Arm® Musca-S1 Test Chip and Board

Technical Reference Manual

arm

101835_0000_00_en

Arm® Musca-S1 Test Chip and Board

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
0000-00	02 December 2019	Non-Confidential	First issue

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

www.arm.com

Conformance Notices

This section contains conformance notices.

Federal Communications Commission Notice

This device is test equipment and consequently is exempt from part 15 of the FCC Rules under section 15.103 (c).

CE Conformity

The *Waste Electrical and Electronic Equipment* (WEEE) marking, that is, the crossed out wheelie-bin figure, indicates that this product must not be disposed of with general waste within the European Union. To prevent possible harm to the environment from uncontrolled waste disposal, the user is required to recycle the product responsibly to promote reuse of material resources. To comply with EU law, you must dispose of the product in one of the following ways:

- Return it to the distributer where it was purchased. The distributer is required to arrange free collection when requested.
- Recycle it using local WEEE recycling facilities. These facilities are now very common and might provide free collection.
- If purchased directly from Arm, Arm provides free collection. Please e-mail *weee@arm.com* for instructions.

The CE Declaration of Conformity for this product is available on request.

The system should be powered down when not in use.

It is recommended that ESD precautions be taken when handling this product.

The product generates, uses, and can radiate radio frequency energy and may cause harmful interference to radio communications. There is no guarantee that interference will not occur in a particular installation. If this equipment causes harmful interference to radio or television reception, which can be determined by turning the equipment off or on, you are encouraged to try to correct the interference by one or more of the following measures:

- Ensure attached cables do not lie across any sensitive equipment.
- Reorient the receiving antenna.
- Increase the distance between the equipment and the receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Note

It is recommended that wherever possible shielded interface cables be used.

Contents

Arm® Musca-S1 Test Chip and Board Technical Reference Manual

Preface

This preface introduces the *Arm® Musca-S1 Test Chip and Board Technical Reference Manual*.

It contains the following:

About this book

This book describes the Arm® Musca-S1 test chip and board.

Intended audience

This book is written for experienced hardware and software developers to enable low-power, secure *Internet of Things* (IoT) endpoint development using the Musca-S1 test chip and board.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction
This chapter introduces the Musca-S1 test chip and Musca-S1 development board.

Chapter 2 Hardware description
This chapter describes the Musca-S1 test chip and Musca-S1 development board.

Chapter 3 Programmers model
This chapter describes the programmers model of the Musca-S1 test chip and board.

Appendix A Signal descriptions
This appendix describes the signals that are present at the board interface connectors.

Appendix B PVT sensors
This appendix describes the *Process, Voltage, and Temperature* (PVT) sensors on the Musca-S1 test chip.

Appendix C IP configuration
This appendix describes the IP configuration of the Musca-S1 test chip.

Appendix D Specifications
This appendix contains electrical specifications of the Musca-S1 development board.

Appendix E 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.

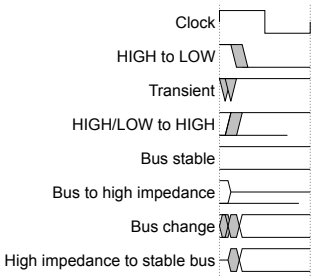

Figure 1 Key to timing diagram conventions

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, n 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® Musca-S1 Test Chip and Board Technical Overview* (101756)
- *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Overview (r1p0)* (101123).
- *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* (101104).
- *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual* (DDI 0571).
- *Arm® Cortex®-M System Design Kit Technical Reference Manual* (DDI 0479).
- *Arm® Cortex®-M33 Processor Technical Reference Manual (r0p2)* (100230).
- *PrimeCell UART (PL011) Technical Reference Manual* (DDI 0183).
- *Arm® PrimeCell Real Time Clock (PL031) Technical Reference Manual* (DDI 0224).
- *CoreSight™ Components Technical Reference Manual* (Arm DDI 0314).
- *Arm® DS-5 Arm DSTREAM User Guide* (Arm DUI 0481).
- *Arm® DS-5 Using the Debug Hardware Configuration Utilities* (Arm DUI 0498).

The following confidential books are only available to licensees or require registration with Arm.

- *Arm® CryptoCell-312 Technical Reference Manual (r1p0)* (100774).
- *Arm® v7-M Architecture Reference Manual* (Arm DDI 0403).
- *Arm® v8-M Architecture Reference Manual* (Arm DDI 0553)
- *Arm® AMBA® 5 AHB Protocol Specification* (Arm IHI 0033).
- *Arm® AMBA® APB Protocol Specification Version 2.0* (Arm IHI 0024).

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 Musca-S1 Test Chip and Board Technical Reference Manual*.
- The number 101835_0000_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 Musca-S1 test chip and Musca-S1 development board.

It contains the following sections:

1.1 Precautions

This section describes precautions that ensure safety and prevent damage to your Musca-S1 development board.

This section contains the following subsections:

1.1.1 Ensuring safety

The Musca-S1 development board operates at 5V supplied through the DAPLink 5V USB connector.

Warning

Do not use the Musca-S1 development board near equipment that is sensitive to electromagnetic emissions, for example, medical equipment.

1.1.2 Operating temperature

The Musca-S1 development board has been tested in the temperature range 15°C-30°C.

1.1.3 Preventing damage

The Musca-S1 development board is intended for use within a laboratory or engineering development environment.

Caution

To avoid damage to the Musca-S1 development board, observe the following precautions:

- Never subject the board to high electrostatic potentials. Observe *ElectroStatic Discharge* (ESD) precautions when handling any board.
- Always wear a grounding strap when handling the board.
- Only hold the board by the edges.
- Avoid touching the component pins or any other metallic element.
- Do not fit an Arduino Expansion Shield while the Musca-S1 development board is powered up.

1.2 About the Musca-S1 test chip and board

The Musca-S1 development board provides access to the Musca-S1 test chip.

Musca-S1 test chip and board

The Musca-S1 test chip demonstrates the foundation of single-chip secure *Internet of Things* (IoT) endpoints. The architecture integrates the recommendations of *Platform Security Architecture* (PSA) using the same subsystem as Musca-A (Arm CoreLink SSE-200 Subsystem for Embedded) but with the addition of:

- Dual on-chip eMRAM and SRAM.
- Secure memory subsystems.
- PSA Level 1 and Functional API certification.

The Musca-S1 test chip implements a SSE-200 subsystem (r1p0) in Samsung Foundry 28nm, *Fully Depleted Silicon on Insulator process* (28FDS). The implementation is ready to be used to form the core processing element of energy-efficiency mainstream IoT devices with secure PSA *Root-of-Trust* (RoT). Musca-S1 can also be used to prototype secure boot, on-chip storage execution and network device management through Trusted Firmware-M (TF-M), Arm Mbed™ OS, and Arm Pelion™ IoT platform integration.

The Musca-S1 development platform is bootable from on-chip eMRAM or off-chip QSPI (on-board Flash.

Major components and systems

The Musca-S1 development board provides the following main features:

- Musca-S1 test chip that includes, but is not limited to, the following:
  - CoreLink SSE-200 subsystem that contains two Arm Cortex-M33 (r0p2) processors.
  - 2MB on-chip eMRAM.
  - 2MB on-chip Code SRAM.
  - Peripheral and Arduino Expansion Shield interfaces.
- On-board DAPLink that provides the following access:
  - *Serial Wire or JTAG Debug Port* (SWJ-DP).
  - *USB Mass Storage Device* (USBMSD) for uploading new firmware.
  - USB serial port. The UART to the DAPLink does not support hardware flow control.
  - Remote reset.
- On-board:
  - 3-axis orientation and motion sensor (gyro sensor).
  - Temperature sensor/ADC/DAC.
  - *Quad Serial Peripheral Interface* (QSPI) 32MB boot flash.
- P-JTAG processor debug and SWD header.
- User RGB LED, status LEDs, user reset, and On/Off push buttons.
- The board is powered from USB 5V power or Li-ion rechargeable battery backup, battery not supplied, selectable by a jumper link.
- Headers for Arduino Expansion Shield to support development of custom designs:
  - 16 3V3 or 1V8 GPIO.
  - UART.
  - SPI.
  - I^2C.
  - I^2S three-channel, master only.
  - 3-channel *Pulse Width Modulation* (PWM).
  - 6-channel analog interface from the on-board combined ADC, DAC, and GPIO.

1.3 The Musca-S1 development board at a glance

The following figure shows the physical layout of the upper face of the Musca-S1 development board.

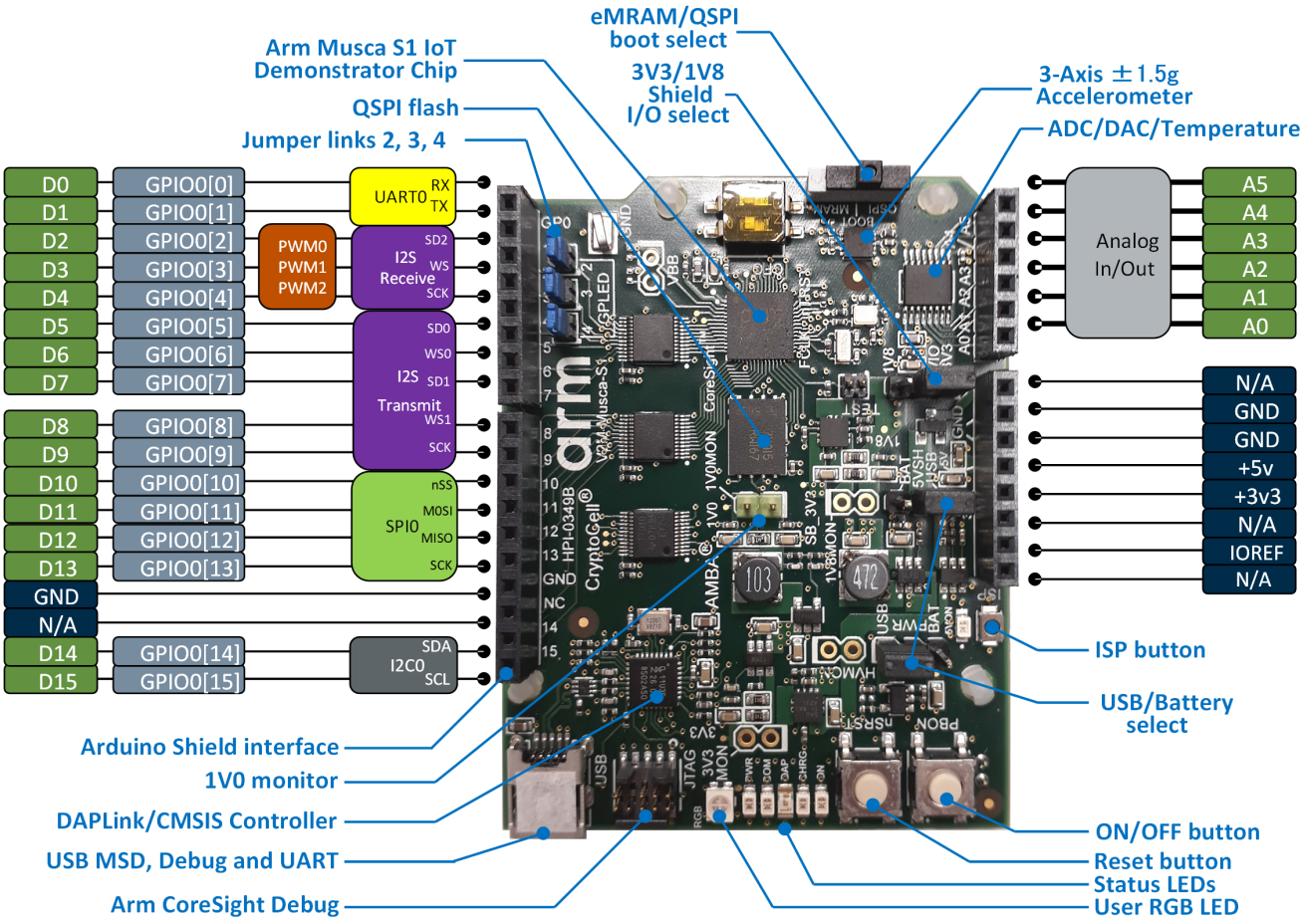

Figure 1-1 Musca-S1 development board

Note

The figure shows the functions that are multiplexed onto the Musca-S1 test chip I/O and which are available on the Arduino Expansion Shield.

See the following for more information:

- *2.2.2 Test chip multiplexed I/O* on page 2-23
- *3.11.1 IOMUX registers* on page 3-120

The following table describes the Musca-S1 development board components.

Table 1-1 Board components

Component	Comment
Musca-S1 test chip	Samsung Foundry 28FDS eMRAM-enabled *Internet of Things* (IoT) test chip.
Boot selector slider switch	eMRAM or QSPI
Expansion Shield analog I/O connector	1V8 or 3V3 I/O. Selected by jumper link J12.

Table 1-1 Board components (continued)

Component	Comment
Expansion Shield power and voltage reference connector	-
Jumper link. Expansion Shield I/O voltage selector J12.	1V8 or 3V3. Default 3V3.
Jumper link. Expansion Shield power supply selector J19.	Battery or USB. Default USB. Use with jumper link J18.
Jumper link. Board power supply selector J18.	Battery or USB. Use with jumper link J19. Default USB.
1V0 supply test point	Musca-S1 test chip core power supply.
ISP push button DAPLink update	To update DAPLink.
On/Off push button (PBON)	Labeled PBON on board
Cortex-M33 system reset and CoreSight component rest (nSRST)	Labeled nRST on board
ON LED	Green system LED. Board power supplies are active. Next to nRST button.
CHRG LED	Orange system LED. Li-ion battery charging in progress. Next to ON LED.
DAP LED	Blue system LED. DAP activity. Next to CHRG LED.
COM LED	Green system LED. USB UART activity. Next to DAP LED.
PWR LED	Orange system LED. Power is connected. Next to COM LED.
RGB user LED	Jumper 2 connects red to GPIO[2] and to Expansion Shield digital I/O connector 1 (17). Jumper 3 connects green to GPIO[3] and to Expansion Shield digital I/O connector 1 (17). Jumper 4 connects blue to GPIO[4]v and to Expansion Shield digital I/O connector 1 (17). Next to PWR LED.
CoreSight debug connector	SWJ-DP
USB mini B connector	-
Expansion Shield digital I/O connector	Jumper J12 selects 1V8 or 3V3.

1.4 Getting started

The Musca-S1 development board is controlled from a USB port that supports UART, *Mass Storage Device* (MSD), and CoreSight debug connection methods.

The board is factory-programmed with the DAPLink firmware and binary QSPI image to enable bootup.

Powering up into the operating state

The minimum actions to boot the development board are as follows:

1. Set the boot select switch to QSPI.
2. Connect a USB cable to the board.
3. Press the PBON button.
4. Connect a serial terminal to the USB UART. The serial port settings must be:
   - 115.2kBaud.
   - 8N1.
   - No hardware or software flow control.

To load a new user image, drag and drop the new image onto the drive labeled MUSCA_S.

Chapter 2
Hardware description

This chapter describes the Musca-S1 test chip and Musca-S1 development board.

It contains the following sections:

2.1 Board hardware

The hardware infrastructure of the Musca-S1 development board provides access to the Musca-S1 test chip and supports Shield expansion.

Overview of the Musca-S1 development board hardware

The test chip interfaces connect directly between the test chip and the peripheral devices on the board, and between test chip and the Shield headers.

The following figure shows the hardware infrastructure of the Musca-S1 development board.

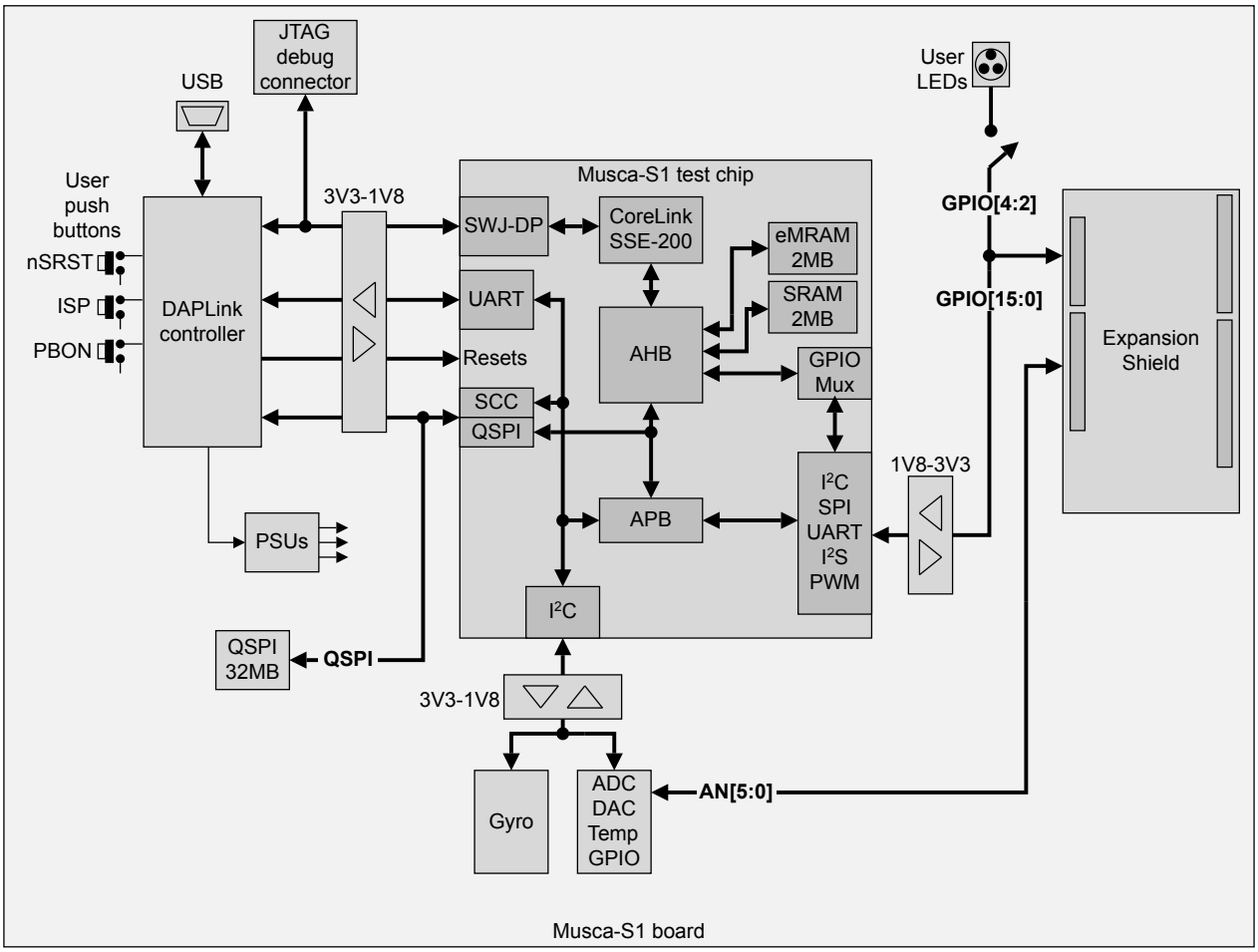

Figure 2-1 Hardware infrastructure of the Musca-S1 development board

Musca-S1 development board components and systems

The development board contains the following components and systems:

- One Musca-S1 test chip with CoreLink SSE-200 Subsystem for Embedded (r1p0). The SSE-200 subsystem includes, but is not limited to, the following:
  - CPU0: One Cortex-M33 (r0p2) processor. *Floating-point unit* (FPU), DSP, no coprocessor.
  - CPU1: One Cortex-M33 (r0p2) processor. FPU, DSP, no coprocessor. Clock system enables operation at ×N speed of CPU0 processor. Body-bias enabled transistors for low-power mode.
  - One 2KB instruction cache and one 2KB data cache for each processor.
  - 4 × 128KB SRAM. One 128KB bank, SRAM3, functions as *Tightly-Coupled Memory* (TCM), Tightly-Coupled to CPU1 and operates at CPU1 clock speed.

  - CryptoCell™-312 (r1p0) with 1Kbyte *One Time Programming* (OTP) emulated using simple registers that are reset by powerup only.
  - Timer, Watchdog peripherals, and system control.

  The following on-chip blocks are outside the SSE-200 subsystem.
  - 2MB Code SRAM.
  - 2×1MB eMRAM.
  - Clock system. Input clock sources from development board.
- Arduino Shield expansion with on-board level converters and a jumper link to enable the Shield voltage to be either the SoC voltage, 1V8, or 3V3. Enables custom designs by providing the following interfaces:
  - UART/USART.
  - I^2S, three-channel, master only.
  - SPI.
  - I^2C.
  - PWM.
  - 6-channel analog interface from the on-board combined ADC, DAC, and GPIO.
  - 16 1V8 or 3V3 GPIO.
  - 1Hz clock.
- On-board DAPLink that enables the following functionality over USB:
  - *Serial Wire Debug* (SWD).
  - *USB Mass Storage Device* (USBMSD) for uploading new firmware.
  - USB serial port. The UART to the DAPLink does not support hardware flow control.
  - Remote reset.
- On-board gyro sensor:
  - MMA7660FC 3-axis orientation and motion detection sensor.
  - I^2C interface to test chip.
- On-board combined ADC/DAC/temperature sensor:
  - AD5593.
  - 6-channel 3V3 ADC/DAC/GPIO interface to Arduino Shield.
  - Temperature indicator.
- Programmable boot select:
  - 32MB On-board QSPI boot flash.
  - 2×1MB on-chip boot eMRAM.
  - 2MB on-chip code SRAM, after being preloaded with execution code.
  - Both Secure and Non-secure access.
- Debug connector that provides access to:
  - P-JTAG processor debug.
  - *Serial Wire Debug* (SWD).
- User push-button:
  - PBON On/Off push-button.
  - nSRST: Cortex-M33 system reset and CoreSight component reset.
  - ISP: Updates DAPLink firmware.
- RGB LED. Jumper connectors provide optional connections between the Arduino Expansion header and the Musca-S1 test chip:
  - Red LED connected to GPIO[2] pin, optional PWM0.
  - Green LED connected to GPIO[3] pin, optional PWM1.
  - Blue LED connected to GPIO[4] pin, optional PWM2.
- Status LEDs.
- 5V USB or battery power, selectable by slider switch:
  - DAPLink 5V USB connector.
  - CLN 523450, Lithium Ion, 3.7V, 950mAh (not supplied).

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

2.2 Musca-S1 test chip

The Musca-S1 test chip is based on the SSE-200 subsystem which features two Cortex-M33 processors.

This section contains the following subsections:

2.2.1 Overview of the Musca-S1 test chip

The SSE-200 subsystem is version r1p0 and the Cortex-M33 processors are version r0p2.

The test chip also implements a memory subsystem, external device interfaces, a clock generator, and *Serial Configuration Control* (SCC) registers for setting default powerup values.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information on the SSE-200 subsystem:

The following figure shows a high-level view of the architecture of the Musca-S1 test chip.

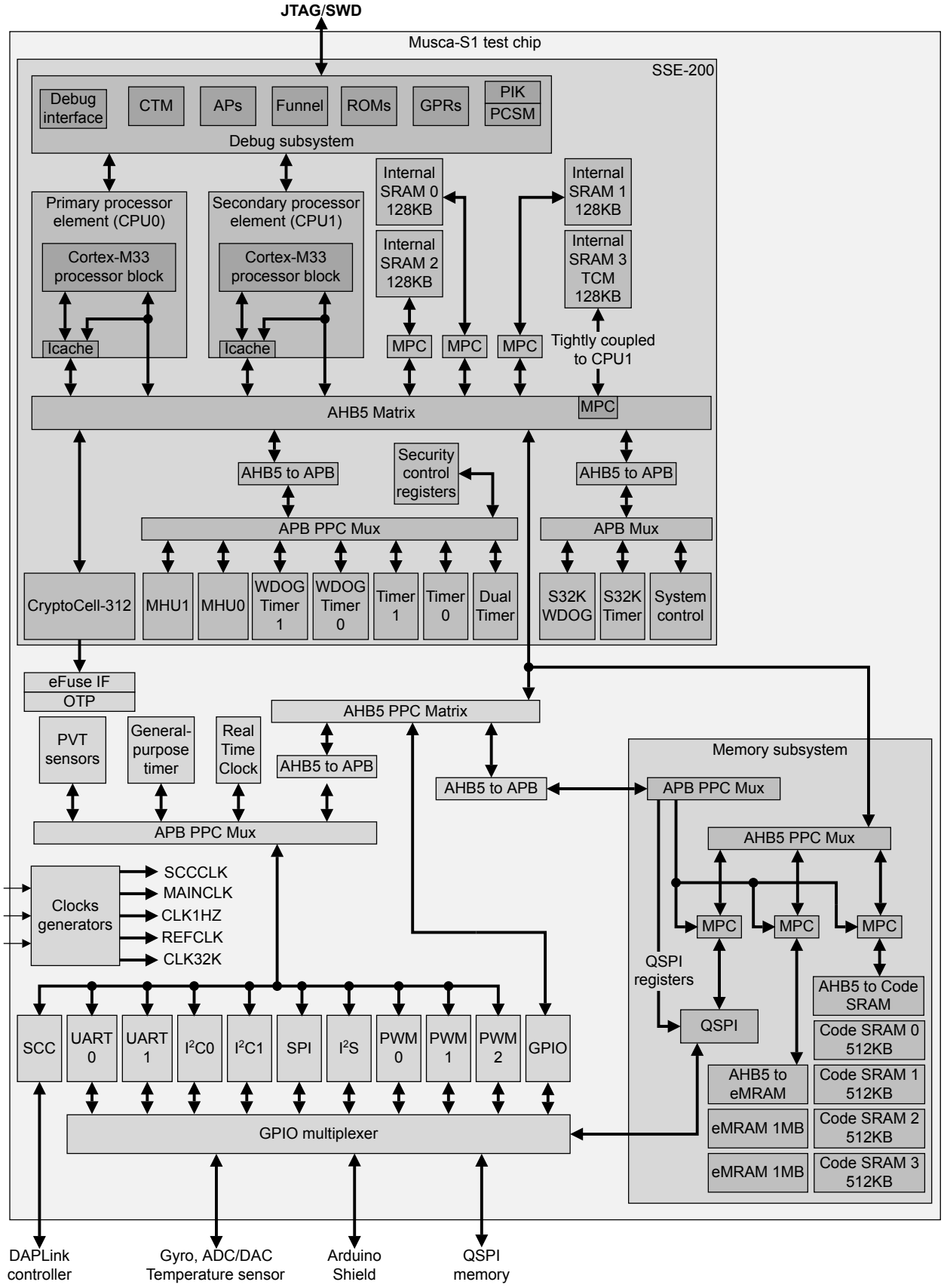

Figure 2-2 Musca-S1 test chip

Major components and systems of the Musca-S1 test chip

SSE-200 subsystem (r1p0)

- Two Cortex-M33 (r0p2) processors with FPU and DSP, and with no coprocessor:
  - CPU0: 50MHz. Used as main processor.
  - CPU1: 200MHz.
- Memory system:
  - One 2KB instruction cache and one 2KB data cache for each Cortex-M33 processor.
  - 4×128KB SRAM. One bank, SRAM3, functions as *Tightly Coupled Memory* (TCM), Tightly Coupled to CPU1 and operating at CPU1 clock speed.
- *Serial Wire Debug* (SWD).
- Secure AMBA interconnect:
  - AHB5 Bus matrix.
  - AHB5 *Exclusive Access Monitors* (EAMs).
  - AHB5 *Access Control Gates* (ACGs).
  - AHB5 to APB bridges.
  - Expansion AHB5 master and slave buses - two of each.
- Security components:
  - AHB5 TrustZone® *Memory Protection Controllers* (MPCs).
  - AHB5 TrustZone *Peripheral Protection Controllers* (PPCs).
  - *Implementation Defined Attribution Unit* (IDAU).
  - CryptoCell-312 (r1p0).
  - Secure and Non-secure configurable peripherals and memory access.
  - Secure boot.
- Secure APB peripherals:
  - One general-purpose timer with configurable security in the **S32KCLK** domain.
  - Two CMSDK timers, Timer0 and Timer1 with configurable security, in the **SYSCLK** domain.
  - One *Cortex®-M System Design Kit* (CMSDK) dual timer with configurable security.
  - One secure watchdog in the **S32KCLK** domain.
  - One secure watchdog in the **SYSCLK** domain.
  - One Non-secure watchdog in the **SYSCLK** domain.

Musca-S1 test chip outside the SSE-200 subsystem

- Two 1MB eMRAM memories.
- 2MB Code SRAM: 4 × 512KB independently power-enabled.
- One *Real Time Clock* (RTC) in the Always ON domain.
- One 32-bit general-purpose timer running at 32.768kHz with programmable interrupts.
- 16 external GPIO interrupts.
- 16 GPIO.
- Three *Process, Voltage, and Temperature* (PVT) sensors:
  - 501-stage ring oscillators. Software can read data from the sensors in the sensor peripheral and group registers.
- Three-channel I^2S:
  - Two master transmitters.
  - One master receiver.
- Three independent *Pulse Width Modulation* (PWM) outputs.
- Two UARTs, UART0 and UART1. The default connectivity is:
  - UART0 to Shield header.
  - UART1 to DAPLink. No hardware flow control.
- Two I^2C, I^2C0 and I^2C1, which can be used as master (default), or slave:
- One SPI interface which can be used as master (default), or slave.
- One alternate function I/O multiplexer.
- One QSPI for external flash control with *Execute in Place* (XIP) capability.
- Programmable boot select:
  - Internal Code eMRAM.
  - External QSPI Flash.
- External powerup reset.
- Three system clock sources:
  - External **REFCLK**, 32.768kHz.
  - External **FASTCLK**, 32MHz.
  - On-chip PLL. Output up to 200MHz.
- One JTAG/SWD debug port.
- One *Serial Configuration Controller* (SCC) with dual access port:
  - SCC serial during reset, accessible by DAPLink only while chip is under powerup reset.
  - APB after reset, accessible by software, or DAPLink while in debug mode (after reset is released).

2.2.2 Test chip multiplexed I/O

The Musca-S1 test chip contains interfaces that are multiplexed onto the Musca-S1 test chip I/O. The IOMUX registers control the GPIO multiplexer that selects the functions that appear at the Musca-S1 test chip I/O.

The IOMUX registers are part of the *Serial Configuration Control* (SCC) registers that select the ALTF1 or ALTF2 alternative I/O functions.

See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer and the IOMUX registers.

——— **Note** ———

The IOMUX registers select each Musca-S1 test chip I/O individually.

———————————

The following table shows the multiplexed Musca-S1 test chip I/O.

Table 2-1 Multiplexed Musca-S1 test chip I/O

Test chip pin	Primary reset or powerup	ALTF1	ALTF2	ALTF3	Destination interface
PA0	GPIO[0]	UART0 RxD	Reserved	Reserved	Arduino Shield
PA1	GPIO[1]	UART0 TxD			
PA2	GPIO[2]	MR_I^{2}S_SD2	PWM0		
PA3	GPIO[3]	MR_I^{2}S_WS	PWM1		
PA4	GPIO[4]	MR_I^{2}S_SCK	PWM2		
PA5	GPIO[5]	MT_I^{2}S_SD0	UART0_CTS		
PA6	GPIO[6]	MT_I^{2}S_WSO	UART0_RTS		
PA7	GPIO[7]	MT_I^{2}S_SD1	UART1_CTS		
PA8	GPIO[8]	MT_I^{2}S1_WS1	UART1_RTS		
PA9	GPIO[9]	MT_I^{2}S_SCK	UART0_SCLK		
PA10	GPIO[10]	SPIO nSS0	Reserved		
PA11	GPIO[11]	SPIO MOSI			
PA12	GPIO[12]	SPIO MISO			
PA13	GPIO[13]	SPIO SCK OUT			
PA14	GPIO[14]	I^{2}C0 Data			
PA15	GPIO[15]	I^{2}C0 Clock	GPIO[0]		
PA16	UART1 RxD	Reserved	GPIO[1]		DAPLink
PA17	UART1 TxD		GPIO[2]		DAPLink
PA18	I^{2}C1 Data		GPIO[3]		Board I^{2}C
PA19	I^{2}C1 Clock		GPIO[4]		Board I^{2}C
PA20	QSPI CS1		GPIO[5]		QSPI
PA21	QSPI IOF0	SCC_LOAD	GPIO[6]		QSPI
PA22	QSPI IOF1	SCC_WNR	GPIO[7]		QSPI
PA23	QSPI IOF2	SCC_DATAIN	GPIO[8]		QSPI
PA24	QSPI IOF3	SCC_CLK	GPIO[9]		QSPI
PA25	QSPI SCLK	SCC_DATAOUT	GPIO[10]		QSPI
PA26	BOOT	Reserved	GPIO[11]		Boot selector slider switch

Note

MT stands for *Master Transmitter*. MR stands for *Master Receiver*.

Related information

3.11.1 IOMUX registers on page 3-120

2.3 Software, firmware, board, and tools setup

Arm supplies software and firmware for the Musca-S1 development board.

You can access software and firmware at the Arm Community pages which are accessible from *https://www.arm.com/musca*.

Connecting to the board

To power the board, connect the USB port to your computer and press the PBON user push button. The DAPLink interface appears in the Windows device manager as an Arm Mbed composite device, part of which is the Mbed serial port, UART. The following figure shows an example configuration that contains the Mbed composite device and the Mbed serial port.

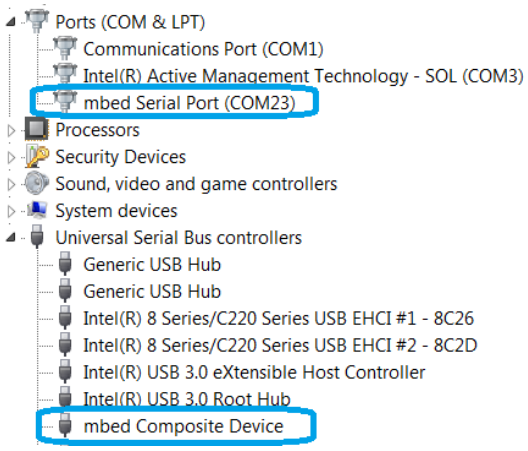

Figure 2-3 DAPLink interface

Note

- Other components of the Mbed composite device are not visible in the Windows device manager. See *2.1 Board hardware* on page 2-18 for the other components of the Mbed composite device.
- UART1 to the DAPLink does not support hardware flow control.

