ARM® IoT Subsystem for Cortex®-M

Revision: r0p0

Technical Reference Manual

ARM DDI 0551A (ID120415)

ARM IoT Subsystem for Cortex-M
Technical Reference Manual

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
13 November 2015	A	Non-Confidential	First release for r0p0 EAC.

Proprietary Notice

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20348

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.

Product Status

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

Web Address

`http://www.arm.com`

Contents

ARM IoT Subsystem for Cortex-M Technical Reference Manual

Preface

This preface introduces the *IoT Subsystem for Cortex-M Technical Reference Manual*. It contains the following sections:

- *About this book* on page vii.
- *Feedback* on page x.

About this book

This book is for the *IoT Subsystem for Cortex-M* (IoT Subsystem). It provides a high-level overview of the IoT Subsystem. It describes architectural information, and as such, facilitates the creation of IoT Subsystem software or an SoC targeted at an *Internet of Things* (IoT) application.

Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this book, for example, r0p0, where:

r*m* Identifies the major revision of the product, for example, r0.

p*n* Identifies the minor revision or modification status of the product, for example, p0.

Intended audience

This book is written for software engineers who want to work with an ARM reference platform. The manual describes the functionality of the IoT Subsystem.

Using this book

This book is organized into the following chapters:

Chapter 1 *Introduction*

Read this for a high-level view of the IoT Subsystem and a description of its features.

Chapter 2 *Functional Description*

Read this for a description of the major interfaces and components of the IoT Subsystem. This chapter also describes how the components operate.

Chapter 3 *Programmers Model*

Read this for a description of the address map and registers of the IoT Subsystem.

Appendix A *Signal Descriptions*

Read this for an overview of the signals present in the IoT Subsystem.

Appendix B *Revisions*

Read this for a description of 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.

The *ARM Glossary* is available on the ARM Infocenter at http://infocenter.arm.com/help/topic/com.arm.doc.aeg0014-/index.html.

Conventions

Conventions that this book can use are described in:

- *Typographical conventions* on page viii.

- *Timing diagrams*.
- *Signals* on page ix.

Typographical conventions

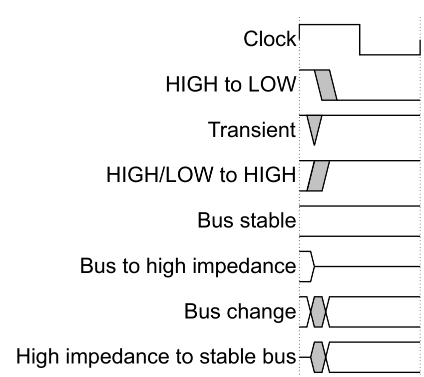
The following table describes the typographical conventions:

Style	Purpose
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.
`monospace`	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, UNKNOWN, and UNPREDICTABLE.

Timing diagrams

The figure named *Key to timing diagram conventions* 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.



Key to timing diagram conventions

Timing diagrams sometimes show single-bit signals as HIGH and LOW at the same time and they look similar to the bus change shown in *Key to timing diagram conventions*. If a timing diagram shows a single-bit signal in this way then its value does not affect the accompanying description.

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.

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

Additional reading

This section lists publications by ARM and by third parties.

See Infocenter, `http://infocenter.arm.com` for access to ARM documentation.

See `www.arm.com/cmsis` for embedded software development resources including the *Cortex® Microcontroller Software Interface Standard* (CMSIS).

See mbed, `https://mbed.org/` for information on the mbed tools including mbed OS and online tools.

ARM publications

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

- *Cortex®-M System Design Kit Technical Reference Manual* (ARM DDI 0479). http://infocenter.arm.com/help/topic/com.arm.doc.ddi0479c/index.html
- *Cortex®-M3 Devices Generic User Manual* (ARM DUI 0552). http://infocenter.arm.com/help/topic/com.arm.doc.dui0552a/index.html
- *ARM® Cortex®-M3 Technical Reference Manual* (ARM 100165_0201_00_en). http://infocenter.arm.com/help/topic/com.arm.doc.100165_0201_00_en/index.html

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

- *ARM® IoT for Cortex-M Implementation and Integration Manual* (ARM DII0300).
- *AMBA® 3 APB Protocol Specification* (ARM IHI 0024). http://infocenter.arm.com/help/topic/com.arm.doc.ihi0024c
- *AMBA® 3 AHB-Lite Protocol Specification* (ARM IHI 0033). http://infocenter.arm.com/help/topic/com.arm.doc.ihi0033a
- *ARM® ARMv7M Architecture Reference Manual* (ARM DDI 0403). http://infocenter.arm.com/help/topic/com.arm.doc.ddi0403e.b/index.html

Feedback

ARM welcomes feedback on this product and its documentation.

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.
- The number, ARM DDI 0551A.
- The page numbers to which your comments apply.
- 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 *IoT Subsystem for Cortex-M* (IoT Subsystem). It contains the following sections:

- *About IoT endpoints* on page 1-2.
- *Features of the IoT Subsystem* on page 1-4.
- *Compliance* on page 1-6.
- *Product revisions* on page 1-7.

1.1 About IoT endpoints

The IoT Subsystem delivers a reference pre-integrated, validated, hardware and software subsystem that can be extended to provide an IoT endpoint system.

Figure 1-1 shows an IoT system consisting of several endpoints and a shared control node:

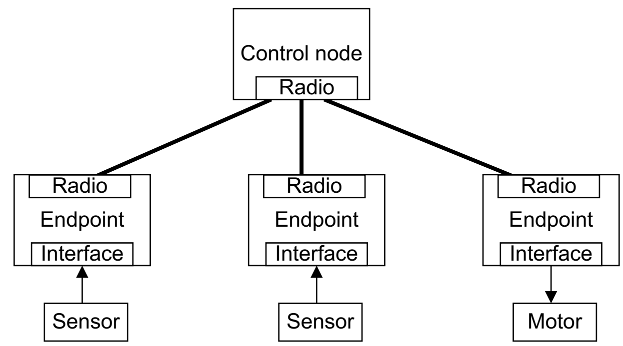


Figure 1-1 An IoT endpoint as part of a larger control system

Figure 1-2 shows a block diagram of the hardware and software in an endpoint solution:

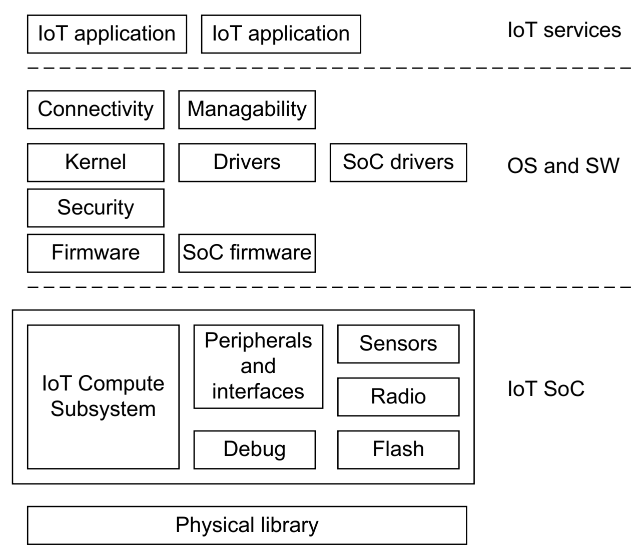


Figure 1-2 IoT endpoint HW and SW solution

A complete endpoint system typically contains the following components:

Compute Subsystem

The IoT Subsystem consists of the Cortex-M3 processor and associated bus, debug, controller, and interface logic supplied by ARM.

Reference system memory and peripherals

Additional memory, control, and peripheral components beyond the minimum IoT Subsystem components.

Licensees of the IoT Subsystem are provided with an example integration layer which includes implementations of eFlash and SRAM. The example integration layer provides a starting point for customizing an SoC.

Communication interface

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

The ARM Cordio® BT4 radio IP is available as an option for the IoT Subsystem. The example integration layer expansion ports are however technology independent and other radio devices could be used instead of the Cordio radio IP. Radio-specific interfaces such as clock, reset, and power control must be implemented at the SoC level.

Sensor or control component

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

Software development environment

ARM provides a complete software development environment which includes the mbed operating system, ARM or GCC compilers and debuggers, and firmware.

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

1.2 Features of the IoT Subsystem

The IoT Subsystem contains the following components:

- A Cortex-M3 processor:
  - Bit-banding enables using standard instructions to read or modify of individual bits. The default implementation does not include bit banding.
  - Eight MPU regions (optional)
  - NVIC providing deterministic, high-performance interrupt handling with a configurable number of interrupts.
  - *Wakeup Interrupt Controller* (WIC) with configurable number of WIC lines (optional). This is a latch-based WIC implementation, and not the standard Cortex-M3 WIC.
  - Little-endian memory addressing only (for compatibility with eFlash controller and Flash cache).

  For more information see the *ARM Cortex-M3 Technical Reference Manual*.

  The Cortex-M3 has a Processor Integration Layer (PIL) to simplify integration of the IoT Subsystem into a multiprocessor system with a SoC-level CoreSight subsystem.

- Configurable Debug and Trace as either:
  - Stand-alone system with a TPIU and a SWJDAP.
  - Full CoreSight integration over a DAP and the ATB buses.

- Multilayer AMBA AHB-Lite interconnect:
  - Low-latency interconnect bus matrix.
  - Two AHB-Lite initiator expansion ports for external AHB masters.
  - Two AHB-Lite target expansion ports for external AHB slaves.
  - Eleven APB4 target expansion ports (each with 4KB address space) to connect APB peripherals.

- Memory system, consisting of:
  - Integrated eFlash cache with configurable cache size from 512 bytes to 8kB. (The cache is two-way set associative instruction cache with a four-word cache line.)
  - Integrated eFlash controller for TSMC 55 ULP-TV2 eFlash.

    Note

    The IoT Subsystem can be easily modified to replace the supplied eFlash controller if a different eFlash technology is used in the SoC, but the warranty is void if the IoT Subsystem is changed.

  - Static memory (configurable as one to four 32KB banks) is provided in the example integration layer.
  - eFlash memory (banked as 2x128KB or 2x256KB) provided in the example integration layer.

- Two APB timers:
  - Interrupt generation when the counter reaches 0.
  - Each timer has an **TIMERnEXTIN** signal that can be used as an enable or external clock.
  - Configurable privileged access mode.

- Cordio BT4 Radio component (optional).
  - Fully integrated Bluetooth Smart controller sub-system IP block.

— Radio transceiver, baseband, integrated link layer (LL) controller.

— LL firmware up to the Host Controller Interface (HCI).

— Delivered as a hard macro (55nm TSMC) with a synthesizable integration wrapper.

— The Cordio BT4 IP is not provided with the IoT Subsystem, and must be separately licensed from ARM.

A third-party BlueTooth solution can be connected to the AHB expansion ports, but that will require customized software and firmware to support the product.

The reference system contains the peripherals required to support a rich OS. The components highlighted in Figure 1-3 are not provided by the IoT Subsystem. Other peripherals not included in the IoT Subsystem might be required for specific application areas.

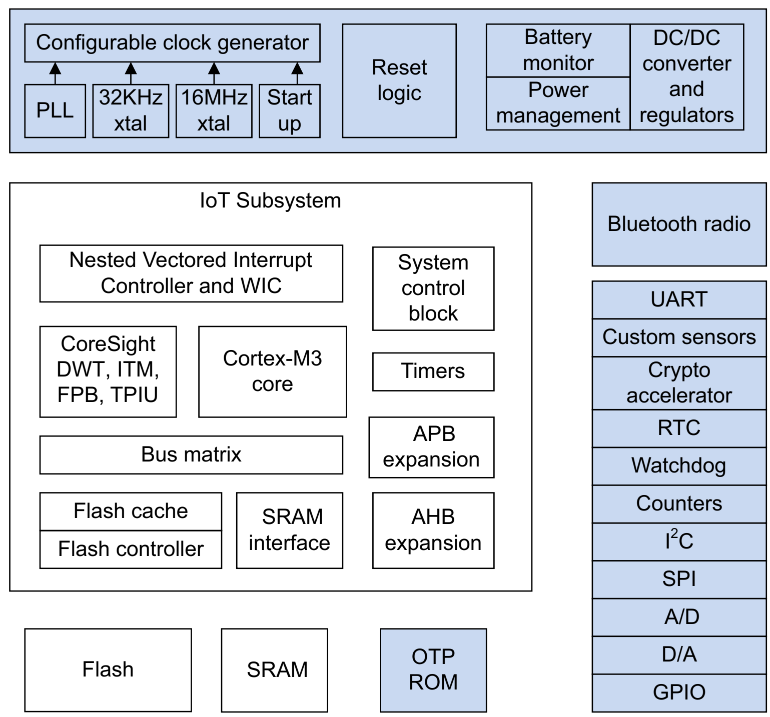


Figure 1-3 Example of an IoT endpoint SoC

1.3 Compliance

The IoT Subsystem complies with, or includes components that comply with, the following specifications:

- *ARM Architecture*.
- *Interrupt controller architecture*.
- *Advanced Microcontroller Bus Architecture*.

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

1.3.1 ARM Architecture

The IoT Subsystem implements the ARMv7-M architecture which executes the ARM-v7M Thumb instruction set.

See the *ARMv7-M Architecture Reference Manual* for more information.

1.3.2 Interrupt controller architecture

The IoT Subsystem implements the ARM *Nested Vectored Interrupt Controller* (NVIC).

See the *ARM Cortex-M3 Technical Reference Manual* for more information.

1.3.3 Advanced Microcontroller Bus Architecture

The IoT Subsystem complies with the:

- *Advanced High Performance Bus* (AHB-Lite) protocol. See the *AMBA® 3 AHB-Lite Protocol Specification*.
- *Advanced Peripheral Bus* (APB) protocol. See the *AMBA® APB Protocol Specification* (Rev 2.0).

1.4 Product revisions

This section describes the differences in functionality between product revisions:

1.0 First release.

Chapter 2
Functional Description

This chapter describes the functionality of the *IoT Subsystem for Cortex-M* (IoT Subsystem).

It contains the following sections:

2.1 System top-level partitioning

The IoT Subsystem consists of partitions of smaller sub-blocks.? The IoT Subsystem is extended by additional components in the SoC integration layer.

Note

The provided example system is for information only and ARM expects that the system designers will customize it for their application requirements.

The figure below shows the top-level block diagram with the AHB-Lite and APB bus interconnections.

