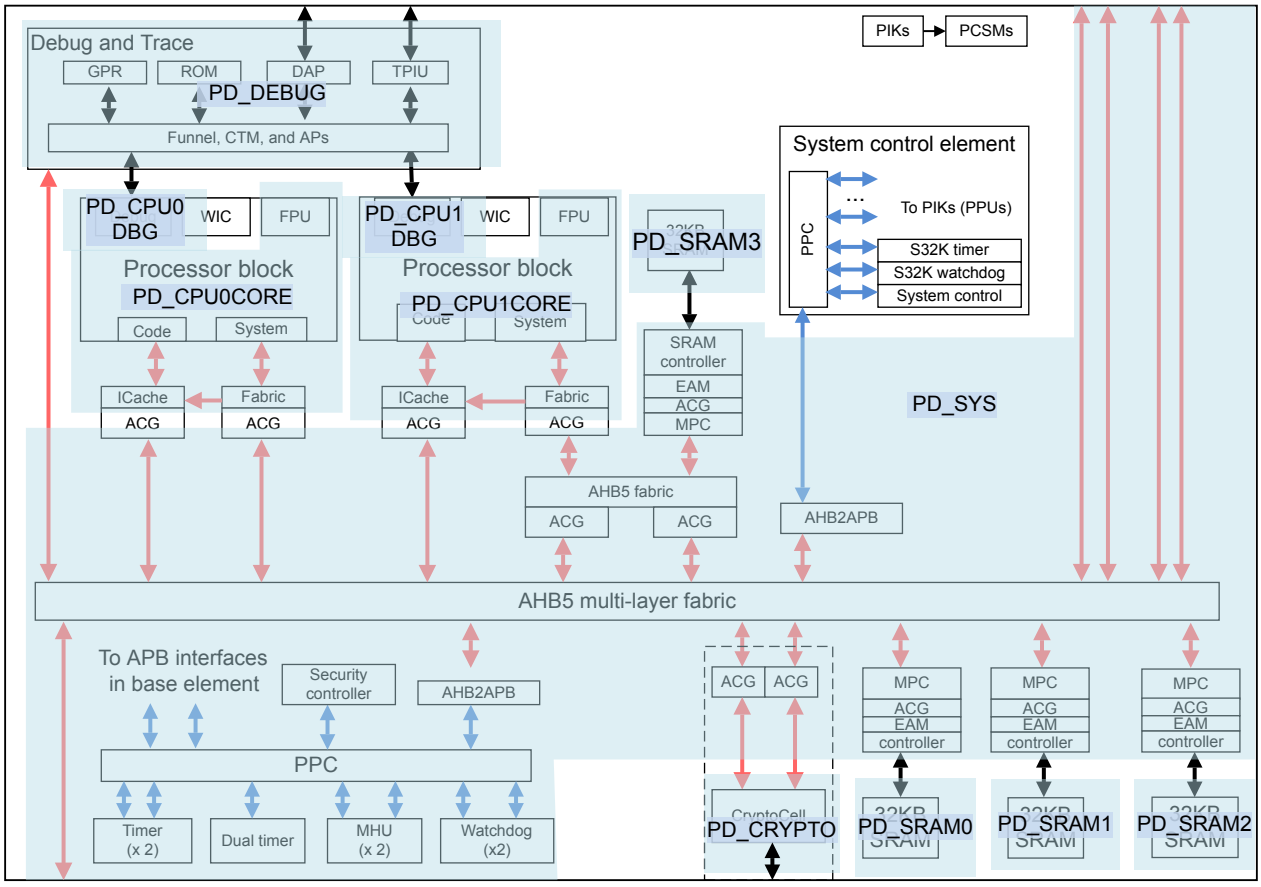

Figure 2-8 SSE-200 power domains

Related references

A.1 Clock, reset, and power control signals on page Appx-A-146.

2.9.2 Power domains and PPUs

PPUs control all power domains in the SSE-200, except for the always on PD_AON domain.

Each PPU uses the following control signals to power the domain up or down:

- All power Q-Channel interfaces from within the power domain.
- *Access Control Gates* ACG access request into power domain.
- The *Power Dependency Control Matrix* (PDCM) register settings configure all the other power dependecy control settings.
- Debug system power up requests for the domain if they exist.

The PPU allows software to do the following:

- Perform static power control. For example, turning the power domain permanently on or off, or putting the domain into a retention state.

  Caution

  If software sets any key power domains, such as, PD_CPU0_CORE, or PD_SYS, to OFF or to retention, the system might not be able to power up, boot, or execute code. This can cause a system lockup, so avoid using static power control except for PD_CRYPTO domain.

- Enable or disable dynamic power transitions and set a minimum power state. It supports a subset of the following transitions:
  - ON to OFF
  - OFF to ON
  - ON to RET (Memory only or Full Retention depending on the requirements of each domain)
  - RET to ON
  - ON to Warm reset
  - Warm reset back to ON.

Each PPU therefore uses a Q-Channel or P-Channel interface to:

- Determine if it can bring the power domain to a required static power mode programmed by software.
- If dynamic power transition is enabled, determine if the power domain must transition to a different power state.

In addition to internal Q-Channel and P-Channel, some power domains are able to perform power transition to ON using requests from the following:

- The ACG at the boundary of the power domain when an access request arrives at the boundary of the domain.
- The GPR in the debug element to power up.
- One of the external wakeup controllers.
- One of the power control Q-Channel expansion interface signals.

The following table lists the types of request that can wake each domain.

Table 2-12 Domains and wake-up requests

Power domain	Access Control Gate	GPR or DAP	EWC	Q-Channel expansion
PD_SYS	No	Yes. CPU0 GPR or CPU1 GPR.	Yes. CPU0 EWC or CPU1 EWC.	Yes
PD_CPU0DBG	No	Yes. CPU0 GPR.	Yes. CPU0 EWC.	No
PD_CPU0CORE	No	Yes. CPU0 GPR.	Yes. CPU0 EWC.	No
PD_CPU1DBG	No	Yes. CPU1 GPR	Yes. CPU0 EWC.	No
PD_CPU1CORE	No	Yes. CPU1 GPR.	Yes. CPU0 EWC.	No
PD_DEBUG	No	Yes. DAP	No	Yes
PD_SRAM0	Yes	No	No	No
PD_SRAM1	Yes	No	No	No
PD_SRAM2	Yes	No	No	No
PD_SRAM3	Yes	No	No	No
PD_CRYPTO	No	No	No	No

Note

- PD_CRYPTO only supports static power control.
- Although wake-up of PD_SYS from OFF is supported using the PD_SYS Power Control Q-Channel interface, Arm recommends an IRQ to wake the system, especially if external ACGs use **ACG_WAIT**. If an interrupt does not wake the processort, and there are no other masters able to clear the ACG_WAIT registers to unblock access into the system, the system can deadlock. Interrupts are not required for wake from Retention.
- PD_CPU0DBG and PD_CPU1DBG power domains do not exist if there is no support for a separate processor debug power domain.

2.9.3 Processor power domains

Each Cortex-M33 processor in the system can have either:

- A single power domain, PD_CPU<N>CORE, that gates both the core and the debug logic of the processor.
- Two power domains, with PD_CPU<N>CORE that gates the core, and a separate debug logic power domain, PD_CPU<N>DBG.

The Cortex-M33 core internally controls, by its Q-Channel, the power state relationship between different power domains of the core. As a result, if `SEPARATE_CPUDBG_PD` configuration is True indicating that the separate processor debug power domain is supported, then whenever PD_CPU<N>DBG power domain is ON, it requests its Q-Channel interface to turn PD_CPU<N>CORE ON. If a separate processor debug power domain is not supported (when `SEPARATE_CPUDBG_PD` configuration is False), then both debug and core exist in one power domain.Any attempt to wake the debug logic wakes the core, and any attempt to wake the core also wakes the debug logic.

2.9.4 System boot when powering up

The power mode of the PPU, along with its integration, ensures that the following domains can power up immediately after Power-on reset:

- PD_CPU0CORE.
- PD_CPU1CORE.
- PD_SYS.
- PD_CRYPTO.
- PD_SRAM0 to PD_SRAM3.

The automatic powering up of PD_CPU0CORE and PD_CPU1CORE is also dependent on the settings of the CPU0_WAIT and CPU1_WAIT respectively in CPUWAIT register:

- If a bit in CPUWAIT is set to HIGH, out of Power-on reset, **nSRST** reset, Watchdog reset, RESETREQ-based reset, or a request from the SWRESET register, the associated processor will not power up.

  This allows you to control which of the processors powerup after any of these reset conditions.
- The use of the CPUWAIT register to control powering up is in addition to the other functionality of CPUWAIT, which is to delay the boot of each associated processor after powering up or reset.
- The parameters `CPU0WAIT_RST` and `CPU1WAIT_RST set the reset values of CPU0_WAIT and CPU1_WAIT`, respectively.

AIRCR.SYSRESETREQ based reset requests do not depend on CPUWAIT.

After a core has been powered up and powered down, without reapplying a reset that depends on the settings in CPUWAIT, the CPUWAIT register is not involved in delaying the powerup of the cores.

2.9.5 External wakeup controllers

Each Cortex-M33 core in SSE-200 subsystem is configured to have a Wakeup Interrupt Controller (WIC). These WICs:

- Allow each Cortex-M33, during DEEPSLEEP, to go into a lower power mode, where the core and the NVIC clocks are turned off and optionally placed into Retention.
- Store the interrupt masks from the NVIC in the WIC, holding any outstanding interrupts.
- Raise a request to wake the core if the masks allow it.

This WIC however does not support waking the Cortex-M33 core if power to the core and the NVIC is turned OFF.

To support waking the Cortex-M33 cores from an OFF state, SSE-200 subsystem implements an External Wakeup Controller (EWC) for each processor. The EWC uses the interrupt mask that is already stored in the WIC, and holds any pulse-type interrupts on behalf of the WIC and NVIC when the processor clocks are OFF. It also wakes the processor if the masks allow it.

Before allowing a processor to move to the OFF state, with the intention to use interrupts to wake it later, you must:

1. Enable its associated WIC using the WICCTRL register.
2. Use the EWCTRL register to enable its associated EWC.
3. After the processor re-awakes and has boot to a point to be able to handle interrupts, the associated EWC<N>EN_CLR must then be used to clear the EWC.

Note

- If a processor is powered down without enabling its associated EWC, interrupts cannot power up the processor. To force a processor in this state to power up, you must either:
  - Write to the PPU and set its power policy register to ON.
  - Use the external debugger to wake the processor with the associated GPR.

  This feature allows you to more permanently leave the secondary core powered off. The feature still allows you to bring the secondary core back into use if necessary, later.
- The EWC does not deal with the event interface and therefore does not support waking the core from an OFF state using events.
- If any EWC wakes its associated processor, it also wakes the main system (PD_SYS).

2.9.6 Power Dependency Control Matrix

The SSE-200 subsystem defines a control matrix that allows the power mode (ON state) of one domain to affect another power domain.

The following table shows how this matrix is structured.

Table 2-13 PDCM structure

Domain	PD_SYS	PD_SRAM0	PD_SRAM1	PD_SRAM2	PD_SRAM3
PD_SYS PPUHWSTATE_ON	Conf	Conf	Conf	Conf	Conf
PD_CPU0CORE PPUHWSTATE_ON	Yes	Conf	Conf	Conf	Conf
PD_CPU1CORE PPUHWSTATE_ON	Yes	Conf	Conf	Conf	Conf
PD_SRAM0 PPUHWSTATE_ON	Yes	Conf	-	-	-
PD_SRAM1 PPUHWSTATE_ON	Yes	-	Conf	-	-
PD_SRAM2 PPUHWSTATE_ON	Yes	-	-	Conf	-
PD_SRAM3 PPUHWSTATE_ON	Yes	-	-	-	Conf
PD_CRYPTO PPUHWSTATE_ON	Yes	-	-	-	-
PDEXP0	Conf	Conf	Conf	Conf	Conf
PDEXP1	Conf	Conf	Conf	Conf	Conf

Table 2-13 PDCM structure (continued)

Domain	PD_SYS	PD_SRAM0	PD_SRAM1	PD_SRAM2	PD_SRAM3
PDEXP2	Conf	Conf	Conf	Conf	Conf
PDEXP3	Conf	Conf	Conf	Conf	Conf

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

- The ON state of each power domain in the system
- Expansion power control dependency signals, **PDEXP0** to **PDEXP3**, that are driven by expansion logic outside the subsystem indicating the ON state of external power domains.

If a power domain is sensitive to a dependency input, after the controlled power domain is ON, the power domain remains ON while any of the dependency inputs is HIGH. Therefore the PDCM is used primarily to define when a power domain must not enter a lower power state. It is not designed to support powering up of any power domain.

The power control matrix interactions have the following features:

- For the PD_SYS power domain:
  - It can be requested to turn ON either by any GPR, by any EWC or by its associated Power Q-Channel Interface. PD_SYS remains ON when any of the other power domains in the system not related to debug, are ON.
  - PD_SYS can be configured to remain ON when any of the Expansion Power Dependency inputs are ON.
  - If PD_SYS is configured to be sensitive to itself, then when it is ON, it stays ON.
- Each processor power domain can be requested to turn ON either by its associated GPR or EWC.
- PD_DEBUG can be request to turn on by the debug DAP or by its associated power Q-Channel interface for each SRAM power domain.
- For any PD_SRAM<n>:
  - Each can be woken by sending access to the SRAM.
  - Each can be configured to remain ON depending on the CPU power state, the PS_SYS power state, or any of the Expansion Power Dependency inputs are ON.
  - If each is configured to be sensitive to itself, then when it is ON, it stays ON.

The PDCM and sensitivity inputs for each power domain allows for the power control of the system to be primarily performed using only dynamic power transitions. This reduces the number of software interactionsthat are required for system management and therefore improves its responsiveness and contributes to further power reduction.

2.9.7 System and processor power states

Each power domain is relatively independent, because of the relationship that is defined in the PDCM and by other request signals. However, it is possible to define several of sensible system level power states that the system as a whole supports.

Table 2-14 System and processor power states

System power state	PD_SYS	PD_ CPU<N>CORE	PD_CPU <N>DBG[a]	PD_ DEBUG	PD_ CRYPTO	PD_ SRAM<N>	Main clock[b]
OFF	OFF	OFF[c]	OFF	OFF	OFF	OFF	OFF
HIBERNATION	OFF	OFF[d]	OFF	OFF/ON	OFF	OFF/ MEM_RET	OFF/ON[e]
		OFF – DEEPSLEEP WITH WIC + EWC					
SYS_RET	RET	OFF[f]	OFF	OFF/ON	OFF	OFF/ MEM_RET	ON
		OFF – DEEPSLEEP WITH WIC + EWC					
		RET[g] – DEEPSLEEP WITH WIC					
SYS_ON	ON	OFF[f]	OFF	OFF/ON	OFF/ON	ON/OFF/ MEM_RET	ON
		OFF – DEEPSLEEP WITH WIC + EWC					
		RET[gh] – DEEPSLEEP WITH WIC					
		ON - DEEPSLEEP WITH WIC	ON/OFF				
		ON - SLEEP					
		ON					

Note

All domains, including the PD_SYS and PD_CPU<N>CORE power domains, are independent power domains with relationships that are enforced by the PDCM.

When any PD_CPU<N>CORE is turned ON, the PD_SYS also turns ON. PD_SYS might take longer to turn ON compared to the CPU, therefore the system might be in a temporary state where the PD_SYS is OFF while PD_CPU<N>CORE is ON.

The implementer must ensure that during physical implementation this is considered, especially when dealing with signal isolation.

2.9.8 Entering lower processor power states

For each processor to enter a lower power state, the processor must enter SLEEP or DEEPSLEEP state.

a The PD_CPU<N>DBG column is not used if there is no support for separate processor debug power domain (if `SEPARATE_CPUDBG_PD` configuration is False).
b Main clock is active as requested by the subsystem. Exclude activity request with the **EXPCLKREQ** and **EXPCLKRDY** signals.
c All CPUs must be OFF.
d This entry is similar to footnote e except that at least one core is expected to be in OFF – DEEPSLEEP WITH WIC + EWC state. If all cores are in the OFF state without enabling EWC, then expansion hardware must request the system to power up using the PD_SYS Power Control Q-Channel interface or using the GPR of the Debug element.
e When in HIBERNATION state, if PD_DEBUG is ON, then **MAINCLK** is running. Otherwise **MAINCLK** is OFF.
f These entries allow one or more of the processors to be turned off and not used. A processor in this state cannot be woken using its interrupts and instead can only be woken by the PDCM, or by another entity writing to the PPU for the processor if the system is in the SYS_ON state.
g Logic Retention support for PD_SYS and PD_CPU<M>CORE are configuration options through the `CPU_SYS_RETENTION` renter configuration. If logic retention is not supported, SYS_RET System Power State does not exist, and RET– DeepSleep with WIC power state of any of the PD_CPU<N>CORE also do not exist.
h If Logic Retention is not supported (when `CPU_SYS_RETENTION` configuration is False), any attempt to enter this state, and therefore enter retention with the processor, results in the processor remaining ON, giving the impression that the processor state is still being retained.

When entering DEEPSLEEP, different combinations of WIC and EWC enable then determine the different processor power state being entered. The following shows the key power states of the processor that the subsystem supports:

- When the PD_CPU<N>CORE is entering the OFF – DEEPSLEEP WITH WIC + EWC state from the ON state, the processor must first enable its WIC, followed by handshaking to enable its associated EWC, then enabling DEEPSLEEP before entering WFI.

  This is the only state that actually supports processor power being fully turned off with the ability to wake thecore from interrupts.
- When the PD_CPU<N>CORE is entering the OFF – DEEPSLEEP WITH WIC state, RET – DEEPSLEEP WITH WIC or ON – DEEPSLEEP WITH WIC from the ON state, the processor must first enable its WIC, then enable DEEPSLEEP before entering WFI.

  DEEPSLEEP WITH WIC can only support the core in retention, or ON with simply the core and NVIC clock turned OFF.
- When the PD_CPU<N>CORE is entering the OFF state, WIC and EWC are not enabled before the processor enters WFI.

  OFF state means that the processor cannot be woken from interrupts, and the only way to wake the processor is by writing to the PPUs to manually turn it ON.
- When the PD_CPU<N>CORE power domain is entering the ON – SLEEP state from the ON state, the processor must first enable its WIC, and enable SLEEP before entering WFI.

  When in CPU SLEEP, the processor is still ON and the NVIC is clocking with the rest of the core clock turned off.

Note

- The EWC does not support waking the processor with events.
- WFE must not be used with WIC and EWC to enter the processor into OFF state if the intention is to wake using the Event interface of the processor.
- Because the FPU is now part of the processor main core power domain, before the processor is able to enter an OFF state, the FPU must be made unavailable by both setting the CP10 field of the CPACR Register in the processor to `0b00` and setting the SU10 field of CPPWR Register in the processor to 1.
- If the FPU in the processor is not disabled, and the processor is allowed to start entering an OFF state without enabling the WID or EWC, the processor could deadlock. To protect against deadlock when the controls are not set correctly, enable PPU to raise an interrupt when an attempt to dynamically enter a lower power state is denied by the processor. This interrupt brings the processor out of WFI.

2.9.9 Hibernation

Hibernation state is the lowest power state of the system that SSE-200 supports. When in this state, most of the system that supports power gating is turned off, except for PD_SRAM<n> in either OFF or MEM_RET, and the PD_DEBUG in either ON or OFF. In this state, normally the **MAINCLK** is also turned off to save power.

To enter the hibernation state, the following conditions must be met:

- All PD_SRAM<n> domains must be configured to dynamically enter OFF or Memory Retention state.
- Any SRAM that is expected to be used immediately after exiting hibernation, for example, for holding stack pointers, must either:
  - Have its PDCM configuration sensitivity, before entering hibernation, set to be sensitive to any of the PD_CPU<n>CORE power ON states that is associated to the core that wakes and uses that SRAM. This ensures that when the SRAM powers up after leaving hibernation, it stays powered until the associated processor turns OFF.
  - Set that SRAM lowest power state to retention so that the SRAM content is always retained even if the SRAM turns off momentarily while waiting for the processor to start accessing memory.

This ensures that after the SRAM powers up at first assess after leaving hibernation, it stays powered until the system or its associated processor turns OFF.

- All GPR in the debug domain are turned off. Otherwise, one or more CPUs cannot power down.
- If you have expansion logic that is interfaced to the PD_SYS power control Q-Channel interface, ensure that the expansion logic is not active and can enter a quiescent state.
- If you have other expansion power domains in the system that are expected to stay ON or to be turned on and starting accesses when the system is still in hibernation, you must make some changes during the integration phase for the expansion region.

  You must provide gates that use the **ACCWAIT** signal to temporarily block access from these regions.

  You must set `ACC_WAITN_RST` parameter to LOW. This ensures that when the system restarts, any access from these power domains are held off until the security configurations of the system can be reinstated. After security considerations have been reinstated, Secure software can set the ACCWAIT register flags to allow access from these power domains to proceed.
- If the intention is to wake the system with an EWC request, all processors must enter OFF state, with at least one processor in the DEEPSLEEP WITH WIC and EWC enabled. Or wake the system with expansion logic that is able to drive the PD_SYS power control Q-Channel interface.

The system leaves the Hibernation state when a processor is woken from the OFF state. When an EWC request is made to wake a CPU, it also requests **MAINCLK** to start running. After the processor has booted, and has restored the Security state, the processor must set the ACC_WAITN register to allow all other masters to access the system.

2.9.10 Wake From Hibernation using PD_SYS power control Q-Channel interface

Along with EWC, expansion blocks outside the subsystem can use the PD_SYS power control Q-Channel interface to wake the system from hibernation.

Waking the system using the interface does not however directly wake the cores or any other domains in this system unless accesses to the system results in a wake interrupt on an EWC, or an SRAM wake request, while accessing SRAM.

Note

- The system **SYSCLK**, and therefore also the **MAINCLK**, coming into the system might not be running during hibernation. A wake request on **SYSPWRQACTIVE** input automatically results in a request for **MAINCLK** to be active so that the Q-Channel handshake can be performed.
- When the base system in the PD_SYS power domain wakes from hibernation, all registers in the domain are in reset state and all peripherals that reside behind the PPC and MPC default to Secure access only. There is typically a requirement to also wake and boot a processor to configure the system before allowing access for other masters to the system.

  Arm recommends that you do not depend on the PD_SYS power control Q-Channel interface to wake the system from hibernation and instead use interrupt signals on the EWC. This allows a request to wake a core which then in turn wakes the system and configures it before allowing other masters to access it.

  To delay access from other masters in the system, you must also deploy access control gates at slave expansion interfaces of the base system. The subsystem provides the ACC_WAITN register and the **ACCWAITn** signal to control access.
- If you require the ability to wake and access the base system from the expansion interfaces without also waking a core, you must ensure the master accessing the system is a Secure master. Alternatively, you can allow a Non-secure master strict access to a control region of memory that you are sure is Non-secure and does not pose a security risk.

  When the system wakes without a processor restoring the configuration of all MPCs and PPCs settings in the system, your Secure master sees all Non-secure memory space as Secure memory space.

You must ensure that if any of these Non-secure memories support retentions, these memory locations are not used for code execution.

2.10 Crypto element

The Crypto element provides the following features:

- 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 the user itself. These asset protection features include:
  - Image verification at boot or during runtime.
  - Authenticated debug.
  - *True Random Number Generation* (TRNG).
  - Lifecycle management.
  - Provisioning of assets.

The CryptoCell-312 implements several key interfaces that are visible to software:

- Two APB4 interfaces, and both interface to the base system through Access Control gates and reside at the following address in the main memory map:
  - APB Configuration Interface at aliased address regions `0x4008_8000-0x4008_BFFF` and `0x5008_8000-0x5008_BFFF`. This interface provides access to programmer visible registers within CryptoCell-312, and CryptoCell-312 itself handles security checking of accesses to its registers on its own.
  - APB Code Interface at aliased addresses `0x0E00_0000-0x0E00_1FFF` and `0x1E00_0000-0x1E00_1FFF`. This interface provides access to the NVM memory, with word address of `0x00A0-0x1FFC`. Note the address offset of `0xA0` being applied to the APB address.
- A single AHB bus master interface that connects to the base system using an ACG, with access only to the following address spaces:
  - Code AHB5 Master Expansion Interfaces, at addresses `0x0000_0000-0x0DFF_FFFF` and `0x1000_0000-0x1DFF_FFFF`.
  - All implemented SRAM blocks with the areas `0x2000_0000-0x20FF_FFFF` and `0x3000_0000-0x30FF_FFFF`.
  - AHB5 Master Expansion Interface 0, at addresses `0x2800_0000-0x2FFF_FFFF`, `0x3800_0000-0x3FFF_FFFF`, and `0x6000_0000-0x7FFF_FFFF`.

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

Note

You must have a license for the CryptoCell-312 IP to access the product documentation.

This section contains the following subsection:

- *2.10.1 Persistent storage* on page 2-70.

2.10.1 Persistent storage

If the Crypto element exists, then the SSE-200 implements a Persistent State Storage Block that resides in the PD_AON power domain. It implements registers that store key state information on behalf of CryptoCell.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Configuration and Integration Manual* for more information.

Warm reset interactions

The Crypto element in general is reset by both Warm reset and Power-up reset.

However, the persistent state storage values are reset to zeros only by Power-up reset. If a system Warm reset is triggered, the contents in the persistent state storage are not cleared. CryptoCell, when restarting, then discovers that the persistent state values are populated and does not try to regenerate it from the

OTP values. Therefore, if trusted firmware decides to lock fields in the persistent state storage, it is not possible to unlock it through Warm reset. This can cause problems at boot.

In addition, since Warm reset conceptually can be triggered at any time, it could interfere with, and cause corruption of data during writing of the persistent state storage.

Warm reset can only be triggered by each processor writing to its AIRCR.SYSRESETREQ control. The hardware requests are then gated by the SYSRSTREQ0_EN and SYSRSTREQ1_EN values in the RESET_MASK register. To avoid problems with persistent state storage, by default, these gating values are set to LOW using the input parameters `SYSRSTREQ0_EN_RST` and `SYSRSTREQ1_EN_RST` to disable Warm reset. Secure firmware can allow it at a later stage after checking that it is safe to allow Warm reset, for example, if:

- There are no persistent state storage values set that causes problems during system reboot.
- There are no expected changes occurring in the persistent state storage.

Chapter 3
Programmers Model

This chapter describes the SSE-200 memory regions and registers, and provides information on how to program a SoC that contains an implementation of the SSE-200.

It contains the following sections:

3.1 About the programmers model

The following information applies to all registers:

- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in unpredictable behavior.
- Unless otherwise stated in the accompanying text:
  - Do not modify undefined register bits.
  - Ignore undefined register bits on reads.
  - Unless otherwise specified, all register bits are reset to a logic 0 by a system or power up reset.
- The following describes the access type:

RW	Read and write.
RO	Read-only.
WO	Write-only.

3.2 Memory map

This section describes the memory maps for the SSE-200.

This section contains the following subsections:

3.2.1 Memory map overview

An overview of the SSE-200 system memory map is shown in the following figure:

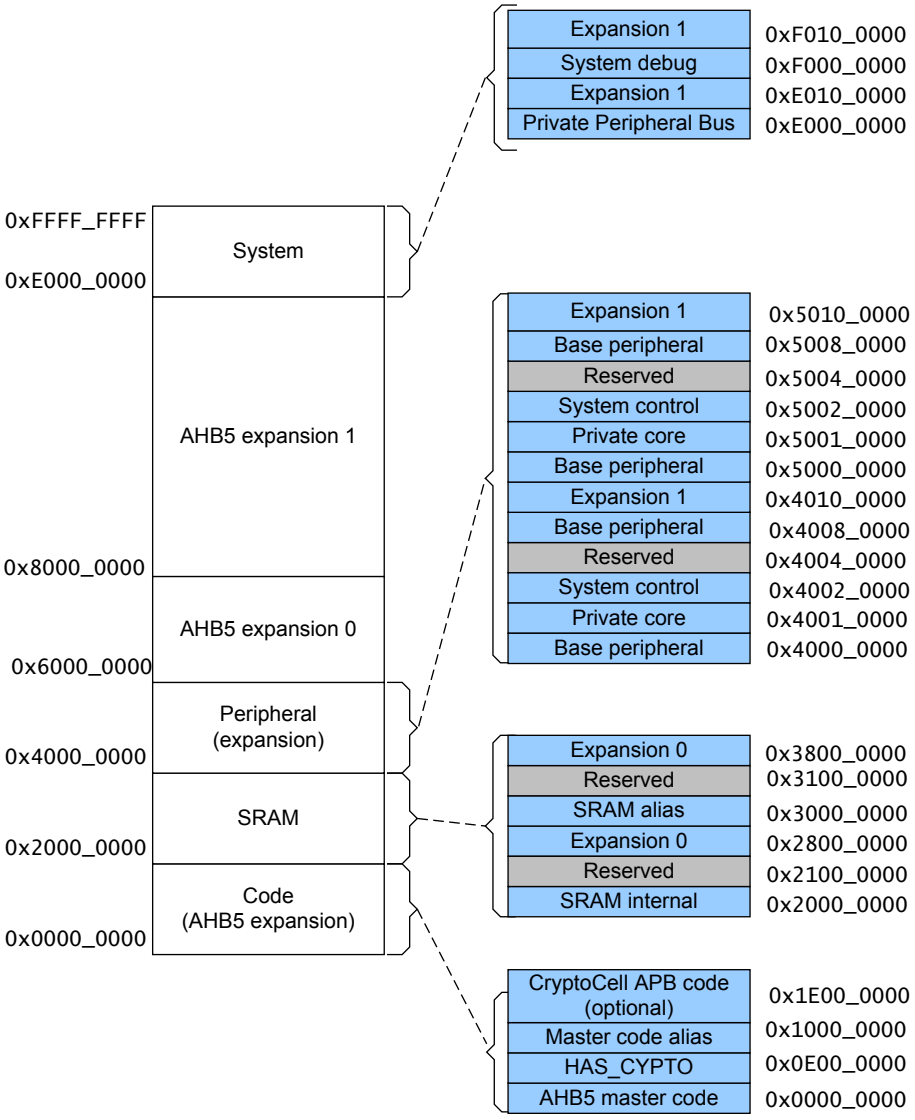

Figure 3-1 SSE-200 top-level memory map

The following table shows the high-level view of the memory map that is defined by SSE-200. 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 that are 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 following table. Software can then choose to allocate each memory block or peripheral as Secure or Non-secure using protection controllers. The *Implementation Define Attribution Unit* (IDAU) region column in the table specifies Security for each area along with ID and *Non-Secure Callable* (NSC) settings for each region. Except when stated, all access to unmapped regions of the memory result in a bus-error response. An exception to that is when accessing unmapped address space within a region taken by a peripheral. The access is *Read As Zero and Write Ignored* (RAZ/WI). Any accesses that result in security violations, either RAZ/WI or result in a bus error response as defined by the SECRESPCFG register setting. Some regions of memory map are reserved to maintain compatibility with future subsystems. Other areas are mapped to AHB Master Expansion 0 interface and AHB Master Expansion 1 interfaces. All accesses targeting populated SRAM regions within `0x2000_0000` to `0x20FF_FFFF` and `0x3000_0000` to `0x30FF_FFFF` support exclusive accesses since they implement exclusive access monitoring, provided the accesses are from:

- The processors.
- Expansion masters that have their corresponding `EXP_SYS_ID_PRESENT` bit matching the Master ID set to HIGH.