Updating DAPLink firmware

To update the DAPLink firmware, you can use the DAPLink drag and drop update method:

1. Press and hold the ISP button while powering up the board using the USB lead.
2. Delete the `firmware.bin` file that appears in the `CRP DISABLD USB` drive.
3. Copy `DAPLink_S1_DUAL.vxxx.bin` to the `CRP DISABLD drive`.
   - From a Windows system, you can simply Drag and Drop the file.
   - On Linux/Mac OS, use the following command:

     ```
     dd if={new_firmware.bin} of=/Volumes/CRP\ DISABLD/firmware.bin conv=notrunc
     ```
4. Power cycle the board using the USB lead. Do not press the ISP button during the power cycle.

Updating the application software image

To update the application image, perform the following steps:

1. Ensure that the boot switch is set to QSPI or eMRAM as required.
2. Power up the board by connecting the USB lead and pressing the PBON button.
3. Drop a `.bin` format software image onto the `MBED` drive, for example `blinky.bin`. The software image is programmed to QSPI or eMRAM depending on the boot selector slider switch position.
4. Power cycle the board or press the nSRST button to reset the system and boot from the new QSPI or eMRAM software image.

Note

The file `blinky.bin` is available at the Arm Community pages which are accessible from *https://www.arm.com/musca*.

DAPLink UART setting

The default DAPLink UART setting is 115,200 baud (8N1).

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

2.4 User components and status LEDs

The Musca-S1 development board provides three user LEDs, reset and on/off push buttons, a DAPLink reset push button, a power-selector slider switch, and system status LEDs.

User LEDs

One RGB LED, which can be connected to Musca-S1 test chip GPIO[4:2] outputs by completing the user jumper connections:

- Red: Jumper J4 connects this LED to GPIO[2].
- Green: Jumper J3 connects this LED to GPIO[3].
- Blue: Jumper J2 connects this LED to GPIO[4].

Note

GPIO[4:2] are on multiplexed Musca-S1 test chip I/O pins. The I/O multiplexer selects the signals that appear on these pins. See the following for information on how to select the required functions at the Musca-S1 test chip I/O pins:

- *2.2.2 Test chip multiplexed I/O* on page 2-23.
- *3.11.1 IOMUX registers* on page 3-120.

Jumper links

A jumper link selects either USB 5V power or Li-ion 3.7V battery power:

- DAPLink 5V USB connector.
- CLN 523450, Lithium Ion, 3.7V, 950mAh, not supplied.

A jumper link selects either 3V3 or 1V8 power for the Arduino Expansion Shield.

See *1.3 The Musca-S1 development board at a glance* on page 1-14 for the location of the jumper links on the board.

Status LEDs

The Musca-S1 development board provides the following system status LEDs:

- PWR: Orange LED. Indicates that power is connected.
- COM: Green LED. Indicates that USB UART is active.
- DAP: Blue LED. Indicates DAP activity.
- CHRG: Orange LED. Indicates that Li-ion battery charging is in progress.
- ON: Green LED. Indicates that board power supplies are active.

User push buttons

The Musca-S1 development board provides the following user push buttons:

- PBON power on/off.
- nSRST: Cortex-M33 system reset and CoreSight debug reset.
- ISP: Updates DAPLink firmware.

See *2.7 Resets and powerup* on page 2-33 for more information on the user push buttons.

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

2.2.2 Test chip multiplexed I/O on page 2-23

3.11 Serial Configuration Control registers on page 3-120

2.7 Resets and powerup on page 2-33

2.5 Clocks

The Musca-S1 development board provides on-board clocks that drive the systems in the Musca-S1 test chip and development board.

Overview of clock system

The on-board clocks are:

FASTCLK
32MHz on-board oscillator clock.

32K
32.768kHz from on-board crystal oscillator, the default system clock. The clock goes to a PLL in the Musca-S1 test chip and is multiplied up to drive the Cortex-M33 processors and SSE-200 subsystem.

SCCCLK
Serial Configuration Controller (SCC) interface clock from the DAPLink.

JTAG TCK
Input clock from the debug connector to the CoreSight components on the chip.

The SCC registers select either **32K** or **FASTCLK** clocks to drive the Musca-S1 test chip. The default chip driver clock is **32K**.

The driver clock goes to an on-chip PLL and divider system. The on-chip system multiplies the clock frequency to drive the Cortex-M33 processors, the SSE-200 subsystem, and other blocks.

The following figure shows the Musca-S1 test chip and board clock system.

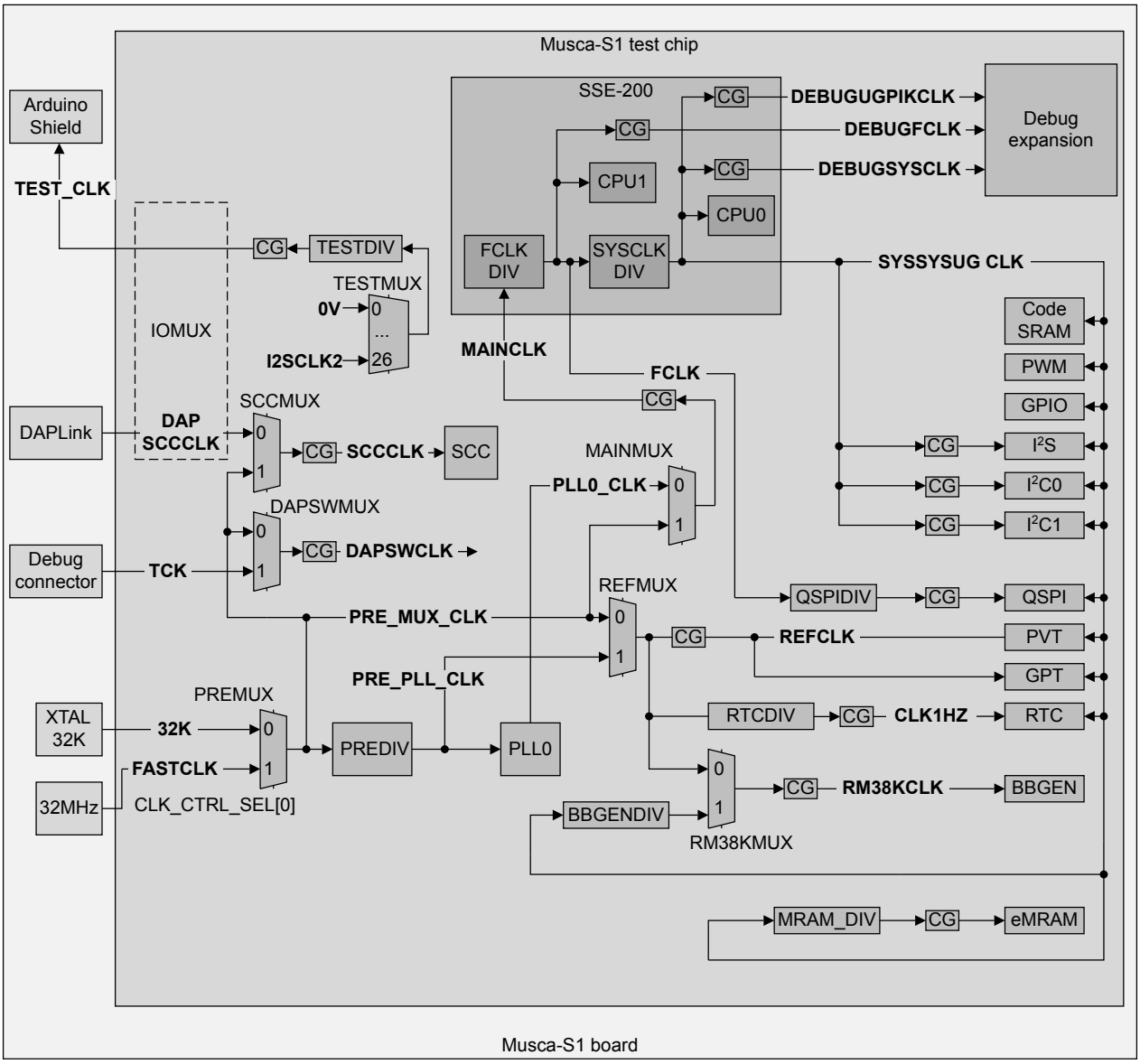

Figure 2-4 Musca-S1 clock system

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for information on the clock system in the SSE-200 subsystem.

Controlling clock frequencies

The SCC registers control the clock system. See *3.11 Serial Configuration Control registers* on page 3-120. The following table shows the SCC clock control registers.

Table 2-2 Clock control SCC registers

Register	Register function	Register description
CLK_CTRL_SEL	Controls the following blocks: • PREMUX • DAPSWMUX • MAINMUX • REFMUX • SCCMUX • RM38KMUX • TESTMUX	*CLK_CTRL_SEL Register* on page 3-126.
CLK_PLL_PREDIV_CTRL	Controls PREDIV.	*CLK_PLL_PREDIV_CTRL Register* on page 3-128.
CLK_BBGEN_DIV_CLK	Controls BBGENDIV	*CLK_BBGEN_DIV_CLK Register* on page 3-129
CLK_POSTDIV_CTRL_QSPI	Controls QSPIDIV.	*CLK_POSTDIV_CTRL_QSPI Register* on page 3-129.
CLK_POSTDIV_CTRL_RTC	Controls RTCDIV.	*CLK_POSTDIV_CTRL_RTC Register* on page 3-130.
CLK_POSTDIV_CTRL_TEST	Controls TESTDIV.	*CLK_POSTDIV_CTRL_TEST Register* on page 3-130.
CTRL_BYPASS_DIV	Controls the clock divider bypass functions.	*CTRL_BYPASS_DIV Register* on page 3-131.
CLK_CTRL_ENABLE	Enables *Clock Gates* (CGs).	*CLK_CTRL_ENABLE Register* on page 3-133.
SCC_MRAM_CTRL0	Enables eMRAM clock CG.	*SCC_MRAM_CTRL0 Register* on page 3-158
SCC_MRAM_CTRL1	Controls eMRAM_DIV	*SCC_MRAM_CTRL2 Register* on page 3-161

The FCLK_DIV and SYSCLK_DIV system control registers control the FCLKDIV and SYSCLKDIV dividers in the SSE-200 subsystem. FCLKDIV derives clock **FCLK** for secondary processor CPU1 and SYSCLKDIV derives **SYSCLK** for primary processor CPU0.

The following table shows system control registers FCLK_DIV and SYSCLK_DIV.

Table 2-3 System control registers FCLK_DIV and SYSCLK_DIV

Register	Register function	Register description
FCLK_DIV	Controls divider block FCLKDIV in SSE-200 subsystem to derive clock **FCLK** for secondary processor CPU1.	*FCLK_DIV Register* on page 3-98.
SYSCLK_DIV	Controls divider block SYSCLKDIV in SSE-200 subsystem to derive clock **SYSCLK** for primary processor CPU0.	*SYSCLK_DIV Register* on page 3-99.

Multiplexed I/O

The **DAPSCCCLK** signal is present on Musca-S1 test chip I/O PA24 which is part of the multiplexed Musca-S1 test chip I/O. The IOMUX registers control the multiplexed Musca-S1 test chip I/O.

Note

The **DAPSCCCLK** input is reserved. In normal operation, software must not change **PRE_MUX_CLK** as the input to multiplexer SCCMUX. See *CLK_CTRL_SEL Register* on page 3-126.

TEST_CLK is present on Musca-S1 test chip I/O PA13 which is also part of the multiplexed Musca-S1 test chip I/O. The IOMUX registers select **TEST_CLK** by selecting alternative function ALTF2 for Musca-S1 test chip I/O PA13.

See the following for information on the multiplexed Musca-S1 test chip I/O and how to select wanted signals at the Musca-S1 test chip I/O pins:

- *2.2.2 Test chip multiplexed I/O* on page 2-23
- *3.11.1 IOMUX registers* on page 3-120.

Related information

3.11.2 SCC registers summary on page 3-123

2.2.2 Test chip multiplexed I/O on page 2-23

3.11.1 IOMUX registers on page 3-120

2.6 CryptoCell™-312 and One Time Programmable security system

The Musca-S1 test chip implements an Arm CryptoCell-312 (r1p0) security subsystem and emulates *One Time Programming* (OTP) secure memory.

CryptoCell-312, in the SSE-200 subsystem, is a cryptographic module that provides fundamental security services to the Cortex-M33 processors and protects them against unauthorized access.

The emulated OTP secure memory consists of registers which emulate non-volatile memory. When bits in the emulated OTP secure memory have been programmed to `0b1`, they are permanent and cannot be cleared until the chip is powered down. The emulated OTP secure memory is connected exclusively to the CryptoCell-312.

The CryptoCell and emulated OTP memory can be used to demonstrate and develop life-cycle management, key storage, and non-volatile firmware counters, and serves as the *Root of Trust* (RoT) for the entire system.

See *3.10 CryptoCell™-312 and One-Time Programmable (OTP) secure memory locations* on page 3-119 for the base addresses of the CryptoCell-312 and OTP registers.

Contact Arm for more information about CryptoCell-312 and the OTP registers.

Related information

2.2.2 Test chip multiplexed I/O on page 2-23
3.11 Serial Configuration Control registers on page 3-120
3.10 CryptoCell™-312 and One-Time Programmable (OTP) secure memory locations on page 3-119

2.7 Resets and powerup

The Musca-S1 development board provides standard resets that the DAPLink controller drives.

Resets

The Musca-S1 development board provides the following resets:

- **CFG_nRST**, the *Serial Configuration Controller* (SCC) interface reset.
- **CB_nRST**, the logic reset.
- **CS_nSRST**, the system reset to the Cortex-M33 processors and the CoreSight components.

User push buttons

The Musca-S1 development board supplies the following user push buttons:

- PBON, the on/off push-button. This button powers up, or powers down, the board.
- nSRST. Generates the reset signal **CS_nSRST**.

Reset sequence

The following figure shows the reset and powerup timing cycle including Musca-S1 test chip and board configuration.

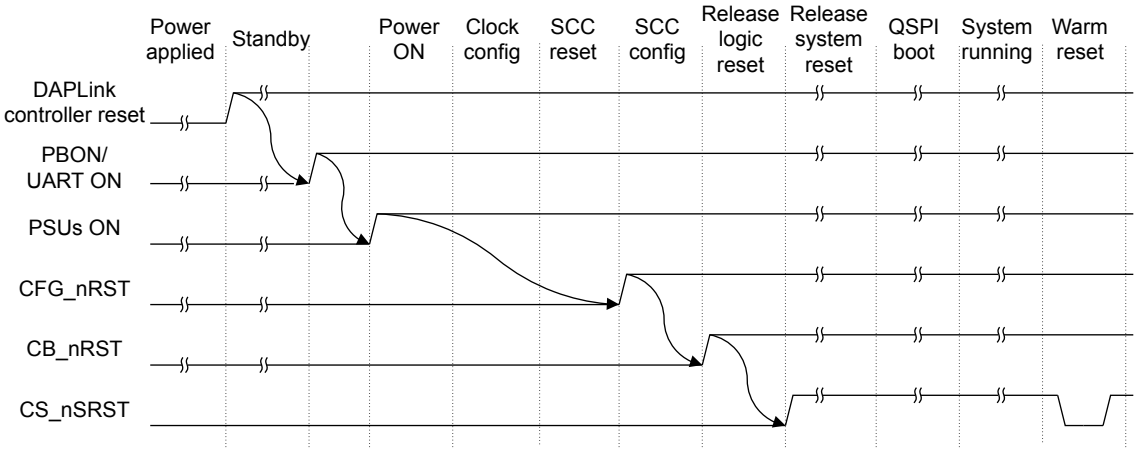

Figure 2-5 Musca-S1 test chip and board reset and configuration timing

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

2.4 User components and status LEDs on page 2-27

2.8 Power

The DAPLink 5V USB connector supplies the power requirements of the Musca-S1 development board. The board also supports use of an external battery as an alternative to the 5V USB supply.

Overview of board power

The Musca-S1 development board provides on-board regulators to supply power rails on the board and to the test chip. The following figure shows the Musca-S1 development board power supplies.

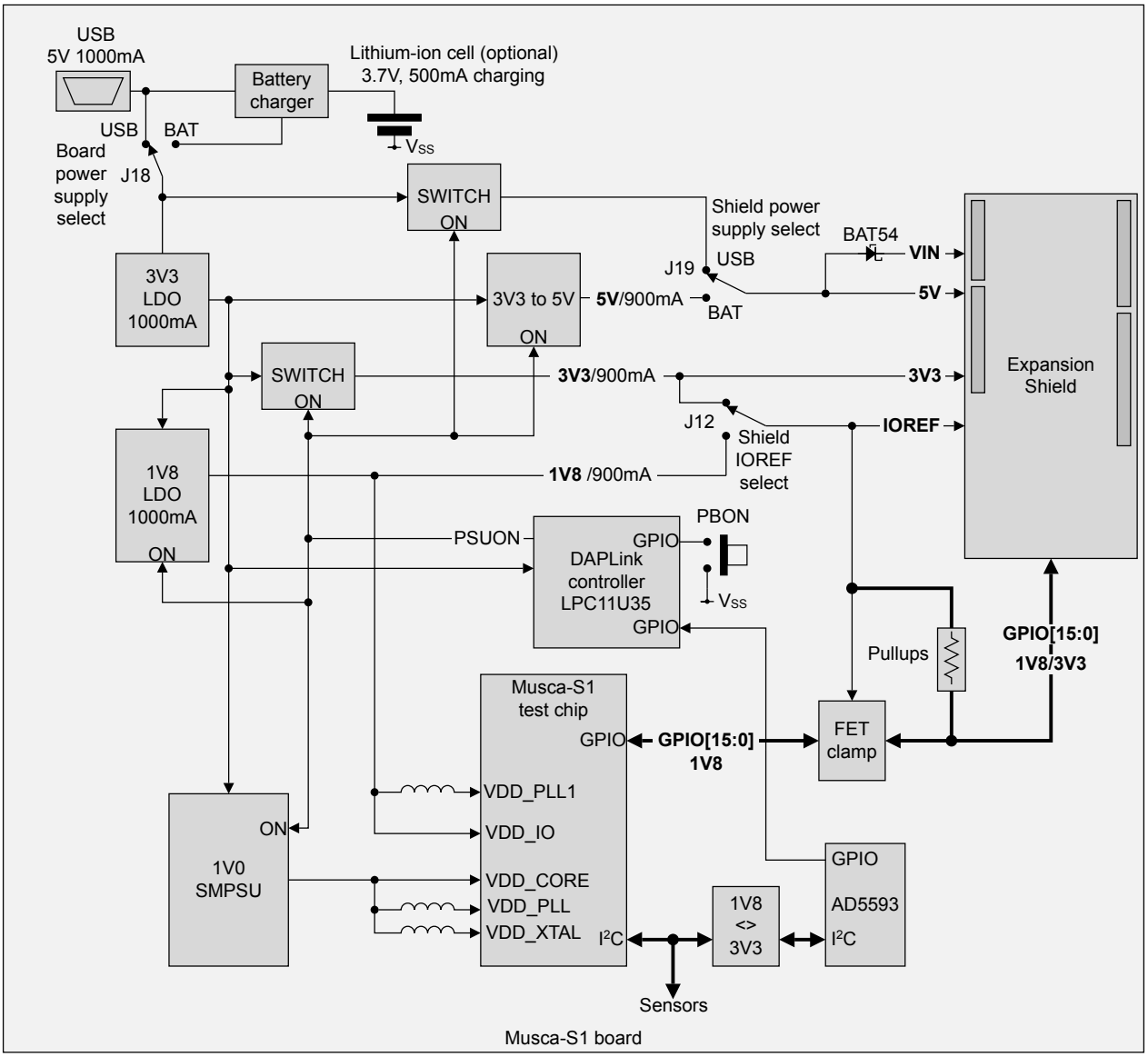

Figure 2-6 Musca-S1 development board power supplies

Caution

Do not fit an Arduino Expansion Shield to the Musca-S1 development board while the Musca-S1 development board is powered up.

Musca-S1 development board and Musca-S1 test chip power rails

The following table shows the maximum loads that the Musca-S1 development board power rails draw from the power supplies.

Table 2-4 Musca-S1 development board and Musca-S1 test chip power rails

Power rail	Voltage	Max load (mA)	Comment
DHUB5V	5V	500 1000	Standard USB 2.0. Some USB 2.0 charging ports permit up to 1000mA. Combined Shield and board current must not exceed the USB current limit.
VBAT	4.2-3.3V	500	Lithium-ion battery, when charging
3V3	3.3V	30	DAPLINK and SoC
		2	I2C sensors (Gyro/Temp/ADC)
		15	QSPI
		900	Shield 3V3 (USB) Combined Shield and board current must not exceed the USB current limit.
		400	Shield 3V3 (battery) Combined Shield and board current must not exceed the battery current limit.
ARD_5V	5V	900	Shield 5V (USB). Combined Shield and board current must not exceed the USB current limit.
		250	Shield 5V (battery). Combined Shield and board current must not exceed the battery current limit.
VDD_CORE	1V0	150	SoC core supply
VDD_IO	1V8	100	SoC I/O supply
VDD_PLL	1V8	5	PLL VDD low power
VDD_XTAL	1V0	1	XTAL VDD

External power

The DAPLink 5V USB connector supplies all external power to the Musca-S1 development board.

Backup battery

A backup battery can power the Musca-S1 development board, using the connector on the lower face of the board.

Arm recommends using the Lithium Ion, CLN 523450, 3.7V, 950mAh battery. The battery is recharged from an external supply during USB 5V operation. If a battery is fitted while external power is connected, circuitry on the board automatically charges the battery with a maximum charging current of 500mA.

Jumper links select the power source, 5V USB or battery, for the board and the Shield.

Note

- Jumper link J18 selects the board power to be either USB or battery.
- Jumper link J19 selects the Shield power to be either USB or battery.

- Jumper link J12 selects the Shield I/O to be either 1V8 or 3V3.
- Jumper links J18 and J19 must be used with each other. The board and Shield must both be set to USB power, or both must be set to battery power.
- See *1.3 The Musca-S1 development board at a glance* on page 1-14 for the location of the jumper links.

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

A.3 USB connector on page Appx-A-180

2.9 I2C interfaces and sensors

To minimize usage of the Musca-S1 test chip pins, the Musca-S1 test chip provides a single I2C interface for the board sensors.

The board sensors provide basic support for *Internet of Things* (IoT) software demonstrations and analog support for the Arduino Expansion Shield.

The analog converter supports programmable I/O that can be ADC, DAC, or GPIO. Two of the I/O are used as GPIO and connect to the DAPLink controller.

The following figure shows the Musca-S1 test chip I2C interface and connected peripherals.

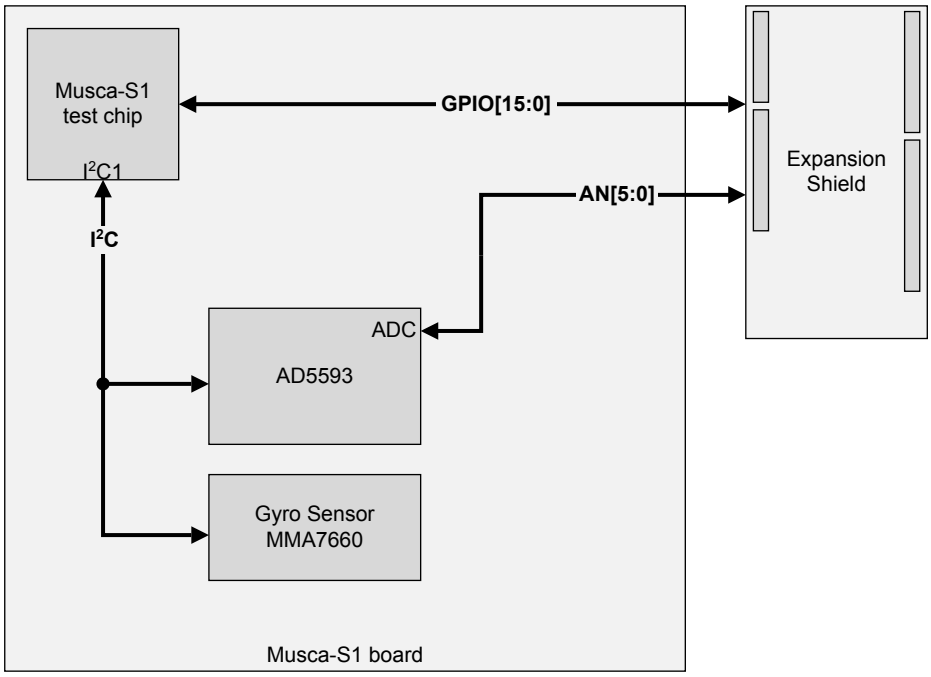

Figure 2-7 I2C interfaces and I2C sensors

Note

The I2C and GPIO[15:0] signals are on multiplexed Musca-S1 test chip I/O pins. The I/O multiplexer must select the correct signals for the required functions to be available. See the following for information on how to select the required functions at the Musca-S1 test chip I/O pins:

- *2.2.2 Test chip multiplexed I/O* on page 2-23.
- *3.11.1 IOMUX registers* on page 3-120.

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

2.10 Arduino Expansion Shield interface

The Musca-S1 development board supports custom system and peripheral design by providing one Arduino Shield interface.

Overview of Arduino Expansion Shield interface

The Arduino Shield interface enables fitting off-the-shelf boards including:

- Sensors.
- Peripherals.
- PHYs.
- Breakout boards for full custom design.

Caution

Do not fit an Arduino Shield while the Musca-S1 development board is powered up.

The following figure shows the Arduino Expansion Shield interface of the Musca-S1 development board.

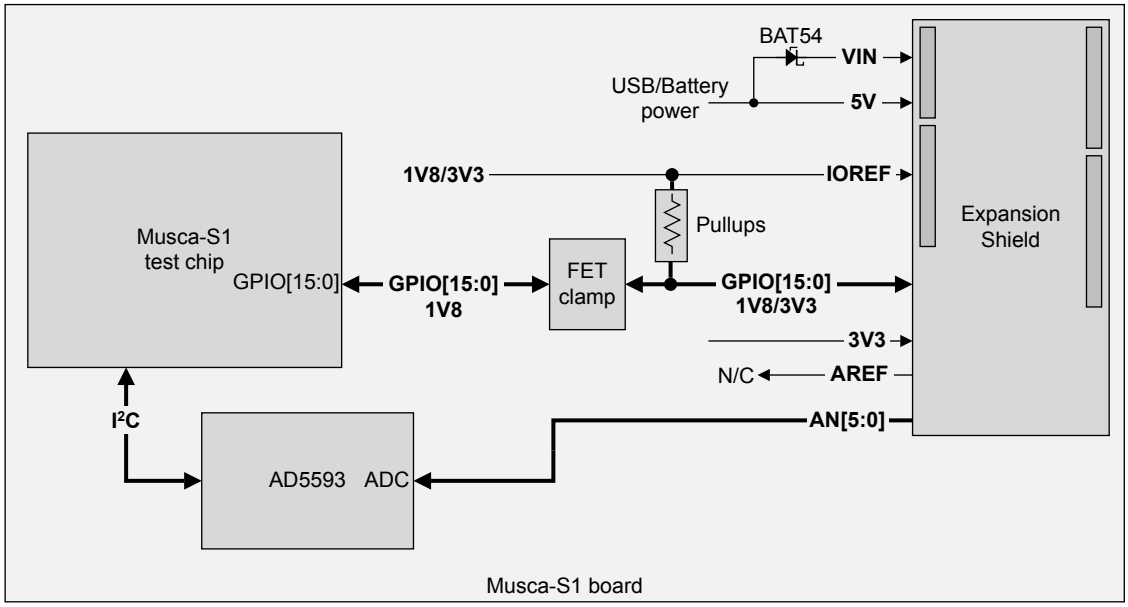

Figure 2-8 Arduino Expansion Shield interface

The Arduino Shield expansion interface provides:

- Up to 16 1V8/3V3 digital I/O.
- I^2C.
- I^2S.
- SPI.
- UART. The UART on the Musca-S1 test chip test chip supports hardware flow control.
- 6-channel 3V3 analog from Expansion Shield to the Musca-S1 development board.
- **IOREF** reference for GPIO level, selectable between 1V8 and 3V3 using jumper link J12.
- **VIN** connects to the 5V source through a reverse protection diode. The 5V power source is selectable between USB or battery using jumper link J19.
- Reset.

See *2.8 Power* on page 2-34 for information on using the jumper links to select board and Shield power sources and selecting the **IOREF** reference voltage.

Caution

The maximum currents available from the Musca-S1 development board for the Arduino Expansion Shield power and reference pins are:

3V3/IOREF Maximum current available is 500mA from USB 2.0, or 900mA from a charging port.

5V Maximum current available is 500mA from USB 2.0, or 900mA from a charging port.

Arduino Expansion Shield multiplexed I/O

The **GPIO[15:0]** signals form part of the multiplexed Musca-S1 test chip I/O. The IOMUX registers control the GPIO multiplexer to select the signals at the Musca-S1 test chip I/O. The registers must select the correct signals for the I^2C, I^2S, SPI, and UART interfaces to be available at the Arduino Expansion Shield.

See the following for information on how to select the required signals at the Musca-S1 test chip I/O pins:

- *2.2.2 Test chip multiplexed I/O* on page 2-23.
- *3.11.1 IOMUX registers* on page 3-120.

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

A.1 Arduino Expansion Shield connectors on page Appx-A-176

2.11 Boot memory

Normal Musca-S1 test chip boot operation is either from internal eMRAM or external QSPI memory.

Boot options

The following boot options are available:

- On-chip eMRAM, 2MB: Non-volatile boot memory.
- External QSPI, 32MB.

Programming boot memory

The following methods of programming boot memory, using the DAPLink controller are available:

- SW debug programming over USB:
  - Programming of the eMRAM or QSPI image is done through the DAPLink SW debug interface which is connected over USB to the host computer. The host computer uses a code development environment such as Arm Keil® µVision® debugger or Arm Development Studio to develop and upload the application code to the Musca-S1 development board using SW debug.
- Drag and Drop:
  - The host computer is connected to the DAPLink interface over USB. The DAPLink firmware enables application code to be dropped into either internal eMRAM or external QSPI memory.
  - Select the appropriate boot option using the boot selector switch on the board. Power cycle the board ON-OFF-ON and drag and drop the new image. The image is automatically programmed to the memory selected by the boot switch.

Note

The QSPI control signals are on the multiplexed Musca-S1 test chip I/O pins. The I/O multiplexer must select the correct signals for QSPI to be available. See the following for information on how to select the required functions at the Musca-S1 test chip I/O pins:

- *2.2.2 Test chip multiplexed I/O* on page 2-23.
- *3.11.1 IOMUX registers* on page 3-120.

2.12 DAPLink controller

The DAPLink controller is an Arm Mbed component that uses a Cortex-M0 processor. The DAPLink controller contains pre-defined firmware that enables access to the CoreSight component in Musca-S1 test chip, *USB Mass Storage Device* (USBMSD), USB UART, and remote reset.

The DAPLink firmware binary image is available at the Arm Community pages which are accessible from *https://www.arm.com/musca*.

Note

The DAPLink controller is only accessible when P-JTAG is disconnected from the debug connector.

2.13 Debug

The Musca-S1 development board provides the following ways of performing debug.

- P-JTAG processor debug available through the debug connector.
- *Serial Wire* or JTAG processor debug (SWJ - DP), available over USB DAPLink:

Related information

1.3 The Musca-S1 development board at a glance on page 1-14

A.2 Debug connector on page Appx-A-179

Chapter 3
Programmers model

This chapter describes the programmers model of the Musca-S1 test chip and board.

It contains the following sections:

3.1 About this programmers model

The following information applies to all registers in this programmers model:

- 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.
  - All register bits are reset to a logic 0 by a system or powerup reset.
  - All register summary tables in this chapter describe register access types as follows:

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

3.2 Memory maps

The memory map in the Musca-S1 test chip is based on the SSE-200 memory map. The SSE-200 memory map alternates between Secure and Non-secure regions every 256MB. Only a few address areas are exempt from security mapping because they are related to debug functionality.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for information on the SSE-200 subsystem memory map.

This section contains the following subsections:

3.2.1 Code (AHB expansion) and SRAM regions memory map

The Musca-S1 test chip memory map implements the code, AHB5 expansion, and SRAM regions of the SSE-200 system memory map.

The following figure shows the Musca-S1 test chip implementation of the code, AHB5 expansion, and SRAM regions of the SSE-200 system memory map.