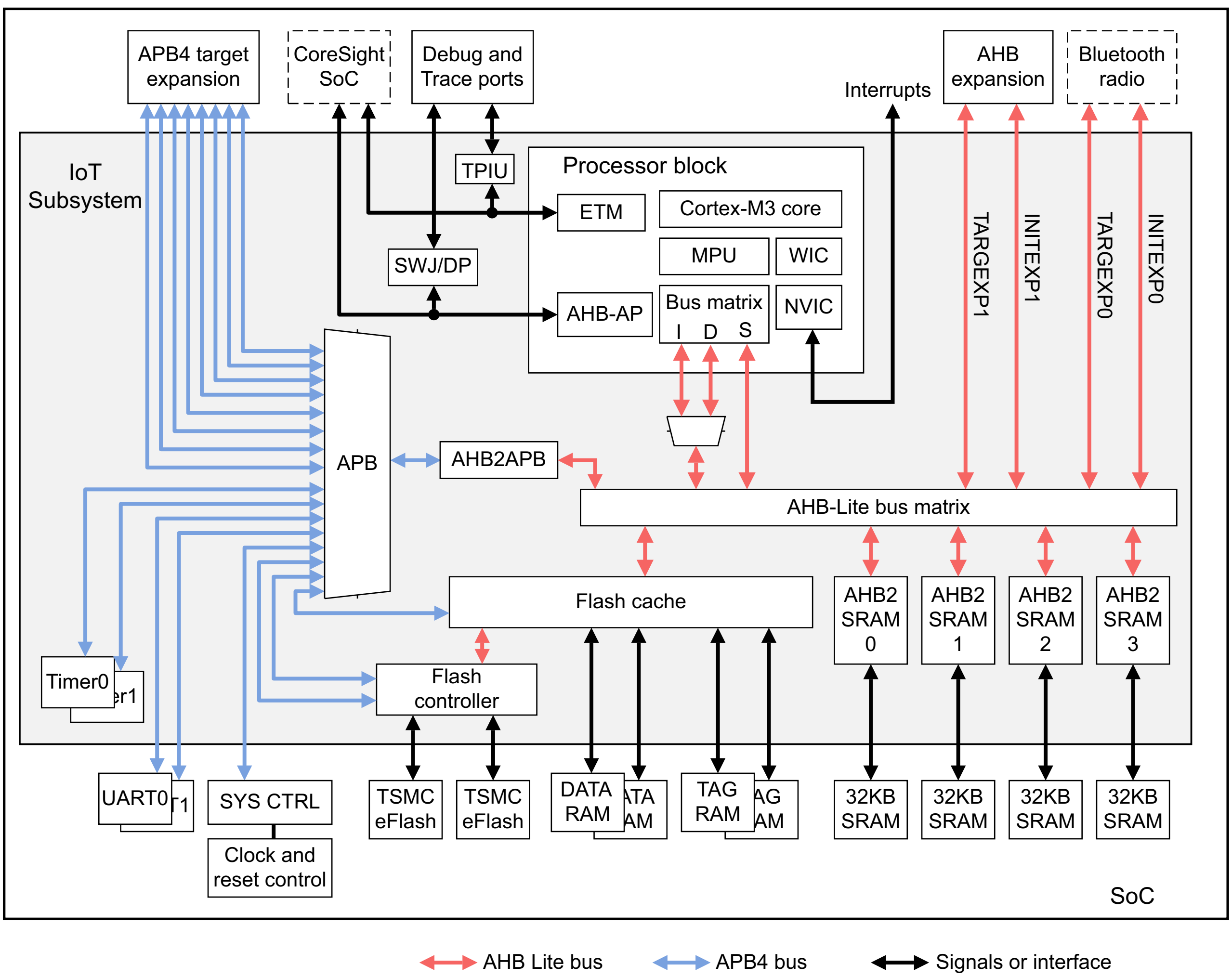


Figure 2-1 Bus interconnections

2.2 Cortex-M3 processor block

The block diagram for the Cortex-M3 processor logic and CoreSight SoC interface is shown in the figure below:

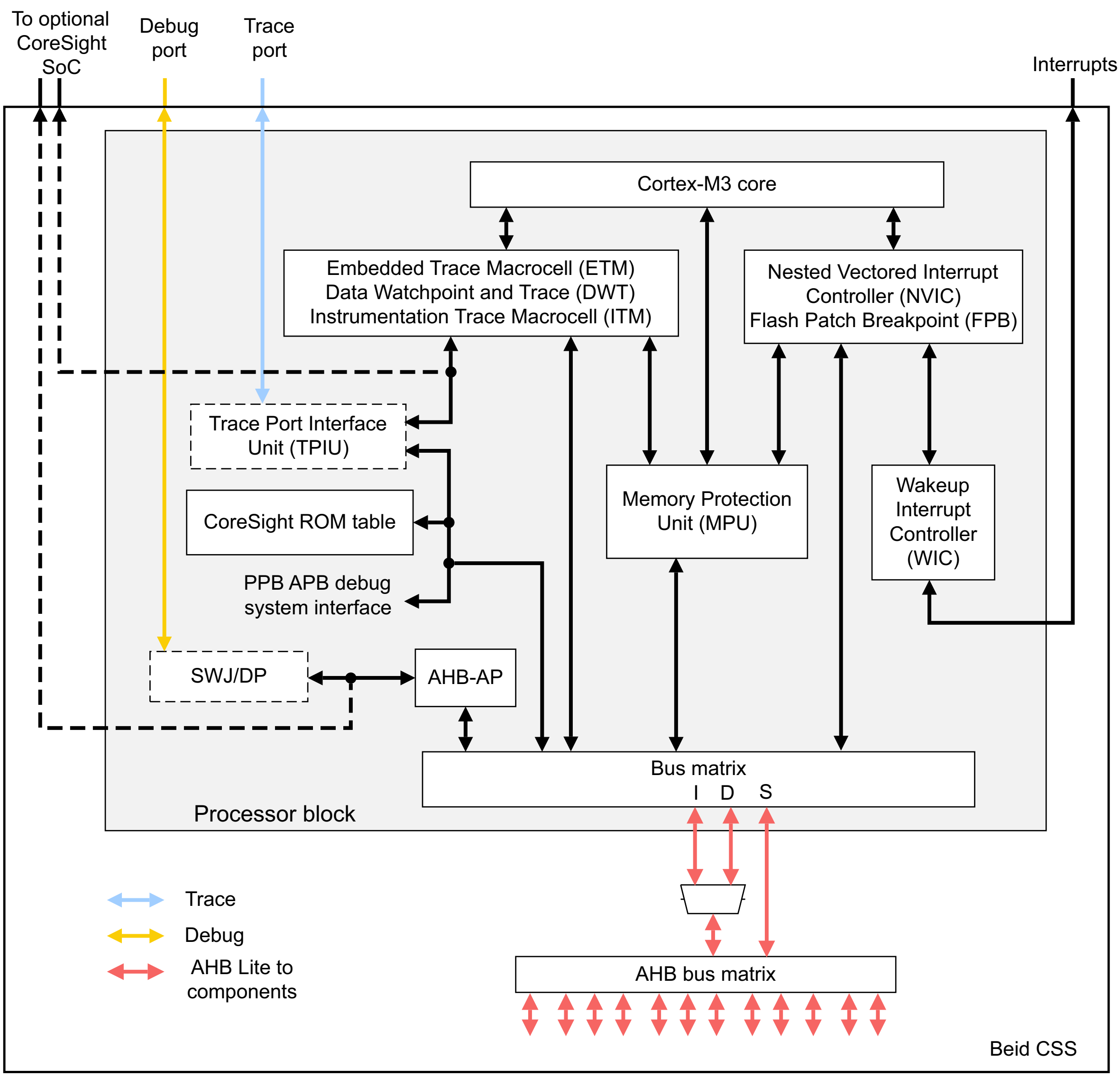


Figure 2-2 Cortex-M3 component

Note

The default implementation reuses the TPIU and SWJDP from Cortex-M3 package and connects the SWJ-DP and TPIU to the Processor Integration layer. For basic usage, there is no requirement to license the CoreSight SoC IP.

The system designer can however choose to design a system with the separately licensed CoreSight SoC debug interface connected to the AHB-AP. In this case the SWJ/DP and TPIU blocks and their corresponding signals are not present.

For more information on the Cortex-M3 and the debug and trace logic, see the following documents:

- *ARMv7-M Architecture Reference Manual (ARM DDI 0403)*.
- *ARM Cortex-M3 Processor Technical Reference Manual (ARM 100165_0201_00_en)*.
- *ARM CoreSight Components Technical Reference Manual (ARM DDI 0314)*.
- *ARM Debug Interface v5 Architecture Specification (ARM IHI 0031)*.
- *ARM Embedded Trace Macrocell Architecture Specification (ARM IHI 0014)*.

2.3 Power management

Low-power operation is essential for most IoT endpoint devices which typically rely on a battery or on harvested energy.

The IoT Subsystem is single power domain system and it does not support control of different power domains within the SoC. It does however define the necessary HW handshake signals that can be connected to a Power Management Unit (PMU) at the SoC level.

SRAM power management is not present in the IoT Subsystem because SRAM models are outside of the IoT Subsystem block. Memory power mode support can be implemented at SoC level.

For power-management signals, see *CPU control, status, and power management signals* on page A-20.

2.4 Clocks

The IoT Subsystem does not implement a clock control infrastructure. The IoT Subsystem is a single clock domain system that provides inputs for the clocks. Therefore all input clocks (except for the debug clocks) are identical in frequency and phase.

The target frequencies for the IoT Subsystem and associated components are:

- The typical configuration is 50MHz when using the TSMC 55 ULP process.
- The minimum operating frequency is 1MHz because of restrictions from the eFlash controller and the eFlash memory.
- 20MHz for the JTAG and Trace components.
- 32kHZ low-power mode (optional). If present, this clock is sampled by FLSHCLK and the rising edge is used as an enable signal for the eFlash erase timing counters. This reduces the toggle rate and allows RTL gating of the erase timer.

2.4.1 Component clocks

The subsystem does not implement architectural clock gating other than the Cortex-M3 and ETM internal gating. The IoT Subsystem is a single clock domain. The component clocks can however be gated by a custom implementation at SoC level.

For a full list of component clocks, see *Clock and reset signals* on page A-2.

For a list of power-management related signals, see *Power management* on page 2-5.

2.5 Resets

The IoT Subsystem has no internal reset generation implemented (except that Cortex-M3 CPU can be reset by the internal AIRCR.VECTRESET MMR bit of the NVIC).

All component resets in the IoT Subsystem are connected to the IoT Subsystem boundary and can therefore be reset using reset input signals.

Note

The IoT Subsystem is not designed to handle arbitrary reset patterns. The SoC integration and software must ensure that all resets are cleanly released before functional operation and no software reset is triggered to functioning components.

All resets are active low and asynchronous. External reset synchronization is required to guarantee the clean de-assertion of the resets in sync with the corresponding clocks.

2.5.1 Reset inputs

For a full list of component reset signals, see *Clock and reset signals* on page A-2.

2.5.2 About boot after reset

There is one Cortex M3 CPU integrated into the IoT Subsystem. After CPU reset de-assertion the CPU starts fetching the addresses as follows:

1. `0x00000000`: Fetch the stack pointer to initialize the SP register
2. `0x00000004`: Fetch the reset vector and jump to the reset vector value
3. *Reset vector*: Start boot code execution

Address `0x00000000` of the IoT Subsystem is mapped to the eFlash controller statically. It is therefore not possible to directly boot from ROM attached to the AHB expansion port.

eFlash reference cell erase is performed during wafer testing. If the flash is not empty, the factory reset must be applied before first use. Because the eFlash main array will be then be empty, the initial reset vector, SP and boot code must be written to eFlash through the debugger.

Note

An alternative way to load the Flash content after wafer testing is to preload it with the Flash DFT controller.

2.5.3 Events

The following table lists events that can be used for multiprocessor systems:

Table 2-1 Cortex-M3 events

Name	Description
CPU0RXEV	RX event input of the Cortex-M3. Causes a wake-up from a WFE instruction. Connect to TXEVs from other processors in a multi-processor system. Input from OR'ing TXEV signals from other processors in the system. If different processors run at different frequencies then synchronizers must be used to guarantee that TXEV is synchronous to this processor. TXEV must also be a single-cycle pulse. Tie to 0 if not used.
CPU0TXEV	TX event output of the Cortex-M3 Event transmitted as a result of SEV instruction. This is a single-cycle pulse. You can use it to implement a more power efficient spin-lock in a multi-processor system. In a multi-processor system, TXEV from each processor can be broadcast to the RXEV input of the other processors.

Note

The system designer can configure the IoT Subsystem with or without support for CoreSight SoC.

If CoreSight SoC is enabled, the Processor Integration Layer is exposed to the top of the IoT Subsystem and can be connected to existing systems to form a multi core system. Event ports are used as described in the table.

If CoreSight SoC is disabled, the Cortex-M3 is expected to be a standalone (single) core. The system designer might chose to connect the RX event port to DMA done signals.

2.6 Timer

The IoT Subsystem includes two instances of APB timers. These are required to satisfy the mbed OS requirements.

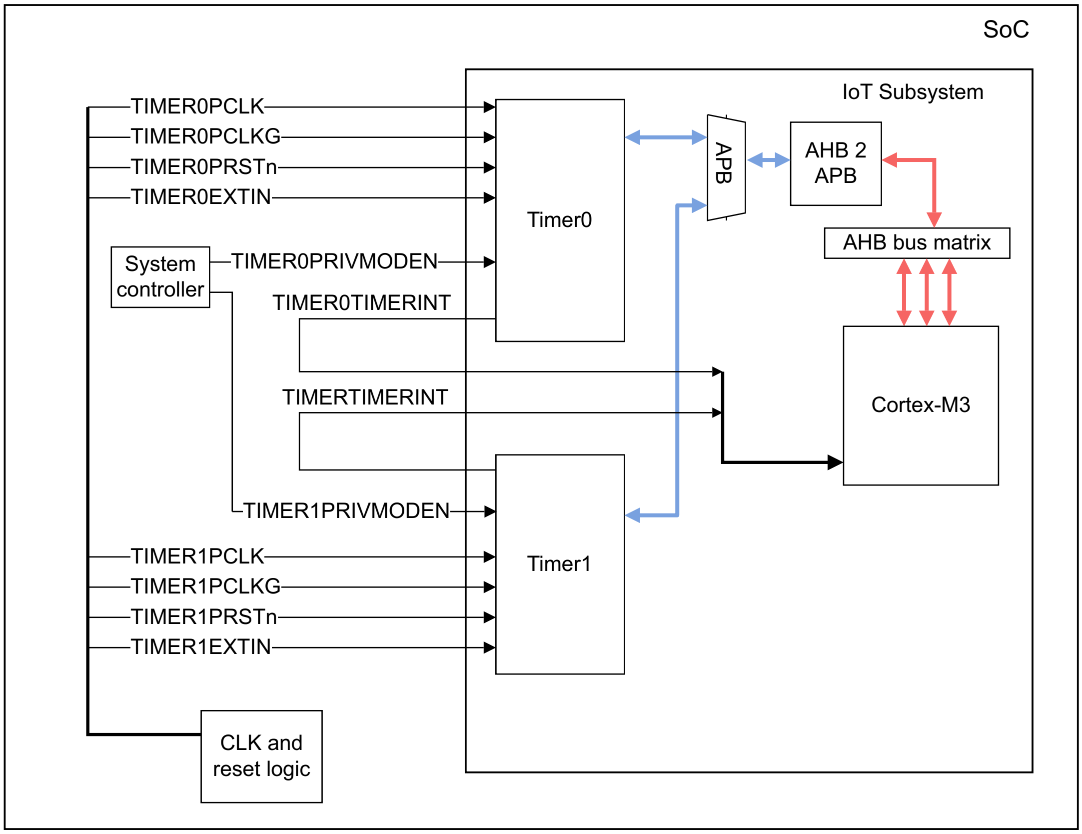


Figure 2-3 Timer interfaces

See also *Timer signals* on page A-10.

2.6.1 Security extension

Privilege mode enable signals determine whether only privileged accesses or both privileged and non-privileged accesses can write to the timer MMRs.

Table 2-2 Privilege mode enable input signals

Signal	Clock	Description
TIMER0PRIVMODEN	**TMER0PCLK**	Defines if the timer memory mapped registers are writeable only by privileged access: • 0: Non privileged access can write MMRs • 1: Only Privileged access can write MMRs
TIMER1PRIVMODEN	**TIMER1PCLK**	Defines if the timer memory mapped registers are writeable only by privileged access: • 0: Non privileged access can write MMRs. • 1: Only Privileged access can write MMRs

2.7 eFlash memory subsystem

The figure below shows the connections to the flash subsystem.

