Arm® Corstone™-1000

Revision: r0p0

Technical Overview

Non-Confidential

Issue 03

102360_0000_03_en

Arm® Corstone™-1000
Technical Overview

Release Information

Document history

Issue	Date	Confidentiality	Change
0000-01	15 April 2021	Non-Confidential	First EAC release for r0p0
0000-02	15 July 2021	Non-Confidential	Second EAC release for r0p0
0000-03	15 June 2022	Non-Confidential	First REL release for r0p0

Proprietary Notice

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

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

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

This document may include technical inaccuracies or typographical errors.

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

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

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

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

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

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

Product Status

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

Feedback

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

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

Inclusive language commitment

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

Previous issues of this document included language that can be offensive. We have replaced this language. See A. Revisions on page 64.

To report offensive language in this document, email terms@arm.com.

Contents

1. Introduction

1.1 Product revision status

The r*x*p*y* identifier indicates the revision status of the product described in this manual, for example, r*1*p*2*, where:

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

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

1.2 Intended audience

This book is written for hardware or software engineers who want an overview of the components and functionality in Corstone-1000.

1.3 Conventions

The following subsections describe conventions used in Arm documents.

Glossary

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

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

Typographic conventions

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

Convention	Use
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`<and>`	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: `MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>`
SMALL CAPITALS	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, **IMPLEMENTATION DEFINED**, **IMPLEMENTATION SPECIFIC**, **UNKNOWN**, and **UNPREDICTABLE**.

Caution

Recommendations. Not following these recommendations might lead to system failure or damage.

Warning

Requirements for the system. Not following these requirements might result in system failure or damage.

Danger

Requirements for the system. Not following these requirements will result in system failure or damage.

Note

An important piece of information that needs your attention.

Tip

A useful tip that might make it easier, better or faster to perform a task.

Remember

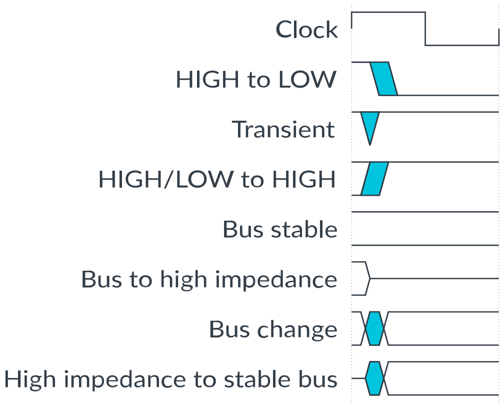
A reminder of something important that relates to the information you are reading.

Timing diagrams

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

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

Figure 1-1: Key to timing diagram conventions



Signals

The signal conventions are:

Signal level

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

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

Lowercase n

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

1.4 Additional reading

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

Table 1-2: Arm Publications

Document name	Document ID	License only Y/N
Arm® Corstone™ SSE-710 Subsystem Technical Reference Manual	102342	N
Arm® Cortex®-M System Design Kit Technical Reference Manual	DDI 0479	N
Arm® Cortex®-M0+ Technical Reference Manual	DDI 0484C	N

Document name	Document ID	License only Y/N
Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual	101526	N
Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual	DDI 0571	N
Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual	100806	N
Arm® CoreSight™ STM-500 System Trace Macrocell Technical Reference Manual	DDI 0528	N
Arm® CoreLink™ GFC-200 Generic Flash Controller Technical Reference Manual	101484	N
Arm® CoreLink™ GFC-100 Generic Flash Controller Technical Reference Manual	101059	N
Arm® CoreLink™ PCK-600 Power Control Kit Technical Reference Manual	101150	N
Arm® CoreSight™ SDC-600 Secure Debug Channel Technical Reference Manual	101130	N
Arm® CoreLink™ NIC-450 Network Interconnect Technical Overview	100459	N
Arm® CoreLink™ XHB-500 Bridge Technical Reference Manual	101375	N
Arm® CoreLink™ LPD-500 Low Power Distributor Technical Reference Manual	100361	N
Arm® CoreLink™ CG092 AHB Flash Cache Technical Reference Manual	DDI 0569	N
PrimeCell UART (PL011) Technical Reference Manual	DDI 0183	N
Arm® PrimeCell Real Time Clock (PL031) Technical Reference Manual	DDI 0224C	N
Arm® Cortex®-M3 Processor Technical Reference Manual	100165	N
Arm® Cortex®-M33 Processor Technical Reference Manual	100230	N
Arm® Cortex®-M23 Processor Technical Reference Manual	DDI 0550	N
CoreLink™ NIC-400 Network Interconnect Technical Reference Manual	DDI 0475	N
Arm® CoreLink™ GIC-400 Generic Interrupt Controller Technical Reference Manual	DDI 0471B	N
Arm® CoreLink™ QoS-400 Network Interconnect Advanced Quality of Service Supplement to Arm® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual	DSU 0026	N
Arm® CoreLink™ QVN-400 Network Interconnect Advanced Quality of Service using Virtual Networks Supplement to Arm® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual	DSU 0027	N
Arm® CoreLink™ TLX-400 Network Interconnect Thin Links Supplement to Arm® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual	DSU 0028	N
Arm® CoreLink™ AXI4 to AHB-Lite XHB-400 Bridge Technical Reference Manual	DDI 0523	N
PrimeCell Infrastructure AMBA® 3 AXI Internal Memory Interface (BP140) Technical Overview	DTO 0009	N
Arm® v8-M Architecture Reference Manual	DDI 0553	N
Arm® Generic Interrupt Controller Architecture Specification	IHI 0048B.b	N
Arm® Corstone™ SSE-710 Subsystem Configuration and Integration Manual	102343	Y
Arm® Corstone™-1000 Cryptographic Extensions Technical Reference Manual Addendum	102276	Y
Arm® CoreLink™ NIC-400 Network Interconnect Implementation Guide	DII 0273	Y
Arm® CoreLink™ NIC-400 Network Interconnect Integration Manual	DII 0269	Y
Arm® Cortex®-M System Design Kit Example System Guide	DUI 0594	Y
Arm® Cortex®-M0 and Cortex®-M0+ System Design Kit Example System Guide	DUI 0559	Y
Arm® Cortex®-M0+ Integration and Implementation Guide	DIT 0032B	Y
Arm® Cortex®-M0+ User Guide Reference Material	DUI 0605B	Y
Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Configuration and Integration Manual	101527	Y
Arm® CoreLink™ SIE-200 System IP for Embedded Configuration and Integration Manual	DIT 0067	Y
Arm® CoreLink™ GFC-200 Generic Flash Controller Configuration and Integration Manual (101485)	101485	Y

Document name	Document ID	License only Y/N
Arm® CoreLink™ GFC-100 Generic Flash Controller Configuration and Integration Manual	101060	Y
Arm® CoreLink™ PCK-600 Power Control Kit Configuration and Integration Manual	101151	Y
Arm® CoreSight™ SDC-600 Secure Debug Channel Configuration and Integration Manual	101131	Y
Arm® CoreLink™ LPD-500 Low Power Distributor Integration and Implementation Manual	100362	Y
Arm® CoreLink™ CG092 AHB Flash Cache Configuration and Integration Manual	DIT 0065B	Y
PrimeCell Infrastructure AMBA® 3 AXI Internal Memory Interface (BP140) Design Manual	DDI 0334	Y
Arm® CoreLink™ ADB-400 AMBA® Domain Bridge User Guide	DUI 0615	Y
Arm® CoreLink™ XHB-500 Configuration and Integration Manual AXI5 to AHB5 bridge and AHB5 to AXI5 bridge	101376	Y
Arm® CoreSight™ System-on-Chip SoC-600 Configuration and Integration Manual	100807	Y
Arm® Corstone™ SSE-710 Subsystem Release Note	CG071-DC-06003	Y
Arm® Corstone™-1000 Release Note	BP319-DC-06003	Y
Arm® Corstone™-1000 Cryptographic Extension Release Note	BP320-DC-06003	Y
Arm® CryptoCell™-312 Technical Reference Manual	100774	Y
Arm® CryptoCell™-312 Configuration and Integration Manual	100775	Y

- See www.arm.com/cmsis for embedded software development resources including the *Cortex Microcontroller Software Interface Standard* (CMSIS).
- See Arm® Mbed™ platform, https://www.mbed.com for information on the Mbed™ tools including Mbed™ OS and online tools.

Note

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

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

2. Overview of Corstone-1000

This chapter describes Corstone-1000.

2.1 Corstone-1000

Corstone-1000 includes a single package.

Note

Each package grants licenses to a set of IP product components. These components must be downloaded separately.

- Pre-verified Corstone SSE-710 subsystem that provides a flexible compute architecture that combines Cortex®-A and Cortex®-M processors.
- Support for Cortex®-A32, Cortex®-A35 and Cortex®-A53 processors.
- Two expansion systems for M-Class (or other) processors for adding sensors, connectivity, video, audio and machine learning at the edge
- System and security IPs to build a secure SoC for a range of rich IoT applications, for example gateways, smart cameras and embedded systems.
- Integrated Secure Enclave providing hardware Root of Trust and supporting seamless integration of the optional CryptoCell™-312 cryptographic accelerator
- Designed to target PSA Certified Level 2 and System Ready IR certifications.

See *Arm® Corstone™-1000 Release Note* for the component versions.

2.2 Product structure

Corstone-1000 includes licenses to the following subsystems: security IP, and system IP.

The following table lists IP that enables extension and modification of the Corstone SSE-710 subsystem. The table is followed by a descriptive list of the Subsystems, Security, and System IP available.