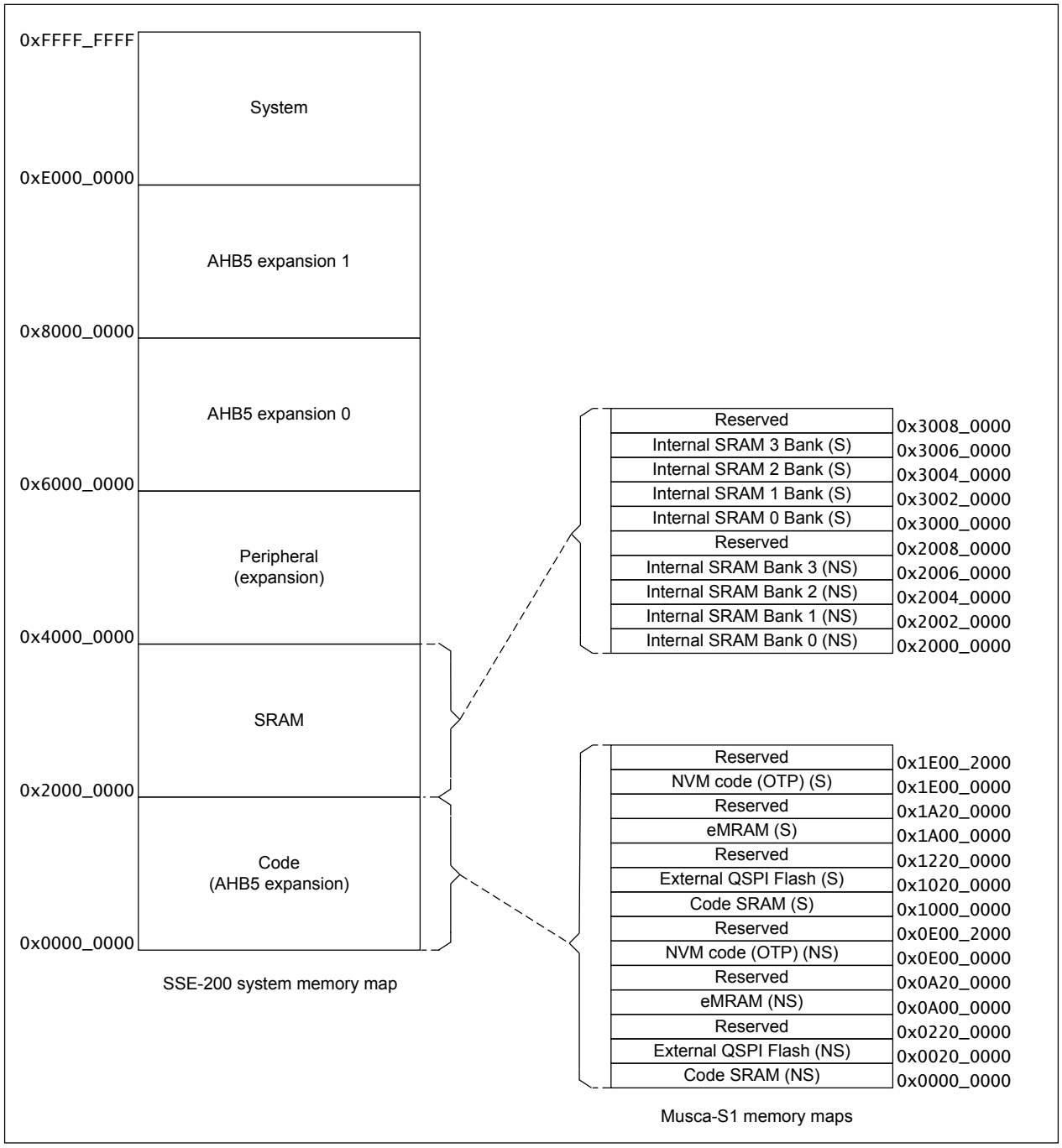

Figure 3-1 Musca-S1 test chip memory map code and SRAM regions

3.2.2 Peripheral (expansion) region memory map

The Musca-S1 test chip implements the Peripheral (expansion) region of the SSE-200 memory map.

The following figure shows the Musca-S1 test chip implementation of the Peripheral (expansion) region of the SSE-200 memory map.

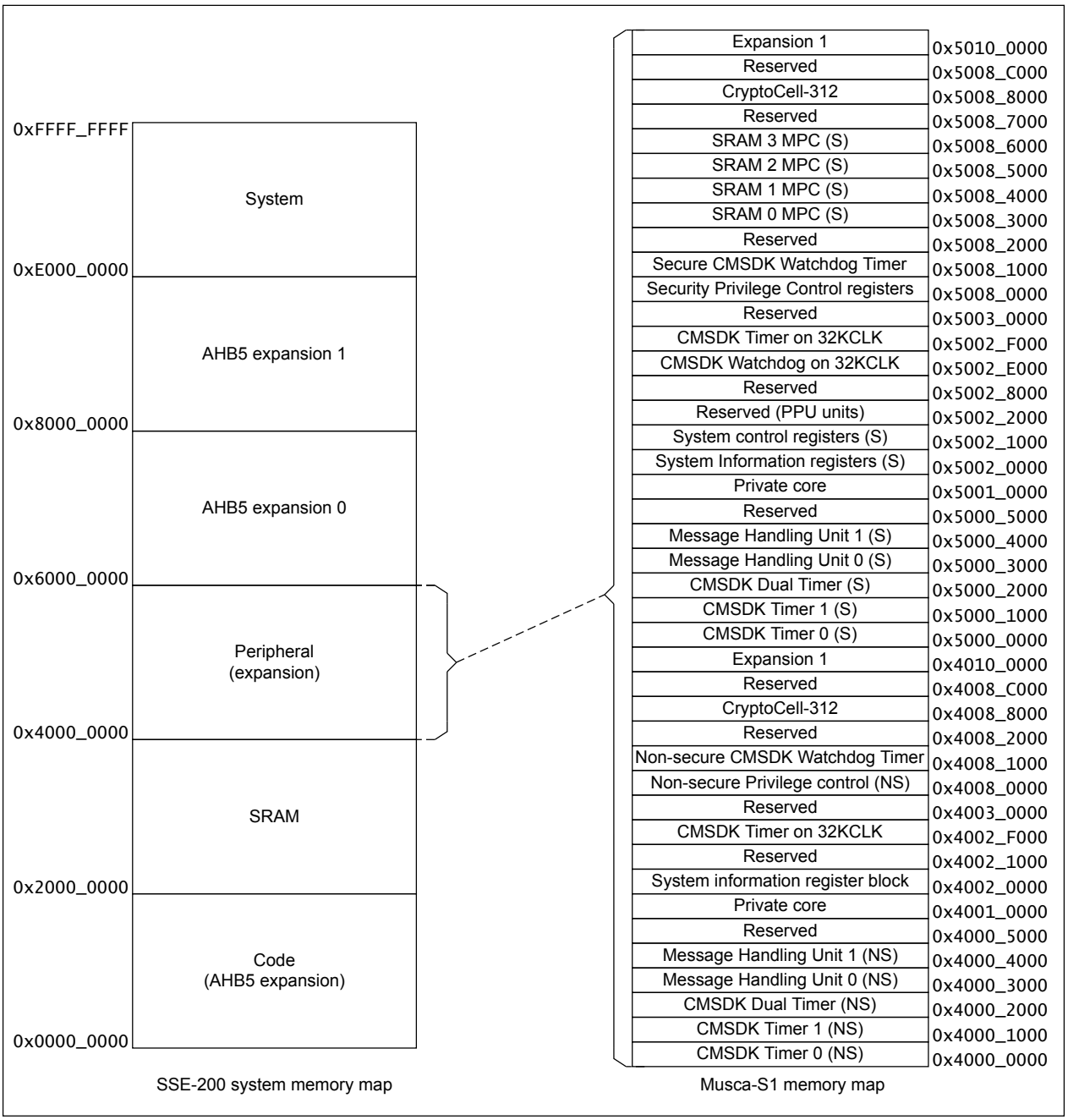

Figure 3-2 Musca-S1 test chip memory map Peripheral region

3.2.3 Non-secure Expansion 1 region memory map

The Musca-S1 test chip implements the Non-secure Expansion 1 region of the SSE-200 memory map.

The following figure shows the Musca-S1 test chip implementation of the Non-secure Expansion 1 region of the SSE-200 memory map.

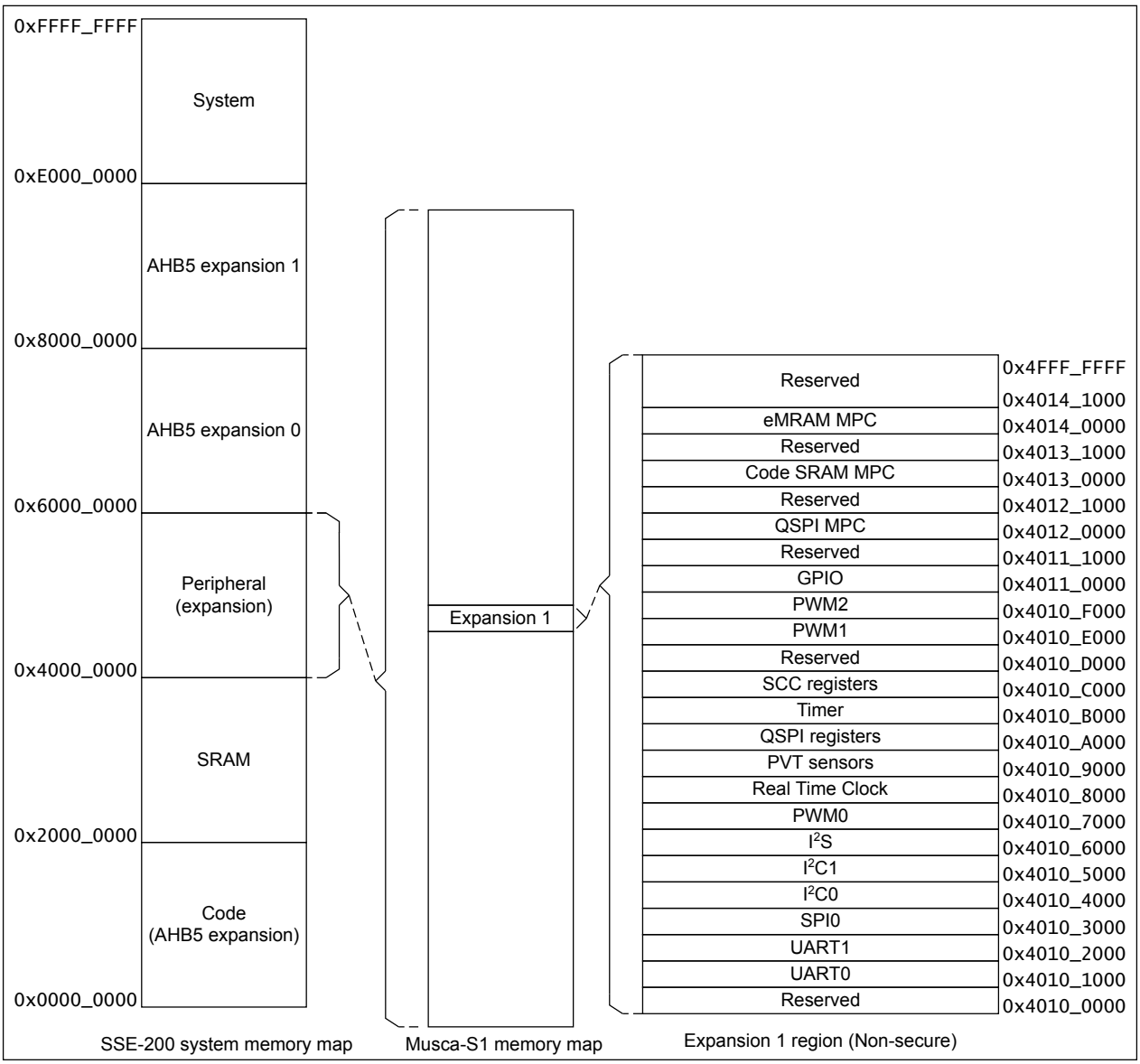

Figure 3-3 Musca-S1 test chip memory map Non-secure Expansion 1 region

3.2.4 Secure Expansion 1 region memory map

The Musca-S1 test chip implements the Secure Expansion 1 region of the SSE-200 memory map.

The following figure shows the Musca-S1 test chip implementation of the Secure Expansion 1 region of the SSE-200 memory map.

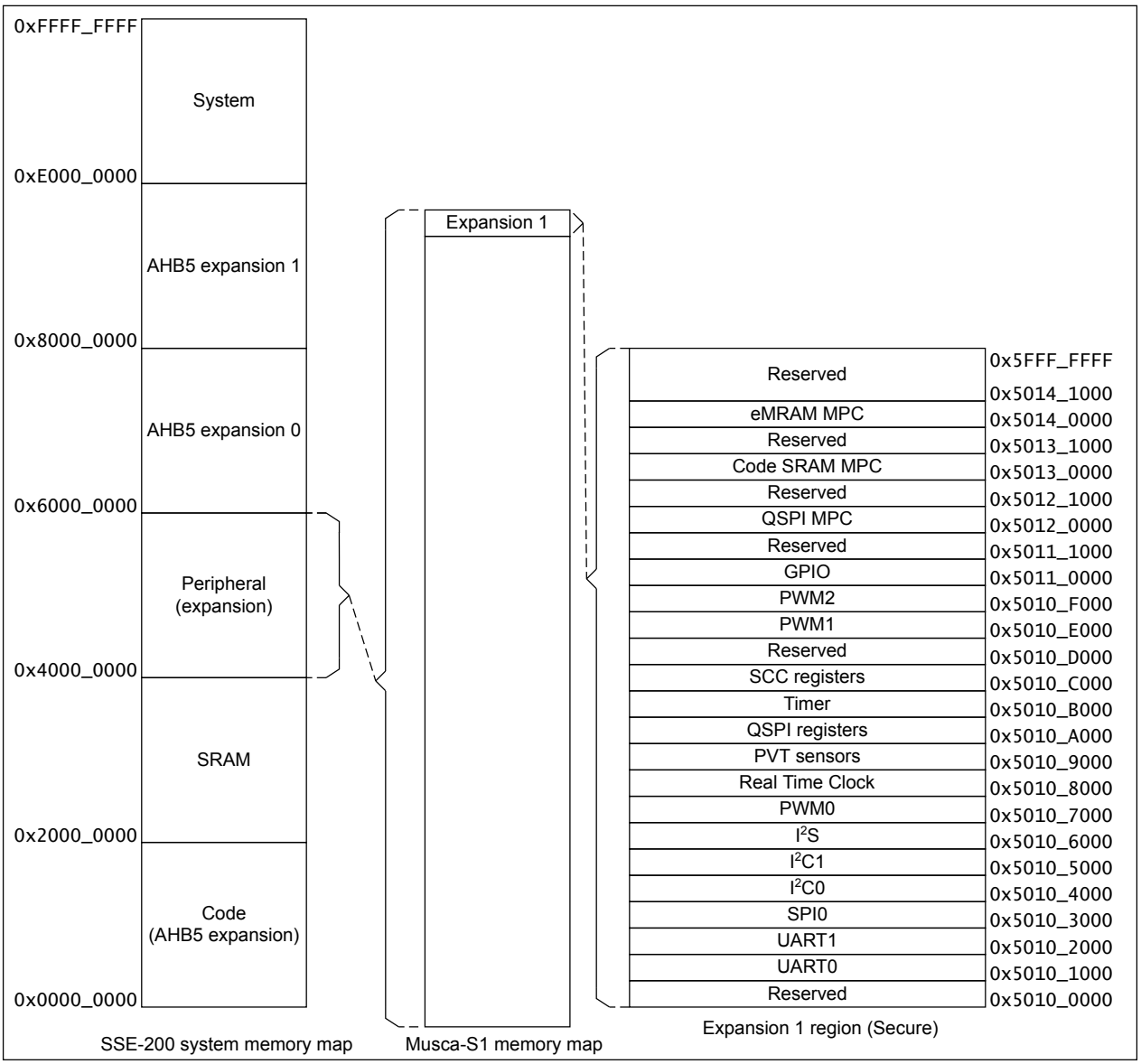

Figure 3-4 Musca-S1 test chip memory map Secure Expansion 1 region

3.2.5 System region memory map

The Musca-S1 test chip implements the System region of the SSE-200 memory map.

The following figure shows the Musca-S1 test chip implementation of the System region of the SSE-200 memory map.

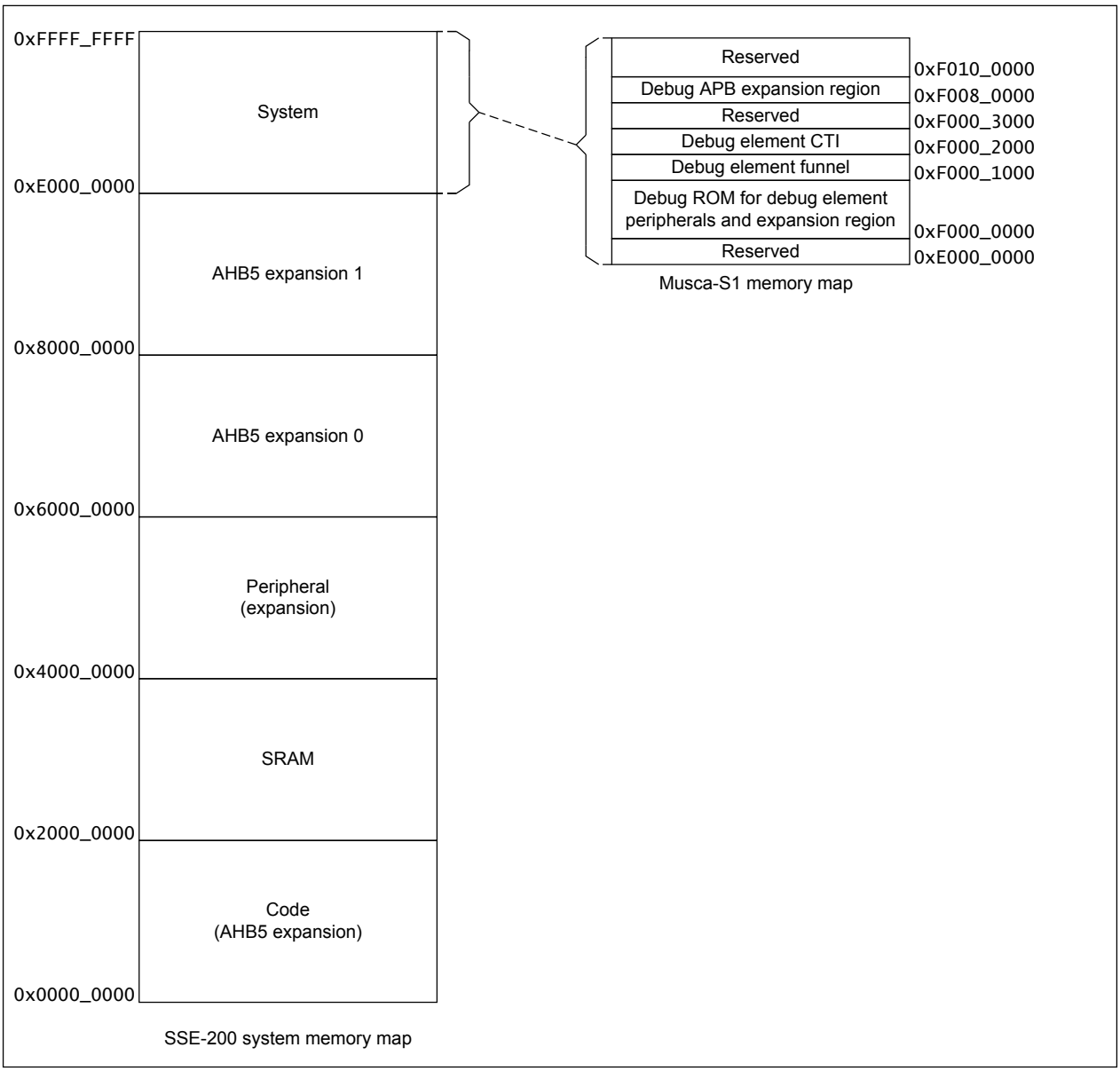

Figure 3-5 Musca-S1 test chip memory map System region

3.2.6 Complete memory map

The following table shows the complete Musca-S1 test chip memory map. Undefined memory locations are reserved and software must not attempt to access these locations.

Table 3-1 Memory map

Non-secure		Secure		Description		
From	To	From	To	Size	Non-secure	Secure
0x0000_0000	0x001F_FFFF	0x1000_0000	0x101F_FFFF	2MB	Code SRAM	Code SRAM
0x0020_0000	0x021F_FFFF	0x1020_0000	0x121F_FFFF	32MB	External QSPI Flash	External QSPI Flash
0x0A00_0000	0x0A1F_FFFF	0x1A00_0000	0x1A1F_FFFF	2MB	eMRAM	eMRAM
0x0E00_0000	0x0E00_0000	0x1E00_0000	0x1E00_0000	128KB	Emulated *One Time Programming* (OTP) secure memory	Emulated *One Time Programming* (OTP) secure memory
0x2000_0000	0x2001_FFFF	0x3000_0000	0x3001_FFFF	128KB	Internal SRAM bank 0	Internal SRAM bank 0
0x2002_0000	0x2003_FFFF	0x3002_0000	0x3003_FFFF	128KB	Internal SRAM bank 1	Internal SRAM bank 1
0x2004_0000	0x2005_FFFF	0x3004_0000	0x3005_FFFF	128KB	Internal SRAM bank 2	Internal SRAM bank 2
0x2006_0000	0x2007_FFFF	0x3006_0000	0x3007_FFFF	8KB	Internal SRAM bank 3	Internal SRAM bank 3
0x4000_0000	0x4000_0FFF	0x5000_0000	0x5000_0FFF	4KB	CMSDK Timer 0	CMSDK Timer 0
0x4000_1000	0x4000_1FFF	0x5000_1000	0x5000_1FFF	4KB	CMSDK Timer 1	CMSDK Timer 1
0x4000_2000	0x4000_2FFF	0x5000_2000	0x5000_2FFF	4KB	CMSDK Dual Timer	CMSDK Dual Timer
0x4000_3000	0x4000_3FFF	0x5000_3000	0x5000_3FFF	4KB	Message Handling Unit 0	Message Handling Unit 0
0x4000_4000	0x4000_4FFF	0x5000_4000	0x5000_4FFF	4KB	Message Handling Unit 1	Message Handling Unit 1
0x4001_0000	0x4001_FFFF	0x5001_0000	0x5001_FFFF	64KB	Private core	Private core
0x4002_0000	0x4002_0FFF	0x5002_0000	0x5002_0FFF	4KB	System information registers	System information registers
-	-	0x5002_1000	0x5002_0FFF	4KB	-	System control registers
-	-	0x5002_E000	0x5002_EFFF	4KB	-	CMSDK Watchdog on 32K
0x4002_F000	0x4002_FFFF	0x5002_F000	0x5002_FFFF	4KB	CMSDK Timer on 32K	CMSDK Timer on 32K
0x4008_0000	0x4008_0FFF	0x5008_0000	0x5008_0FFF	4KB	Non-secure privilege control	Secure privilege control
0x4008_1000	0x4008_1FFF	0x5008_1000	0x5008_1FFF	4KB	Non-secure CMSDK Watchdog Timer	Secure CMSDK Watchdog Timer
-	-	0x5008_3000	0x5008_3FFF	4KB	-	SRAM 0 Memory Protection Controller

Table 3-1 Memory map (continued)

Non-secure		Secure		Description		
From	To	From	To	Size	Non-secure	Secure
-	-	0x5008_4000	0x5008_4FFF	4KB	-	SRAM 1 Memory Protection Controller
-	-	0x5008_5000	0x5008_5FFF	4KB	-	SRAM 2 Memory Protection Controller
-	-	0x5008_6000	0x5008_6FFF	4KB	-	SRAM 3 Memory Protection Controller
0x4008_8000	0x4008_BFFF	0x5008_8000	0x5008_BFFF	16KB	CryptoCell-312 (r1p0)	CryptoCell-312 (r1p0)
0x4010_1000	0x4010_1FFF	0x5010_1000	0x5010_1FFF	4KB	UART0	UART0
0x4010_2000	0x4010_2FFF	0x5010_2000	0x5010_2FFF	4KB	UART1	UART1
0x4010_3000	0x4010_3FFF	0x5010_3000	0x5010_3FFF	4KB	SPI0	SPI0
0x4010_4000	0x4010_4FFF	0x5010_4000	0x5010_4FFF	4KB	I^2C0	I^2C0
0x4010_5000	0x4010_5FFF	0x5010_5000	0x5010_5FFF	4KB	I^2C1	I^2C1
0x4010_6000	0x4010_6FFF	0x5010_6000	0x5010_6FFF	4KB	I^2S	I^2S
0x4010_7000	0x4010_7FFF	0x5010_7000	0x5010_7FFF	4KB	PWM0	PWM0
0x4010_8000	0x4010_8FFF	0x5010_8000	0x5010_8FFF	4KB	Real Time Clock	Real Time Clock
0x4010_9000	0x4010_9FFF	0x5010_9000	0x5010_9FFF	4KB	PVT sensors	PVT sensors
0x4010_A000	0x4010_AFFF	0x5010_A000	0x5010_AFFF	4KB	QSPI registers	QSPI registers
0x4010_B000	0x4010_BFFF	0x5010_B000	0x5010_BFFF	4KB	General -purpose timer	General -purpose timer
0x4010_C000	0x4010_CFFF	0x5010_C000	0x5010_CFFF	4KB	SCC registers	SCC registers
0x4010_E000	0x4010_EFFF	0x5010_E000	0x5010_EFFF	4KB	PWM1	PWM1
0x4010_F000	0x4010_FFFF	0x5010_F000	0x5010_FFFF	4KB	PWM2	PWM2
0x4011_0000	0x4011_0FFF	0x5011_0000	0x5011_0FFF	4KB	GPIO	GPIO
0x4012_0000	0x4012_0FFF	0x5012_0000	0x5012_0FFF	4KB	QSPI MPC	QSPI MPC
0x4013_0000	0x4013_0FFF	0x5013_0000	0x5013_0FFF	4KB	Code SRAM *Memory Protection Controller* (MPC)	Code SRAM *Memory Protection Controller* (MPC)
0x4014_0000	0x4014_0FFF	0x5014_0000	0x5014_0FFF	4KB	eMRAM MPC	eMRAM MPC
-	-	0xF000_0000	0xF000_0FFF	4KB	-	Debug system ROM
-	-	0xF000_1000	0xF000_1FFF	4KB	-	Debug element funnel

Table 3-1 Memory map (continued)

Non-secure		Secure		Description		
From	**To**	**From**	**To**	**Size**	**Non-secure**	**Secure**
-	-	0xF000_2000	0xF000_2FFF	4KB	-	Debug element Cross Trigger Interface (CTI)
-	-	0xF008_0000	0xF00F_FFFF	512KB	-	Debug APB expansion region.

3.3 Processor elements

The SSE-200 subsystem in the Musca-S1 test chip implements two processor elements. Each element contains a Cortex-M33 (r0p2) core.

Processor 0, CPU0, is the main processor. It has FPU and DSP, and no coprocessor. The operating frequency is 50MHz.

Processor 1, CPU1, is the secondary processor. It has FPU and DSP, and no coprocessor. The operating clock frequency is 200MHz.

This section contains the following subsections:

3.3.1 Private processor regions

Both processor elements in the system implement a private memory region that only it can see.

The base memory addresses of the private processor regions are:

- `0x4001_0000` in the Non-secure region.
- `0x5001_0000` in the Secure region.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information on the private processor regions.

3.3.2 Instruction cache configuration interface registers

The following table shows the instruction cache configuration interface registers. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-2 Instruction cache configuration interface registers

Offset	Name	Type	Reset	Width	Description
`0x0000`	ICHWPARAMS	RO	`0x0000_0000`	32	Hardware Parameter Register
`0x0004`	ICCTRL	RW	`0x0000_0000`	32	Instruction cache Control Register
`0x0100`	ICIRQSTAT	RO	`0x0000_0000`	32	Interrupt Request Status Register
`0x0104`	ICHRQSCLR	WO	`0x0000_0000`	32	Interrupt Status Clear Register
`0x0108`	ICIRQEN	RW	`0x0000_0000`	32	Interrupt Enable Register
`0x010C`	ICDBGFILLERR	RO	`0x0000_0000`	32	Debug Fill Error Register
`0x0300`	ICSHR	RO	`0x0000_0000`	32	Instruction cache Statistic Hit Register
`0x0304`	ICSMR	RO	`0x0000_0000`	32	Instruction cache Statistic Miscount Register
`0x0308`	ICSUC	RO	`0x0000_0000`	32	Instruction cache Statistic Uncached Count Register
`0x0FD0`	PIDR4	RO	`0x0000_0004`	32	Product ID Register 4
`0x0FE4`	PIDR1	RO	`0x0000_00B8`	32	Product ID Register 1

Table 3-2 Instruction cache configuration interface registers (continued)

Offset	Name	Type	Reset	Width	Description
0x0FE8	PIDR2	RO	0x0000_000B	32	Product ID Register 2
0x0FEC	PIDR3	RO	0x0000_0000	32	Product ID Register 3
0x0FF0	CIDR0	RO	0x0000_000D	32	Component ID Register 0
0x0FF4	CIDR1	RO	0x0000_00F0	32	Component ID Register 1
0x0FF8	CIDR2	RO	0x0000_0005	32	Component ID Register 2
0x0FFC	CIDR3	RO	0x0000_00B1	32	Component ID Register 3

Note

- All instruction cache configuration interface registers are Secure Privilege access only.
- See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information about the instruction cache configuration interface registers.

3.3.3 Processor cache programming

The following practices and techniques are recommended when programming the L1 cache in the Musca-S1 test chip:

Initialization

After powerup or reset, the cache powers up in a disabled state and begins the invalidation process. Accesses arriving at the cache are not cached and bypass the cache. The cache can be enabled during the invalidation process by setting the CACHEEN control bit in the *Instruction Cache Control Register*, ICCTRL, to `0b1`. However, all accesses are still treated as uncached and bypass the cache until the cache invalidation process completes.

At the end of the cache invalidation process, the interrupt status signal, **IC**, in the *Interrupt Request Status Register*, ICIRQSTAT, is asserted. If that interrupt is already enabled or is enabled later, an interrupt is raised. To enable caching of code fetches, you can poll this status register, or wait for this interrupt to be raised before continuing code execution.

Cache disable

The cache can be disabled by clearing the CACHEEN control bit in the ICCTRL. Outstanding accesses are completed before the cache is disabled. Software can read the CDC bit in ICIRQSTAT 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 by setting the partial invalidate bit, PINV, or the full invalidate bit, FINV, in the ICCTRL Register. Because the cache does not support Locked Lines, setting either of these bits initiates a full cache invalidation. During cache invalidation, all accesses through the cache are treated as uncached and bypass the cache until the invalidation process completes. At the end of the invalidation process, the interrupt status, IC, is asserted. If that interrupt is already enabled, or is enabled later, an interrupt is raised.

Performance targets

The cache improves the average performance of the connected processor by holding local copies of previously accessed or specified memory locations. The improvement in performance cannot be determined precisely because many design parameters, code behavior, and system considerations affect the performance.

The cache can reduce processor performance in the following events:

- Uncacheable memory. The cache adds a cycle of latency to the transaction.
- Writes are treated as uncacheable, and create an extra cycle of bus latency.
- 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 time that is taken for the memory subsystem to return the rest of the WRAP4 transaction determines the extra latency.

3.3.4 Ensuring the cache handles memory modifications

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

The software must invalidate the cache to modify the external location.

——— Caution ———

Non-coherency between cached lines in the cache and the lines in the external code memory is a security issue.

———————————

To invalidate the cache, do the following:

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

Cache misses occur when a modification to the Secure Access Unit, or the Memory Protection Controller, changes the security setting of a recently cached memory region. The instruction cache retains the old security attribute and disables hits on the cached line using the new security attribute. This situation can result in Secure and Non-secure versions of the same memory location residing in the cache, reducing its efficiency. If the older cached line, not intended to be available in that system, is accessed with the original access attribute, the presence of the two versions poses a security risk.

3.3.5 Interrupts

The Musca-S1 test chip implements an Arm *Nested Vector Interrupt Controller* (NVIC) and an Arm *Wakeup Interrupt Controller* (WIC).

See the following documentation for more information on the interrupt controller.

- *Arm® Cortex®-M33 Processor Technical Reference Manual (r0p2)*.
- *Arm® v7-M Architecture Reference Manual*.

Nested Vector Interrupt Controller (NVIC) features

The NVIC in the Musca-S1 test chip supports the following features:

- 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).

Interrupts from the SSE-200 subsystem

The following table shows the interrupt signals and exceptions to the two processor cores from blocks in the SSE-200 subsystem.

Table 3-3 SSE-200 interrupt signals

Interrupt input	CPU0 and CPU1 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]	MHU0 CPU0 Interrupt: MHU0 CPU1 Interrupt
IRQ[7]	MHU1 CPU0 Interrupt: MHU1 CPU1 Interrupt
IRQ[8]	Reserved
IRQ[9]	MPC Combined (Secure)
IRQ[10]	PPC Combined (Secure)
IRQ[11]	MSC Combined (Secure)
IRQ[12	Bridge error combined interrupt (Secure)
IRQ[13]	CPU0 instruction cache invalidation interrupt
IRQ[14]	Reserved
IRQ[15]	SYS_PPU
IRQ[16]	CPU0_PPU
IRQ[17]	CPU1_PPU
IRQ[18]	CPU0DBG_PPU
IRQ[19]	CPU1DBG_PPU
IRQ[20]	Reserved
IRQ[21]	Reserved
IRQ[22]	RAM0_PPU
IRQ[23]	RAM1_PPU
IRQ[24]	RAM2_PPU
IRQ[25]	RAM3_PPU
IRQ[26]	DBG_PPU
IRQ[27]	Reserved
IRQ[28]	CPU0CTIIRQ0, CPU1CTIIRQ0
IRQ[29]	CPU0CTIIRQ1, CPU1CTIIRQ1
IRQ[31:30]	Reserved

Interrupts from outside the SSE-200 subsystem

The following table shows the expansion interrupt signals, that is, from Musca-S1 test chip blocks outside the SSE-200 subsystem.