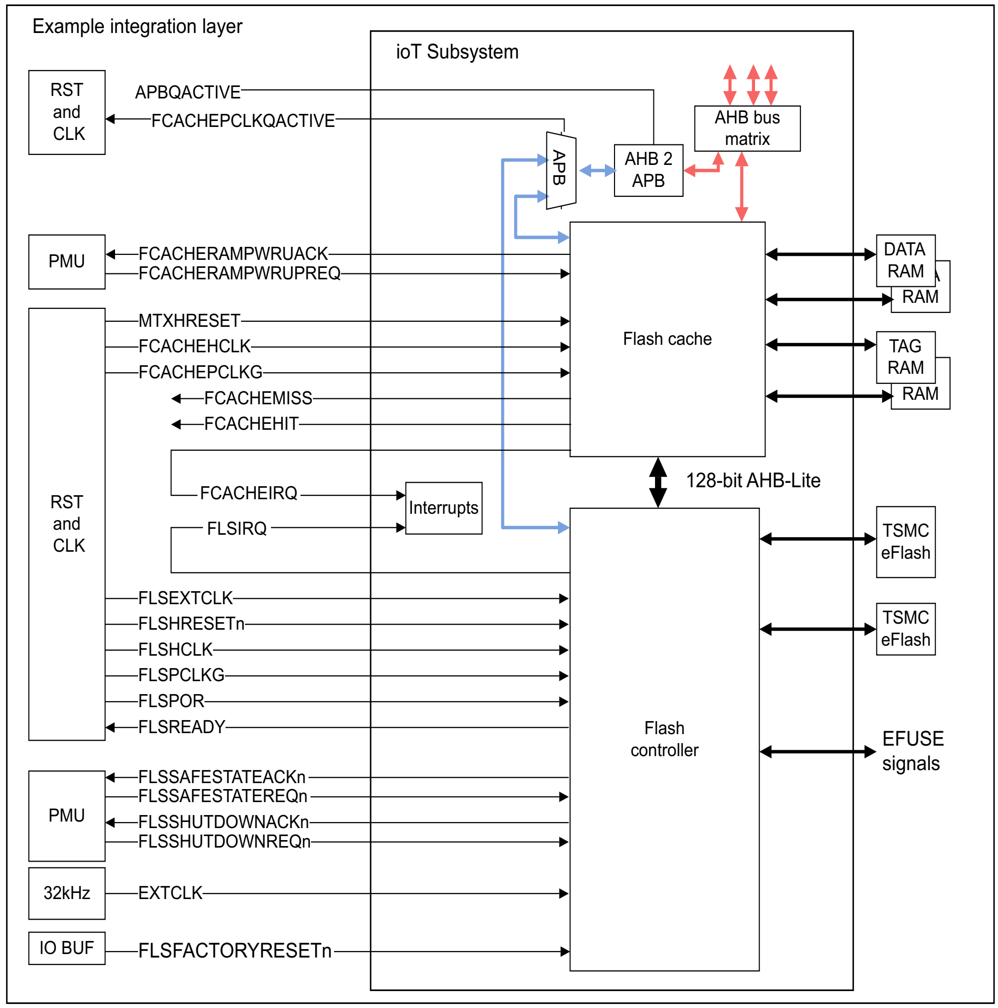


Figure 2-4 eFlash interface

The system designer can select generation of status registers that register cache hits and misses. If more detailed statistics are required, the designer can use the **FCACHEHIT** and **FCACHEMISS** signals and implement custom statistics collection logic. See also *eFlash signals* on page A-5.

2.7.1 eFlash cache

The IoT Subsystem includes an AHB instruction cache connected to the eFlash Controller to reduce eFlash accesses due to *execute in place* instructions fetches. This reduces the power consumption of the IoT Subsystem.

Note

Because the cache is only for instructions, AHB writes are bypassed and ignored. If a write changes program memory, software must invalidate the cache region.

The eFlash cache has the following features:

- Configurable cache size (minimum 512 bytes).
- Two way set associative.
- Configurable address bus size (based on flash memory size) so that tag memory size can be minimized.
- 128-bit AHB master to the AHB-Lite slave in the eFlash controller.
- 32 bit APB interface for configuration/status and write access
- Automatic/Manual power up.

  If the cache is powered down, the RAMs might be in retention. During power up the software can avoid invalidation of the cache RAMs and therefore save energy.
- Automatic/Manual invalidate cache RAM.
- Optional run-time configurable pre-fetcher.
- Compile time configurable performance counters accessible by SW.
- PMU interface supporting SRAM power-down or retention modes.

2.7.2 TSMC eFlash controller

The features of the eFlash controller include:

- 128-bit AHB slave interface to connect with eFlash cache master for read accesses.
- 32 bit APB interface for configuration/status and write access
- One IRQ line to notify status changes to the CPU and optionally let the CPU sleep or wake up while the eFlash is being programmed/erased.
- Supports two banks of flash memory as 2x128KB or 2x256KB.
- Supports eFlash Info page and Trim page with automatic self-repair function, and emulated security fuses.
- Factory reset request can perform an autonomous mass erase.
- Compatible with TSMC ULP55-TV2 embedded flash macros.
- Can be implemented with support for external low-power 32KHz clock to sequence program and erase operations.

2.8 Banked SRAM subsystem

The IoT Subsystem infrastructure supports up to four 32KB SRAMs. At least one 32KB SRAM must be implemented (SRAM bank 0).

If a SRAM bank is not implemented and the corresponding MTXREMAP bits are 1, then the corresponding address space is mapped to the AHB Slave expansion port of the interconnect.

In Figure 2-5 on page 2-13, the MTXREMAP signals from the configuration logic are static during functional operation.

The SRAM modules are outside of the IoT Subsystem. The IoT Subsystem does not implement retention or power-down supports for the SRAMs. It is the responsibility of the SoC integration to implement power domains for the SRAM, control the power modes of the SRAM banks with a SoC level PMU.

The AHB2SRAM bridge always responds with OKAY to all AHB accesses, even if the connected SRAM is not functional because for example it is powered-down or in retention mode.

It is the responsibility of the software to not read or write the SRAMs when they are not functional. If the SW tries to access non-accessible SRAM, the AHB2SRAM bridge implementation ensures that system will not go to deadlock state because of a non-responsive SRAM. Read data is implementation specific.

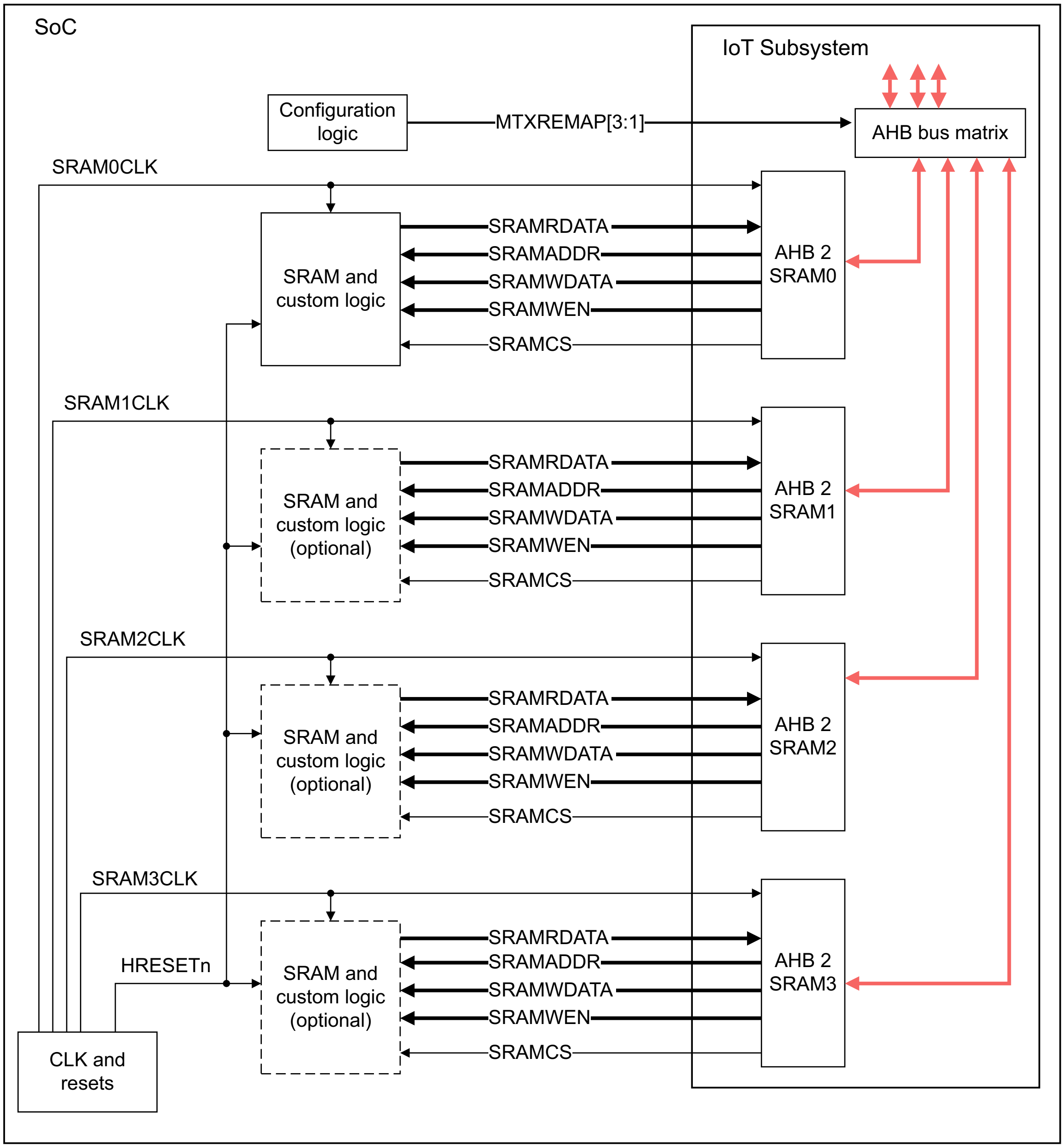


Figure 2-5 SRAM interface

See also *SRAM signals* on page A-9.

2.9 AHB and APB expansion

The AHB and APB bus structure is shown in the figure below:

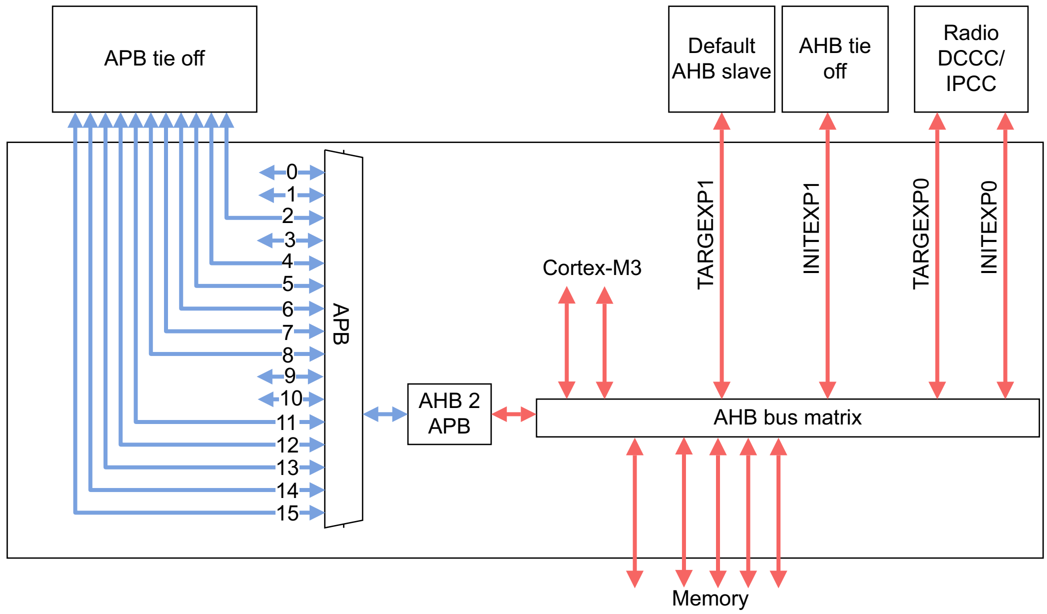


Figure 2-6 AHB and APB expansion buses

The APB and AHB tie off connections are for unused ports, and it is expected that these will be removed during synthesis.

See also *Bus signals* on page A-11.

2.9.1 APB slave multiplexer

The APB slave multiplexer, supports sixteen APB slaves in the IoT Subsystem. All ports are synchronous to the AHB expansion ports. Five APB ports are used for internal usage and eleven APB ports are hooked-up to the IoT Subsystem boundary and can be connected to external peripherals. The table below shows the usage of the APB ports in the IoT Subsystem:

Table 2-3 APB ports

APB port	Connection
PORT0	TIMER0
PORT1	TIMER1
PORT2	APBTARGEXP2
PORT3	eFlash cache
PORT4	APBTARGEXP4
PORT5	APBTARGEXP5
PORT6	APBTARGEXP6
PORT7	APBTARGEXP7
PORT8	APBTARGEXP8

Table 2-3 APB ports (continued)

APB port	Connection
PORT9	eFlash controller
PORT10	eFlash controller
PORT11	APBTARGEXP11
PORT12	APBTARGEXP12
PORT13	APBTARGEXP13
PORT14	APBTARGEXP14
PORT15	APBTARGEXP15

If an APB expansion interface is not used to connect a peripheral, the port must be tied off properly at SoC integration level.

2.9.2 AHB expansion

This section describes the AHB expansion features of the IoT Subsystem.

AHB initiator ports

Two AHB INITIATOR ports permit external AHB masters to be connected to the IoT Subsystem. These ports are prefixed with INITEXP<0..1>:

- INITEXP0 port is reserved for the AHB DMA master port of the BlueTooth Radio.
- INITEXP1 port can be used to connect any additional AHB master to the system.

Note

If one of the initiator ports is not used, then it must be tied off properly at SoC integration level.

AHB target ports

Two AHB TARGET ports permit external AHB masters to be connected to the IoT Subsystem. These ports are prefixed with TARGETEXP<0..1>:

- TARGETEXP0 port is reserved for the AHB DMA slave port of the BlueTooth Radio.
- TARGETEXP1 port can be used to connect any additional AHB slave to the system.

Note

If one of the target ports is not used, then the default slave must be connected to the port.

2.10 Debug and Trace

The SWJ-DP is a combined JTAG-DP and SW-DP that enables you to connect either an SWD or JTAG probe to a target. It is the standard CoreSight debug port.

To make efficient use of package pins, the JTAG pins use an auto-detect mechanism that switches between JTAG-DP and SW-DP depending on which probe is connected.

The Cortex-M3 TPIU is an optional component that acts as a bridge between the on-chip trace data from the *Embedded Trace Macrocell* (ETM) and the *Instrumentation Trace Macrocell* (ITM), with separate IDs, to a data stream. The TPIU encapsulates IDs where required, and the data stream is then captured by a *Trace Port Analyzer* (TPA). The Cortex-M3 TPIU is specially designed for low-cost debug.

Note

The default implementation reuses the TPIU and SWJ/DP from the Cortex-M3 package. This configuration is sufficient for basic use.

For more sophisticated multi-processor debug solution, a full CoreSight SoC IP solution can be licensed and implemented. If the CoreSight SoC option is selected, the TPIU and SWJ/DP blocks and corresponding interface signals are not present.

See also *Debug and Trace signals* on page A-14.

Chapter 3
Programmers Model

This chapter describes the *IoT Subsystem* memory regions and registers, and provides information on how to program a SoC that contains an implementation of the IoT Subsystem.

It contains the following sections:

3.1 About this programmers model

The following information applies to all registers:

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

  RW Read and write.

  RO Read-only.

  WO Write-only.