Exclusive accesses are not supported for other regions implementing within the subsystem. For regions that reside in user expansion areas, the user expansion logic defines exclusive access support.

Note

If an exclusive access targets regions that do not support exclusive accesses, or have Master IDs that do not have the corresponding `EXP_SYS_ID_PRESENT` bit set to 1, these accesses are not monitored for exclusive access and might still update their target memory locations regardless of their associated exclusive responses.

The following table lists the main regions in the memory map.

Table 3-1 Memory map overview

ID (alias)	Address		Size	Region name	Description	IDAU security[a]	IDAUID	IDAU NSC
	From	To						
1 (4)	`0x0000_0000`	`0x0DFF_FFFF`	224MB	Code Memory[b]	Maps to AHB5 master expansion code interface.	NS	0	0
2 (5)	`0x0E00_0000`	`0x0E00_1FFF`	8KB	NVM code[c]	CryptoCell APB code interface for NVM.			
3	`0x0E00_2000`	`0x0FFF_FFFF`	-	Reserved	Reserved.			
4 (1)	`0x1000_0000`	`0x1DFF_FFFF`	224MB	Code Memory[b]	Maps to AHB5 master expansion code interface.	S	1	CODENSC[d]
5 (2)	`0x1E00_0000`	`0x1E00_1FFF`	8KB	NVM code	CryptoCell APB code interface for NVM.	S		
6	`0x1E00_2000`	`0x1FFF_FFFF`	-	Reserved	Reserved.			

[a] The IDAU security value does not define privileged or unprivileged accessibility. That is defined by the MPC, PPC, or register blocks mapped to each area.
[b] Even though these regions are not aliased at the interface, they are expected to be aliased in the expansion system to support Non-secure and Secure shared code memory. In addition you must use Memory Protection Controllers externally to selectively map each block of memory between Secure and Non-secure memory regions.
[c] This region is reserved and responds with bus error if Crypto element does not exist.
[d] The IDAU NSC values are defined by registers in the Secure Privilege Control registers.

Table 3-1 Memory map overview (continued)

ID (alias)	Address		Size	Region name	Description	IDAU security[a]	IDAUID	IDAU NSC
	From	To						
7 (10)	0x2000_0000	0x20FF_FFFF	16MB	Internal SRAM	Internal SRAM area.	NS	2	0
8	0x2100_0000	0x27FF_FFFF	112MB	Reserved	Reserved.			
9	0x2800_0000	0x2FFF_FFFF	128MB	Expansion 0	Maps to AHB5 master expansion 0 interface.			
10 (7)	0x3000_0000	0x30FF_FFFF	16MB	Internal SRAM	Internal SRAM Area.	S	3	RAMNSC[d]
11	0x3100_0000	0x37FF_FFFF	112MB	Reserved	Reserved.			
12	0x3800_0000	0x3FFF_FFFF	128MB	Expansion 0	Maps to AHB5 master expansion 0 interface.			
13	0x4000_0000	0x4000_FFFF	64KB	Base Peripheral	Base element peripheral region.	NS	4	0
14 (21)	0x4001_0000	0x4001_FFFF	64KB	Private CPU	CPU element peripheral region.			
15 (22)	0x4002_0000	0x4003_FFFF	128KB	System Control	System Control element peripheral region.			
16	0x4004_0000	0x4004_FFFF		Reserved	Reserved.			
17	0x4005_0000	0x4007_FFFF		Reserved	Reserved.			
18	0x4008_0000	0x400F_FFFF	512KB	Base Peripheral	Base element peripheral region.			
19	0x4010_0000	0x4FFF_FFFF	255MB	Expansion 1	Maps to AHB5 master expansion 1 interface.			
20	0x5000_0000	0x5000_FFFF	64KB	Base Peripheral	Base element peripheral region.	S	5	0
21 (14)	0x5001_0000	0x5001_FFFF	64KB	Private CPU	CPU element peripheral region.			
22 (15)	0x5002_0000	0x5003_FFFF	128KB	System Control	System Control element peripheral region.			
23	0x5004_0000	0x5000_FFFF		Reserved	Reserved.			
24	0x5001_0000	0x5007_FFFF		Reserved	Reserved.			
25	0x5008_0000	0x500F_FFFF	512KB	Base Peripheral	Base element peripheral region.			
26	0x5010_0000	0x5FFF_FFFF	255MB	Expansion 1	Maps to AHB5 master expansion 1 interface.			
27	0x6000_0000	0x6FFF_FFFF	256MB	Expansion 0	Maps to AHB5 master expansion 0 interface.	NS	6	0
28	0x7000_0000	0x7FFF_FFFF	256MB	Expansion 0	Maps to AHB5 master expansion 0 interface.	S	7	0

[a] The IDAU security value does not define privileged or unprivileged accessibility. That is defined by the MPC, PPC, or register blocks mapped to each area.

Table 3-1 Memory map overview (continued)

ID (alias)	Address		Size	Region name	Description	IDAU security[a]	IDAUID	IDAU NSC
	From	To						
29	0x8000_0000	0x8FFF_FFFF	256MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	NS	8	0
30	0x9000_0000	0x9FFF_FFFF	256MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	S	9	0
31	0xA000_0000	0xAFFF_FFFF	256MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	NS	A	0
32	0xB000_0000	0xBFFF_FFFF	256MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	S	B	0
33	0xC000_0000	0xCFFF_FFFF	256MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	NS	C	0
34	0xD000_0000	0xDFFF_FFFF	256MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	S	D	0
35	0xE000_0000	0XE00F_FFFF	1MB	PPB	Private Peripheral Bus. Local to Each processor.	Exempt		
36	0xE010_0000	0XEFFF_FFFF	255MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	NS	E	0
37	0xF000_0000	0XF00F_FFFF	1MB	System debug	System debug.	Exempt		
38	0xF010_0000	0XFFFF_FFFF	255MB	Expansion 1	Maps to AHB5 master expansion 1 interface.	S	F	0

3.2.2 SRAM overview

The subsystem supports four SRAM elements.

All SRAMs are of the same size and all SRAMs form a contiguous area of memory. Collectively they are mapped into both the Secure and Non-secure regions. The remainder of the regions are reserved. A memory protection controller then determines how the memory locations within the SRAM are mapped to the Secure and Non-secure regions.

The following table shows an example configuration of four memory banks of 32KB each.

Table 3-2 Internal SRAM regions

ID (alias)	Address		Size	Region name	Description	Security
From	To					
1 (6)	0x2000_0000	0x2000_7FFF	32KB	SRAM Bank 0	Maps to Internal SRAM Bank 0.	NS-MPC
2 (7)	0x2000_8000	0x2000_FFFF	32KB	SRAM Bank 1	Maps to Internal SRAM Bank 1.	NS-MPC
3 (8)	0x2001_0000	0x2001_7FFF	32KB	SRAM Bank 2	Maps to Internal SRAM Bank 2.	NS-MPC
4 (9)	0x2001_8000	0x2001_FFFF	32KB	SRAM Bank 3	Maps to Internal SRAM Bank 3.	NS-MPC
5	0x2002_0000	0x20FF_FFFF	-	Reserved	Reserved.	-
6 (1)	0x3000_0000	0x3000_7FFF	32KB	SRAM Bank 0	Maps to Internal SRAM Bank 0.	S-MPC

[a] The IDAU security value does not define privileged or unprivileged accessibility. That is defined by the MPC, PPC, or register blocks mapped to each area.

Table 3-2 Internal SRAM regions (continued)

ID (alias)	Address		Size	Region name	Description	Security
7 (2)	0x3000_8000	0x3000_FFFF	32KB	SRAM Bank 1	Maps to Internal SRAM Bank 1.	S-MPC
8 (3)	0x3001_0000	0x3001_7FFF	32KB	SRAM Bank 2	Maps to Internal SRAM Bank 2.	S-MPC
9 (4)	0x3001_8000	0x3001_FFFF	32KB	SRAM Bank 3	Maps to Internal SRAM Bank 3.	S-MPC
10	0x3002_0000	0x30FF_FFFF	-	Reserved	Reserved.	-

Note

- For NS-MPC, any Secure access targeting this region is blocked. An MPC controls Non-secure access to this region.
- For S-MPC, any Non-secure access targeting this region is blocked. An MPC controls Secure access targeting this region.

3.2.3 Processor overview

This section describes memory and registers associated with the Cortex-M33 processors.

Private CPU regions

Each of the two processor elements in the SSE-200 can only see its own implementation of the Private CPU Region within the CPU element.

This region consists of a Secure region from 0x4001_000-0x4001_FFFF, and a Non-secure region from 0x5001_0000-0x05001_FFFF, as shown in the following table.

Table 3-3 Private core regions

ID (alias)	Address		Size	Region name	Description	Security
	From	To				
-	0x4001_0000	0x4001_EFFF	-	Reserved	Reserved.	-
1 (4)	0x4001_F000	0x4001_FFFF	4KB	CPU_IDENTITY	CPU Identity Unit.	NS
2	0x5001_0000	0x5001_0FFF	4KB	ICACHE	Local instruction cache.	SP
3	0x5001_1000	0x5001_1FFF	4KB	CPUSECCTRL	CPU Local Security Control.	SP
-	0x5001_2000	0x5001_EFFF	-	Reserved	Reserved.	-
4 (1)	0x5001_F000	0x5001_FFFF	4KB	CPU_IDENTITY	CPU Identity Unit.	S

Note

- These regions are not accessible from any other master in the system, including the expansion slave interfaces. An external debugger can however access the regions through the Debug AHB access interface.
- Only 32-bit writes are supported.
- NS indicates Non-secure access only.
- SP indicates Secure privilege access only.
- S indicates Secure access only.
- For CPU_IDENTITY, both Secure and Non-Secure areas are always accessible. Any writes access to it is ignored.

PPB regions

The Private Peripheral Bus (PPB) provides access to the internal processor resources.

See *3.3.6 PPB regions* on page 3-89.

Related references

3.3.6 PPB regions on page 3-89.

3.2.4 Base peripheral overview

Base peripheral region is where most peripherals within the subsystem reside. There are four regions in total, two Secure and two Non-secure regions:

- `0x4000_0000-0x4000_FFFF` which is a Non-secure region.
- `0x4008_0000-0x400F_FFFF` which is a Non-secure region.
- `0x5000_0000-0x5000_FFFF` which is a Secure region.
- `0x5008_0000-0x500F_FFFF` which is a Secure region.

Some peripherals in the base element are aliased to both Secure and Non-secure regions. The final mapping to both the Secure or Non-secure regions, and Privileged or Non-Privileged access support is determined by *Peripheral Protection Controller*s (PPCs).

Table 3-4 Base peripheral regions

ID (alias)	Address		Size	Region name	Description	Security
	From	To				
1 (10)	`0x4000_0000`	`0x4000_0FFF`	4KB	TIMER 0	CMSDK Timer.	NS-PPC
2 (11)	`0x4000_1000`	`0x4000_1FFF`	4KB	TIMER 1	CMSDK Timer.	NS-PPC
3 (12)	`0x4000_2000`	`0x4000_2FFF`	4KB	DUAL TIMER	CMSDK Dual Timer.	NS-PPC
4 (13)	`0x4000_3000`	`0x4000_3FFF`	4KB	MHU 0	Message Handling Unit 0.	NS-PPC
5 (14)	`0x4000_4000`	`0x4000_4FFF`	4KB	MHU 1	Message Handling Unit 1.	NS-PPC
-	`0x4000_5000`	`0x4000_FFFF`	-	-	RAZ/WI.	-
6	`0x4008_0000`	`0x4008_0FFF`	4KB	NSPCTRL	Non-secure Privilege Control Block.	NSP
7	`0x4008_1000`	`0x4008_1FFF`	4KB	NON-SECURE WATCHDOG	Non-secure CMSDK Watchdog.	NSP
-	`0x4008_2000`	`0x4008_7FFF`	-	Reserved	Reserved.	-
8	`0x4008_8000`	`0x4009_BFFF`	16KB	CryptoCell	CryptoCell-312 (if present)[a]	NS
-	`0x4009_C000`	`0x400F_FFFF`	-	Reserved	Reserved.	-
10 (21)	`0x5000_0000`	`0x5000_0FFF`	4KB	TIMER 0	CMSDK Timer.	S-PPC
11 (22)	`0x5000_1000`	`0x5000_1FFF`	4KB	TIMER 1	CMSDK Timer.	S-PPC
12 (3)	`0x5000_2000`	`0x5000_2FFF`	4KB	DUAL TIMER	CMSDK Dual Timer.	S-PPC
13 (4)	`0x5000_3000`	`0x5000_3FFF`	4KB	MHU 0	Message Handling Unit 0.	S-PPC
14 (5)	`0x5000_4000`	`0x5000_4FFF`	4KB	MHU 1	Message Handling Unit 1.	S-PPC
	`0x5000_5000`	`0x5000_FFFF`	-	-	RAZ/WI	-
15	`0x5008_0000`	`0x5008_0FFF`	4KB	SPCTRL	Secure Privilege Control Block.	SP
16	`0x5008_1000`	`0x5008_1FFF`	4KB	SECURE WATCHDOG	Secure CMSDK Watchdog.	SP

[a] These regions only exist if the Crypto element is present (when HAS_CRYPTO is True). Otherwise, these regions are reserved and when accessed they respond with bus error.

Table 3-4 Base peripheral regions (continued)

ID (alias)	Address		Size	Region name	Description	Security
	From	To				
	0x5008_2000	0x5008_2FFF	-	Reserved	Reserved.	
17	0x5008_3000	0x5008_3FFF	4KB	SRAM0MPC	SRAM 0 Memory Protection Controller.	SP
18	0x5008_4000	0x5008_4FFF	4KB	SRAM1MPC	SRAM 1 Memory Protection Controller.	SP
19	0x5008_5000	0x5008_5FFF	4KB	SRAM2MPC	SRAM 2 Memory Protection Controller.	SP
20	0x5008_6000	0x5008_6FFF	4KB	SRAM3MPC	SRAM 3 Memory Protection Controller.	SP
	0x5008_7000	0x5008_7FFF	-	-	RAZ/WI	-
21 (8)	0x5008_8000	0x5008_BFFF	16KB	CryptoCell	CryptoCell-312 (if present)[a]	S
	0x5008_C000	0x5008_FFFF	4KB	-	Reserved.	-
22	0x5009_0000	0x500F_FFFF	64KB	-	Reserved.	-

Note

- For NS_PPC, any Secure access targeting this region is blocked. A PPC controls Non-secure access to this region.
- For S_PPC, any Non-secure access targeting this region is blocked. A PPC controls Secure access targeting this region.
- NSP indicates Non-secure private access only.
- SP indicates Secure privilege access only.
- S indicates Secure access only.

3.2.5 System control overview

The System Control region contains the peripherals in the System Control element. The System control region occupies two areas:

- 0x4002_0000-0x4003_FFFF, which is Non-secure.
- 0x5002_0000-0x5003_FFFF, which is Secure.

The following table shows the System control regions.

Table 3-5 System control regions

Row ID (alias)	Address		Size	Region name	Description	Security
	From	To				
1 (5)	0x4002_0000	0x4002_0FFF	4KB	SYSINFO	System Information Registers Block.	NS
2	0x4002_1000	0x4002_EFFF	-	-	Reserved, RAZ/WI.	-
3 (18)	0x4002_F000	0x4002_FFFF	4KB	S32KTIMER	CMSDK Timer running on **S32KCLK**.	NS-PPC
4	0x4003_0000	0x4003_FFFF	-	-	Reserved.	-
5 (1)	0x5002_0000	0x5002_0FFF	4KB	SYSINFO	System Information Registers Block.	S
6	0x5002_1000	0x5002_1FFF	4KB	S_SYSCONTROL	System Control Registers Block.	SP
7	0x5002_2000	0x5002_2FFF	4KB	SYS_PPU	System Power Policy Unit.	SP
8	0x5002_3000	0x5002_3FFF	4KB	CPU0CORE_PPU	CPU 0 Core Power Policy Unit.	SP

Table 3-5 System control regions (continued)

Row ID (alias)	Address		Size	Region name	Description	Security
	From	To				
9	0x5002_4000	0x5002_4FFF	4KB	CPU0DEBUG_PPU[a]	CPU 0 Debug Power Policy Unit.	SP
10	0x5002_5000	0x5002_5FFF	4KB	CPU1CORE_PPU	CPU 1 Core Power Policy Unit.	SP
11	0x5002_6000	0x5002_6FFF	4KB	CPU1DBG_PPU[a]	CPU 1 Debug Power Policy Unit.	SP
	0x5002_7000	0x5002_8FFF	4KB	-	Reserved, RAZ/WI.	-
12	0x5002_9000	0x5002_9FFF	4KB	DBG_PPU	System Debug Power Policy Unit.	SP
13	0x5002_A000	0x5002_AFFF	4KB	RAM0_PPU	SRAM Bank 0 Power Policy Unit.	SP
14	0x5002_B000	0x5002_BFFF	4KB	RAM1_PPU	SRAM Bank 1 Power Policy Unit.	SP
15	0x5002_C000	0x5002_CFFF	4KB	RAM2_PPU	SRAM Bank 2 Power Policy Unit.	SP
16	0x5002_D000	0x5002_DFFF	4KB	RAM3_PPU	SRAM Bank 3 Power Policy Unit.	SP
17	0x5002_E000	0x5002_EFFF	4KB	S32KWATCHDOG	CMSDK Watchdog on **S32KCLK**.	SP
18 (3)	0x5002_F000	0x5002_FFFF	4KB	S32KTIMER	CMSDK Timer on **S32KCLK**.	S-PPC
19	0x5003_0000	0x5003_FFFF	-	-	Reserved.	-

Note

- For NS_PPC, any Secure access targeting this region is blocked. A PPC controls Non-secure access to this region.
- For S_PPC, any Non-secure access targeting this region is blocked. A PPC controls Secure access targeting this region.
- NSP indicates Non-secure private access only.
- SP indicates Secure privilege access only.
- S indicates Secure access only.
- Reserved regions respond with RAZ/WI when accessed. The System Information Registers Block is mapped to both the Secure and Non-secure region and is visible to both without any security protection.

Related references

3.6 System control element on page 3-122.

[a] CPU0DBG_PPU and CPU1DBG_PPU regions do not exist if separate CPU debug power domains are not supported.

3.3 CPU element

This section lists memory locations related to the processor.

This section contains the following subsections:

3.3.1 Processor L1 cache registers

The following table lists the cache registers. All registers are Secure privileged access only.

Table 3-6 Summary of processor L1 cache registers

Offset	Register name	Access	Reset value	Full name
0x000	ICHWPARAMS	RO	Depends on configuration.	*Hardware Parameter register, ICHWPARAMS* on page 3-83
0x004	ICCTRL	RW	0x0	*Instruction Cache Control Register, ICCTRL* on page 3-83
0x008-0x0FC	-	-	-	Reserved.
0x100	ICIRQSTAT	RO	0x0	Interrupt Request Status register, see *Instruction Cache Interrupt Registers, ICIRQSTAT, ICIRQSCLR, and ICIRQEN* on page 3-84.
0x104	ICIRQSCLR	WO	0x0	Interrupt Status Clear register
0x108	ICIRQEN	RW	0x0	Interrupt Enable register
0x10C	ICDBGFILLERR	RO	0x0	*Debug Fill Error Register, ICDBGFILLERR* on page 3-85.
0x200-0x2FC	-	-	-	Reserved.
0x300	ICSH	RO	0x0	*Instruction Cache Statistic Hit register, ICSH* on page 3-85.
0x304	ICSM	RO	0x0	*Instruction Cache Statistic Miss Count register, ICSM* on page 3-85.
0x308	ICSUC	RO	0x0	*Instruction Cache Statistic Uncached Count register, ICSUC* on page 3-86.
0x30C-0xFCC	-	-	-	Reserved.
0xFD0	PIDR4	RO	0x04	Product ID Register 4.
0xFD4	PIDR5	RO	0x0	Product ID Register 5.
0xFD8	PIDR6	RO	0x0	Product ID Register 6.
0xFDC	PIDR7	RO	0x0	Product ID Register 7.
0xFE0	PIDR0	RO	0x57	Product ID Register 0.
0xFE4	PIDR1	RO	0xB8	Product ID Register 1.
0xFE8	PIDR2	RO	0x0B	Product ID Register 2.
0xFEC	PIDR3	RO	0x00	Product ID Register 3.

Table 3-6 Summary of processor L1 cache registers (continued)

Offset	Register name	Access	Reset value	Full name
0xFF0	CIDR0	RO	0x0D	Component ID Register 0.
0xFF4	CIDR1	RO	0xF0	Component ID Register 1.
0xFF8	CIDR2	RO	0x05	Component ID Register 2.
0xFFC	CIDR3	RO	0xB1	Component ID Register 3.

Hardware Parameter register, ICHWPARAMS

The ICHWPARAMS is a read-only register that describes the instruction cache configuration.

Table 3-7 ICHWPARAMS register

Bits	Name	Access	Reset value	Description
[31:16]	COFFSET	RO	0x0	Cacheable Offset Address. Set the top address 31:16 of the cacheable address region.
[15:12]	COFFSIZE	RO	0x3	Cacheable Block Size. Sets the block size of the cacheable region and also indicates the number of top address bits on an access that is not used in cache lookup but is compared against the top bits of COFFSET, for example: 0x0: 4GB 0x1: 2GB 0x2: 1GB 0x3: 512MB
[11:7]	Reserved	RO	-	Reserved.
[6]	INVMAT	RO	Configuration dependent	Indicates invalidate cache line on write match is enabled. This bit depends on CPU0_ICACHEINVMAT for CPU0, or CPU1_ICACHEINVDMA for CPU1.
[5]	DMA	RO	0x0	Presence of DMA Engine: 0 = Instruction cache does not support prefetch and locking. 1 = Instruction cache supports prefetch and locking.
[4]	STATS	RO	0x1	Presence of Statistic Functionality.
[3:0]	CSIZE	RO	Parameterized	Cache size. Defines the size of the instruction cache: 9: 512 byte 10: 1 KB 11: 2 KB 12: 4 KB 13: 8 KB 14: 16 KB Other value reserved.

Instruction Cache Control Register, ICCTRL

The ICCTRL register allows software to control the cache. This includes enabling or disabling the cache, starting invalidations and configuring what must be cached or uncached based on the **HHINT[2]** signal from the processor.

For write accesses, this register only supports 32-bit writes. Any write that is not 32 bits is ignored.

Table 3-8 ICCTRL register

Bits	Name	Access	Reset value	Description
[31:6]	Reserved	RO	0x0	-
5	HALLOC	RW	0x0	Enable Handler Allocation: • When set to LOW, all incoming handler code fetches are not allocated a cache line if a miss occurs. If the fetch results in a hit in the cache, the access is treated as if it is cached. HALLOC LOW allows handler code accesses to be more deterministic. It also avoids cache trashing if there are many interrupt service routines. HALLOC has no effect on DMA fetches and line locking. • When HALLOC is set to HIGH, then handler code access is treated like any other code access arriving at its interface.
4	STATC	WO	0x0	Clear Statistic values. Writing a 1 to this register triggers the instruction cache to start clear all cache statistic counters.
3	STATEN	RW	0x0	Enable Statistic function. When set to HIGH, cache statistic counters are enabled. When set to LOW, cache statistic counters are disabled.
2	FINV	WO	0x0	Full Cache Invalidate. Writing a 1 to this register triggers the instruction cache to start invalidating all cache lines.
1	Reserved	RO	0x0	Reserved.
0	CACHEEN	RW	0x0	Enable Cache. When set to HIGH, caching is enabled. When set to LOW, all accesses bypass the cache.

Instruction Cache Interrupt Registers, ICIRQSTAT, ICIRQSCLR, and ICIRQEN

The instruction cache interrupt registers allow software to determine the source of an interrupt coming from the instruction cache. They also allow software to clear, disable, or enable the interrupts:

- ICIRQSTAT is read-only (RO). The Instruction Cache Interrupt Status register holds the status of all interrupts before being masked.
- ICIRQSCLR is write-only (WO). The Instruction Cache Interrupt Clear register allows the status of active interrupts to be cleared by writing a 1 to is associated field.
- ICIRQEN is read/write (RW). The Instruction Cache Interrupt Masks register allows each interrupt status to be enabled or disabled from driving the instruction cache interrupt signal.

All interrupt registers share fields, as the following table shows.

Note

When enabling an interrupt, it is possible to receive an interrupt immediately on the disable event if that interrupt event occurred before the programming event.

Table 3-9 Instruction cache interrupt registers

Bits	Name	Reset value	Description
[31:6]	-	0x0	Reserved.
[5]	SS	0	Statistics Saturated IRQ. Indicates that the internal statistic counters have saturated.
[4]	SV	0	Security violation IRQ status.

Table 3-9 Instruction cache interrupt registers (continued)

Bits	Name	Reset value	Description
[3]	CFE	0	Cache Fill Error IRQ. Indicates that a bus error occurred while filling a cache line.
[2]	CEC	0	Cache Enable Complete IRQ. Indicates that a request to enable the cache has been completed.
[1]	CDC	0	Cache Disable Complete IRQ. Indicates that a request to disable the cache has been completed.
[0]	IC	0	Invalidate Complete IRQ. Indicates that a cache invalidation process has been completed.

Debug Fill Error Register, ICDBGFILLERR

The ICDBGFILLERR register allows software to discover the address that is involved in a recent fill error.

Table 3-10 ICDBGFILLERR register

Bits	Name	Access	Reset value	Description
[31:0]	ERRADDR	RO	`0x0`	Address where the latest fill error was seen.

Note

The address might change between the associated interrupt and the read by the processor, and is therefore intended only as a crude debug assist.

The two least significant bits of this register always return zeros.

Instruction Cache Statistic Hit register, ICSH

The *Instruction Cache Statistic Hit Register* (ICSHR) register is a register that allows software to read the instruction cache hit counter value.

The ICSH register allows software to read the instruction cache hit counter value.

This register counts the number of read accesses that results in cache hits since the last counter clear operation. This register ignores write accesses.

Table 3-11 ICSH register

Bits	Name	Access	Reset value	Description
[31:0]	CSHR	RO	`0x0`	Cache Hit Counter register.

Instruction Cache Statistic Miss Count register, ICSM

The ICSM register allows software to read the instruction cache miss counter value. This register counts the number of read accesses that results in cache misses since the last counter clear operation. This register ignores write accesses.

Table 3-12 ICSM register

Bits	Name	Access	Reset value	Description
[31:0]	CSM	RO	`0x0`	Cache miss counter register.

Instruction Cache Statistic Uncached Count register, ICSUC

The ICSUC register allows software to read the instruction cache Uncached access counter value. This register counts the number of uncached read accesses seen by the instruction cache since the last counter clear operation. This register ignores write accesses.

Table 3-13 ICSUC register

Bits	Name	Access	Reset value	Description
[31:0]	CSUC	RO	0x0	Uncached access Counter Register.

CIDRx, PIDRx

The CIDR and PIDR registers are statically configured at compile time. PIDR3 is specified from a static register to allow ECO modifications to the identity.

The default values are listed in the following tables.

Table 3-14 Product and component ID registers

Name	Function	Access	Reset value	Description
PIDR0	Part number	RO	0x57	Product ID Register 0.
PIDR1	Part number	RO	0xB8	Product ID Register 1.
PIDR2	Part revision	RO	0x0B	Product ID Register 2.
PIDR3	Revision number	RO	0x00	Product ID Register 3.
PIDR4	Reserved	RO	0x04	Product ID Register 4.
PIDR5	Reserved	RO	0x00	Product ID Register 5.
PIDR6	Reserved	RO	0x00	Product ID Register 6.
PIDR7	Reserved	RO	0x00	Product ID Register 7.
CIDR0	Preamble	RO	0x0D	Component ID Register 0.
CIDR1	Preamble	RO	0xF0	Component ID Register 1.
CIDR2	Preamble	RO	0x05	Component ID Register 2.
CIDR3	Preamble	RO	0xB1	Component ID Register 3.

3.3.2 Processor L1 cache programming

This section describes the recommended practice for programming the processor L1 cache. It is not however, intended as an exhaustive guide on writing driver software.

Initialization

After power and reset have been applied, the cache starts up in a disabled state and begins its invalidation process. Because the cache is disabled, all accesses arriving at the cache are not cached and bypass the cache.

During the invalidation process, you can always enable the cache by setting the CACHEEN control bit to 1. However, all accesses through the cache are still treated as uncached and bypass the cache until the cache invalidation process completes.

At the end of the cache invalidation process, the IC_STATUS interrupt status is asserted, and if that interrupt is already enabled or is enabled at a later stage, an interrupt is raised. If caching of code fetches is important, you can poll this status register, or wait for this interrupt to be raised before continuing the rest of your code execution.

Cache disable

You can disable the cache at anytime by clearing the CACHEEN control bit.

If there are outstanding accesses occurring, these are completed before the cache is disabled. Therefore, to determine when the cache is finally disabled, the software can either poll the CDC_STATUS register or enable the CDC interrupt and wait for the interrupt to arrive, after clearing the CACHEEN bit.

Cache invalidation

You can invalidate the cache at anytime by setting FINV. Setting this bit triggers a full cache invalidation.

During cache invalidation, all accesses through the cache are still treated as uncached and bypass the cache until the cache invalidation process completes. At the end of the cache invalidation process, the IC_STATUS interrupt status is asserted. If that interrupt is already enabled or is enabled at a later stage, an interrupt is raised.

Performance targets

The cache aims to improve the average performance of the connected processor by holding a local copy of previously accessed or specified memory locations.

With zero-cycle access time on address hits, the design improves performance on average by an unspecified amount. This amount is not possible to be determined precisely as it affected by many design parameters, code behaviors, and system considerations.

The cache can actively reduce processor performance in the following scenarios:

Writes
: If INVMAT feature is used and a write access has to be compared, then there is an extra cycle of bus latency.

Cache misses
: A cache miss causes a fetch to occur, and causes an extra cycle of bus latency for the initial data.

  Subsequent transactions are also stalled while the rest of the fetch process occurs. The extra latency is determined by the time it takes for the memory subsystem to return the rest of the WRAP4 transaction.

3.3.3 Ensuring the cache handles memory modifications

The instruction cache does not support the ability to maintain coherency between an external code location with a corresponding cache line that is already in the cache.