Table 3-4 Expansion interrupt signals from blocks outside the SSE-200

Interrupt input	CPU0 and CPU1 interrupt source	Wake up	Description
IRQ[32]	Reserved	-	Reserved
IRQ[33]	GPTIMERINTR	Yes	General-purpose timer combined interrupt.
IRQ[34]	I2C0INTR	-	I^2C0 interrupt
IRQ[35]	I2C1INTR	-	I^2C1 interrupt
IRQ[36]	I2SINTR	-	I^2S interrupt
IRQ[37]	SPIINTR	-	SPI interrupt
IRQ[38]	QSPIINTR	-	QSPI interrupt
IRQ[39]	UARTRXINTR0	-	UART0 receive FIFO interrupt, active-HIGH.
IRQ[40]	UARTTXINTR0	-	UART0 transmit FIFO interrupt, active-HIGH.
IRQ[41]	UARTRTINTR0	-	UART0 receive timeout interrupt, active-HIGH.
IRQ[42]	UARTMSINTR0	-	UART0 modem status interrupt, active-HIGH.
IRQ[43]	UARTEINTR0	-	UART0 error interrupt, active-HIGH.
IRQ[44]	UARTINTR0	-	UART0 interrupt, active-HIGH.
IRQ[45	UARTRXINTR1	-	UART1 receive FIFO interrupt, active-HIGH.
IRQ[46]	UARTTXINTR1	-	UART1 transmit FIFO interrupt, active-HIGH.
IRQ[47]	UARTRTINTR1	-	UART1 receive timeout interrupt, active-HIGH.
IRQ[48]	UARTMSINTR1	-	UART1 modem status interrupt, active-HIGH.
IRQ[49]	UARTEINTR1	-	UART1 error interrupt, active-HIGH.
IRQ[50]	UARTINTR1	-	UART1 interrupt, active-HIGH.
IRQ[66:51]	GPIOINT[15:0]	-	GPIO interrupts.
IRQ[67]	COMBINT	-	GPIO combined interrupt.
IRQ[68]	PVTINTR	-	PVT sensor interrupt.
IRQ[69]	-	-	Reserved.
IRQ[70]	PWMINT0	-	PWM0 interrupt.
IRQ[71]	RTCINT	-	RTC interrupt.
IRQ[72]	GPTIMERINT1	Yes	General-purpose timer interrupt[1] (Comparator 1).
IRQ[73]	GPTIMERINT0	Yes	General-purpose timer interrupt[0] (Comparator 0).
IRQ[74]	PWMINT1	-	PWM1 interrupt.
IRQ[75]	PWMINT2	-	PWM2 interrupt.
IRQ[76]	GPIO_COMB_NONSEC_INTR	-	GPIO Non-secure interrupt.
IRQ[95:77]	-	-	Reserved

Interrupt controller registers

The following table shows the Musca-S1 test chip interrupt controller registers. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-5 Summary of interrupt controller registers

Address	Name	Type	Reset value	Description
0xE000E004	ICTR	RO	-	Interrupt Controller Type Register
0xE000_E100-0xE000_E11C	NVIC_ISER0-NVIC_ISER7	RW	0x0000_0000	Interrupt Set Enable Registers
0xE000_E180-0xE000_E19C	NVIC_ICER0-NVIC_ICER7	RW	0x0000_0000	Interrupt Clear Enable Registers
0xE000_E200-0xE000_E21C	NVIC_ISPR0-NVIC_ISPR7	RW	0x0000_0000	Interrupt Set Pending Registers
0xE000_E280-0xE000_E29C	NVIC_ICPR0-NVIC_ICPR7	RW	0x0000_0000	Interrupt Clear Pending Registers
0xE000_E300-0xE000_E31C	NVIC_IABR0-NVIC_IABR7	RO	0x0000_0000	Interrupt Active Bit Registers
0xE000_E400-0xE000_E41F	NVIC_IPRO-NVIC_IPR7	RW	0x0000_0000	Interrupt Priority Registers

See the following documents for more information on the interrupt controller:

- *Arm® Cortex®-M33 Processor Technical Reference Manual (r0p2)*.
- *Arm® v7-M Architecture Reference Manual*.

Processor core Interrupt Registers

The SSE-200 block implements CPU0 and CPU1 core Interrupt Registers. The Interrupt Registers enable software to raise interrupts, clear interrupts, and check the written value that raises the interrupts to the cores.

Set and Clear registers support setting and clearing of individual bits which means the individual bits can represent events that can be independently set and cleared.

The CPU0 and CPU1 Interrupt Registers are:

- CPU0INTR_STAT - Core 0 Interrupt Status Register.
- CPU0INTR_SET - Core 0 Interrupt Set Register.
- CPU0INTR_CLR - Core 0 Interrupt Clear Register.
- CPU1INTR_STAT - Core 1 Interrupt Status Register.
- CPU1INTR_SET - Core 1 Interrupt Set Register.
- CPU1INTR_CLR - Core 1 Interrupt Clear Register.

See the following for more information on the CPU0 and CPU1 Interrupt Registers.

- *3.4.8 Message Handling Unit* on page 3-89.
- *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)*.

Wakeup Interrupt Controller (WIC)

The WIC is a peripheral that detects an interrupt signal and wakes the processor from deep-sleep mode. The WIC is active only when the system is in deep-sleep mode.

The WIC is not programmable and does not have registers or a user interface. It operates entirely under the control of hardware signals.

When the WIC is enabled and the processor is in deep-sleep mode, the *Power Management Unit* (PMU) can power down most of the processor. When the WIC receives an interrupt, it takes several clock cycles

to wake up the processor to a state where it can service the interrupt. Latency is increased in deep-sleep mode.

Note

The IoT System uses latches to implement the WIC, unlike in the standard Cortex-M33 processor. **FCLK** can be gated completely during WIC-based deep-sleep. This complete gating is not a standard Cortex-M33 processor feature.

See the *Arm® Cortex®-M33 Processor Technical Reference Manual (r0p2)* for more information on the WIC.

3.4 Base element

This section describes control registers that are associated with several base element components of the Musca-S1 test chip.

This section contains the following subsections:

3.4.1 Internal SRAM regions

The base element contains four internal SRAM regions of the same size that form a contiguous area of memory. The SRAMs are mapped to both the Secure and Non-secure regions of memory.

A *Memory Protection Controller* (MPC) determines how the memory locations with internal SRAM are mapped to the Secure and Non-secure regions.

See *3.2.2 Peripheral (expansion) region memory map* on page 3-47 and *3.2.6 Complete memory map* on page 3-51.

3.4.2 Base peripheral regions

The base peripheral regions are where the peripherals of the base element reside. There are four regions, two Secure and two Non-secure.

See *3.2.2 Peripheral (expansion) region memory map* on page 3-47 and *3.2.6 Complete memory map* on page 3-51. The base peripheral regions are:

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

Some peripherals are aliased to both the Secure and the Non-secure regions. The Peripheral Protection Controllers determine the final mapping to both the Secure and Non-secure regions and Privileged or Non-Privileged access support.

3.4.3 CMSDK timers

The base element of the Musca-S1 test chip contains two CMSDK timers and associated control registers.

TIMER 0 registers are at the following base memory addresses:

- `0x4000_0000` in the Non-secure region.
- `0x5000_0000` in the Secure region.

TIMER 1 registers are at the following base memory addresses:

- `0x4000_1000` in the Non-secure region.
- `0x5000_1000` in the Secure region.

Cross Trigger Interface (CTI) triggers from the debug subsystem can halt the timers.

Note

The EXTIN input of the timers is connected to the CTI debug halt logic, and if there is a debug halt access it is used to stop the timer counter logic.

To enable this functionality, the EXTIN must be enabled by writing to the CTRL Register.

- CTRL bit[2] = `0b1`.
- CTRL bit[1] = `0b0`.

The timers reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**.

The following table shows the CMSDK timer control registers in the base element in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

See the *Arm® Cortex®-M System Design Kit Technical Reference Manual* for full descriptions of the registers.

Table 3-6 CMSDK timer control registers summary

Offset	Name	Type	Reset	Width	Function
`0x0000`	CTRL	RW	`0x0000_0000`	32	Bit[3]: Interrupt enable. Bit[2]: Select external input as clock. Bit[1]: Select external input as enable. Bit[0]: Enable.
`0x0004`	VALUE	RW	`0x0000_0000`	32	Current value.
`0x0008`	RELOAD	RW	`0x0000_0020`	32	Reload value. A write to this register sets the current value.
`0x000C`	INSTATUS INTCLEAR	RW	`0x0000_0020`	32	Timer interrupt. Write `0x1` to clear.
`0x0FDO`	PID4	RO	`0x0000_0004`	32	Peripheral ID Register 4
`0x0FD4`	PID5	RO	`0x0000_0000`	32	Peripheral ID Register 5
`0x0FD8`	PID6	RO	`0x0000_0000`	32	Peripheral ID Register 6
`0x0FDC`	PID7	RO	`0x0000_0000`	32	Peripheral ID Register 7
`0x0FE0`	PID0	RO	`0x0000_0022`	32	Peripheral ID Register 0. Bits [7:0] Part number [7:0].
`0x0FE4`	PID1	RO	`0x0000_00B8`	32	Peripheral ID Register 1: Bits [7:4] jep106_id_3_0. Bits [3:0] Part number [11:8].

Table 3-6 CMSDK timer control registers summary (continued)

Offset	Name	Type	Reset	Width	Function
0x0FE8	PID2	RO	0x0000_000B	32	Peripheral ID Register 2: Bits [7:4] Revision. Bits [3] jedec_used. Bits [2:0] jep106_id_6_4.
0x0FEC	PID3	RO	0x0000_0000	32	Peripheral ID Register 3: Bits [7:4] ECO revision number. Bits [3:0] Customer modification number.
0x0FFO	CID0	RO	0x0000_000D	32	Component ID Register 0
0x0FF4	CID1	RO	0x0000_00F0	32	Component ID Register 1
0x0FF8	CID2	RO	0x0000_0005	32	Component ID Register 2
0x0FFC	CID3	RO	0x0000_00B1	32	Component ID Register 3

3.4.4 CMSDK dual timer

The base element of the Musca-S1 test chip contains a CMSDK dual timer and associated control registers.

The timers can be halted by CTI triggers from the debug subsystem.

The dual timer resides in the PD_SYS power domain and is reset by **nWARMRESETSYS**.

The base memory addresses of the CMSDK dual timer are:
- 0x4000_2000 in the Non-secure region.
- 0x5000_2000 in the Secure region.

The following table shows the dual timer control registers in the Musca-S1 test chip in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

See *Arm® Cortex®-M System Design Kit Technical Reference Manual* for full descriptions of the registers.

Table 3-7 CMSDK dual timer control registers summary

Offset	Name	Type	Reset	Width	Function
0x0000	DTIMER1LOAD	RW	0x0000_0000	32	Dual timer 1 load register.
0x0004	DTIMER1VALUE	RO	0xFFFF_FFFF	32	Dual timer 1 current value register.
0x0008	DTIMER1CONTROL	RW	0x0000_0020	32	Dual timer 1 control register. Bits [31:8] are reserved.
0x000C	DTIMER1INTCLR	WO	-	32	Dual timer 1 interrupt clear register.
0x0010	DTIMER1RIS	RO	0x0000_0000	32	Dual timer 1 raw interrupt status register. Bits [31:1] are reserved.

Table 3-7 CMSDK dual timer control registers summary (continued)

Offset	Name	Type	Reset	Width	Function
0x0014	DTIMER1MIS	RO	0x0000_0000	32	Dual timer 1 interrupt status register. Bits [31:1] are reserved.
0x0018	DTIMER1BGLOAD	RW	0x0000_0000	32	Dual timer 1 background load register.
0x0020	DTIMER2LOAD	RW	0x0000_0000	32	Dual timer 2 load register.
0x0024	DTIMER2VALUE	RO	0xFFFF_FFFF	32	Dual timer 2 current value register.
0x0028	DTIMER2CONTROL	RW	0x0000_0020	32	Dual timer 2 control register. Bits [31:8] are reserved.
0x002C	DTIMER2INTCLR	WO	-	32	Dual timer 2 interrupt clear register.
0x0030	DTIMER2RIS	RO	0x0000_0000	32	Dual timer 2 raw interrupt status register. Bits [31:1] are reserved.
0x0034	DTIMER2MIS	RO	0x0000_0000	32	Dual timer 2 interrupt status register. Bits [31:1] are reserved.
0x0038	DTIMER2BGLOAD	RW	0x0000_0000	32	Dual timer 2 background load register.
0x0F00	DTIMERITCR	RW	0x0000_0000	32	Integration test control register.
0x0F04	DTIMERITOP	WO	0x0000_0000	32	Integration test output set register. Bits [31:2] are reserved.
0x0FD0	DTIMERPERIPHID4	RO	0x0000_0004	32	Peripheral ID Register 4. Bits [31:8] are reserved.
0x0FE0	DTIMERPERIPHID0	RO	0x0000_0023	32	Peripheral ID Register 0. Bits [31:8] are reserved.
0x0FE4	DTIMERPERIPHID1	RO	0x0000_00B8	32	Peripheral ID Register 1. Bits [31:8] are reserved.
0x0FE8	DTIMERPERIPHID2	RO	0x0000_000B	32	Peripheral ID Register 2. Bits [31:8] are reserved.
0x0FEC	DTIMERPERIPHID3	RO	0x0000_0000	32	Peripheral ID Register 3. Bits [31:8] are reserved.
0x0FF0	DTIMERPCELLID0	RO	0x0000_000D	32	Component ID Register 0. Bits [31:8] are reserved.
0x0FF4	DTIMERPCELLID1	RO	0x0000_00F0	32	Component ID Register 1. Bits [31:8] are reserved.

Table 3-7 CMSDK dual timer control registers summary (continued)

Offset	Name	Type	Reset	Width	Function
0x0FF8	DTIMERPCELLID2	RO	0x0000_0005	32	Component ID Register 2. Bits [31:8] are reserved.
0x0FFC	DTIMERPCELLID3	RO	0x0000_00B1	32	Component ID Register 3. Bits [31:8] are reserved.

3.4.5 CMSDK watchdog timers

The base element of the Musca-S1 test chip contains two CMSDK watchdog timers.

The base memory addresses of the two CMSDK watchdog timers are:

- 0x4008_1000 in the Non-secure region.
- 0x5008_1000 in the Secure region.

Each watchdog is permanently mapped to either a Secure or a Non-secure region of address space:
- The Secure watchdog can raise a *non-maskable interrupt* (NMI) to both processor cores. However, in this case, a watchdog reset event resets the entire system.
- The Non-secure watchdog can raise an interrupt to both processor 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.

CTI triggers from the debug subsystem can halt the watchdog timers.

The timers reside in the PD_SYS power domain and are reset by **nWARMRESETSYS**.

The following table shows the watchdog timer control registers in the base element in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

See *Arm® Cortex®-M System Design Kit Technical Reference Manual* for full descriptions of the registers.

Table 3-8 CMSDK watchdog timers control registers summary

Offset	Name	Type	Reset	Width	Function
0x0000	WDOGLOAD	RW	0xFFFF_FFFF	32	Watchdog load register. The counter decrements from the value in this register. The count restarts from this value immediately following a Write-Access. Minimum write value: 0b1
0x0004	WDOGVALUE	RO	0xFFFF_FFFF	32	Current value of watchdog counter register.
0x0008	WDOGCONTROL	RW	0x0000_0000	8	Timer 1 control register. Bits [31:2] are reserved.
0x000C	WDOGINTCLR	WO	-	32	Watchdog interrupt clear register.
0x0010	WDOGRIS	RO	0x0000_0000	1	Watchdog interrupt status register. Bits [31:1] are reserved.

Table 3-8 CMSDK watchdog timers control registers summary (continued)

Offset	Name	Type	Reset	Width	Function
0x0014	WDOGMIS	RO	0x0000_0000	1	Watchdog status register. Bits [31:1] are reserved.
0x0C00	WDOGLOCK	RW	0x0000_0000	32	Watchdog lock register.
0x0F00	WDOGITCR	RW	0x0000_0000	32	Watchdog integration test control register. Bits [31:1] are reserved.
0x0F04	WDOGITOP	WO	0x0000_0000	32	Watchdog integration test output set register.
0x0FD0	WDOGPERIPHID4	RO	0x0000_0004	8	Peripheral ID Register 4. Bits [31:8] are reserved.
0x0FE0	WDOGPERIPHID0	RO	0x0000_0024	32	Peripheral ID Register 0. Bits [31:8] are reserved.
0x0FE4	WDOGPERIPHID1	RO	0x0000_00B8	1	Peripheral ID Register 1. Bits [31:8] are reserved.
0x0FE8	WDOGPERIPHID2	RO	0x0000_000B	1	Peripheral ID Register 2. Bits [31:8] are reserved.
0x0FEC	WDOGPERIPHID3	RO	0x0000_0000	32	Peripheral ID Register 3. Bits [31:8] are reserved.
0x0FF0	WDOGPCELLID0	RO	0x0000_000D	8	Component ID Register 0. Bits [31:8] are reserved.
0x0FF4	WDOGCELLID1	RO	0x0000_00F0	8	Component ID Register 1. Bits [31:8] are reserved.
0x0FF8	WDOGPCELLID2	RO	0x0000_0005	8	Component ID Register 2. Bits [31:8] are reserved.
0x0FFC	WDOGPCELLID3	RO	0x0000_00B1	8	Component ID Register 3. Bits [31:8] are reserved.

3.4.6 Secure Privilege Control Block

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

The base memory address of the Secure Privilege Control Block is 0x5008_0000.

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

Reads and writes are supported only from Secure Privileged access.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

The following table shows the registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-9 Secure Privilege Control Block registers

Offset	Name	Type	Reset value	Function
0x0000	SPCSECCTRL	RW	0x0000_0000	Secure Privilege Controller Secure Configuration Control Register. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x0004	BUSWAIT	RW	0x0000_0001	Bus Access wait control after reset. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x0010	SECRESPCFG	RW	0x0000_0000	Security Violation Response Configuration Register. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x0014	NSCCFG	RW	0x0000_0000	Non-secure Callable Configuration for IDAU. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x001C	SECMPCINTSTATUS	RO	0x0000_0000	Secure MPC Interrupt Status. See *SECMPCINTSTATUS Register* on page 3-69 for information on how this register is implemented in the Musca-S1 test chip.
0x0020	SECPPCINTSTAT	RO	0x0000_0000	Secure PPC Interrupt Status. See *SECPPCINTSTAT Register* on page 3-70 for information on how this register is implemented in the Musca-S1 test chip.
0x0024	SECPPCINTCLR	WO	0x0000_0000	Secure PPC Interrupt Clear. See *SECPPCINTCLR Register* on page 3-71 for information on how this register is implemented in the Musca-S1 test chip.
0x0028	SECPPCINTEN	RW	0x0000_0000	Secure PPC Interrupt Enable. See *SECPPCINTEN Register* on page 3-72 for information on how this register is implemented in the Musca-S1 test chip.
0x0040	BRGINTSTAT	RO	0x0000_0000	Bridge buffer error interrupt status register. See *BRGINTSTAT Register* on page 3-74 for information on how this register is implemented in the Musca-S1 test chip.
0x0044	BRGINTCLR	WO (Bit[0])	0x0000_0000	Bridge buffer error interrupt status register. See *BRGINTCLR Register* on page 3-74 for information on how this register is implemented in the Musca-S1 test chip.
0x0048	BRGINTEN	WO (Bit[0])	0x0000_0000	Bridge buffer error interrupt status register. See *BRGINTEN Register* on page 3-75 for information on how this register is implemented in the Musca-S1 test chip.

Table 3-9 Secure Privilege Control Block registers (continued)

Offset	Name	Type	Reset value	Function
0x0060	AHBNSPPCEXP0	RW	0x0000_0000	Expansion 0 Non-secure AHB slave Peripheral Protection Control. See *AHBNSPPCEXP0 Register* on page 3-75 for information on how this register is implemented in the Musca-S1 test chip.
0x0070	APBNSPPC0	RW	0x0000_0000	Non-secure Access APB slave Peripheral Protection Control 0. This register controls peripherals in the base element. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x0074	APBNSPPC1	RW	0x0000_0000	Non-secure Access APB slave Peripheral Protection Control 1. This register controls peripherals in the system control element. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x0080	APBNSPPCEXP0	RW	0x0000_0000	Expansion 0 Non-secure access APB slave Peripheral Protection Control. See *APBNSPPCEXP0 Register* on page 3-76 for information on how this register is implemented in the Musca-S1 test chip.
0x0084	APBNSPPCEXP1	RW	0x0000_0000	Expansion 1 Non-secure access APB slave Peripheral Protection Control. See *APBNSPPCEXP1 Register* on page 3-77 for information on how this register is implemented in the Musca-S1 test chip.
0x0090	AHBSPPPC0	RO	0x0000_0000	Secure Unprivileged Access AHB slave Peripheral. Protection Control 0. See *AHBSPPPC0 Register* on page 3-79 for information on how this register is implemented in the Musca-S1 test chip.
0x00A0	AHBSPPPCEXP0	RW	0x0000_0000	Expansion 0 Secure Unprivileged Access AHB slave Peripheral Protection Control. See *AHBSPPPCEXP0 Register* on page 3-80 for information on how this register is implemented in the Musca-S1 test chip.
0x00B0	APBSPPPC0	RW	0x0000_0000	Secure Unprivileged Access APB slave Peripheral. Protection Control 0. This register controls the PPC within the Base element. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x00B4	APBSPPPC1	RW	0x0000_0000	Secure Unprivileged Access APB slave Peripheral. Protection Control 1. This register controls the PPC within the System Control element. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
0x00C0	APBSPPPCEXP0	RW	0x0000_0000	Expansion 0 Secure Unprivileged access APB slave Peripheral Protection Control. See *APBSPPPCEXP0 Register* on page 3-80 for information on how this register is implemented in the Musca-S1 test chip.

Table 3-9 Secure Privilege Control Block registers (continued)

Offset	Name	Type	Reset value	Function
0x00C4	APBSPPPCEXP1	RW	0x0000_0000	Expansion 1 Secure Unprivileged access APB slave Peripheral Protection Control. See *APBSPPPCEXP1 Register* on page 3-81 for information on how this register is implemented in the Musca-S1 test chip.
0x0FD0	PID4	RO	0x0000_0004	Peripheral ID 4
0x0FE0	PID0	RO	0x0000_0052	Peripheral ID 0
0x0FE4	PID1	RO	0x0000_00B8	Peripheral ID 1
0x0FE8	PID2	RO	0x0000_000B	Peripheral ID 2
0x0FEC	PID3	RO	0x0000_0000	Peripheral ID 3
0x0FF0	CID0	RO	0x0000_000D	Component ID 0
0x0FF4	CID1	RO	0x0000_00F0	Component ID 1
0x0FF8	CID2	RO	0x0000_0005	Component ID 2
0x0FFC	CID3	RO	0x0000_00B1	Component ID 3

SECMPCINTSTATUS Register

The Secure *Memory Protection Controller* (MPC) Interrupt Status Register characteristics are:

Purpose

Stores the interrupt statuses of the *Memory Protection Controllers* (MPCs).
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the SECMPCINTSTATUS Register.

Table 3-10 SECMPCINTSTATUS Register bit assignments

Bits	Name	Function
[31:19]	-	Reserved.
[18]	S_MPCMRAM_STATUS	Interrupt status of eMRAM Memory Protection Controller. Reset value: 0b0.
[17]	S_MPCSRAM_STATUS	Interrupt status of Code SRAM Memory Protection Controller. Reset value: 0b0.
[16]	S_MPCQSPI_STATUS	Interrupt status of QSPI Memory Protection Controller. Reset value: 0b0.

Table 3-10 SECMPCINTSTATUS Register bit assignments (continued)

Bits	Name	Function
[15:4]	-	Reserved.
[3]	S_MPCSRAM3_STATUS	Interrupt Status of SRAM bank 3 Memory Protection Controller. Reset value: `0b0`.
[2]	S_MPCSRAM2_STATUS	Interrupt Status of SRAM bank 2 Memory Protection Controller. Reset value: `0b0`.
[1]	S_MPCSRAM1_STATUS	Interrupt Status of SRAM bank 1 Memory Protection Controller. Reset value: `0b0`.
[0]	S_MPCSRAM0_STATUS	Interrupt Status of SRAM bank 0 Memory Protection Controller. Reset value: `0b0`.

SECPPCINTSTAT Register

The Secure *Peripheral Protection Controller* (PPC) Interrupt Status Register characteristics are:

Purpose

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised from a combined interrupt to the Cortex-M33 *Nested Vector Interrupt Controller* (NVIC). The PPC Secure PPC Interrupt Status (SECPPCINTSTAT), Clear (SECPPCINTCLR) and Enable (SECPPCINTEN) Registers enable software to determine the source of the interrupt, Clear the interrupt, and enable or disable (Mask) the interrupt.
The SECPPCINTSTAT Register stores the interrupt statuses of *Peripheral Protection Controllers* (PPCs).
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the SECPPCINTSTAT Register.

Table 3-11 SECPPCINTSTAT Register bit assignments

Bits	Name	Function
[31:21]	-	Reserved.
[20]	S_AHBPPCGPIO_STATUS	Interrupt status of Peripheral Protection Controller for the AHB GPIO slaves. Reset value: `0b0`.
[19:6]	-	Reserved.

Table 3-11 SECPPCINTSTAT Register bit assignments (continued)

Bits	Name	Function
[5]	S_APBPPCSYSP_STATUS	Interrupt status of Peripheral Protection Controller for APB slaves within the Musca-S1 test chip system level peripherals. Reset value: `0b0`.
[4]	S_APBPPCMEM_STATUS	Interrupt status of Peripheral Protection Controller for APB slaves within the memory subsystem. Reset value: `0b0`.
[3:2]	-	Reserved.
[1]	S_APBPPC1PERIP_STATUS	Interrupt status of Peripheral Protection Controller for APB slaves within the system control element. Reset value: `0b0`.
[0]	S_APBPPC0PERIP_STATUS	Interrupt status of Peripheral Protection Controller for APB slaves within the base element. Reset value: `0b0`.

SECPPCINTCLR Register

The Secure *Peripheral Protection Controller* (PPC) Interrupt Clear Register characteristics are:

Purpose

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised from a combined interrupt to the Cortex-M33 *Nested Vector Interrupt Controller* (NVIC). The PPC Secure PPC Interrupt Status (SECPPCINTSTAT), Clear (SECPPCINTCLR) and Enable (SECPPCINTEN) Registers enable software to determine the source of the interrupt, Clear the interrupt, and enable or disable (Mask) the interrupt.
The SECPPINTCLR Register clears the *Peripheral Protection Controller* (PPC) interrupts.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is write-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the SECPPCINTCLR Register.

Table 3-12 SECPPCINTCLR Register bit assignments

Bits	Name	Function
[31:21]	-	Reserved.
[20]	S_AHBPPCGPIO_CLR	Interrupt Clear of Peripheral Protection Controller for the AHB GPIO slaves: `0b0`: No effect. `0b1`: Clear interrupt. Reset value: `0b0`.
[19:6]	-	Reserved.
[5]	S_APBPPCSYSP_CLR	Interrupt Clear of Peripheral Protection Controller for APB slaves within the Musca-S1 test chip system level peripherals: `0b0`: No effect. `0b1`: Clear interrupt. Reset value: `0b0`.
[4]	S_APBPPCMEM_CLR	Interrupt Clear of Peripheral Protection Controller for APB slaves within the memory subsystem: `0b0`: No effect. `0b1`: Clear interrupt. Reset value: `0b0`.
[3:2]	-	Reserved.
[1]	S_APBPPC1PERIP_CLR	Interrupt Clear of Peripheral Protection Controller for APB slaves within the system control element: `0b0`: No effect. `0b1`: Clear interrupt. Reset value: `0b0`.
[0]	S_APBPPC0PERIP_CLR	Interrupt Clear of Peripheral Protection Controller for APB slaves within the base element. `0b0`: No effect. `0b1`: Clear interrupt. Reset value: `0b0`.

SECPPCINTEN Register

The *Secure Peripheral Protection Controller* (PPC) Interrupt Enable Register characteristics are:

Purpose

When access violations occur on any Peripheral Protection Controller, a level interrupt is raised from a combined interrupt to the Cortex-M33 *Nested Vector Interrupt Controller* (NVIC). The PPC Secure PPC Interrupt Status (SECPPCINTSTAT), Clear (SECPPCINTCLR) and Enable (SECPPCINTEN) Registers enable software to determine the source of the interrupt, Clear the interrupt, and enable or disable (Mask) the interrupt.
The SECPPCINTEN Register enables or disables, that is, masks, the *Peripheral Protection Controller* (PPC) interrupts.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the SECPPCINTEN Register.

Table 3-13 SECPPCINTEN Register bit assignments

Bits	Name	Function
[31:21]	-	Reserved.
[20]	S_AHBPPCGPIO_EN	Interrupt Enable of Peripheral Protection Controller for the AHB GPIO slaves: `0b0`: Mask interrupt. `0b1`: Enable interrupt. Reset value: `0b0`.
[19:6]	-	Reserved.
[5]	S_APBPPCSYSP_EN	Interrupt Enable of Peripheral Protection Controller for APB slaves within the Musca-S1 test chip system level peripherals: `0b0`: Mask interrupt. `0b1`: Enable interrupt. Reset value: `0b0`.
[4]	S_APBPPCMEM_EN	Interrupt Enable of Peripheral Protection Controller for APB slaves within the memory subsystem: `0b0`: Mask interrupt. `0b1`: Enable interrupt. Reset value: `0b0`.
[3:2]	-	Reserved.

Table 3-13 SECPPCINTEN Register bit assignments (continued)

Bits	Name	Function
[1]	S_APBPPC1PERIP_EN	Interrupt Enable of Peripheral Protection Controller for APB slaves within the system control element: `0b0`: Mask interrupt. `0b1`: Enable interrupt. Reset value: `0b0`.
[0]	S_APBPPC0PERIP_EN	Interrupt Enable of Peripheral Protection Controller for APB slaves within the base element: `0b0`: Mask interrupt. `0b1`: Enable interrupt. Reset value: `0b0`.

BRGINTSTAT Register

The Bridge Buffer Error Interrupt Status Register characteristics are:

Purpose

Stores the interrupt status of the bridge between CPU1 and the system.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the BRGINTSTAT Register.

Table 3-14 BRGINTSTAT Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved.
[0]	BRG_CPU1SYS_STATUS	Interrupt Status of write buffer bridge error for bridge between CPU1 and the system. Reset value `0b0`.

BRGINTCLR Register

The Bridge Buffer Error Interrupt Clear Register characteristics are:

Purpose

Clears the interrupts of the bridge between CPU1 and the system.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is write-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the BRGINTCLR Register.

Table 3-15 BRGINTCLR Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved.
[0]	BRG_CPU1SYS_CLR	Clear interrupt of write buffer bridge error for bridge between CPU1 and the system. `0b0`: No effect. `0b1`: Clear interrupt. Reset value `0b0`.

BRGINTEN Register

The Bridge Buffer Error Interrupt Enable Register characteristics are:

Purpose

Enables or disables the interrupt of the write buffer bridge error for bridge between CPU1 and the system.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is write-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the BRGINTEN Register.

Table 3-16 BRGINTEN Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved.
[0]	BRG_CPU1SYS_EN	Enable or disable the interrupt of the write buffer bridge error for the bridge between CPU1 and the system. `0b0`: Disable (mask) interrupt. `0b1`: Enable interrupt. Reset value `0b0`.

AHBNSPPCEXP0 Register

The Expansion 0 Non-secure Access AHB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the security access settings for the associated AHB slave *Peripheral Protection Controllers* (PPCs) outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the AHBNSPPCEXP0 Register.

Table 3-17 AHBNSPPCEXP0 Register bit assignments

Bits	Name	Function
[31:2]	-	Reserved.
[1]	NS_GPIO	Defines the security access setting for the GPIO: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[0]	-	Reserved.

APBNSPPCEXP0 Register

The Expansion 0 Non-secure Access APB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the security access settings for the associated APB slave *Peripheral Protection Controllers* (PPCs) for the memory subsystem, outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the APBNSPPCEXP0 Register.

Table 3-18 APBNSPPCEXP0 Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2]	NS_MRAM_MPC	Defines the security access setting for the eMRAM *Memory Protection Controller* (MPC): `0b0`: Secure access only. `0b1`:Secure and Non-secure access. Reset value `0b0`.

Table 3-18 APBNSPPCEXP0 Register bit assignments (continued)

Bits	Name	Function
[1]	NS_SRAM_MPC	Defines the security access setting for the Code SRAM MPC: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[0]	NS_QSPI_MPC	Defines the security access setting for the QSPI MPC: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.

APBNSPPCEXP1 Register

The Expansion 1 Non-secure Access APB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the security access settings for the associated APB slave *Peripheral Protection Controllers* (PPCs) outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the APBNSPPCEXP1 Register.

Table 3-19 APBNSPPCEXP1 Register bit assignments

Bits	Name	Function
[31:14]	-	Reserved.
[13]	NS_PWM2	Defines the security access setting for the PWM2 interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[12]	NS_PWM1	Defines the security access setting for the PWM1 interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.

Table 3-19 APBNSPPCEXP1 Register bit assignments (continued)

Bits	Name	Function
[11]	NS_SCC	Defines the security access setting for the *Serial Configuration Control* (SCC): `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[10]	NS_GPTIMER	Defines the security access setting for the General-purpose timer: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[9]	NS_QSPI	Defines the security access setting for the QSPI: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[8]	NS_PVT	Defines the security access setting for the PVT sensors: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[7]	NS_RTC	Defines the security access setting for the Real Time Clock: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[6]	NS_PWM0	Defines the security access setting for the PWM0 interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[5]	NS_I2S	Defines the security access setting for the I^2S interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.

Table 3-19 APBNSPPCEXP1 Register bit assignments (continued)

Bits	Name	Function
[4]	NS_I2C1	Defines the security access setting for the I^2C1 interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[3]	NS_I2C0	Defines the security access setting for the I^2C0 interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[2]	NS_SPI	Defines the security access setting for the SPI interface: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[1]	NS_UART1	Defines the security access setting for the UART1: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.
[0]	NS_UART0	Defines the security access setting for the UART0: `0b0`: Secure access only. `0b1`: Secure and Non-secure access. Reset value `0b0`.

AHBSPPPC0 Register

The Secure Unprivileged Access AHB slave Peripheral Protection Control 0 Register characteristics are:

Purpose

Defines the Secure Unprivileged access setting for the associated AHB slave *Peripheral Protection Controller* (PPC) outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the AHBSPPPC0 Register.

Table 3-20 AHBNSPPCEXP0 Register bit assignments

Bits	Name	Function
[31:2]	-	Reserved.
[1]	NS_GPIO	Defines the Secure Unprivileged access setting for the GPIO: `0b0`: Secure privileged access only. `0b1`: Secure Unprivileged and privileged access. Reset value `0b0`.
[0]	-	Reserved.

AHBSPPPCEXP0 Register

The Expansion 0 Secure Unprivileged Access AHB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the Secure Unprivileged access setting for the associated AHB slave *Peripheral Protection Controller* (PPC) outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the AHBSPPPCEXP0 Register.

Table 3-21 AHBSPPPCEXP0 Register bit assignments

Bits	Name	Function
[31:2]	-	Reserved.
[1]	NS_GPIO	Defines the Secure Unprivileged access setting for the GPIO: `0b0`: Secure privileged access only. `0b1`: Secure Unprivileged and privileged access. Reset value `0b0`.
[0]	-	Reserved.

APBSPPPCEXP0 Register

The Expansion 0 Secure Unprivileged Access APB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the Secure Unprivileged access settings for the associated APB slave *Peripheral Protection Controllers* (PPCs) for the memory subsystem, outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the APBSPPPCEXP0 Register.

Table 3-22 APBSPPPCEXP0 Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2]	S_MRAM_MPC	Defines the Secure Unprivileged access setting for the eMRAM *Memory Protection Controller* (MPC): `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[1]	S_SRAM_MPC	Defines the Secure Unprivileged access setting for the Code SRAM MPC: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[0]	S_QSPI_MPC	Defines the Secure Unprivileged access setting for the QSPI MPC: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.