3.2 Memory map

The memory map for the IoT Subsystem is shown in the figure below:

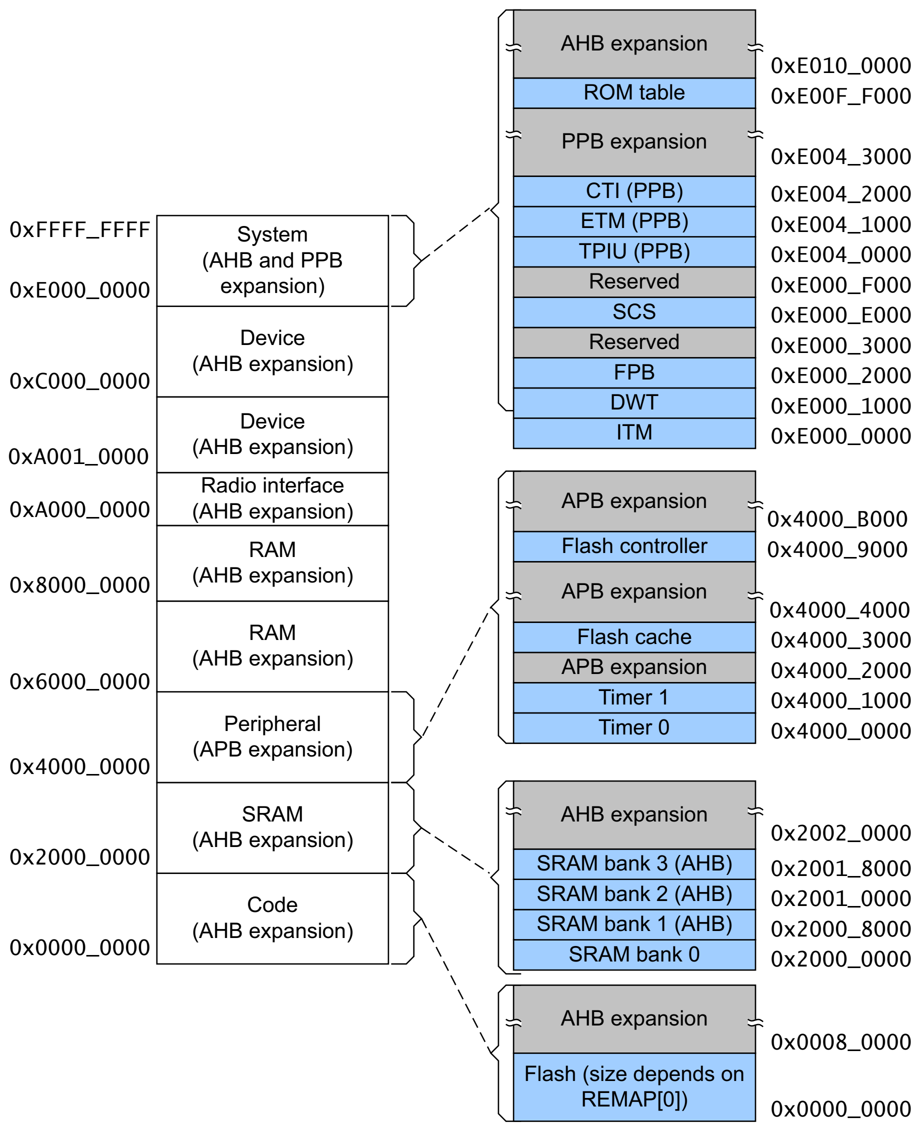


Figure 3-1 IoT Subsystem top-level memory map

3.2.1 Remap

The remapping feature of the bus matrix provides the following remapping options:

REMAP[0] The Embedded flash memory region represents the maximum supported eFlash size: 512KB. If the size of the actual eFlash banks is 256kB, the upper 256kB can be remapped to the AHB expansion port as follows:

- 0: 512KB eFlash
- 1: 256KB eFlash upper 256kbytes mapped to AHB expansion port

REMAP[1] If SRAM1 is not present the corresponding memory range can be mapped to AHB expansion port as follows:

- 0: SRAM1 present
- 1: SRAM1 not available, mapped to AHB expansion port

REMAP[2] If SRAM2 is not present the corresponding memory range can be mapped to AHB expansion port as follows:

- 0: SRAM2 present
- 1: SRAM2 not available, mapped to AHB expansion port

REMAP[3] If SRAM3 is not present the corresponding memory range can be mapped to AHB expansion port as follows:

- 0: SRAM3 present
- 1: SRAM3 not available, mapped to AHB expansion port

The PPB address region of the Cortex-M3 memory map is assigned to the default slave of the MTX thus returns SLVERR response.

Table 3-1 Code and SRAM regions

Region	Start Address	End Address	Peripheral name	Size	AHB bus matrix port	Remap 3	2	1	0	Comment
Code	`0x0`	`0x0003FFFF`	eFlash 2x128K	256K	TARG_FLASH0	-	-	-	-	Actual size is implementation defined
	`0x00040000`	`0x0007FFFF`	eFlash 2x256K	256K	TARG_FLASH0	-	-	-	0	Implementation defined
			AHB Expansion	2x128K	TARG_EXP1	-	-	-	1	-
	`0x00080000`	`0x01FFFFFF`	AHB Expansion	511M	TARG_EXP1	-	-	-	-	-
SRAM	`0x20000000`	`0x20007FFF`	SRAM0	32K	TARG_SRAM0	-	-	-	-	Bit band region
	`0x20008000`	`0x2000FFFF`	SRAM1	32K	TARG_SRAM1	-	-	0	-	Bit band region
			AHB Expansion	32K	TARG_EXP1	-	-	1	-	Bit band region
	`0x20010000`	`0x20017FFF`	SRAM2	32K	TARG_SRAM2	-	0	-	-	Bit band region
			AHB Expansion	32K	TARG_EXP1	-	1	-	-	Bit band region
	`0x20018000`	`0x2001FFFF`	SRAM	32K	TARG_SRAM3	0	-	-		Bit band region
			AHB Expansion	32K	TARG_EXP1	1	-	-		Bit band region
	`0x20020000`	`0x3FFFFFFF`	AHB Expansion	511M	TARG_EXP1	-	-	-	-	Bit band region `0x20020000` to `0x200FFFFF` Bit band alias `0x22000000` to `0x23FFFFFF`

3.2.2 Peripheral, expansion, and system regions

Table 3-2 Expansion and system map

Type	Start	End	Peripheral	Size	AHB bus matrix	Bus fabric	Comment
Periph	0x40000000	0x40000FFF	Timer0	4KB	TARG_APB0	APB port 0	Bit band region
Periph	0x40001000	0x40001FFF	Timer1	4KB	TARG_APB0	APB port 1	Bit band region
Periph	0x40002000	0x40002FFF	APB expansion	4KB	TARG_APB0	APB port 2	Bit band region
Periph	0x40003000	0x40003FFF	eFlash cache	4KB	TARG_APB0	APB port 3	Bit band region
Periph	0x40004000	0x40004FFF	APB expansion	4KB	TARG_APB0	APB port 4	Bit band region Optional UART0
Periph	0x40005000	0x40005FFF	APB expansion	4KB	TARG_APB0	APB port 5	Bit band region Optional UART1
Periph	0x40006000	0x40006FFF	APB expansion	4KB	TARG_APB0	APB port 6	Bit band region
Periph	0x40007000	0x40007FFF	APB expansion	4KB	TARG_APB0	APB port 7	Bit band region
Periph	0x40008000	0x40008FFF	APB expansion	4KB	TARG_APB0	APB port 8	Bit band region
Periph	0x40009000	0x40009FFF	eFlash controller MMRs	4KB	TARG_APB0	APB port 9	Bit band region
Periph	0x4000A000	0x4000A7FF	eFlash Ctrl Info page 0	4KB	TARG_APB0	APB port 10	Bit band region
Periph	0x4000A800	0x4000AFFF	eFlash Ctrl Info page 1	4KB	TARG_APB0	APB port 10	Bit band region
Periph	0x4000B000	0x4000BFFF	APB expansion	4KB	TARG_APB0	APB port 11	Bit band region
Periph	0x4000C000	0x4000CFFF	APB expansion	4KB	TARG_APB0	APB port 12	Bit band region
Periph	0x4000D000	0x4000EFFF	APB expansion	4KB	TARG_APB0	APB port 13	Bit band region
Periph	0x4000E000	0x4000EFFF	APB expansion	4KB	TARG_APB0	APB port 14	Bit band region
Periph	0x4000F000	0x4000FFFF	APB expansion	4KB	TARG_EXP1	APB port 15	Bit band region
Periph	0x40010000	0x5FFFFFFF	AHB expansion	511MB	TARG_APB0	APB port 15	Bit band region
RAM	0x60000000	0x7FFFFFFF	AHB Expansion	512MB	TARG_EXP1	-	-
RAM	0x80000000	0x9FFFFFFF	AHB Expansion	512MB	TARG_EXP1	-	-
Device	0xA0000000	0xA000FFFF	AHB Expansion	64KB	TARG_EXP0	-	Radio ICCC
Device	0xA0010000	0xBFFFFFFF	AHB Expansion	511MB	TARG_EXP1	-	-
Device	0xC0000000	0xDFFFFFFF	AHB Expansion	512MB	TARG_EXP1	-	-

Table 3-2 Expansion and system map (continued)

Type	Start	End	Peripheral	Size	AHB bus matrix	Bus fabric	Comment
System	0xE0000000	0xE0000FFF	ITM (or reserved)	4K	Default slave	PPB-Internal	Implementation defined
	0xE0001000	0xE0001FFF	DWT (or reserved)	4K	Default slave	PPB-Internal	Implementation defined
	0xE0002000	0xE0002FFF	FPB (or reserved)	4K	Default slave	PPB-Internal	Implementation defined
	0xE0003000	0xE000DFFF	Reserved	44K	Default slave	PPB-Internal	-
	0xE000E000	0xE000EFFF	SCS (or reserved)	4K	Default slave	PPB-Internal	Implementation defined
	0xE000F000	0xE003FFFF	Reserved	196K	Default slave	PPB-Internal	-
	0xE0040000	0xE0040FFF	TPIU (or PPB expansion)	4K	Default slave	PPB-External	Implementation defined
	0xE0041000	0xE0041FFF	ETM (or PPB expansion)	4K	Default slave	PPB-External	Implementation defined
	0xE0042000	0xE0042FFF	CTI (or PPB expansion)	4K	Default slave	PPB-External	Implementation defined
	0xE0043000	0xE0043FFF	PPB expansion	4K	Default slave	PPB-External	-
	0xE0044000	0xE0044FFF	PPB expansion	4K	Default slave	PPB-External	-
	0xE0045000	0xE00FEFFF	PPB expansion	754K	Default slave	PPB-External	-
	0xE00FF000	0xE00FFFFF	ROM table	4K	Default slave	PPB-External	-
	0xE0100000	0xFFFFFFFF	AHB expansion	511K	TARG_EXP1	-	-

3.3 eFlash controller

This section defines all the memory mapped registers that are present in the eFlash controller. These registers are used for control, status, configuration, write data to the flash banks, and read-emulated fuse values.

3.3.1 eFlash interrupts

The eFlash Controller provides a HW interrupt signal (IRQ) that can be connected to the CPU. Interrupt status register can be used to identify the exact source of the interrupt.

The IRQ output is synchronous to HCLK. The type of interrupt is level interrupt. The active level is high.

The interrupt output will be high if at least one bit of IRQ_MASKED_STATUS register value is 1.

To clear an interrupt, software writes 1 to the corresponding bit of the IRQ_CLR_STATUS register.

eFlash Controller has the following interrupt sources:

READY Any write or erase operation finished normally.

TIMEOUT Row-write operation aborted by time out.

NEXT Row write operation ready to accept the next word.

There is a corresponding bit for each interrupt source in the IRQ_SET_STATUS, IRQ_CLR_STATUS, IRQ_SET_ENA, IRQ_CLR_ENA and IRQ_MASKED_STATUS registers.

3.3.2 APB memory map

Table 3-3 APB memory map for eFlash controller

Memory Name	Type	Width	Number of words	Striped	Address offset within eFlash controller
FLASH0_INFO	RO	32	512	false	0x1000
FLASH1_INFO	RO	32	512	false	0x1800

3.3.3 Register summary

Table 3-4 APB register map for eFlash controller

Register name	Type	Width	Reset value	Address offset	Description
IRQ_SET_ENA	RW	32	0x00000000	0x0000	*IRQ_SET_ENA register* on page 3-9
IRQ_CLR_ENA	RW	32	0x00000000	0x0004	*IRQ_CLR_ENA register* on page 3-9
IRQ_SET_STATUS	RW	32	0x00000000	0x0008	*IRQ_SET_STATUS register* on page 3-10
IRQ_CLR_STATUS	RW	32	0x00000000	0x000C	*IRQ_CLR_STATUS register* on page 3-10
IRQ_MASKED_STATUS	RO	32	0x00000000	0x0010	*IRQ_MASKED_STATUS register* on page 3-11
CTRL	RW	32	0x00000000	0x0014	*CTRL register* on page 3-12
STATUS	RO	32	0x00000000	0x0018	*STATUS register* on page 3-12
CONFIG0	RW	32	0x0008003F	0x001C	*CONFIG0 register* on page 3-12
CONFIG1	RW	32	0x00080000	0x0020	*CONFIG1 register* on page 3-13
CONFIG2	RW	32	0x00080000	0x0024	*CONFIG2 register* on page 3-14
WADDR	RW	32	0x00000000	0x0028	*WADDR register* on page 3-14
WDATA	RW	32	0x00000000	0x002C	*WDATA register* on page 3-15
EFUSE	RO	32	0x00000000	0x0030	*EFUSE register* on page 3-15
HWPARAMS0	RO	32	0x00002011	0x0034	*HWPARAMS0 register* on page 3-15
HWPARAMS1	RO	32	0x0000003F	0x0038	*HWPARAMS1 register* on page 3-16
HWPARAMS2	RO	32	0x00000000	0x003C	*HWPARAMS2 register* on page 3-16
HWPARAMS3	RO	32	0x00000000	0x0040	*HWPARAMS3 register* on page 3-16
PIDR4	RO	32	0x00000014	0x0FD0	*Product ID Register, PIDR4* on page 3-17
PIDR5	RO	32	0x00000000	0x0FD4	*Product ID Register, PIDR5* on page 3-17
PIDR6	RO	32	0x00000000	0x0FD8	*Product ID Register, PIDR6* on page 3-17
PIDR7	RO	32	0x00000000	0x0FDC	*Product ID Register, PIDR7* on page 3-17
PIDR0	RO	32	0x00000030	0x0FE0	*Product ID Register, PIDR0* on page 3-18
PIDR1	RO	32	0x000000B8	0x0FE4	*Product ID Register, PIDR1* on page 3-18
PIDR2	RO	32	0x0000000B	0x0FE8	*Product ID Register, PIDR2* on page 3-18

Table 3-4 APB register map for eFlash controller (continued)

Register name	Type	Width	Reset value	Address offset	Description
PIDR3	RO	32	`0x00000000`	`0x0FEC`	*Product ID Register, PIDR3* on page 3-19
CIDR0	RO	32	`0x0000000D`	`0x0FF0`	*Component ID Register, CIDR0* on page 3-19
CIDR1	RO	32	`0x000000F0`	`0x0FF4`	*Component ID Register, CIDR1* on page 3-19
CIDR2	RO	32	`0x00000005`	`0x0FF8`	*Component ID Register, CIDR2* on page 3-20
CIDR3	RO	32	`0x000000B1`	`0x0FFC`	*Component ID Register, CIDR3* on page 3-20

IRQ_SET_ENA register

Enables, or reads the enable state of interrupts.

RW register at offset `0x0000`.

For the non-reserved bits:

- On reads:
  - 0: interrupt disabled.
  - 1: interrupt enabled.

    If not masked, a HW interrupt generated if the corresponding bit of the IRQ_STATUS register is set.
- On writes:
  - 0: no effect.
  - 1: enable interrupt.

Table 3-5 IRQ_SET_ENA register

Bits	Name	Description	Access	Reset
`[31:3]`	Reserved	-	RO, RAZ	`0`
`[2]`	NEXT	-	RW	`0`
`[1]`	TIMEOUT	-	RW	`0`
`[0]`	READY	-	RW	`0`

IRQ_CLR_ENA register

Disables, or reads, the enable state of interrupts.

RW register at offset `0x0004`.

For the non-reserved bits:

- On reads:
  - 0: interrupt disabled.
  - 1: interrupt enabled.

    If not masked, a HW interrupt generated if the corresponding bit of the IRQ_STATUS register is set.

- On writes:
  - 0: no effect.
  - 1: disable interrupt.

Table 3-6 IRQ_SET_ENA register

Bits	Name	Description	Access	Reset
[31:3]	Reserved	-	RO, RAZ	0
[2]	NEXT	-	RW	0
[1]	TIMEOUT	-	RW	0
[0]	READY	-	RW	0

IRQ_SET_STATUS register

Shows the current raw status of interrupts or sets the status of interrupts.

RW register at offset `0x0008`.

For the non-reserved bits:

- On reads:
  - 0: interrupt not pending.
  - 1: interrupt pending.
- On writes:
  - 0: no effect.
  - 1: sets the state of the interrupt to pending.

Table 3-7 IRQ_SET_STATUS register

Bits	Name	Description	Access	Reset
[31:3]	Reserved	-	RO, RAZ	0
[2]	NEXT	This interrupt is set by HW during word-write operation whenever HW is ready to accept the next word.	RW	0
[1]	TIMEOUT	This interrupt is set by HW when any row-write operation finished by HW as a result of SW not clearing the NEXT interrupt within the specified time.	RW	0
[0]	READY	This interrupt set by HW when any word-write, row-write, page-erase, mass-erase operation finishes.	RW	0

IRQ_CLR_STATUS register

Shows the current raw status of interrupts or clears the status of interrupts.