If the external location is to be modified, the system software must invalidate the cache. Having Secure cached lines in the cache that are not coherent to the lines in external code memory is a security issue that must be avoided. To maintain coherency when modifying code space contents:

1. Disable the instruction cache.
2. Manually invalidate the full instruction cache.
3. Modify the code space content.
4. Re-enable the instruction cache.

If SAU or MPC is modified so that a region in memory that is recently cached has moved from one security setting to another, because the instruction cache maintains the security attribute, it is not allowed a hit on the cached line using the new security attribute and results in a cache miss. Therefore, this can result in Secure and Non-secure versions of the same memory location residing in the cache and reducing its efficiency. It can also potentially pose a security risk if the older cache line is accessed again with the original access attribute when it is no longer intended to be available in that world. Therefore, Arm recommends that you invalidate the cache to avoid this risk. To maintain coherency and security when modifying code space contents security attributes:

1. Disable the instruction cache.
2. Manually invalidate the instruction cache.

3. Reprogram and reconfigure the code area contents and security behavior.
4. Enable the instruction cache.

3.3.4 CPU Local Security Control Register

Each CPU element contains a register block that contains registers to allow the security locks of each processor to be configured. Each register block resides in the same reset and power domain as its associated core. When a processor is powered down, they are also cleared when poweredup.

Table 3-15 CPU Local Security Configuration register map

Offset	Register name	Access	Reset value	Full name
0x000	CPUSECCFG	RW	0x0	*CPU Local Security Configuration register, CPUSECCFG* on page 3-88
0x004-0xFC8	-	-	-	Reserved.
0xFD0	PIDR4	RO	0x04	Product ID Register 4.
0xFD4	PIDR5	RO	0x0	Product ID Register 5.
0xFD8	PIDR6	RO	0x0	Product ID Register 6.
0xFDC	PIDR7	RO	0x0	Product ID Register 7.
0xFE0	PIDR0	RO	0x59	Product ID Register 0.
0xFE4	PIDR1	RO	0xB8	Product ID Register 1.
0xFE8	PIDR2	RO	0x0B	Product ID Register 2.
0xFEC	PIDR3	RO	0x00	Product ID Register 3.
0xFF0	CIDR0	RO	0x0D	Component ID Register 0.
0xFF4	CIDR1	RO	0xF0	Component ID Register 1.
0xFF8	CIDR2	RO	0x05	Component ID Register 2.
0xFFC	CIDR3	RO	0xB1	Component ID Register 3.

CPU Local Security Configuration register, CPUSECCFG

The CPUSECCFG register allows software to set security lock bits at the interface of each associated processor.

Table 3-16 CPUSECCFG register

Bits	Name	Access	Reset value	Description
[31:2]	-	-	0x0	Reserved
[1]	LOCKSAU	Write one to set.	0	Controls the **LOCKSAU** signal on the processor. When set to 1, disables writes to the SAU_CTRL, SAU_RNR, SAU_RBAR, and SAU_RLAR registers from software or from a debug agent connected to the processor. When set to 1, it cannot be cleared until reset.
[0]	LOCKSVTAIRCR	Write one to set.	0	Controls the **LOCKSVTAIRCR** signal on the processor. When set to 1, disables writes to the VTOR_S, AIRCR.PRIS, and AIRCR.BFHFNMINS registers. When set to 1, it cannot be cleared until reset.

3.3.5 CPU_IDENTITY

The CPU element also implements a CPU_IDENTITY register block that is only visible to accesses on the System interface from the processor within this element.

The base address of this read only register is `0x4001_F000` in a Non-Secure region and `0x5001_F000` in the Secure region and both areas are always accessible. Any writes access to it is ignored. The following table lists the registers in this block.

Table 3-17 CPUSECCFG register

Offset	Name	Access	Reset value	Description
`0x000`	CPUID	Read-only	CPU<N>_CPUID[3:0]	Unique CPU Identity Number, where <N> is '0' for CPU 0 and '1' for CPU 1. Set to zero for a single processor system.
`0x004` - `0xFCC`	Reserved	-	-	-
`0xFD0`	PIDR4	Read-only	`0x0000_0004`	Peripheral ID 4
`0xFD4`	PIDR5	Read-only	`0x0000_0000`	Reserved
`0xFD8`	PIDR6	Read-only	`0x0000_0000`	Reserved
`0xFDC`	PIDR7	Read-only	`0x0000_0000`	Reserved
`0xFE0`	PIDR0	Read-only	`0x0000_0055`	Peripheral ID 0
`0xFE4`	PIDR1	Read-only	`0x0000_00B8`	Peripheral ID 1
`0xFE8`	PIDR2	Read-only	`0x0000_000B`	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

3.3.6 PPB regions

The *Private Peripheral Bus* (PPB) includes the following regions that provide access to the processor core internal processor resources:

- The *Instrumentation Trace Macrocell* (ITM) if included.
- The *Data Watchpoint and Trace* (DWT) if included.
- The *Flash Patch and Breakpoint* (FPB) if included.
- The *System Control Space* (SCS) which includes the *Memory Protection Unit* (MPU) and the *Nested Vectored Interrupt Controller* (NVIC).
- The *Embedded Trace Macrocell* (ETM), if included.
- The *Cross Trigger Interface* (CTI), if included.
- The debug ROM table.

This memory region is as defined in the *Arm®v8-M Architecture Reference Manual* and the *Arm® Cortex®-M33 Processor Integration and Implementation Manual*.

3.3.7 Interrupts

This section describes the (*Nested Vectored Interrupt Controller*) NVIC and the interrupt signal map.

The NVIC supports:

- An implementation-defined number of interrupts in the range 4-240.
- A programmable priority level of 0-255 for each interrupt. A higher level corresponds to a lower priority, so level 0 is the highest interrupt priority.
- Level and pulse detection of interrupt signals.
- Dynamic reprioritization of interrupts.
- Grouping of priority values into group priority and subpriority fields.

- Interrupt tail-chaining.
- An external *Non-Maskable Interrupt* (NMI)
- Optional External Wake Up Controllers that provide ultra-low power sleep mode support.

Interrupt signals

This section describes interrupt signals and exceptions.

Table 3-18 Interrupt signals

Interrupt input	CPU 0 interrupt source	CPU 1 interrupt source
NMI	Combined SECURE WATCHDOG, S32KWATCHDOG, and NMI_Expansion	Combined SECURE WATCHDOG, S32KWATCHDOG, and NMI_Expansion.
IRQ[0]	NON-SECURE WATCHDOG Reset Request	NON-SECURE WATCHDOG Reset Request.
IRQ[1]	NON-SECURE WATCHDOG Interrupt	NON-SECURE WATCHDOG Interrupt.
IRQ[2]	S32K Timer	S32K Timer.
IRQ[3]	TIMER 0	TIMER 0.
IRQ[4]	TIMER 1	TIMER 1.
IRQ[5]	DUAL TIMER	DUAL TIMER.
IRQ[6]	Message Handling Unit 0 CPU0 Interrupt	Message Handling Unit 0 CPU1 Interrupt.
IRQ[7]	Message Handling Unit 1 CPU0 Interrupt	Message Handling Unit 1 CPU1 Interrupt.
IRQ[8]	Reserved	Reserved.
IRQ[9]	MPC Combined (Secure)	MPC Combined (Secure).
IRQ[10]	PPC Combined (Secure)	PPC Combined (Secure).
IRQ[11]	MSC Combined (Secure)	MSC Combined (Secure).
IRQ[12]	Bridge Error Combined Interrupt (Secure)	Bridge Error Combined Interrupt (Secure).
IRQ[13]	CPU 0 Instruction Cache Interrupt	CPU 1 Instruction Cache Interrupt.
IRQ[14]	Reserved	Reserved.
IRQ[15]	SYS_PPU	SYS_PPU.
IRQ[16]	CPU0_PPU	CPU0_PPU.
IRQ[17]	CPU1_PPU	CPU1_PPU.
IRQ[18]	CPU0DBG_PPU	CPU0DBG_PPU.
IRQ[19]	CPU1CBG_PPU	CPU1DBG_PPU.
IRQ[20]	Reserved	Reserved.
IRQ[21]	Reserved	Reserved.
IRQ[22]	RAM0_PPU	RAM0_PPU.
IRQ[23]	RAM1_PPU	RAM1_PPU.
IRQ[24]	RAM2_PPU	RAM2_PPU.
IRQ[25]	RAM3_PPU	RAM3_PPU.
IRQ[26]	DEBUG_PPU	DEBUG_PPU.
IRQ[27]	Reserved	Reserved.

Table 3-18 Interrupt signals (continued)

Interrupt input	CPU 0 interrupt source	CPU 1 interrupt source
IRQ[28]	CPU0CTIIRQ0	CPU1CTIIRQ0.
IRQ[29]	CPU0CTIIRQ1	CPU1CTIIRQ1.
IRQ[31:30]	Reserved	Reserved.
IRQ[95:32]	Expansion Interrupt Inputs	Expansion Interrupt Inputs.

Interrupt controller registers

A summary of the interrupt controller registers is listed in the following table.

Table 3-19 Summary of interrupt controller registers

Address	Name	Access	Reset value	Description
`0xE000E004`	ICTR	RO	-	Interrupt Controller Type Register.
`0xE000E100` – `0xE000E11C`	NVIC_ISER0-NVIC_ISER7	RW	`0`	Interrupt Set Enable Registers.
`0xE000E180` – `0xE000E19C`	NVIC_ICER0-NVIC_ICER7	RW	`0`	Interrupt Clear Enable Registers.
`0xE000E200` – `0xE000E21C`	NVIC_ISPR0-NVIC_ISPR7	RW	`0`	Interrupt Set Pending Registers.
`0xE000E280` – `0xE000E29C`	NVIC_ICPR0-NVIC_ICPR7	RW	`0`	Interrupt Clear Pending Registers.
`0xE000E300` – `0xE000E31C`	NVIC_IABR0-NVIC_IABR7	RO	`0`	Interrupt Active Bit Registers.
`0xE000E400` – `0xE000E41F`	NVIC_IPRO-NVIC_IPR7	RW	`0`	Interrupt Priority Registers.

For more information on the interrupt controller, see the following documents:

- *Arm® Cortex®-M33 Processor Technical Reference Manual*.
- *Arm®v8-M Architecture Reference Manual*.

3.4 Base element

This section describes control registers associated with several base element components.

This section contains the following subsections:

3.4.1 CMSDK timer

The base element has two CMSDK Timers:

- CMSDK TIMER 0 is located in Non-secure region at `0x4000_0000` and in Secure region at `0x5000_0000`.
- CMSDK TIMER 1 is located in Non-secure region at `0x4000_1000` and in Secure region at `0x5000_1000`.

CTI triggers from the debug subsystem halt the timer.

Note

The **EXTIN** input of the timers is connected to the CTI debug halt logic, and it is used to stop the timer counter logic, if there is a debug halt access.
To enable this functionality the **EXTIN** must be enabled by writing to the CTRL register:

- CTRL bit[2] = 0.
- CTRL bit[1] = 1.

nWARMRESETSYS resets the timer, which resides in the PD_SYS power domain.

The following table lists a summary of the registers in the timer.

Table 3-20 Summary of CMSDK Timer registers

Offset	Name	Access	Width	Reset value	Description
`0x000`	CTRL	RW	4	`0x0`	3: Interrupt enable. 2: Select external input as clock. 1: Select external input as enable. 0: Enable.
`0x004`	VALUE	RW	32	`0x0`	Current value.
`0x008`	RELOAD	RW	32	`0x0`	Reload value. A write to this register sets the current value.
`0x00C`	INTSTATUS INTCLEAR	RW	1	`0x0`	Timer interrupt. Write 1 to clear.
`0xFD0`	PID4	RO	8	`0x04`	Peripheral ID register 4.
`0xFD4`	PID5	RO	8	`0x0`	Peripheral ID register 5.
`0xFD8`	PID6	RO	8	`0x0`	Peripheral ID register 6.
`0xFDC`	PID7	RO	8	`0x0`	Peripheral ID register 7.

Table 3-20 Summary of CMSDK Timer registers (continued)

Offset	Name	Access	Width	Reset value	Description
0xFE0	PID0	RO	8	0x22	Peripheral ID register 0. [7:0]: Part number[7:0].
0xFE4	PID1	RO	8	0xB8	Peripheral ID register 1. [7:4]: jep106_id_3_0. [3:0]: Part number[11:8].
0xFE8	PID2	RO	8	0x0B	Peripheral ID register 2. [7:4]: Revision. [3]: jedec_used. [2:0]: jep106_id_6_4.
0xFEC	PID3	RO	8	0x0	Peripheral ID register 3 [7:4]: ECO revision number. [3:0]: Customer modification number.
0xFF0	CID0	RO	8	0x0D	Component ID register 0.
0xFF4	CID1	RO	8	0xF0	Component ID register 1.
0xFF8	CID2	RO	8	0x05	Component ID register 2.
0xFFC	CID3	RO	8	0xB1	Component ID register 3.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual* for register details.

3.4.2 CMSDK dual timer

CTI triggers from the debug subsystem halt the timers.

nWARMRESETSYS reset the timers, which reside in the PD_SYS power domain.

The base element has a single CMSDK DUAL TIMER in Non-secure region at 0x4000_2000 and in Secure region at 0x5000_2000.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual* for register details for the dual timer.

Table 3-21 Summary of CMSDK Dual Timer registers

Offset	Name	Access	Width	Reset value	Description
0x00	TIMER1LOAD	RW	32	0x0	The value from which the counter is to decrement.
0x04	TIMER1VALUE	RO	32	0xFFFFFFFF	The current value of the decrementing counter.
0x08	TIMER1CONTROL	RW	8	0x20	Timer interrupt and enable control.
0x0C	TIMER1INTCLR	WO	-	-	Any write clears the interrupt output from the counter.
0x10	TIMER1RIS	RO	1	0x0	Indicates the raw interrupt status from the counter.
0x14	TIMER1MIS	RO	1	0x0	Indicates the masked interrupt status from the counter.
0x18	TIMER1BGLOAD	RW	32	0x0	Contains the value from which the counter is to decrement.
0x20	TIMER2LOAD	RW	32	0x0	The value from which the counter is to decrement.

Table 3-21 Summary of CMSDK Dual Timer registers (continued)

Offset	Name	Access	Width	Reset value	Description
0x24	TIMER2VALUE	RO	32	0xFFFFFFFF	The current value of the decrementing counter.
0x28	TIMER2CONTROL	RW	8	0x20	Timer interrupt and enable control.
0x2C	TIMER2INTCLR	WO	-	-	Any write clears the interrupt output from the counter.
0x30	TIMER2RIS	RO	1	0x0	Indicates the raw interrupt status from the counter.
0x34	TIMER2MIS	RO	1	0x0	Indicates the masked interrupt status from the counter.
0x38	TIMER2BGLOAD	RW	32	0x0	Contains the value from which the counter is to decrement.
0xF00	TIMERITCR	RW	1	0x0	Enables integration test mode. When in this mode, the Integration Test Output Set Register directly controls the masked interrupt outputs.
0xF04	TIMERITOP	WO	2	0x0	When in integration test mode, the values directly drive the enabled interrupt outputs.
0xFD0	TIMERPERIPHID4	RO	8	0x04	Peripheral ID Register 4: [7:4]: Block count. [3:0]: jep106_c_code.
0xFD4	TIMERPERIPHID5	RO	8	0x00	Peripheral ID Register 5.
0xFD8	TIMERPERIPHID6	RO	8	0x00	Peripheral ID Register 6.
0xFDC	TIMERPERIPHID7	RO	8	0x00	Peripheral ID Register 7.
0xFE0	TIMERPERIPHID0	RO	8	0x23	Peripheral ID Register 0: [7:0]: Part number[7:0].
0xFE4	TIMERPERIPHID1	RO	8	0xB8	Peripheral ID Register 1: [7:4]: jep106_id_3_0.[3:0]: Part number[11:8].
0xFE8	TIMERPERIPHID2	RO	8	0x1B	Peripheral ID Register 2: [7:4]: Revision. [3]: jedec_used. [2:0]: jep106_id_6_4.
0xFEC	TIMERPERIPHID3	RO	8	0x00	Peripheral ID Register 3: [7:4]: ECO revision number. [3:0]: customer modification number.
0xFF0	TIMERPCELLID0	RO	8	0x0D	Component ID Register 0.
0xFF4	TIMERPCELLID1	RO	8	0xF0	Component ID Register 1.
0xFF8	TIMERPCELLID2	RO	8	0x05	Component ID Register 2.
0xFFC	TIMERPCELLID3	RO	8	0xB1	Component ID Register 3.

3.4.3 CMSDK watchdog timers

The Base element has two CMSDK watchdog timers:

- Non-secure CMSDK watchdog in the Non-secure region at `0x4008_1000`
- Secure CMSDK watchdog in the Secure region at `0x5008_1000`

Each watchdog is permanently mapped to either a Secure or a Non-secure region of address space:

- The Non-secure watchdog can raise an interrupt to both cores. On a watchdog reset request event, a separate interrupt is raised instead, but software can also choose to allow it to directly reset the system.
- The Secure watchdog can raise a *Non-Maskable Interrupt* (NMI) to both cores. However, in this case, a watchdog reset event resets the entire system.

CTI triggers from the debug subsystem halt the timers.

nWARMRESETSYS resets the timers, which reside in the PD_SYS power domain.

The following table shows a summary of the CMSDK watchdog registers.

Table 3-22 Summary of Watchdog registers

Offset	Name	Access	Width	Reset value	Description
`0x00`	WDOGLOAD	RW	32	`0xFFFFFFFF`	Contains the value from which the counter is to decrement.
`0x04`	WDOGVALUE	RO	32	`0xFFFFFFFF`	Contains the current value of the decrementing counter.
`0x08`	WDOGCONTROL	RW	2	`0x0`	Enables the software to control the watchdog unit.
`0x0C`	WDOGINTCLR	WO	-	-	A write of any value register clears the watchdog interrupt, and reloads the counter from the value in WDOGLOAD.
`0x10`	WDOGRIS	RO	1	`0x0`	Indicates the raw interrupt status from the counter.
`0x14`	WDOGMIS	RO	1	`0x0`	Indicates the masked interrupt status from the counter.
`0xC00`	WDOGLOCK	RW	32	`0x0`	Disables write accesses to all other registers.
`0xF00`	WDOGITCR	RW	1	`0x0`	Enables integration test mode.
`0xF04`	WDOGITOP	WO	2	`0x0`	When the WDOGITOP Register is in integration test mode, the values in this register directly drive the enabled interrupt output and reset output.
`0xFD0`	WDOGPERIPHID4	RO	8	`0x04`	Peripheral ID Register 4: [7:4]: Block count. [3:0]: jep106_c_code.
`0xFD4`	WDOGPERIPHID5	RO	8	`0x0`	Peripheral ID Register 5.
`0xFD8`	WDOGPERIPHID6	RO	8	`0x0`	Peripheral ID Register 6.
`0xFDC`	WDOGPERIPHID7	RO	8	`0x0`	Peripheral ID Register 7.
`0xFE0`	WDOGPERIPHID0	RO	8	`0x24`	Peripheral ID Register 0: [7:0]: Part number[7:0].
`0xFE4`	WDOGPERIPHID1	RO	8	`0xB8`	Peripheral ID Register 1: [7:4]: jep106_id_3_0. [3:0]: Part number[11:8].

Table 3-22 Summary of Watchdog registers (continued)

Offset	Name	Access	Width	Reset value	Description
0xFE8	WDOGPERIPHID2	RO	8	0x1B	Peripheral ID Register 2: [7:4]: Revision. [3]: jedec_used. [2:0]: jep106_id_6_4.
0xFEC	WDOGPERIPHID3	RO	8	0x0	Peripheral ID Register 3: [7:4]: ECO revision number. [3:0]: Customer modification number.
0xFF0	WDOGPCELLID0	RO	8	0x0D	Component ID Register 0.
0xFF4	WDOGPCELLID1	RO	8	0xF0	Component ID Register 1.
0xFF8	WDOGPCELLID2	RO	8	0x05	Component ID Register 2.
0xFFC	WDOGPCELLID3	RO	8	0xB1	Component ID Register 3.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual* for register details of the watchdog timer.

3.4.4 AHB5 TrustZone Memory Protection Controller

The base element implements a memory protection controller for each SRAM block. Each MPC APB configuration interface is mapped to the following base addresses.

- 0x5008_3000 for SRAM Bank 0.
- 0x5008_4000 for SRAM Bank 1.
- 0x5008_5000 for SRAM Bank 2.
- 0x5008_6000 for SRAM Bank 3.

See *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* for more information on the AHB5 TrustZone Memory Protection Controller.

The configuration registers are listed in the following table:

Table 3-23 Summary of MPC registers

Offset	Name	Access	Reset value	Description
0x000	CTRL	RW	0x0	Bit[0]: Reserved. Bit[2:1]: Reserved Bit[4]: Security error response configuration (CFG_SEC_RESP) -> 0:RAZ-WI, 1: Bus Error. Bit[5]: Reserved. Bit[6]: Data interface gating request. Bit[7]: Data interface gating acknowledge (RO). Bit[8]: Auto-increment. Bit[30:9]: Reserved. Bit[31]: Security lockdown.
0x004 – 0x00C	RSVD	RO	0x0	Reserved.
0x010	BLK_MAX	RO	-	Maximum value of block-based index register.
0x014	BLK_CFG	RO	-	Bit[3:0] Block size: 0: 32 Bytes 1: 64 Bytes … Block size = 1 << (BLK_CFG+5) Bit[30:4]: Reserved. Bit[31]: Initialization in progress.
0x018	BLK_IDX	RW	0x0	Index value for accessing block-based lookup table.
0x01C	BLK_LUT[n]	RW	- (IMPLEMENTATION DEFINED)	Block based gating Look Up Table (LUT): Access to block based lookup configuration space pointed to by BLK_IDX. Bit[31:0]: each bit indicate one block: If BLK_IDX is 0, bit[0] is block #0, bit[31] is block #31. If BLK_IDX is 1, bit[0] is block #32, bit[31] is block #63. … For each configuration bit, 0 indicates Secure, 1 indicates Non-secure. A full word write or read to this register automatically increments the BLK_IDX by one.
0x020	INT_STAT	RO	0x0	Bit[0]: mpc_irq triggered. Bit[31:1]: Reserved.
0x024	INT_CLEAR	WO	0x0	Bit[0]: mpc_irq clear (cleared automatically). Bit[31:1]: Reserved.

Table 3-23 Summary of MPC registers (continued)

Offset	Name	Access	Reset value	Description
0x028	INT_EN	RW	0x0	Bit[0]: mpc_irq enable. Bits are valid when mpc_irq triggered is set.
0x02C	INT_INFO1	RO	0x0	haddr[31:0] when the first mpc_irq triggered. Bits are valid when mpc_irq triggered is set.
0x030	INT_INFO2	RO	0x0	Various debug bits when the first mpc_irq triggered; Bit [15:0]: hmaster. Bit [16]: hnonsec. Bit [17]: cfg_ns. Bit [31:18]: Reserved. Bits are valid when mpc_irq triggered is set.
0x034	INT_SET	WO	0x0	Bit[0]: mpc_irq set. Debug purpose only. Bit[31:1]: Reserved.
0x038 – 0xFCC	RSVD	RO	0x0	Reserved.
0xFD0	PIDR4	RO	0x04	Peripheral ID 4 [7:4]: block count. [3:0]: jep106_c_code.
0xFD4	PIDR5	RO	0x0	Peripheral ID 5 (not used).
0xFD8	PIDR6	RO	0x0	Peripheral ID 6 (not used).
0xFDC	PIDR7	RO	0x0	Peripheral ID 7 (not used).
0xFE0	PIDR0	RO	0x60	Peripheral ID 0 (Part number [7:0].)
0xFE4	PIDR1	RO	0xB8	Peripheral ID 1 [7:4]: jep106_id_3_0. [3:0]: Part number.
0xFE8	PIDR2	RO	0x0B	Peripheral ID 2 [7:4]: revision, [3]: jedec_used, [2:0]: jep106_id_6_4.
0xFEC	PIDR3	RO	0x0	Peripheral ID 3 [7:4]: ECO revision number, [3:0]: Customer modification number.
0xFF0	CIDR0	RO	0x0D	Component ID 0.
0xFF4	CIDR1	RO	0xF0	Component ID 1 (PrimeCell class).
0xFF8	CIDR2	RO	0x05	Component ID 2.
0xFFC	CIDR3	RO	0xB1	Component ID 3.

Look Up Table (LUT) examples

The contents of the LUT can be accessed in several ways that might require different configurations of the autoincrement function of the BLK_IDX register.

To dump the full contents of the LUT:

1. Set the autoincrement enable bit, CTRL[8], to `0x1`.
2. Read the BLK_MAX register. This has a value `0xN` which represents the last address in the LUT.
3. Write `0x0` to the BLK_IDX register.
4. Read the BLK_LUT register to `0xN` times to read the complete LUT.

To rewrite the full contents of the LUT:

1. Set autoincrement enable bit, CTRL[8], to `0x1`.
2. Read the BLK_MAX register. This has a value `0xN` which represents the last address in the LUT.
3. Write `0x0` to the BLK_IDX register.
4. Write the new values to the BLK_LUT register `0xN` times to fill the complete LUT.

To read-modify-write:

1. Set autoincrement enable bit, CTRL[8], to `0x0`.
2. Write the required address to the BLK_IDX.
3. Read the current contents of the LUT.
4. Write the new contents to the LUT.

   ——— **Note** ———

   Even byte accesses can be used to update only the required byte of the register without reading the full contents:

   ———————————

Configuration lockdown

The AHB5 TrustZone MPC provides a configuration lockdown feature that prevents malicious software from changing the security configuration. Writing `0x1` to the security lockdown bit, CTRL[31], enables the configuration lockdown feature.

After the configuration lockdown feature is enabled:

- It can only be disabled by a component reset which resets CTRL[31] to `0`.
- The following registers are read-only:
  - CTRL.
  - BLK_LUT.
  - INT_EN.

——— **Note** ———

Arm recommends that you write `0x1` to the LUT auto-increment bit, CTRL[8] before enabling the configuration lockdown feature.

When the feature is enabled, only LUT dumping is available which is simpler when BLK_IDX increments automatically during the dump.

———————————

3.4.5 Message handling unit

Two *Message Handling Units* (MHU) allow software to raise interrupts to the cores.

Each MHU is mapped to a Secure and a Non-secure area as follows:

- MHU0 in Non-secure region at `0x4000_3000` and secure region at `0x5000_3000`.
- MHU1 in Non-secure region at `0x4000_4000` and secure region at `0x5000_4000`.

The APB PPC can control which area the MHU reside in.

If there is only one core in the system, the MHU1 does not exist and the two regions are reserved. Any accesses to the regions is RAZ/WI.

For write access to these registers, only 32-bit writes are supported. Any Byte and halfword writes result in its write data being ignored.

See *CPU 0 interrupt registers* on page 3-100.

Each MHU has the following register map.

Table 3-24 Summary of MHU registers

Offset	Name	Access	Reset value	Description
0x000	CPU0INTR_STAT	RO	0x0	CPU 0 Interrupt Status Register.
0x004	CPU0INTR_SET	WO	0x0	CPU 0 Interrupt Set Register.
0x008	CPU0INTR_CLR	WO	0x0	CPU 0 Interrupt Clear Register.
0x00C	Reserved	-	0x0	Reserved.
0x010	CPU1INTR_STAT	RO	0x0	CPU 1 Interrupt Status Register.
0x014	CPU1INTR_SET	WO	0x0	CPU 1 Interrupt Set Register.
0x018	CPU1INTR_CLR	WO	0x0	CPU 1 Interrupt Clear Register.
0x01C – 0xFC8	Reserved	-	0x0	Reserved.
0xFD0	PIDR4	RO	0x0000_0004	Peripheral ID 4.
0xFD4 – 0xFDC	Reserved	RO	0x0	Reserved.
0xFE0	PIDR0	RO	0x0000_0056	Peripheral ID 0.
0xFE4	PIDR1	RO	0x0000_00B8	Peripheral ID 1.
0xFE8	PIDR2	RO	0x0000_000B	Peripheral ID 2.
0xFEC	PIDR3	RO	0x0000_0000	Peripheral ID 3.
0xFF0	CIDR0	RO	0x0000_000D	Component ID 0.
0xFF4	CIDR1	RO	0x0000_00F0	Component ID 1.
0xFF8	CIDR2	RO	0x0000_0005	Component ID 2.
0xFFC	CIDR3	RO	0x0000_00B1	Component ID 3.

CPU 0 interrupt registers

The CPU 0 Interrupt registers, CPU0INTR_STAT, CPU0INTR_SET and CPU0INTR_CLR, allow software to raise an interrupt, clear an interrupt, and also check the value written that is used to raise the interrupt to CPU 0.

Separate Set and Clear registers allow the individual bit of the interrupt status to be set and cleared. This supports software, where each bit is used to represent an event that can be independently set and cleared.

Table 3-25 CPU0INTR_STAT register

Bits	Name	Access	Width	Reset value	Description
[31:4]	Reserved	RO	28	0x000000	Reserved
[3:0]	CPU0INTR_STAT	RO	4	0x0	CPU 0 Interrupt Status. When any bit is set to HIGH, the MHU interrupt signal to CPU 1 is set to HIGH.

Table 3-26 CPU0INTR_SET register

Bits	Name	Access	Width	Reset value	Description
[31:4]	Reserved	RO	28	0x000000	Reserved
[3:0]	CPU0INTR_SET	WO	4	0x0	CPU 0 Interrupt Set. When a 1 is written to CPU0INTR_SET[n], the corresponding CPU0INTR_STAT[n] is set to HIGH.

Table 3-27 CPU0INTR_CLR register

Bits	Name	Access	Width	Reset value	Description
[31:4]	Reserved	RO	28	0x000000	Reserved
[3:0]	CPU0INTR_CLR	WO	4	0x0	CPU 0 Interrupt Set. When a 1 is written to CPU0INTR_CLR[n], the corresponding CPU0INTR_STAT[n] is set to LOW.

CPU 1 interrupt registers

The CPU 1 Interrupt registers, CPU1INTR_STAT, CPU1INTR_SET and CPU1INTR_CLR, allow software to raise an interrupt, clear an interrupt, and also check that the value written is used to raise the interrupt to CPU 1.

Separate Set and Clear registers allow the individual bit of the interrupt status to be set and cleared. This supports software, where each bit is used to represent an event that can be independently set and cleared.

Note

In a single processor system, these registers do not exist. Any access to the registers results in RAZ/WI.

Table 3-28 CPU1INTR_STAT register

Bits	Name	Access	Width	Reset value	Description
[31:4]	Reserved	RO	28	0x000000	Reserved
[3:0]	CPU1INTR_STAT	RO	4	0x0	CPU 1 Interrupt Status. When any bit is set to HIGH, the MHU interrupt signal to CPU 1 is set to HIGH.