APBSPPPCEXP1 Register

The Expansion 1 Secure Unprivileged Access APB slave Peripheral Protection Control Register characteristics are:

Purpose
Defines the Secure access settings for the associated APB slave *Peripheral Protection Controllers* (PPCs) for peripherals on the APB PPC Multiplexer, outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.4.6 Secure Privilege Control Block* on page 3-66.

The following table shows the bit assignments of the APBSPPPCEXP1 Register.

Table 3-23 APBSPPPCEXP1 Register bit assignments

Bits	Name	Function
[31:14]	-	Reserved.
[13]	S_PWM2	Defines the Secure access setting for PWM2: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[12]	S_PWM1	Defines the Secure access setting for PWM1: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[11]	S_SCC	Defines the Secure access setting for the *Serial Configuration Controller* (SCC): `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[10]	S_GPTIMER	Defines the Secure access setting for the General-purpose timer: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[9]	S_QSPI	Defines the Secure access setting for the QSPI: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[8]	S_PVT	Defines the Secure access setting for the PVT sensor system: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.

Table 3-23 APBSPPPCEXP1 Register bit assignments (continued)

Bits	Name	Function
[7]	S_RTC	Defines the Secure access setting for the *Real Time Clock* (RTC): `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[6]	S_PWM0	Defines the Secure access setting for PWM0: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[5]	S_I2S	Defines the Secure access setting for I^2S: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[4]	S_I2C1	Defines the Secure access setting for I^2C1: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[3]	S_I2C0	Defines the Secure access setting for I^2C0: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[2]	S_SPI	Defines the Secure access setting for the SPI: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.

Table 3-23 APBSPPPCEXP1 Register bit assignments (continued)

Bits	Name	Function
[1]	S_UART1	Defines the Secure access setting for UART1: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.
[0]	S_UART0	Defines the Secure access setting for UART0: `0b0`: Secure Privileged access only. `0b1`: Secure Unprivileged and Privileged access. Reset value `0b0`.

3.4.7 Non-secure Privilege Control Block

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

The base memory address of the Non-secure Privilege Control Block is `0x4008_0000`.

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

Reads and writes are supported only from Non-secure Privileged access.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

The following table shows the registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-24 Non-secure Privilege Control Block registers

Offset	Name	Type	Reset value	Function
`0x00A0`	AHBNSPPPCEXP0	RW	`0x0000_0000`	Expansion 0 Non-secure Unprivileged Access AHB slave Peripheral Protection Control. See *AHBNSPPPCEXP0 Register* on page 3-85 for information on how this register is implemented in the Musca-S1 test chip.
`0x00B0`	APBNSPPPC0	RW	`0x0000_0000`	Non-secure Unprivileged Access APB slave Peripheral Protection Control 0. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
`0x00B4`	APBNSPPPC1	RW	`0x0000_0000`	Non-secure Unprivileged Access APB slave Peripheral Protection Control 1. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.
`0x00C0`	APBNSPPPCEXP0	RW	`0x0000_0000`	Expansion 0 Non-secure Unprivileged Access APB slave Peripheral Protection Control. See *APBNSPPPCEXP0 Register* on page 3-86 for information on how this register is implemented in the Musca-S1 test chip.

Table 3-24 Non-secure Privilege Control Block registers (continued)

Offset	Name	Type	Reset value	Function
0x00C4	APBNSPPPCEXP1	RW	0x0000_0000	Expansion 1 Non-secure Unprivileged Access APB slave Peripheral Protection Control. See *APBNSPPPCEXP1 Register* on page 3-86 for information on how this register is implemented in the Musca-S1 test chip.
0x0FD0	PID4	RO	0x0000_0004	Peripheral ID 4
0x0FE0	PID0	RO	0x0000_0053	Peripheral ID 0
0x0FE4	PID1	RO	0x0000_00B8	Peripheral ID 1
0x0FE8	PID2	RO	0x0000_000B	Peripheral ID 2
0x0FEC	PID3	RO	0x0000_0000	Peripheral ID 3
0x0FF0	CID0	RO	0x0000_000D	Component ID 0
0x0FF4	CID1	RO	0x0000_00F0	Component ID 1
0x0FF8	CID2	RO	0x0000_0005	Component ID 2
0x0FFC	CID3	RO	0x0000_00B1	Component ID 3

AHBNSPPPCEXP0 Register

The Expansion 0 Non-secure Unprivileged Access AHB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the Non-secure unprivileged access settings for the associated AHB slave *Peripheral Protection Controllers* (PPCs) outside the SSE-200 subsystem.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.7 Non-secure Privilege Control Block* on page 3-84.

The following table shows the bit assignments of the AHBNSPPPCEXP0 Register.

Table 3-25 AHBNSPPPCEXP0 Register bit assignments Register

Bits	Name	Function
[31:2]	-	Reserved.
[1]	NS_GPIO	Defines the Non-secure access setting for the GPIO: 0b0: Non-secure privileged access only. 0b1: Non-secure unprivileged and privileged access. Reset value 0b0.
[0]	-	Reserved.

APBNSPPPCEXP0 Register

The Expansion 0 Non-secure unprivileged Access APB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the Non-secure unprivileged access settings for the associated APB slave *Peripheral Protection Controllers* (PPCs), for the memory subsystem, outside the SSE-200 subsystem. See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.7 Non-secure Privilege Control Block* on page 3-84.

The following table shows the bit assignments of the APBNSPPPCEXP0 Register.

Table 3-26 APBNSPPPCEXP0 Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2]	NS_MRAM_MPC	Defines the Non-secure access settings for the eMRAM *Memory Protection Controller* (MPC): 0: Non-secure Privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[1]	NS_SRAM_MPC	Defines the Non-secure access settings for the eFlash 1 controller: 0: Non-secure Privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[0]	NS_QSPI_MPC	Defines the Non-secure access settings for the QSPI MPC: 0: Non-secure Privileged access only. 1: Non-secure unPrivileged and privileged access. Reset value `0b0`.

APBNSPPPCEXP1 Register

The Expansion 1 Non-secure unprivileged Access APB slave Peripheral Protection Control Register characteristics are:

Purpose

Defines the Non-secure unprivileged access settings for the associated APB slave *Peripheral Protection Controllers* (PPCs) outside the SSE-200 subsystem.
See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)* for more information.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.4.7 Non-secure Privilege Control Block* on page 3-84.

The following table shows the bit assignments of the APBNSPPPCEXP1 Register.

Table 3-27 APBNSPPPCEXP1 Register bit assignments

Bits	Name	Function
[31:14]	-	Reserved.
[13]	NS_PWM2	Defines the Non-secure access setting for PWM2: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[12]	NS_PWM1	Defines the Non-secure access setting for PWM1: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[11]	NS_SCC	Defines the Non-secure access setting for the *Serial Configuration Controller* (SCC): 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[10]	NS_GPTIMER	Defines the Non-secure access setting for the General-purpose timer: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.

Table 3-27 APBNSPPPCEXP1 Register bit assignments (continued)

Bits	Name	Function
[9]	NS_QSPI	Defines the Non-secure access setting for the QSPI: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[8]	NS_PVT	Defines the Non-secure access setting for the PVT sensor system: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[7]	NS_RTC	Defines the Non-secure access setting for the *Real Time Clock* (RTC): 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[6]	NS_PWM0	Defines the Non-secure access setting for PWM0: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[5]	NS_I2S	Defines the Non-secure access setting for I^2S: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[4]	NS_I2C1	Defines the Non-secure access setting for I^2C1: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.

Table 3-27 APBNSPPPCEXP1 Register bit assignments (continued)

Bits	Name	Function
[3]	NS_I2C0	Defines the Non-secure access setting for I^2C0: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[2]	NS_SPI	Defines the Non-secure access setting for the SPI: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[1]	NS_UART1	Defines the Non-secure access setting for UART1: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.
[0]	NS_UART0	Defines the Non-secure access setting for UART0: 0: Non-secure privileged access only. 1: Non-secure Privileged and Unprivileged access. Reset value `0b0`.

3.4.8 Message Handling Unit

Two *Message Handling Units* (MHUs) in the base element enable software to raise interrupts to the processor cores.

MHU0 base memory addresses are:

- `0x4000_3000` in the Non-secure region.
- `0x5000_3000` in the Secure region.

MHU1 base memory addresses are:

- `0x4000_4000` in the Non-secure region.
- `0x5000_4000` in the Secure region.

The *Peripheral Protection Controller* (PPC) controls the area in which the MHU resides.

Only 32-bit writes are supported. Byte and halfword writes are ignored.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)*.

The following table shows the MHU0 and MHU1 registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-28 MHU registers

Offset	Name	Type	Reset value	Description
0x0000	CPU0INTR_STAT	RO	0x0000_0000	Core 0 interrupt status register.
0x0004	CPU0INTR_SET	WO	0x0000_0000	Core 0 interrupt set register.
0x0008	CPU0INTR_CLR	WO	0x0000_0000	Core 0 interrupt clear register.
0x0010	CPU1INTR_STAT	RO	0x0000_0000	Core 1 interrupt status register.
0x0014	CPU1INTR_SET	WO	0x0000_0000	Core 1 interrupt set register.
0x0018	CPU1INTR_CLR	WO	0x0000_0000	Core 1 interrupt clear register.
0x0FD0	PIDR4	RO	0x0000_0004	Peripheral ID 4.
0x0FE0	PIDR0	RW	0x0000_0056	Peripheral ID 0.
0x0FE4	PIDR1	RO	0x0000_00B8	Peripheral ID 1.
0x0FE8	PIDR2	RO	0x0000_0000	Peripheral ID 2.
0x0FEC	PIDR3	WO	0x0000_0000	Peripheral ID 3.
0x0FF0	CIDR0	RO	0x0000_000D	Component ID 0.
0x0FF4	CIDR1	RO	0x0000_00F0	Component ID 1.
0x0FF8	CIDR2	RO	0x0000_0005	Component ID 2.
0x0FFC	CIDR3	RO	0x0000_00B1	Component ID 3.

3.4.9 AHB5 TrustZone Memory Protection Controllers

The Musca-S1 test chip implements AHB5 TrustZone *Memory Protection Controllers* (MPCs) for certain blocks including blocks in the base element.

The base memory addresses of the MPC APB configuration interfaces are in the Secure region. The base memory addresses of the configuration interfaces in the base element are:

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

The following MPC base memory addresses are not in the base element but are shown here for convenience:

In the Non-secure region:

- 0x4012_0000 for QSPI.
- 0x4013_0000 for SRAM.
- 0x4014_0000 for eMRAM.

In the Secure region:

- 0x5012_0000 for QSPI.
- 0x5013_0000 for SRAM.
- 0x5014_0000 for eMRAM.

The AHB5 TrustZone MPC gates transactions towards a memory interface when a security violation occurs. The security checking is done based on block/page level which is configured externally by the security controller through an APB interface.

The configuration registers can only be set by the security controller in the system with secure accesses (PRTO[1]==0). Any type of access can read the identification registers.

APB accesses are internally aligned to word boundaries, so PADDR[1:0] bits are ignored. The PSTRB[3:0] write strobe signals indicate which byte or bytes of the data bus contain valid data.

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

The following table shows the AHB5 TrustZone MPC registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-29 AHB5 TrustZone MPC registers

Offset	Name	Type	Reset value	Function
0x0000	CTRL	RW	0x0000_0000	Bit[31]: Security lockdown Bit[30:9]: Reserved Bit[8]: Autoincrement Reserved when BLK_SIZE > ADDR_WIDTH-11 Bit[7]: Data interface gating acknowledge (RO) Reserved when GATE_PRESENT = 0 Bit[6]: Data interface gating request. Reserved when GATE_PRESENT = 0 Bit[5]: Reserved Bit[4]: Security error response configuration (CFG_SEC_RESP) 0:RAZ-WI 1: Bus Error Bit[3:0]: Reserved
0x0010	BLK_MAX	RO	-	Maximum value of block-based index register.
0x0014	BLK_CFG	RO	-	Bit[31]: Init in progress Bit[30:4]: Reserved Bit[3:0]: Block size: 0: 32 Bytes 1: 64 Bytes … 15: 1MByte Block size = 1 << (BLK_CFG+5)
0x0018	BLK_IDX	RW	0x0000_0000	Index value for accessing block-based lookup table.

Table 3-29 AHB5 TrustZone MPC registers (continued)

Offset	Name	Type	Reset value	Function
0x001C	BLK_LUT[n]	RW	0x0000_0000	Block-based gating Look Up Table (LUT): Access to block-based lookup configuration space pointed to by BLK_IDX. Bit[31:0]: each bit indicates one block: If BLK_IDX is 0x0, bit[0] is block #0, bit[31] is block#31. If BLK_IDX is 0x1, bit[0] is block #32, bit[31] is block#63. If BLK_IDX is 0x2, bit[0] is block#64, bit[31] is block#95 If BLK_IDX is 0xFFF, bit[0] is block#131040, bit[31] is block#131071. The maximum value of BLK_IDX is defined by the BLK_MAX register. 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 if enabled by CTRL[8]. The upper bits are reserved if BLK_SIZE > ADDR_WIDTH - 11.
0x0020	INT_STAT	RO	0x0000_0000	Bits[31:1]: Reserved. Bit[0]: **mpc_irq** triggered.
0x0024	INT_CLEAR	WO	0x0000_0000	Bits[31:1]: Reserved Bit[0]: **mpc_irq** clear (cleared automatically).
0x0028	INT_EN	RW	0x0000_0000	Bits[31:1]: Reserved. Bit[0]: **mpc_irq** enable. Enables interrupt output generation. The INT_STAT, INT_INFO1, and INT_INFO2 registers are still set for errors.
0x002C	INT_INFO1	RO	0x0000_0000	**haddr[31:0]** of the first security violating address. Bits are valid when **mpc_irq** is triggered. Subsequent security violation transfers remain blocked, that is, not captured in this register and the register retains its value until **mpc_irq** is cleared.
0x0030	INT_INFO2	RO	0x0000_0000	Additional control bits of the first security violating transfer. Bit [31:18]: Reserved. Bit [17]: **cfg_ns**. Bit [16]: **hnonsec**. Bit [15:0]: **hmaster**. Bits are valid when **mpc_irq** is triggered. Subsequent security violating transfers remain blocked, that is, not captured in this register and the register retains its value until mpc_irq is cleared.
0x0034	INT_SET	WO	0x0000_0000	Bit[31:1]: Reserved. Bit[0]: **mpc_irq** set. Debug purpose only. Sets **mpc_irq** triggered in INT_STAT regardless of the **mpc_irq_enable** input.

Table 3-29 AHB5 TrustZone MPC registers (continued)

Offset	Name	Type	Reset value	Function
0x0FD0	PIDR4	RO	0x0000_0004	Peripheral ID 4. Bits[7:4] block count. Bits [3:0] jep106_c_code.
0x0FD4	PIDR5	RO	0x0000_0000	Peripheral ID 5 (not used).
0x0FD8	PIDR6	RO	0x0000_0000	Peripheral ID 6, not used.
0x0FDC	PIDR7	RO	0x0000_0000	Peripheral ID 7 (not used).
0x0FE0	PIDR0	RO	0x0000_0060	Peripheral ID 0. Bits [31:8]: Reserved Bits [7:0]. Part number [7:0].
0x0FE4	PIDR1	RO	0x0000_00B8	Peripheral ID 1. Bits[7:4] jep106_id_3_0. Bits[3:0] Part number[11:8]).
0x0FE8	PIDR2	RO	00000_000B	Peripheral ID 2. Bits[7:4] revision. Bit[3] jedec_used Bits[2:0] jep106_id_6_4.
0x0FEC	PIDR3	RO	0x0000_0000	Peripheral ID 3. Bits[7:4] ECO revision number. Bits[3:0] customer modification number.
0x0FF0	CIDR0	RO	0x0000_000D	Component ID 0.
0x0FF4	CIDR1	RO	0x0000_00F0	Component ID 1 (PrimeCell class).
0x0FF8	CIDR2	RO	0x0000_0005	Component ID 2.
0x0FFC	CIDR3	RO	0x0000_00B1	Component ID 3.

Look Up Table examples

The contents of the *Look Up Table* (LUT) can be accessed in several ways that might require different configurations of the autoincrement function of the BLK_IDX register.

To read the full contents of the LUT:

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

To write the full contents of the LUT:

1. Set autoincrement enable bit, CTRL[8], to `0x1`.
2. Read the BLK_MAX register. This register 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/write/modify a single location:

1. Set autoincrement enable bit, CTRL[8], to `0x1`.
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.

When the configuration lockdown feature is enabled:

- The lockdown feature can only be disabled by a component reset that resets CTRL[31] to `0x0`.
- The following registers are read-only:
  - CTRL.
  - BLK_LUT.
  - INT_EN.

Note

Arm recommends that you write `0x1` to the LUT autoincrement bit, CTRL[8], before enabling the configuration lockdown feature. When the feature is enabled, only LUT reading is available which is simpler when BLK_IDX increments automatically during the read sequence.

3.5 System control element

This section describes the registers that control the blocks in the SSE-200 subsystem.

This section contains the following subsections:

3.5.1 System control regions

The system control regions contain 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 3-30 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* (PPU)	SP
8	`0x5002_3000`	`0x5002_3FFF`	4KB	CPU0CORE_PPU	CPU0 core *Power Policy Unit* (PPU)	SP
9	`0x5002_4000`	`0x5002_4FFF`	4KB	CPU0DBG_PPU	CPU0 debug *Power Policy Unit* (PPU)	SP
10	`0x5002_5000`	`0x5002_5FFF`	4KB	CPU1CORE_PPU	CPU1 core *Power Policy Unit* (PPU)	SP
11	`0x5002_6000`	`0x5002_6FFF`	4KB	CPU1DBG_PPU	CPU1 debug *Power Policy Unit* (PPU)	SP
	`0x5002_7000`	`0x5002_7FFF`	4KB	CRYPTO_PPU	CryptoCell (r1p0) *Power Policy Unit* (PPU)	SP
	`0x5002_8000`	`0x5002_8FFF`	-	Reserved	Reserved.[b]	-
12	`0x5002_9000`	`0x5002_9FFF`	4KB	DEBUG_PPU	System debug *Power Policy Unit* (PPU).	SP
13	`0x5002_A000`	`0x5002_AFFF`	4KB	RAM0_PPU	SRAM Bank 0*Power Policy Unit* (PPU)	SP
14	`0x5002_B000`	`0x5002_BFFF`	4KB	RAM1_PPU	SRAM Bank 1 *Power Policy Unit* (PPU)	SP
15	`0x5002_C000`	`0x5002_CFFF`	4KB	RAM2_PPU	SRAM Bank 2 *Power Policy Unit* (PPU)	SP
16	`0x5002_D000`	`0x5002_DFFF`	4KB	RAM3_PPU	SRAM Bank 3 system *Power Policy Unit* (PPU)	SP

[a] This region is RZZ/WI.
[b] This region is RAZ/WI.

Table 3-30 System control regions (continued)

Row ID (alias)	Address		Size	Region name	Description	Security
	From	To				
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	Reserved	-

3.5.2 System Information Register Block

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.

The base memory addresses of the System Information Register Block are:

- 0x4002_0000 in the Non-secure region.
- 0x5002_0000 in the Secure region.

Note

The System Information Registers Block is visible to both regions without any security protection.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)*.

The following table shows the System Information registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-31 System Information Registers summary

Offset	Name	Access	Reset value	Description	Security
0x000	SYS_VERSION	RO	0x2004_1743	System Version Register	All
0x004	SYS_CONFIG	RO	0x2230_1544	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
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

3.5.3 System Control Register Block

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

The base memory address of the System Control Register Block is `0x5002_1000` in the Secure region of the base peripheral region.

The System Control Registers are secure privilege access only and support only 32-bit writes. Attempted byte and halfword writes are ignored.

See the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)*.

The following table shows the System Control Registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-32 System Control Registers summary

Offset	Name	Access	Reset value	Description
`0x0000`	SECDBGSTAT	RO	`0x0000_0000`	Secure debug configuration status.
`0x0004`	SECDBGSET	WO	`0x0000_0000`	Secure debug configuration set.
`0x0008`	SECDBGCLR	WO	`0x0000_0000`	Secure debug configuration clear.
`0x000C`	SCSECCTRL	RW	`0x0000_0000`	System security control.
`0x0010`	FCLK_DIV	RW	`0x0000_0000`	Fast clock divider configuration.
`0x0014`	SYSCLK_DIV	RW	`0x0000_0000`	System clock divider configuration.
`0x0018`	CLOCK_FORCE	RW	`0x0000_0000`	Clock Force.
`0x0100`	RESET_SYNDROME	RW	`0x0000_0001`	Reset syndrome. Register only cleared at powerup reset.
`0x0104`	RESET_MASK	RW	`0x0000_0030`	Reset mask.
`0x0108`	SWRESET	WO	`0x0000_0000`	Software reset.
`0x010C`	GRETREG	RW	`0x0000_0000`	General-purpose retention.
`0x0110`	INITSVRTOR0	RW	`0x0020_0000`	Initial Secure reset vector Register for CPU0.
`0x0114`	INITSVRTOR1	RW	`0x0020_0000`	Initial Secure reset vector Register for CPU1.
`0x0118`	CPUWAIT	RW	`0x0000_0000`	CPU boot wait control after reset.
`0x011C`	NMI_ENABLE	RW	`0x0000_0001`	NMI Enable.
`0x0120`	WICCTRL	RW	`0x0000_0000`	WIC request and acknowledge handshake.
`0x0124`	EWCTRL	RW	`0x0000_0000`	External Wakeup Control.
`0x0200`	PDCM_PD_SYS_SENSE	RW	`0x0000_007F`	Power Control Dependency Matrix. PD_SYS power domain sensitivity.
`0x020C`	PDCM_PD_SRAM0_SENSE	RW	`0x0000_0000`	Power Control Dependency Matrix. PD_SRAM0 power domain sensitivity.
`0x0210`	PDCM_PD_SRAM1_SENSE	RW	`0x0000_0000`	Power Control Dependency Matrix. PD_SRAM1 power domain sensitivity.

Table 3-32 System Control Registers summary (continued)

Offset	Name	Access	Reset value	Description
0x0214	PDCM_PD_SRAM2_SENSE	RW	0x0000_0000	Power Control Dependency Matrix. PD_SRAM2 power domain sensitivity.
0x0218	PDCM_PD_SRAM3_SENSE	RW	0x0000_0000	Power Control Dependency Matrix. PD_SRAM3 power domain sensitivity.
0x0FD0	PIDR4	RO	0x0000_0004	Peripheral ID4
0x0FE0	PIDR0	RO	0x0000_0054	Peripheral ID0
0x0FE4	PIDR1	RO	0x0000_00B8	Peripheral ID1
0x0FE8	PIDR2	RO	0x0000_000B	Peripheral ID2
0x0FEC	PIDR3	RO	0x0000_0000	Peripheral ID3
0x0FF0	CIDR0	RO	0x0000_000D	Component ID0
0x0FF4	CIDR1	RO	0x0000_00F0	Component ID1
0x0FF8	CIDR2	RO	0x0000_0005	Component ID2
0x0FFC	CIDR3	RO	0x0000_00B1	Component ID3

FCLK_DIV Register

The FCLK_DIV Register characteristics are:

Purpose

Controls the divider value of clock divider FCLKDIV that derives **FCLK**, in the SSE-200 subsystem, from **MAINCLK** in the Musca-S1 test chip. **FCLK** drives the secondary processor element, CPU1.

Usage constraints

Bits[20:16] are read-only. Bits[4:0] are read/write. The other bits are reserved.

Memory offset and full register reset value

See *3.5.3 System Control Register Block* on page 3-97.

The following table shows the bit assignments.

Table 3-33 FCLK_DIV Register bit assignments

Bits	Name	Function
[31:21]	-	Reserved.
[20:16]	FCLKDIV_CUR	Current value of FCLKDIV: The division value of FCLKDIV divider is FCLKDIV_CUR+1. These bits are read-only. Reset value 0b00000.

Table 3-33 FCLK_DIV Register bit assignments (continued)

Bits	Name	Function
[15:5]	-	Reserved.
[4:0]	FCLKDIV	Controls FCLKDIV division value in SSE-200 subsystem: Division value = FCLKDIV+1. These bits are read/write. Reset value `0b00000`, no division.

SYSCLK_DIV Register

The SYSCLK_DIV Register characteristics are:

Purpose

Controls the divider value of clock divider SYSCLKDIV that derives **SYSCLK** from **FCLK** in the Musca-S1 test chip. **SYSCLK** drives the primary processor element, CPU0.

Usage constraints

Bits[20:16] are read-only. Bits[4:0] are read/write. The other bits are reserved.

Memory offset and full register reset value

See *3.5.3 System Control Register Block* on page 3-97.

The following table shows the bit assignments.

Table 3-34 SYSCLK_DIV Register bit assignments

Bits	Name	Function
[31:21]	-	Reserved.
[20:16]	SYSCLKDIV_CUR	Current value of SYSCLKDIV: The division value of SYSCLKDIV divider is SYSCLKDIV_CUR+1. These bits are read-only. Reset value `0b00011`.
[15:5]	-	Reserved.
[4:0]	SYSCLKDIV	Controls SYSCLKDIV division value in SSE-200 subsystem: Division value = SYSCLKDIV+1. These bits are read/write. Reset value `0b00011`.

3.5.4 CMSDK timer

The system control element implements a CMSDK watchdog timer running on the **S32KCLK** clock.

The base memory addresses of the control registers of the CMSDK timer in the system control element are:

- `0x4002_F000` in the Non-secure region.
- `0x5002_F000` in the Secure region.

The system control element APB *Peripherals Protection Controller* (PPC) determines the region in which the timer resides.

See *3.4.3 CMSDK timers* on page 3-61 for a summary of the CMSDK timer control registers.

See *Arm® Cortex®-M System Design Kit Technical Reference Manual* for full descriptions of the CMSDK timer control registers.

3.5.5 CMSDK watchdog timer

The system control element implements a CMSDK watchdog timer running on the **S32KCLK** clock.

The CMSDK watchdog timer in the system control element is mapped to the secure region only. The base memory address is:

- `0x5002_E000` in the secure region.

The system control element APB Peripherals Protection Controller (PPC) determines the region in which the timer resides.

See *3.4.5 CMSDK watchdog timers* on page 3-65 for a summary of the CMSDK timer control registers.

See *Arm® Cortex®-M System Design Kit Technical Reference Manual* for full descriptions of the CMSDK timer control registers.

3.6 SSE-200 subsystem debug system

The debug access interface of the SSE-200 subsystem provides access to three Debug Access Ports within the debug subsystem. The ports provide access to the System debug region of the memory map, the processor Debug Access Ports, and associated debug logic.

System debug region

The system debug region is only accessible through the debug access interface and is not visible to any other master interface in the system.

Debug access interface

The debug access interface of the subsystem provides three access ports (APs) within the debug subsystem. The following table shows the address map of the interface.

Table 3-35 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. The following table shows the memory map that can be accessed by the system APB-AP.

A CoreSight ROM is also expected at address 0xF008_0000 in the debug expansion logic which catalogs all CoreSight expansion debug components outside the subsystem which are are accessible through the debug APB expansion interface.

Table 3-36 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	SYSCROM	Debug system CoreSight ROM.
3	0xF000_1000	0xF000_1FFF	256B	SYSFUNNEL	Debug system trace funnel.
4	0xF000_2000	0xF000_2FFF	-	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	0xF010_0000	0xFFFF_FFFF	-	-	Reserved.

CPU0 AHB-AP is for CPU0, primary core, debug access and also for certification access. It also maps a CoreSight ROM and a *Granular Power Requester* (GPR).

The values of CERTDISABLE, CERTDISABLED, CERTREADEN, and CERTREADENABLED control the accessibility of the certification access path.

The address map for CPU0 depends on the value of CERTDISABLED.

Note

CERTDISABLE, CERTDISABLED, CERTREADEN, and CERTREADENABLED are indicated by the register SCSECCTRL. See *3.5.3 System Control Register Block* on page 3-97 and the *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)*.

The following table shows the map for CPU0 when CERTDISABLED is LOW.

Table 3-37 CPU0 AHB-AP address map when CERTDISABLED 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 processor 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 *Granular Power Requester* (GPR)
4	0xF000_A000	0xFFFF_FFFF	-	-	System memory access by the CPU0 Debug Access Port.

The following table shows the map for CPU0 when CERTDISABLED is HIGH.

Table 3-38 CPU0 AHB-AP address map when CERTDISABLED 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	CPU0CSCROM	CPU0 access CoreSight ROM.
3	0xF000_9000	0xF000_9FFF	4KB	CPU0GPR	CPU0 *Granular Power Requester* (GPR).
4	0xF000_A000	0xFFFF_FFFF	-	-	System memory access by the CPU0 Debug Access Port.

CPU1 AHB-AP is for CPU1, secondary core, debug access. It also maps a CoreSight ROM and a *Granular Power Requester* (GPR).

The following table shows the memory map for CPU1 AHB-AP.

Table 3-39 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	CPU1SCROM	CPU1 Access CoreSight ROM.
3	0xF000_9000	0xF000_9FFF	4KB	CPU1GPR	CPU1 *Granular Power Requester* (GPR).
4	0xF000_A000	0xFFFF_FFFF	-	-	System memory access by the CPU1 Debug Access Port.

Security violations from debug memory accesses to system memory through CPU0 AHB-AP or CPU1 AHB-AP are blocked in a similar way to a non-debug failed access. However, in these cases, the PPC or MPC do not raise an interrupt for these failed accesses.

Debug system CoreSight™ ROM

The debug system CoreSight ROM is only accessible through the debug System Access APB-AP and is located at address `0xF000_0000`.

The following table shows the CoreSight ROM in address offset order from the base memory address. Undefined locations are reserved. Software must not attempt to read from these locations.

Table 3-40 Debug System CoreSight ROM

Offset	Name	Access	Value	Description
`0x000`	Entry 0	RO	`0x0000_1003`	ROM entry that points to debug System trace funnel.
`0x004`	Entry 1	RO	`0x0000_2003`	ROM entry that points to debug System Cross Trigger Interface.
`0x008`	Entry 2	RO	`0x0008_0003`	ROM entry that points to an external ROM on the APB expansion interface.
`0xFCC`	MEMTYPE	RO	`0x0000_0000`	MEMTYPE register.
`0xFD0`	PIDR4	RO	`0x0000_0004`	Peripheral ID 4. PIDR4[3:0]: JEP106 continuation code which is set by **TARGETIDSYS[11:8]**.
`0xFE0`	PIDR0	RO	`0x0000_0043`	Peripheral ID 0. PIDR0[7:0]: Part number [7:0] which is set by **TARGETIDSYS[23:16]**.
`0xFE4`	PIDR1	RO	`0x0000_00B7`	Peripheral ID 1. PIDR1[3:0]: Part number [11:8] which is set by **TARGETIDSYS[27:24]**. PIDR1[7:4], JEP106 identity code [3:0] which is set by **TARGETIDSYS[4:1]**.
`0xFE8`	PIDR2	RO	`0x0000_000B`	Peripheral ID 2. PIDR2[2:0]: JEP106 identity code [6:4] which is set by **TARGETIDSYS[7:5]**. PIDR2[3]: JEDEC identifier. PIDR2[7:4]: Revision code which is set by **TARGETIDSYS[31:28]**.
`0xFF0`	CIDR0	RO	`0x0000_000D`	Component ID 0.
`0xFF4`	CIDR1	RO	`0x0000_0010`	Component ID 1.
`0xFF8`	CIDR2	RO	`0x0000_0005`	Component ID 2.
`0xFFC`	CIDR3	RO	`0x0000_00B1`	Component ID 3.

Note

The PID values are dependent on the **TARGETIDSYS[31:0]** static configuration signal value. This is set to `0x0743_0477` by default. See *Arm® CoreLink™ SSE-200 Subsystem for Embedded Technical Reference Manual (r1p0)*.

CPU Access CoreSight™ ROM

There are two processor system CoreSight ROMs, one for each processor. Each ROM is accessible through its associated processor access AHB-P at address `0xF000_8000`. The two ROMs are identical and each one has the following contents.

Table 3-41 CPU Access CoreSight ROM

Offset	Name	Access	Value	Description
`0x000`	Entry 0	RO	`0x0000_1003`	ROM entry that points to the *Granular Power Controller* (GPC).
`0x004`	Entry 1	RO	`0x0000_2003`	ROM entry that points to the internal ROM table of the processor.
`0xFCC`	MEMTYPE	RO	`0x0000_0000`	MEMTYPE register.
`0xFD0`	PIDR4	RO	`0x0000_0004`	Peripheral ID 4. PIDR4[3:0]: JEP106 continuation code.
`0xFE0`	PIDR0	RO	`0x0000_0043`	Peripheral ID 0. PIDR0[7:0]: Part number [7:0].
`0xFE4`	PIDR1	RO	`0x0000_00B7`	Peripheral ID 1. PIDR1[3:0]: Part number [11:8]. PIDR1[7:4], JEP106 identity code [3:0].
`0xFE8`	PIDR2	RO	`0x0000_000B`	Peripheral ID 2. PIDR2[2:0]: JEP106 identity code [6:4]. PIDR2[3]: JEDEC identifier. PIDR2[7:4]: Revision code.
`0xFF0`	CIDR0	RO	`0x0000_000D`	Component ID 0.
`0xFF4`	CIDR1	RO	`0x0000_0010`	Component ID 1.
`0xFF8`	CIDR2	RO	`0x0000_0005`	Component ID 2.
`0xFFC`	CIDR3	RO	`0x0000_00B1`	Component ID 3.

3.7 Real Time Clock

The Musca-S1 test chip implements an Arm PrimeCell Real Time Clock.

The base memory addresses of the *Real Time Clock* (RTC) control registers are:

- 0x4010_8000 in the Non-secure region.
- 0x5010_8000 in the Secure region.

See the *Arm® PrimeCell Real Time Clock (PL031) Technical Reference Manual*.

Caution

Warm reset of the Musca-S1 test chip resets the Real Time Clock.