RW register at offset `0x000C`.

For the non-reserved bits:

- On reads:
  - 0: interrupt not pending.

— 1: interrupt pending.

- On writes:
  — 0: no effect.
  — 1: clears the pending state of the interrupt.

Table 3-8 IRQ_CLR_STATUS register

Bits	Name	Description	Access	Reset
[31:3]	Reserved	-	RO, RAZ	0
[2]	NEXT	This interrupt is set by HW during word-write operation whenever HW is ready to accept the next word.	RW	0
[1]	TIMEOUT	This interrupt is set by HW when any row-write operation finished by HW as a result of SW not clearing the NEXT interrupt within the specified time. On reads: 0 = Interrupt is not pending. 1 = Interrupt is pending. On writes: 0 = No effect. 1 = Clears the pending state of the interrupt.	RW	0
[0]	READY	This interrupt is set by HW when any word-write, row-write, page-erase, or mass-erase operation finishes. On reads: 0 = Interrupt is not pending. 1 = Interrupt is pending. On writes: 0 = No effect. 1 = Clears the pending state of the interrupt.	RW	0

IRQ_MASKED_STATUS register

Shows for each interrupt if it is pending and the cause of the interrupt line being asserted.

RO register at offset `0x0010`.

- On reads:
  — 0: interrupt is not causing IRQ line assertion.
  — 1: interrupt is cause of IRQ line assertion. Interrupt is pending and enabled.

Table 3-9 IRQ_MASKED_STATUS register

Bits	Name	Description	Access	Reset
[31:3]	Reserved	-	RO, RAZ	0
[2]	NEXT	-	RO	0
[1]	TIMEOUT	-	RO	0
[0]	READY	-	RO	0

CTRL register

eFlash control register.

If **SAFESTATEREQn** or **SHUTDOWNREQn** is asserted, the eFlash controller will reject any write attempt to this register and respond with an **APB ERROR** response.

RW register at offset `0x0014`.

Table 3-10 CTRL register

Bits	Name	Description	Access	Reset
`[31:5]`	Reserved	-	RO, RAZ	`0`
`[4]`	STOP	Stop any write or erase operation. High voltage discharge is taken care of by eFlash Controller.	RW	`0`
`[3]`	MASS_ERRASE	Erase all pages of eFlash.	RW	`0`
`[2]`	ERASE	Erase one page of eFlash.	RW	`0`
`[1]`	ROW_WRITE	Write one or more words (32 bit) to a row of eFlash (to sequential addresses) during one high voltage period.	RW	`0`
`[0]`	WRITE	Write one word (32 bit) of data to eFlash.	RW	`0`

STATUS register

Status or read or erase operation.

RO register at offset `0x0018`.

Table 3-11 STATUS register

Bits	Name	Description	Access	Reset
`[31:2]`	Reserved	-	RO, RAZ	`0`
`[1]`	LOCK	Write/Erase lock. Lock conditions are **SAFESTATEREQn** asserted or **SHUTDOWNREQn** asserted. 0: Write and Erase operations can be executed by the eFlash controller. 1: The eFlash controller will reject any write or erase with an APB **ERROR** response.	RO	`1`
`[0]`	BUSY	eFlash Controller is executing any write or erase operation. Indicates that any eFlash bank is in HV state.	RO	`0`

CONFIG0 register

Configuration register.

RW register at offset `0x001C`.

Table 3-12 CONFIG0 register

Bits	Name	Description	Access	Reset
[31:26]	Reserved	-	RO, RAZ	0
[25:16]	ER_CLK_COUNT	Erase clock configuration register. Set the number of clock cells in 1ms period. (ER_CLK_COUNT+1) * Clock_period > 1ms Minimum value is the nearest integer that results in a period > 1ms. The clock source is always EXTCLK. The valid EXTCLK frequency range is 1kHz - 1MHz. This register is not implemented if EXTCLKEN parameter set to 0 (RAZ).	RW	ERCLKCOUNTRST
[15:8]	WR_CLK_COUNT	Write clock configuration register. Set the number of clock cells in 1us period. If EXT_CLK_CONF is `0x0` or `0x1` then(WR_CLK_COUNT * HCLK-period) >= 1us. If EXT_CLK_CONF is `0x2` then(WR_CLK_COUNT * EXTCLK-period) >= 1us.	RW	WRCLKCOUNTRST
[7:6]	ETC_CLK_CONF	Write/erase timers source clock configuration. If EXTCLKEN parameter is set to 0, then this register is tied to `0x0`. `0x0` [Internal] External clock not used. `0x1` [Erase] External clock used for erase counters (>1ms). HCLK used for write counters. `0x2` [Write] External clock used for write and erase counters (>1us). `0x3` [Reserved].	RW	0x0
[5:0]	RD_CLK_CONF	Read clock configuration register. `0x0` Reserved. `0x1` 1_cycle_read_mode. This value is allowed only if HALFCLKREAD parameter is set to 1. Read from flash in 1 clock cycle over AHB interface,Read from flash in 2 clock cycles over APB interface. `0x2-03F` normal_read_mode. eFlash read operation requires RD_CLK_COUNT number of HCLK cycles.	RW	RDCLKCOUNTRST

CONFIG1 register

Configuration register.

RW register at offset `0x0020`.

Table 3-13 CONFIG1 register

Bits	Name	Description	Access	Reset
`[31:26]`	TNVH	eFlash timing parameter. NVSTR hold time in microseconds.	RW	TNVH_RST
`[23:16]`	TPROG	eFlash timing parameter. Programming time in microseconds.	RW	TPROG_RST
`[15:8]`	TPGS	eFlash timing parameter. NVSTR to program setup time in microseconds.	RW	TPGS_RST
`[7:0]`	ETC_CLK_CONF	eFlash timing parameter. PROG or ERASE to NVSTR setup time in microseconds.	RW	TNVS_RST

CONFIG2 register

Configuration register.

RW register at offset `0x0024`.

Table 3-14 CONFIG2 register

Bits	Name	Description	Access	Reset
`[31:24]`	TME	eFlash timing parameter. Mass erase time in ms.	RW	TME_RST
`[23:16]`	TERASE	eFlash timing parameter. Erase time in ms.	RW	TERASE_RST
`[15:8]`	TRCV	eFlash timing parameter. Recovery time in microseconds.	RW	TRCV_RST
`[7:0]`	TNVH1	eFlash timing parameter. NVSTR1 hold time in microseconds.	RW	TNVH1_RST

WADDR register

Write/Erase address register.

Only word addressing is allowed. Bits [1:0] are tied to 0.

Attempting to write an unmapped address into this register results in an APB ERROR response and the register value is not modified.

It is responsibility of software to ensure that the addressed word of eFlash is in erased state.

The bits have special addressing for information pages:

- `[31]`: Select Bank-1 information page.
- `[30]`: Select Bank-0 information page.
- `[10:2]`: Info page word address offset.

RW register at offset `0x0028`.

Table 3-15 WADDR register

Bits	Name	Description	Access	Reset
[31:2]	WADDR1	-	RW	0
[1:0]	Reserved	-	RO	0

WDATA register

Write data register.

RW register at offset `0x002C`.

Table 3-16 WDATA register

Bits	Name	Description	Access	Reset
[31:0]	WDATA	-	RW	0

EFUSE register

Each bit of this register corresponds to an emulated fuse value.

RO register at offset `0x0030`.

Table 3-17 EFUSE register

Bits	Name	Description	Access	Reset
[31:0]	EFUSE	EFUSE[0] is efuse 0.	RO	0

HWPARAMS0 register

Timeout and clock control register. The value of this register is defined by the system designer.

RO register at offset `0x0034`.

Table 3-18 HWPARAMS0 register

Bits	Name	Description	Access	Reset
[31:16]	Reserved	-	RO, RAZ	0
[15:8]	TIMEOUT	Row-write timeout parameter	RO	0x20
[7]	Reserved	-	RO, RAZ	0
[6]	EXTCLKEN	Enable EXTCLK input	RO	0
[5]	HALFCLKRD	Allow setting RD_CLK_COUNT to 0	RO	0
[4:0]	FLASHSIZE	FLASHSIZE parameter = log2(flash size in bytes)	RO	0x11

HWPARAMS1 register

Clock count parameters. The value of this register is defined by the system designer.

RO register at offset 0x0038.

Table 3-19 HWPARAMS1 register

Bits	Name	Description	Access	Reset
[31:26	Reserved	-	RO, RAZ	0
[25:12]	ERCLKCOUNTRST	Reset value of ER_CLK_COUNT register.	RO	0
[11:4]	WRCLKCOUNTRST	Reset value of WR_CLK_COUNT register,	RO	0
[3:0]	RDCLKCOUNTRST	Reset value of RD_CLK_COUNT register.	RO	0x3F

HWPARAMS2 register

eFlash timing parameters. The value of this register is defined by the system designer.

RO register at offset 0x003C.

Table 3-20 HWPARAMS2 register

Bits	Name	Description	Access	Reset
[31:24]	TNVH_RST	eFlash timing parameter. NVSTR hold time in microseconds.	RO	0
[23:16]	TPROG_RST	eFlash timing parameter. Programming time in microseconds.	RO	0
[15:8]	TPGS_RST	eFlash timing parameter. NVSTR to program setup time in microseconds.	RO	0
[7:0]	TNVS_RST	eFlash timing parameter. PROG/ERASE to NVSTR setup time in microseconds.	RO	0

HWPARAMS3 register

eFlash timing parameters for erase and hold. The value of this register is defined by the system designer.

RO register at offset 0x0040.

Table 3-21 HWPARAMS3 register

Bits	Name	Description	Access	Reset
[31:24]	TME_RST	eFlash timing parameter. Mass erase time in ms.	RO	0
[23:16]	TERASE_RST	eFlash timing parameter. Erase time in ms.	RO	0
[15:8]	TRCV_RST	eFlash timing parameter. Recovery time in microseconds.	RO	0
[7:0]	TNVH1_RST	eFlash timing parameter. NVSTR1 hold time in microseconds.	RO	0

Product ID Register, PIDR4

eFlash parameters for address space.

RO register at offset `0x0FD0`.

Table 3-22 PIDR4 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:4]	SIZE	8k address space	RO	0x1
[3:0]	DES_2	JEP 106 continuation code	RO	0x4

Product ID Register, PIDR5

Reserved.

RO register at offset `0x0FD4`.

Table 3-23 PIDR5 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	Reserved	-	RO, RAZ	0

Product ID Register, PIDR6

Reserved.

RO register at offset `0x0FD8`.

Table 3-24 PIDR6 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	Reserved	-	RO, RAZ	0

Product ID Register, PIDR7

Reserved.

RO register at offset `0x0FDC`.

Table 3-25 PIDR7 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	`0`
`[7:0]`	Reserved	-	RO, RAZ	`0`

Product ID Register, PIDR0

eFlash part number.

RO register at offset `0x0FE0`.

Table 3-26 PIDR0 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	`0`
`[7:0]`	PART_0	Bits [7:0] of the part number.	RO, RAZ	`0x30`

Product ID Register, PIDR1

eFlash part number.

RO register at offset `0x0FE4`.

Table 3-27 PIDR1 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	`0`
`[7:4]`	DES_0	Bits [11:8] of the part number.	RO, RAZ	`0xB`
`[3:0]`	PART_0	Bits [11:8] of the part number.	RO, RAZ	`0x8`

Product ID Register, PIDR2

eFlash revision number.

RO register at offset 0x0FE8.

Table 3-28 PIDR2 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:4]	REVISION	Revision number of the peripheral.	RO, RAZ	0
[3]	JEDEC	Always set. Indicates that a JEDEC assigned value is used.	RO, RAZ	0x1
[2:0]	DES_1	JEP106 identification code, bits[6:4]	RO, RAZ	0x3

Product ID Register, PIDR3

eFlash customer-modified number.

RO register at offset 0x0FEC.

Table 3-29 PIDR3 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:4]	REVAND	ECO revision	RO, RAZ	0
[3:0]	CMOD	Customer modified number	RO, RAZ	0

Component ID Register, CIDR0

eFlash parameter register.

RO register at offset 0x0FF0.

Table 3-30 CIDR0 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	PRMBL_0	-	RO	0x0D

Component ID Register, CIDR1

eFlash parameter register for IP component.

RO register at offset `0x0FF4`.

Table 3-31 CIDR1 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	`0`
`[7:4]`	CLASS	Component class	RO	`0xF`
`[3:0]`	PRMBL_1	-	RO	`0x0`

Component ID Register, CIDR2

eFlash parameter register.

RO register at offset `0x0FF8`.

Table 3-32 CIDR2 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	0
`[7:0]`	PRMBL_2	-	RO	`0x05`

Component ID Register, CIDR3

eFlash parameter register.

RO register at offset `0x0FFC`.

Table 3-33 CIDR3 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	0
`[7:0]`	PRMBL_3	-	RO	`0xB1`

3.4 eFlash cache

The eFlash cache is an instruction cache instantiated between the interconnect and the eFlash controller. The cache interfaces with the interconnect (master) over a 32-bit AHB-Lite bus and interfaces with the eFlash over a 128-bit AHB-Lite bus. The cache maintenance operations are performed through an APB4 interface

3.4.1 Register summary

The table below summarizes the registers in the eFlash cache.

Table 3-34 eFlash cache registers

Register name	Type	Width	Reset Value	Address Offset	Description
CCR	RW	32	0x00000040	0x000	*Configuration and Control Register, CCR* on page 3-22
SR	RO	32	0x00000000	0x004	*Status Register, SR* on page 3-23
IRQMASK	RW	32	0x00000000	0x008	*Interrupt Request Mask register, IRQMASK* on page 3-23
IRQSTAT	RW	32	0x00000000	0x00C	*Interrupt Request Status register, IRQSTAT* on page 3-23
HWPARAMS	RO	32	0x00003992	0x010	*Hardware Parameters register, HWPARAMS* on page 3-24
CSHR	RW	32	0x00000000	0x014	*Cache Statistic Hit Register, CSHR* on page 3-25
CSMR	RW	32	0x00000000	0x018	*Cache Statistic Miss Register, CSMR* on page 3-25
PIDR4	RO	32	0x00000004	0xFD0	*Product ID Register, PIDR4* on page 3-25
PIDR5	RO	32	0x00000000	0xFD4	*Product ID Register, PIDR5* on page 3-26
PIDR6	RO	32	0x00000000	0xFD8	*Product ID Register, PIDR6* on page 3-26
PIDR7	RO	32	0x00000000	0xFDC	*Product ID Register, PIDR7* on page 3-26
PIDR0	RO	32	0x00000029	0xFE0	*Product ID Register, PIDR0* on page 3-26
PIDR1	RO	32	0x000000B8	0xFE4	*Product ID Register, PIDR1* on page 3-27
PIDR2	RO	32	0x0000000B	0xFE8	*Product ID Register, PIDR2* on page 3-27
PIDR3	RO	32	0x00000000	0xFEC	*Product ID Register, PIDR3* on page 3-27
CIDR0	RO	32	0x0000000D	0xFF0	*Component ID Register, CIDR0* on page 3-28

Table 3-34 eFlash cache registers (continued)

Register name	Type	Width	Reset Value	Address Offset	Description
CIDR1	RO	32	`0x000000F0`	`0xFF4`	*Component ID Register, CIDR1* on page 3-28
CIDR2	RO	32	`0x00000005`	`0xFF8`	*Component ID Register, CIDR2* on page 3-28
CIDR3	RO	32	`0x000000B1`	0xFFC	*Component ID Register, CIDR3* on page 3-29

Configuration and Control Register, CCR

Configuration and control register.

RW register at offset `0x000`.