Table 3-29 CPU1INTR_SET register

Bits	Name	Access	Width	Reset value	Description
[31:4]	Reserved	RO	28	0x000000	Reserved
[3:0]	CPU1INTR_SET	WO	4	0x0	CPU 1 Interrupt Set. When a 1 is written to CPU1INTR_SET[n], the corresponding CPU1INTR_STAT[n] is set to HIGH.

Table 3-30 CPU1INTR_CLR register

Bits	Name	Access	Width	Reset value	Description
[31:4]	Reserved	RO	28	0x000000	Reserved
[3:0]	CPU1INTR_CLR	WO	4	0x0	CPU 1 Interrupt Set. When a 1 is written to CPU1INTR_CLR[n], the corresponding CPU1INTR_STAT[n] is set to LOW.

3.4.6 Security Privilege Control Block

The Secure Privilege Control Block implements program-visible states that allow software to control security gating units within the design.

Writes to the registers must be 32 bits wide. Attempted byte and halfword writes are ignored.

Reads are only permitted from Secure privileged access. **nWARMRESETSYS** resets all the registers, which reside in the PD_SYS power domain.

The register block base address is `0x5008_0000`. The following table lists the privilege control registers.

Table 3-31 Summary of Secure Privilege Control registers

Offset	Name	Access	Reset value	Description
`0x000`	SPCSECCTRL	RW	`0x0`	Secure Privilege Controller Secure Configuration Control register.
`0x004`	BUSWAIT	RW	Parameterized	Bus Access wait control after reset.
`0x008`	Reserved	-	`0x0`	Reserved.
`0x010`	SECRESPCFG	RW	`0x0`	Security Violation Response Configuration register.
`0x014`	NSCCFG	RW	`0x0`	Non Secure Callable Configuration for IDAU.
`0x018`	Reserved	-	`0x0`	Reserved.
`0x01C`	SECMPCINTSTATUS	RO	`0x0`	Secure MPC Interrupt Status.
`0x020`	SECPPCINTSTAT	RO	`0x0`	Secure PPC Interrupt Status.
`0x024`	SECPPCINTCLR	WO	`0x0`	Secure PPC Interrupt Clear.
`0x028`	SECPPCINTEN	RW	`0x0`	Secure PPC Interrupt Enable.
`0x02C`	Reserved	-	`0x0`	Reserved.
`0x030`	SECMSCINTSTAT	RO	`0x0`	Secure MSC Interrupt Status.
`0x034`	SECMSCINTCLR	RW	`0x0`	Secure MSC Interrupt Clear.
`0x038`	SECMSCINTEN	RW	`0x0`	Secure MSC Interrupt Enable.
`0x03C`	Reserved	-	`0x0`	Reserved.
`0x040`	BRGINTSTAT	RO	`0x0`	Bridge Buffer Error Interrupt Status.
`0x044`	BRGINTCLR	WO	`0x0`	Bridge Buffer Error Interrupt Clear.
`0x048`	BRGINTEN	RW	`0x0`	Bridge Buffer Error Interrupt Enable.
`0x04C`	Reserved	-	`0x0`	Reserved.
`0x050`	AHBNSPPC0	RW	`0x0`	Non-secure Access AHB slave Peripheral Protection Control #0. Defines the Non-secure access settings for peripherals in the Base element.
`0x054` – `0x05C`	Reserved	-	`0x0`	Reserved.
`0x060`	AHBNSPPCEXP0	RW	`0x0`	Expansion 0 Non-Secure Access AHB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.

Table 3-31 Summary of Secure Privilege Control registers (continued)

Offset	Name	Access	Reset value	Description
0x064	AHBNSPPCEXP1	RW	0x0	Expansion 1 Non-Secure Access AHB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.
0x068	AHBNSPPCEXP2	RW	0x0	Expansion 2 Non-Secure Access AHB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.
0x06C	AHBNSPPCEXP3	RW	0x0	Expansion 3 Non-Secure Access AHB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: ' 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.
0x070	APBNSPPC0	RW	0x0	Non-secure Access APB slave Peripheral Protection Control #0.
0x074	APBNSPPC1	RW	0x0	Non-secure Access APB slave Peripheral Protection Control #1. This register controls the PPC within the System Control element.
0x078 – 0x07C	Reserved	-	0x0	Reserved.
0x080	APBNSPPCEXP0	RW	0x0	Expansion 0 Non-Secure Access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.
0x084	APBNSPPCEXP1	RW	0x0	Expansion 1 Non-Secure Access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.

Table 3-31 Summary of Secure Privilege Control registers (continued)

Offset	Name	Access	Reset value	Description
0x088	APBNSPPCEXP2	RW	0x0	Expansion 2 Non-Secure Access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.
0x08C	APBNSPPCEXP3	RW	0x0	Expansion 3 Non-Secure Access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1: Allow Non-secure access. 0: Disallow Non-secure access. Resets to 0.
0x090	AHBSPPPC0	RO	0x0	Secure Unprivileged Access AHB slave Peripheral Protection Control #0.
0x094 – 0x09C	Reserved	-	0x0	Reserved.
0x0A0	AHBSPPPCEXP0	RW	0x0	Expansion 0 Secure Unprivileged Access AHB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.
0x0A4	AHBSPPPCEXP1	RW	0x0	Expansion 1 Secure Unprivileged Access AHB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.
0x0A8	AHBSPPPCEXP2	RW	0x0	Expansion 2 Secure Unprivileged Access AHB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.

Table 3-31 Summary of Secure Privilege Control registers (continued)

Offset	Name	Access	Reset value	Description
0x0AC	AHBSPPPCEXP3	RW	0x0	Expansion 3 Secure Unprivileged Access AHB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.
0x0B0	APBSPPPC0	RW	0x0	Secure Unprivileged Access APB slave Peripheral. Protection Control #0. This register control the PPC within the Base element.
0x0B4	APBSPPPC1	RW	0x0	Secure Unprivileged Access APB slave Peripheral. Protection Control #1. This register controls the PPC within the System Control element.
0x0B8 – 0x0BC	Reserved	-	0x0	Reserved.
0x0C0	APBSPPPCEXP0	RW	0x0	Expansion 0 Secure Unprivileged Access APB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.
0x0C4	APBSPPPCEXP1	RW	0x0	Expansion 1 Secure Unprivileged Access APB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0
0x0C8	APBSPPPCEXP2	RW	0x0	Expansion 2 Secure Unprivileged Access APB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.
0x0CC	APBSPPPCEXP3	RW	0x0	Expansion 3 Secure Unprivileged Access APB slave Peripheral Protection Control. Each field defines the Secure unprivileged access settings for an associated peripheral: 1: Allow Secure unprivileged access. 0: Disallow Secure unprivileged access. Resets to 0.

Table 3-31 Summary of Secure Privilege Control registers (continued)

Offset	Name	Access	Reset value	Description
0x0D0	NSMSCEXP	RO	0x0	Expansion MSC Non-secure Configuration. Each field defines if a Master connected to an Expansion Master Security Controller is Secure or Non-secure: 1: Master is Non-secure, 0: Master is Secure.
0x0D4 – 0xFCC	Reserved	-	0x0	Reserved
0xFD0	PID4	RO	0x0000_0004	Peripheral ID 4
0xFD4	PID5	RO	0x0	Reserved
0xFD8	PID6	RO	0x0	Reserved
0xFDC	PID7	RO	0x0	Reserved
0xFE0	PID0	RO	0x0000_0052	Peripheral ID 0
0xFE4	PID1	RO	0x0000_00B8	Peripheral ID 1
0xFE8	PID2	RO	0x0000_000B	Peripheral ID 2
0xFEC	PID3	RO	0x0	Peripheral ID 3
0xFF0	CID0	RO	0x0000_000D	Component ID 0
0xFF4	CID1	RO	0x0000_00F0	Component ID 1
0xFF8	CID2	RO	0x0000_0005	Component ID 2
0xFFC	CID3	RO	0x0000_00B1	Component ID 3

SPCSECCTRL

The Security Privilege Controller Security Configuration Control register implements the security lock register.

Table 3-32 SPCSECCTRL register

Bits	Name	Access	Reset value	Description
[31:1]	Reserved	RO	0x0	Reserved.
0	SPCSECCFGLOCK	Write one to set.	0x0	Active High control to disable writes to security-related control registers in the Secure Privilege Control register block. After being set to HIGH, it can no longer be cleared to zero except by reset or the base system powering down. Registers that can no longer be modified when SPCSECCFGLOCK is set to HIGH are: • NSCCFG • AHBNSPPC0 • AHBNSPPCEXP<N> • APBNSPPC0 • APBNSPPC1 • APBNSPPCEXP<N> • AHBSPPPC0 • AHBSPPPCEXP<N> • APBSPPPC0 • APBSPPPC1 • APBSPPPCEXP<N> • NSMSCEXP.

BUSWAIT

The Bus Access Wait register allows software to gate access entering the Base element from specific masters in the system. This causes them to stall so that the processor can complete the configuration of the MPCs or other Security registers in the system before the stalled accesses commences.

Table 3-33 BUSWAIT register

Bits	Name	Access	Reset value	Description
[31:17]	Reserved	RO	-	Reserved
16	ACC_WAITN_STATUS	RW	0x0	Request gating units in the system to block bus access to system: 1: allow access. 0: block access. This control only affects the ACG in the system that feeds into the main AHB fabric, and it excludes access from both cores. It also drives the output signal **ACCWAITN**.
[15:1]	Reserved	RO	-	Reserved
0	ACC_WAITN	RW	ACC_WAITN_RST	This status register indicates the status of any gating units that are used to block bus access to the system: 1: allow access. 0: block access. This register reflects the values on **ACCWAITNSTATUS**.

SECRESPCFG

The Security Violation Response Configuration register is used to define a slave response to an access that causes security violation on the Bus Fabric.

Table 3-34 SECRESPCFG register

Bits	Name	Access	Reset value	Description
[31:1]	Reserved	RO	`0x0`	Reserved
0	SECRESPCFG	RW	`0x0`	This field configures the slave response in case of a security violation: 0: Read-Zero Write Ignore 1: bus error

Note

Some slaves, for example the system MPCs, provide their own control registers to configure their response.

These slaves do not depend on this control bit.

NSCCFG

The Non-secure Callable Configuration register allows software to define callable regions of memory. The register can do this if the Secure Code region is `0x1000_0000` to `0x1FFF_FFFF` and the Secure RAM region is `0x3000_0000` to `0x3FFF_FFFF`.

Table 3-35 NSCCFG register

Bits	Name	Access	Reset value	Description
[31:2]	Reserved	RO	`0x0`	Reserved
1	RAMNSC	RW	`0x0`	Configures if the RAM region (`0x3000_0000` to `0x3FFF_FFFF`) is Non-secure Callable: 0: Not Non-secure Callable 1: Non-secure Callable.
0	CODENSC	RW	`0x0`	Configures if the CODE region (`0x1000_0000` to `0x1FFF_FFFF`) is Non-secure Callable: 0: Not Non-secure Callable 1: Non-secure Callable.

SECMPCINTSTATUS

The interrupt signals from all *Memory Protection Controllers* (MPC), both within the SSE-200 subsystem and in the expansion logic are merged and sent to the Cortex-M33 NVIC on a single Interrupt signal.

The Secure MPC Interrupt Status Register therefore provides Secure software with the ability to check which one of the MPC is causing the interrupt. After the source of the interrupt is identified, you must use the MPC register interface to clear the interrupt.

Table 3-36 SECMPCINTSTATUS register

Bits	Name	Access	Reset value	Description
[31:16]	S_MPCEXP_STATUS	RO	0x0	Interrupt Status for Expansion Memory Protection Controller. Each bit *n* shows the status of input signal S_MPCEXP_STATUS[n]. The parameter MPCEXP_DIS defines if each bit within this register is implemented so that if MPCEXP_DIS[i] = 1 then S_MPCEXP_STATUS[i] is disabled and always reads as zeros.
[15:4]	Reserved	RO	0x0	Reserved.
3	S_MPCSRAM3_STATUS	RO	0x0	Interrupt Status for Memory Protection Controller of SRAM BANK 3. If BANK 3 does not exist, this area is reserved and Read as zeros.
2	S_MPCSRAM2_STATUS	RO	0x0	Interrupt Status for Memory Protection Controller of SRAM BANK 2. If BANK 2 does not exist, this area is reserved and Read as zeros.
1	S_MPCSRAM1_STATUS	RO	0x0	Interrupt Status for Memory Protection Controller of SRAM BANK 1. If BANK 1 does not exist, this area is reserved and Read as zeros.
0	S_MPCSRAM0_STATUS	RO	0x0	Interrupt Status for Memory Protection Controller of SRAM BANK 0.

SECPPCINTSTAT, SECPPCINTCLR, and SECPPCINTEN

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised from a combined interrupt to the Cortex-M33 NVIC. The PPC Secure PPC Interrupt Status, Clear and Enable Registers allow software to determine source of the interrupt, Clear the interrupt, and enable or disable (Mask) the interrupt.

The following table describes the bits used in the SECPPINSTAT register.

Table 3-37 PPC SECPPCINTSTAT register

Bits	Name	Access	Reset value	Description
[31:24]	Reserved	RO	0x0	Reserved.
[23:20]	S_AHBPPCEXP_STATUS	RO	0x0	Interrupt Status of Expansion Peripheral Protection Controller for AHB slaves. Each bit *n* shows the status of input signal S_AHBPPCEXP_STATUS[*n*]
[19:16]	Reserved	RO	0x0	Reserved.
[15:8]	Reserved	RO	0x0	Reserved.
[7:4]	S_APBPPCEXP_STATUS	RO	0x0	Interrupt Status of Expansion Peripheral Protection Controller for APB slaves. Each bit *n* shows the status of input signal S_AHBPPCEXP_STATUS[*n*].
[3:2]	Reserved	RO	0x0	Reserved.
1	S_APBPPC1PERIP_STATUS	RO	0x0	Interrupt Status of Peripheral Protection Controller for APB slaves within the System Control element.
0	S_APBPPC0PERIP_STATUS	RO	0x0	Interrupt Status of Peripheral Protection Controller for APB slaves within the Base element.

The following table describes the bits used in the SECPPINCLR register.

Table 3-38 SECPPCINTCLR Register

Bits	Name	Access	Reset value	Description
[31:24]	Reserved	RO	0x0	Reserved.
[23:20]	S_AHBPPCEXP_CLR	WO	0x0	Interrupt Clear of Expansion Peripheral Protection Controller for AHB slaves. Each bit *n* drives the output signal S_AHBPPCEXP_CLEAR[*n*].
[19:16]	Reserved	RO	0x0	Reserved.
[15:8]	Reserved	RO	0x0	Reserved.
[7:4]	S_APBPPCEXP_CLR	WO	0x0	Interrupt Clear of Expansion Peripheral Protection Controller for APB slaves. Each bit *n* drives the output signal S_APBPPCEXP_CLEAR[*n*].
[3:2]	Reserved	RO	0x0	Reserved.
1	S_APBPPC1PERIP_CLR	WO	0x0	Interrupt Clear of Peripheral Protection Controller for APB slaves within the System Control element. Write 1 to clear.
0	S_APBPPC0PERIP_CLR	WO	0x0	Interrupt Clear of Peripheral Protection Controller for APB slaves within the Base element. Write 1 to clear.

The following table describes the bits used in the SECPPINEN register.

Table 3-39 SECPPCINTEN Register

Bits	Name	Access	Reset value	Description
[31:24]	Reserved	RO	0x0	Reserved
[23:20]	S_AHBPPCEXP_EN	RW	0x0	Interrupt Enable of Expansion Peripheral Protection Controller for AHB slaves. Each bit *n* Enables or disable an interrupt from S_AHBPPCEXP_STATUS[*n*]
[19:16]	Reserved	RO	0x0	Reserved
[15:8]	Reserved	RO	0x0	Reserved
[7:4]	S_APBPPCEXP_EN	RW	0x0	Interrupt Enable of Expansion Peripheral Protection Controller for APB slaves. Each bit *n* Enables or disable an interrupt from S_APBPPCEXP_STATUS[*n*]
[3:2]	Reserved	RO	0x0	Reserved
1	S_APBPPC1PERIP_EN	RW	0x0	Interrupt Enable of Peripheral Protection Controller for APB slaves within the System Control element. Write 1 to enable and 0 to mask this interrupt source.
0	S_APBPPC0PERIP_EN	RW	0x0	Interrupt Enable of Peripheral Protection Controller for APB slaves within the Base element. Write 1 to enable and 0 to mask this interrupt source.

SECMSCINTSTAT, SECMSCINTCLR, and SECMSCINTEN

When security violation occurs at any Master Security Controller (MSC) in the SSE-200 and also in the expansion logic, an interrupt is raised from a combined interrupt to the CPU NVIC.

The Secure MSC Interrupt Status Clear Register and Enable Register allow software to determine source of the interrupt, clear the interrupt, and enable or disable (Mask) the interrupt.

The following table describes the bits used in the SECMSCINTSTAT register.

Table 3-40 SECMSCINTSTAT register

Bits	Name	Access	Reset value	Description
[31:16]	S_MSCEXP_STATUS	RO	0x0	Interrupt Status for Expansion MSC. Each bit *n* shows the status of input signal SMSCEXPSTATUS[*n*]. The parameter MSCEXP_DIS defines if each bit within this register is implemented so that if MSCEXP_DIS[*i*] = 1, then S_MSCEXP_STATUS[*i*] is disabled and always reads as zeros.
[15:1]	Reserved	RO	0x0	Reserved.
0	Reserved	RO	0x0	Reserved.

The following table describes the bits used in the SECMSCINTCLR register.

Table 3-41 SECMSCINTCLR Register

Bits	Name	Access	Reset value	Description
[31:16]	S_MSCEXP_CLR	WO	0x0	Interrupt Clear for Expansion MSC. Each bit *n* drives the output signal]SMSCEXPCLEAR[*n*. The parameter MSCEXP_DIS defines if each bit within this register is implemented so that if MSCEXP_DIS[*i*] = 1, then S_MSCEXP_CLR[*i*] is disabled and any writes to it are ignored.
[15:1]	Reserved	RO	0x0	Reserved.
0	Reserved	RO	0x0	Reserved.

The following table describes the bits used in the SECMSCINTEN register.

Table 3-42 SECMSCINTEN Register

Bits	Name	Access	Reset value	Description
[31:16]	S_MSCEXP_EN	RW	0x0	Interrupt Enable for Expansion MSC. Each bit *n* enables or disables the input interrupt signal]SMSCEXPSTATUS[*n*. The parameter MSCEXP_DIS defines if each bit within this register is implemented so that if MSCEXP_DIS[*i*] = 1, then S_MSCEXP_EN[*i*] is disabled and any writes to it are ignored.
[15:1]	Reserved	RO	0x0	Reserved.
0	Reserved	RO	0x0	Reserved.

BRGINTSTAT, BRGINTCLR, and BRGINTEN

AHB bus bridges are necessary to handle clock domain crossing.

To improve system performance, some of these bridges are able to buffer write data, and complete a write access on their slave interfaces before any potential error response is received for the write access on their master interfaces. When this occurs, these bridges can raise a combined interrupt to the Cortex-M33 NVIC.

The Bridge Buffer Error Interrupt Status, Clear, and Enable registers allow software to determine source of the interrupt, clear the interrupt, and enable or disable (Mask) the interrupt.

Since the secondary processor (CPU 1) can operate at a higher clock frequency, **FCLK**, compared to the main system **SYSCLK**, a bridge within the SSE-200 supports a write buffer to improve performance.

The following table describes the bits used in the BRGINTSTAT register.

Table 3-43 BRGINTSTAT register

Bits	Name	Access	Reset value	Description
[31:16]	BRGEXP_STATUS	RO	`0x0`	Interrupt Clear of Expansion Bridge Buffer Error Interrupts. Each bit *n* shows the status of BRGEXPSTATUS[*n*]. The parameter `BRGEXP_DIS` defines if each bit within this register is implemented so that if BRGEXP_DIS[`i`] = 1, then BRGEXP_STATUS[`i`] is disabled and always reads as zero.
[15:1]	Reserved	RO	`0x0`	Reserved
0	BRG_CPU1SYS_STATUS	RO	`0x0`	Interrupt Status of Write Buffer Bridge Error for Bridge between CPU1 and System. If CPU 1 does not exist, then this register is reserved, reading as zero.

The following table describes the bits used in the BRGINTCLR register.

Table 3-44 BRGINTCLR register

Bits	Name	Access	Reset value	Description
[31:16]	BRGEXP_CLR	WO	`0x0`	Interrupt Status of Expansion Bridge Buffer Error Interrupts. Each bit *n* drives the output signal BRGEXPCLEAR[*n*]. The parameter `BRGEXP_DIS` defines if each bit within this register is implemented so that if BRGEXP_DIS[`i`] = 1, then BRGEXP_CLR[`i`] is disabled and any writes to it is ignored.
[15:1]	Reserved	RO	`0x0`	Reserved
0	BRG_CPU1SYS_CLR	WO	`0x0`	Interrupt Clear of Write Buffer Bridge Error for Bridge between CPU1 and System. If CPU 1 does not exist, this register is reserved and any writes to this register are ignored.

The following table describes the bits used in the BRGINTEN register.

Table 3-45 BRGINTEN register

Bits	Name	Access	Reset value	Description
[31:17]	Reserved	RO	0x0	Reserved
[16]	BRGEXP_EN	RW	0x0	Interrupt Enable of Expansion Bridge Buffer Error Interrupts. Each bit *n* enables the input interrupt BRGEXPSTATUS[*n*]. The parameter BRGEXP_DIS defines if each bit within this register is implemented so that if BRGEXP_DIS[*i*] = 1, then BRGEXP_EN[*i*] is disabled and any writes to it are ignored.
[15:1]	Reserved	RO	0x0	Reserved
0	BRG_CPU1SYS_EN	WO	0x0	Interrupt Enable of Write Buffer Bridge Error for Bridge between CPU1 and System. If CPU 1 does not exist, this register is reserved and any writes to this register are ignored.

AHBNSPPC0

The Non-secure Access AHB Slave Peripheral Protection Controller Register allows software to configure if each AHB peripheral that it controls from an AHB 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.
- 2: Allow Secure access only.

SSE-200 does not have an AHB slave interface that needs security configuration support of the PPC. This register is reserved and RAZ/WI.

Table 3-46 AHBNSPPC0 register

Bits	Name	Access	Reset value	Description
[31:0]	Reserved	RO	0x0	Reserved.

AHBNSPPCEXP0, AHBNSPPCEXP1, AHBNSPPCEXP2, and AHBNSPPCEXP3

The Expansion Non-secure Access AHB Slave Peripheral Protection Controller registers 0, 1, 2 and 3 allow software to configure each AHB peripheral that it controls from each AHB PPC that resides in the expansion subsystem outside the SSE-200.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:
- 1: Allow Non-secure access only.
- 2: 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, N where N is from 0-3, as described in the following table.

Table 3-47 AHBNSPPCEXP0 register

Bits	Name	Access	Reset value	Description
[31:16]	Reserved	RO	0x0	Reserved.
[15:0]	AHBNSPPCEXP<N>	RW	0x0	Expansion N Non-Secure Access AHB slave Peripheral Protection Control. Each bit 'n' drives the output signal AHBNSPPCEXP<N>[n]. The parameter AHBPPCEXP_DIS<N> defines if each bit within this register is implemented so that if AHBPPCEXP_DIS<N>[i] = 1, AHBNSPPCEXP<N>[i] is disabled, reads as zeros, and any writes to it are ignored.

APBNSPPC0 and APBNSPPC1

A Non-secure Access APB slave Peripheral Protection Control Register allows software to configure if each APB peripheral that it controls from an APB 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. Secure access is not allowed.
2. Allow Secure access only. Non-secure access is not allowed.

The APBNSPPC0 register controls peripherals that are in the Base element, while APBNSPPC1 register controls peripherals that are in the System Control element.

Table 3-48 APBNSPPC0 register

Bits	Name	Access	Reset value	Description
[31:5]	Reserved	RO	0x0	Reserved
4	NS_MHU1	RW	0x0	APB access security setting for MHU 1
3	NS_MHU0	RW	0x0	APB access security setting for MHU 0
2	NS_DTIMER	RW	0x0	APB access security setting for DUAL TIMER
1	NS_TIMER1	RW	0x0	APB access security setting for TIMER 1
0	NS_TIMER0	RW	0x0	APB access security setting for TIMER 0

The following table describes the bits used in the APBSPPC1 register.

Table 3-49 APBSPPPC1 Register

Bits	Name	Access	Reset value	Description
[31:1]	Reserved	RO	0x0	Reserved
0	NS_S32K Timer	RW	0x0	S32K Timer

AHBSPPPC0

Secure Unprivileged Access AHB Slave Peripheral Protection Controller register allows software to configure if each AHB peripheral that it controls from an AHB PPC is Secure privileged, Secure Access, or if it 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.
2. Allow Secure privileged access only.

SSE-200 does not have an AHB slave interface that needs security configuration support of the PPC. This register is reserved and RAZ/WI.

Table 3-50 AHBSPPPC0 Register

Bits	Name	Access	Reset value	Description
[31:0]	Reserved	RO	0x0	Reserved

AHBSPPPCEXP0, AHBSPPPCEXP1, AHBSPPPCEXP2, and AHBSPPPCEXP3

The Expansion Secure Privilege Access AHB Slave Peripheral Protection Controller register 0, 1, 2 and 3 allow software to configure each AHB peripheral that it controls from each AHB PPC. These resides in the expansion subsystem outside of the SSE-200, and allow Secure privileged Access only or both Secure unprivileged and privlegded access.

Each field defines this for an associated peripheral, by the following settings:

- 1: Allow Secure unprivileged and privileged access.
- 2: 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, *N* where *N* is from 0 to 3, is as described in the following table.

Table 3-51 AHBSPPPCEXP0 register

Bits	Name	Access	Reset value	Description
[31:16]	Reserved	RO	0x0	Reserved
[15:0]	AHBSPPPCEXP<*N*>	RW	0x0	Expansion N Secure Privilege Access AHB slave Peripheral Protection Control. Each bit *n* drives the output signal AHBPPPCEXP<*N*>[*n*] if AHBNSPPCEXP<*N*>[*n*] is LOW, where N is 0 to 3. The parameter AHBPPCEXP_DIS<N> defines if each bit within this register is implemented so that if AHBPPCEXP_DIS<*N*>[*i*] = 1, AHBSPPPCEXP<*N*>[*i*] is disabled, it reads as zeros, and any writes to it are ignored.

APBSPPPC0 and APBSPPPC1

Each APB peripheral controlled by an APB PPC is either Secure privileged access only or Secure unprivileged access is also enabled. Each field defines this for an associated peripheral by the following:

1. Enable Secure unprivileged and privileged access.
2. Enable Secure privileged access only.

Table 3-52 APBSPPPC0 register

Bits	Name	Access	Reset value	Description
[31:5]	Reserved	RO	0x0	Reserved
4	SP_MHU1	RW	0x0	APB access Secure privileged setting for MHU 1
3	SP_MHU0	RW	0x0	APB access Secure privileged setting for MHU 0
2	SP_DTIMER	RW	0x0	APB access Secure privileged setting for DUAL TIMER
1	SP_TIMER1	RW	0x0	APB access Secure privileged setting for TIMER 1
0	SP_TIMER0	RW	0x0	APB access Secure privileged setting for TIMER 0

Table 3-53 APBSPPPC1 Register

Bits	Name	Access	Reset value	Description
[31:1]	Reserved	RO	0x0	Reserved
0	SP_S32KTIMER	RW	0x0	APB access Secure privileged setting for S32KCLK Timer

APBSPPPCEXP0, APBSPPPCEXP1, APBSPPPCEXP2, and APBSPPPCEXP3

The Expansion Secure Privilege Access APB Slave Peripheral Protection Controller register 0, 1, 2 and 3 allow software to configure each APB peripheral that it controls from each APB PPC. These reside in the expansion subsystem outside the SSE-200, allowing Secure privileged access only or both Secure unprivileged and privileged access.

Each field defines this for an associated peripheral, by the following settings:

- 1: Allow Secure unprivileged and privileged access.
- 2: Allow Secure privileged access only.

These controls directly control the expansion signals on the Security Control Expansion interface. All four registers are similar and each register, *N* where *N* is from 0-3,are listed in the following table.

Table 3-54 APBSPPPCEXP0 register

Bits	Name	Access	Reset value	Description
[31:16]	Reserved	RO	0x0	Reserved
[15:0]	APBSPPPCEXP<*N*>	RW	0x0	Expansion *N* Secure Privilege Access APB slave Peripheral Protection Control. Each bit 'n' drives the output signal APB_P_PPCEXP<*N*>[n] if APBNSPPCEXP<*N*>[n] is LOW, where *N* is 0 to 3. The parameter APBPPCEXP_DIS<N> defines if each bit within this register is implemented so that if APBPPCEXP_DIS<*N*>[i] = 1, APBSPPPCEXP<*N*>[i] is disabled, it reads as zeros, and any writes to it are ignored.

NSMSCEXP

The Non-secure Expansion Master Security Controller register allows software to configure if each master that is located behind each MSC in the expansion subsystem is a Secure or Non-secure device.

Table 3-55 NSMSCEXP register

Bits	Name	Access	Reset value	Description
[31:16]	NS_MSCEXP	RW	0x0	Expansion MSC Non-secure Configuration. Each bit 'n' controls the Non-secure configuration of each MSC and drives the signals NS_MSCEXP[n]. Set to HIGH to define a Master as Non-secure. Else it is Secure. The parameter MSCEXP_DIS<N> defines if each bit within this register is implemented so that if MSCEXP_DIS<N>[i] = 1, NS_MSCEXP[i] is disabled, it reads as zeros, and any writes to it are ignored.
[15:0]	Reserved	RO	0x0	Reserved

3.4.7 Non-secure Privilege Control Block

The Non-secure Privilege Control Block implements program visible states that allow software to control various security gating units within the design.

This register is Non-secure privileged access only and supports 32-bit R/W access. Any byte and halfword writes result in its write data being ignored.

This register block base address is `0x4008_0000`. The following table lists the Non-secure privilege control registers.