The following table shows the Real Time Clock registers in the Musca-S1 test chip in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-42 Real Time Clock control registers summary

Offset	Name	Type	Reset	Width	Function
0x0000	RTCDR	RO	0x0000_0000	32	Data register.
0x0004	RTCMR	RW	0x0000_0000	32	Match register.
0x0008	RTCLR	RW	0x0000_0000	8	Load register.
0x000C	RTCCR	RW	0x0000_0000	32	Control register.
0x0010	RTCIMSC	RW	0x0000_0000	1	Interrupt mask set and clear register.
0x0014	RTCRIS	RO	0x0000_0000	1	Raw interrupt status register.
0x0018	RTCMIS	RO	0x0000_0000	32	Masked interrupt status register.
0x001C	RTCICR	WO	0x0000_0000	32	Interrupt clear register.
0x0FE0	RTCPeriphID0	RO	0x0000_0031	8	Peripheral ID register bits [7:0]
0x0FE4	RTCPeriphID1	RO	0x0000_0010	8	Peripheral ID register bits [15:8]
0x0FE8	RTCPeriphID2	RO	0x0000_0004	8	Peripheral ID register bits [23:16]
0x0FEC	RTCPeriphID3	RO	0x0000_0000	8	Peripheral ID register bits [31:24]
0x0FF0	RTCPCellID0	RO	0x0000_000D	8	PrimeCell ID register bits [7:0]
0x0FF4	RTCPCellID1	RO	0x0000_00F0	8	PrimeCell ID register bits [15:8]
0x0FF8	RTCPCellID2	RO	0x0000_0005	8	PrimeCell ID register bits [23:16]
0x0FFC	RTCPCellID3	RO	0x0000_00B1	8	PrimeCell ID register bits [31:24]

3.8 General-purpose timer

The Musca-S1 test chip implements a general-purpose timer (GPT) in the **32K** domain.

The base memory addresses of the general-purpose timer control registers are:

- `0x4010_C000` in the Non-secure region.
- `0x5010_C000` in the Secure region.

The following table shows the general-purpose timer registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-43 General-purpose timer control registers summary

Offset	Name	Type	Reset	Width	Function
`0x0000`	GPTRESET	RO	`0x0000_0000`	32	Reset Control Register. See *3.8.1 GPTRESET Register* on page 3-106.
`0x0004`	GPTINTM	RW	`0x0000_0000`	32	Masked interrupt status register. See *3.8.2 GPTINTM Register* on page 3-107.
`0x0008`	GPTINTC	RW	`0x0000_0000`	8	Interrupt clear register. See *3.8.3 GPTINTC Register* on page 3-107.
`0x0010`	GPTALARM0	RW	`0x0000_0000`	32	ALARM0 data value register. See *3.8.4 GPTALARM0 Register* on page 3-108.
`0x0014`	GPTALARM1	RW	`0x0000_0000`	1	ALARM1 data value register. See *3.8.5 GPTALARM1 Register* on page 3-108.
`0x0018`	GPTINTR	RO	`0x0000_0000`	1	Raw interrupt status register. See *3.8.6 GPTINTR Register* on page 3-109.
`0x001C`	GPTCOUNTER	RO	`0x0000_0000`	32	Counter data value register. See *3.8.7 GPTCOUNTER Register* on page 3-109.

This section contains the following subsections:

3.8.1 GPTRESET Register

The GPTRESET Register characteristics are:

Purpose

- A write resets the general-purpose timer counter to 1.
- A read returns the current value of the general-purpose timer counter.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTRESET Register.

Table 3-44 GPTRESET Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved.
[0]	GPTRESET	CPU0 interrupt status. Software reset of the timer counter: `0b0`: No effect. `0b1`: Software reset. Reset value `0b0`.

3.8.2 GPTINTM Register

The GPTINTM Register characteristics are:

Purpose

- Writing 1 to the relevant bit enables the ALARM0 or ALARM1 interrupt.
- Reading the relevant bit gives the current masked status value of the corresponding interrupt.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTINTM Register.

Table 3-45 GPTINTM Register bit assignments

Bits	Name	Function
[31:2]	-	Reserved.
[1:0]	GPTINTM	Current masked status of the interrupt. Writing `0b1` enables the ALARM[n] interrupt: `0b0`: No effect. `0b1`: Enable ALARM[n] interrupt. Bit[1] = ALARM1 interrupt. Bit[0]=ALARM0 interrupt. Reset value `0b00`.

3.8.3 GPTINTC Register

The GPTINTC Register characteristics are:

Purpose

- Writing 1 to the relevant bit clears the ALARM0 or ALARM1 interrupt.
- Reading a bit returns the current value of the bit.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTINTC Register.

Table 3-46 GPTINTC Register bit assignments

Bits	Name	Function
[31:2]	-	Reserved.
[1:0]	GPTINTC	Writing `0b1` disables the ALARM[n] interrupt: `0b0`: No effect. `0b1`: Clear interrupt. Bit[1] = ALARM1 interrupt. Bit[0]=ALARM0 interrupt. Reset value `0b00`.

3.8.4 GPTALARM0 Register

The GPTALARM0 Register characteristics are:

Purpose

- The ALARM0 data value register, GPTALARM0 stores the 32-bit value that triggers the interrupt when the counter reaches that value.
- Reading the register returns the trigger value.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTALARM0 Register.

Table 3-47 GPTALARM0 Register bit assignments

Bits	Name	Function
[31:0]	GPTALARM0_DATA	Value that triggers the ALARM0 interrupt when the counter reaches that value. Reset value `0x0000_0000`.

3.8.5 GPTALARM1 Register

The GPTALARM1 Register characteristics are:

Purpose

- The ALARM1 data value register, GPTALARM1 stores the 32-bit value that triggers the interrupt when the counter reaches that value.
- Reading the register returns the trigger value.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTALARM1 Register.

Table 3-48 GPTALARM1 Register bit assignments

Bits	Name	Function
[31:0]	GPTALARM1_DATA	Value that triggers the ALARM1 interrupt when the counter reaches that value. Reset value `0x0000_0000`.

3.8.6 GPTINTR Register

The GPTINTR Register characteristics are:

Purpose

- The raw interrupt status register, GPTINTR, stores the current raw status value of the corresponding interrupt before masking.
- Reading the register returns the trigger value.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTINTR Register.

Table 3-49 GPTINTR Register bit assignments

Bits	Name	Function
[31:1]	-	Reserved.
[2:0]	GPTINTR	Raw interrupt state, before masking, of the GPTINTR interrupt. • Bit[0]: ALARM0 interrupt status. • Bit[1]: ALARM1 interrupt status. • Bit[2]: Or-ed ALARM0 and ALARM1 interrupt status. Reset value `0b000`.

3.8.7 GPTCOUNTER Register

The GPTCOUNTER Register characteristics are:

Purpose

- The counter data value register, GPTCOUNTER, stores the current 32-bit value of the general-purpose timer counter.
- Reading the register returns the trigger value.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.8 General-purpose timer* on page 3-106.

The following table shows the bit assignments of the GPTCOUNTER Register.

Table 3-50 GPTCOUNTER Register bit assignments

Bits	Name	Function
[31:0]	GPTCOUNTER	Current value of 32-bit timer counter. Reset value `0000_0000`.

3.9 PVT sensor registers

The Musca-S1 test chip implements registers that monitor the *Process, Voltage, Temperature* (PVT) sensor system.

This section contains the following subsection:

3.9.1 PVT sensor control registers summary

The PVT sensor control registers are mapped to both the Non-secure and Secure Expansion 1 regions.

The base memory addresses of the PVT sensor control registers are:

- `0x4010_9000` in the Non-secure region.
- `0x5010_9000` in the Secure region.

The following table shows the PVT sensor control registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-51 PVT sensor control registers summary

Offset	Name	Type	Reset	Width	Description
`0x0000`	CTRL_REF_CNTR	RW	`0x0000_0000`	32	See *CTRL_REF_COUNTER Register* on page 3-111.
`0x0004`	CRTL_ENABLE	RW	`0x0000_0000`	32	See *CTRL_ENABLE Register* on page 3-113.
`0x0008`	CTRL_AUTOCLEAR	RW	`0xFFFF_FFFF`	32	See *CTRL_AUTOCLEAR Register* on page 3-113.
`0x000C`	CTRL_CLKSEL	RW	`0x0000_0000`	32	See *CTRL_CLKSEL Register* on page 3-113.
`0x0010`	CTRL_SAMPLE	RW	`0x0000_0000`	32	See *CTRL_SAMPLE Register* on page 3-114.
`0x0014`	CTRL_PERIOD	RW	`0x0000_00FF`	32	See *CTRL_PERIOD Register* on page 3-114.
`0x0018`	OVERFLOW-STATUS	RO	`0x0000_0000`	32	See *OVERFLOW_STATUS Register* on page 3-115.
`0x001C`	INTR_STATUS	RO	`0x0000_0000`	32	See *INTR_STATUS Register* on page 3-115.
`0x0020`	CLEARED_STATUS	RO	`0x0000_0000`	32	See *CLEARED_STATUS Register* on page 3-116.
`0x0024`	SAMPLED_STATUS	RO	`0x0000_0000`	32	See *SAMPLED_STATUS Register* on page 3-116.
`0x0028`	COUNTER_STATUS	RO	`0x0000_0000`	32	See *COUNTER_STATUS Register* on page 3-117.
`0x0080`	SENSOR0_VAL	RO	`0x0000_0001`	32	See *SENSOR0_VAL Register* on page 3-117.
`0x0084`	SENSOR1_VAL	RO	`0x0000_0000`	32	See *SENSOR1_VAL Register* on page 3-117.
`0x0088`	SENSOR2_VAL	RO	`0x0000_0000`	32	See *SENSOR2_VAL Register* on page 3-118.

CTRL_REF_COUNTER Register

The CTRL_REF_COUNTER Register characteristics are:

Purpose

Controls the PVT sensors reference counter.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CTRL_REF_COUNTER Register bit assignments.

Table 3-52 CTRL_REF_COUNTER Register bit assignments

Bits	Name	Function
[31:19]	-	Reserved.
[18]	CLEAR_SAMPLED_VAL	Clear sensors sampled flags: `0b0`: No effect. `0b1`: Clear sensors sampled flags. Reset value `0b0`.
[17]	CLEAR_OVERFLOW	Clear sensors overflows: `0b0`: No effect. `0b1`: Clear overflows: Reset value `0b0`.
[16]	CLEAR_CNTR	Clear sensors counters: `0b0`: No effect. `0b1`: Clear counters: Reset value `0b0`.
[15:4]	-	Reserved.
[3]	CTRL_IRQ_CLEAR	Clear PVT interrupt: `0b0`: No effect. `0b1`: Clear interrupt: Reset value `0b0`.
[2]	CTRL_IRQ_EN	Enable PVT interrupt: `0b0`: No effect. `0b1`: Clear interrupt: Reset value `0b0`.
[1]	CTRL_AUTORESTART_EN	Select operating mode of PVT sensors: `0b0`: One-shot mode. `0b1`: Repeat mode: Reset value `0b0`.
[0]	CTR_CNTR_EN	Enable reference counter: `0b0`: Not enabled. `0b1`: Enabled: Reset value `0b0`.

CTRL_ENABLE Register

The CTRL_ENABLE Register characteristics are:

Purpose
Individually enables or disables the nine PVT sensors.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CTRL_ENABLE Register bit assignments.

Table 3-53 CTRL_ENABLE Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	CTRL_ENABLE[2:0]	PVT sensors Enable input: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0x000`.

CTRL_AUTOCLEAR Register

The CTRL_AUTOCLEAR Register characteristics are:

Purpose
Individually enables the nine PVT sensors autoclear functions.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CTRL_AUTOCLEAR Register bit assignments.

Table 3-54 CTRL_AUTOCLEAR Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	CTRL_AUTOCLEAR[2:0]	Enable or disable PVT sensors autoclear functions: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0x1FF`.

CTRL_CLKSEL Register

The CTRL_CLKSEL Register characteristics are:

Purpose
Individually selects the nine PVT sensors input clocks.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CTRL_CLKSEL Register bit assignments.

Table 3-55 CTRL_CLKSEL Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	CTRL_CLKSEL[2:0]	Selects the PVT sensor input clocks: `0b0`: Ring oscillator clock. `0b1`: **REFCLK**. Reset value `0x000`.

CTRL_SAMPLE Register

The CTRL_SAMPLE Register characteristics are:

Purpose
Individually initiates the PVT measurements.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CTRL_SAMPLE Register bit assignments.

Table 3-56 CTRL_SAMPLE Register bit assignments

Bits	Name	Function
[31:3]		Reserved.
[2:0]	CTRL_SAMPLE[2:0]	Initiate PVT measurements: `0b0`: No effect. `0b1`: Initiate measurement. Reset value `0x000`.

CTRL_PERIOD Register

The CTRL_PERIOD Register characteristics are:

Purpose
Stores the reference counter period in **REFCLK** cycles.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CTRL_PERIOD Register bit assignments.

Table 3-57 CTRL_PERIOD Register bit assignments

Bits	Name	Function
[31:0]	CTRL_PERIOD	Reference counter period in **REFCLK** periods: Reset value `0x0000_00FF`.

OVERFLOW_STATUS Register

The OVERFLOW_STATUS Register characteristics are:

Purpose
Individually indicates the PVT sensors overflow status.

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the OVERFLOW_STATUS Register bit assignments.

Table 3-58 OVERFLOW_STATUS Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	OVERFLOW_STATUS[2:0]	Indicates the PVT sensors overflow status: `0b0`: No overflow. `0b1`: Overflow: Reset value `0x000`.

INTR_STATUS Register

The INTR_STATUS Register characteristics are:

Purpose
Indicates the reference counter interrupt status.

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the INTR_STATUS bit assignments.

Table 3-59 INTR_STATUS Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	INTR_STATUS[2:0]	Indicates the reference counter interrupt status: `0b0`: No interrupt. `0b1`: Interrupt: Reset value `0b0`.

CLEARED_STATUS Register

The CLEARED_STATUS Register characteristics are:

Purpose
Stores the cleared statuses of the sensors.

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the CLEARED_STATUS Register bit assignments.

Table 3-60 CLEARED_STATUS Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	CLEARED_STATUS[2:0]	Stores the cleared statuses of the sensors: `0b0`: Not cleared. `0b1`: Cleared. Reset value `0x000`.

SAMPLED_STATUS Register

The SAMPLED_STATUS Register characteristics are:

Purpose
Individually indicates that PVT measurements are valid.

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the SAMPLED_STATUS Register bit assignments.

Table 3-61 SAMPLED_STATUS Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2:0]	SAMPLED_STATUS[2:0]	Indicates that PVT measurements are valid: `0b0`: Not valid. `0b1`: Valid. Reset value `0x000`.

COUNTER_STATUS Register

The COUNTER_STATUS Register characteristics are:

Purpose
Stores the current value of the reference counter.

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the COUNTER_STATUS Register bit assignments.

Table 3-62 COUNTER_STATUS Register bit assignments

Bits	Name	Function
[31:0]	COUNTER_STATUS	Stores the current value of the reference counter. Reset value `0x0000_0000`.

SENSOR0_VAL Register

The SENSOR0_VAL Register characteristics are:

Purpose
Stores the value measured by PVT sensor 0.

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the SENSOR0_VAL Register bit assignments.

Table 3-63 SENSOR0_VAL Register bit assignments

Bits	Name	Function
[31:0]	SENSOR0_VAL	Value measured by PVT sensor 0. Reset value `0x0000_0000`.

SENSOR1_VAL Register

The SENSOR1_VAL Register characteristics are:

Purpose

Stores the value measured by PVT sensor 1.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the SENSOR1_VAL Register bit assignments.

Table 3-64 SENSOR1_VAL Register bit assignments

Bits	Name	Function
[31:0]	SENSOR1_VAL	Value measured by PVT sensor 1. Reset value `0x0000_0000`.

SENSOR2_VAL Register

The SENSOR2_VAL Register characteristics are:

Purpose

Stores the value measured by PVT sensor 2.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.9.1 PVT sensor control registers summary* on page 3-111.

The following table shows the SENSOR2_VAL Register bit assignments.

Table 3-65 SENSOR2_VAL Register bit assignments

Bits	Name	Function
[31:0]	SENSOR2_VAL	Value measured by PVT sensor 2. Reset value `0x0000_0000`.

3.10 CryptoCell™-312 and *One-Time Programmable* (OTP) secure memory locations

The Musca-S1 test chip implements an Arm CryptoCell-312 (r1p0) security subsystem and emulates 1KB of *One Time Programming* (OTP) secure memory.

See *2.6 CryptoCell™-312 and One Time Programmable security system* on page 2-32 for more information.

The base memory addresses of CryptoCell-312 are:

- 0x4008_8000 in the Non-secure region.
- 0x5008_8000 in the Secure region.

The base memory addresses of the emulated OTP memory are:

- 0x0E00_8000 in the Non-secure region.
- 0x1E00_8000 in the Secure region.

OTP user areas are:

- 0x0E00_0000 in the Non-secure region.
- 0x1E00_0000 in the Secure region.

Contact your Arm representative for information on CryptoCell-312 and the emulated OTP memory.

Related information

2.6 CryptoCell™-312 and One Time Programmable security system on page 2-32

3.11 Serial Configuration Control registers

The *Serial Configuration Control* (SCC) registers contain the initial settings of blocks before bootup. Write and read accesses to the registers during runtime enable software to alter and to read the block settings.

This section contains the following subsections:

3.11.1 IOMUX registers

The IOMUX registers, which are a subset of the SCC register bank, control the multiplexer logic that drives Musca-S1 test chip I/O pins PA26-PA0.

The multiplexer controlsMusca-S1 test chip I/O PA26-PA0. The following figure shows the multiplexer logic.

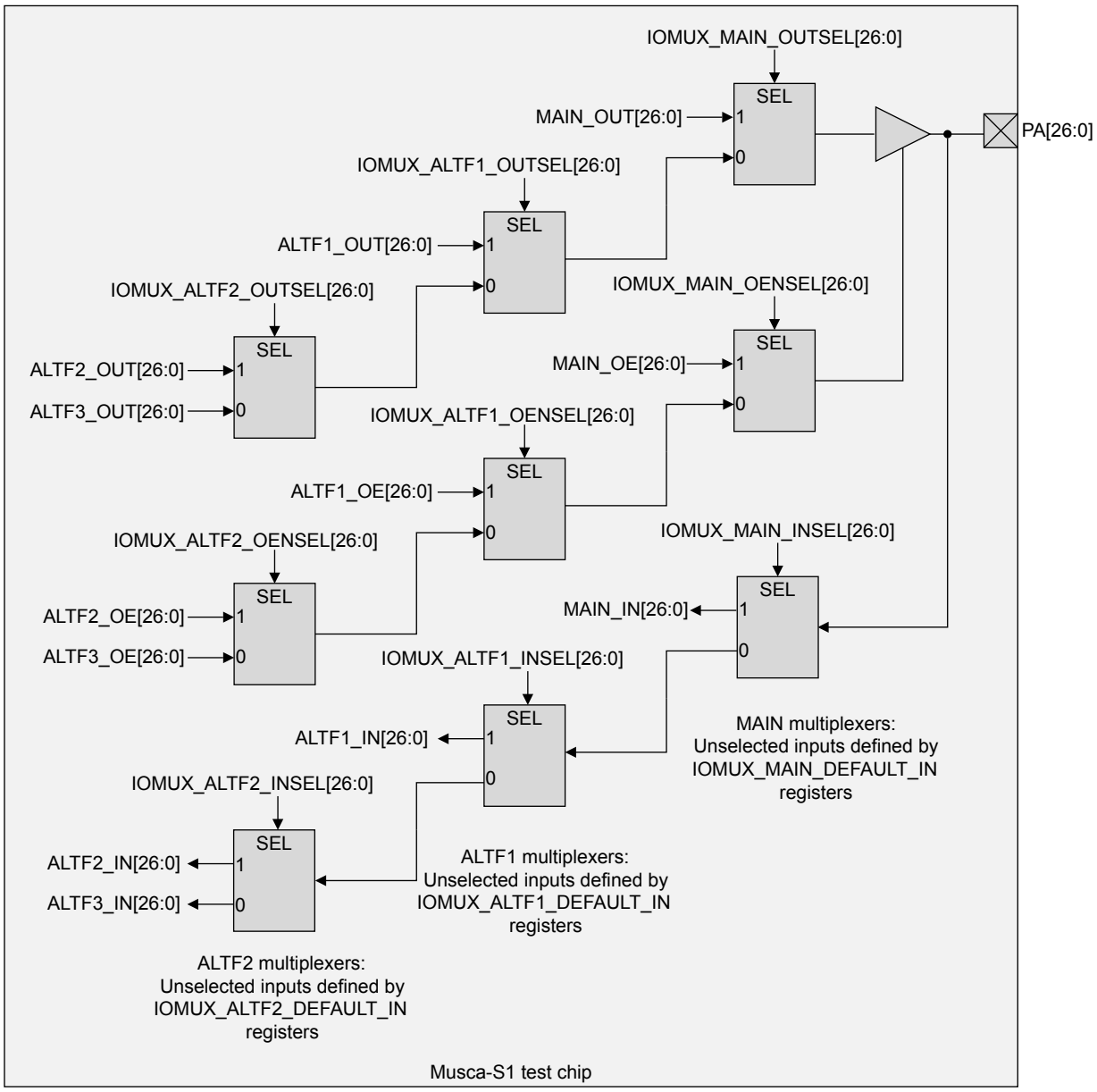

Figure 3-6 Test chip I/O multiplexer logic

The IOMUX registers control the IOMUX multiplexer logic. The following table shows the IOMUX registers in offset order from the SCC base memory address of `0x4010_C000` in the Non-secure region, or `0x5010_C000` in the Secure region.

Note

See *3.11.2 SCC registers summary* on page 3-123 for the read/write access characteristics.

Table 3-66 IOMUX registers

Offset	Register	Register function	Register description
0x0868	IOMUX_MAIN_INSEL	ControlsMusca-S1 test chip I/O PA31-PA0. ConnectsMusca-S1 test chip input to either MAIN_IN or ALTF1.	*IOMUX_MAIN_INSEL Register* on page 3-146
0x0870	IOMUX_MAIN_OUTSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Connects either MAIN_OUT or ALTF1 toMusca-S1 test chip output.	*IOMUX_MAIN_OUTSEL Register* on page 3-146
0x0878	IOMUX_MAIN_OENSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Selects either MAIN_OE or ALTF1_OENSEL asMusca-S1 test chip output enable signal.	*IOMUX_MAIN_OENSEL Register* on page 3-147
0x0088	IOMUX_MAIN_DEFAULT_IN	Controls the Musca-S1 test chip I/O PA31-PA0. Drives unselected outputs of MAIN input multiplexers to defined logic levels to prevent floating nodes.	*IOMUX_MAIN_DEFAULT_IN Register* on page 3-148
0x0888	IOMUX_ALTF1_INSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Routes connection from MAIN input multiplexer to either ALTF1_IN or ALTF2.	*IOMUX_ALTF1_INSEL Register* on page 3-148
0x0890	IOMUX_ALTF1_OUTSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Connects either ALTF1_OUT or ALTF2 to MAIN output multiplexer.	*IOMUX_ALTF1_OUTSEL Register* on page 3-149
0x0898	IOMUX_ALTF1_OENSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Connects either ALTF1_OE or ALTF2 to MAIN_OESEL multiplexer.	*IOMUX_ALTF1_OENSEL Register* on page 3-150
0x08A0	IOMUX_ALTF1_DEFAULT_IN	Controls the Musca-S1 test chip I/O PA31-PA0. Drives unselected outputs of ALTF1 input multiplexers to defined logic levels to prevent floating nodes.	*IOMUX_ALTF1_DEFAULT_IN Register* on page 3-150
0x08A8	IOMUX_ALTF2_INSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Routes connection from ALTF1 input multiplexers to either ALTF2_IN or ALTF3_IN.	*IOMUX_ALTF2_INSEL Register* on page 3-151

Table 3-66 IOMUX registers (continued)

Offset	Register	Register function	Register description
0x08B0	IOMUX_ALTF2_OUTSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Connects either ALTF1_OUT or ALTF3_OUT to ALTF1 output data multiplexer.	*IOMUX_ALTF2_OUTSEL Register* on page 3-152
0x08B8	IOMUX_ALTF2_OENSEL	Controls the Musca-S1 test chip I/O PA31-PA0. Connects either ALTF2_OE or ALTF3_OE to ALTF1_OENSEL multiplexer.	*IOMUX_ALTF2_OENSEL Register* on page 3-152
0x08C0	IOMUX_ALTF2_DEFAULT_IN	Controls the Musca-S1 test chip I/O PA32-PA0. Drives unselected outputs of ALTF2 input multiplexers to defined logic levels to prevent floating nodes.	*IOMUX_ALTF2_DEFAULT_IN Register* on page 3-153

See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the ALTF1 and ALTF2 pin functions.

Related information

2.2.2 Test chip multiplexed I/O on page 2-23

3.11.2 SCC registers summary

The base memory address of the Serial Configuration Control (SCC) registers are mapped to both the Secure and Non-secure regions of the Expansion 1 region.

The base memory addresses of the SCC registers are:

- 0x4010_C000 in the Non-secure region.
- 0x5010_C000 in the Secure region.

The following table shows the registers in offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-67 SCC registers summary

Offset	Name	Type	Reset	Width	Description
0x0800	CLK_CTRL_SEL	RW	0x0000_0062	32	See *CLK_CTRL_SEL Register* on page 3-126.
0x0804	CLK_PLL_PREDIV_CTRL	RW	0x0000_0000	32	See *CLK_PLL_PREDIV_CTRL Register* on page 3-128.
0x0808	CLK_BBGEN_DIV_CLK	RW	0x0000_0028	32	See *CLK_BBGEN_DIV_CLK Register* on page 3-129.
0x0810	CLK_POSTDIV_QSPI	RW	0x0000_0000	32	See *CLK_POSTDIV_CTRL_QSPI Register* on page 3-129.
0x0814	CLK_POSTDIV_RTC	RW	0x0000_7FFF	32	See *CLK_POSTDIV_CTRL_RTC Register* on page 3-130.
0x081C	CLK_POSTDIV_TEST	RW	0x0000_000A	32	See *CLK_POSTDIV_CTRL_TEST Register* on page 3-130.
0x0820	CTRL_BYPASS_DIV	RW	0x0000_0001	32	See *CTRL_BYPASS_DIV Register* on page 3-131 .

Table 3-67 SCC registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x0824	PLL_CTRL_PLL0_CLK	RW	0x0010_05F4	32	See *PLL_CTRL_PLL0_CLK Register* on page 3-132.
0x0830	CLK_CTRL_ENABLE	RW	0x0000_FFFF	32	See *CLK_CTRL_ENABLE Register* on page 3-133.
0x0834	CLK_STATUS	RW	0x0000_001F	32	See *CLK_STATUS Register* on page 3-135.
0x0840	RESET_CTRL	RW	0xFFFF_FFFF	32	See *RESET_CTRL Register* on page 3-135.
0x0848	DBG_CTRL	RW	0x0000_001F	32	See *DBG_CTRL Register* on page 3-138.
0x084C	SRAM_CTRL	RW	0x0000_0000	32	See *SRAM_CTRL Register* on page 3-139.
0x0850	INTR_CTRL	RW	0x0000_0000	32	See *INTR_CTRL Register* on page 3-144.
0x0858	CPU0_VTOR	RW	0x1020_0000	32	See *CPU0_VTOR Register* on page 3-144.
0x085C	CPU0_VTOR_1	RW	0x1A00_0000	32	See *CPU0_VTOR_1 Register* on page 3-145.
0x0860	CPU1_VTOR	RW	0x1020_0000	32	See *CPU1_VTOR Register* on page 3-145.
0x0864	CPU1_VTOR_1	RW	0x1A00_0000	32	See *CPU1_VTOR_1 Register* on page 3-145.
0x0868	IOMUX_MAIN_INSEL	RW	0xFFFF_FFFF	32	See *IOMUX_MAIN_INSEL Register* on page 3-146 and *3.11.1 IOMUX registers* on page 3-120.
0x0870	IOMUX_MAIN_OUTSEL	RW	0xFFFF_FFFF	32	See *IOMUX_MAIN_OUTSEL Register* on page 3-146 and *3.11.1 IOMUX registers* on page 3-120.
0x0878	IOMUX_MAIN_OENSEL	RW	0xFFFF_FFFF	32	See *IOMUX_MAIN_OENSEL Register* on page 3-147 and *3.11.1 IOMUX registers* on page 3-120.
0x0880	IOMUX_MAIN_DEFAULT_IN	RW	0x0000_0000	32	See *IOMUX_MAIN_DEFAULT_IN Register* on page 3-148 and *3.11.1 IOMUX registers* on page 3-120.
0x0888	IOMUX_ALTF1_INSEL	RW	0x0000_0000	32	See *IOMUX_ALTF1_INSEL Register* on page 3-148 and *3.11.1 IOMUX registers* on page 3-120.
0x0890	IOMUX_ALTF1_OUTSEL	RW	0xFFFF_FFFF	32	See *IOMUX_ALTF1_OUTSEL Register* on page 3-149 and *3.11.1 IOMUX registers* on page 3-120.
0x0898	IOMUX_ALTF1_OENSEL	RW	0xFFFF_FFFF	32	See *IOMUX_ALTF1_OENSEL Register* on page 3-150 and *3.11.1 IOMUX registers* on page 3-120.
0x08A0	IOMUX_ALTF1_DEFAULT_IN	RW	0x0000_0000	32	See *IOMUX_ALTF1_DEFAULT_IN Register* on page 3-150 and *3.11.1 IOMUX registers* on page 3-120.
0x08A8	IOMUX_ALTF2_INSEL	RW	0x0000_0000	32	See *IOMUX_ALTF2_INSEL Register* on page 3-151 and *3.11.1 IOMUX registers* on page 3-120.
0x08B0	IOMUX_ALTF2_OUTSEL	RW	0xFFFF_FFFF	32	See *IOMUX_ALTF2_OUTSEL Register* on page 3-152 and *3.11.1 IOMUX registers* on page 3-120.

Table 3-67 SCC registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x08B8	IOMUX_ALTF2_OENSEL	RW	0xFFFF_FFFF	32	See *IOMUX_ALTF2_OENSEL Register* on page 3-152 and *3.11.1 IOMUX registers* on page 3-120.
0x08C0	IOMUX_ALTF2_DEFAULT_IN	RW	0x0000_0000	32	See *IOMUX_ALTF2_DEFAULT_IN Register* on page 3-153 and *3.11.1 IOMUX registers* on page 3-120.
0x08E8	IOPAD_DS0	RW	0x03F0_0000	32	See *IOPAD_DS0 and IOPAD_DS1 Registers* on page 3-154.
0x08F0	IOPAD_DS1	RW	0x000F_FFFF	32	*IOPAD_DS0 and IOPAD_DS1 Registers* on page 3-154
0x08F8	IOPAD_PE	RW	0xFFFF_FFFF	32	See *IOPAD_PE Register* on page 3-155.
0x0900	IOPAD_PS	RW	0xF81F_FFFF	32	See *IOPAD_PS Register* on page 3-155.
0x0908	IOPAD_SR	RW	0xFFFF_FFFF	32	See *IOPAD_SR Registers* on page 3-156.
0x0910	IOPAD_IS	RW	0xFFFF_FFFF	32	See *IOPAD_IS Registers* on page 3-156.
0x0930	SPARE0	RW	0x0000_0000	32	See *SPARE0 Register* on page 3-156.
0x093C	STATIC_CONF_SIG1	RW	0x0000_0000	32	See *STATIC_CONF_SIG1 Register* on page 3-157.
0x0998	SCC_MRAM_CTRL0	RW	0x4014_4003	32	See *SCC_MRAM_CTRL0 Register* on page 3-158.
0x099C	SCC_MRAM_CTRL1	RW	0x0000_0000	32	See *SCC_MRAM_CTRL1 Register* on page 3-161.
0x09A0	SCC_MRAM_CTRL2	RW	0xFFFF_0119	32	See *SCC_MRAM_CTRL2 Register* on page 3-161.
0x09B0	SCC_MRAM_DIN0	RW	0x0000_0000	32	See *SCC_MRAM_DIN0 Register* on page 3-162.
0x09B4	SCC_MRAM_DIN1	RW	0x0000_0000	32	See *SCC_MRAM_DIN1 Register* on page 3-163.
0x09C0	SCC_MRAM_DOUT0	RO	0x0200_F2CE	32	See *SCC_MRAM_DOUT0 Register* on page 3-163.
0x09C4	SCC_MRAM_DOUT1	RO	0x9A09_5880	32	See *SCC_MRAM_DOUT1 Register* on page 3-163.
0x09C8	SCC_MRAM_DOUT2	RO	0x0000_1296	32	See *SCC_MRAM_DOUT2 Register* on page 3-164.
0x09CC	SCC_MRAM_STATUS	RO	0x0000_000C	32	See *SCC_MRAM_STATUS Register* on page 3-164.
0x09E0	SELECTION_CONTROL_REG	RW	0x0100_0200	32	See *SELECTION_CONTROL_REG Register* on page 3-165.
0x0A20	BBGEN_CTRL	RW	0x0000_0000	32	See *BBGEN_CTRL Register* on page 3-166.
0x0A24	SPARE_CTRL1	RW	0x0000_0000	32	See *SPARE_CTRL1 Register* on page 3-167.
0x0C00	CHIP_ID	RO	0x0799_0477	32	See *CHIP_ID Register* on page 3-167.
0x0C04	IO_IN_STATUS	RO	0x05FF_FFE3	32	See *IO_IN_STATUS Register* on page 3-168.

CLK_CTRL_SEL Register

The CLK_CTRL_SEL Register characteristics are:

Purpose

Controls the clock select multiplexers.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_CTRL_SEL Register bit assignments.

Table 3-68 CLK_CTRL_SEL Register bit assignments

Bits	Name	Function
[31:13]	-	Reserved.
[12]	CTRL_PLL_MUX_CLK_SEL	Select PLL MUX input: `0b0`: PLL0 `0b1`: Not used Reset value `0b0`.