Table 2-1: Corstone-1000

Name	Description	Used in subsystem	Used in integration layer	Useful for extension
Corstone SSE-710 subsystem including Secure Enclave, firewall and MHU	Main subsystem	Yes	No	No
Cortex-M0+	-	Yes	No	Yes

Name	Description	Used in subsystem	Used in integration layer	Useful for extension
CoreLink NIC-450 network interconnect for AXI with options	Main interconnect	Yes	Yes	Yes
CoreLink XHB-500 – AXI to AHB bridge	Bridging component	No	Yes	Yes
CoreSight STM-500 System Trace Macro	Part of debug	Yes	No	No
CoreLink PCK-600 Power Control Kit	Power control elements. Includes LPD-500 Low Power InterfaceDistributor	Yes	Yes	Yes
CoreSight SDC-600 Secure Debug Channel	Secure debug interface	Yes	No	No
CoreSight SoC-600	Debug infrastructure	Yes	Yes	Yes
IntMemAxi	SRAM controller	No	Yes	No
UART	Serial port	Yes	No	Yes
CM0SDK / CMSDK Cortex-M System Design Kit	System IP for Cortex-M	Yes	Yes	Yes
AHB Flash Cache	Useful if using an eFlash	No	No	Yes
RTC Real Time Clock	Reference time	No	Yes	No
CoreLink™ SIE-300 AXI5 System IP for Embedded	AXI5 Security aware components	NO	Yes	Yes
CoreLink SIE-200 System IP for AHB5	Necessary for Arm V8-M systems	Yes	No	Yes
CoreLink GFC-100 Flash Controller (single port)	Useful if using an eFlash	No	Yes	No
CoreLink GFC-200 Flash Controller (secure dual port)	Useful if using an eFlash	No	No	Yes
CoreLink GIC-400 Generic Interrupt Controller	Interrupt controller for the chosen A class processor.	Yes	No	No

IP not included in Corstone-1000

The Crypto Accelerator, Cortex processors, ISP, video, NPUs and so on are not included in the Corstone-1000 and are licensed separately. Contact your licensing manager to learn more about your licensing options.

Table 2-2: Corstone-1000 Cryptographic Extension

Name	Description	Used in subsystem	Used in integration layer	Useful for extension
CryptoCell™-312	Crypto Accelerator	Yes	No	No
Corstone-1000 Cryptographic Extension	Provides fully verified integration of CryptoCell™-312 into Crypto Accelerator Socket of Secure Enclave	Yes	No	No

Corstone-1000 subsystems

SSE-710 subsystem

The Corstone SSE-710 subsystem contains a Secure Enclave subsystem. The Secure Enclave is a Cortex-M0+ based security subsystem that acts as the RoT for the system.
See 3.1 SSE-710 overview on page 19

Cortex®-M System Design Kit (CMSDK)

The CMSDK provides example systems for the Cortex®-M0, Cortex®-M0+, Cortex®-M3, and Cortex®-M4 cores, with reusable AMBA® components for system-level development.
See 3.3 Cortex-M System Design Kit on page 22.

Security and System IP

CryptoCell-312

Arm® CryptoCell™-312 is an embedded security solution for high-efficiency systems, with emphasis on small footprint and low power-consumption.

See 3.2 CryptoCell-312 on page 21.

CoreLink™ SIE-300 AXI5 System IP for Embedded

The CoreLink™ SIE-300 AXI5 System IP for Embedded provides a set of configurable AXI5 security-aware components. The components can protect peripherals and memories that are unaware of security, so that a peripheral or memory is only accessible to trusted software. The SIE-300 also provides clock synchronizing bridges and an access control gate.

See 3.5 CoreLink SIE-300 AXI5 System IP for Embedded on page 25.

CoreLink™ SIE-200 System IP for Embedded

SIE-200 is a collection of interconnect, peripheral, and TrustZone® controller components for use with a core that complies with the Arm®v8-M core architecture.
See 3.6 SIE-200 System IP for Embedded on page 34.

CoreLink™ NIC-450 Network Interconnect

NIC-450 is a library of IP that supports the creation of optimized, multi-domain, AMBA® interconnect solutions.
See 3.7 NIC-450 Network Interconnect on page 35.

CoreSight™ System-on-Chip SoC-600

SoC-600 is a member of the Arm embedded debug and trace component family. It supports the Arm® Debug Interface v6 and CoreSight v3 Architectures that enable you to build debug and trace functionality into your systems and to support debug and trace over existing functional interfaces.

See 3.8 SoC-600 on page 37.

CoreSight™ SDC-600 Secure Debug Channel

SDC-600 provides a dedicated channel for authentication between an external debugger and a debug target platform by using an unlocking mechanism.
See 3.9 SDC-600 Secure Debug Channel on page 39.

CoreSight™ STM-500 System Trace Macrocell

STM-500 is a trace source that is integrated into a CoreSight system, and that is designed primarily for high-bandwidth trace of instrumentation embedded into software.
See 3.10 STM-500 System Trace Macrocell on page 41.

CoreLink™ PCK-600 Power Control Kit

PCK-600 provides a set of configurable RTL components so you can create SoC clock and power control infrastructure. The components use the Arm® Q-Channel and P-Channel low-power interfaces.
See 3.11 PCK-600 Power Control Kit on page 43.

CoreLink™ GFC-200 Generic Flash Controller

GFC-200 comprises the generic part of a Flash controller in a SoC, so you can easily integrate an embedded Flash macro into your system. The GFC-200 supports accesses from two managers that can operate in separate domains, such as a Non-secure domain and a Secure domain.
See 3.12 GFC-200 Generic Flash Controller on page 44.

CoreLink™ GFC-100 Generic Flash Controller

GFC-100 comprises the generic part of a Flash controller in a SoC. GFC-100 enables an embedded Flash macro to be integrated easily into your system.
See 3.13 GFC-100 Generic Flash Controller on page 47.

CoreLink™ LPD-500 Low Power Distributor

LPD-500 is a standalone configurable component to distribute Q-Channel interfaces to multiple devices and subsystems. You can use Q-Channels to manage clock gating and power control. In the Corstone-700 system.. For any other clock gating and power control purposes, PCK-600 should be used.
See *Arm® CoreLink™ LPD-500 Low Power Distributor Technical Reference Manual*.

CoreLink™ CG092 AHB Flash Cache

CG092 is an instruction cache that is instantiated between the bus interconnect and the *embedded Flash* (eFlash) controller.
See 3.14 Corelink AHB Flash Cache on page 50.

PrimeCell Infrastructure AMBA® 3 AXI™ Internal Memory Interface (BP140)

You can use the `IntMemAxi` component to interface to a synthesized SRAM. It can also be connected to ROM.
See 3.15 AXI Internal Memory Interface (BP140) on page 52.

CoreLink™ XHB-500 Bridge

The XHB-500 product provides an AMBA® AXI5 to AHB5 bridge and an AHB5 to AXI5 bridge.
See 3.16 XHB-500 bridge on page 53.

UART (PL011)

The UART is an *Advanced Microcontroller Bus Architecture* (AMBA®) compliant *System-on-Chip* (SoC) peripheral that is developed, tested, and licensed by Arm.
See 3.17 UART on page 57.

Real Time Clock (PL031)

The *Real Time Clock* (RTC) is an AMBA® subordinate module that connects to the *Advanced Peripheral Bus* (APB). A 1Hz clock input to the RTC generates counting in one second intervals. The RTC provides an alarm function or long time base counter by generating an interrupt signal after counting a programmed number of cycles of the clock input.
See 3.18 Real Time Clock on page 58.

GIC-400 Generic Interrupt Controller

The GIC-400 is an interrupt controller compliant with *Arm® Generic Interrupt Controller Architecture Specification* version 2.0.

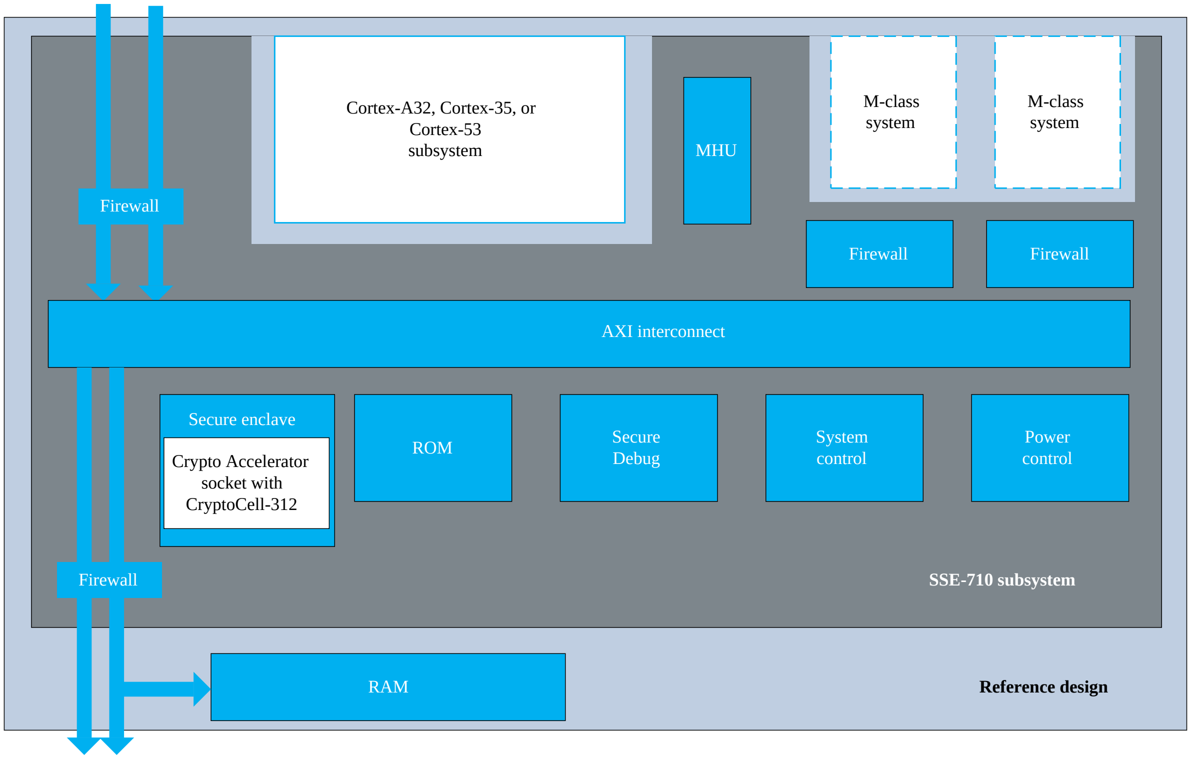
See 3.19 GIC-400 Generic Interrupt Controller on page 59.