Table 3-35 CCR register

Bits	Name	Description	Access	Reset
`[31:7]`	Reserved	-	RO, RAZ	0
`6`	STATISTIC_EN	Enable statistics logic: 0: Disabled. Counters are stalled. 1: Enable. Counters are running.	RW	1
`5`	SET_PREFETCH	Cache Prefetch Setting: 0: Disable cache. 1: Enable cache.	RW	0
`4`	SET_MAN_INV	Cache Invalidate Setting: 0: Automatic cache invalidate when cache enabled. 1: Manual cache invalidate mode.	RW	0
`3`	SET_MAN_POW	Power Control Setting: 0: Automatic. 1: Manual.	RW	0
`2`	POW_REQ	Manual SRAM power request.	RW	0
`1`	INV_REQ	Manual invalidate request. Functional only when SET_MAN_INV is set. Automatically cleared when invalidation is finished or power or invalidation error occurs. Cannot be cleared manually. Manual invalidation request should be set only when cache is disabled otherwise it causes invalidation error interrupt.	RW	0
`0`	EN	Cache Enable: 0: Disable cache. 1: Enable cache.	RW	0

Status Register, SR

Status register.

RO register at offset `0x004`.

Table 3-36 SR register

Bits	Name	Description	Access	Reset
[31:5]	Reserved	-	RO, RAZ	0
4	POW_STAT	SRAM power acknowledges. Real-time registered value of RAMPWRUPACK port.	RO	0
3	Reserved	-	RO, RAZ	0
2	INV_STAT	Invalidating Status. Indicates if invalidation process is ongoing.	RO	0
[1:0]	CS	Cache status: 0: Cache disabled. 1: Cache enabling. 2: Cache enabled. 3: Cache disabling.	RO	0

Interrupt Request Mask register, IRQMASK

Interrupt request mask register. Set to 0 to enable interrupts for events, and set to 1 to mask interrupts.

RW register at offset `0x008`.

Table 3-37 IRQMASK register

Bits	Name	Description	Access	Reset
[31:2]	Reserved	-	RO, RAZ	0
1	MAN_INV_ERR	Mask interrupt request on manual invalidation error indication (IRQSTAT.MAN_INV_ERR is set).	RW	0
0	POW_ERR	Mask interrupt request on Power Error indication (IRQSTAT.POW_ERR is set).	RW	0

Interrupt Request Status register, IRQSTAT

Interrupt Request Status Register. IRQSTAT register status bits cannot be masked. They are set on the corresponding error event regardless of IRQMASK settings.

RW register at offset `0x00C`.

Table 3-38 IRQSTAT register

Bits	Name	Description	Access	Reset
[31:2]	Reserved	-	RO, RAZ	0
1	MAN_INV_ERR	Manual invalidation error status. Set when manual invalidation is requested meanwhile the cache is not disabled. Write 1 to clear.	RW	0
0	POW_ERR	SRAM power error. Write 1 to clear. Power acknowledge de-asserted during operation. Manual power request de-asserted while cache is enabled and operating in manual power mode.	RW	0

Hardware Parameters register, HWPARAMS

Hardware parameters register holding implementation-defined parameter values.

RO register at offset `0x010`.

Table 3-39 HWPARAMS register

Bits	Name	Description	Access	Reset
[31:14]	Reserved	-	RO, RAZ	0
13	GEN_STAT_LOGIC	Indicates GEN_STAT_LOGIC hardware parameter value.	RO	1
12	RESET_ALL_REGS	Indicates RESET_ALL_REGS hardware parameter value.	RO	1
[11:10]	CACHE_WAY	Implementation-defined value for number of cache ways: • 2: Two cache ways.	RO	2
[9:5]	CW	Implementation-defined value for cache way width: • 8: 256B. • 9: 512B. • 11: 2KB. • 12: 4KB.	RO	0x0C
[4:0]	AW	Implementation-defined value for AHB-Lite bus width for the flash address space: • 18: 256KB. • 19: 512KB.	RO	0x12

Cache Statistic Hit Register, CSHR

Cache Statistic Hit Register.

Including this register in a design is optional. If not present and the register is accessed, a slave **OKAY** response is given.

RW register at offset `0x014`.

Table 3-40 CSHR register

Bits	Name	Description	Access	Reset
`[31:0]`	CSHR	Counts the number of cache hits during cache look up. Only cacheable read transactions are looked up by the eFlash cache. Writing to the register clears the contents.	RW	0

Cache Statistic Miss Register, CSMR

Cache Statistic Miss Register.

Including this register in a design is optional. If not present and the register is accessed, a slave **OKAY** response is given.

RW register at offset `0x018`.

Table 3-41 CMSR register

Bits	Name	Description	Access	Reset
`[31:0]`	CSMR	Counts the number of cache misses during cache look up. Only cacheable read transactions are looked up by the eFlash cache. Writing to the register clears the contents.	RW	0

Product ID Register, PIDR4

Product ID register.

RO register at offset `0xFD0`.

Table 3-42 PIDR4 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	0
`[7:4]`	SIZE	-	RO	`0x0`
`[3:0]`	DES_2	JEP 106 continuation code	RO	`0x4`

Product ID Register, PIDR5

Product ID register.

RO register at offset `0xFD4`.

Table 3-43 PIDR5 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	Reserved	-	RO, RAZ	0

Product ID Register, PIDR6

Product ID register.

RO register at offset `0xFD8`.

Table 3-44 PIDR6 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	Reserved	-	RO, RAZ	0

Product ID Register, PIDR7

Product ID register.

RO register at offset `0xFDC`.

Table 3-45 PIDR7 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	Reserved	-	RO, RAZ	0

Product ID Register, PIDR0

Product ID register.

RO register at offset `0xFE0`.

Table 3-46 PIDR0 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	PART_0	Part number bits [7:0].	RO	0x29

Product ID Register, PIDR1

Product ID register.

RO register at offset `0xFE4`.

Table 3-47 PIDR1 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:4]	DES_0	PJEP106 identification code bits [3:0].	RO	0xB
[3:0]	PART_1	Part number bits [11:8].	RO	0x8

Product ID Register, PIDR2

Product ID register.

RO register at offset `0xFE8`.

Table 3-48 PIDR2 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:4]	REVISION	Revision number of the peripheral	RO	0x0
3	JEDEC	Always set. Indicates that a JEDEC assigned value is used.	RO	1
[2:0]	DES_1	JEP106 identification code bits [11:8].	RO	0x3

Product ID Register, PIDR3

Product ID register.

RO register at offset `0xFEC`.

Table 3-49 PIDR3 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	0
`[7:4]`	REVAND	ECO revision.	RO	`0x0`
`[2:0]`	CMOD	Customer modified number.	RO	`0x0`

Component ID Register, CIDR0

Component ID register.

RO register at offset `0xFF0`.

Table 3-50 CIDR0 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	0
`[7:0]`	PRMBL_0	-	RO	`0x0D`

Component ID Register, CIDR1

Component ID register.

RO register at offset `0xFF4`.

Table 3-51 CIDR1 register

Bits	Name	Description	Access	Reset
`[31:8]`	Reserved	-	RO, RAZ	0
`[7:4]`	CLASS	Component Class. Returns `0xE` for a generic IP component.	RO	`0x0F`
`[3:0]`	PRMBL_1	-	RO	`0x0`

Component ID Register, CIDR2

Component ID register.

RO register at offset `0xFF8`.

Table 3-52 CIDR2 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	PRMBL_2	-	RO	0x05

Component ID Register, CIDR3

Component ID register.

RO register at offset `0xFFC`.

Table 3-53 CIDR3 register

Bits	Name	Description	Access	Reset
[31:8]	Reserved	-	RO, RAZ	0
[7:0]	PRMBL_3	-	RO	0xB1

3.5 Interrupts

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

- An implementation-defined number of interrupts, in the range 4-240 interrupts.
- 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 re-prioritization of interrupts.
- Grouping of priority values into group priority and sub-priority fields.
- Interrupt tail-chaining.
- An external *Non-Maskable Interrupt* (NMI)
- Optional WIC, providing ultra-low power sleep mode support.

3.5.1 Interrupt signals

This section describes interrupts and exceptions handled by the Cortex-M3 in the IoT Subsystem.

Table 3-54 Exceptions

No.	Exception type	Priority	Description
1	Reset	-3	Reset is invoked on power up or a warm reset
2	NMI	-2	CPU0INTNMI port input
3	Hard-Fault	-1	A Hard Fault is an exception that occurs because of an error during exception processing
4	Memory manage fault	Programmable	Exception from MPU
5	Bus fault	Programmable	Bus error of bus access to the area that is not controlled by the MPU
6	Use fault	Programmable	Error about operating instruction including undefined instruction
7-10	Reserved	-	-
11	SVCall	Programmable	Call
12	Debug Monitor	Programmable	Exception that is triggered by the SVC instruction
13	Reserved	-	
14	PendSV	Programmable	PendSV is an interrupt-driven request for system-level service
15	SysTick	Programmable	SysTick exception is an exception the system timer generates when it reaches zero.
16	Interrupt Specific	programmable	An exception signaled by a peripheral, or generated by a software request.

The reference interrupt table are listed in the table below:

Table 3-55 Interrupts

No.	NAME	Source	Description
16	CPU0INTISR0	**FCACHEIRQ**	Interrupt from eFlash cache (Instruction Cache) module.
17	CPU0INTISR1	**FLSIRQ**	Interrupt from the eFlash Controller module.
18	CPU0INTISR2	**TIMER0TIMERINT**	Interrupt from APB Timer 0 module.
19	CPU0INTISR3	**TIMER1TIMERINT**	Interrupt from APB Timer 0 module.
20	CPU0INTISR4	**BTTXIRQINT**	Combined TX interrupt of BT radio.
21	CPU0INTISR5	**BTRXIRQINT**	Combined RX interrupt of BT radio.
22	CPU0INTISR6	**UART0INT**	Combined interrupt from APB_UART0 module.
23	CPU0INTISR7	**UART1INT**	Combined interrupt from APB_UART1 module.
-	CPU0INTISR8-31	Not used	-

——— **Note** ———

The interrupt signals from the peripherals are not connected directly to the interrupt controller. All signals go to the integration layer and might be connected differently by the system designer.

3.5.2 Registers

A summary of the interrupt controller registers are listed in the table below:

Table 3-56 Summary of interrupt controller registers

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

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

- *Cortex-M3 Technical Reference Manual*.
- *ARMv7M Architecture Reference Manual*.

3.6 Wakeup Interrupt Controller (WIC)

The *Wakeup Interrupt Controller* (WIC) is a peripheral that can detect an interrupt and wake the processor from deep sleep mode. The WIC is enabled only when the system is in deep sleep mode.

The WIC is not programmable, and does not have any registers or user interface. It operates entirely from hardware signals.

When the WIC is enabled and the processor enters deep sleep mode, the power management unit in the system can power down most of the Cortex-M3 processor. When the WIC receives an interrupt, it takes a number of clock cycles to wakeup the processor and restore its state to enable it to process the interrupt. This means interrupt latency is increased in deep sleep mode.

Note

Unlike in the standard Cortex-M3, the IoT System WIC is implemented by latches. This means that FCLK can be gated completely during WIC based deep sleep. This is not a standard CM3 feature.

For more information on the WIC, see the *Cortex-M3 Technical Reference Manual*.

3.7 Timer

A summary of the registers in the timer is provided in Table 3-57.

Table 3-57 Summary of timer registers

Name	Type	Width	Reset value	Address	Description
CTRL	RW	4	`0x0`	Base +`0x000`	3: Interrupt enable. 2: Select external input as clock. 1: Select external input as enable. 0: Enable.
VALUE	RW	32	`0x00000000`	Base +`0x004`	Current value
RELOAD	RW	32	`0x00000000`	Base +`0x008`	Reload value. A write to this register sets the current value.
INTSTATUS INTCLEAR	RW	1	`0x0`	Base +`0x00C`	Timer interrupt Write 1 to clear
PID4	RO	8	`0x04`	Base +`0xFD0`	Peripheral ID register 4
PID5	RO	8	`0x00`	Base +`0xFD4`	Peripheral ID register 5
PID6	RO	8	`0x00`	Base +`0xFD8`	Peripheral ID register 6
PID7	RO	8	`0x00`	Base +`0xFDC`	Peripheral ID register 7
PID0	RO	8	`0x22`	Base +`0xFE0`	Peripheral ID register 0 [7:0] Part number[7:0].
PID1	RO	8	`0xB8`	Base +`0xFE4`	Peripheral ID register 1 [7:4] jep106_id_3_0. [3:0] Part number[11:8].
PID2	RO	8	`0x0B`	Base +`0xFE8`	Peripheral ID register 2 [7:4] Revision. [3] jedec_used. [2:0] jep106_id_6_4.
PID3	RO	8	`0x00`	Base +`0xFEC`	Peripheral ID register 3 [7:4] ECO revision number. [3:0] Customer modification number.
CID0	RO	8	`0x0D`	Base +`0xFF0`	Component ID register 0
CID1	RO	8	`0xF0`	Base +`0xFF4`	Component ID register 1
CID2	RO	8	`0x05`	Base +`0xFF8`	Component ID register 2
CID3	RO	8	`0xB1`	Base +`0xFFC`	Component ID register 3

3.8 System registers

For information on the Cortex-M3 registers, see the following documents:

- *Cortex-M System Design Kit Technical Reference Manual*.
- *Cortex-M3 Technical Reference Manual*.
- *ARMv7M Architecture Reference Manual*.

The following table lists the IoT Subsystem, flash cache, and flash controller ID registers:

Table 3-58 Part number ID values

Class	Part Number	Part	Rev	ID0	ID1	ID2	ID3	ID4
`0xF`	`0x829`	eFlash cache IP	r0p0	`0x29`	`0xB8`	`0x0B`	`0x00`	`0x04`
`0xF`	`0x830`	eFlash controller IP	r0p0	`0x30`	`0xB8`	`0x0B`	`0x00`	`0x14`
`0x1`	`0x741`	IoT Subsystem	r0p0	`0x41`	`0xB7`	`0x0B`	`0x00`	`0x04`

3.9 Debug and Trace

The IoT Subsystem has two configuration options for debug and trace. The system implementer can select either:

- CortexM3 debug and trace capabilities with no requirement for CoreSight SoC
- CoreSight SoC compatible configuration which requires CoreSight SoC license.In this case you must develop your own CoreSight system.

For more information on debug and trace, see the *Cortex-M3 Processor Technical Reference Manual*.

Note

If the system is configured for CoreSight SoC, a license is required for that IP and the related ARM documentation:

- *CoreSight SoC-400 System Design Guide*.
- *CoreSight SoC-400 Technical Reference Manual*.
- *CoreSight SoC-400 User Guide*.
- *CoreSight TPIU-Lite Technical Reference Manual*.
- *CoreSight DAP-Lite Technical Reference Manual*.

Appendix A
Signal Descriptions

This chapter summarizes the interface signals present in the *IoT Subsystem for Cortex-M* (IoT Subsystem).

It contains the following sections:

A.1 Clock and reset signals

The table below lists clock and reset signals in the IoT Subsystem interface.