Table 3-68 CLK_CTRL_SEL Register bit assignments (continued)

Bits	Name	Function
[11:7]	CTRL_SEL_TEST_MUX_CLK[4:0]	Select TESTMUX input: `0b00000`: No output. `0b00001`: JTAG_TCK. `0b00010`: PRE_MUX_CLK. `0b00011`: SCCCLK. `0b00100`: JTAG TCK. `0b00101`: 32K. `0b00110`: REF_MUX_CLK. `0b01000`: FASTCLK. `0b01001`: PLL0_CLK. `0b01010`: PRE_MUX_CLK. `0b01011`: PRE_PLL_CLK. `0b01100`: SYSSYSUGCLK. `0b01101`: FLCLK. `0b01110`: DAPSWCLK `0b01111`: MAINCLK. `0b10000`: REFCLK `0b10001`: CLK1HZ. `0b10010`: RM38KCLK `0b10101`: QSPIPHYCLK. `0b10111`: PVT_SENSOR_OUT. `0b11000`: I2SCLK0. `0b11001`: I2SCLK1. `0b11010`: I2SCLK2. Undefined settings are reserved. Reset value `0b00000`.
[6]	SEL_RM38P4_PREMUX_CLK	Select RM38KPREMUX input: `0b0`: SYSSYSSUGCLK. `0b1`: NRM138P4 (not used). Reset value `0b1`.
[5]	SEL_SCCMUX_CLK	Select SCCMUX input: `0b0`: SCCCLK. `0b1`: PRE_MUX_CLK. Reset value `0b1`.

Table 3-68 CLK_CTRL_SEL Register bit assignments (continued)

Bits	Name	Function
[4]	SEL_RM38KMUX_CLK	Select RM38KMUX input: `0b0`: REF_MUX_CLK. `0b1`: RM38K (not used). Reset value `0b0`.
[3]	SEL_REFMUX_CLK	Select REFMUX input: `0b0`: PRE_MUX_CLK. `0b1`: PRE_PLL_CLK. Reset value `0b0`.
[2]	SEL_MAINMUX_CLK	Select MAINMUX input: `0b0`: PLL0_CLK. `0b1`: PRE_MUX_CLK. Reset value `0b0`.
[1]	SEL_DAPSWMUX_CLK	Select DAPSWMUX input: `0b0`: PRE_MUX_CLK. `0b1`: JTAG TCK. Reset value `0b1`.
[0]	SEL_PREMUX_CLK	Select PREMUX input: `0b0`: 32K. `0b1`: FASTCLK. Reset value `0b0`.

Note

Multiplexer PLL_MUX is not shown in the clock system diagram in *2.5 Clocks* on page 2-28.

CLK_PLL_PREDIV_CTRL Register

The CLK_PLL_PREDIV_CTRL Register characteristics are:

Purpose
Controls the PLL pre-divider division value.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_PLL_PREDIV_CTRL Register bit assignments.

Table 3-69 CLK_PLL_PREDIV_CTRL Register bit assignments

Bits	Name	Function
[31:10]	-	Reserved.
[9:0]	PREDIV_CTRL[9:0]	PLL0 pre-divider value. Divison value =PREDIV_CTRL+1. `0x000`: Minimum divide value =1, no division. `0x3FF`: Maximum divide value =1024. Reset value `0x000`, no division.

CLK_BBGEN_DIV_CLK Register

The CLK_BBGEN_DIV_CLK Register characteristics are:

Purpose
Controls the Body Bias Generator clock divider, BBGEN, division value.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_BBGEN_DIV_CLK Register bit assignments.

Table 3-70 CLK_BBGEN_DIV_CLK Register bit assignments

Bits	Name	Function
[31:8]	-	Reserved.
[7:0]	BBGEN_DIV[7:0]	BBGEN pre-divider value. Divison value =BBGEN_DIV+1. `0x00`: Minimum divide value =1, no division. `0xFF`: Maximum divide value =1024. Reset value `0x28`.

CLK_POSTDIV_CTRL_QSPI Register

The CLK_POSTDIV_CTRL_QSPI Register characteristics are:

Purpose
Controls the QSPI clock post PLL clock divider, QSPIDIV, division value.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_POSTDIV_CTRL_QSPI bit assignments.

Table 3-71 CLK_POSTDIV_CTRL_QSPI Register bit assignments

Bits	Name	Function
[31:8]	-	Reserved.
[7:0]	POSTDIV_CTRL_QSPI_DIV[7:0]	QSPI clock divider, QSPIDIV, division value: Divison value =POSTDIV_CTRL_QSPI_DIV +1. `0x00`: Minimum division value =1 (no division). `0xFF`: Maximum division value =256. Reset value `0x00`, division value = 1.

CLK_POSTDIV_CTRL_RTC Register

The CLK_POSTDIV_CTRL_RTC Register characteristics are:

Purpose
Controls the RTC clock post PLL clock divider, RTCDIV, division value.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_POSTDIV_CTRL_RTC Register bit assignments.

Table 3-72 CLK_POSTDIV_CTRL_RTC Register bit assignments

Bits	Name	Function
[31:0]	POSTDIV_CTRL_RTC_DIV[31:0]	RTC clock divider division value: Divison value =POSTDIV_CTRL_RTC_DIV +1. `0x0000_0000`: Minimum division value =1 (no division). `0x0000_7FFF`: Maximum division value =32768. Reset value `0x0000_7FFF`.

CLK_POSTDIV_CTRL_TEST Register

The CLK_POSTDIV_CTRL_TEST Register characteristics are:

Purpose
Controls the TEST_CLK clock post PLL clock divider, TESTDIV, division value.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_POSTDIV_CTRL_TEST Register bit assignments.

Table 3-73 CLK_POSTDIV_CTRL_TEST Register bit assignments

Bits	Name	Function
[31:8]	-	Reserved.
[7:0]	POSTDIV_CTRL_TEST_DIV[7:0]	TEST_CLK clock divider, TESTDIV, division value: Divison value =POSTDIV_CTRL_SD_DIV +1. `0x00`: Minimum division value =1 (no division). `0xFF`: Maximum division value =256. Reset value `0x0A`.

CTRL_BYPASS_DIV Register

The CTRL_BYPASS_DIV Register characteristics are:

Purpose
Controls the post PLL clock divider bypass functions.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CTRL_BYPASS_DIV Register bit assignments.

Table 3-74 CTRL_BYPASS_DIV Register bit assignments

Bits	Name	Function
[31:7]	-	Reserved.
[6]	BYPASS_TEST_DIV_CLK	Bypass clock divider TESTDIV: `0b0`: Not bypass. `0b1`: Bypass. Reset value `0b0`.
[5]	-	Reserved.
[4]	BYPASS_RTC_DIV_CLK	Bypass clock divider RTCDIV: `0b0`: Not bypass. `0b1`: Bypass. Reset value `0b0`.
[3]	BYPASS_QSPI_DIV_CLK	Bypass clock divider QSPIDIV: `0b0`: Not bypass. `0b1`: Bypass. Reset value `0b0`.

Table 3-74 CTRL_BYPASS_DIV Register bit assignments (continued)

Bits	Name	Function
[2:1]	-	Reserved.
[0]	BYPASS_PLL_PREDIV_CLK	Bypass clock divider PREDIV: `0b0`: Not bypass. `0b1`: Bypass. Reset value `0b1`.

PLL_CTRL_PLL0_CLK Register

The PLL_CTRL_PLL0_CLK Register characteristics are:

Purpose

Controls PLL0. Sets the internal oscillator target frequency and the output divider value.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the PLL_CTRL_PLL0_CLK Register bit assignments.

Table 3-75 PLL_CTRL_PLL0_CLK Register bit assignment

Bits	Name	Function
[31:15]	-	Reserved.
[14:12]	PLL0_S	Controls the output divider that derives the PLL0 output, **PLL0_CLK**, from the internal oscillator frequency: `0b000`: **PLL0_CLK=INT_OSC**/1. `0b001`: **PLL0_CLK=INT_OSC**/2. `0b010`: **PLL0_CLK=INT_OSC**/4. `0b011`: **PLL0_CLK=INT_OSC**/8. `0b100`: **PLL0_CLK=INT_OSC**/16. `0b101`: **PLL0_CLK=INT_OSC**/32. `0b110`: **PLL0_CLK=INT_OSC**/64. `0b111`: **PLL0_CLK=INT_OSC**/128. Reset value `0b000`.
[11:0]	PLL0_M	Controls the feedback divider of PLL0 to set the target frequency of the internal oscillator: **INT_OSC**=4×PLL0_M×**PRE_PLL_CLK**. Reset value `0x5F4`.

Note

INT_OSC is not shown in the clock system diagram in *2.5 Clocks* on page 2-28.

CLK_CTRL_ENABLE Register

The CLK_CTRL_ENABLE Register characteristics are:

Purpose
Controls clock gate enable functions.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_CTRL_ENABLE Register bit assignments.

Table 3-76 CLK_CTRL_ENABLE Register bit assignment

Bits	Name	Function
[31:16]	-	Reserved.
[15]	CTRL_ENABLE_TESTCLK	Enable TEST_CLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[14]	CTRL_ENABLE_TAPTCK	Enable TCK MUX: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[13]	-	Reserved.
[12]	CTRL_ENABLE_SCCCLK	Enable SCCCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[11]	CTRL_ENABLE_BBGEN	Enable BBGEN clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[10]	CTRL_ENABLE_REFCLK	Enable REFCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.

Table 3-76 CLK_CTRL_ENABLE Register bit assignment (continued)

Bits	Name	Function
[9]	CTRL_ENABLE_QSPI_PHY_CLK	Enable QSPI PHY clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[8]	CTRL_ENABLE_MAINCLK	Enable MAINCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[7:6]	-	Reserved..
[5]	CTRL_ENABLE_I2SCLK2	Enable IS2CLK2 SYSSYSUGCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[4]	CTRL_ENABLE_I2SCLK1	Enable I2SCLK1 SYSSYSUGCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[3]	CTRL_ENABLE_I2SCLK0	Enable I2SCLK0 SYSSYSUGCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[2]	CTRL_ENABLE_GPIOHCLK	Enable GPIO SYSSYSUGCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.

Table 3-76 CLK_CTRL_ENABLE Register bit assignment (continued)

Bits	Name	Function
[1]	CTRL_ENABLE_DAPSWCLK	Enable DAPSWCLK clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[0]	CTRL_ENABLE_1HZ	Enable RTC clock gate: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.

CLK_STATUS Register

The CLK_STATUS Register characteristics are:

Purpose

Stores PLL status values.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CLK_STATUS Register bit assignments.

Table 3-77 CLK_STATUS Register bit assignment

Bits	Name	Function
[31:2]	-	Reserved.
[1]	STATUS_LOCK_SIGNAL_PLL0_CLK	PLL lock status: `0b0`: Not locked. `0b1`: Locked. Reset value `0b1`.
[0]	STATUS_OUT_CLK_MAINCLK_READY	Main clock ready status: `0b0`: Not ready. `0b1`: Ready. Reset value `0b1`.

RESET_CTRL Register

The RESET_CTRL Register characteristics are:

Purpose

Resets Musca-S1 test chip peripherals.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the RESET_CTRL Register bit assignments.

Table 3-78 RESET_CTRL Register bit assignment

Bits	Name	Function
[31:18]	-	Reserved.
[17]	SSE_200_NRST	Reset SSE-200 subsystem: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[16]	SSE_200_NRST_PSI_SEL	Reset SSE-200 subsystem: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[15]	SSE_200_NRST_SEL	Reset SSE-200 subsystem: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[14]	RTC_RESET	Reset Real Time Clock: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[13]	PWM2_RESET	Reset PWM2: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[12]	PWM1_RESET	Reset PWM1: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[11]	PWM0_RESET	Reset PWM0: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.

Table 3-78 RESET_CTRL Register bit assignment (continued)

Bits	Name	Function
[10]	PVT_RESET	Reset PVT: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[9]	GPIO_RESET	Reset GPIO: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[8]	UART1_RESET	Reset UART1: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[7]	UART0_RESET	Reset UART0: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[6]	QSPI_RESET	Reset QSPI: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[5]	SPI_RESET	Reset SPI: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[4]	I2S_RESET	Reset I^2S: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[3]	I2C1_RESET	Reset I^2C1: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.

Table 3-78 RESET_CTRL Register bit assignment (continued)

Bits	Name	Function
[2]	I2C0_RESET	Reset I^2C0: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[1]	GPTIMER_RESET	Reset general-purpose timer: `0b0`: Reset. `0b1`: No effect. Reset value `0b1`.
[0]	-	Reserved.

DBG_CTRL Register

The DBG_CTRL Register characteristics are:

Purpose
Controls debug authentication signals.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the DBG_CTRL Register bit assignments.

Table 3-79 DBG_CTRL Register bit assignment

Bits	Name	Function
[31]	DBG_DCU_FORCE	SSE-200 debug ports control: `0b0`: Use Crypto DCU. `0b1`: Use SCC signals (Force). Reset value `0b0`.
[30:9]	-	Reserved.
[8]	TODBGENSEL1	Enable or mask, bypass, Flush input from the Cross Trigger Interface: `0b0`: Enabled. `0b1`: Mask, or bypass. Reset value `0b0`.

Table 3-79 DBG_CTRL Register bit assignment (continued)

Bits	Name	Function
[7]	TODBGENSEL0	Enable or mask, bypass, Trigger input from the Cross Trigger Interface: `0b0`: Enabled. `0b1`: Mask, or bypass. Reset value `0b0`.
[6:4]	-	Reserved.
[3]	SSE-200 SPNIDENIN	Secure Privilege Non-Invasive Debug Enable Input: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[2]	SSE-200 SPIDENIN	Secure Privilege Invasive Debug Enable Input: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[1]	SSE-200 NIDENIN	Non-Invasive Debug Enable Input: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.
[0]	SSE-200 DBGENIN	Debug Enable Input: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b1`.

SRAM_CTRL Register

The SRAM_CTRL Register characteristics are:

Purpose
Controls SRAM power gate enable signals.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SRAM_CTRL Register bit assignments.

Table 3-80 SRAM_CTRL Register bit assignment

Bits	Name	Function
[31]	CODE_SRAM31_PGEN	32nd 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[30]	CODE_SRAM30_PGEN	31st 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[29]	CODE_SRAM29_PGEN	30th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[28]	CODE_SRAM28_PGEN	29th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[27]	CODE_SRAM27_PGEN	28th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[26]	CODE_SRAM26_PGEN	27th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[25]	CODE_SRAM25_PGEN	26th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[24]	CODE_SRAM24_PGEN	25th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.

Table 3-80 SRAM_CTRL Register bit assignment (continued)

Bits	Name	Function
[23]	CODE_SRAM23_PGEN	24th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[22]	CODE_SRAM22_PGEN	23rd 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[21]	CODE_SRAM21_PGEN	22nd 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[20]	CODE_SRAM20_PGEN	21st 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[19]	CODE_SRAM19_PGEN	20th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[18]	CODE_SRAM18_PGEN	19th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[17]	CODE_SRAM17_PGEN	18th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[16]	CODE_SRAM16_PGEN	17th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.

Table 3-80 SRAM_CTRL Register bit assignment (continued)

Bits	Name	Function
[15]	CODE_SRAM15_PGEN	16th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[14]	CODE_SRAM14_PGEN	15th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[13]	CODE_SRAM13_PGEN	14th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[12]	CODE_SRAM12_PGEN	13th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[11]	CODE_SRAM11_PGEN	12th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[10]	CODE_SRAM10_PGEN	11th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[9]	CODE_SRAM9_PGEN	10th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[8]	CODE_SRAM8_PGEN	9th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.

Table 3-80 SRAM_CTRL Register bit assignment (continued)

Bits	Name	Function
[7]	CODE_SRAM7_PGEN	8th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[6]	CODE_SRAM6_PGEN	7th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[5]	CODE_SRAM5_PGEN	6th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[4]	CODE_SRAM4_PGEN	5th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[3]	CODE_SRAM3_PGEN	4th 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[2]	CODE_SRAM2_PGEN	3rd 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[1]	CODE_SRAM1_PGEN	2nd 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.
[0]	CODE_SRAM0_PGEN	1st 64KB SRAM cell power gate enable: `0b0`: Not enabled. `0b1`: Enabled. Reset value `0b0`.

INTR_CTRL Register

The INTR_CTRL Register characteristics are:

Purpose
Controls MPC interrupt signals.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the INTR_CTRL Register bit assignments.

Table 3-81 INTR_CTRL Register bit assignment

Bits	Name	Function
[31:3]	-	Reserved.
[2]	MRAM_MPC_CFG_INIT_VALUE	Initial security map at startup for eMRAM MPC: `0b0`: Secure mode. `0b1`: Non-secure mode. Reset value `0b0`.
[1]	QSPI_MPC_CFG_INIT_VALUE	Initial security map at startup for QSPI MPC: `0b0`: Secure mode. `0b1`: Non-secure mode. Reset value `0b0`.
[0]	SRAM_MPC_CFG_INIT_VALUE	Initial security map at startup for SRAM MPC: `0b0`: Secure mode. `0b1`: Non-secure mode. Reset value `0b0`.

CPU0_VTOR Register

The CPU0_VTOR Register characteristics are:

Purpose
Controls reset vector for CPU0 secure mode.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CPU0_VTOR Register bit assignments.

Table 3-82 CPU0_VTOR Register bit assignments

Bits	Name	Function
[31:0]	INITSVTOR0_RST_0	Reset vector for CPU0 secure mode to external QSPI Flash when BOOT=`0b0`. Reset value `0x1020_0000`.

CPU0_VTOR_1 Register

The CPU0_VTOR_1 Register characteristics are:

Purpose
Controls reset vector for CPU0 secure mode.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CPU0_VTOR_1 Register bit assignments.

Table 3-83 CPU0_VTOR_1 Register bit assignments

Bits	Name	Function
[31:0]	INITSVTOR0_RST_1	Reset vector for CPU0 secure mode to code eMRAM when BOOT=`0b1`: Reset value `0x1A00_0000`.

CPU1_VTOR Register

The CPU1_VTOR Register characteristics are:

Purpose
Controls reset vector for CPU1 secure mode.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CPU1_VTOR Register bit assignments.

Table 3-84 CPU1_VTOR Register bit assignments

Bits	Name	Function
[31:0]	INITSVTOR1_RST_0	Reset vector for CPU1 Secure mode to external QSPI Flash when BOOT=`0b0`: Reset value `1020_0000`.

CPU1_VTOR_1 Register

The CPU1_VTOR_1 Register characteristics are:

Purpose
Controls reset vector for CPU1 secure mode.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CPU1_VTOR_1 Register bit assignments.

Table 3-85 CPU1_VTOR_1 Register bit assignments

Bits	Name	Function
[31:0]	INITSVTOR1_RST_1	Reset vector for CPU1 Secure mode to code eMRAM when BOOT=`0b1`: Reset value `1A00_0000`.

IOMUX_MAIN_INSEL Register

The IOMUX_MAIN_INSEL Register characteristics are:

Purpose
Selects either MAIN_IN or ALTF1 as destination of input signals from multiplexed Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_MAIN_INSEL Register bit assignments.

Table 3-86 IOMUX_MAIN_INSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_MAIN_INSEL[31:0]	Main function input data select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF1. `0b1`: Select MAIN_IN. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_MAIN_OUTSEL Register

The IOMUX_MAIN_OUTSEL Register characteristics are:

Purpose
Selects either MAIN_OUT or ALTF1 as output data for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_MAIN_OUTSEL bit assignments.

Table 3-87 IOMUX_MAIN_OUTSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_MAIN_OUTSEL[31:0]	Main function output data select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF1. `0b1`: Select MAIN_OUT. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_MAIN_OENSEL Register

The IOMUX_MAIN_OENSEL Register characteristics are:

Purpose

Selects either MAIN_OE or ALTF1 as output enable signal for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_MAIN_OENSEL Register bit assignments.

Table 3-88 IOMUX_MAIN_OENSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_MAIN_OENSEL_0[31:0]	I/O main function output enable select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF1. `0b1`: Select MAIN_OE. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_MAIN_DEFAULT_IN Register

The IOMUX_MAIN_DEFAULT_IN Register characteristics are:

Purpose

Musca-S1 test chip I/O PA31-PA0: Drives unselected outputs of MAIN input multiplexers to defined logic levels to prevent floating nodes.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_MAIN_DEFAULT_IN Register bit assignments.

Table 3-89 IOMUX_MAIN_DEFAULT_IN Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_MAIN_DEFAULT_IN_0[31:0]	Defines value of unselected outputs of ALTF1 input multiplexers for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Default to `0b0`. `0b1`: Default to `0b1`. Reset value `0x0000_0000`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF1_INSEL Register

The IOMUX_ALTF1_INSEL Register characteristics are:

Purpose

Selects either ALTF1 or ALTF2 as destination of input signals from MAIN input multiplexer for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF1_INSEL Register bit assignments.

Table 3-90 IOMUX_ALTF1_INSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF1_INSEL[31:0]	Selects either ALTF1 or ALTF2 as destination of MAIN input multiplexer for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF1_IN. `0b1`: Select ALTF2. Reset value `0x0000_0000`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF1_OUTSEL Register

The IOMUX_ALTF1_OUTSEL Register characteristics are:

Purpose

Selects either ALTF1_OUT or ALTF2 as output data for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF1_OUTSEL Register bit assignments.

Table 3-91 IOMUX_ALTF1_OUTSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF1_OUTSEL[31:0]	Main function output data select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF2. `0b1`: Select ALTF1_OUT. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF1_OENSEL Register

The IOMUX_ALTF1_OENSEL Register characteristics are:

Purpose

Selects either ALTF1_OE or ALTF2 as output enable signal for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF1_OENSEL Register bit assignments.

Table 3-92 IOMUX_ALTF1_OENSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF1_OENSEL[31:0]	I/O main function output enable select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF2. `0b1`: Select ALTF1_OE. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF1_DEFAULT_IN Register

The IOMUX_ALTF1_DEFAULT_IN Register characteristics are:

Purpose

Test chip I/O PA31-PA0: Drives unselected outputs of ALTF1 input multiplexers to defined logic levels to prevent floating nodes.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF1_DEFAULT_IN Register bit assignments.

Table 3-93 IOMUX_ALTF1_DEFAULT_IN Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF1_DEFAULT_IN[31:0]	Defines value of unselected outputs of ALTF1 input multiplexers for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Default to `0b0`. `0b1`: Default to `0b1`. Reset value `0x0000_0000`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF2_INSEL Register

The IOMUX_ALTF2_INSEL Register characteristics are:

Purpose

Selects either ALTF2_IN or ALTF3_IN as destination of input signals from ALTF1 input multiplexer for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF2_INSEL Register bit assignments.

Table 3-94 IOMUX_ALTF2_INSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF2_INSEL[31:0]	Selects either ALTF2_IN or ALTF3_IN as destination of ALTF1 input multiplexer for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF3_IN. `0b1`: Select ALTF2_IN. Reset value `0x0000_0000`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF2_OUTSEL Register

The IOMUX_ALTF2_OUTSEL Register characteristics are:

Purpose

Selects either ALTF2_OUT or ALTF3_OUT as output data for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF2_OUTSEL Register bit assignments.

Table 3-95 IOMUX_ALTF2_OUTSEL Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF2_OUTSEL[31:0]	Main function output data select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF3_OUT. `0b1`: Select ALTF2_OUT. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF2_OENSEL Register

The IOMUX_ALTF2_OENSEL_0 Register characteristics are:

Purpose

Selects either ALTF2_OE or ALTF3 as output enable signal for Musca-S1 test chip I/O PA31-PA0.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF2_OENSEL_0 Register bit assignments.

Table 3-96 IOMUX_ALTF2_OENSEL_0 Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF2_OENSEL_0[31:0]	I/O main function output enable select for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Select ALTF3. `0b1`: Select ALTF2_OE. Reset value `0xFFFF_FFFF`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOMUX_ALTF2_DEFAULT_IN Register

The IOMUX_ALTF2_DEFAULT_IN Register characteristics are:

Purpose

Test chip I/O PA31-PA0: Drives unselected outputs of ALTF1 input multiplexers to defined logic levels to prevent floating nodes.
See *3.11.1 IOMUX registers* on page 3-120 for information on the Musca-S1 test chip I/O multiplexer.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOMUX_ALTF2_DEFAULT_IN Register bit assignments.

Table 3-97 IOMUX_ALTF2_DEFAULT_IN Register bit assignments

Bits	Name	Function
[31:0]	IOMUX_ALTF2_DEFAULT_IN[31:0]	Defines value of unselected outputs of ALTF2 input multiplexers for Musca-S1 test chip multiplexed I/O PA31-PA0: `0b0`: Default to `0b0`. `0b1`: Default to `0b1`. Reset value `0x0000_0000`. **Note** See *2.2.2 Test chip multiplexed I/O* on page 2-23 for the functions that are available on the multiplexed Musca-S1 test chip I/O.

IOPAD_DS0 and IOPAD_DS1 Registers

The IOPAD_DS0 and IOPAD_DS1 Register characteristics are:

Purpose

The corresponding bits of the two registers combine to form two-bit values that define the corresponding drive strengths ofMusca-S1 test chip I/O PA31-PA0. The following table shows how the bits of the IOPAD_DS0 and IOPAD_DS1 Registers define the drive strengths.

Table 3-98 Test chip I/O drive strengths

IOPAD_DS1/DS0_0	Drive strength (mA)
`0b00`	2 Default for PA31-PA26
`0b01`	8, default for PA25-PA20
`0b10`	4, default for PA19-PA0
`0b11`	12

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following tables show the IOPAD_DS0 and IOPAD_DS1 Register bit assignments.

Table 3-99 IOPAD_DS0 Register bit assignments

Bits	Name	Function
[31:0]	DRIVE_STRENGTH0	Least significant bits of the two-bit values that define drive strengths ofMusca-S1 test chip I/O PA31-PA0. Reset value `0x03F0_0000`.

Table 3-100 IOPAD_DS1 Register bit assignments

Bits	Name	Function
[31:0]	DRIVE_STRENGTH1	Most significant bits of the two-bit values that define drive strengths ofMusca-S1 test chip I/O PA31-PA0. Reset value `0x000F_FFFF`.

IOPAD_PE Register

The IOPAD_PE Register characteristics are:

Purpose
Register IOPAD_PE enables pull resistors onMusca-S1 test chip I/O PA31-PA0.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOPAD_PE Register bit assignments.

Table 3-101 IOPAD_PE Register bit assignments

Bits	Name	Function
[31:0]	PULL_ENABLE	Enable pull resistors ofMusca-S1 test chip I/O PA31-PA0. `0b0`: Not enabled. `0b1`: Enabled. Reset value `0xFFFF_FFFF`.

IOPAD_PS Register

The IOPAD_PS Register characteristics are:

Purpose
Register IOPAD_PS controls the pull resistor modes onMusca-S1 test chip I/O PA31-PA0.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOPAD_PS Register bit assignments.

Table 3-102 IOPAD_PS Register bit assignments

Bits	Name	Function
[31:0]	PULL_SELECT	Selects pull mode of pull resistors onMusca-S1 test chip I/O PA31-PA0. `0b0`: Pull down. `0b1`: Pull up. Reset value `0xF81F_FFFF`.

IOPAD_SR Registers

The IOPAD_SR Register characteristics are:

Purpose

Register IOPAD_SR controls the slew rates of Musca-S1 test chip I/O PA31-PA0.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOPAD_SR Register bit assignments.

Table 3-103 IOPAD_SR Register bit assignments

Bits	Name	Function
[31:0]	SLEW_RATE	Selects the slew rate ofMusca-S1 test chip I/O I/O PA31-PA0. `0b0`: Fast. `0b1`: Slow. Reset value `0xFFFF_FFFF`.

IOPAD_IS Registers

The IOPAD_IS Register characteristics are:

Purpose

Register IOPAD_IS controls the input modes on Musca-S1 test chip I/O PA31-PA0.PA32.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IOPAD_IS Register bit assignments.

Table 3-104 IOPAD_IS Register bit assignments

Bits	Name	Function
[31:0]	INPUT_SELECT	Selects input mode onMusca-S1 test chip I/O PA31-PA0. `0b0`: CMOS. `0b1`: Schmitt. Reset value `0xFFFF_FFFF`.

SPARE0 Register

The SPARE0 Register characteristics are:

Purpose

Spare read/write register for use by software.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SPARE0 Register bit assignments.

Table 3-105 SPARE0 Register bit assignments

Bits	Name	Function
[31:0]	SPARE0[31:0]	Spare read/write register for software. Software assigns the bit meanings. Reset value `0x0000_0000`.

STATIC_CONF_SIG1 Register

The STATIC_CONF_SIG1 Register characteristics are:

Purpose

Static configuration control register.

Usage constraints

There are no usage constraints.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the STATIC_CONF_SIG1 Register bit assignments.

Table 3-106 STATIC_CONF_SIG1 Register bit assignments

Bits	Name	Function
[31:28]	-	Reserved.
[27:24]	TODBGENSEL	**DBGEN** mask on **CTITRIGOUT**: `0b0`: Mask trigger output of associated Cross Trigger Interface output when **DBGEN** is LOW. `0b1`: Not mask trigger output of associated Cross Trigger Interface output. Reset value `0b0000`.
[23:16]	TINIDENSEL	**NIDEN** mask on **CTITRIGINT**: `0b0`: Mask trigger input of associated Cross Trigger Interface output when **NIDEN** is LOW. `0b1`: Not mask trigger output of associated Cross Trigger Interface output. Reset value `0x00`.

Table 3-106 STATIC_CONF_SIG1 Register bit assignments (continued)

Bits	Name	Function
[15:12]	TIHSBYPASS	Cross Trigger Interface handshake bypass on **CTITRIGOUT**. Disables the **SPIDEN** selector logic and forces **SPIDEN** to use **SPIDENIN**: `0b0`: Not disable. `0b1`: Disable. Reset value `0b0000`.
[11:8]	TISBYPASSACK	Cross Trigger Interface synchronous bypass on **CTITRIGOUTACK**. Set HIGH to bypass the synchronization logic if the **CTITRIGOUTACK** input is synchronous with **DBGSYSCLK** and is driven from the same clock domain: `0b0`: Not bypass. `0b1`: Bypass. Reset value `0b0000`.
[7:0]	TISBYPASSIN	Cross Trigger Interface synchronous bypass on **CTITRIGIN**. Set HIGH to bypass the synchronization logic if the **CTITRIGIN** input is synchronous with **DBGSYSCLK** and is driven from the same clock domain: `0b0`: Not bypass. `0b1`: Bypass. Reset value `0x00`.

SCC_MRAM_CTRL0 Register

The SCC_MRAM_CTRL0 Register characteristics are:

Purpose
Controls the eMRAM memory controller and clock.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_CTRL0 Register bit assignments.

Table 3-107 SCC_MRAM_CTRL0 Register bit assignments

Bits	Name	Function
[31]	MRAM_DA_EN	Enable eMRAM direct access: `0b0`: Disabled. `0b1`: Enabled. Reset value `0b0`.
[30]	MRAM_CLK_SYNC_BYPASS	Bypass eMRAM clock divider sync: `0b0`: Disable. `0b1`: Enabled. Reset value `0b1`.
[29]	MRAM_OTP_CLK_EN	Enable eMRAM OTP clock: `0b0`: Disabled-automatic when needed. `0b1`: Enabled-forced always on. Reset value `0b0`.
[28:24]	-	Reserved.
[23:20]	READ_CSN_CLKS	Number of clocks for single read Reset value `0b0001`.
[19:16]	CSN_HIGH_CLKS	Number of clock cycles to wait when **CSN** is HIGH before going to new access (**CSN** low) Reset value `0b0100`.
[15:12]	WRITE_CSN_CLKS	Number of clock cycles for single write operation Reset value `0b0100`.
[11]	PG_VDD18_1	eMRAM1 PG VDD18: `0b0`: Powered up. `0b1`: Powered down. Reset value `0b0`.
[10]	PG_VDD_1	eMRAM1 PG VDD: `0b0`: Powered up. `0b1`: Powered down. Reset value `0b0`.
[9]	PG_VDD18_0	eMRAM1 PG VDD18_0: `0b0`: Powered up. `0b1`: Powered down. Reset value `0b0`.

Table 3-107 SCC_MRAM_CTRL0 Register bit assignments (continued)

Bits	Name	Function
[8]	PG_VDD_0	eMRAM0 PG VDD: `0b0`: Powered up. `0b1`: Powered down. Reset value `0b0`.
[7]		Reserved.
[6:5]	MRAM_DOUT_SEL	Select eMRAM0 output data. `0b00`: Wait one cycle for eMRAM output data sample: `0b01`: Wait two cycles for eMRAM output data sample. `0b10`: Wait three cycles for eMRAM output data sample. `0b11`: Wait four cycles for eMRAM output data sample (very long wait). Reset value `0b00`.
[4]	FAST_READ_EN	Enable fast read: `0b0`: Normal read. `0b1`: Fast read Reset value `0b0`.
[3]	MRAM_INV_CLK_SEL	Select clock inversion: `0b0`: Inverted. `0b1`: Not inverted. Reset value `0b0`.
[2]	AUTOSTOP_EN	Enable autostop: `0b0`: Disabled. `0b1`: Enabled. Reset value `0b0`.
[1]	PROC_SPEC_CLK_EN	Enable eMRAM controller clock: `0b0`: Disabled. `0b1`: Enabled. Reset value `0b1`.
[0]	MRAM_CLK_EN	Enable eMRAM clock: `0b0`: Disabled. `0b1`: Enabled. Reset value `0b1`.

SCC_MRAM_CTRL1 Register

The SCC_MRAM_CTRL1 Register characteristics are:

Purpose
Controls the eMRAM memory direct access.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_CTRL1 Register bit assignments.

Table 3-108 SCC_MRAM_CTRL1 Register bit assignments

Bits	Name	Function
[31:30]	-	Reserved.
[29:24]	MRAM_DA_CTRL	eMRAM direct access controls: [29]: **MRAM_STOP**. [28]: **MRAM_RESTB**. [27]: **MRAM_LOAD_START**. [26]: **MRAM_LOAD_RSTN**. [25]: **MRAM_WEN**. [24]: **MRAM_CSN**. Reset value `0b000000`.
[23:0]	MRAM_DA_ADDR	eMRAM direct access address. Reset value `0x00_0000`.

SCC_MRAM_CTRL2 Register

The SCC_MRAM_CTRL2 Register characteristics are:

Purpose
Controls the eMRAM memory direct access.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_CTRL2 Register bit assignments.