2.3 Corstone SSE-710 subsystem features

Corstone SSE-710 subsystem is a flexible subsystem designed for secure, rich *Internet of Things* (IoT) applications. It integrates Arm® Cortex-A and Arm® Cortex-M processors together, with built-in security as one of its primary features.

The Corstone SSE-710 subsystem contains a Secure Enclave subsystem. The Secure Enclave is a Cortex-M0+ based security subsystem that acts as the RoT for the system. It holds and generates keys, and provides cryptographic services and security controls to the host system. Corstone-1000 Cryptographic Extension integrates CryptoCell-312 as a cryptographic accelerator inside the Secure Enclave.

Figure 2-1: Corstone SSE-710 subsystem design



Key features of the Corstone SSE-710 subsystem are:

- Host system based on Linux-capable Cortex-A32, Cortex-A35, or Cortex-A53 processor.

- Up to two external systems, M-Class or other, for application-specific processing.
- Dedicated integrated Secure Enclave based on Cortex-M0+ to provide root of trust and cryptographic acceleration functions.
- Interrupt routing logic to distribute interrupts to the all the processing elements.
- Inter-processor communication handled by the *Message Handling Unit* (MHU), supporting for both Secure and Non-secure messages among all the processing elements.
- Extensive system security, with full TrustZone® support and hardware compartmentalization via the firewall.
- Pre-built advanced power management for hardware control of power and clock domains.
- Secure and authenticated system debug for all processing elements.
- CryptoCell-312 is preintegrated into the Secure Enclave.
- Secure Enclave is PSA (Platform Security Architecture) Level 2 certification ready, Corstone-1000 provides a PSA Level 2 Certification Guidance document.

The Corstone SSE-710 subsystem components form just part of the SoC. You must extend and customize the subsystems for your application requirements.

To create a SoC, extend the Corstone SSE-710 subsystem. A complete system typically contains the following components:

Compute subsystem

The Corstone SSE-710 subsystem consists of one to four Cortex®-A32, Cortex-A35, or Cortex-A53 processor processor cores and associated bus, debug, controller, peripherals, and interface logic.

Memory and peripherals

An extended SoC requires extra memory, control, and peripheral components beyond the minimum subsystem components. Flash memory, for example, is not provided with the Corstone SSE-710 subsystem, but can be added through implemented interfaces.

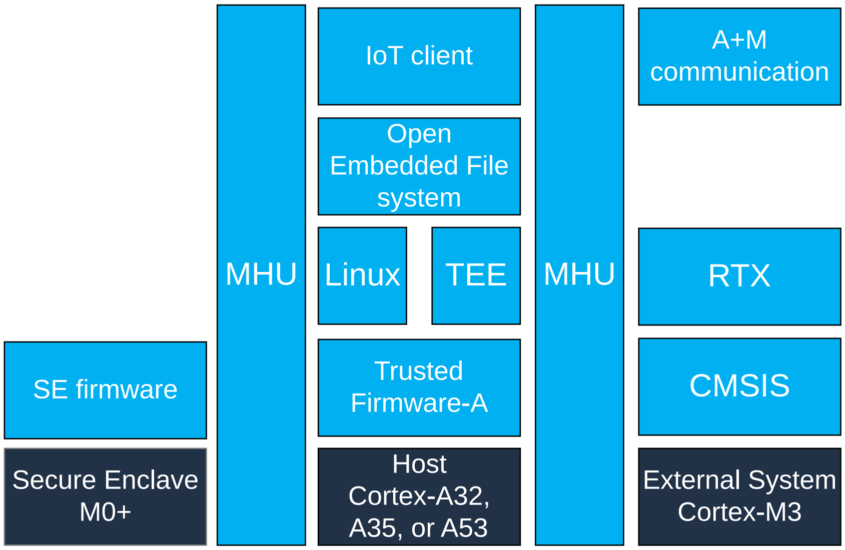
Sensors and actuators

The reference design can be extended by adding sensors or actuator logic, such as temperature input or motor control output.

Software development environment

A reference Open Source software stack is available for Corstone-1000. This enables product software development and helps bootstrap a variety of Arm® Partner software and product solutions.

Figure 2-2: Reference Open Source software stack



The same software stack runs on the Corstone-1000 *Fixed Virtual Platform* (FVP) model and Corstone-1000 for MPS3 (AN550) FPGA. These are freely and publicly available.

This software stack provides:

- A reference OS/RTOS port for each of the compute components, including a Cortex-A hosted Linux stack with a reduced memory footprint
- Example drivers for key IP components

Supporting code is available, either directly in the relevant upstream projects or via public-facing git repositories hosted by Linaro.

For information on the software stack, FPGA, and FVP, see Arm Ecosystem FVPs.

For information on how to use the components that are licensed by Corstone-1000, see the relevant component IP product documentation, starting with their Technical Reference Manual.

3. Corstone-1000 IP descriptions

This chapter describes the IP products included in the Corstone-1000.

3.1 SSE-710 overview

This section gives an overview of the product and its features.

For more information, see the SSE-710 documentation set:

- *Arm® Corstone™-710 Subsystem Technical Reference Manual*
- *Arm® Corstone™-710 Subsystem Configuration and Integration Manual*
- *Arm® Corstone™-1000 Cryptographic Extensions Technical Reference Manual Addendum*

3.1.1 About Corstone SSE-710 subsystem

The Corstone SSE-710 subsystem (SSE-710) is a flexible subsystem designed to provide a secure solution for rich *Internet of Things* (IoT) applications based on Arm® supported Cortex®-A processors, Cortex®-M, or other managers that are present in External Systems.

SSE-710 is designed to cover a range of *Power, Performance, and Area* (PPA) applications, and enable extension for use-case specific applications, for example, sensors, cloud connectivity, and edge compute.

Connected embedded devices are exposed to many different security threats. SSE-710 implements the reference subsystem for an SoC that targets secure, rich IoT applications. Built-in security is one of its key features.

SSE-710 consists of:

- A reference subsystem
- An example integration layer
- An example Cortex®-A32, Cortex®-A35, Cortex®-A53 (the supported Cortex®-A processors for SSE-710), and Arm® GIC-400 integration.
- A set of documentation including PSA L2 certification Guidance document for Secure Enclave.

To create a SoC, the Corstone SSE-710 subsystem subsystem must be extended. A complete system typically contains the following components:

Compute subsystem

In SSE-710 the compute subsystem consists of one to four supported Cortex®-A processors, processor cores and associated bus, debug, controller, peripherals, and interface logic.

Memory and peripherals

An extended SoC requires extra memory and peripheral components beyond the minimum subsystem components. Flash memory, for example, is not provided with the SSE-710, but can be added through implemented interfaces.

Sensors and actuators

The reference design can be extended by adding sensors or actuator logic, such as temperature input or motor control output.

Software development environment

Arm® provides a complete software development environment, which includes the Arm® Mbed™ *Operating System* (OS) and Linux, Mbed™ or GNU (GCC) compilers and debuggers, and firmware.

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

3.1.2 Features of SSE-710

SSE-710 provides the minimum set of features for an SoC.

You can configure some aspects of the SSE-710 design so that you can meet the specific requirements of your SoC and intended application. For example, you can configure the number of shared interrupt inputs or Host System firewalls.

SSE-710 standardizes the following product features:

- Memory and interrupt maps of the Host System and Secure Enclave system.
- Hardware address compartmentalization.
- Boot and security, including a hardware isolated *Root of Trust* (RoT).
- Communication between different parts of the SoC.
- Timer and watchdog infrastructure.
- Debug requirements.
- Power, clock, and reset control.
- Requirements for adding new or existing External Systems for use-case specific applications.
- Requirements for adding use-case specific managers and peripherals to the Host System.

The SSE-710 includes the following key components:

- A Cortex®-A processor cluster and *Generic Interrupt Controller* (GIC) socket to support up to four Cortex®-A cores, and a GIC-400.
- A Secure Enclave socket into which you can integrate a Crypto Accelerator. The socket is based on a Cortex®-M0 processor with dedicated SRAM, ROM, and peripherals, such as timers and watchdogs.

- Secure Enclave contains a Crypto Accelerator socket with Arm® CryptoCell™-312 pre-integrated, but other cryptographic engines can also be integrated.
- Two External System Harnesses support integration of External Systems into SSE-710.
- *Message Handling Units* (MHUs) for communication between the different systems in the SSE-710. An MHU provides a unidirectional communication channel, therefore MHUs are provided in pairs to allow for bidirectional communication:
    - Two pairs of MHUs for communication between supported Cortex®-A processors and Secure Enclave.
    - Two pairs of MHUs for communication between supported Cortex®-A processors and each External System.
    - Two pairs of MHUs for communication between Secure Enclave and each External System.
- Interrupt Router to handle routing interrupts from shared peripherals to the interrupt controller of a specific system
- Firewall to provide:
    - Hardware compartmentalization of the Host System address space
    - Translation between the address space of the system and the External System into the address space of the Host System
- Common debug infrastructure supporting single and multi-system debug, by self-hosted and external debug agents.
- Certificate-based debug authentication is supported by SDC-600 Secure Debug Channel.
- Advanced hardware autonomous power control design.
- Shared peripherals, such as counters, timers, and watchdog, that any system in the SSE-710 can use.

The integrator can tailor the final implementation to the target use-cases by using:

- The supported configuration options.
- The sockets for the Host processor, Host GIC, and External Systems.
- The expansion interfaces of SSE-710.

3.2 CryptoCell-312

This section gives an overview of the product and its features.

For more information, see the CryptoCell™-312 documentation set:

- *Arm® CryptoCell™-312 Technical Reference Manual*
- *Arm® CryptoCell™-312 Configuration and Integration Manual*

3.2.1 Introduction

Arm® CryptoCell™-312 is an embedded security solution for high-efficiency systems, with emphasis on small footprint and low power-consumption.