Table 3-56 Summary of Non-secure Privilege Control registers

Offset	Name	Access	Reset value	Description
`0c000` – `0x06C`	Reserved	-	-	Reserved.
`0x090`	AHBNSPPPC0	RW	`0x0`	Non-secure Unprivileged Access AHB slave Peripheral Protection Control 0. Each bit in this register defines the Non-secure unprivileged access settings for an associated AHB slave peripheral.
`0x094` – `0x09C`	Reserved	RO	`0x0`	Reserved.
`0x0A0`	AHBNSPPPCEXP0	RW	`0x0`	Expansion 0 Non-secure Unprivileged Access AHB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0A4`	AHBNSPPPCEXP1	RW	`0x0`	Expansion 1 Non-secure Unprivileged Access AHB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0A8`	AHBNSPPPCEXP2	RW	`0x0`	Expansion 2 Non-secure Unprivileged Access AHB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0AC`	AHBNSPPPCEXP3	RW	`0x0`	Expansion 3 Non-secure Unprivileged Access AHB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0B0`	APBNSPPPC0	RW	`0x0`	Non-secure Unprivileged Access APB slave Peripheral Protection Control 0. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0B4`	APBNSPPPC1	RW	`0x0`	Non-secure Unprivileged Access APB slave Peripheral Protection Control 0. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0B8` – `0x0BC`	Reserved	-	`0x0`	Reserved
`0x0C0`	APBNSPPPCEXP0	RW	`0x0`	Expansion 0 Non-secure Unprivileged Access APB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0C4`	APBNSPPPCEXP1	RW	`0x0`	Expansion 1 Non-secure Unprivileged Access APB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0C8`	APBNSPPPCEXP2	RW	`0x0`	Expansion 2 Non-secure Unprivileged Access APB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0CC`	APBNSPPPCEXP3	RW	`0x0`	Expansion 3 Non-secure Unprivileged Access APB slave Peripheral Protection Control. Each bit in this register defines the Non-secure unprivileged access settings for an associated peripheral.
`0x0D0` – `0xFCC`	Reserved	-	`0x0`	Reserved
`0xFD0`	PIDR4	RO	`0x0`	Peripheral ID 4
`0xFD4`	PIDR5	RO	`0x0`	Reserved

Table 3-56 Summary of Non-secure Privilege Control registers (continued)

Offset	Name	Access	Reset value	Description
0xFD8	PIDR6	RO	0x0	Reserved
0xFDC	PIDR7	RO	0x0	Reserved
0xFE0	PIDR0	RO	0x0000_0053	Peripheral ID 0
0xFE4	PIDR1	RO	0x0000_00B8	Peripheral ID 1
0xFE8	PIDR2	RO	0x0000_000B	Peripheral ID 2
0xFEC	PIDR3	RO	0x0000_0000	Peripheral ID 3
0xFF0	CIDR0	RO	0x0000_000D	Component ID 0
0xFF4	CIDR1	RO	0x0000_00F0	Component ID 1
0xFF8	CIDR2	RO	0x0000_0005	Component ID 2
0xFFC	CIDR3	RO	0x0000_00B1	Component ID 3

AHBNSPPPC0

Non-secure Unprivileged Access AHB Slave Peripheral Protection Controller Register allows software to configure if each AHB peripheral that it controls from an AHB PPC is Non-secure privileged access only or is allowed Non-secure Unprivileged access. Each field defines this for an associated peripheral, by the following settings:

- 1: Allow Secure unprivileged and privileged access.
- 2: Allow Secure privileged access only.

SSE-200 does not have an AHB slave interface that needs Non-secure Unprivileged Access configuration support of the PPC. This register is reserved and RAZ/WI.

Table 3-57 AHBNSPPPC0 register

Bits	Name	Access	Reset value	Description
[31:0]	Reserved	RO	0x0	Reserved

AHBNSPPPCEXP0, AHBNSPPPCEXP1, AHBNSPPPCEXP2, and AHBNSPPPCEXP3

The Expansion Non-secure Privilege Access AHB Slave Peripheral Protection Controller Register 0, 1, 2 and 3 allow software to configure each AHB peripheral that it controls from each AHB PPC. These reside in the expansion subsystem outside of the SSE-200, and only Non-Secure privileged access only or both Non-Secure unprivileged and privileged access are allowed. Each field defines this for an associated peripheral, by the following settings:

1. Allow Secure unprivileged and privileged access.
2. Allow Secure privileged access only.

These directly control the expansion signals on the Security Control Expansion interface.

All four registers are similar. The bits for register *N*, where *N* is from 0-3, are listed in the following table.

Table 3-58 AHBNSPPPCEXP0 register

Bits	Name	Access	Reset value	Description
[31:16]	Reserved	RO	0x0	Reserved
[15:0]	AHBNSPPPCEXP<*N*>	RW	0x0	Expansion N Non-secure Privilege Access AHB slave Peripheral Protection Control. Each bit *n* drives the output signal AHBPPPCEXP<N>[*n*] if AHBNSPPCEXP<N>[n] is HIGH, where N is 0-3. The parameter AHBPPCEXP_DIS<N> defines if each bit within this register is implemented so that if AHBPPCEXP_DIS<N>[i] = 1, AHBNSPPPCEXP<N>[i] is disabled, it reads as zeros, and any writes to it are ignored.

APBNSPPPC0 and APBNSPPPC1

Non-secure Unprivileged Access APB Slave Peripheral Protection Controller Register allows software to configure if each APB peripheral that it controls (by an APB PPC) is Non-secure privileged access only or is also allowed Non-secure Unprivileged access.

Table 3-59 APBNSPPPC0 register

Bits	Name	Access	Reset value	Description
[31:5]	Reserved	RO	0x0	Reserved
4	NS_MHU1	RW	0x0	APB access Non-secure privileged setting for MHU 1
3	NS_MHU0	RW	0x0	APB access Non-secure privileged setting for MHU 0
2	NSP_DTIMER	RW	0x0	APB access Non-secure privileged setting for DUAL TIMER
1	NSP_TIMER1	RW	0x0	APB access Non-secure privileged setting for TIMER 1
0	NSP_TIMER0	RW	0x0	APB access Non-secure privileged setting for TIMER 0

Table 3-60 APBNSPPPC1 register

Bits	Name	Access	Reset value	Description
[31:1]	Reserved	RO	0x0	Reserved
0	NSP_S32KTIMER	RW	0x0	APB access Non-secure privileged setting for S32KCLK Timer

APBNSPPPCEXP0, APBNSPPPCEXP1, APBNSPPPCEXP2, and APBNSPPPCEXP3

The Expansion Non-secure Privilege Access APB Slave Peripheral Protection Controller Register 0, 1, 2 and 3 allow software to configure each APB peripheral that it controls from each APB PPC, that resides in the expansion subsystem outside the SSE-200, is Non-secure privileged access only or is allowed Non-secure Unprivileged access.

Each field defines this for an associated peripheral, by the following settings:

- 1: Allow Non-secure unprivileged and privileged access.
- 2: Allow Non-secure privileged access only.

These controls directly control the expansion signals on the Security Control Expansion interface. All four registers are similar and each register, N where N is from 0-3, is listed in the following table.

Table 3-61 APBNSPPPCEXP0 register

Bits	Name	Access	Reset value	Description
[31:16]	Reserved	RO	0x0	Reserved
[15:0]	APBNSPPPCEXP<N>	RW	0x0	Expansion N Non-secure Privilege Access APB slave Peripheral Protection Control. Each bit *n* drives the output signal APBPPPCEXP<N>[*n*] if APBNSPPCEXP<N>[*n*] is HIGH, where N is 0-3. The parameter APBPPCEXP_DIS<N> defines if each bit within this register is implemented so that if APBPPCEXP_DIS<N>[i] = 1, APBNSPPPCEXP<N>[i] is disabled, it reads as zeros, and any writes to it are ignored.

Related references

3.4.1 CMSDK timer on page 3-92.
3.4.2 CMSDK dual timer on page 3-93.
3.4.3 CMSDK watchdog timers on page 3-94.
3.4.4 AHB5 TrustZone Memory Protection Controller on page 3-96.
3.4.5 Message handling unit on page 3-99.
3.4.6 Security Privilege Control Block on page 3-101.
3.4.7 Non-secure Privilege Control Block on page 3-116.

3.5 SRAM element

Up to four SRAM elements can be present. The MPCs in the Base element control security for the SRAM regions.

Up to four *Memory Protection Controllers* (MPC) are included, one on each path to a SRAM block so that accesses can be blocked when a security violation occurs.

Each SRAM block is implemented within an SRAM element. Each MPC APB configuration interface is mapped to the following base addresses:

- `0x5008_3000` for SRAM Bank 0.
- `0x5008_4000` for SRAM Bank 1.
- `0x5008_5000` for SRAM Bank 2.
- `0x5008_6000` for SRAM Bank 3.

If any of the SRAM banks do not exist, the associated MPC does not exist and the address area of that MPC is reserved. Any access to it is RAZ/WI.

The cfg_init_value of each MPC is tied to 0 so that at boot, the SRAM is Secure only. Software must change or restore the settings in the MPC to release memory for Non-secure world use.

nWARMRESETSYS resets all SRAM MPCs, which reside in the PD_SYS power domain.

Power management for the SRAM elements is from the PIKs which are controlled from the system control element.

3.6 System control element

This section describes the registers that are associated with controlling the SSE-200.

For information on the Cortex-M33 registers, see the following documents:

- *Arm® Cortex®-M System Design Kit Technical Reference Manual*.
- *Arm® Cortex®-M33 Processor Technical Reference Manual*.
- *Arm®v7-M Architecture Reference Manual*.

This section contains the following subsections:

3.6.1 System control registers

The System Control Region contains the peripherals in the System Control element.

The System Control Region occupies the following areas:

- `0x4002_0000` to `0x4003_FFFF`, which is Non-secure
- `0x5002_0000` to `0x5003_FFFF`, which is Secure.

Table 3-62 System control regions

Row ID (alias)	Address		Size	Region name	Description	Security
	From	To				
1 (5)	`0x4002_0000`	`0x4002_0FFF`	4KB	SYSINFO	System Information Registers Block.	NS
2	`0x4002_1000`	`0x4002_EFFF`		Reserved	Reserved[a]	
3 (18)	`0x4002_F000`	`0x4002_FFFF`	4KB	S32KTIMER	CMSDK Timer running on **S32KCLK**.	NS-PPC
4	`0x4003_0000`	`0x4003_FFFF`		Reserved	Reserved	
5 (1)	`0x5002_0000`	`0x5002_0FFF`	4KB	SYSINFO	System Information Registers Block.	S
6	`0x5002_1000`	`0x5002_1FFF`	4KB	S_SYSCONTROL	System Control Registers Block.	SP
7	`0x5002_2000`	`0x5002_2FFF`	4KB	SYS_PPU	System Power Policy Unit.	SP
8	`0x5002_3000`	`0x5002_3FFF`	4KB	CPU0CORE_PPU	CPU 0 Core Power Policy Unit.	SP
9	`0x5002_4000`	`0x5002_4FFF`	4KB	CPU0DEBUG_PPU[b]	CPU 0 Debug Power Policy Unit.	SP
10	`0x5002_5000`	`0x5002_5FFF`	4KB	CPU1CORE_PPU	CPU 1 Core Power Policy Unit.	SP
11	`0x5002_6000`	`0x5002_6FFF`	4KB	CPU1DEBUG_PPU[b]	CPU 1 Debug Power Policy Unit.	SP
12	`0x5002_7000`	`0x5002_7FFF`	4KB	CRYPTO_PPU	CryptoCell Power Policy Unit.	SP
-	`0x5002_8000`	`0x5002_8FFF`	4KB	Reserved	Reserved[c]	
13	`0x5002_9000`	`0x5002_9FFF`	4KB	DEBUG_PPU	System Debug Power Policy Unit.	SP
14	`0x5002_A000`	`0x5002_AFFF`	4KB	RAM0_PPU	SRAM Bank 0 Power Policy Unit.	SP

[a] This region is RAZ/WI.
[b] CPU0DEBUG_PPU and CPU1DEBUG_PPU regions do not exist if SEPARATE_CPUDEBUG_PD configuration is False, indicating that separate CPU debug power domains is not supported. If they do not exist, these regions are RAZ/WI.
[c] This region is RAZ/WI.

Table 3-62 System control regions (continued)

Row ID (alias)	Address		Size	Region name	Description	Security
	From	To				
15	0x5002_B000	0x5002_BFFF	4KB	RAM1_PPU	SRAM Bank 1 Power Policy Unit.	SP
16	0x5002_C000	0x5002_CFFF	4KB	RAM2_PPU	SRAM Bank 2 Power Policy Unit.	SP
17	0x5002_D000	0x5002_DFFF	4KB	RAM3_PPU	SRAM Bank 3 Power Policy Unit.	SP
18	0x5002_E000	0x5002_EFFF	4KB	S32KWATCHDOG	CMSDK Watchdog on S32KCLK.	SP
19 (3)	0x5002_F000	0x5002_FFFF	4KB	S32KTIMER	CMSDK Timer on S32KCLK.	S-PPC
20	0x5003_0000	0x5003_FFFF		Reserved	Reserved.	

Note

- For NS_PPC, any Secure access targeting these regions is blocked. PPCs control Non-secure access to these regions.
- For S_PPC, any Non-Secure access targeting this region is blocked. PPCs control Secure access to this region.
- NSP indicates Non-secure private access only.
- SP indicates Secure privilege access only.
- S indicates Secure access only.
- Reserved regions respond with RAZ/WI when accessed.

3.6.2 System information registers

The System Information Register Block provides information on the system configuration and identity. This register block is read-only and accessible by accesses of any security attributes.

This module resides at base address 0x5002_0000 in the Secure region, and 0x4002_0000 in the Non-secure region of the Base Peripheral Region.

Note

The System information registers block is mapped to both the Secure and Non-secure regions and is visible to both regions without any security protection.

Table 3-63 Summary of System Information Block registers

Offset	Name	Access	Reset value	Description	Security
0x000	SYS_VERSION	RO	0x2004_1743	System Version register	All
0x004	SYS_CONFIG	RO	System configuration dependent	System Hardware Configuration register	All
0x010 – 0xFCC	Reserved	-	-	-	-
0xFD0	PIDR4	RO	0x0000_0004	Peripheral ID 4	All
0xFD4	PIDR5	RO	0x0	Reserved	-
0xFD8	PIDR6	RO	0x0	Reserved	-
0xFDC	PIDR7	RO	0x0	Reserved	-
0xFE0	PIDR0	RO	0x0000_0058	Peripheral ID 0	All
0xFE4	PIDR1	RO	0x0000_00B8	Peripheral ID 1	All

Table 3-63 Summary of System Information Block registers (continued)

Offset	Name	Access	Reset value	Description	Security
0xFE8	PIDR2	RO	0x0000_000B	Peripheral ID 2	All
0xFEC	PIDR3	RO	0x0000_0000	Peripheral ID 3	All
0xFF0	CIDR0	RO	0x0000_000D	Component ID 0	All
0xFF4	CIDR1	RO	0x0000_00F0	Component ID 1	All
0xFF8	CIDR2	RO	0x0000_0005	Component ID 2	All
0xFFC	CIDR3	RO	0x0000_00B1	Component ID 3	All

SYS_VERSION

The System Version register enables software to read the system part number and revision.

Table 3-64 SYS_VERSION Register

Bits	Name	Access	Reset value	Description
[31:28]	CONFIGURATION	RO	0x2	Set to 0x2 for SSE-200 r1.
[27:24]	MAJOR_REVISION	RO	0x0	Set to 0x0.
[23:20]	MINOR_REVISION	RO	0x0	Set to 0x0.
[19:12]	DESIGNER_ID	RO	0x41	Arm Product with designer code 0x41.
[11:0]	PART_NUMBER	RO	0x743	Part Number for the SSE-200.

SYS_CONFIG

The System Hardware Configuration register enables software to determine the system configuration.

Table 3-65 SYS_CONFIG Register

Bits	Name	Access	Reset value	Description
[31:28]	CPU1_TYPE	RO	0b0010	CPU 1 Core Type: 0b0000 Does not exist. 0b0010 Cortex-M33 core. Other Reserved.
[27:24]	CPU0_TYPE	RO	0b0010	CPU 0 Core Type: 0b0000 Does not exist. 0b0010 Cortex-M33 core. Other Reserved.

Table 3-65 SYS_CONFIG Register (continued)

Bits	Name	Access	Reset value	Description
[23:20]	CPU1_TCM_BANK_NUM	RO	`0x3` if 4 SRAM banks. `0x2` if 3 SRAM banks. `0x1` if 2 SRAM banks. Otherwise `0x0`.[a]	Number of SRAM banks: `b11` if 4 SRAM banks. `b10` if 3 SRAM banks. `b01` if 2 SRAM banks. Otherwise `b00`. The SRAM Bank that maps CPU1 Data TCM.
[19:16]	CPU0_TCM_BANK_NUM	RO	`0x0`	The SRAM Bank that maps CPU0 Data TCM.
[15:13]	Reserved	RO	`0b000`	Reserved.
12	HAS_CRYPTO	RO	Configuration dependent	CryptoCell Included: 0 No. 1: Yes.
11	Reserved	RO	`0b0`	Reserved.
10	CPU1_HAS_TCM	RO	'1' if CPU1 exist, otherwise '0'. [a]	CPU 1 has Data TCM: 0: No. 1: Yes.
9	CPU0_HAS_TCM	RO	`0x0`	CPU 0 has Data TCM: 0: No. 1: Yes.
[8:4]	SRAM_ADDR_WIDTH	RO	Configuration dependent	SRAM Bank Address Width. The size of each SRAM bank is equal to $2^{SRAM\_ADDR\_WIDTH}$. Supported values are in the range of: Minimum of 10: 1KB. Maximum of 24: 16MB (only if 1 SRAM element exists).
[3:0]	SRAM_NUM_BANK	RO	`0b0100`	SRAM Number of Banks: SSE-200 supports a minimum of 1 for a single processor configuration and 2 for a dual processor configuration.

3.6.3 CMSDK timer

The System Control element implements:

- • A single CMSDK Timer that reside in non-Secure region at `0x4002_F000` and in Secure region at `0x5002_F000`.
- A single CMSDK Watchdogs that resides in the Secure region at `0x5002_E000`.

See *3.4.1 CMSDK timer* on page 3-92 for a summary of the control registers.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

[a] These are derived from other system configurations.

Related references

3.4.1 CMSDK timer on page 3-92.

3.6.4 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 `0x5002_1000` in the Secure region of the base peripheral region. The System Control Register Block is Secure privilege access only.

For write access to these registers, only 32-bit writes are supported. Any byte or halfword writes result in the write data being ignored. The following table lists the registers in this block.

Table 3-66 Summary of System Control registers

Offset	Name	Access	Reset	Description
`0x000`	SECDBGSTAT	RO	`0x0000_0000`	Secure Debug Configuration Status.
`0x004`	SECDBGSET	WO	`0x0000_0000`	Secure Debug Configuration Set.
`0x008`	SECDBGCLR	WO	`0x0000_0000`	Secure Debug Configuration Clear.
`0x00C`	SCSECCTRL	RW	`0x0000_0000`	System Control Security Control.
`0x010`	FCLK_DIV	RW	Parameterized	Fast Clock Divider Configuration.
`0x014`	SYSCLK_DIV	RW	Parameterized	System Clock Divider Configuration.
`0x018`	CLOCK_FORCE	RW	`0x0000_0000`	Clock Force.
`0x01C – 0x0FC`	-	RO	`0x0000_0000`	Reserved.
`0x100`	RESET_SYNDROME	RW	`0x0000_0001`	Reset Syndrome. Register only cleared at Power-on Reset.
`0x104`	RESET_MASK	RW	`0x0000_0030`	Reset Mask.
`0x108`	SWRESET	WO	`0x0000_0000`	Software Reset.
`0x10C`	GRETREG	RW	`0x0000_0000`	General Purpose Retention.
`0x110`	INITSVRTOR0	RW	Static tied input	Initial Secure Reset Vector Register For CPU 0.
`0x114`	INITSVRTOR1	RW	Static tied input	Initial Secure Reset Vector Register For CPU 1.
`0x118`	CPUWAIT	RW	Parameterized	CPU Boot wait control after reset.
`0x11C`	NMI_ENABLE	RW	`0x0000_0000`	NMI Enable Register.
`0x120`	WICCTRL	RW	`0x0000_0000`	WIC request and acknowledge handshake.
`0x124`	EWCTRL	RW	`0x0000_0000`	External Wakeup Control.
`0x128 – 0x1FF`	-	-	`0x0000_0000`	Reserved.
`0x200`	PDCM_PD_SYS_SENSE	RW	`0x0000_007F`	Power Control Depedendency Matrix PD_SYS Power Domain Sensitivity.
`0x204`	-	-	`0x0000_0000`	Reserved.
`0x208`	-	-	`0x0000_0000`	Reserved.
`0x20C`	PDCM_PD_SRAM0_SENSE	RW	`0x0000_0000`	Power Control Depedendency Matrix PD_SRAM0 Power Domain Sensitivity.

Table 3-66 Summary of System Control registers (continued)

Offset	Name	Access	Reset	Description
0x210	PDCM_PD_SRAM1_SENSE	RW	0x0000_0000	Power Control Depedendency Matrix PD_SRAM1 Power Domain Sensitivity.
0x214	PDCM_PD_SRAM2_SENSE	RW	0x0000_0000	Power Control Depedendency Matrix PD_SRAM2 Power Domain Sensitivity.
0x218	PDCM_PD_SRAM3_SENSE	RW	0x0000_0000	Power Control Depedendency Matrix PD_SRAM3 Power Domain Sensitivity.
0x21C – 0x22C	-	-	-	Reserved.
0x230	-	-	0x0000_0000	Reserved.
0x234 – 0x23C	-	-	0x0000_0000	Reserved.
0x240 – 0x24C	-	-	0x0000_0000	Reserved.
0x250 – 0xFCC	-	-	-	Reserved.
0xFD0	PIDR4	RO	0x0000_0004	Peripheral ID 4.
0xFD4	PIDR5	RO	0x0000_0000	Reserved.
0xFD8	PIDR6	RO	0x0000_0000	Reserved.
0xFDC	PIDR7	RO	0x0000_0000	Reserved.
0xFE0	PIDR0	RO	0x0000_0054	Peripheral ID 0.
0xFE4	PIDR1	RO	0x0000_00B8	Peripheral ID 1.
0xFE8	PIDR2	RO	0x0000_000B	Peripheral ID 2.
0xFEC	PIDR3	RO	0x0000_0000	Peripheral ID 3.
0xFF0	CIDR0	RO	0x0000_000D	Component ID 0.
0xFF4	CIDR1	RO	0x0000_00F0	Component ID 1.
0xFF8	CIDR2	RO	0x0000_0005	Component ID 2.
0xFFC	CIDR3	RO	0x0000_00B1	Component ID 3.

SECDBGSTAT, SECDBGSET, and SECDBGCLR

The Secure Debug Configuration registers are used to select the source value for the combined Secure Debug Authentication Signals **DBGEN**, **NIDEN**, **SPIDEN**, and **SPNIDEN**.

For each signal, a selector is provided to select between an internal register value and the value on the boundary of the SSE-200.

Secure software can set each value by setting the associated bit in the SECDBGSET register or clear a value by setting the associated bit in the SECDBGCLR register.

Secure software can read the system-wide value by reading the associated SECDBGSTAT register bit.

For example, DBGEN_SEL selects the source of the **DBGEN** value used in the system, where:

- DBGEN_SEL is LOW, the input **DBGENIN** signal is used to define the system-wide DBGEN value.
- DBGEN_SEL is HIGH, the internal register value DBGEN_I is used to define the system-wide DBGEN value.

For example:

- To set DBGEN_I to HIGH, write to the SECDBGSET register with DBGEN_I_SET set HIGH.
- To set DBGEN_SEL to HIGH, write to the SECDBGSET register with DBGEN_SEL_SET set to HIGH.
- To set DBGEN_I to LOW, write to the SECDBGCLR register with DBGEN_I_CLR set to HIGH.
- To set DBGEN_SEL to LOW, write to the SECDBGCLR register with DBGEN_SEL_CLR set to HIGH.
- To read the value of DBGEN, read the SECDBGSTAT register for the DBGEN_SEL_STAT value.

The DGBEN value is also made available to external expansion logic from the DBGEN output signal of the SSE-200.

Only **nPORESETAON** resets these registers.

Top-level static configuration signals **DBGENSELDIS**, **NIDENSELDIS**, **SPIDENSELDIS**, and **SPNIDENSELDIS** are provided to enable each of the selectors, DBGEN_SEL_STATUS, NIDEN_SEL_STATUS, SPIDEN_SEL_STATUS, and SPNIDEN_SEL_STATUS respectively to be forced to zero, forcing each respective input to use its external value. This method can be used to prevent Secure firmware from modifying or overriding the Debug Authentication value, in particular, when the Crypto element is present (when HAS_CRYPTO is True) in the system and the intention is to use signals that are derived from **CCDCUEN** to control the Debug Authentication signal.

The following table lists the bits for the Secure Debug Configuration Status register.

Table 3-67 SECDBGSTAT register

Bits	Name	Access	Reset value	Description
[31:10]	Reserved	RO	0x0	Reserved
7	SPNIDEN_SEL_STATUS	RO	0x0	Active HIGH Secure privilege non-invasive debug enable selector value.
6	SPNIDEN_STATUS	RO	SPNIDEN_IN1[a]	Active HIGH Secure privilege non-invasive debug enable value.
5	SPIDEN_SEL_SET	RO	0x0	Active HIGH Secure privilege invasive debug enable selector value.
4	SPIDEN_I_SET	RO	SPIDEN_IN[a]	Active HIGH Secure privilege invasive debug enable value.
3	NIDEN_SEL_SET	RO	0x0	Active HIGH non-invasive debug enable selector value.
2	NIDEN_I_SET	RO	NIDEN_IN[a]	Active HIGH non-invasive debug enable value.
1	DBGEN_SEL_SET	RO	0x0	Active HIGH debug enable selector value.
0	DBGEN_I_SET	RO	DBGEN_IN[a]	Active HIGH debug enable value.

The table lists the bits for the Secure Debug Configuration Set register:

Table 3-68 SECDBGSET register

Bits	Name	Access	Reset value	Description
[31:10]	Reserved	RO	0x0	Reserved
7	SPNIDEN_SEL_SET	WO	0x0	Active HIGH Secure privilege non-invasive debug enable selector set control.
6	SPNIDEN_I_SET	WO	0x0	Active HIGH Secure privilege non-invasive debug enable set control.
5	SPIDEN_SEL_SET	WO	0x0	Active HIGH Secure privilege invasive debug enable selector set control.

[a] DBGEN_IN, NIDEN_IN, SPIDEN_IN and SPNIDEN_IN are input signals on the Debug Authentication Interface.

Table 3-68 SECDBGSET register (continued)

Bits	Name	Access	Reset value	Description
4	SPIDEN_I_SET	WO	0x0	Active HIGH Secure privilege invasive debug enable set control.
3	NIDEN_SEL_SET	WO	0x0	Active HIGH non-invasive debug enable selector set control.
2	NIDEN_I_SET	WO	0x0	Active HIGH non-invasive debug enable set control.
1	DBGEN_SEL_SET	WO	0x0	Active HIGH debug enable selector set control.
0	DBGEN_I_SET	WO	0x0	Active HIGH debug enable set control.

The table lists the bits for the Secure Debug Configuration clear register:

Table 3-69 SECDBGCLR register

Bits	Name	Access	Reset value	Description
[31:10]	Reserved	RO	0x0	Reserved
7	SPNIDEN_SEL_CLR	WO	0x0	Active HIGH Secure privilege non-invasive debug enable selector clear control.
6	SPNIDEN_I_CLR	WO	0x0	Active HIGH Secure privilege non-invasive debug enable clear control.
5	SPIDEN_SEL_CLR	WO	0x0	Active HIGH Secure privilege invasive debug enable selector clear control.
4	SPIDEN_I_CLR	WO	0x0	Active HIGH Secure privilege invasive debug enable clear control.
3	NIDEN_SEL_CLR	WO	0x0	Active HIGH non-invasive debug enable selector clear control.
2	NIDEN_I_CLR	WO	0x0	Active HIGH non-invasive debug enable clear control.
1	DBGEN_SEL_CLR	WO	0x0	Active HIGH debug enable selector clear control.
0	DBGEN_I_CLR	WO	0x0	Active HIGH debug enable clear control.

SCSECCTRL

The System Control Security Controls provide register bits to configure the Certificate access path and the Secure Configuration lock of this register block.

nPORESETAON resets the register.

Table 3-70 SCSECCTRL register

Bits	Name	Access	Reset value	Description
[31:18]	Reserved	RO	0x0	-
[15:3]	Reserved	RO	0x0	-
17	CERTREADENABLED	RO	0x0	Active HIGH status that indicates that the certification read access is enabled. This status is also set to LOW whenever the PD_DEBUG Power Domain is in the OFF state. It is also set to LOW when CERTDISABLED is HIGH.
16	CERTDISABLED	RO	0x0	Active HIGH status that indicates that the Certification write path has been disabled. This status is also set to HIGH whenever the PD_DEBUG Power Domain is in the OFF state.

Table 3-70 SCSECCTRL register (continued)

Bits	Name	Access	Reset value	Description
2	SCSECCFGLOCK	Write 1 to set.	0x0	Active HIGH control to disable writes to security-related control registers in this register block. After setting HIGH, it can no longer be cleared to zero except by a Power-on reset. When set to HIGH, write access to SECDBGSET, SECDBGCLR, INITSVTOR0 and INITSVTOR1 are ignored.
1	CERTREADEN	RW	0x0	Active HIGH control to enable read access on the certification path as long as CERTDISABLE is also LOW. If CERTDISABLE is HIGH, read access is disabled on the certification path and CERTREADEN value is ignored.
0	CERTDISABLE	Write 1 to set.	0x0	Active HIGH control to disable certification path. After setting HIGH, this bit cannot be cleared to zero except by a Power-on reset. It can also be set HIGH by: • Driving **CERTDISABLEEXT** HIGH. • Turning PD_DEBUG OFF. • Changing PD_CPU0CORE power domain from ON.

FCLK_DIV

The Fast Clock Divider register allows software to configure the divider ratio that generates **FCLK** from **MAINCLK**. After writing to the FCLKDIV field, always check that the new clock divider ratio has been applied by reading it back on FCLKDIV_CUR before doing any other operations.

nPORESETAON resets the register.

Table 3-71 FCLK_DIV register

Bits	Name	Access	Reset value	Description
[31:21]	Reserved	RO	0x0	-
[20:16]	FCLKDIV_CUR	RO	0x0	Clock Divider Current Value. This field returns the currently selected clock divider value FCLKDIV used to generate **FCLK** from **MAINCLK**. The clock divider setting is the value of FCLKDIV + 1. For example, setting a value of 0 indicates a divider value of 1.
[15:5]	Reserved	RO	0x0	-
[4:0]	FCLKDIV	RW	FCLKDIV_RST	**FCLK** from **MAINCLK** Clock Divider Ratio Request. The clock divider setting is the value of FCLKDIV + 1. For example, setting a value of 0 indicates a divider value of 1.

SYSCLK_DIV

The System Clock Divider register allows software to configure the divider ratio that generates **SYSCLK** from **FCLK**.

After writing to the SYSCLKDIV field, always check that the new clock divider ratio has been applied by reading it back on SYSCLKDIV_CUR before doing any other operations.

nPORESETAON resets the register.