Table A-1 Clocks and resets

Name	Direction	Width	Description
AHB2APBHCLK	Input	1	AHB to APB bridge Clock
CPU0CTICLK	Input	1	Clock for the CoreSight CTI. When active, this must be the same clock as CPU0FCLK.
CPU0CTICLKEN	Input	1	An enable signal for CPU0CTICLK.
CPU0CTIRESETn	Input	1	Resets the CTI trigger interface and CTI wrapper. De-assert the reset synchronously to CPU0CTICLK.
CPU0DAPCLK	Input	1	A clock signal for the debug bus interface from the Debug Port (DP) component, for example SWJ-DP. This can be asynchronous to other clock signals
CPU0DAPCLKEN	Input	1	An enable signal for CPU0DAPCLK
CPU0DAPRESETn	Input	1	Resets the debug bus connected to the AHB-AP inside the Cortex-M3 processor,
CPU0FCLK	Input	1	A free-running clock. This must be active when the processor is running, for debugging, and for the Nested Vectored Interrupt Controller (NVIC) to detect interrupts. The IoT Subsystem implements the WIC with LATCH thus during WIC based sleep it can be gated
CPU0HCLK	Input	1	A system clock. This must be the same as CPU0FCLK when active. You can gate this off when the processor is in sleep
CPU0PORESETn	Input	1	Power-on reset. Resets the entire Cortex-M3 system and debug components, but excluding the CTI trigger interface and CTI wrapper.
CPU0STCALIB	Input	26	Calibration signal for alternative clock source of SysTick timer
CPU0STCLK	Input	1	Clock enable of the alternative clock source of SysTick timer. CPU0STCLK CPU0FCLK is gated with this CPU0STCLK
CPU0SYSRESETn	Input	1	System Reset. Resets the processor and the WIC excluding debug logic in the NVIC, FPB, DWT, and ITM.
DAPNTRST	Input	1	Debug nTRST reset initializes the state of the SWJ-DP TAP controller. This reset is typically used by the RealView® ICE module for hot-plug connection of a debugger to a system. It initializes the SWJ-DP controller without affecting the normal operation of the processor.
DAPSWCLKTCK	Input	1	Serial wire clock and JTAG Test clock. Can be asynchronous.
DAPNPOTRST	Input	1	Debug port power on reset. Deassertion must be synchronized to SWCLKTCK.
FCACHEHCLK	Input	1	AHB Bus clock. This clock is used for all always on logic

Table A-1 Clocks and resets (continued)

Name	Direction	Width	Description
FCACHEPCLKG	Input	1	Gated clock input for register interface (APB). It must be the same frequency and same phase as **FCACHEHCLK**. Can be gated, when there are no APB activities. It is expected to run while APB interface PSEL signal is **asserted**
FLSEXTCLK	Input	1	Low speed clock input (for example 32KHz) for program/erase operation. This input is used only if the FLS_EXTCLKEN parameter is set to high.
FLSHCLK	Input	1	AHB clock of eFlash controller.
FLSHRESETn	Input	1	De-assert the reset synchronously to FLSHCLK.
FLSPCLKG	Input	1	**FLSPCLKG** clock is used for APB interface and the register block that contains the SW writeable registers. **FLSPCLKG** can be dynamically turned off by APB subsystem controller. **FLSPCLKG** must be synchronous to **FLSHCLK** whenever the two clocks are on at the same time.
FLSPORESETn	Input	1	Power-On-Reset for the eFlash controller. Triggering the in-built self-repair operation of the eFlash Controller to repair eFlash pages and the eFuse values are read and FLSEFUSE is updated. De-assert the reset synchronously to FLSHCLK.
MTXHCLK	Input	1	AHB matrix interconnect clock
MTXHRESETn	Input	1	AHB matrix interconnect reset.
SRAM<0..3>HCLK	Input	1	AHB clock of the SRAM bridges.
TIMER0PCLK	Input	1	The free running clock used for timer operation. This must be the same frequency as, and synchronous to, the **TIMER0PCLKG** signal.
TIMER0PCLKG	Input	1	APB register read or write logic that permits the clock to peripheral register logic to stop when there is no APB activity.
TIMER0PRESETn	Input	1	De-assert the reset synchronously to **TIMER0PCLK**.
TIMER1PCLK	Input	1	The free running clock used for timer operation. This must be the same frequency as, and synchronous to, the **TIMER1PCLKG** signal.
TIMER1PCLKG	Input	1	APB register read or write logic that permits the clock to peripheral register logic to stop when there is no APB activity.
TIMER1PRESETn	Input	1	De-assert the reset synchronously to **TIMER0PCLK**.
TPIUCLK	Input	1	APB and ATB interface clock.
TPIUTRACECLKIN	Input	1	Trace out port source clock.

A.2 Interrupt signals

The table below lists clock and reset signals in the IoT Subsystem interface.

Table A-2 Interrupt signals

Name	Direction	Width	Description
CPU0INTISR[239:0]	Input	240	External interrupt signals. The number of functional interrupt signals depends on your implementation. The Cortex-M3 processor does not implement synchronizers for the **CPU0INTISR** input. To use asynchronous interrupts, you must implement external synchronizers to reduce the possibility of metastability issues.
CPU0INTNMI	Input	1	Non-mask able Interrupt. The number of functional interrupt signals depends on your implementation. The Cortex-M3 processor does not implement synchronizers for the **CPU0INTNMI** input. To use asynchronous interrupts, you must implement external synchronizers to reduce the possibility of metastability issues. If the input is connected to an IO pad, a noise filter must be applied.
CPU0CURRPRI[7:0]	Output	8	Indicates what priority interrupt, or base boost, is being used. CURRPRI represents the pre-emption priority, and does not indicate secondary priority.
CPU0AUXFAULT[31:0]	Input	32	Auxiliary fault status information from the system.
CPU0CTIINTISR	Output	2	CTI Interrupt request to top level CTI to system. Acknowledged by writing to the CTIINTACK register in ISR.

A.3 eFlash signals

This section lists signals for the Flash controller and the Flash cache.

——— **Note** ———

Some flash signal widths are implementation defined.

———————

A.3.1 Flash cache

The table below lists the interrupt signals for the Flash cache subsystem.

Table A-3 eFlash interrupts

Name	Direction	Width	Description
FCACHEIRQ	Output	1	eFlash cache interrupt output

The table below lists the SRAM signals for the eFlash subsystem.

Table A-4 eFlash cache DATA SRAM Interfaces

Name	Direction	Width	Description
FCACHERAMCLD<0..1>ADDR	Output	Implementation defined	Parametrized width data address bus.
FCACHERAMCLD<0..1>WE	Output	1	Write control for 128 bits (same cycle as address).
FCACHERAMCLD<0..1>RD	Output	4	Read control per word (same cycle as address).
FCACHERAMCLD<0..1>CS	Output	4	Chip select per word (same cycle as address).
FCACHERAMCLD<0..1>WDATA	Output	128	Write data (same cycle as address).
FCACHERAMCLD<0..1>RDATA	Input	128	Read data (1 cycle after address).

The table below lists the statistics signals for the eFlash subsystem.

Table A-5 eFlash cache statistics

Name	Direction	Width	Description
FCACHECACHEMISS	Output	1	Active high single cycle pulses indicating that a cache miss happened during cache look up.
FCACHECACHEHIT	Output	1	Active high single cycle pulses indicating that a cache hit happened during cache look up

The table below lists the TAG SRAM signals for the eFlash subsystem.

Table A-6 eFlash cache TAG SRAM Interfaces

Name	Direction	Width	Description
FCACHERAMTAG<0..1>ADDR	Output	Implementation defined	Parametrized width data address bus
FCACHERAMTAG<0..1>WE	Output	1	Write control (same cycle as address)
FCACHERAMTAG<0..1>RD	Output	1	Read control (same cycle as address)
FCACHERAMTAG<0..1>CS	Output	1	Chip select (same cycle as address)
FCACHERAMTAG<0..1>WDATA	Output	Implementation defined	Write data (same cycle as address)
FCACHERAMTAG<0..1>RDATA	Input	Implementation defined	Read data (1 cycle after address)

A.3.2 Flash controller

The table below lists the interrupt signals for the Flash controller subsystem.

Table A-7 eFlash interrupts

Name	Direction	Width	Description
FLSIRQ	Output	1	eFlash controller interrupt output

The table below lists the controller signals for the eFlash subsystem.

Table A-8 TSMC eFlash Controller

Name	Direction	Width	Description
FLSEFUSE	Output	32	eFuse output values
FLSEXTCLK	Input	1	Low speed clock input (for example 32KHz) for program/erase operation. This input is used only if FLS_EXTCLKEN parameter is set to high value.
FLSFACTORYRESETn	Input	1	Completely erase the eFlash contents and reset the eFlash emulated eFuses.
FLSREADY	Output	1	The Built-in Self Repair is done the redundancy pages are mapped. The fuse values are ready after reset removal.
FLSOBSERVATION	Output	32	Additional status signals for test
FLSSAFESTATEACKn	Output	1	Asserted as a response to **FLSSAFESTATEREQn** when none of the flash macros are executing any operation which requires high voltage state.
FLSSAFESTATEREQn	Input	1	When asserted it prevents starting any high voltage operation of the flash banks by masking the CTRL register's WRITE, ROW_WRITE, ERASE, MASS_ERASE bits.

The table below lists the eFlash0 interface signals that are specific to the TSMC 55ULP-TV2 process.

Table A-9 eFlash0 interface

Name	Direction	Width	Description
FLSXADR0	Output	Implementation defined	X address input, access rows, XADR[2:0] select one row within a page of main memory block or information block.
FLSYADR0	Output	5	Y address input, access data within a row.
FLSDOUT0	Input	128	Data output bus.
FLSDIN0	Output	128	Data input bus.
FLSXE0	Output	1	X address enable, all rows are disabled when XE=0.
FLSYE0	Output	1	Y address enable, YMUX is disabled when YE=0.
FLSSE0	Output	1	Sense amplifier enable.
FLSIFREN0	Output	1	Information block enable.
FLSERASE0	Output	1	Defines erase cycle.
FLSMAS10	Output	1	Defines mass erase cycle, erase whole block.
FLSPROG0	Output	1	Defines program cycle.
FLSNVSTR0	Output	1	Defines non-volatile store cycle.
FLSIFREN10	Output	1	Repaired page/status information read-only access enable.
FLSREDEN0	Output	1	Redundancy page enable for read, program and erase.

The table below lists the eFlash1 interface signals that are specific to the TSMC 55ULP-TV2 process.

Table A-10 eFlash1 interface

Name	Direction	Width	Description
FLSXADR1	Output	Implementation defined	X address input, access rows, XADR[2:0] select one row within a page of main memory block or information block.
FLSYADR1	Output	5	Y address input, access data within a row.
FLSDOUT1	Input	128	Data output bus.
FLSDIN1	Output	128	Data input bus.
FLSXE1	Output	1	X address enable, all rows are disabled when XE=0.
FLSYE1	Output	1	Y address enable, YMUX is disabled when YE=0.
FLSSE1	Output	1	Sense amplifier enable.
FLSIFREN1	Output	1	Information block enable.

Table A-10 eFlash1 interface (continued)

Name	Direction	Width	Description
FLSERASE1	Output	1	Defines erase cycle.
FLSMAS11	Output	1	Defines mass erase cycle, erase whole block.
FLSPROG1	Output	1	Defines program cycle.
FLSNVSTR1	Output	1	Defines non-volatile store cycle.
FLSIFREN11	Output	1	Repaired page/status information read-only access enable.
FLSREDEN1	Output	1	Redundancy page enable for read, program and erase.

A.4 SRAM signals

The table below lists the interface signals for the AHB2SRAM subsystem.

Table A-11 AHB2SRAM Interfaces

Name	Direction	Width	Description
SRAM<0..3>RDATA	Input	32	SRAM Read data bus.
SRAM<0..3>ADDR	Output	13	SRAM address (word address).
SRAM<0..3>WREN	Output	4	SRAM Byte write enable. Active HIGH.
SRAM<0..3>WDATA	Output	32	SRAM Write data.
SRAM<0..3>CS	Output	1	SRAM Chip select. Active HIGH.

A.5 Timer signals

The tables below lists the interface signals for the timer subsystem.

Table A-12 Timer

Name	Direction	Width	Description
TIMER0EXTIN	Input	1	Timer0 external input. The external clock. This must be slower than half of the TMER0PCLK clock because it is sampled by a double flip-flop and then goes through edge-detection logic when the external inputs act as a clock.
TIMER1EXTIN	Input	1	Timer0 external input. The external clock, must be slower than half of the TMER1PCLK clock because it is sampled by a double flip-flop and then goes through edge-detection logic when the external inputs act as a clock.
TIMER0PRIVMODEN	Input	1	Defines if the timer memory mapped registers are writeable only by privileged access.
TIMER1PRIVMODEN	Input	1	Defines if the timer memory mapped registers are writeable only by privileged access.
TIMER0TIMERINT	Output	1	Timer0 interrupt output,
TIMER1TIMERINT	Output	1	Timer1 interrupt output

A.6 Bus signals

The table below lists the signals for the two AHB master interfaces. Not shown in the list is the **TARGEXP<0..1>** prefix before each signal name.

Table A-13 External AHB target port signals

Name	Direction	Width	Description
HSEL	Output	1	Slave Select.
HADDR	Output	32	Address bus.
HTRANS	Output	2	Transfer Type.
HWRITE	Output	1	Transfer Direction.
HSIZE	Output	3	Transfer Size.
HBURST	Output	3	Burst type.
HPROT	Output	4	Protection Control.
HMASTER	Output	4	Master Select.
HWDATA	Output	32	Write Data.
HMASTLOCK	Output	1	Locked Sequence.
HREADYMUX	Output	1	Transfer done.
HAUSER	Output	1	Address USER signals (Not used by the subsystem Cortex-M3 CPU).
EXREQ	Output	1	Exclusive request.
MEMATTR	Output	2	Memory attributes.
HWUSER	Output	4	Write-data USER signals (Not used by the subsystem Cortex-M3 CPU).
HRDATA	Input	32	Read data bus.
HREADYOUT	Input	1	**HREADY** feedback.
HRESP	Input	1	Transfer response.
HRUSER	Input	3	Read-data USER signals (Not used by the subsystem Cortex-M3 CPU).
EXRESP	Input	1	Exclusive response.

The table below lists the signals for the two AHB slave interfaces. Not shown in the list is the **INITEXP<0..1>** prefix before each signal name.

Table A-14 External AHB initiator port signals

Name	Direction	Width	Description
HSEL	Input	1	Slave Select.
HADDR	Input	32	Address bus.
HTRANS	Input	2	Transfer Type.
HWRITE	Input	1	Transfer Direction.

Table A-14 External AHB initiator port signals (continued)

Name	Direction	Width	Description
HSIZE	Input	3	Transfer Size.
HBURST	Input	3	Burst type.
HPROT	Input	4	Protection Control.
HMASTER	Input	4	Master Select.
HWDATA	Input	32	Write Data.
HMASTLOCK	Input	1	Locked Sequence.
HAUSER	Input	1	Address USER signals (Not used by the subsystem Cortex M3 CPU).
EXREQ	Input	1	Exclusive Request signal.
MEMATTR	Input	2	Memory Attribute signals.
HWUSER	Input	4	Write-data USER signals (Not used by the subsystem Cortex-M3 CPU).
HRDATA	O	32	Read data bus.
HREADY	Output	1	HREADY feedback.
HRESP	Output	1	Transfer response.
HRUSER	Output	3	Read-data USER signals (Not used by the subsystem Cortex-M3 CPU).
EXRESP	Output	1	Exclusive Response.

The table below lists the signals for the two APB slave interfaces. Not shown in the list is the **APBTARGETEXP<n>** prefix before each signal name, where **n** is 2, 4, 5, 6, 7, 8, 11, 12, 13, 14, or 15.

Table A-15 External APB target port signals

Name	Direction	Width	Description
PSEL	Output	1	Slave select signal.
PENABLE	Output	1	Strobe to time all accesses. Used to indicate the second cycle of an APB transfer.
PADDR	Output	12 [11:0]	Address bus.
PWRITE	Output	1	APB transfer direction.
PWDATA	Output	32	32-bit write data bus.
PRDATA	Input	32	32-bit read data bus.
PREADY	Input	1	Driven LOW if extra wait states are required to complete the transfer.

Table A-15 External APB target port signals (continued)

Name	Direction	Width	Description
PSLVERR	Input	1	Indicates SLVERR response.
PSTRB	Output	1	Write strobes. This signal indicates which byte lanes to update during a write transfer. There is one write strobe for each eight bits of the write data bus. **PSTRB[n]** corresponds to **PWDATA[(8n + 7):(8n)]**. Write strobes must not be active during a read transfer
PPROT	Output	3	Protection type. This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access.

A.7 Debug and Trace signals

This section lists signals related to debug and trace. It has the following sections:

- *DAP signals*.
- *JTAG and SWD signals* on page A-15.
- *CPU debug signals* on page A-15.
- *Secure debug control* on page A-16.
- *CPU PPB expansion signals* on page A-16.
- *Trace signals* on page A-17.

A.7.1 DAP signals

The table below lists the interface signals for the DAP subsystem.