It offers platform security services and a rich set of cryptographic services that target multiple threats. The services that CryptoCell™-312 offers are needed across various IoT domains. For example, home automation, factory automation, smart energy, Industrial IoT, and any other domain where there is potential usage of a Cortex®-M processor.

CryptoCell™-312 provides the following benefits:

- Improved user experience – achieved through HW-based performance acceleration, when compared to a SW-based implementation of the services that CryptoCell™-312 provides.
- Improved energy consumption – critical to battery operated devices. The improvement is achieved through dedicated HW-based implementation of security-related functions.
- Reduction of trust in SW – achieved through HW-based management of roots-of-trust, which is critical for single-execution environment systems such as the ones based on Arm® ARMv7-M.
- Fulfillment of various TEE aspects requiring HW components. For example, random number generation.

3.2.2 Features

Arm® CryptoCell™-312 uses a variety of security features to protect your assets.

CryptoCell™-312 supports the following features:

- Cryptographic acceleration for the protection of data-in-transit and data-at-rest.
- Multiple modes and algorithms.
- Protection of various assets (for example, code or data), belonging to various stakeholders (for example, ICV, OEM, or service provider).
  These asset-protection features include:
  - Image verification at boot or during runtime.
  - Authenticated debug.
  - Random number generation.
  - Security life-cycle-state management.
  - Asset Provisioning.
  - HW OTP keys masking.
  - Test Chip mode or Production Chip mode.
  - Secure provisioning support.

3.3 Cortex-M System Design Kit

This section gives an overview of the product and its features.

For more information, see the CMSDK documentation set:

- *Arm® Cortex®-M System Design Kit Technical Reference Manual*
- *Arm® Cortex®-M System Design Kit Example System Guide*
- *Arm® Cortex®-M0 and Cortex®-M0+ System Design Kit Example System Guide*

3.3.1 About the Cortex-M System Design Kit

The Cortex®-M System Design Kit helps you design products using Arm® Cortex®-M processors.

The design kit contains the following:

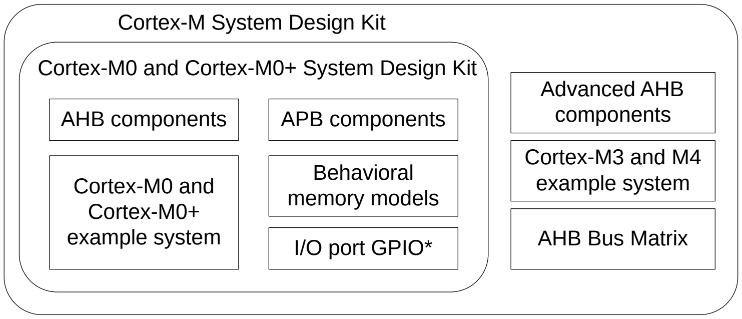
- A selection of AHB-Lite and APB components, including several peripherals such as GPIO, timers, watchdog, and UART
- Example systems for the Cortex-M0, Cortex-M0+, Cortex-M3, and Cortex-M4 cores
- Example synthesis scripts for the example systems
- Example compilation and simulation scripts for the Verilog environment that supports ModelSim, VCS, and NC Verilog
- Example code for software drivers
- Example test code to demonstrate various operations of the systems
- Example compilation scripts and example software project files that support:
    - Arm Development Studio 5 (DS-5)
    - Arm RealView Development Suite
    - Keil® *Microcontroller Development Kit* (MDK)
    - GNU tools for Arm embedded processors (Arm GCC).

The Cortex-M System Design Kit is available as:

- Cortex-M0 and Cortex-M0+ System Design Kit. This supports Cortex-M0 and Cortex-M0+.
- Cortex-M System Design Kit, full version. This supports Cortex-M0, Cortex-M0+, Cortex-M3, and Cortex-M4.

The other differences between the Cortex-M0 and Cortex-M0+ version, and the Cortex-M version of the design kit are the example systems, and the components provided.

Figure 3-1: Difference between the two versions of the design kit


Cortex-M System Design Kit
Cortex-M0 and Cortex-M0+ System Design Kit
AHB components
APB components
Cortex-M0 and Cortex-M0+ example system
Behavioral memory models
I/O port GPIO*
Advanced AHB components
Cortex-M3 and M4 example system
AHB Bus Matrix

* For use with the Cortex-M0+ directly, or as a subcomponent within AHB GPIO module.

3.4 Cortex®-M0+ Processor

This section gives an overview of the product and its features.

For more information, see:

- *Arm® Cortex®-M0+ Technical Reference Manual*
- *Arm® Cortex®-M0+ User Guide Reference Material*
- *Arm® Cortex®-M0+ Integration and Implementation Guide*

3.4.1 About the Cortex-M0+ processor

The Cortex-M0+ processor is a very low gate count, highly energy efficient processor that is intended for microcontroller and deeply embedded applications that require an area optimized, low-power processor.

The processor features and benefits are:

- Tight integration of system peripherals reduces area and development costs
- Arm® Thumb® instruction set combines high code density with 32-bit performance
- Support for single-cycle I/O access
- Power control optimization of system components
- Integrated sleep modes for low power consumption
- Fast code execution enables running the processor with a slower clock or increasing sleep mode time
- Optimized code fetching for reduced flash and ROM power consumption
- Hardware multiplier
- Deterministic, high-performance interrupt handling for time-critical applications

- Deterministic instruction cycle timing
- Support for system level debug authentication
- Serial Wire Debug reduces the number of pins required for debugging
- Support for optional instruction trace

3.5 CoreLink™ SIE-300 AXI5 System IP for Embedded

This section gives an overview of the product and its features.

For more information, see the CoreLink™ SIE-300 documentation set:

- *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual*
- *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Configuration and Integration Manual*

3.5.1 About the AXI5 System IP for Embedded

The CoreLink™ SIE-300 AXI5 System IP for Embedded provides a set of configurable AXI5 security-aware components. The components can protect peripherals and memories that are unaware of security, so that a peripheral or memory is only accessible to trusted software. The SIE-300 also provides clock synchronizing bridges and an access control gate.

The SIE-300 consists of the following components:

Manager Security Controller (MSC)

The MSC acts as security gate for AXI transactions, and it can transform the security attribute.

Memory Protection Controller (MPC)

The MPC acts as security gate for AXI transactions that target a memory interface. The security checks operate on block or page level, and are programmable by using the APB controller interface.

Peripheral Protection Controller (PPC)

The PPC gates AXI5 transactions to, and responses from, peripherals when a security violation occurs.

Access Control Gate (ACG)

The ACG component can be placed on a clock or power domain boundary to pass or block AXI5 transactions whenever the downstream component cannot accept the transaction, or is explicitly asked not to do so. The transaction is latched internally and the ACG generates automatic responses when necessary.

Sync-Down Bridge (SDB)

The SDB synchronizes AXI5 interfaces where the upstream side is faster than the downstream side, and the clocks are synchronous, in phase, and have an N:1 frequency ratio.

Sync-Up Bridge (SUB)

The SUB synchronizes AXI5 interfaces where the upstream side is slower than the downstream side, and the clocks are synchronous, in phase, and have a 1:N frequency ratio.

SRAM Memory Controller (SMC)

The SMC enables on-chip synchronous RAM blocks to attach to an AXI5 interface. The SMC supports 32, 64, 128, or 256-bit SRAM with byte writes.

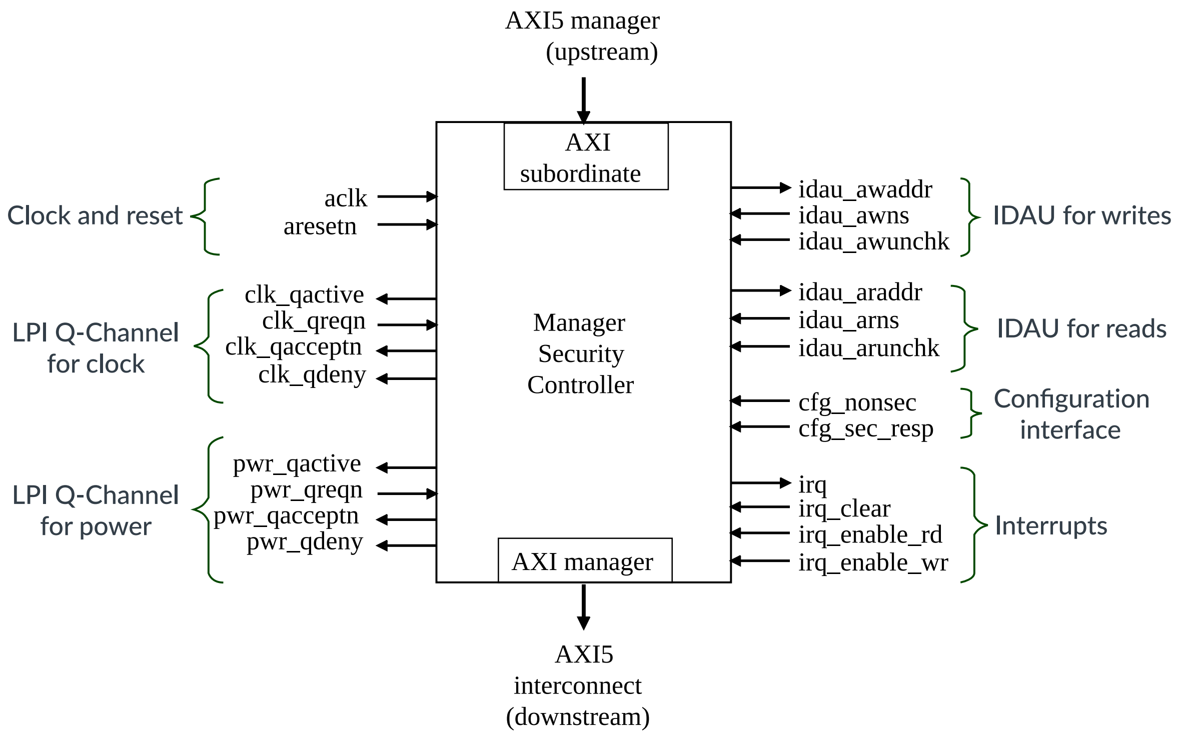
3.5.2 About the MSC

The *Manager Security Controller* (MSC) acts as security gate for AXI transactions, and it can transform the security attribute.