Table 3-72 System Clock Divider register

Bits	Name	Access	Reset value	Description
[31:21]	Reserved	RO	0x0	-
[20:16]	SYSCLKDIV_CUR	RO	0x0	Clock Divider Current Value. This field returns the currently selected clock divider value SYSCLKDIV used to generate **SYSCLK** from FCLK. The clock divider setting is the value of SYSCLKDIV_CUR + 1. For example, setting a value of 0 indicates a divider value of 1.
[15:5]	Reserved	RO	0x0	-
[4:0]	SYSCLKDIV	RW	SYSCLKDIV_RST	**SYSCLK** from FCLK Clock Divider Ratio Request. The clock divider setting requested is the value of SYSCLKDIV + 1. For example, setting a value of 0 indicates a divider value of 1.

CLOCK_FORCE

The Clock Force register allows software to override dynamic clock gating that might be implemented in the system and keep each clock running.

Note

Clock force signals do not apply to clock gates that compliment power gating. Instead, it is applied to hierarchical dynamic clock gating within the system. Forcing a clock ON can reduce the latency that is incurred as a result of dynamic clock control, but can also increase the dynamic power consumption of the system.

nPORESETAON resets the register.

Table 3-73 CLOCK_FORCE register

Bits	Name	Access	Reset value	Description
[31:9]	Reserved	RO	0x0	-
8	FCLKHINTGATE_ENABLE	RW	0x0	Enable **FCLK** gating by **HINTSYSCLKEN** when CPU 1 is OFF. Clear this bit to LOW to improve SRAM3 access latency at the cost of increased power consumption.
7	CRYPTOSYSCLK_FORCE	RW	0x0	Force all CryptoCell clocks to run when set to HIGH.
6	CPUFCLK_FORCE	RW	0x0	Force all CPU **FCLK** to run when set to HIGH.
5	CPUSYSCLK_FORCE	RW	0x0	Force all CPU **SYSCLK** to run when set to HIGH.
4	SRAMFCLK_FORCE	RW	0x0	Force SRAM Local **FCLK** to run when set to HIGH.
3	SRAMSYSCLK_FORCE	RW	0x0	Force SRAM Local **SYSCLK** to run when set to HIGH.
2	SYSFCLK_FORCE	RW	0x0	Force Base element Local **FCLK** to run when set to HIGH.
1	SYSSYSCLK_FORCE	RW	0x0	Force Base element Local **SYSCLK** to run when set to HIGH.
0	MAINCLK_FORCE	RW	0x0	Force **MAINCLK** to run when set to HIGH.

RESET_SYNDROME

This register stores the reason for the last RESET event.

The register is cleared by **nPORESET** input or by software writing zeros to each bit to clear them.

Writing HIGH to a bit results in that bit maintaining its previous value.

Note

LOCKUP0 and LOCKUP1 do not generate reset, but when HIGH, indicate that a processor has locked-up and could be a precursor to another reset event, for example, a watchdog timer reset request.

Table 3-74 RESET_SYNDROME register

Bits	Name	Access	Reset value	Description
[31:10]	Reserved	RO	0x0	
9	SWRESETREQ	Write 0 to clear.	0x0	Software Reset Request.
8	RESETREQ	Write 0 to clear.	0x0	External Reset Request.
7	LOCKUP1	Write 0 to clear.	0x0	CPU 1 Lock-up Status.
6	LOCKUP0	Write 0 to clear.	0x0	CPU 0 Lock-up Status.
5	SYSRSTREQ1	Write 0 to clear.	0x0	CPU 1 System Reset Request.
4	SYSRSTREQ0	Write 0 to clear.	0x0	CPU 0 System Reset Request.
3	S32KWD	Write 0 to clear.	0x0	Watchdog on the **S32KCLK** clock.
2	SWD	Write 0 to clear.	0x0	Secure watchdog.
1	NSWD	Write 0 to clear.	0x0	Non-secure watchdog.
0	PoR	Write 0 to clear.	0x1	Power-on.

RESET_MASK

The RESET_MASK register allows software to control which reset sources are merged to generate the system-wide Warm reset, **nSYSRESETAON**, or the **nPORESETAON** signal.

Set each bit to HIGH to enable each source. The **nPORESET** input resets the register.

Note

Each of these mask bits, if cleared, not only prevents the reset source generating the reset, it also prevents the associated RESET_SYNDROME register bit from recording the event.

Table 3-75 RESET_MASK register

Bits	Name	Access	Reset value	Description
[31:6]	Reserved	RO	0x0	Reserved.
5	SYSRSTREQ1_EN	RW	SYSRSTREQ1_EN_RST	Enable Merging CPU 1 System Reset Request.
4	SYSRSTREQ0_EN	RW	SYSRSTREQ0_EN_RST	Enable Merging CPU 0 System Reset Request.
[3:2]	Reserved	RO	0x0	Reserved.
1	NSWD_EN	RW	0x0	Enable NON-SECURE WATCHDOG Reset.
0	Reserved	RO	0x0	Reserved.

SWRESET

The SWRESET register allows software to request for a System reset. This register is reset by **nPORESETAON** and **nWARMRESETAON**.

Table 3-76 SWRESET register

Bits	Name	Access	Reset value	Description
[31:10]	Reserved	RO	0x0	Reserved.
9	SWRESETREQ	WO	0x0	Software Reset Request. Set to HIGH to request a system reset that is equivalent to a watchdog or power-on reset.
[8:0]	Reserved	RO	0x0	Reserved.

GRETREG

The General Purpose Retention Register provides 16 bits of retention register for general storage, especially through power down of the rest of the system. **nPORESETAON** resets this register.

Table 3-77 GRETREG register

Bits	Name	Access	Reset value	Description
[31:16]	Reserved	RO		Reserved.
[15:0]	GRETREG	RW	0x0	General Purpose Retention Register.

INITSVTOR0, INITSVTOR1

This contains the value of the Secure Vector table offset address (VTOR_STBLOFF[31:7]), for CPU*n*.

The reset value is set by the static configuration signals **INITSVTORn_RST** from the top level.

Register INITSVTOR0 is connected to the CPU0 element, and INITSVTOR1 is connected to the CPU1 element. If CPU 1 does not exist, this register is RAZ/WI.

The following tables show how the same bit assignments apply for both the INITSVTOR0 and INITSVTOR1 registers.

Table 3-78 INITSVTOR0 register

Bits	Name	Access	Reset value	Description
[6:0]	Reserved	RO	-	Reserved.
[31:7]	INITSVTOR0	RW	INITSVTOR0_RST[31:7]	Default Secure Vector table offset at reset for CPU 0.

Table 3-79 INITSVTOR1 register

Bits	Name	Access	Reset value	Description
[6:0]	Reserved	RO	-	Reserved.
[31:7]	INITSVTOR1	RW	INITSVTOR1_RST[31:7]	Default Secure Vector table offset at reset for CPU 1.

CPUWAIT

This register provides controls to force each processor to wait after reset rather than Boot Immediately. This allows another entity in the expansion system or the debugger to access the system before the CPU booting. **nPORESETAON** resets this register.

Table 3-80 CPUWAIT register

Bits	Name	Access	Reset value	Description
[31:2]	Reserved	RO		Reserved.
1	CPU1WAIT	RW	CPU1WAIT_RST	CPU 1 waits at boot: • 0: boot normally • 1: wait at boot. If CPU 1 does not exist, this field is RAZ/WI. From Power-on reset, nSRST reset or Watchdog reset, this bit also controls if CPU 0 powers up. • 1: Do not power-up • 0: Power-up.
0	CPU0WAIT	RW	CPU0WAIT_RST	CPU 0 waits at boot: • 0: boot normally. • 1: wait at boot. From Power ON reset, nSRST reset or Watchdog Reset, this bit also controls if CPU 1 powers up. • 1: Do not power-up • 0: Power-up.

NMI_ENABLE

This register provides controls to enable, or disable, internally or externally generated Non-Maskable Interrupt sources from generating an NMI interrupt on each core. This allows a processor to take control of all internal NMI interrupt sources and mask external NMI interrupts, if necessary.

This register is reset by **nPORESETAON** only and its reset value is defined by configuration parameters.

Table 3-81 NMI_ENABLE register

Bits	Name	Access	Width	Reset value	Description
32:18	Reserved	RO	14		Reserved.
17	CPU1_EXPNMI_ENABLE	RW	1	CPU1_EXPNMI_ENABLE_RST	CPU1 Externally Sourced NMI Enable. This determines if the top-level pin, **CPU1EXPNMI**, can raise an NMI interrupt on CPU1: • HIGH, allowed. • LOW, masked and not allowed.
16	CPU0_EXPNMI_ENABLE	RW	1	CPU0_EXPNMI_ENABLE_RST	CPU0 Externally Sourced NMI Enable. This determines if the top-level pin, **CPU0EXPNMI**, can raise an NMI interrupt on CPU0: • HIGH, allowed. • LOW, masked and not allowed.
[15:2]	Reserved	RO	14		Reserved

Table 3-81 NMI_ENABLE register (continued)

Bits	Name	Access	Width	Reset value	Description
1	CPU1_INTNMI_ENABLE	RW	1	CPU1_INTNMI_ENABLE_RST	CPU1 Internally Sourced NMI Enable. This determines if the subsystem internally generated NMI interrupt sources can raise an NMI interrupt on CPU1: • HIGH, allowed. • LOW, masked and not allowed.
0	CPU0_INTNMI_ENABLE	RW	1	CPU0_INTNMI_ENABLE_RST	CPU0 Internally Sourced NMI Enable. This determines if the subsystem internally generated NMI interrupt sources can raise an NMI interrupt on CPU0: • HIGH, allowed. • LOW, masked and not allowed.

WICCTRL

The WIC Control register allows software to perform the WIC Enable handshake for each individual processor. This register is reset by **nPORESETAON** and **nWARMRESET**.

Table 3-82 WICCTRL register

Bits	Name	Access	Reset value	Description
[31:18]	Reserved	RO	0x0	Reserved
17	CPU1WICRDY	RO	0x0	CPU 1 WIC Enable Acknowledge. If CPU 1 does not exist, this field is RAZ/WI.
16	CPU0WICRDY	RO	0x0	CPU 0 WIC Enable Acknowledge.
[15:10]	Reserved	RO	0x0	Reserved
9	CPU1WICEN_CLR	WO	0x0	CPU 1 WIC Enable Request. Clear. Write 1 to clear CPU1WICEN_STATUS to LOW. If CPU 1 does not exist, this field is RAZ/WI.
8	CPU0WICEN_CLR	WO	0x0	CPU 0 WIC Enable Request Clear. Write 1 to clear CPU0WICEN_STATUS to LOW.
[7:6]	Reserved	RO	0x0	Reserved
5	CPU1WICEN_SET	WO	0x0	CPU 1 WIC Enable Request Set. Write 1 to set CPU1WICEN_STATUS to HIGH. If CPU 1 does not exist, this field is RAZ/WI.
4	CPU0WICEN_SET	WO	0x0	CPU 0 WIC Enable Request Set. Write 1 to set CPU0WICEN_STATUS to HIGH.
[3:2]	Reserved	RO	0x0	Reserved
1	CPU1WICEN_STATUS	RO	0x0	CPU 1 WIC Enable Request Status. Set to HIGH by writing 1 to CPU1WICEN_SET to request enabling of the CPU1 WIC. If CPU 1 does not exist, this field is RAZ/WI.
0	CPU0WICEN_STATUS	RO	0x0	CPU 0 WIC Enable Request Status. Set to HIGH by writing 1 to CPU0WICEN_SET to request enabling of the CPU0 WIC.

EWCTRL

The External Wakeup Control register allows software to perform handshake with the External Wakeup Controllers that is associated with each individual processor to support waking the associated core when it is fully powered down.

This register is reset by **nPORESETAON** and **nWARMRESET**.

Table 3-83 External Wakeup Control register

Bits	Name	Access	Reset value	Description
[31:10]	Reserved	RO	0x0	Reserved
9	EWC1EN_CLR	WO	0x0	External Wakeup Controller 1 Clear. Writing 1 to this bit clears EWC1EN_STATUS. This field always returns 0 when read. If CPU 1 does not exist, this field is RAZ/WI.
8	EWC0EN_CLR	WO	0x0	External Wakeup Controller 0 Clear. Writing 1 to this bit clears EWC1EN_STATUS. This field always returns 0 when read. If CPU 1 does not exist, this field is RAZ/WI.
[7:6]	Reserved	RO	0x0	Reserved
5	EWC1EN_SET	WO	0x0	External Wakeup Controller 1 Set. Writing 1 to this bit sets EWC1EN_STATUS. This field always returns 0 when read. If CPU 1 does not exist, this field is RAZ/WI.
4	EWC0EN_SET	WO	0x0	External Wakeup Controller 0 Set. Writing 1 to this bit sets EWC1EN_STATUS. This field always returns 0 when read. If CPU 1 does not exist, this field is RAZ/WI.
[3:2]	Reserved	RO	0x0	Reserved
1	EWC1EN_STATUS	RO	0x0	External Wakeup Controller 1 Enable. If HIGH, on entering Deepsleep with the WIC enabled, the External Wakeup Controller starts the process of holding interrupts that might arrive when the CPU 1 is powered down, and depending on the interrupt mask, attempt to wake CPU 1. If CPU 1 does not exist, this field is RAZ/WI.
0	EWC0EN_STATUS	RO	0x0	External Wakeup Controller 0 Enable. If HIGH, on entering Deepsleep with the WIC enabled, the External Wakeup Controller , starts the process of holding interrupts that might arrive when the CPU 0 is powered down, and depending on the interrupt mask, attempt to wake CPU 0.

PDCM_PD_SYS_SENSE

The Power Dependency Control Matrix System Power domain (PD_SYS) Sensitivity register is used to define whatkeeps the PD_SYS domain awake. This register is reset by **nPORESETAON** and **nWARMRESET**.

Table 3-84 Power Dependency Control Matrix System Power Domain Sensitivity register

Bits	Name	Access	Reset value	Description
[31:20]	Reserved	RO	0x0	Reserved.
19	S_PD_EXP3_IN	RW	0x0	Enable PDEXPIN[3] signal Sensitivity.
18	S_PD_EXP2_IN	RW	0x0	Enable PDEXPIN[2] signal Sensitivity.
17	S_PD_EXP1_IN	RW	0x0	Enable PDEXPIN[1] signal Sensitivity.
16	S_PD_EXP0_IN	RW	0x0	Enable PDEXPIN[0] signal Sensitivity.
[15:13]	Reserved	RO	0x0	Reserved.
12	S_PD_CRYPTO_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to keep ON if S_PD_CRYPTO_ON is ON.
[11:7]	Reserved	RO	0x0	Reserved.
6	S_PD_SRAM3_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to keep ON if SRAM3 power domain is ON.
5	S_PD_SRAM2_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to keep ON if SRAM2 power domain is ON.
4	S_PD_SRAM1_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to keep ON if SRAM1 power domain is ON.
3	S_PD_SRAM0_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to keep ON if SRAM0 power domain is ON.
2	S_PD_CPU1CORE_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to stay ON if PD_CPU1CORE is ON.
1	S_PD_CPU0CORE_ON	RO	0x1	Tied to HIGH. PD_SYS always tries to stay ON if PD_CPU0CORE is ON.
0	S_PD_SYS_ON	RW	0x1	Enable PD_SYS ON Sensitivity. Set HIGH to keep PD_SYS awake after powered ON.

PDCM_PD_SRAM<N>_SENSE

The Power Dependency Control Matrix SRAM<N> Power domain (PD_SRAM) Sensitivity registers are used to define what keeps each of the PD_SRAM<N> power domains awake, where <n> is 0:3.

This register is reset by **nPORESETAON** and **nWARMRESET**.

Table 3-85 PDCM_PD_SRAM0_SENSE registers

Bits	Name	Access	Reset value	Description
[31:20]	Reserved	RO	0x0	Reserved.
19	S_PD_EXP3_IN	RW	0x0	Enable PDEXPIN[3] signal Sensitivity.
18	S_PD_EXP2_IN	RW	0x0	Enable PDEXPIN[2] signal Sensitivity.
17	S_PD_EXP1_IN	RW	0x0	Enable PDEXPIN[1] signal Sensitivity.
16	S_PD_EXP0_IN	RW	0x0	Enable PDEXPIN[0] signal Sensitivity.
[15:13]	Reserved	RO	0x0	Reserved.
12	S_PD_CRYPTO_ON	RO	0x0	Tied to LOW. Ignores PD_CRYPTO.
[11:7]	Reserved	RO	0x0	Reserved.

Table 3-85 PDCM_PD_SRAM0_SENSE registers (continued)

Bits	Name	Access	Reset value	Description
6	S_PD_SRAM3_ON	RO	0x0	Tied LOW. Ignores PD_SRAM3 state.
5	S_PD_SRAM2_ON	RO	0x0	Tied LOW. Ignores PD_SRAM2 state.
4	S_PD_SRAM1_ON	RO	0x0	Tied LOW. Ignores PD_SRAM1 state.
3	S_PD_SRAM0_ON	RW	0x0	Enable sensitivity to PD_SRAM0.
2	S_PD_CPU1CORE_ON	RW	0x0	Enable sensitivity to PD_CPU1CORE.
1	S_PD_CPU0CORE_ON	RW	0x0	Enable sensitivity to PD_CPU0CORE.
0	S_PD_SYS_ON	RW	0x0	Enable sensitivity to PD_SYS.

Table 3-86 PDCM_PD_SRAM1_SENSE registers

Bits	Name	Access	Reset value	Description
[31:20]	Reserved	RO	0x0	Reserved.
19	S_PD_EXP3_IN	RW	0x0	Enable PDEXPIN[3] signal Sensitivity.
18	S_PD_EXP2_IN	RW	0x0	Enable PDEXPIN[2] signal Sensitivity.
17	S_PD_EXP1_IN	RW	0x0	Enable PDEXPIN[1] signal Sensitivity.
16	S_PD_EXP0_IN	RW	0x0	Enable PDEXPIN[0] signal Sensitivity.
[15:13]	Reserved	RO	0x0	Reserved.
12	S_PD_CRYPTO_ON	RO	0x0	Tied to LOW. Ignores PD_CRYPTO.
[11:7]	Reserved	RO	0x0	Reserved.
6	S_PD_SRAM3_ON	RO	0x0	Tied LOW. Ignores PD_SRAM3 state.
5	S_PD_SRAM2_ON	RO	0x0	Tied LOW. Ignores PD_SRAM2 state.
4	S_PD_SRAM1_ON	RW	0x0	Enable sensitivity to PD_SRAM1.
3	S_PD_SRAM0_ON	RO	0x0	Tied LOW. Ignores PD_SRAM0 state.
2	S_PD_CPU1CORE_ON	RW	0x0	Enable sensitivity to PD_CPU1CORE.
1	S_PD_CPU0CORE_ON	RW	0x0	Enable sensitivity to PD_CPU0CORE.
0	S_PD_SYS_ON	RW	0x0	Enable sensitivity to PD_SYS.

Table 3-87 PDCM_PD_SRAM2_SENSE registers

Bits	Name	Access	Reset value	Description
[31:20]	Reserved	RO	0x0	Reserved.
19	S_PD_EXP3_IN	RW	0x0	Enable PDEXPIN[3] signal Sensitivity.
18	S_PD_EXP2_IN	RW	0x0	Enable PDEXPIN[2] signal Sensitivity.
17	S_PD_EXP1_IN	RW	0x0	Enable PDEXPIN[1] signal Sensitivity.
16	S_PD_EXP0_IN	RW	0x0	Enable PDEXPIN[0] signal Sensitivity.
[15:13]	Reserved	RO	0x0	Reserved.
12	S_PD_CRYPTO_ON	RO	0x0	Tied to LOW. Ignores PD_CRYPTO.

Table 3-87 PDCM_PD_SRAM2_SENSE registers (continued)

Bits	Name	Access	Reset value	Description
[11:7]	Reserved	RO	0x0	Reserved.
6	S_PD_SRAM3_ON	RO	0x0	Tied LOW. Ignores PD_SRAM3 state.
5	S_PD_SRAM2_ON	RW	0x0	Enable sensitivity to PD_SRAM2.
4	S_PD_SRAM1_ON	RO	0x0	Tied LOW. Ignores PD_SRAM1 state.
3	S_PD_SRAM0_ON	RO	0x0	Tied LOW. Ignores PD_SRAM0 state.
2	S_PD_CPU1CORE_ON	RW	0x0	Enable sensitivity to PD_CPU1CORE.
1	S_PD_CPU0CORE_ON	RW	0x0	Enable sensitivity to PD_CPU0CORE.
0	S_PD_SYS_ON	RW	0x0	Enable sensitivity to PD_SYS.

Table 3-88 PDCM_PD_SRAM3_SENSE registers

Bits	Name	Access	Reset value	Description
[31:20]	Reserved	RO	0x0	Reserved.
19	S_PD_EXP3_IN	RW	0x0	Enable PDEXPIN[3] signal Sensitivity.
18	S_PD_EXP2_IN	RW	0x0	Enable PDEXPIN[2] signal Sensitivity.
17	S_PD_EXP1_IN	RW	0x0	Enable PDEXPIN[1] signal Sensitivity.
16	S_PD_EXP0_IN	RW	0x0	Enable PDEXPIN[0] signal Sensitivity.
[15:13]	Reserved	RO	0x0	Reserved.
12	S_PD_CRYPTO_ON	RO	0x0	Tied to LOW. Ignores PD_CRYPTO.
[11:7]	Reserved	RO	0x0	Reserved.
6	S_PD_SRAM3_ON	RW	0x0	Enable sensitivity to PD_SRAM3.
5	S_PD_SRAM2_ON	RO	0x0	Tied LOW. Ignores PD_SRAM2 state.
4	S_PD_SRAM1_ON	RO	0x0	Tied LOW. Ignores PD_SRAM1 state.
3	S_PD_SRAM0_ON	RO	0x0	Tied LOW. Ignores PD_SRAM0 state.
2	S_PD_CPU1CORE_ON	RW	0x0	Enable sensitivity to PD_CPU1CORE.
1	S_PD_CPU0CORE_ON	RW	0x0	Enable sensitivity to PD_CPU0CORE.
0	S_PD_SYS_ON	RW	0x0	Enable sensitivity to PD_SYS.

PDCM_PD_CRYPTO_SENSE

The Power Dependency Control Matrix CryptoCell Power domain (PD_CRYPTO) Sensitivity register defines what keeps thePD_CRYPTO domains awake.

Note

- The power domain cannot be configured to be sensitive to other power domain power states.
- The CryptoCell power domain cannot be configure to be sensitive to other power domain power state, and you can only perform static power control of this domain through the PD_CRYPTO Power Policy Unit.

Table 3-89 PDCM_PD_CRYPTO_SENSE register

Bits	Name	Access	Reset value	Description
[31:20]	Reserved	RO	0x0	Reserved.
19	S_PD_EXP3_IN	RO	0x0	Tied LOW. Ignores the **PDEXPIN[3]** signal.
18	S_PD_EXP2_IN	RO	0x0	Tied LOW. Ignores the **PDEXPIN[2]** signal.
17	S_PD_EXP1_IN	RO	0x0	Tied LOW. Ignores the **PDEXPIN[1]** signal.
16	S_PD_EXP0_IN	RO	0x0	Tied LOW. Ignores the **PDEXPIN[0]** signal.
[15:13]	Reserved	RO	0x0	Reserved.
12	S_PD_CRYPTO_ON	RO	0x1	Tied LOW. Ignores PD_CRYPTO.
[11:7]	Reserved	RO	0x0	Reserved.
6	S_PD_SRAM3_ON	RO	0x0	Tied LOW. Ignores PD_SRAM3 state.
5	S_PD_SRAM2_ON	RO	0x0	Tied LOW. Ignores PD_SRAM2 state.
4	S_PD_SRAM1_ON	RO	0x0	Tied LOW. Ignores PD_SRAM1 state.
3	S_PD_SRAM0_ON	RO	0x0	Tied LOW. Ignores PD_SRAM0 state.
2	S_PD_CPU1CORE_ON	RO	0x0	Tied LOW. Ignores PD_CPU1CORE state.
1	S_PD_CPU0CORE_ON	RO	0x0	Tied LOW. Ignores PD_CPU0CORE state.
0	S_PD_SYS_ON	RO	0x0	Tied LOW. Ignores PD_SYS.

3.6.5 Power Policy Unit registers

The following table lists the PPU registers.

Each of the PPU configurations is detailed in *2.7.5 Power Policy Units* on page 2-50.

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

Table 3-90 Summary of PPU registers

Offset	Name	Access	Short Name
0x000	Power Policy Register	RW	PPU_PWPR.
0x004	Power Mode Emulation Register	RW	PPU_PMER.
0x008	Power Status Register	RO	PPU_PWSR.
0x00C	Reserved	RO	-
0x010	Device Interface Input Current Status Register	RO	PPU_DISR.
0x014	Miscellaneous Input Current Status Register	RO	PPU_MISR.
0x018	Stored Status Register	RO	PPU_STSR.
0x01C	OFF Unlock register	RW	PPU_UNLK.
0x020	Power Configuration Register	RW	PPU_PWCR.
0x024	Power Mode Transition Configuration Register	RW	PPU_PTCR.
0x030	Interrupt Mask Register	RW	PPU_IMR.
0x034	Extra Interrupt Mask Register	RW	PPU_AIMR.

Table 3-90 Summary of PPU registers (continued)

Offset	Name	Access	Short Name
0x038	Interrupt Status Register	RW	PPU_ISR.
0x03C	Extra Interrupt Status Register	RW	PPU_AISR.
0x040	Input Edge Sensitivity Register	RW	PPU_IESR.
0x044	Operating Mode Active Edge Sensitivity Register	RW	PPU_OPSR.
0x050	Functional Retention RAM Configuration Register	RW	PPU_FUNRR.
0x054	Full Retention RAM Configuration Register	RW	PPU_FULRR.
0x058	Memory Retention RAM Configuration Register	RW	PPU_MEMRR.
0x160	Power Mode Entry Delay Register 0	RW	PPU_EDTR0.
0x164	Power Mode Entry Delay Register 1	RW	PPU_EDTR1.
0x170	Device Control Delay Configuration Register 0	RW	PPU_DCCR0.
0x174	Device Control Delay Configuration Register 1	RW	PPU_DCCR1.
0xFB0	PPU Identification Register 0	RO	PPU_IDR0.
0xFB4	PPU Identification Register 1	RO	PPU_IDR1.
0xFC8	Implementation Identification Register	RO	PPU_IIDR.
0xFCC	Architecture Identification Register	RO	PPU_AIDR.
0xFD0- FFF	IMPLEMENTATION DEFINED Identification Registers	RO	-

3.6.6 CMSDK Watchdog timer

The System Control element implements a CMSDK Watchdog timer running on the **S32KCLK** clock. This watchdog timer is mapped to a secure region and is able to raise NMI interrupt to both cores in the system.

The System Control element implements:

- • A single CMSDK Timer that reside in non-Secure region at `0x4002_F000` and in Secure region at `0x5002_F000`.
- A single CMSDK Watchdogs that resides in the Secure region at `0x5002_E000`.

See *3.4.3 CMSDK watchdog timers* on page 3-94 for a summary of the control registers.

Related references

3.4.3 CMSDK watchdog timers on page 3-94.

3.7 Debug and trace

This section describes the SSE-200 programmable options for debug and trace.

When accessing system memory through CPU0 AHB-AP and CPU1 AHB-AP, if a debug system memory access that originates from the APB-AP causes a security violation at a Peripheral Protection Controller (PPC) or a Memory Protection Controller (MPC) in the subsystem, it is blocked. This is like a normal non-debug failing access from the processor. However, the PPC and MPC do not raise an interrupt for these failing accesses.

For more information on debug and trace, see the *Arm® Cortex®-M33 Processor Technical Reference Manual*.

Note

If the system is configured for CoreSight SoC, a license is required for that IP and the following corresponding Arm documentation:

- *Arm® CoreSight™ SoC-400 System Design Guide*.
- *Arm® CoreSight™ SoC-400 Technical Reference Manual*.
- *Arm® CoreSight™ SoC-400 User Guide*.
- *Arm® CoreSight™ TPIU-Lite Technical Reference Manual*.
- *Arm® CoreSight™ DAP-Lite Technical Reference Manual*.

This section contains the following subsection:

- *3.7.1 Debug access interface* on page 3-142.

3.7.1 Debug access interface

The Debug access interface of the subsystem provides access to three *Access Ports* (APs) within the debug subsystem.

The following table lists the debug address map.

Table 3-91 Debug access region interface

Row ID	Address		Size	Region name	Description
	From	To			
1	`0x0000`	`0x00FF`	256B	SYSTEM APB-AP	Debug System Access APB-AP.
2	`0x0100`	`0x01FF`	256B	CPU0 AHB-AP	CPU0 Access AHB-AP.
3	`0x0200`	`0x02FF`	256B	CPU1 AHB-AP	CPU1 Access AHB-AP.
4	`0x0300`	`0xFFFF`	-	-	Reserved.

The Debug System APB-AP is used to access debug components that are in the debug subsystem, which includes components in the Debug element and components that are connected to the Debug APB Expansion Interface. This AP is only assessable when NIDEN is HIGH.

Note

A CoreSight ROM is also expected at address `0xf008_0000` in your debug expansion logic which catalogs all CoreSight expansion debug components that are deployed outside the SSE-200 subsystem that are accessible through the Debug APB Expansion Interface.

Table 3-92 System APB-AP address map

Row ID	Address		Size	Region Name	Description
	From	To			
1	0x0000_0000	0xEFFF_FFFF	-	-	Reserved.
2	0xF000_0000	0xF000_0FFF	4KB	SYSCSROM	Debug System CoreSight ROM.
3	0xF000_1000	0xF000_1FFF	4KB	SYSFUNNEL	Debug System Trace Funnel.
4	0xF000_2000	0xF000_2FFF	4KB	SYSCTI	Debug System Cross Trigger Interface.
5	0xF000_3000	0xF007_FFFF	500KB	-	Reserved.
6	0xF008_0000	0xF00F_FFFF	512KB	Debug APB Expansion Interface	Debug APB Expansion Interface Region.
4	0x_F010_0000	0xFFFF_FFFF	-	-	Reserved.

CPU0 AHB-AP is for CPU0 (Primary processor) debug access and also for certification access. It also maps a CoreSight ROM and a *Granular Power Requester* (GPR).

The accessibility of the access path for the certification is controlled by the CERTDISABLE, CERTDISABLED, CERTREADEN, and CERTREADENABLED control signals.

The following table lists the map for CPU0 AHB-AP, when CERTDISABLED is LOW.

Table 3-93 CPU0 AHB-AP Address Map when CERTDSIABLED is LOW

Row ID	Address		Size	Region name	Description
	From	To			
1	0x0000_0000	0x2FFF_FFFF		-	System memory access by the CPU0 Debug Access Port.
2	0x3000_0000	0x3000_1FFF	8KB	CERTMEM	Certificate Access Memory region, residing in SRAM0. Write access is allowed and read data is masked to zero if CERTREADENABLED is LOW. Access bypasses the core.
3	0x3000_2000	0xF000_7FFF		-	System memory access by the CPU0 Debug Access Port.
2	0xF000_8000	0xF000_8FFF	4KB	CPU0CSROM	CPU0 Access CoreSight ROM
3	0xF000_9000	0xF000_9FFF	4KB	CPU0GPR	CPU0 GPR
4	0xF000_A000	0xFFFF_FFFF		-	System memory access by the CPU0 Debug Access Port.