Table 3-109 SCC_MRAM_CTRL2 Register bit assignments

Bits	Name	Function
[31:16]	TIME_TO_STOP	If autostop is enabled, number of microseconds to wait in inactive mode before going to STOP mode. Reset value `0xFFFF`.
[15:8]	MRAM_CLK_DIV	eMRAM clock divider: `0x00`: No division (bypass). `0x01`: Divide by two (default). `0x10`: Divide by three. `0x11`: Divide by four. Undefined values are not used. Reset value `0x01`.
[7:0]	PRESCALE	Clock prescaler to generate the timebase for the eMRAM clock. The value of PRESCALE must always meet the following equality: PRESCALE=eMRAM clock(MHz). Preserving this equality generates a 1µS timebase and ensures correct operation of the eMRAM memory and clock. If you adjust the eMRAM clock, you must adjust PRESCALE to preserve the equality. For example, if: • eMRAM clock =15MHz: — PRESCALE=`0x0F` (15). • eMRAM clock =20MHz: — PRESCALE=`0x14` (20). • eMRAM clock =30MHz: — PRESCALE=`0x1E` (30). Reset value `0x19` (25MHz).

SCC_MRAM_DIN0 Register

The SCC_MRAM_DIN0 Register characteristics are:

Purpose
eMRAM data input[31:0].

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_DIN0 Register bit assignments.

Table 3-110 SCC_MRAM_DIN0 Register bit assignments

Bits	Name	Function
[31:0]	MRAM_DIN[31:0]	eMRAM data input[31:0]. Reset value `0x0000_0000`.

SCC_MRAM_DIN1 Register

The SCC_MRAM_DIN1 Register characteristics are:

Purpose
eMRAM data input[63:32].

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_DIN1 Register bit assignments.

Table 3-111 SCC_MRAM_DIN1 Register bit assignments

Bits	Name	Function
[31:0]	MRAM_DIN[63:32]	eMRAM data input[63:32]. Reset value `0x0000_0000`.

SCC_MRAM_DOUT0 Register

The SCC_MRAM_DOUT0 Register characteristics are:

Purpose
eMRAM memory data output[31:0].

Usage constraints
This register is read-only.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_DOUT0 Register bit assignments.

Table 3-112 SCC_MRAM_DOUT0 Register bit assignments

Bits	Name	Function
[31:0]	MRAM_DOUT[31:0]	eMRAM memory data output[31:0]. Reset value `0x0200_F2CE`.

SCC_MRAM_DOUT1 Register

The SCC_MRAM_DOUT1 Register characteristics are:

Purpose
eMRAM memory data output[63:32].

Usage constraints
This register is read-only.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_DOUT1 Register bit assignments.

Table 3-113 SCC_MRAM_DOUT1 Register bit assignments

Bits	Name	Function
[31:0]	MRAM_DOUT[63:32]	eMRAM memory data output[63:32]. Reset value `0x9A09_5880`.

SCC_MRAM_DOUT2 Register

The SCC_MRAM_DOUT2 Register characteristics are:

Purpose

eMRAM memory data output[77:64], the ECC values that are added by the eMRAM Error Correcting Code (ECC) module.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_DOUT2 Register bit assignments.

Table 3-114 SCC_MRAM_DOUT2 Register bit assignments

Bits	Name	Function
[31:0]	MRAM_DOUT[77:64]	eMRAM memory data output[77:64]. These bits store the ECC values automatically added by the eMRAM ECC module. Reset value `0x0000_1296`.

SCC_MRAM_STATUS Register

The SCC_MRAM_STATUS Register characteristics are:

Purpose

Stores status of eMRAM controller.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SCC_MRAM_STATUS Register bit assignments.

Table 3-115 SCC_MRAM_STATUS Register bit assignments

Bits	Name	Function
[31:5]	-	Reserved.
[4:0]	FSM_STATE	eMRAM controller FSM state: `0b00000`: OTP_RST. `0b00001`: OTP_LD. `0b00010`: PORD. `0b00011`: RESETB. `0b00100`: PORM. `0b00101`: PWR_UP. `0b00110`: PWR_DWN. `0b00111`: FAST_RD. `0b01000`: READ. `0b01001`: WRITE. `0b01010`: WAIT. `0b01011`: CSN_WAIT. `0b01100`: STOP. `0b01101`: DIRECT. `0b01110`: OTP_RST_WAIT. `0b01111`: FAST_HOLD. `0b10000`: `0b10001`: Reset value `0b01100`.

SELECTION_CONTROL_REG Register

The SELECTION_CONTROL_REG Register characteristics are:

Purpose

Controls clock phase shift control signals.

Usage constraints

There are no usage register read or write constraints.

——— **Note** ———

Arm recommends that you do not alter the default values during normal operation.

———————————

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SELECTION_CONTROL_REG Register bit assignments.

Table 3-116 SELECTION_CONTROL_REG Register bit assignments

Bits	Name	Function
[31:3]	-	Reserved.
[2]	CLOCK_PHASE_SHIFTER_BYPASS	QSPI input clock phase shift control: `0b0`: Clock phase shift activated. `0b1`: Clock phase shift is bypassed and clock delayed is selected from the pad SCLK_OUT. Reset value `0b0`.
[1:0]	CLOCK_PHASE_SHIFTER_SELECT	QSPI input clock phase shift control: `0b00`: No phase shift. `0b01`: 90° phase shift. `0b10`: 180° phase shift. `0b11`: 270° phase shift. Reset value `0b00`.

BBGEN_CTRL Register

The BBGEN_CTRL Register characteristics are:

Purpose
Controls transistor body-biasing.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the BBGEN_CTRL Register bit assignments.

Table 3-117 BBGEN_CTRL Register bit assignments

Bits	Name	Function
[31:8]	-	Reserved.
[7:5]	N_CTRL	Select VBBN_OUT range: `0b001`: VBBN_OUT=-0.4V. `0b010`: VBBN_OUT=-0.6V. `0b011`: VBBN_OUT=-0.8V. `0b100`: VBBN_OUT=-1.0V. `0b101`: VBBN_OUT=-1.2V. `0b110`: VBBN_OUT=-1.4V (recommended). Reset value `0b000`.

Table 3-117 BBGEN_CTRL Register bit assignments (continued)

Bits	Name	Function
[4]	BP_HIGH	Bypass AVDD to Positive BBG. To apply body-biasing in bypass mode, mix settings with external voltages apply to VBBP and VBBN pads: `0b0`: Apply VBBP=VDD_CORE, VBBN=0V. `0b1`: Bypass AVDD for enhanced bypass. Apply VBBP=AVDD, VBBN=0V. Reset value `0b0`.
[3:1]	P_CTRL	Select VBBP range: `0b001`: VBBP_OUT=DVDD+0.2V. `0b010`: VBBP_OUT=DVDD+0.3V. `0b011`: VBBP_OUT=DVDD+0.4V. Reset value `0b000`.
[0]	EN	Enable body-bias function: `0b0`: Bypass mode, or OFF if no external bypass voltage is applied on VBBP and VBBN. `0b1`: Body-bias enabled. BBGen is ON. Reset value `0b0`.

SPARE_CTRL1 Register

The SPARE_CTRL1 Register characteristics are:

Purpose
Spare control register.

Usage constraints
There are no usage constraints.

Memory offset and full register reset value
See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the SPARE_CTRL1 Register bit assignments.

Table 3-118 SPARE_CTRL1 Register bit assignments

Bits	Name	Function
[31:0]	SPARE_CTRL1	Spare control register. Software assigns the bit meanings. Reset value `0x0000_0000`.

CHIP_ID Register

The CHIP_ID Register characteristics are:

Purpose

Stores component identification information.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the CHIP_ID Register bit assignments.

Table 3-119 CHIP_ID Register bit assignments

Bits	Name	Function
[31:0]	CHIP_ID	Component ID information. The value in the Musca-S1 test chip is `0x0799_0477`.

IO_IN_STATUS Register

The IO_IN_STATUS Register characteristics are:

Purpose

Stores component identification information.

Usage constraints

This register is read-only.

Memory offset and full register reset value

See *3.11.2 SCC registers summary* on page 3-123.

The following table shows the IO_IN_STATUS Register bit assignments.

Table 3-120 IO_IN_STATUS Register bit assignments

Bits	Name	Function
[31:0]	IO_IN_STATUS	Real time I/O pads input status. Bit number corresponds to pad number, bit[0]=PA0). Reset value `0x05FF_FFE3`.

3.12 UART control registers

The Musca-S1 test chip contains registers that control the two UARTs, UART0 and UART1.

The base memory addresses of UART0 are:

- `0x4010_1000` in the Non-secure region.
- `0x5010_1000` in the Secure region.

The base memory addresses of UART1 are:

- `0x4010_2000` in the Non-secure region.
- `0x5010_2000` in the Secure region.

See the *PrimeCell UART (PL011) Technical Reference Manual*.

——— **Note** ———

The default connectivity of the UARTs is:

- UART0 to the Shield header.
- UART1 to DAPLink. No hardware flow control.

.

———————————

The following table shows the UART0 and UART1 control registers in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-121 UART control registers summary

Offset	Name	Type	Reset value	Width	Function
`0x0000`	UART0DR	RW	-	32	Data Register.
`0x0004`	UART0RSR/UART0ECR	RW	`0x0000_0000`	32	Receive Status Register/Error Clear Register.
`0x0018`	UART0FR	RO	`0x0000_0012`	32	Flag Register.
`0x0020`	UART0ILPR	RW	`0x0000_0000`	32	IrDA Low-Power Counter Register.
`0x0024`	UART0IBRD	RW	`0x0000_0000`	32	Integer Baud Rate Register.
`0x0028`	UART0FBRD	RW	`0x0000_0000`	32	Fractional Baud Rate Register.
`0x002C`	UART0LCR_H	RW	`0x0000_0000`	32	Line Control Register.
`0x0030`	UART0CR	RW	`0x0000_0300`	32	Control Register.
`0x0034`	UART0IFLS	RW	`0x0000_0012`	32	Interrupt FIFO Level Select Register.
`0x0038`	UART0IMSC	RW	`0x0000_0000`	32	Interrupt Mask Set/Clear Register.
`0x003C`	UART0RIS	RO	`0x0000_0000`	32	Raw Interrupt Status Register.
`0x0040`	UART0MIS	RO	`0x0000_0000`	32	Masked Interrupt Status Register.
`0x0044`	UART0ICR	WO	-	32	Interrupt Clear Register.
`0x0048`	UART0DMACR	RW	`0x0000_0000`	32	DMA Control Register.
`0x0FE0`	UART0PeriphID0	RO	`0x0000_0011`	32	UART0 peripheral ID Register 0.

Table 3-121 UART control registers summary (continued)

Offset	Name	Type	Reset value	Width	Function
0x0FE4	UART0PeriphID1	RO	0x0000_0010	32	UART0 peripheral ID Register 1.
0x0FE8	UART0PeriphID2	RO	0x0000_0004	32	UART0 peripheral ID Register 2.
0x0FEC	UART0PeriphID3	RO	0x0000_0000	32	UART0 peripheral ID Register 3.
0x0FF0	UART0PCellID0	RO	0x0000_000D	32	UART0 component ID Register 0.
0x0FF4	UART0PCellID1	RO	0x0000_00F0	32	UART0 component ID Register 1.
0x0FF8	UART0PCellID2	RO	0x0000_0005	32	UART0 component ID Register 2.
0x0FFC	UART0PCellID3	RO	0x0000_00B1	32	UART0 component ID Register 3.
0x1000	UART1DR	RW	-	32	Data Register.
0x1004	UART1RSR/UART1ECR	RW	0x0000_0000	32	Receive Status Register/Error Clear Register.
0x1018	UART1FR	RO	0x0000_0012	32	Flag Register.
0x1020	UART1ILPR	RW	0x0000_0000	32	IrDA Low-Power Counter Register.
0x1024	UART1IBRD	RW	0x0000_0000	32	Integer Baud Rate Register.
0x1028	UART1FBRD	RW	0x0000_0000	32	Fractional Baud Rate Register.
0x102C	UART1LCR_H	RW	0x0000_0000	32	Line Control Register.
0x1030	UART1CR	RW	0x0000_0300	32	Control Register.
0x1034	UART1IFLS	RW	0x0000_0012	32	Interrupt FIFO Level Select Register.
0x1038	UART1IMSC	RW	0x0000_0000	32	Interrupt Mask Set/Clear Register.
0x103C	UART1RIS	RO	0x0000_0000	32	Raw Interrupt Status Register.
0x1040	UART1MIS	RO	0x0000_0000	32	Masked Interrupt Status Register.
0x1044	UART1ICR	WO	-	32	Interrupt Clear Register.
0x1048	UART1DMACR	RW	0x0000_0000	32	DMA Control Register.
0x1FE0	UART1PeriphID0	RO	0x0000_0011	32	UART1 peripheral ID Register 0.
0x1FE4	UART1PeriphID1	RO	0x0000_0010	32	UART1 peripheral ID Register 1.
0x1FE8	UART1PeriphID2	RO	0x0000_0004	32	UART1 peripheral ID Register 2.
0x1FEC	UART1PeriphID3	RO	0x0000_0000	32	UART1 peripheral ID Register 3.
0x1FF0	UART1PCellID0	RO	0x0000_000D	32	UART1 component ID Register 0.
0x1FF4	UART1PCellID1	RO	0x0000_00F0	32	UART1 component ID Register 1.

Table 3-121 UART control registers summary (continued)

Offset	Name	Type	Reset value	Width	Function
0x1FF8	UART1PCellID2	RO	0x0000_0005	32	UART1 component ID Register 2.
0x1FFC	UART1PCellID3	RO	0x0000_00B1	32	UART1 component ID Register 3.

3.13 GPIO control registers

The Musca-S1 test chip implements GPIO registers which control the GPIO interface.

Bits [15:0] control the Musca-S1 test chip I/O to the Arduino Expansion Shield interface. Bits [31:16] are reserved.

The base memory addresses of the GPIO control registers are:

- `0x4011_0000` in the Non-secure region.
- `0x5011_0000` in the Secure region.

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

The following table shows the GPIO control registers in the Musca-S1 test chip in address offset order from the base memory address. Undefined registers are reserved. Software must not attempt to access these registers.

Table 3-122 GPIO control registers summary

Offset	Name	Type	Reset	Width	Function
`0x0000`	GPIODATA	RW	`0x0000_0000`	32	Data value. Bits [31:16] are reserved.
`0x0004`	GPIODATAOUT	RW	`0x0000_0000`	32	Data output value. Bits [31:16] are reserved.
`0x0010`	GPIOOUTENSET	RW	`0x0000_0000`	32	Output enable set. Bits [31:16] are reserved.
`0x0014`	GPIOOUTENCLR	RW	`0x0000_0000`	32	Output enable clear. Bits [31:16] are reserved.
`0x0020`	GPIOINTENSET	RW	`0x0000_0000`	32	Interrupt enable set. Bits [31:16] are reserved.
`0x0024`	GPIOINTENCLR	RW	`0x0000_0000`	32	Interrupt enable clear. Bits [31:16] are reserved.
`0x0028`	GPIOINTTYPESET	RW	`0x0000_0000`	32	Interrupt type set. Bits [31:16] are reserved.
`0x002C`	GPIOINTTYPECLR	RW	`0x0000_0000`	32	Interrupt type clear. Bits [31:16] are reserved.
`0x0030`	GPIOINTPOLSET	RW	`0x0000_0000`	32	Polarity-level, edge IRQ configuration. Set interrupt polarity bit. Bits [31:16] are reserved.
`0x0034`	GPIOINTPOLCLR	RW	`0x0000_0000`	32	Polarity-level, edge IRQ configuration. Clear interrupt polarity bit. Bits [31:16] are reserved.
`0x0038`	GPIOINTSTATUS INTCLEAR	RW	`0x0000_0000`	32	Clear interrupt request. Bits [31:16] are reserved.

Table 3-122 GPIO control registers summary (continued)

Offset	Name	Type	Reset	Width	Function
0x0FD0	GPIOPID4	RW	0x0000_0000	32	Peripheral ID Register 4. Bits [31:8] are reserved.
0x0FE0	GPIOPID0	RW	0x0000_0000	32	Peripheral ID Register 0. Bits [31:8] are reserved.
0x0FE4	GPIOPID1	RW	0x0000_0000	32	Peripheral ID Register 1. Bits [31:8] are reserved.
0x0FE8	GPIOPID2	RW	0x0000_0000	32	Peripheral ID Register 2. Bits [31:8] are reserved.
0x0FEC	GPIOPID3	RW	0x0000_0000	32	Peripheral ID Register 3. Bits [31:8] are reserved.
0x0FF0	GPIOCID0	RW	0x0000_0000	32	Component ID Register 0. Bits [31:8] are reserved.
0x0FF4	GPIOCID1	RW	0x0000_0000	32	Component ID Register 1. Bits [31:8] are reserved.
0x0FF8	GPIOCID2	RW	0x0000_0000	32	Component ID Register 2. Bits [31:8] are reserved.
0x0FFC	GPIOCID3	RW	0x0000_0000	32	Component ID Register 3. Bits [31:8] are reserved.

3.14 Third-party IP

The Musca-S1 test chip implements third-party IP, including control registers.

The Musca-S1 test chip implements the following Cadence IP:

- QSPI controller (IP6514E), no DMA support:
  - Base memory address `0x4010_A000` in the Non-secure region.
  - Base memory address `0x5010_A000` in the Secure region.
- I^2C interface (IP6510):
  - I2C0: Base memory address `0x4010_4000` in the Non-secure region.
  - I2C0: Base memory address `0x5010_4000` in the Secure region.
  - I2C1: Base memory address `0x4010_5000` in the Non-secure region.
  - I2C1: Base memory address `0x5010_5000` in the Secure region.
- I^2S-MT/MR controller (IP6718E), three channels, master only:
  - Base memory address `0x4010_6000` in the Non-secure region.
  - Base memory address `0x5010_6000` in the Secure region.
- Pulse Width Modulator IP (IP6512):
  - PWM0: Base memory address `0x4010_7000` in the Non-secure region.
  - PWM0: Base memory address `0x5010_7000` in the Secure region.
  - PWM1: Base memory address `0x4010_E000` in the Non-secure region.
  - PWM1: Base memory address `0x5010_E000` in the Secure region.
  - PWM2: Base memory address `0x4010_F000` in the Non-secure region.
  - PWM2: Base memory address `0x5010_F000` in the Secure region.
- SPI master interface (IP6524), master only:
  - Base memory address `0x4010_A000` in the Non-secure region.
  - Base memory address `0x5010_A000` in the Secure region.

Contact your local Cadence representative for information about the QSPI, I2C, I2S, PWM, and SPI.

Appendix A
Signal descriptions

This appendix describes the signals that are present at the board interface connectors.

It contains the following sections:

A.1 Arduino Expansion Shield connectors

Connectors on the Musca-S1 development board provide one Shield expansion interface. The interface provides 16 digital I/O and six analog I/O. The digital and analog I/O operating voltage is 1V8 or 3V3.

Arduino Shield interface

The following figure shows the Arduino Shield interface connectors.

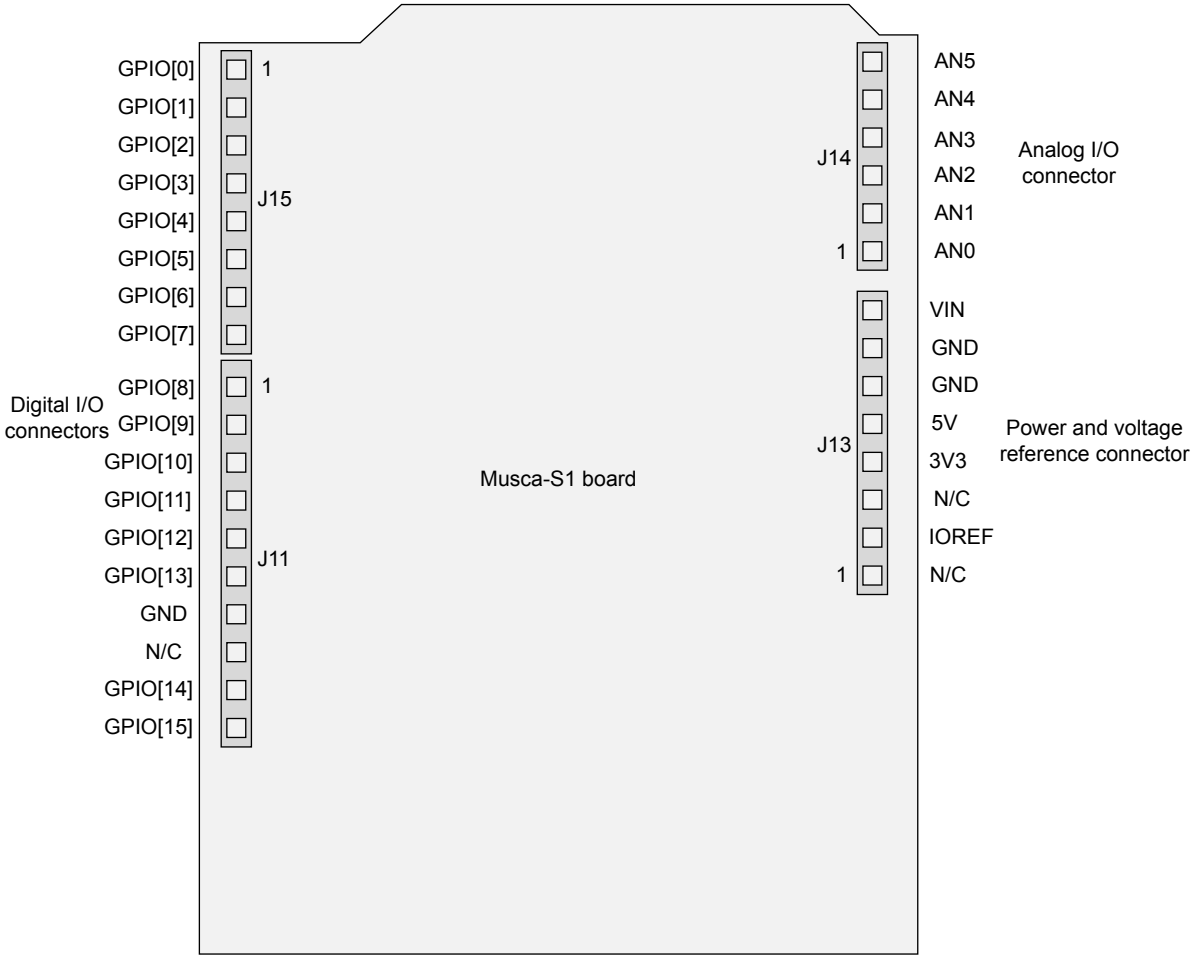

Figure A-1 Arduino Shield interface connectors

Digital I/O connectors, J11, J15

Connector J11 provides Shield digital I/O GPIO[15:8], and connector J15 provides Shield digital I/O GPIO[7:0]. Connector J11 also provides the analog I/O reference voltage.

The IOMUX registers select one of the Shield interface GPIO pin functions sets, ALTF1, ALTF2, or ALTF3. The IOMUX registers are part of the *Serial Configuration Control* (SCC) registers. See *3.11.1 IOMUX registers* on page 3-120 and *2.2.2 Test chip multiplexed I/O* on page 2-23.

The following table shows the pin mappings for connector J11.

Table A-1 Shield digital I/O connector J11 signal list

Pin	Primary reset or powerup	ALTF1	ALTF2	ALTF3
1	**GPIO[8]**	**MT_I2S_WS1**	UART1_RTS	Reserved
2	**GPIO[9]**	**MT_I2S_SCK**	UART0_SCLK	
3	**GPIO[10]**	**SPI0 nSS0**	Reserved	
4	**GPIO[11]**	**SPI0 MOSI**		
5	**GPIO[12]**	**SPI0 MISO**		
6	**GPIO[13]**	**SPI0 SCK**		
7	**GND**	**GND**	GND	GND
8	N/C	N/C	N/C	N/C
9	**GPIO[14]**	**I2C0 Data (SDA)**	Reserved	Reserved
10	**GPIO[15]**	**I2C0 Clock (SCL)**	GPIO[0]	

The following table shows the pin mappings for connector J15.

Table A-2 Shield digital I/O connector J15 signal list

Pin	Primary reset or powerup	ALTF1	ALTF2	ALTF3
1	**GPIO[0]**	**UART0 RxD**	Reserved	Reserved
2	**GPIO[1]**	**UART0 TxD**		
3	**GPIO[2]**	**MR_I2S_SD**	PWM0	
4	**GPIO[3]**	**MR_I2S_WS**	PWM1	
5	**GPIO[4]**	**MR_I2S_SCK**	PWM2	
6	**GPIO[5]**	**MT_I2S_SD0**	UART0_CTS	
7	**GPIO[6]**	**MT_I2S_WS0**	UART0_RTS	
8	**GPIO[7]**	**MT_I2S_SD1**	UART1_CTS	

Shield analog I/O connector J14

Connector J14 provides six analog I/O for the Arduino Expansion Shield.

The following table shows the pin mapping for connector J14.

Table A-3 Analog I/O connector J14 signal list

Pin	Signal
1	**AN[0]**
2	**AN[1]**
3	**AN[2]**
4	**AN[3]**

Table A-3 Analog I/O connector J14 signal list (continued)

Pin	Signal
5	**AN[4]**
6	**AN[5]**

Shield power and voltage reference connector J13.

Connector J13 provides power and voltage references for the Arduino Expansion Shield.

The following table shows the pin mapping for connector J13.

Table A-4 Shield power and voltage reference connector J13 signal list

Pin	Signal
1	N/C
2	**IOREF**
3	N/C
4	**3V3**
5	**5V**
6	**GND**
7	**GND**
8	**VIN**

Related information

1.3 The Musca-S1 development board at a glance on page 1-14
2.10 Arduino Expansion Shield interface on page 2-38

A.2 Debug connector

The Musca-S1 development board provides one 3V3 10-pin CoreSight debug connector. The connector supports Serial Wire or JTAG processor debug (SWJ - DP) to enable connection of a DSTREAM or Arm Keil ULINK-Plus™ debug adapter, or a compatible third-party debugger.

The following figure shows the 10-pin debug connector.

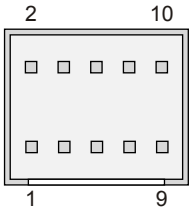

Figure A-2 Debug connector

The following table shows the pin mapping for the debug connector.

Table A-5 Debug connector pin mapping

Pin	Signal	Pin	Signal
1	**3V3**	2	**SWDIO/TMS**
3	**GND**	4	**SWDCLK/TCK**
5	**GND**	6	**SWO/TDO**
7	N/C	8	NC/**TDI**
9	**GNDDETECT**	10	**nSRST**

Note

Pins 2, 4, 6, 8, 9, and 10 have pullup resistors to **3V3**.

Related information

1.3 The Musca-S1 development board at a glance on page 1-14
2.13 Debug on page 2-42

A.3 USB connector

The Musca-S1 development board provides one mini-B USB connector that enables access to the CoreSight block in the Musca-S1 test chip. The connector also enables external 5V power to the board.

The following figure shows the USB connector.

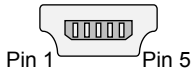

Figure A-3 Mini-B USB connector

The following table shows the pin mapping of the mini-B USB connector.

Table A-6 Mini-B USB connector

Pin	Signal	Pin	Signal
1	**5V**	2	**DATA-**
3	**DATA+**	4	**ID**
5	**GND**	6	**GND_EARTH**

Note

The **GND_EARTH** connection is the casing of the mini-B connector.

Related information

1.3 The Musca-S1 development board at a glance on page 1-14
2.8 Power on page 2-34

Appendix B
PVT sensors

This appendix describes the *Process, Voltage, and Temperature* (PVT) sensors on the Musca-S1 test chip.

It contains the following section:

B.1 PVT sensors

The Musca-S1 test chip implements three *Process, Voltage, and Temperature* (PVT) sensors to enable power and performance characterization of the SSE-200 subsystem and other integrated systems.

Main features of the PVT sensors

The PVT sensors consist of a series of 501-stage ring oscillators and associated blocks that are mapped to APB. The following figure shows a high-level view of the PVT system.

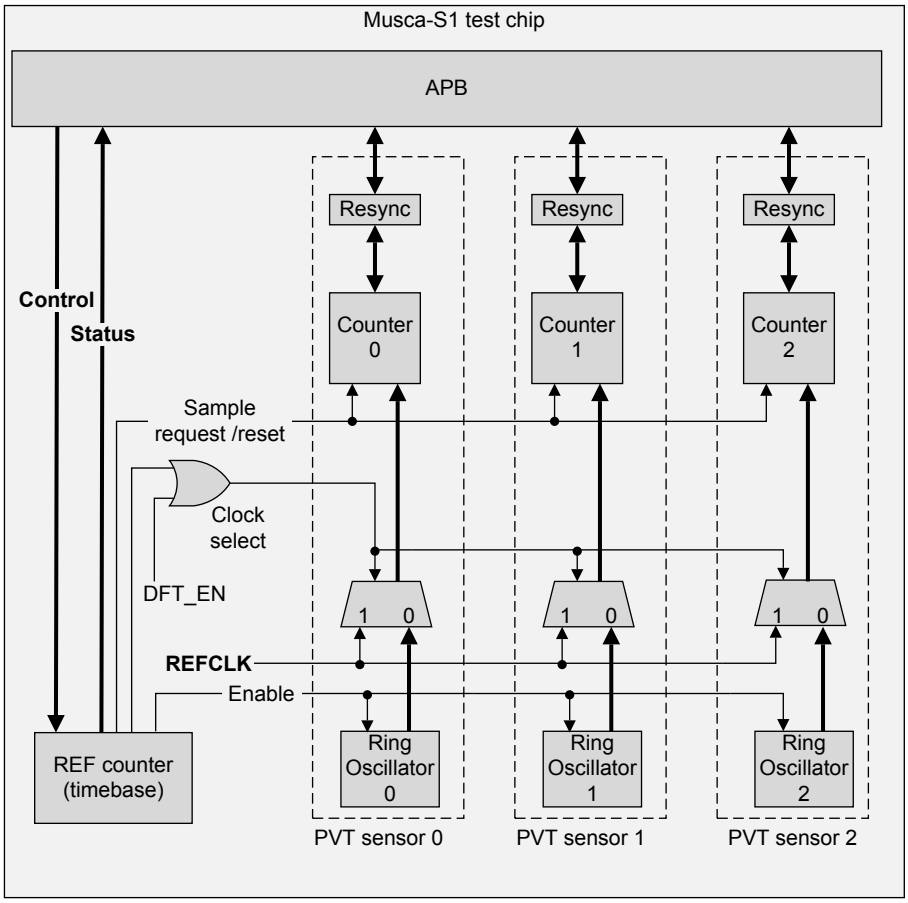

Figure B-1 PVT sensors and system

The PVT sensor system has the following main features:

- Polling of the sensor outputs by regularly checking the status flags.
- Event mode where the REF counter interrupt signal indicates that PVT measurements are ready.
- Synchronized start and enable.
- One-shot mode, or repeat mode.
- Each sensor is controllable independently of the others.
- Built-in test infrastructure.
- DFT bypass mode.
- Overflow indicator that enables timebase tuning.

Reference counter, timebase

The reference counter initiates and controls the PVT measurements. It stores the programmed measurement time window and when the reference counter reaches the programmed value it:

- Generates an interrupt if the interrupt is enabled.
- Requests that each enabled sensor stores the number of ring oscillator pulses that it receives during the measurement window.
- Restarts the PVT counters after the sensors have stored the measurements.

The reference counter has the following operating modes:

- One-shot mode: Performs a single measurement and then waits.
- Repeat mode: Repeats the measurements until controls signals stop them or the system is powered down.

Counter

The counters count and store the number of ring oscillator output pulses that they receive during the measurement window:

- The CTRL_ENABLE Register generates an enable signal for each sensor.
- The reference counter synchronizes the sensor counter measurement process.
- Each sensor counter has its own flags:
  - Sample flags to indicate that a new measurement value is ready.
  - Overflow to indicate that the sensor counter has reached `0xFFFF_FFFF`. The overflow flag stays HIGH until the reference counter initiates a new measurement.

Ring oscillators

The ring oscillators consist of 501 inverting stages and the PVT sensor control registers can enable or disable them. The ring oscillators are built from different inverting cells, NOT, NAND, and NOR gates to represent the main cells that are available. The following table shows the ring oscillators with their sensor ID, voltage domain, and location on the die.

Table B-1 PVT ring oscillators

Sensor ID	Cell	Voltage domain
0	inv_svt_c50	1V1
1	inv_lvt_c40	
2	inv_svt_c40	

Controlling and reading data from the PVT sensors

The PVT sensor control registers control and read data from the active PVT sensor. See *3.9 PVT sensor registers* on page 3-111.

Appendix C
IP configuration

This appendix describes the IP configuration of the Musca-S1 test chip.

It contains the following section:

C.1 IP configuration

The Musca-S1 test chip implements Arm CoreLink SSE-200 Subsystem version r1p0.

The following table shows the IP configuration of the Musca-S1 test chip

Table C-1 Musca-S1 test chip IP configuration

Product code	Product name	Version
CG062-BU-50000	Arm CoreLink SSE-200 Subsystem	r1p0-00eac0
AT623-BU-50000	Arm Cortex-M33 MCU	r0p2-00rel0
BP200-BU-00000	Arm Cortex-M0/M0+ System Design Kit	r1p0-00rel0
BP300-BU-50000	Arm CoreLink SIE-200 System IP for Embedded	r3p1-00rel0
PL408-BU-50000	Arm CoreLink LPD-500 Low Power Interface Distributor	r0p0-00rel0
TM100-BU-50000	Arm CoreSight SoC-400	r3p2-50rel1
TM976-BU-50000	Arm CoreSight ETM-M33	r0p2-00rel0
AT624-MN-22110	Arm Cortex-M33 FPU	r0p2-00rel0
BP210-BU-00000	Arm Cortex-M System Design Kit	r1p0-01rel0
CC010-BU-50000	Arm CryptoCell-312	r1p0-00eac0

Appendix D
Specifications

This appendix contains electrical specifications of the Musca-S1 development board.

It contains the following section:

D.1 Electrical specifications

The electrical specifications of the Musca-S1 development board are as follows:

See *2.8 Power* on page 2-34 for information on the Musca-S1 development board power supply rails and maximum current loads.

Appendix E
Revisions

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

It contains the following section:

- *E.1 Revisions* on page Appx-E-189.

E.1 Revisions

The following table lists the technical changes between released issues of this book.

Table E-1 Issue 101835_0000_00

Change	Location	Affects
No changes, first release.	-	-