The MSC enables AXI managers that are designed for A-class systems to be inserted into M-class systems. Since A-class and M-class systems handle security differently, the MSC can transform the security attributes of a transaction to satisfy the M-class requirements.

The following figure shows the MSC interfaces.

Figure 3-2: MSC interfaces



The AXI subordinate and AXI manager interfaces provide the AXI data path from the AXI manager to the interconnect.

To support low-power quiescence, the MSC has two Q-Channel interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

Configuration interface

The **cfg_nonsec** input tells the MSC whether the AXI5 manager, which connects to its subordinate interface, is in the Secure state or the Non-secure state. The MSC uses this information to control whether it blocks a transaction from going downstream.

When the MSC blocks a transaction, the **cfg_sec_resp** controls whether the MSC:

- Responds with an AXI subordinate error (SLVERR).
- Ignores a write transaction or returns zero for a read transaction.

IDAU interfaces

The MSC has two *Implementation Defined Attribution Unit* (IDAU) interfaces that it uses to discover the Security state of an addressed region. One IDAU is for read transactions and the other IDAU is for write transactions.

When the MSC receives an AXI transaction, it accesses the corresponding IDAU and retrieves the Security state for that transaction address. By using the Security state information, the incoming AXI access permissions (**AxPROT**), and the state of **cfg_nonsec**, the MSC can do one of the following:

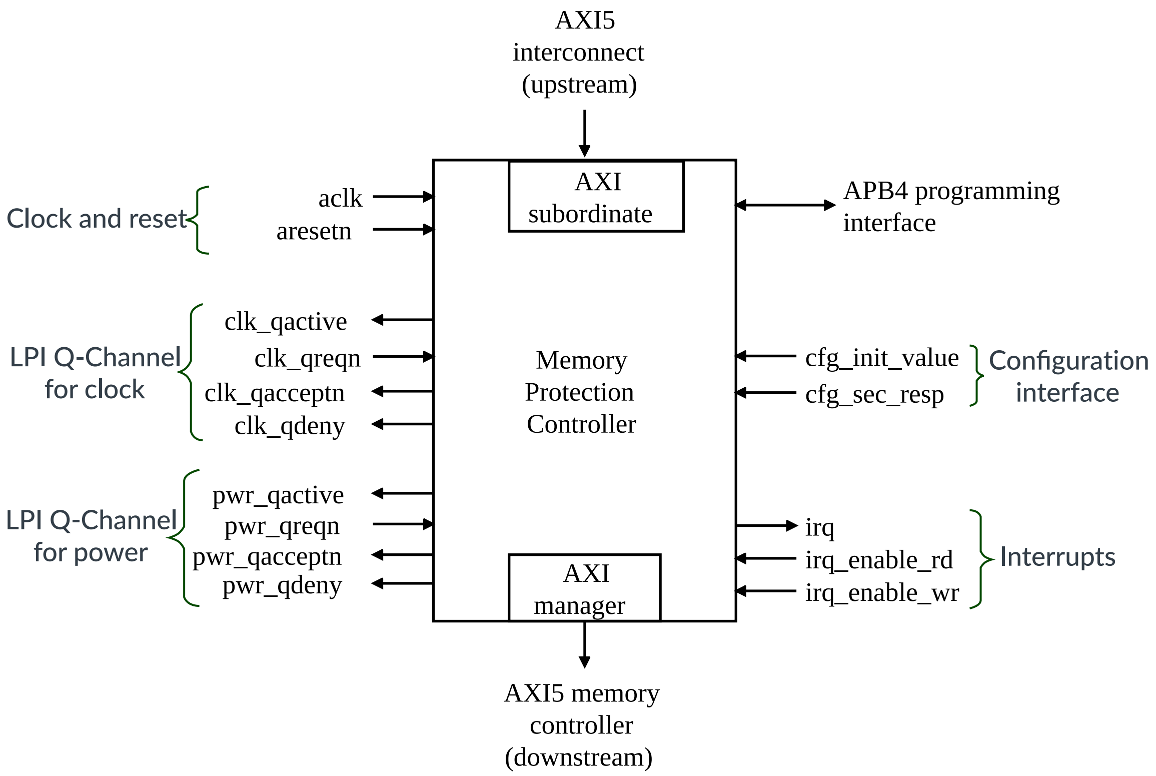
- Block the transaction from going downstream.
- Forward the transaction.
- Transform the security attributes and then forward the transaction.

3.5.3 About the MPC

The *Memory Protection Controller* (MPC) acts as security gate for AXI transactions that target a memory interface. The security checks operate on block or page level, and are programmable by using the APB completer interface.

The following figure shows the MPC interfaces.

Figure 3-3: MPC interfaces



The AXI subordinate and AXI manager interfaces provide the AXI data path from the interconnect to the memory controller.

To support low-power quiescence, the MPC has two Q-Channel interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

Configuration interface

At powerup, the MPC uses the value of the **cfg_init_value** input as the initialization value for the *Look Up Table* (LUT) to be Secure or Non-secure for the entire memory range that the MPC protects.

If a security violation occurs, the MPC generates an interrupt and the **cfg_sec_resp** controls whether the MPC:

- Responds with an AXI subordinate error (SLVERR).
- Ignores a write transaction or returns zero for a read transaction.

Note

- When accessing all internal registers (except for the PID/CID registers) with a Non-secure APB transaction, the response is either an error or RAZ/WI depending on the value of the **cfg_sec_resp** input signal.

- When accessing all internal registers (except for the PID/CID registers) with a Secure but unprivileged APB transaction, the response is always RAZ/WI, regardless of the value of the **cfg_sec_resp** input signal.

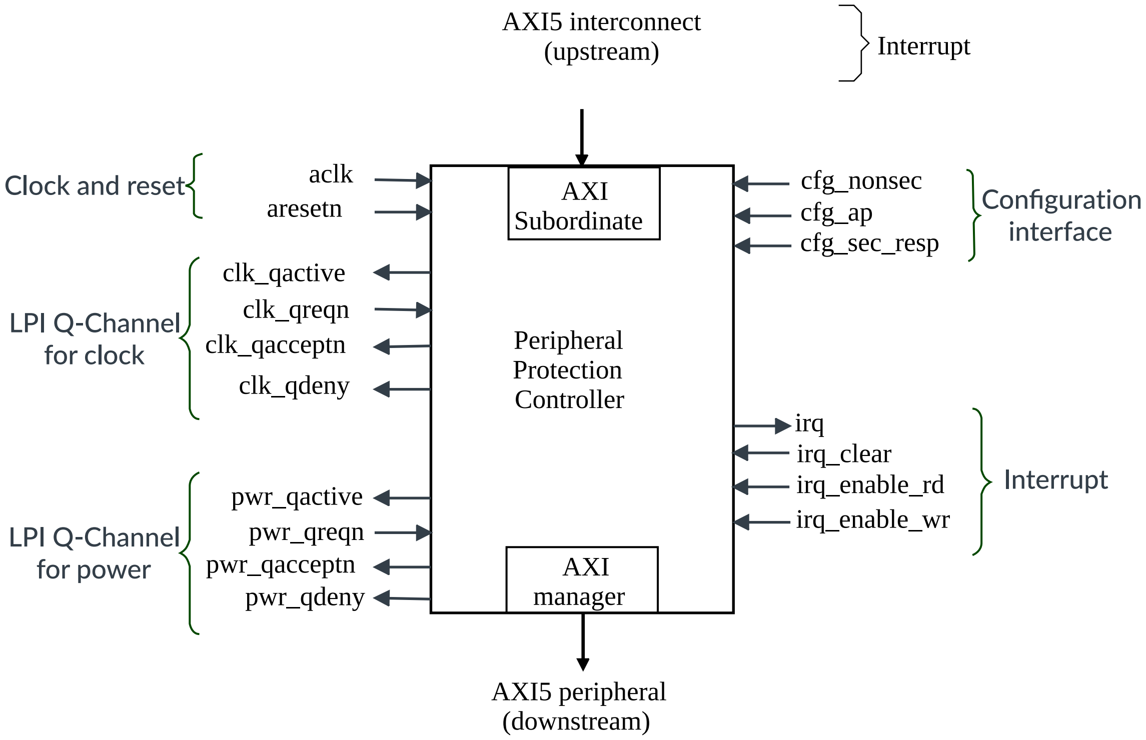
3.5.4 About the PPC

The *Peripheral Protection Controller* (PPC) provides security checks for AXI peripherals.

The PPC gates AXI transactions towards a peripheral when a security violation occurs. It can be instantiated in the system in connection to any non-security aware AXI5 peripheral. Security checking is performed against the state of the **cfg_ap** and **cfg_nonsec** signals, which indicate the privilege and Security state of the peripheral.

The following figure shows the PPC interfaces.

Figure 3-4: PPC interfaces



The AXI subordinate and AXI manager interfaces provide the AXI data path from the AXI manager to the attached peripheral.

To support low-power quiescence, the PPC has two Q-Channel interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

Configuration interface

The **cfg_nonsec** signal controls the security settings of the attached peripheral:

- If HIGH, only Non-secure accesses to the peripheral are allowed.
- If LOW, only Secure accesses to the peripheral are allowed.

The **cfg_ap** signal controls the privilege settings of the attached peripheral:

- If HIGH, only privileged accesses to the peripheral are allowed.
- If LOW, the privilege attribute is ignored for security checks.

When the PPC blocks a transaction, the **cfg_sec_resp** signal controls whether the PPC:

- Responds with an AXI subordinate error (SLVERR).
- Ignores a write transaction or returns zero for a read transaction.

3.5.5 About the SMC

The *SRAM Memory Controller* (SMC) is an AXI5 memory controller for static memory devices.