The following table lists the map for CPU0 AHB-AP, when CERTDISABLED is HIGH. table.

Table 3-94 CPU0 AHB-AP Address Map when CERTDSIABLED is HIGH

Row ID	Address		Size	Region name	Description
	From	To			
1	0x0000_0000	0xF000_7FFF		-	System memory access by the CPU0 Debug Access Port.
2	0xF000_8000	0xF000_8FFF	4KB	CPU0CSROM	CPU0 Access CoreSight ROM.
3	0xF000_9000	0xF000_9FFF	4KB	CPU0GPR	CPU0 GPR
4	0xF000_A000	0xFFFF_FFFF	-	-	System memory access by the CPU0 Debug Access Port.

CPU1 AHB-AP is for CPU1 (Secondary processor) debug access. It also maps a CoreSight ROM and a *Granular Power Requester* (GPR). The following table lists the memory map for CPU1 AHB-AP.

Table 3-95 CPU1 AHB-AP Address Map

Row ID	Address		Size	Region name	Description
	From	To			
1	0x0000_0000	0xF000_7FFF	-	-	System memory access by the CPU1 Debug Access Port.
2	0xF000_8000	0xF000_8FFF	4KB	CPU1CSROM	CPU1 Access CoreSight ROM
3	0xF000_9000	0xF000_9FFF	4KB	CPU1GPR	CPU1 GPR
4	0xF000_A000	0xFFFF_FFFF	-	-	System memory access by the CPU1 Debug Access Port.

Appendix A
Signal Descriptions

This appendix summarizes the interface signals present in the SSE-200 elements.

The SSE-200 defines a set of interfaces at the boundary of the subsystem. These interfaces are intended to remain largely unchanged with configuration, but provide a degree of configurability, allowing for customer scalability while meeting the system requirements.

It contains the following sections:

A.1 Clock, reset, and power control signals

This section describes the clock, reset, and power control signals.

This section contains the following subsections:

A.1.1 Functional clock and reset signals

The following table lists the clock and reset signals in the SSE-200 interfaces.

Table A-1 Clock and reset signals

Signal name	Width	Direction	Power Domain	Description
MAINCLK	1	Input	AON	Main Clock Input. This clock is used by the system to generate most other clocks that are used within the system.
MAINCLKREQ	1	Output	AON	Main Clock Request Signal. 1 indicates a request for the main clock to be active. 0 indicates that the clock can be turned off.
MAINCLKRDY	1	Input	AON	Main Clock Ready Signal. 1 indicates that the clock is running, stable, and the system can start to use it. 0 indicates that the clock has stopped or is unstable and must not be used.
S32KCLK	1	Input	AON	Slow Clock. Typically, a 32KHz clock input and is asynchronous to the other clocks in the system.
nPORESET	1	Input	AON	Active LOW Power-on Reset Input Signal.
nSRST	1	Input	AON	Active LOW System Reset Input from Debugger.
RESETREQ	1	Input	AON	Active HIGH Request to perform a system reset.
EXPWARMRESETREQ	1	Output	AON	Active HIGH Request to expansion logic to prepare for a Warm reset.
EXPWARMRESETACK	1	Input	AON	Active HIGH Acknowledge for expansion logic to indicate that it is ready for Warm reset.
nPORESETAON	1	Output	AON	Active LOW Power-on Reset for the Expansion System. This Power-on reset merges other reset sources within the system with **nPORESET** to generate this reset.
nWARMRESETAON	1	Output	AON	Active LOW System Reset Output
FCLK	1	Output	AON	Ungated Fast Clock. This is generated from **MAINCLK**.
SYSCLK	1	Output	AON	Ungated System Clock. This clock is generated from **FCLK** and is synchronous to **FCLK**.

Table A-1 Clock and reset signals (continued)

Signal name	Width	Direction	Power Domain	Description
HINTSYSCLKENCLK	1	Output	AON	The **HINTSYSCLKENCLK** hint function is a clock pulse that is generated to indicate when, relative to the divided clock **FCLK** an enable must be generated on **SYSCLK**.
EXPCLKREQ	1	Input	AON	Clock Request signal from expansion hardware to request for **FCLK** and **SYSCLK** to be active. 1 indicates request for clocks to be active. 0 indicates that clocks can be turned off.
EXPCLKRDY	1	Output	AON	Clock Ready signal to expansion hardware to indicate that **FCLK** and **SYSCLK** are active. 1 indicates that clocks are running, stable, and the system can start to use them. 0 indicates that clocks have stopped or are unstable and must not be used.
SYSFCLK	1	Output	PD_SYS	Base element Fast System Clock. This clock is to be used for Base element Expansion. This clock is synchronous to **FCLK** and is the PD_SYS power gated and Base element hierarchically clock gated version of **FCLK**.
SYSSYSCLK	1	Output	PD_SYS	Base element System Clock. This clock is to be used for Base element Expansion. This clock is synchronous to **SYSCLK** and is the PD_SYS power gated and Base element hierarchically clock gated version of **SYSCLK**.
SYSSYSUGCLK	1	Output	AON	Base element Ungated System Clock. This clock is to be used for Base element Expansion. This clock is synchronous to **SYSCLK** and is the PD_SYS power gated version of **SYSCLK**. This clock does not include Base element hierarchical clock gating. This signal is gated in the AON domain and is intended to be used only in the PD_SYS power domain.
SYSFUGCLK	1	Output	AON	Base element Ungated Fast Clock. This clock is for Base element expansion. It is synchronous to **FCLK** and is the PD_SYS power gated version of **FCLK**. This clock does not include Base element hierarchical clock gating. This signal is gated in the AON domain and intended to be used only in the PD_SYS power domain.
nWARMRESETSYS	1	Output	PD_SYS	Base element Active Low Warm Reset Output.
DEBUGFCLK	1	Output	AON	Debug Fast clock. This clock is to be used for Debug element Expansion only. This clock is a clock gated version of **FCLK** and is synchronous to **FCLK**. This signal is gated in the AON domain and intended to be used only in the PD_DEBUG power domain.

Table A-1 Clock and reset signals (continued)

Signal name	Width	Direction	Power Domain	Description
DEBUGSYSCLK	1	Output	AON	Debug System clock. This clock is to be used for Debug element Expansion only. This clock is a clock gated version of **SYSCLK** and is synchronous to **SYSCLK**. This signal is gated in the AON domain and intended to be used only in the PD_DEBUG power domain.
DEBUGHINTSYSCLKENCLK	1	Output	AON	**DEBUGHINTSYSCLKENCLK** is the equivalent but gated version of **HINTSYSCLKENCLK** for the debug power domain expansion. It is a clock pulse that is generated by the divider to indicate when, relative to the divided clock **DEBUGFCLK** the enable must be generated.
nPORESETDBG	1	Output	PD_DEBUG	Debug System Active LOW Power-on Reset Output. This signal is generated in the AON domain and intended to be used only in the PD_DEBUG power domain.
CRYPTOSYSCLK	1	Output	PD_CRYPTO	Crypto element System Clock. This clock is to be used for Crypto element Expansion. This clock is synchronous to **CRYPTOSYSCLK** and is the **PD_CRYPTO** power gated and Crypto element hierarchically clock gated version of **SYSCLK**. This output only exists if the Crypto element exists.
nWARMRESETCRYPTO	1	Output	PD_CRYPTO	Crypto element Active LOW Warm Reset Output. This output only exists if the Crypto element exists. The Crypto element itself can only be reset at boot or when resuming from power gating.
CPUDEBUGPIKCLK	1	Output	AON	CPU and Debug element Power Integration Clock. This is a hierarchically clock gated version of **SYSCLK**. It is intended for use by expansion power control logic that is expected to be reset using **nCPUDEBUGPIKRESET**, and controlled using the **CPUDEBUGPIKCLK** Q-Channel interface.
nCPUDEBUGPIKRESET	1	Output	AON	**CPUDEBUGPIKCLK** Reset. This reset is a **CPUDEBUGPIKCLK** resynchronized version of **nPORESETAON** reset.
BCRYPTOSPIKCLK	1	Output	AON	Base, Crypto, and System Power Integration Clock. This is a hierarchically clock gated version of **SYSCLK**. It is intended for use by expansion power control logic that is expected to be reset using **nBCRYPTOSPIKRESET**, and controlled using the **BCRYPTOSPIKCLK** Q-Channel interface.
nBCRYPTOSPIKRESET	1	Output	AON	**BCRYPTOSPIKCLK** Reset. This reset is a **BCRYPTOSPIKCLK** resynchronized version of **nWARMRESETAON** reset.

A.1.2 Clock control Q-Channel signals

The subsystem provides a Q-Channel interface for each of the output clocks to allow expansion logic to control the availability of each clock output. These are used to support hierarchical clock gating.

- Base element SYSCLK Q-Channel interface for **SYSSYSCLK** which includes:
  - **SYSSYSCLKQREQn** output,
  - **SYSSYSCLKQACCEPTn** input,
  - **SYSSYSCLKQDENY** input,
  - **SYSSYSCLKQACTIVE** input.
- Base element FCLK Q-Channel interface for **SYSFCLK** which includes:
  - **SYSFCLKQREQn** output,
  - **SYSFCLKQACCEPTn** input,
  - **SYSFCLKQDENY** input,
  - **SYSFCLKQACTIVE** input.
- Crypto element SYSCLK Q-Channel interface for **CRYPTOSYSCLK** which includes:
  - **CRYPTOSYSCLKQREQn** output,
  - **CRYPTOSYSCLKQACCEPTn** input,
  - **CRYPTOSYSCLKQDENY** input,
  - **CRYPTOSYSCLKQACTIVE** input.
- CPU and Debug element Power Integration Q-Channel interface for **CPUDEBUGPIKCLK** which includes:
  - **CPUDEBUGPIKCLKQREQn** output,
  - **CPUDEBUGPIKCLKQACCEPTn** input,
  - **CPUDEBUGPIKCLKQDENY** input,
  - **CPUDEBUGPIKCLKQACTIVE** input.
- Base, Crypto and System Power Integration Q-Channel interface for **BCRYPTOSPIKCLK** which includes:
  - **BCRYPTOSPIKCLKQREQn** output,
  - **BCRYPTOSPIKCLKQACCEPTn** input,
  - **BCRYPTOSPIKCLKQDENY** input,
  - **BCRYPTOSPIKCLKQACTIVE** input.

Each of these interfaces is asynchronous. If an interface is not used, the **QACTIVE** and **QDENY** signals must be tied LOW and the **QREQn** output loop back into its **QACCEPTn** input.

The subsystem does not provide a Q-Channel interface to control **DEBUGSYSCLK** and **DEBUGFCLK**. These clocks are controlled by the debug power domain, PD_DEBUG, and as long as the debug power domain is ON, **DEBUGSYSCLK** and **DEBUGFCLK** are active.

——— **Note** ———

The Q-Channel Interfaces do not support waking the system from hibernation. To wake from hibernation, you must use the EWC, power control Q-Channel interfaces, or the **PDEXPIN** inputs.

The Crypto element SYSCLK Q-Channel interfaces only exist if the Crypto element exists.

———————————

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

A.1.3 Power control Q-Channel signals

The subsystem provides power control Q-Channel interfaces to allow expansion logic to control the readiness of external expansion logic to power up and down.

- PD_SYS Power Q-Channel interface for the PD_SYS power domain which includes the following signals:
  - **SYSPWRQREQn** output,
  - **SYSPWRQACCEPTn** input,
  - **SYSPWRQDENY** input,
  - **SYSPWRQACTIVE** input.

  These signals must be driven by logic within the PD_SYS power domain and are used to determine if it is safe to change to a lower PD_SYS power state. The interface can also be driven from a power

domain more ON than the PD_SYS domain to request PD_SYS to power up. This interface is synchronous to **SYSSYSUGCLK** and **SYSSYSCLK**.

If you use the **SYSPWRQACTIVE** signal to request PD_SYS to power up, you must drive the **SYSPWRQACTIVE** signal HIGH at least until the PD_SYS power domain ON indication, **PDSYSON**, goes HIGH. This ensures that the PD_SYS is turned on correctly.

- PD_DEBUG Power Q-Channel interface for the PD_DEBUG power domain which includes the following signals:
  — **DEBUGPWRQREQn** output,
  — **DEBUGPWRQACCEPTn** input,
  — **DEBUGPWRQDENY** input,
  — **DEBUGPWRQACTIVE** input.

  These signals must be driven by expansion logic that resides within the PD_DEBUG power domain. They are used to determine if it is safe to change to a lower PD_DEBUG power state. It can also be driven from a different power domain to wake the PD_DEBUG domain. This is synchronous to **DEBUGSYSCLK**.

  If you use the **DEBUGPWRQACTIVE** signal to request PD_DEBUG to power up, you must drive the **DEBUGPWRQACTIVE** signal HIGH at least until the PD_DEBUG power domain ON indication, **PDDEBUGON**, goes HIGH. This ensures that the PD_DEBUG is turned on correctly.
- PD_CRYPTO Power Q-Channel interface for the PD_CRYPTO power domain which includes:
  — **CRYPTOPWRQREQn** output,
  — **CRYPTOPWRQACCEPTn** input,
  — **CRYPTOPWRQDENY** input,
  — **CRYPTOPWRQACTIVE** input.

  This Q-Channel interface only exists if the Crypto element exists.

  These signals must be driven by expansion logic that within the PD_CRYPTO power domain which includes logic in the CryptoCell Non-Volatile Memory Interface. The signals determine if it is safe to change to a lower PD_CRYPTO power state. PD_CRYPTO only supports static power control, so this interface cannot be used to request the PD_CRYPTO power domain to power up. This interface is synchronous to **CRYPTOSYSCLK**.

If an interface is not used, then **QACTIVE** and **QDENY** signal must be tied LOW and the **QREQn** output loop back into its **QACCEPTn** input.

——— **Note** ———

When using the PD_SYS Power Q-Channel interface to wake the system from hibernation, you must consider context restoration.

———————————

A.1.4 Expansion power control dependency signals

The subsystem provides a set of four signals 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 are the Power Dependency Inputs, **PDEXPIN[3:0]**.

These are asynchronous signals and reside in the PD_AON power domain.

A.1.5 Power domain ON status signals

The SSE-200 provides a set of output signals that indicate if a power domain is in the ON state:

- **PDSYSON** when HIGH indicates that the PD_SYS power domain is ON. When LOW it indicates that the power domain is at a lower power state.
- **PDCPU0COREON** when HIGH indicates that the PD_ CPU0CORE power domain is ON. When LOW it indicates that the power domain is at a lower power state.

- **PDCPU1COREON** when HIGH indicates that the PD_ CPU1CORE power domain is ON. When LOW it indicates that the power domain is at a lower power state.
- **PDDEBUGON** when HIGH indicates that the PD_ DEBUG power domain is ON. When LOW it indicates that the power domain is at a lower power state.

These status signals can be used by external expansion logic for separate power domains to determine if the power domain should remain powered, or to go to a lower power state.

A.2 Interrupt signals

The following table lists interrupt signals in the SSE-200 interface. These connect to the interrupt controller of each processor within the SSE-200 through an External Wakeup Controller (EWC) associated with the processor, and the Wakeup Interrupt Controller (WIC) of the Cortex-M33.

Table A-2 Interrupt signals

Signal name	Width	Direction	Description
CPU0EXPIRQ[CPU0_EXP_NUMIRQ-1:0]	CPU0_EXP_NUMIRQ[a]	Input	These are Interrupt inputs from the expansion subsystem to the CPU 0 interrupt controller within the SSE-200. The processor in the SSE-200 implements a configurable number of external interrupt lines and 32 of these are reserved for internal use and the remaining are made available here. **Note** Each bit **CPU0EXPIRQ[n]** is ultimately connected to IRQ[32+*n*] of the NVIC for CPU 0.
CPU1EXPIRQ[CPU1_EXP_NUMIRQ-1:0]	CPU1_EXP_NUMIRQ[b]	Input	These are Interrupt inputs from the expansion subsystem to the CPU 1 interrupt controller within the SSE-200. The processor in the SSE-200 implements a configurable number of external interrupt lines and 32 of these are reserved for internal use and the remaining are made available here. **Note** Each bit **CPU1EXPIRQ[n]** is ultimately connected to IRQ[32+*n*] of the NVIC for CPU 1. If the system is configured to not include CPU 1, then this interface does not exist. .

[a] CPU0_EXP_NUMIRQ defines the number of interrupts made available as expansion interrupts for CPU 0.
[b] CPU1_EXP_NUMIRQ defines the number of interrupts made available as expansion interrupts for CPU 1.

Table A-2 Interrupt signals (continued)

Signal name	Width	Direction	Description
CPU0EXPNMI	1	Input	This provides a non-maskable interrupt input from the expansion subsystem to the interrupt controller of CPU 0 within the SSE-200. This input is merged with other non-maskable interrupt sources within the SSE-200 before been seen by the NVIC of the core.
CPU1EXPNMI	1	Input	This provides a non-maskable interrupt input from the expansion subsystem to the interrupt controller of CPU 1 within the SSE-200. This input is merged with other non-maskable interrupt sources within the SSE-200 before been seen by the NVIC of the core. If the system is configured to not include CPU 1, then this interface does not exist.

Related references

A.5.4 Master security expansion interface on page Appx-A-163.

A.3 AHB expansion bus signals

The following table lists the signals for the two AHB master interfaces.

Table A-3 External AHB target port signals

Signal name	Width	Direction	Description
HSEL	1	Output	Slave Select.
HADDR	32	Output	Address bus.
HBURST	3	Output	Burst type.
HMASTLOCK	1	Output	Locked Sequence.
HPROT	7	Output	Protection Control.
HSIZE	3	Output	Transfer Size.
HNONSEC	1	Output	Indicates that the current transfer is either a Non-secure transfer or a Secure transfer.
HEXCL	1	Output	Exclusive Transfer. Indicates that the transfer is part of an Exclusive access sequence.
HMASTER	4	Output	Master Select.
HTRANS	2	Output	Transfer Type.
HWDATA	32	Output	Write Data.
HWRITE	1	Output	Transfer Direction.
HAUSER	2	Output	Address USER signals (Not used by the SSE-200 processors).
HWUSER	2	Output	Write-data USER signals (Not used by the SSE-200 processors).
HRUSER	2	Input	Read-data USER signals (Not used by the SSE-200 processors).
HRDATA	32	Input	Read data bus.
HREADYOUT	1	Input	When HIGH, the **HREADY** signal indicates to the master and all slaves, that the previous transfer is complete.
HRESP	1	Input	Transfer response.
HEXOKAY	1	Input	Exclusive OK.

The following table lists the signals for the two AHB slave interfaces.

Table A-4 External AHB initiator port signals

Signal name	Width	Direction	Description
HSEL	1	Input	Slave Select.
HADDR	32	Input	Address bus.
HBURST	3	Input	Burst type.
HMASTLOCK	1	Input	Locked Sequence.
HPROT	7	Input	Protection Control.
HSIZE	3	Input	Transfer Size.
HNONSEC	1	Input	Indicates that the current transfer is either a Non-secure transfer or a Secure transfer.
HEXCL	1	Input	Exclusive Transfer. Indicates that the transfer is part of an Exclusive access sequence.

Table A-4 External AHB initiator port signals (continued)

Signal name	Width	Direction	Description
HMASTER	4	Input	Master Select.
HTRANS	2	Input	Transfer Type.
HWDATA	32	Input	Write Data.
HWRITE	1	Input	Transfer Direction.
HAUSER	2	Input	Address USER signals (Not used by the SSE-200 processors).
EXREQ	1	Input	Exclusive Request signal.
HWUSER	2	Input	Write-data USER signals (Not used by the SSE-200 processors).
HRUSER	2	Output	Read-data USER signals (Not used by the SSE-200 processors).
HRDATA	32	Output	Read data bus.
HREADY	1	Output	HREADY feedback.
HRESP	1	Output	Transfer response.
HREADYOUT	1	Output	When HIGH, the **HREADY** signal indicates to the master and all slaves, that the previous transfer is complete.
HEXOKAY	1	Output	Exclusive OK.

The following table lists the signals on the code expansion bus:

Table A-5 External code bus signals

Signal name	Width	Direction	Description
CODEEXPHSEL	1	Output	Slave select.
CODEEXPHADDR	32	Output	Address bus.
CODEEXPHBURST	3	Output	Burst type.
CODEEXPHMASTLOCK	1	Output	Locked sequence.
CODEEXPHPROT	7	Output	Protection control.
CODEEXPHSIZE	3	Output	Transfer size.
CODEEXPHNONSEC	1	Output	Indicates that the current transfer is either a Non-secure transfer or a Secure transfer.
CODEEXPHEXCL	1	Output	Exclusive transfer. Indicates that the transfer is part of an Exclusive access sequence.
CODEEXPHMASTER	4	Output	Master select.
CODEEXPHTRANS	2	Output	Transfer type.
CODEEXPHWDATA	32	Output	Write data.
CODEEXPHWRITE	1	Output	Transfer direction.
CODEEXPHAUSER	2	Output	Address USER signals.
CODEEXPHWUSER	2	Output	Write channel USER signals.
CODEEXPHRUSER	2	Input	Read channel USER signals.
CODEEXPHRDATA	32	Input	Read data bus.

Table A-5 External code bus signals (continued)

Signal name	Width	Direction	Description
CODEEXPHREADYOUT	1	Input	When HIGH, the **HREADY** signal indicates to the master and all slaves, that the previous transfer is complete.
CODEEXPHRESP	1	Input	Transfer response.
CODEEXPHEXOKAY	1	Input	Exclusive OK.

A.4 Debug and Trace signals

This section lists signals related to debug and trace.

This section contains the following subsections:

A.4.1 DAP signals

The following table lists the DAP Access Bus Interface signals which allow an external DAP to access debug logic within the system and to also request for the debug system to wake.

Note

All signals on this interface, unless stated otherwise, are:

- Synchronous to **DEBUGSYSCLK**.
- Reset by **nPORESETDEBUG**.
- In the PD_DEBUG power domain.

Table A-6 DAP Access Buffer signals

Signal name	Width	Direction	Clock domain	Description
DAPCADDRS	14	Input	DEBUGSYSCLK	DAP compressed address bus.
DAPSELS	1	Input	DEBUGSYSCLK	Select signal generated from the DAP decoder to each AP. This signal indicates that the slave device is selected, and a data transfer is required.
DAPENABLES	1	Input	DEBUGSYSCLK	DAP enable.
DAPWRITES	1	Input	DEBUGSYSCLK	When HIGH indicates a DAP write access from DP to AHB-AP. When LOW, indicates a read access.
DAPWDATAS	32	Input	DEBUGSYSCLK	DAP write data bus.
DAPABORTS	1	Input	DEBUGSYSCLK	DAP abort.
DAPRDATAS	32	Output	DEBUGSYSCLK	DAP read data bus.
DAPREADYS	1	Output	DEBUGSYSCLK	DAP ready.
DAPSLVERRS	1	Output	DEBUGSYSCLK	DAP error.

A.4.2 Timestamp interface

The following table lists the timestamp interface signals. This timestamp is expected to be driven by a timestamp generator in the expansion subsystem. This input is synchronous to **DEBUGSYSCLK** and resides in the PD_DEBUG power domain.

Table A-7 Timestamp signals

Signal name	Width	Direction	Clock domain	Description
TSVALUEB	64	Input	DEBUGSYSCLK	Timestamp input value in binary.

A.4.3 Cross Trigger interfaces

The following table lists the cross trigger channel interface signals. This interface allows partners to expand the cross trigger infrastructure. This interface is synchronous to **DEBUGSYSCLK** and resides in the PD_DEBUG power domain.

Table A-8 Cross trigger channel signals

Signal name	Width	Direction	Clock domain	Description
CTMCHOUT	4	Output	DEBUGSYSCLK	Channel out port
CTMCHOUTACK[a]	4	Input	DEBUGSYSCLK[b]	Channel out acknowledge port 0
CTMCHIN	4	Input	DEBUGSYSCLK[b]	Channel in port
CTMCHINACK	4	Output	DEBUGSYSCLK	Channel in acknowledge port 0

Related references

A.6 Miscellaneous top-level signals on page Appx-A-165.

A.4.4 Cross trigger signals

Most of the trigger signals are used within the system. The following table lists the remaining signals that are available for system expansion.

Table A-9 Cross trigger signals

Signal name	Width	Direction	Clock domain	Description
CTITRIGIN	8	Input	DEBUGSYSCLK[c]	Trigger inputs ports.
CTITRIGINACK	8	Output	DEBUGSYSCLK	Trigger inputs acknowledge.
CTITRIGOUT	4	Output	DEBUGSYSCLK	Trigger outputs port.
CTITRIGOUTACK[d]	4	Input	DEBUGSYSCLK[e]	Trigger outputs acknowledge.

This interface is synchronous to **DEBUGSYSCLK** and resides in the PD_DEBUG power domain.

The CTI also outputs the configuration signals **TINIDENSEL**, **TIHSBYPASS**, **TISBYPASSACK**, **TISBYPASSIN**, and **TODEBUGENSEL**. See *A.6 Miscellaneous top-level signals* on page Appx-A-165.

For more details on the CTI, see *Arm® CoreSight™ SoC-400 Technical Reference Manual*.

A.4.5 Debug APB expansion interface

The following table lists the debug APB expansion interface signals which allow partners to add more debug functionality to the SSE-200.

This interface:

- Is only accessible from the debug interface using an external DAP.
- Is synchronous to **DEBUGSYSCLK**.
- Is reset using **nPORESETAON**.
- Resides in the PD_DEBUG power domain.

[a] If not bypassed (Parameter `CTMCHCIHSBYPASS` = LOW). If bypassed, the signal is not needed and is tied HIGH.
[b] If Parameter `CTMCHCISBYPASS` = LOW. Else this signal is asynchronous.
[c] If parameter `TISBYPASSIN` = HIGH. Else this signal is asynchronous.
[d] If not bypassed (Parameter `TIHSBYPASS` = LOW). If bypassed, the signal is not required and must be tied HIGH.
[e] If parameter `TISBYPASSACK` = HIGH. Else this signal is asynchronous.

Table A-10 Debug APB expansion signals

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

A.4.6 ATB Trace interface

The following table lists the ATB Trace interface signals that are intended to connect to an external TPIU.

This trace interface is synchronous to **DEBUGFCLK**, is reset using **nPORESETDEBUG**, and resides in the PD_DEBUG power domain.

Table A-11 Trace signals

Signal name	Width	Direction	Clock domain	Description
ATVALID	1	Output	DEBUGFCLK	A transfer is valid during this cycle. If LOW, all the other ATB signals must be ignored in this cycle.
ATID	7	Output	DEBUGFCLK	An ID that uniquely identifies the source of the trace.
ATBYTE	1	Output	DEBUGFCLK	The number of bytes on ATDATA to be captured, minus 1.
ATDATA	16	Output	DEBUGFCLK	Trace data bus.
ATREADY	1	Input	DEBUGFCLK	Slave is ready to accept data.
AFVALID	1	Input	DEBUGFCLK	This is the flush signal. All buffers must be flushed because trace capture is about to stop.
AFREADY	1	Output	DEBUGFCLK	This is a flush acknowledge. Asserted when buffers are flushed.
SYNCREQ	1	Input	DEBUGFCLK	Trace synchronization request trace sinks

A.4.7 Debug authentication interface

The interface input signals define the Debug Authentication signal values when the signals are not overridden by the internal Secure Debug Configuration registers. The final debug authentication signals are then output to the rest of the system.

Table A-12 Debug authentication

Signal name	Width	Direction	Description
DEBUGENIN	1	Input	Debug Enable Input
NIDENIN	1	Input	Non-Invasive Debug Enable Input
SPIDENIN	1	Input	Secure Privilege Invasive Debug Enable Input
SPNIDENIN	1	Input	Secure Privilege Non-Invasive Debug Enable Input
DEBUGEN	1	Output	Merged Debug Enable Output
NIDEN	1	Output	Merged Non-Invasive Debug Enable Output
SPIDEN	1	Output	Merged Secure Privilege Invasive Debug Enable Output
SPNIDEN	1	Output	Merged Secure Privilege Non-Invasive Debug Enable Output

Note

SSE-200 does not contain a Debug Access Port (DAP) that can use some of these signals to control access. Instead they are used by processors in the system and also Access Ports (AP) in the Debug element to control access into the debug subsystem and into the main system.

A.5 Security component interfaces

This section lists components that are related to security.

Note

- These interface signals allow all the components to be controlled using the same set of security control registers already implemented within the subsystem.
- While the SSE-200 defines a full set of signals as described in this document, the configuration parameters disable many of these interfaces and even individual bits. Disabling each bit of the interface or the entire interface does not remove the interface, but does remove the logic behind it and ties any unused outputs to zero.
- All signals in this section are synchronous to **SYSSYSCLK**, and the Base element **SYSCLK** Q-Channel interface needs to control the availability of **SYSSYSCLK**. These signals reside in the PD_SYS power domain.

This section contains the following subsections:

A.5.1 Memory protection controller interface

SSE-200 supports up to 16 MPCs to be added to the expansion system. The signals that are listed in the following table allow the interrupts of the MPCs to be merged to the single MPC Combined interrupt internally.

Table A-13 MPC expansion

Signal name	Width	Direction	Description
SMPCEXPSTATUS	16	Input	Interrupt Status inputs from all Expansion Memory Protection Controller. These are visible to the programmer by the S_MPCEXP_STATUS register in the Secure Privilege Control Register Block and are used to raise an interrupt using the MPC Combined Interrupt. The top-level parameter `MPCEXP_DIS` allows each individual bit of the interface to be disabled.

A.5.2 APB peripheral protection controller interface

The following table lists the APB PPC interface signals. Up to four extra PPCs can be added to the system.