Table A-16 DAP

Name	Direction	Width	Description
CPU0DAPADDR	Input	8	DAP address bus
CPU0DAPSEL	Input	1	Select signal generated from the DAP decoder to each AP. This signal indicates that the slave device is selected, and a data transfer is required. There is a **DAPSEL** signal for each slave. The decoder monitors the address bus and asserts the relevant DAPSEL
CPU0DAPENABLE	Input	1	This signal is used to indicate the second and subsequent cycles of a DAP transfer from DP to AHB-AP
CPU0DAPWRITE	Input	1	When HIGH indicates a DAP write access from DP to AHB-AP. When LOW indicates a read access
CPU0DAPABORT	Input	1	Aborts the current transfer. The AHB-AP returns **DAPREADY** HIGH without affecting the state of the transfer in progress in the AHB master port.
CPU0DAPWDATA	Input	32	The write bus is driven by the DP block during write cycles, when **DAPWRITE** is HIGH.
CPU0DAPRDATA	Output	32	The read bus is driven by the selected AHB-AP during read cycles, when **DAPWRITE** is LOW.
CPU0DAPREADY	Output	1	The AHB-AP uses this signal to extend a DAP transfer
CPU0DAPSLVERR	Output	1	The error response.

A.7.2 JTAG and SWD signals

The table below lists the interface signals for the JTAG and SWD subsystem.

Table A-17 JTAG and SW Debug access functional signals

Name	Direction	Width	Description
DAPSWDITMS	Input	1	Debug TMS.
DAPSWDO	Output	1	Serial Wire Data Out.
DAPSWDOEN	Output	1	Serial Wire Output Enable.
DAPTDI	Input	1	Debug TDI.
DAPTDO	Output	1	Debug TDO.
DAPNTDOEN	Output	1	TDO output pad control signal.
DAPJTAGNSW	Output	1	JTAG or Serial-Wire selection JTAG mode(1) or SW mode(0).
DAPJTAGTOP	Output	1	JTAG-DP TAP controller in one of four top states TLR, RTI, Sel-DR, Sel-IR.

A.7.3 CPU debug signals

The table below lists the interface signals related to CPU debug.

Table A-18 CPU debug signals

Name	Direction	Width	Description
CPU0EDBGRQ	Input	1	External debug request. Combined debug request from ETM trace unit and multiprocessor debug support to connect to CoreSight Embedded Cross Trigger. This signal must be synchronous to **CPU0FCLK**.
CPU0ETMDBGREQ	Output	1	Debug request from ETM
CPU0DBGRESTART	Input	1	External restart request. The processor exits the halt state when the **CPU0DBGRESTART** signal is de-asserted during 4-phase handshaking.
CPU0DBGRESTARTED	Output	1	Handshake for **CPU0DBGRESTART**. Devices driving **CPU0DBGRESTART** must observe this signal to generate the required 4-phase handshaking.
CPU0CTICHIN	Input	4	Debug event channel inputs.
CPU0CTICHOUT,	Output	4	Debug event channel inputs.
CPU0CTIASICCTL	Output	8	ASIC auxiliary control from CTI.

A.7.4 Secure debug control

The table below lists the interface signals related to secure debug.

Table A-19 Secure debug control signals

Port name	Direction	Width	Description
CPU0DBGEN	Input	1	External debug enable. If **CPU0DBGEN** is de-asserted, the halt debugging feature of the processor is disabled and the invasive debug features on the CTI are also disabled. If **CPU0DBGEN** is asserted, you can use debug features, but you must set other enables, **C_DEBUGEN** for example, to enable debug events such as halt to occur. Either tie HIGH or connect to a debug access controller if required.
CPU0NIDEN	Input	1	Non Invasive debug enable. **NIDEN** must be HIGH to enable the ETM trace unit to trace instructions.
CPU0DAPEN	Input	1	Input AHB-AP enable. Enables the AHB-AP memory access functionality. In a typical arrangement, you can tie this signal HIGH. Connect HIGH when enabling debug accesses. If this signal is LOW, the debugger can still access registers inside the AHB-AP module inside the Cortex-M3 processor, but cannot access the memory map using the AHB interface.
CPU0FIXMASTERTYPE	Input	1	The AHB-AP can issue AHB transactions with a **HMASTER** value of either 1, to indicate DAP, or 0, to indicate processor data side, depending on how the AHB-AP is configured using the MasterType bit in the AHB-AP Control and Status Word Register. You can use **FIXMASTERTYPE** to prevent this if required. If it is tied to 0b1, then the **HMASTER** that the AHB-AP issues is always 1, to indicate DAP, and it cannot imitate the processor. If it is tied to 0b0, then **HMASTER** can be issued as either 0 or 1.

A.7.5 CPU PPB expansion signals

The table below lists the PPB interface signals related to the DAP subsystem.

Table A-20 CPI0 PPB expansion

Name	Direction	Width	Description
CPU0PREADY	Input	1	Bus slave ready.
CPU0PSLVERR	Input	1	Slave response.
CPU0PRDATA	Input	32	Read data.
CPU0PADDR	Output	31	Connect bits [19:2] to **PADDR[19:2]** from the Cortex-M3 processor.
CPU0PADDR31	Output	1	Indicates whether the transfer originates from the processor (0) or the debugger (1).
CPU0PWRITE	Output	1	If this is 1, it indicates that the transfer is a write operation.

Table A-20 CPI0 PPB expansion (continued)

Name	Direction	Width	Description
CPU0PENABLE	Output	1	This ENABLE signal indicates the second and subsequent cycles of an APB transfer.
CPU0PSELEXT	Output	1	PSEL for external PPB. This excludes the debug components inside the PIL.
CPU0PWDATA	Output	32	Write data.

A.7.6 Trace signals

The table below lists the HTM interface signals for the Trace subsystem.

Table A-21 HTM signals

Name	Direction	Width	Description
CPU0HTMDHADDR	Output	32	HTM data.
CPU0HTMDHTRANS	Output	2	HTM data.
CPU0HTMDHSIZE	Output	3	HTM data.
CPU0HTMDHBURST	Output	3	HTM data.
CPU0HTMDHPROT	Output	4	HTM data.
CPU0HTMDHWDATA	Output	32	HTM data.
CPU0HTMDHWRITE	Output	1	HTM data.
CPU0HTMDHRDATA	Output	32	HTM data.
CPU0HTMDHREADY	Output	1	HTM data.
CPU0HTMDHRESP	Output	2	HTM data.
CPU0INTERNALSTATE	Output	149	Enables the internal operation of core to be observed. OBSERVATION must be implemented to enable this signal to be used.

Table A-22 CPU trace signals

Name	Direction	Width	Description
CPU0TSVALUEB	Input	48	Global timestamp value.
CPU0TSCLKCHANGE	Input	1	Timestamp clock ratio change
CPU0ETMFIFOFULL	Output	1	ETMFIFOFULL is asserted when the ETM FIFO reaches a watermark.
CPU0ETMINTNUM	Output	9	Marks the interrupt number of the current execution context.
CPU0ETMINTSTAT	Output	3	Interrupt status.
CPU0ATBYTESETM	Output	2	ATB number of valid bytes, LSB aligned, Always tied to 0 to indicate the byte size

Table A-22 CPU trace signals (continued)

Name	Direction	Width	Description
CPU0AFREADYETM	Output	1	ATB data flush complete. This is a flush acknowledge. Asserted when buffers are flushed.
CPU0ATREADYETM	Input	1	Transfer destination ready for ETM.
CPU0ATREADYITM	Input	1	Transfer destination ready for ITM.
CPU0TPIUBAUD	Input	1	Unsynchronized baud indicator from TPIU.
CPU0TPIUACTV	Input	1	TPIU has data.
CPU0ATIDETM	Output	7	ID value for trace source.
CPU0ATVALIDETM	Output	1	Transfer data valid.
CPU0ATDATAETM	Output	8	Transfer data.
CPU0ATIDITM	Output	7	ID value for trace source.
CPU0ATVALIDITM	Output	1	Transfer data valid.
CPU0ATDATAITM	Output	8	Transfer data.
CPU0ATBYTESITM	Output	2	Transfer size (fixed to `b00`).
CPU0AFREADYITM	Output	1	ATB flush ready.
CPU0ETMTRIGOUT	Output	1	Trigger output from ETM (to TPIU).
CPU0DSYNC	Output	1	Synchronization trigger from DWT to Cortex-M3 TPIU Ignore if using CoreSight TPIU.

The table below lists the interface signals for the Trace Port Interface Unit.

Table A-23 TPIU clock reset and control

Name	Direction	Width	Description
TPIUTRACECLKIN	Input	1	Trace out port source clock.
TPIUTRESETn	Input	1	Trace out port active-LOW reset. This is asynchronously asserted and must be synchronously de-asserted to **TPIUTRACECLKIN**.
TPIUCLK	Input	1	APB and ATB interface clock.
TPIUCLKEN	Input	1	Clock enable for **TPIUCLK**.
TPIUTRACECLK	Output	1	Output clock, used by the TPA to sample the other pins of the trace out port. This runs at half the speed of **TPIUTRACECLKIN**, and data is valid on both edges of this clock (clock derived from **TPIUTRACECLKIN**).

Table A-23 TPIU clock reset and control (continued)

Name	Direction	Width	Description
TPIUTRACEDATA[3:0]	Output	1	Output data. A system might not connect all the bits of this signal to the trace port pins, depending on the number of pins available and the bandwidth required to output trace.
TPIUTRACESWO	Output	1	Serial Wire Viewer data.
TPIUSWOACTIVE	Output	1	Controls the multiplexor if SWO shared with **TRACEDATA[0]**. ARM recommends implementing SWO shared with JTAG-TDO, therefore this pin shall be left unconnected.

A.8 CPU control, status, and power management signals

This section describes power-management and control signals. Power management requires design choices made at the SoC level.

It has the following sections:

- *CPU status and control signals*.
- *Power management signals* on page A-21.

A.8.1 CPU status and control signals

The table below lists the status and control signals for the CPU subsystem.

Table A-24 CPU control and status

Name	Direction	Width	Description
CPU0HALTED	Output	1	In halting mode debug. HALTED remains asserted while the core is in debug.
CPU0MPUDISABLE	Input	1	If asserted the MPU is invisible and unusable. Tie HIGH to disable the MPU. Tie LOW to enable the MPU, if present.
CPU0ETMFIFOFULLEN	Input	1	Enable ETM FIFIO FULL feature (stall processor when ETM FIFO is full).
CPU0SLEEPING	Output	1	Indicated that the processor is in sleep mode (sleep mode).
CPU0SLEEPDEEP	Output	1	Indicates that the processor is in deep sleep mode.
CPU0SLEEPHOLDREQn	Input	1	Request to extend sleep. Can only be asserted when **CPU0SLEEPING** is HIGH.
CPU0SLEEPHOLDACKn	Output	1	Acknowledge for **CPU0SLEEPHOLDREQn**.
CPU0WAKEUP	Output	1	Active-HIGH signal to the PMU that indicates that a wake-up event has occurred and the processor system domain requires its clocks and power to be restored.
CPU0WICSENSE	Output	Implementation defined	Active HIGH set of signals. These indicate the input lines that cause the WIC to generate the WAKEUP signal. (optional, for testing).
CPU0WICENREQ	Input	1	Active-HIGH request for deep sleep to be WIC-based deep sleep. The PMU drives this.
CPU0WICENACK	Output	1	Active-HIGH acknowledge signal for WICENREQ. If you do not require PMU, then tie this signal HIGH to enable the WIC if the WIC is implemented.
CPU0TRCENA	Output	1	Active HIGH signal that indicates an Trace is Enabled maybe used to gate **TPIUCLK**.
CPU0ETMEN	Output	1	Active HIGH signal that indicates ETM is enabled maybe used to gate **TPIUCLK**.
CPU0SYSRESETREQ	Output	1	Processor control - system reset request. AIRC.SYSRESETREQ MMR controls this bit.
CPU0LOCKUP	Output	1	Indicates that the core is locked up.
CPU0BRCHSTAT	Output	4	Branch status in decode.

A.8.2 Power management signals

The table below lists the power-management signals for the debug subsystem. See also *Debug and Trace signals* on page A-14. on page A-14

Table A-25 Debug power-management signals

Signal name	Direction	Clock	Description
DAPCDBGPWRUPREQ	Output	**DAPSWCLKTCK**	Active HIGH signal that indicates an external debugger request to the PMU to power-up the debug domain. This signal must be synchronized before use.
DAPCDBGPWRUPACK	Input	Asynchronous	Active HIGH signal that indicates the debug domain is powered up in response to **DAPCDBGPWRUPREQ** being HIGH. This signal is re-synchronized internally to **DAPSWCLKTCK**.
DAPCSYSPWRUPREQ	Output	**DAPSWCLKTCK**	Active HIGH signal that indicates an external debugger request to the PMU to power-up the debug domain. This signal must be synchronized before use.
DAPCSYSPWRUPACK	Input	Asynchronous	Active HIGH signal that indicates the debug domain is powered up in response to **DAPCSYSPWRUPREQ** being HIGH. This signal is re-synchronized internally to **DAPSWCLKTCK**.
DAPNCDBGPWRDN	Input	Asynchronous	Debug infrastructure power-down control. Controls the clamps of the DAPAPB interface.
DAPCDBGRSTREQ	Output	**DAPSWCLKTCK**	Active HIGH signal that indicates an external debugger requested a debug reset to reset the controller This signal must be synchronized before use.
DAPCDBGRSTACK	Input	Asynchronous	Active HIGH signal that indicates the debug domain is reset has completed This signal is re-synchronized internally by **DAPSWCLKTCK**.

The table below lists the power-management signals for the interconnect subsystem. See also *Bus signals* on page A-11 on page A-11.

Table A-26 Interconnect power-management signals

Signal name	Direction	Clock	Description
APBQACTIVE	Output	**MTXHCLK**	APB bus active signal for global clock gating control of all APB peripherals attached to the IoT Subsystem that supports gated APB clock.
TIMER0PCLKQACTIVE	Output	**MTXHCLK**	APB bus active signal for clock gating control of Timer 0 APB clock TIMER0PCLKG (for example PSEL).
TIMER1PCLKQACTIVE	Output	**MTXHCLK**	APB bus active signal for clock gating control of Timer 1 APB clock TIMER1PCLKG (i.e. PSEL).

Table A-26 Interconnect power-management signals (continued)

Signal name	Direction	Clock	Description
FLSPCLKQACTIVE	Output	**MTXHCLK**	APB bus active signal for clock gating control of eFlash Controller APB clock FLSPCLKG (for example PSEL).
FCACHEPCLKQACTIVE	Output	**MTXHCLK**	APB bus active signal for clock gating control of eFlash cache APB clock FCACHEPCLKG.
APBTARGEXPnPSEL	Output	**MTXHCLK**	External APB TARGET SEL signals (APB Slave interface).

The table below lists the power-management signals for the eFlash subsystem. See also *eFlash signals* on page A-5.

Table A-27 eFlash and eFlash cache power-management signals

Signal name	Direction	Clock	Description
FLSSHUTDOWNREQn	Input	Asynchronous	Powered-up and power-down requests. Internally synchronized to **FLSHCLK**.
FLSSHUTDOWNACKn	Output	**FLSHCLK**	Power-down acknowledge from the eFlash controller for the eFlash banks.
FCACHERAMPWRUPREQ	Output	**FCACHEHCLK**	The eFlash cache module indicates to the PMU to requests power up the DATA and TAG RAM (from either power down or retention).
FCACHERAMPWRUPACK	Input	Asynchronous	Acknowledge from the PMU that eFlash cache RAMs are powered-up and ready to use. Internally synchronized to **FCACHEHCLK**.

A.9 Memory remap signals

The remapping of the boot memory range is controlled by the MTXREMAP signals.

Table A-28 Memory remap signals

Name	Direction	Width	Description
MTXREMAP	Input	3	REMAP[0]: The Embedded flash memory region represents the maximum supported eFlash size. REMAP[1]: If SRAM1 is not present the corresponding memory range can be mapped to the AHB expansion port. REMAP[2]: If SRAM2 is not present the corresponding memory range can be mapped to AHB expansion port. EMAP[3]: If SRAM3 is not present the corresponding memory range can be mapped to AHB expansion port.

A.10 DFT signals

The table below lists signals related to test mode.

Table A-29 DFT signals

Port name	Direction	Width	Description
DFTSCANMODECPU0	Input	1	Reset bypass to disable internal generated reset for testing (for example ATPG). Make WIC latch transparent.
DFTCGENCPU0	Input	1	Clock gating bypass to disable internal clock gating for testing. This signal is used to ensure safe shift where the clock is forced on during the shift mode.

Appendix B
Revisions

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

Table B-1 Issue A

Change	Location	Affects
First release	-	-