The SMC has the following features:

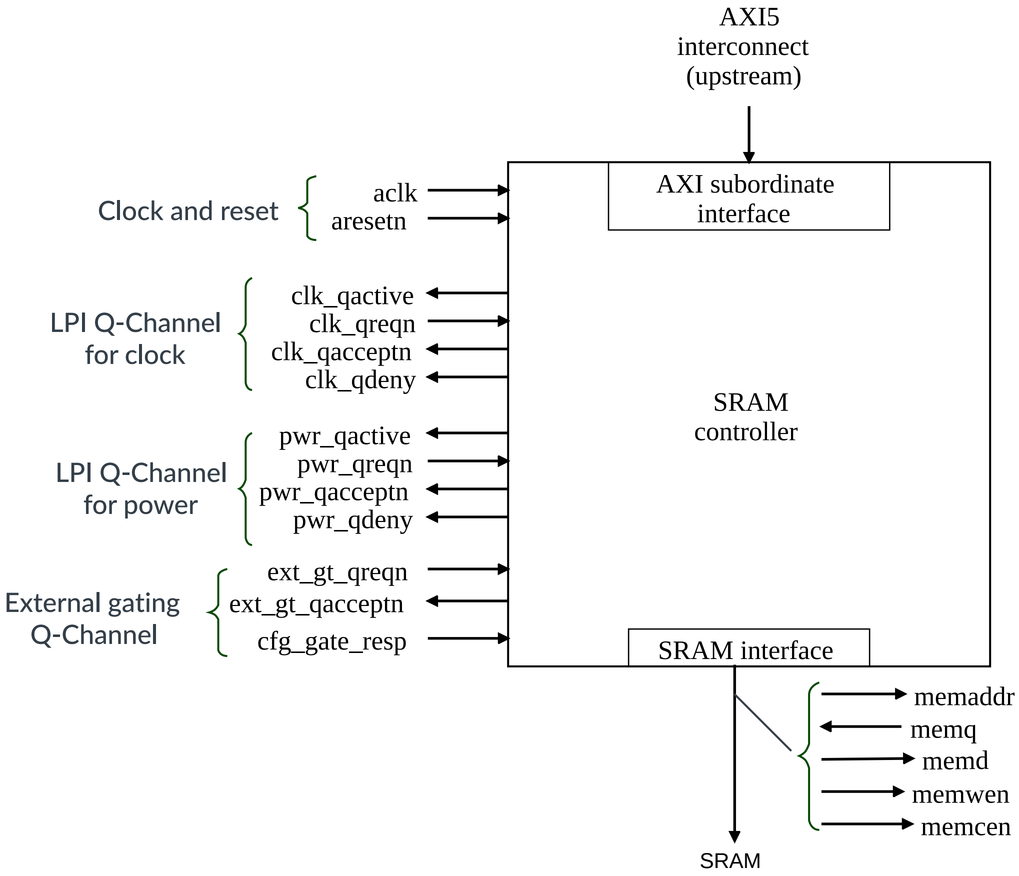
- A single clock and reset domain.
- An AXI5 subordinate interface.
- An SRAM interface.
- Two Q-Channels for clock and power control.
- No data width conversion.
- An external-gating interface that prevents the SMC from issuing new transactions on the SRAM interface.

Note

The SRAM controller primarily supports memory macros that the Arm® SRAM Compiler generates.

The following figure shows the SMC interfaces.

Figure 3-5: SMC interfaces



To support low-power quiescence, the SMC has two Q-Channel device interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

The SMC also provides a partial Q-Channel device interface to support external gating, which stops the SMC from starting any new transactions on the SRAM interface.

Early write response

The SMC buffers the write transactions and for non-exclusive writes it returns an early write response. If the write buffers become full, then the SMC does not return an early write response.

Read and write transaction scheduling

The AXI has separate buses for reads and writes, but the SRAM interface is a single bus for reads and writes. Therefore, the SMC must arbitrate between the AXI channels. The SMC performs arbitration between read and write bursts.

If the write buffers become full, then the arbitration scheme uses the QoS value of the incoming read, **arqos**, and write, **awqos** signal. For example, if the SMC receives a write with a QoS value that is higher than the read QoS, it forwards the oldest transaction from the write queue to the SRAM to allow the new write into the write buffer.

Note

Provided the write buffers (address or data) are not full, the SMC gives priority to read bursts.

To prevent system livelock or starvation issues, the SMC uses a balancing counter to track how many read bursts are granted while a write burst request is present but not granted. If this counter reaches a predefined value, then the SMC grants the write at the next arbitration point, regardless of the QoS values.

Poison

The `AXI5_POISON_EN` configuration parameter controls whether the SMC supports data poisoning.

When data poisoning is enabled, the SMC provides 1 bit of poison information for each 64 bits of data. The SMC uses the MSB of the **memwen** write enable bus on the SRAM memory side to control the writes to the storage element that holds the poison information.

Note

When narrow writes are written to the SRAM, the poison information always gets updated with the new value, regardless of the previously stored content.

The following table lists which signals contain the poison information for different `DATA_WIDTH` configurations.

Table 3-1: Poison bit locations

`DATA_WIDTH`	AXI poison bits	Poison bits on the SRAM interface
32	**wpoison[0]** and **rpoison[0]**	**memd[32]** and **memq[32]**
64	**wpoison[0]** and **rpoison[0]**	**memd[64]** and **memq[64]**
128	**wpoison[1:0]** and **rpoison[1:0]**	**memd[129:128]** and **memq[129:128]**
256	**wpoison[3:0]** and **rpoison[3:0]**	**memd[259:256]** and **memq[259:256]**

Exclusive accesses

The SMC can contain up to 16 *Exclusive Access Monitors* (EAMs), depending on the setting of the `EXCLUSIVE_MONITORS` configuration parameter.

If `EXCLUSIVE_MONITORS` > 0, then Exclusive Load transactions always return an EXOKAY response and the SMC stores the transaction details (address, ID) in an internal TAG buffer.

For Exclusive Store transactions, the SMC checks if the address and ID are present in the TAG buffer, and if so the SMC forwards the write to the SRAM and returns an EXOKAY write response. If the check fails, the SMC ignores the write data and it returns an OKAY response, which indicates an exclusive access failure.

If a non-exclusive write transaction accesses a location that is stored in a TAG buffer, then the SMC clears the TAG.

If all EAMs are occupied, and the SMC receives a new Exclusive Load transaction with an:

- ID that exists in the TAG table, then the new transaction replaces an old entry.
- ID that does not exist in the TAG table, then the SMC overwrites an entry in the TAG table that the round-robin algorithm selects, and returns an EXOKAY response.

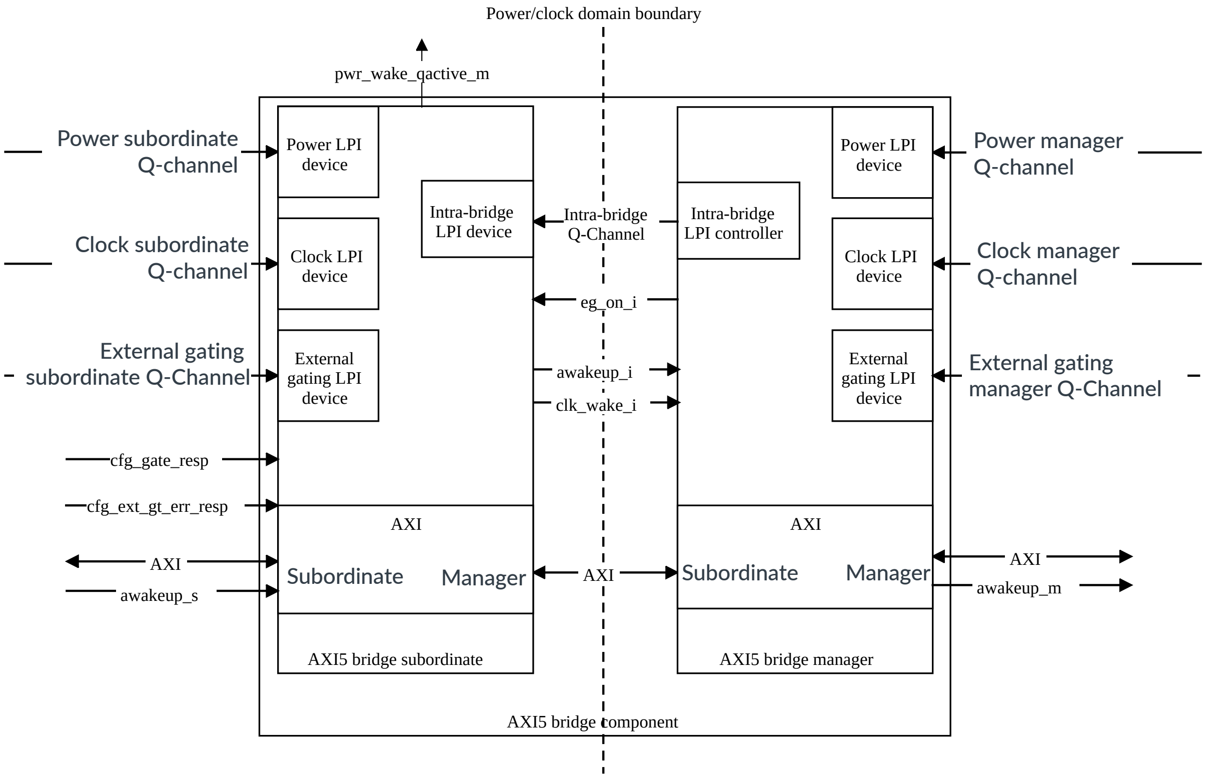
If an SMC is configured to contain no EAMs (`EXCLUSIVE_MONITORS` == 0), then exclusive writes always fail. The SMC ignores the write and returns an OKAY response.

3.5.6 About the bridge components

Bridge components provide low-power management and external gating on boundaries between clock and power domains along the AXI5 data bus. They also have configurable registering options to ease timing on long AXI5 paths.

The following figure shows the interfaces of a bridge component.

Figure 3-6: Bridge component interfaces



The following table lists the bridge components.