Table A-14 APB peripheral protection controller

Signal name	Width	Direction	Description
SAPBPPCEXPSTATUS	4	Input	APB PPC Interrupt Status Input. Each bit *N* is to be connected to a single APB PPC <*N*>, where *N* is 0-3. These are associated to the S_APBPPCEXP_STATUS field in the SECPPCINTSTAT register.
SAPBPPCEXPCLEAR	4	Output	APB PPC Interrupt Clear Output. Each bit *N* is to be connected to a single APB PPC<*N*>, where *N* is 0-3. These are associated to the S_APBPPCEXP_CLR field in the SECPPCINTCLR register.
APBNSPPCEXP0	16	Output	APB PPC Non-secure Gating Control. These are a set of four 16-bit interfaces, and each interface connects to a PPC. When each bit *m* of an interface is HIGH, it defines the APB<*m*> interface that the target PPC controls as Non-secure access only. Each 16-bit signal **APBNSPPCEXP<N>** is driven by the APBNSPPCEXP<*N*> register, where *N* is 0-3. The top level parameters `APBPPCEXP_DIS<N>` allows individual bits of each 16-bit bus to be disabled.
APBNSPPCEXP1	16	Output	
APBNSPPCEXP2	16	Output	
APBNSPPCEXP3	16	Output	
APBPPPCEXP0	16	Output	APB PPC Privilege Gating Control. These are a set of four 16-bit interfaces. When each bit *m* of an interface is HIGH, it defines the APB<*m*> interface that the target PPC controls as privilege access only. Each bit of each 16-bit signal is selected from either APBSPPPCEXP<N>[*m*] if **APBNSPPCEXP<N>[m]** is 0 or APBNSPPPCEXP<*N*>[*m*] otherwise, where *N* is 0-3. The top-level parameters `APBPPCEXP_DIS<N>` allows individual bits of each 16-bit bus to be disabled.
APBPPPCEXP1	16	Output	
APBPPPCEXP2	16	Output	
APBPPPCEXP3	16	Output	

A.5.3 AHB peripheral protection controller interface

The following table lists the AHB PPC interface signals. Up to four extra PPCs can be added to the system.

Table A-15 PPC signals

Signal name	Width	Direction	Description
SAHBPPCEXPSTATUS	4	Input	AHB PPC Interrupt Status Input. Each bit *N* is to be connected to a single AHB PPC <*N*> where *N* is 0-3. These are associated to the S_AHBPPCEXP_STATUS field in the SECPPCINTSTAT register.
SAHBPPCEXPCLEAR	4	Output	AHB PPC Interrupt Clear Output. Each bit *N* is to be connected to a single AHB PPC<*N*> where *N* is 0-3. These are associated to the S_AHBPPCEXP_CLR field in the SECPPCINTCLR register.

Table A-15 PPC signals (continued)

Signal name	Width	Direction	Description
AHBNSPPCEXP0	16	Output	AHB PPC Non-secure Gating Control. These are a set of four 16-bit interfaces, and each interface connects to a PPC. When each bit *m* of an interface is HIGH, it defines the AHB<*m*> interface that the target PPC controls as Non-secure access only. Each 16-bit signal **AHBNSPPCEXP<N>** is driven by the AHBNSPPCEXP<*N*> register, where *N* is 0-3. The top-level parameters `AHBPPCEXP_DIS<N>` allows individual bits of each 16-bit bus to be disabled.
AHBNSPPCEXP1	16	Output	
AHBNSPPCEXP2	16	Output	
AHBNSPPCEXP3	16	Output	
AHBPPPCEXP0	16	Output	Four 16-bit AHB PPC Privilege Gating Control interfaces. When each bit *m* of an interface is HIGH, it defines the AHB<*m*> interface that the target PPC controls as privilege access only. Each bit of an interface is selected from either: • AHBSPPPCEXP<*N*>[*m*] if AHBNSPPCEXP<*N*>[*m*] = 0 • AHBNSPPPCEXP<*N*>[*m*] otherwise, where *N* is 0-3. The top-level parameters `AHBPPCEXP_DIS<N>` allow individual bits of each 16-bit bus to be disabled.
AHBPPPCEXP1	16	Output	
AHBPPPCEXP2	16	Output	
AHBPPPCEXP3	16	Output	

A.5.4 Master security expansion interface

The following table lists the master security expansion signals.

Table A-16 Master security expansion

Signal name	Width	Direction	Description
SMSCEXPSTATUS	16	Input	MSC Interrupt Status Input. Each bit *N* is to be connected to a single MSC <*N*> where *N* is 0-15. These are associated with the S_MSCEXP_STATUS field in the SECMSCINTSTAT register. The top-level parameter `MSCEXP_DIS` allows each individual bit of the interface to be disabled.
SMSCEXPCLEAR	16	Output	MSC Interrupt Clear Output. Each bit *N* is to be connected to a single MSC <*N*> where *N* is 0-15. These are associated with the S_MSCEXP_CLR field in the SECMSCINTCLR register. The top-level parameter `MSCEXP_DIS` allows each individual bit of the interface to be disabled.
NSMSCEXP	16	Output	MSC Non-secure Configuration. Each bit *N* is to be connected to a single MSC <*N*> where *N* is 0-15. Set to HIGH to configure a master as Non-secure. These are associated with the NS_MSCEXP field in the NSMSCECP register The top-level parameter `MSCEXP_DIS` allows each individual bit of the interface to be disabled.

A.5.5 Bridge buffer error interface

The following table lists the bridge buffer interface signals. Up to 16 extra bridges with buffer error signaling to be added to the expansion system.

Table A-17 Bridge buffer signals

Signal name	Width	Direction	Description
BRGEXPSTATUS	16	Input	Bridge Error Interrupt Status Input. Each bit *N* is to be connected to a single bridge <*N*> where *N* is 0-15. These are associated with the BRGEXP_STATUS field in the BRGINSTAT Register. The top-level parameter `BRGEXP_DIS` allows each individual bit of the interface to be disabled.
BRGEXPCLEAR	16	Output	Bridge Error Interrupt Clear Output. Each bit *N* is to be connected to a single bridge <*N*> where *N* is 0-15. These are associated with the BRGEXP_CLR field in the BRGINSTAT Register. The top-level parameter `BRGEXP_DIS` allows each individual bit of the interface to be disabled.

A.5.6 Miscellaneous security expansion signals

The following table lists the signals required by the PPCs and MSCs in the system.

Table A-18 Security expansion signals

Signal name	Width	Direction	Description
SECRESPCFG	1	Output	This signal configures how to respond to an access when a security violation occurs. 0 - Read-Zero Write Ignore 1 - bus error This signal is controlled by the SECRESPCFG register.
ACCWAITNSTATUS	1	Output	This signal controls any external gating unit that might be required to block accesses to the system from the AHB Slave Expansion interfaces. 1 – No Gating 0 – Access Gated. This signal is controlled by the BUSWAIT register.
ACCWAITN	1	Input	This status signal indicates the state of any external gating unit that can be used to block accesses to the system from the AHB Slave Expansion interfaces. 1 – No Gating 0 – Access Gated. This signal can be read from the BUSWAIT register.
CERTDISABLEEXT	1	Input	The Certificate Path Disable External input signal allows an external entity to drive this signal HIGH to directly set the CERTDISABLE register bit in the SCSECCTRL register. Tie to HIGH to disable certificate access by default. Tie to LOW if this signal is not used.

A.6 Miscellaneous top-level signals

This section lists the top-level signals.

This section contains the following subsections:

A.6.1 Top-level signals

The following table lists signals at the top-level of the SSE-200.

Table A-19 SSE-200 top-level signals

Signal name	Width	Direction	Description
LOCKNSVTOR0	1	Input	When HIGH, disables writes to the CPU 0 Non-secure vector table base address register, VTOR_NS. If not used, tie to LOW.
LOCKNSVTOR1	1	Input	When HIGH, disables writes to the CPU 1 Non-secure vector table base address register, VTOR_NS. This signal does not exist if CPU 1 does not exist in the system. If not used, tie to LOW.
LOCKNSMPU0	1	Input	When HIGH, disables writes to the MPU_CTRL_NS, MPU_RNR_NS, MPU_RBAR_NS, MPU_RLAR_NS, MPU_RBAR_A_NSn and MPU_RLAR_A_NSn in CPU 0 When HIGH all writes to the registers are ignored. This signal has no effect if the CPU 0 has been configured without any Non-secure MPU regions. If not used, tie to LOW.
LOCKNSMPU1	1	Input	When HIGH, disables writes to the MPU_CTRL_NS, MPU_RNR_NS, MPU_RBAR_NS, MPU_RLAR_NS, MPU_RBAR_A_NSn and MPU_RLAR_A_NSn in CPU 1 When HIGH, all writes to the registers are ignored. This signal has no effect if the CPU 1 does not exist or CPU 1 has been configured without any Non-secure MPU regions. If not used, tie to LOW.
CPU0TRCENA	1	Output	Trace Enable. This signal reflects the setting of the DEMCR.TRCENA bit for CPU0. It can be used as control signal to enable or disable trace-related functionality outside the subsystem.
CPU1TRCENA	1	Output	Trace Enable. This signal reflects the setting of the DEMCR.TRCENA bit for CPU1. It can be used as control signal to enable or disable trace-related functionality outside the subsystem.
SYSSYNCHCLAMPREADY	1	Input	Synchronous clamping ready status. When ‘1’, indicates that all clamps, requested by **SYSSYNCHCLAMP*** signals are applied.

Table A-19 SSE-200 top-level signals (continued)

Signal name	Width	Direction	Description
SYSSYNCHCLAMPRETOFFHIGH	1	Output	Synchronous clamping enable for outputs to clamp high in OFF or RETENTION states. These are set to HIGH before the PD_SYS domain is isolated and enters OFF/Retention state.
SYSSYNCHCLAMPRETOFFLOW	1	Output	Synchronous clamping enable for outputs to clamp low in OFF or RETENTION states. These are set to HIGH before the PD_SYS domain is isolated and enters OFF/Retention state.
SYSSYNCHCLAMPOFFHIGH	1	Output	Synchronous clamping enable for outputs to clamp high only in OFF state. These are set to HIGH before the PD_SYS domain is isolated and enters OFF state.
SYSSYNCHCLAMPOFFLOW	1	Output	Synchronous clamping enable for outputs to clamp low only in OFF state. These are set to HIGH before the PD_SYS domain is isolated and enters OFF state.

A.6.2 Top-level static configuration signals

The following table lists the static configuration signals that are at the top level of the SSE-200. These static signals are asynchronous and are expected to be unchanged after de-assertion of Power-on reset.

Table A-20 Top-level static configuration signals

Signal name	Width	Direction	Description
SYSCLKENTRYDELAY	8	Input	The **SYSCLKENTRYDELAY** defines the number of delay cycles that a **SYSCLK**-related hierarchical clock gated domain is idle before clocks are sequenced to turn off.
FCLKENTRYDELAY	8	Input	The **FCLKENTRYDELAY** defines the number of delay cycles that an **FCLK**-related hierarchical clock gated domain is idle before clocks are sequenced to turn off.
INITSVTOR0_RST	25	Input	Reset Value of the CPU 0 (Primary) Secure Vector table offset present in the address register in the System Control Register
INITSVTOR1_RST	25	Input	Reset Value of the CPU 1 (Secondary) Secure Vector table offset present in the address register in the System Control Register.
CTMCHCISBYPASS	1	Input	Defines if the **CTMCHIN** and **CTMCHOUTACK[3:0]** of the Cross Trigger Channel Interface is synchronous or asynchronous to the **DEBUGSYSCLK**: 0: Asynchronous. Signals are resynchronized internally. 1: Synchronous. Signals are not resynchronized internally.
CTMCHCIHSBYPASS	4	Input	Defines whether the handshake logic that is associated to each **CTMCHOUT** pin is used. Tie HIGH to disable the handshake logic. Handshake logic is not required if **CTMCHOUT** drives synchronous logic.
TINIDENSEL	8	Input	**NIDEN** mask on **CTITRIGIN**. When each bit is set to LOW, it masks the associated Cross Trigger Interface's trigger input when **NIDEN** is LOW.
TIHSBYPASS	4	Input	Cross Trigger interface handshake bypass on **CTITRIGOUT**. Tie each bit to HIGH to disable the associated handshake logic on the respective **CTITRIGOUT** bit.

Table A-20 Top-level static configuration signals (continued)

Signal name	Width	Direction	Description
TISBYPASSACK	4	Input	Cross Trigger Interface synchronous bypass on **CTITRIGOUTACK**. If a **CTITRIGOUTACK** input is synchronous to **DEBUGSYSCLK**, and is driven from the same clock domain, tie its associated **TISBYPASSACK** pin HIGH to bypass the synchronization logic.
TISBYPASSIN	8	Input	Cross Trigger Interface synchronous bypass on **CTITRIGIN**. If a **CTITRIGIN** input is synchronous to **DEBUGSYSCLK**, and is driven from the same clock domain, tie its associated **TISBYPASSIN** pin HIGH to bypass the synchronization logic.
TODBGENSEL	4	Input	**DBGEN** mask on **CTITRIGOUT**. When each bit is set to LOW, it masks the associated Cross Trigger Interface's trigger output when **DBGEN** is LOW.
DBGENSELDIS	1	Input	**DBGEN** Selector Disable. When set HIGH, disables the **DBGEN** Selector Logic and forces **DBGEN** to use **DBGENIN**. If the Crypto element exists, this must be tied to HIGH.
NIDENSELDIS	1	Input	**NIDEN** Selector Disable. When set HIGH, disables the **NIDEN** Selector Logic and forces **NIDEN** to use **NIDENIN**. If the Crypto element exists, this must be tied to HIGH.
SPIDENSELDIS	1	Input	**SPIDEN** Selector Disable. When set to HIGH disables the **SPIDEN** Selector Logic and forces **SPIDEN** to use **SPIDENIN**. If the Crypto element exists, this must be tied to HIGH.
SPNIDENSELDIS	1	Input	**SPNIDEN** Selector Disable. When set to HIGH disables the **SPNIDEN** Selector Logic and forces **SPNIDEN** to use **SPNIDENIN**. If the Crypto element exists, this must be tied to HIGH.

Note

For a top-level definition of the configuration points that are used to render the design to the build system, see *Arm® CoreLink™ SSE-200 Subsystem for Embedded Configuration and Integration Manual*.

A.7 CryptoCell-312 signals

The Crypto element implements cryptographic accelerators.

The CryptoCell signals are not present if the Crypto element is absent.

This section contains the following subsections:

A.7.1 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. The following table shows the output signals on the CryptoCell Lifecycle Indication Interface. All signals are synchronous to **CRYPTOSYSCLK** and are on the Always ON power domain.

Table A-21 CryptoCell Lifecycle Indication signals

Signal name	Width	Direction	Description
CRYPTOLCS	3	Output	Lifecycle State values: `0b000` Chip Manufacture (CM). `0b001` Device Manufacture (DM). `0b101` Secure State (SE). `0b111` Return to Manufacturer (RMA).
CRYPTOLCSVALID	1	Output	Lifecycle State values on CRYPTOLCS are Valid, where: 1: **CRYPTOLCS[2:0]** is valid. 0: **CRYPTOLCS[2:0]** is not valid.

A.7.2 CryptoCell Debug Control Enable

The CryptoCell Debug Control Enable Interface provides signals to be used by customers to control the Debug Authentication signals at the top level of system, provide tests, debug, and security-related functionality in SoC.

The following table shows the output signals on the CryptoCell Debug Control Enable Interface. All signals are synchronous to **CRYPTOSYSCLK**.

Table A-22 CryptoCell debug control enable signals

Signal name	Width	Direction	Description
CRYPTODCUEN	127	Output	Debug Control Enable Values.

A.7.3 CryptoCell Non-Volatile Memory Interface

The CryptoCell Non-Volatile Memory Interface is the Crypto interface to the OTP memory.

The following table shows the output signals on the CryptoCell Debug Control Enable Interface. All signals are synchronous to **CRYPTOSYSCLK** and are reset by **nWARMRESETCRYPTO**.

Table A-23 CryptoCell Non-Volatile Memory signals

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
CRYPTONVMPSLVERR	1	Input	APB Slave Error
CRYPTONVMPROGCOMPLETED	1	Input	Program Completed status. This signal indicates that any programming of the OTP-NVM memory is completed.

A.8 Top-level parameters

The SSE-200 provides Verilog parameters that configure many of the SSE-200 features.

The following table lists the top-level Verilog parameters, but excludes the security control expansion, interrupt, and the Crypto element parameters.

Table A-24 Top-level parameters

Parameter	Default value	Description
INITNSVTOR0_RST	0x0000_0000	Sets the reset value of the Non-secure vector table offset address in the Cortex-M33 processor, in CPU element 0.
INITNSVTOR1_RST		Sets the reset value of the Non-secure vector table offset address in the Cortex-M33 processor, in CPU element 1.
SRAM_MPC_BLK_SIZE	3	SRAM MPC block size = $2^{SRAM\_MPC\_BLK\_SIZE + 5}$ bytes: 3 = 256 byte block size. Others = Reserved. The block size must be consistent across different architectures because it has a major impact on the software. Different choices for the block size, increases the software porting effort.
CPU0WAIT_RST	0	For CPU element 1, controls whether the processor waits at the boot phase: 0 = Boot normally. 1 = Wait at boot. From Cold reset, **nSRST** reset, or Watchdog reset, this parameter also controls if CPU 0 powers up: 0 = Power-up. 1 = Do not power up.
CPU1WAIT_RST	1	For CPU element 1, controls whether the processor waits at the boot phase: 0 = Boot normally. 1 = Wait at boot. From Cold reset, **nSRST** reset, or Watchdog reset, this parameter also controls if CPU 1 powers up: 0 = Power-up. 1 = Do not power up.
ACC_WAITN_RST	1	Sets the reset value of the BUSWAIT.ACC_WAITN register bit, which controls the value of the **ACCWAITn** output signal. This parameter can stall accesses from AHB masters until the CPU element restores security state: 1 = When exiting reset, allow AHB masters to access the AHB interconnect. 0 = When exiting reset, prevent AHB masters from accessing the AHB interconnect.
CPU0_CPUID	0x0	Sets the CPUID value in the CPU_IDENTITY register for CPU element 0.
CPU1_CPUID	0x1	Sets the CPUID value in the CPU_IDENTITY register for CPU element 1.

Table A-24 Top-level parameters (continued)

Parameter	Default value	Description
EXP_SYS_ID_PRESENT	0xFFFF	Each bit[n] of this vector defines whether the *Exclusive Access Monitor* (EAM) monitors the AHB master with **HMASTERID** == n. When a master is bypassed by the EAM, the EAM provides a HEXOKAY fail response but the data transfer occurs, that is, data is written to memory. This behavior might not be desirable. Therefore, Arm recommends that for masters with exclusive access capability, you must set the corresponding EXP_SYS_ID_PRESENT[n] bit to 1. Bits[15:0] are IDs for internal use only and are not available on this interface.
SRAM0_BUFFER_ENABLE	0	When set to one, it adds buffering and a 2-cycle latency to an exclusive write access through an EAM that is associated with SRAM0. When enabled, the buffering improves the synthesis timing path through that EAM.
SRAM1_BUFFER_ENABLE	0	When set to one, it adds buffering and a 2-cycle latency to an exclusive write access through an EAM that is associated with SRAM1. When enabled, the buffering improves the synthesis timing path through that EAM.
SRAM2_BUFFER_ENABLE	0	When set to one, it adds buffering and a 2-cycle latency to an exclusive write access through an EAM that is associated with SRAM2. When enabled, the buffering improves the synthesis timing path through that EAM.
SRAM3_BUFFER_ENABLE	1	When set to one, it adds buffering and a 2-cycle latency to an exclusive write access through an EAM that is associated with SRAM3. When enabled, the buffering improves the synthesis timing path through that EAM.
CPU0_FPU	0	Indicates if the *Floating Point Unit* (FPU) is present in CPU element 0: 0 = FPU is not present. 1 = FPU is present.
CPU1_FPU	HAS_FPU	Indicates if the FPU is present in CPU element 1: 0 = FPU is not present. 1 = FPU is present.
CPU0_DSP	0	Controls whether the Cortex-M33 processor, in CPU element 0, supports the DSP extension instructions: 0 = DSP not supported. 1 = DSP supported.
CPU1_DSP	1	Controls whether the Cortex-M33 processor, in CPU element 1, supports the DSP extension instructions: 0 = DSP not supported. 1 = DSP supported.
CPU0_CPIF	0	Controls whether CPU0 element has a coprocessor interface: 0 = Coprocessor interface is not present. 1 = Reserved.
CPU1_CPIF	0	Controls whether CPU1 element has a coprocessor interface: 0 = Coprocessor interface is not present. 1 = Reserved.

Table A-24 Top-level parameters (continued)

Parameter	Default value	Description
CPU0_MPU_NS	8	Sets the number of Non-secure MPU entries in CPU element 0.
CPU1_MPU_NS	8	Sets the number of Non-secure MPU entries in CPU element 1.
CPU0_MPU_S	8	Sets the number of Secure MPU entries in CPU element 0.
CPU1_MPU_S	8	Sets the number of Secure MPU entries in CPU element 1.
CPU0_SAU	8	Sets the number of SAU entries in CPU element 0.
CPU1_SAU	8	Sets the number of SAU entries in CPU element 1.
CPU0_DBGLVL	2	Sets the number of debug resources in CPU element 0: 2 = 4 watchpoint and 8 breakpoint comparators. 1 = 2 watchpoint and 4 breakpoint comparators.
CPU1_DBGLVL	2	Sets the number of debug resources in CPU element 1: 2 = 4 watchpoint and 8 breakpoint comparators. 1 = 2 watchpoint and 4 breakpoint comparators.
CPU0_ICACHESIZE	11	Sets the instruction cache size in CPU element 0: 9 = 512bytes. 10 = 1KB. 11 = 2KB. 12 = 4KB. 13 = 8KB. 14 = 16KB. Others = Reserved.
CPU1_ICACHESIZE	11	Sets the instruction cache size in CPU element 1: 9 = 512bytes. 10 = 1KB. 11 = 2KB. 12 = 4KB. 13 = 8KB. 14 = 16KB. Others = Reserved.
CPU0_ICACHEDMA	0	Defines the existence of micro DMA capability and also line locking capability for the CPU element 0 ICache. When set to 1, the ICache provides cache line prefetch and locking capability.
CPU1_ICACHEDMA	0	Defines the existence of micro DMA capability and also line locking capability for the CPU element 1 ICache. When set to 1, the ICache provides cache line prefetch and locking capability.

Table A-24 Top-level parameters (continued)

Parameter	Default value	Description
CPU0_ICACHESTATS	1	Controls whether the ICache supports statistics functionality in CPU element 0: 1 = Statistics supported. 0 = Statistics not supported.
CPU1_ICACHESTATS	1	Controls whether the ICache supports statistics functionality in CPU element 1: 1 = Statistics supported. 0 = Statistics not supported.
CPU0_ICACHEINVMAT	0	Enable Invalidate on Write Match for the CPU0 element ICache. When set to 1, any writes to a location that also exists in the cache, results in the invalidation of that cache line.
CPU1_ICACHEINVMAT	0	Enable Invalidate on Write Match for the CPU1 element ICache. When set to 1, any writes to a location that also exists in the cache, results in the invalidation of that cache line.
CPU0_XOM	0	Enable CPU0 ICache *Execute Only Memory* (XOM) support. When set to 1, the **HRUSER[0]** signal on the AHB5 Master Expansion Code Interface indicates if the current read data is Execute Only. If the data type access targets a XOM location, the ICache masks the data. When set to 0, the ICache ignores **HRUSER[0]**.
CPU1_XOM	0	Enable CPU1 ICache XOM support. When set to 1, the **HRUSER[0]** signal on the AHB5 Master Expansion Code Interface indicates if the current read data is Execute Only. If the data type access targets a XOM location, the ICache masks the data. When set to 0, the ICache ignores **HRUSER[0]**.
ICACHERRDS	1	Reduce ICache Tag Reads. When set to 1, the ICache masks off an access to the Tag RAM if this set was previously accessed and the RAM data is valid. ——— **Note** ——— This option requires a deselected RAM to continue outputting the last value that was read from it. ———
FCLKDIV_RST	15	Sets the **MAINCLK** to **FCLK** divider ratio at reset. The divider ratio is FCLKDIV_RST[4:0] + 1. You must ensure that the default divider value does not result in an overclocked design after reset.
SYSCLKDIV_RST	0b00000	Sets the **FCLK** to **SYSCLK** divider ratio at reset. The divider ratio is SYSCLKDIV_RST[4:0] + 1. You must ensure that the default divider value does not result in an overclocked design after reset.
FCLK_DIVRATIO_PIPELINE	2	Adds extra clock cycles, or delay cycles, to the resynchronization pipeline that passes the divider ratio values in the **FCLK** generation clock divider. The delay is 3 + FCLK_DIVRATIO_PIPELINE.

Table A-24 Top-level parameters (continued)

Parameter	Default value	Description
SYSCLK_DIVRATIO_PIPELINE	2	Adds extra clock cycles, or delay cycles, to the resynchronization pipeline that passes the divider ratio values in the **SYSCLK** generation clock divider. The delay is 3 + SYSCLK_DIVRATIO_PIPELINE.
SYSRSTREQ0_EN_RST	0	SYSRSTREQ0_EN reset value. SYSRSTREQ0_EN_RST controls the reset value of the SYSRSTREQ0_EN value in the RESET_MASK register. That register bit can mask the reset request from the CPU0 element. When set to 0, at reset, CPU0 element cannot cause a system Warm reset by setting AIRCR.SYSRESETREQ = 1, in its Application Interrupt and Reset Control Register, AIRCR. When set to 1, CPU0 element can cause a system Warm reset by setting AIRCR.SYSRESETREQ = 1. The default value is 0, due to its interaction with the CryptoCell-312 core in the Subsystem.
SYSRSTREQ1_EN_RST	0	SYSRSTREQ1_EN reset value. SYSRSTREQ1_EN_RST controls the reset value of the SYSRSTREQ1_EN value in the RESET_MASK register. That register bit can mask the reset request from the CPU1 element. When set to 0, at reset, CPU1 element cannot cause a system Warm reset by setting AIRCR.SYSRESETREQ = 1, in its Application Interrupt and Reset Control Register, AIRCR. When set to 1, CPU1 element can cause a system Warm reset by setting AIRCR.SYSRESETREQ = 1. The default value is 0, due to its interaction with the CryptoCell-312 core in the Subsystem.
TARGETIDSYS	0x07430477	Sets the Peripheral ID values for the CoreSight ROM table in the Debug element. A Debugger can read this value and discover the product that it connects to. The system integrator must modify the value to use their JEP106 code and their part number for the product. See the *Arm® Debug Interface Architecture Specification ADIv5.0 to ADIv5.2* for more information. Two target system IDs must use different IDs unless the designs they identify are either: • Identical in all ways, including trigger network and the expansion part. • Some configurable pure subset, for example, configurable number of modules.

A.9 Top-level render time configurations

The SSE-200 provides several render configuration options. You can use these options to control whether the subsystem includes the Crypto element, FPUs, which CMSDK product it uses, and whether to improve timing or reduce latency.

The following table lists the configurable render options.

Table A-25 Configurable render options

Parameter name	Default value	Description
HAS_CRYPTO	1	Defines whether to include the Crypto element: 0 = Exclude the cryptographic functionality. The render process instantiates a dummy element that provides no cryptographic functionality. 1 = Include the Crypto element, which contains the TrustZone CryptoCell-312. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Release Note* for more information about the CC010 CryptoCell bundle name and availability.
BASE_MTX_ZERO_LATENCY_ARBITRATION_EN	0	When a downstream port selects a different upstream port to service, this parameter can add latency: 0 = Inserts one extra clock cycle of latency. 1 = Zero extra clock latency added. With this setting, after a locked transaction, the bus matrix does not insert an IDLE transfer. **Note** The *Arm® AMBA® 5 AHB Protocol Specification* recommends that a bus master inserts an IDLE transfer after a locked transfer.
HAS_FPU	1	Defines whether to include the Cortex-M33 processor FPU in the CPU elements: 0 = Exclude the FPU from the CPU element. 1 = Render the FPU in the CPU element. If you set CPU0_FPU or CPU1_FPU to 1, then you must select this value. See *A.8 Top-level parameters on page Appx-A-170* for more information about those two Verilog parameters. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Release Note* for more information about the FPU bundle name.
SRAM_ADDR_WIDTH	15	SRAM bank address width: 15, 16, or 17.
SRAM_NUM_BANK	4	Number of SRAM banks: 4 = 4 banks. Others = Reserved.

Table A-25 Configurable render options (continued)

Parameter name	Default value	Description
CPU0_TYPE	0b0010	Processor in CPU0 element: 0b0000 = Not present. 0b0010 = Cortex-M33 processor. Others = Reserved.
CPU1_TYPE	0b0010	Processor in CPU1 element: 0b0000 = Not present. 0b0010 = Cortex-M33 processor. Others = Reserved.
NUM_AHBSEXP	2	Number of slave AHB expansion ports: 2 = 2 slave expansion ports. Others = Reserved.
CPU0_HAS_TCM	0	CPU0 element has a Data TCM: 0 = Not present. 1 = Reserved.
CPU1_HAS_TCM	1	CPU0 element has a Data TCM: 0 = Not present. 1 = Data TCM is present.
CPU0_TCM_BANK_NUM	0	The SRAM bank that maps to the CPU0 element Data TCM.
CPU1_TCM_BANK_NUM	3	The SRAM bank that maps to the CPU1 element Data TCM.
SEPARATE_CRYPTO_PD	1	PD_CRYPTO power domain exists: 0 = Reserved. 1 = PD_CRYPTO power domain exists if Crypto element exists.
SEPARATE_CPUDBG_PD	0	PD_CPU<N>CORE and PD_CPU<N>DBG are separate power domains (not merged): 0 = Both PD_CPU<N>CORE and PD_CPU<N>DBG are merged to become PD_CPU<N>CORE. PD_CPU<N>DBG does not exist. 1 = PD_CPU<N>CORE and PD_CPU<N>DBG are separate power domains.
CPU_SYS_RETENTION	0	Remove the retention support: 0 = Retention is supported. 1 = Retention is not supported.

Appendix B
Revisions

This appendix describes the technical changes between released issues of this book.

It contains the following section:

- *B.1 Revisions* on page Appx-B-178.

B.1 Revisions

This appendix describes technical changes between released issues of this book.

Table B-1 Issue A

Change	Location	Affects
First release	-	-

Table B-2 Differences between issue A and issue 0100-00

Change	Location	Affects
Numerous minor technical updates applied throughout document.	All sections	r1p0 EAC release
Added Warm reset signal information.	*2.3.2 Reset inputs and outputs* on page 2-28	r1p0 EAC release
Added PPU description to Resets section.	*2.3.5 nWARMRESETAON* on page 2-30	r1p0 EAC release
Multiple changes to processor configuration options	*2.4 CPU elements* on page 2-32	r1p0 EAC release
Debug and Certificate access updates.	*Certificate Access* on page 2-54	r1p0 EAC release
Memory map figure and notes updated.	*3.2.1 Memory map overview* on page 3-74	r1p0 EAC release
CPU_IDENTIY register block section added to CPU element chapter.	*3.3.5 CPU_IDENTITY* on page 3-88	r1p0 EAC release
Interrupt signals updated.	*A.2 Interrupt signals* on page Appx-A-152	r1p0 EAC release
New section added to Appendix: Top-level parameters	*A.8 Top-level parameters* on page Appx-A-170	r1p0 EAC release
New section added to Appendix: Top-level render time configurations	*A.9 Top-level render time configurations* on page Appx-A-175	r1p0 EAC release