Table 3-2: Supported bridge components

Bridge component	Upstream to downstream clock ratio
Access Control Gate (ACG)	One-to-one
Sync-Down Bridge (SDB)	N-to-one
Sync-Up Bridge (SUB)	One-to-N

Each bridge component consists of an upstream side and a downstream side. To allow communication across clock and power domains, each side of the bridge has one intra-bridge Q-Channel interface and one intra-bridge AXI interface. The **eg_on_i** signal provides the upstream side with information about the state of external gating on the downstream side. The intra-bridge uses a standard Q-Channel LPI interface.

Each half of a bridge component supports the following features:

- A single clock and reset domain.
- An AXI5 subordinate interface.
- An AXI5 manager interface.
- Two Q-Channels for clock and power control to support low-power quiescence.
- An external-gating interface that prevents the bridge component from issuing new transactions on the AXI interface. The external-gating interface is a Q-Channel implementation without the **QDENY** and **QACTIVE** signals.

Configuration interface

The **cfg_gate_resp** controls how the upstream side of the bridge component responds, when the bridge is closed by external gating or downstream power quiescence:

- Responds with an AXI subordinate error (SLVERR).
- The bridge component sets the relevant AXI **ready** signals LOW, which stalls any AXI transactions, until the bridge is able to forward the transfers to the downstream side.

The **cfg_ext_gt_err_resp** signal controls how the bridge component responds to AXI transactions, when the upstream external gating is in quiescence. However, if the **cfg_gate_resp** is set to error, then the bridge returns an error response. Therefore, when the upstream external gating is in quiescence, the bridge:

- Responds with an AXI subordinate error (SLVERR), when **cfg_gate_resp** or **cfg_ext_gt_err_resp** are HIGH.
- Stalls the transaction until the external gating request is released, that is, **ext_gt_qreqn_s** goes HIGH. This response behavior requires that **cfg_gate_resp** and **cfg_ext_gt_err_resp** are LOW.

3.6 SIE-200 System IP for Embedded

This section gives an overview of the product and its features.

For more information, see the SIE-200 documentation set:

- *Arm® CoreLink™ SIE-200 System IP for Embedded Technical Reference Manual*
- *Arm® CoreLink™ SIE-200 System IP for Embedded Configuration and Integration Manual*

3.6.1 About SIE-200

The CoreLink™ SIE-200 System IP for Embedded product is a collection of interconnect, peripheral, and TrustZone® controller components for use with a processor that complies with the Arm®v8-M processor architecture.

Bus architecture

SIE-200 supports the following bus protocols:

- AMBA® 5 AHB5 Protocol
- AMBA® 4 APB4 Protocol
- AMBA® 3 APB3 Protocol
- AMBA® 3 AHB-Lite Protocol

Bus naming convention

It is important to always view each AMBA point-to-point connection as a manager to subordinate connection. To distinguish between external AMBA managers or subordinates and the conceptual managers or subordinates on the component, managers and subordinates on the interconnect are referred to as manager ports or subordinate ports. External managers and subordinates are referred to as managers and subordinates.

3.6.2 Features of SIE-200

SIE-200 consists of the following components and models that support the AHB5 standard:

- AHB5 system components
- AHB5 bridge components
- TrustZone® protection controllers
- Verification components

3.7 NIC-450 Network Interconnect

This section gives an overview of the product and its features.

For more information, see the NIC-450 and associated products documentation sets:

- *Arm CoreLink NIC-450 Network Interconnect Technical Overview*
- *Arm CoreLink NIC-400 Network Interconnect Technical Reference Manual*
- *Arm CoreLink NIC-400 Network Interconnect Integration Manual*

- *Arm CoreLink NIC-400 Network Interconnect Implementation Guide*
- *Arm CoreLink QoS-400 Network Interconnect Advanced Quality of Service Supplement to Arm CoreLink NIC-400 Network Interconnect Technical Reference Manual*
- *Arm CoreLink QVN-400 Network Interconnect Advanced QoS for Virtual Networks Supplement to Arm CoreLink NIC-400 Network Interconnect Technical Reference Manual*
- *Arm CoreLink TLX-400 Network Interconnect Thin Links Supplement to Arm CoreLink NIC-400 Network Interconnect Technical Reference Manual*
- *Arm CoreLink ADB-400 AMBA Domain Bridge User Guide*
- *Arm CoreLink AXI4 to AHB-Lite XHB-400 Bridge Technical Reference Manual*

3.7.1 About NIC-450

Arm® NIC-450 is a library of highly configurable and multi-power domain tools.

NIC-450 is a library of key interconnect IP that enables you to build a scalable and configurable network interconnect. NIC-450 includes:

NIC-400 Network Interconnect

NIC-400 is a cascading, routing interconnect component. NIC-400 is a hierarchical, low latency, and low-power connection for various other components.

QoS-400 Network Interconnect Advanced Quality of Service

QoS-400 provides programmable QoS facilities for any attached managers.

QVN-400 Advanced Quality of Service for Virtual Networks

QVN-400 provides a mechanism to avoid head-of-line blocking and cross-path blocking between different data flows.

TLX-400 Network Interconnect Thin Links

TLX-400 provides a mechanism to reduce the number of signals in an AXI point-to-point connection and enable it to be routed over a longer distance.

ADB-400 AMBA® Domain Bridge

ADB-400is an asynchronous bridge between two components or systems that can be in a different power, clock, or voltage domains.

AXI4 to AHB-Lite XHB-400 Bridge

XHB-400 converts AXI4 protocol to AHB-Lite protocol, and has an AXI4 subordinate interface and an AHB-Lite manager interface.

LPD-500 Low Power Distributor

LPD-500 is a standalone configurable component to distribute Q-Channel interfaces to multiple devices and subsystems.

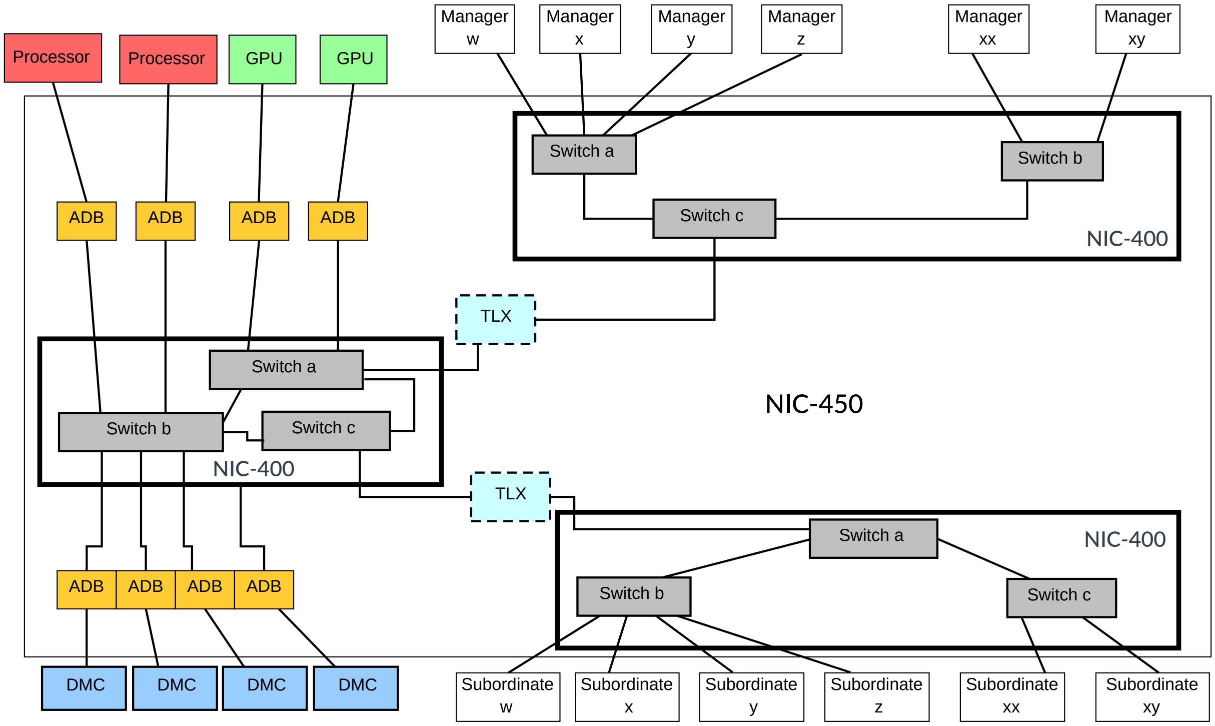
You can integrate the NIC-400 with the ADB-400 AMBA® Domain Bridge or TLX-400 Network Interconnect Thin Links bridges into a single interconnect. You can utilize the high level of configurability of NIC-450 for optimization and tuning.

The benefits of using the NIC-450 are:

- Unified low-power interfaces when applicable
- Single design environment to configure IP blocks and connect them together

Use NIC-450 with Socrates™, a tool that employs algorithms to aid the creation of valid configurations that are based on your specific design requirements.

Figure 3-7: NIC-450 block diagram



3.8 SoC-600

This section gives an overview of the product and its features.

For more information, see the SoC-600 documentation set:

- *Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual*
- *Arm® CoreSight™ System-on-Chip SoC-600 Configuration and Integration Manual*

3.8.1 About this product

CoreSight™ SoC-600 is a member of the Arm embedded debug and trace component family that is based on the Cortex-M processor.

Some of the features that CoreSight™ SoC-600 provides are:

- Components that can be used for debug and trace of Arm SoCs. These SoCs can be simple single-processor designs to complex multiprocessor and multi-cluster designs that include many heterogeneous processors.
- Support for the *Arm® Debug Interface* (ADI) v6 and CoreSight™ v3 Architectures that enable you to build debug and trace functionality into your systems. It supports debug and trace over existing functional interfaces.
- Components that support the development of low-power system implementations through architected fine-grained power control.
- Q-Channel interfaces for clock and power quiescence.
- Can be integrated with the Arm® CoreLink™ LPD-500 as part of a full-chip power and clock control methodology.
- The Arm® CoreSight™ SDC-600 can be integrated with CoreSight™ SoC-600, with an applicable licence, as part of a certificate-based authenticated debug solution.

The CoreSight™ SoC-600 bundle includes:

- A library of configurable CoreSight™ components that are written in Verilog, and that are compliant with the *Verilog-2001 Standard* (IEEE Std 1364-2001).
- Example timing constraint files for each component in SDC format.

Note

- CoreSight™ was previously configurable in Socrates™ System Builder.
- Socrates™ System Builder is now in maintenance. There will be no functionality updates to Socrates™ System Builder, although there might be maintenance updates.
- The new IP Tooling platform, Socrates™, enables configuration and build of individual CoreSight™ components. However, there is no CoreSight™ creation flow, and no system-stitching capability.
- Socrates™ functionality can be enabled using a legacy Socrates™ System Builder license.

3.8.2 Features

Features and capabilities that the SoC-600 provides include:

Debug

- *Arm® Debug Interface Architecture Specification ADIv6.0*-compliant debug port. This debug port supports JTAG and Serial Wire protocols for connection to an off-chip debugger.

This connection is achieved using a low-pin-count connection that is suitable for bare-metal debug and silicon bring-up.

- *Arm® CoreSight™ Architecture Specification v3.0* compliance enables debug over functional interfaces, suitable for application development and in-field debug without a dedicated debug interface.
- Infrastructure components supporting system identification and integration with other CoreSight™ IP.

Trace

- Versatile *Trace Memory Controller* (TMC) supporting local on-chip storage, and buffering of trace data.
- TMC router configuration supports efficient hand-off of trace data to other system managers. This feature enables trace over functional interfaces, suitable for application development and in-field debug without a dedicated debug and trace interface.
- TMC streaming configuration supports integration to third-party *High Speed Serial Trace Ports* (HSSTPs) for high bandwidth, low pin count trace solutions.
- Infrastructure components supporting filtering and routing of trace data on chip.

Embedded Cross Triggering

- *Cross Trigger Interface* (CTI) supports up to 32 trigger inputs and outputs with a single component instance.
- *Cross Trigger Matrix* (CTM) supports up to 33 CTI or CTM connections without cascading.

Power

- *Arm® CoreSight™ Architecture Specification v3.0*-compliant *Granular Power Requester* (GPR) enables fine-grained debug and system power control at all levels of debug hierarchy.
- Components are designed for low-power implementation, supporting clock and power quiescence and wakeup signaling where necessary.
- Components support Q-Channel *Low-Power Interfaces* (LPI) for integration with power controllers to support system-level clock and power gating where necessary.
- Infrastructure components support implementation across multiple clock and power domains.

Miscellaneous

- Some components, such as the bridges and *Serial Wire Debug Port* (SW-DP), use two Verilog modules to span clock and power domains. This design can ease implementation in complex SoC designs that have multiple clock and power domains.
- Infrastructure components support integration with legacy IP including *Arm® CoreSight™ Architecture Specification v2.0*-compliant, and JTAG components.

3.9 SDC-600 Secure Debug Channel

This section gives an overview of the product and its features.

For more information, see the SDC-600 documentation set:

- *Arm® CoreSight™ SDC-600 Secure Debug Channel Technical Reference Manual*
- *Arm® CoreSight™ SDC-600 Secure Debug Channel Configuration and Integration Manual*

3.9.1 About SDC-600

Arm® CoreSight™ SDC-600 provides a dedicated channel for authentication between an external debugger and a debug target platform by using an unlocking mechanism.

The SDC-600-based architecture provides an interface through which secure debug certificates can be injected to the platform. This is done in a standard way through the *Debug Access Port* (DAP), which is normally used to debug the platform. It eliminates the need for OEM proprietary delivery mechanisms for such certificates.

SDC-600 performs the following tasks:

- Requests power and optionally reboots the servicing agent.
- Establishes and maintains a link between a port on the external side, which is serviced by the debugger, and a port on the internal side, which is serviced by an agent on the target system.
- Transports messages from an external debugger to a hardware or software agent on a target system through a point-to-point link.

The debugged target and the servicing agent are typically the same processor or processor subsystem, but they can be separate entities as well.

The authentication process can involve a hardware- or software-based cryptographic engine on the target. The cryptographic engine verifies the debug certificate that is passed to the servicing agent through the SDC-600. The debugger and the servicing agent run a protocol on top of the SDC-600, which:

1. Identifies the SoC (SoC_ID).
2. Injects the appropriate debug certificate to the debug target for processing by the cryptographic engine.

The following is a high-level description of a sample authentication process:

1. The debugger wants to access the target's debug resources.
2. The debugger uses the CoreSight™ ID registers and discovery process to identify the SDC-600's external interface.
3. The debugger accesses the SDC-600 to start the unlocking process.
4. The SDC-600 requests the powerup of the rest of its functional blocks.
5. The debugger asks for a SoC_ID from the servicing target to identify the target system.

6. A certificate is generated by the debugger for the SoC_ID that is transmitted to the servicing target.
7. The servicing agent decides whether the debugger has the rights to access the debug target based on the provided certificate.
8. If access is granted, the target agent drives the authentication signals accordingly on the Access Ports so that the connected devices can be accessed by the debugger.

3.10 STM-500 System Trace Macrocell

This section gives an overview of the product and its features.

For more information, see the STM-500 documentation set:

- *Arm® CoreSight™ STM-500 System Trace Macrocell Technical Reference Manual*

3.10.1 About STM-500

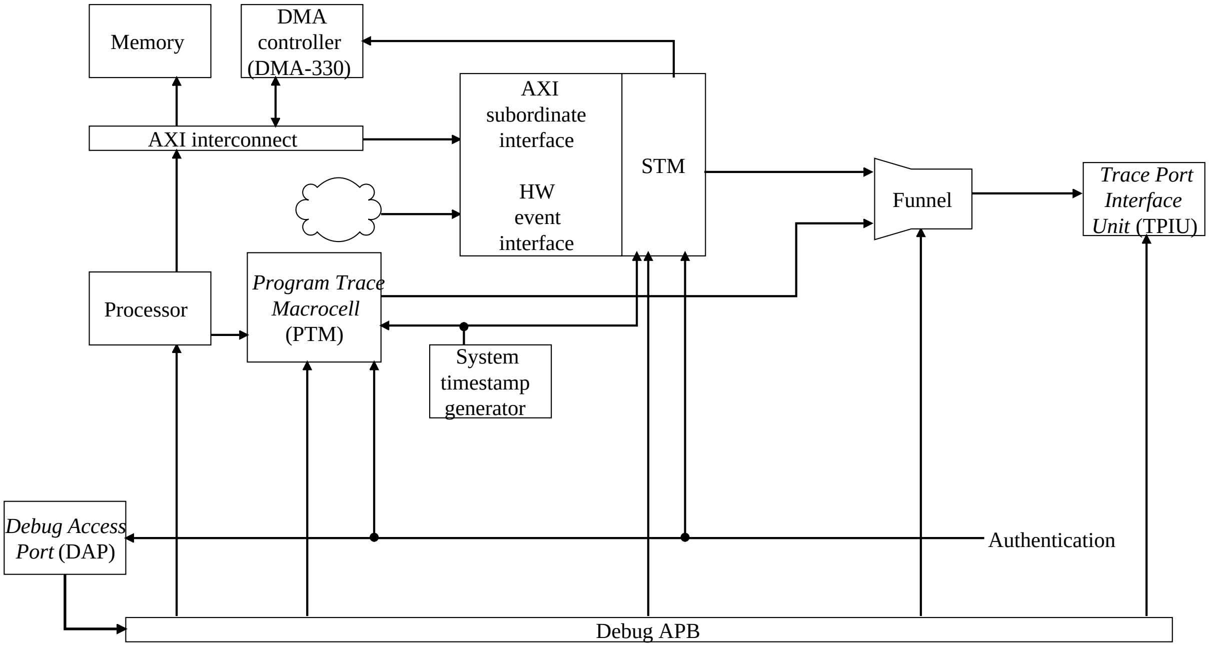
STM-500 is a trace source that is integrated into a CoreSight system, and is designed primarily for high-bandwidth trace of instrumentation embedded into software. This instrumentation is made up of memory-mapped writes to the STM Advanced eXtensible Interface (AXI) subordinate, which carry information about the behavior of the software.

STM-500 is a natural successor to the CoreSight *Instrumentation Trace Macrocell* (ITM) in mid-to high-performance applications. The STM provides the following advantages over the ITM for software instrumentation:

- It has a dedicated AXI subordinate interface for receiving the instrumentation information. The AXI subordinate interface is in addition to the *Advanced Peripheral Bus* (APB) interface that you can use for programming the STM-500 registers. The AXI subordinate interface has significantly higher performance than the APB interface of the ITM.
- Multiple processors and processes can share and directly access STM-500 without being aware of each other, by being allocated different pages in the STM stimulus space. 128 managers, each supporting 65,536 stimulus ports. Each 4KB page of the STM stimulus space provides 16 stimulus ports. Stimulus ports are also known as channels.
- STM-500 can optionally stall the AXI when its FIFO becomes full, ensuring that no data is lost because of overflow, without having to poll the FIFO status in software. This behavior depends on the address written to, and can therefore be controlled by each stimulus port independently.
- An improved, configurable FIFO, supporting up to 32 transactions, reduces the likelihood of the FIFO becoming full.
- Timestamping can be requested for each write independently, based on the address written to. You can also optimize the bandwidth by requesting a timestamp for only one write transaction in a message made up of several writes.
- Timestamps are automatically correlated with other timestamping trace sources in the CoreSight system, enabling automatic correlation with, for example, processor execution trace.

In addition to the AXI subordinate, STM-500 provides a hardware event interface. STM-500 generates trace when signals are asserted on this interface. Alternatively, you can implement advanced custom system tracing features by generating AXI write accesses directly to the AXI subordinate.

Figure 3-8: System integration



STM-500 AXI subordinate is connected to a system interconnect that enables all system managers, such as processors and *Direct Memory Access* (DMA) controllers, to generate trace by writing to the STM-500 stimulus ports.

For interaction with DMA controllers, STM-500 provides a DMA request interface that is compatible with the AMBA DMA Controller DMA-330.

For configuration purposes, STM-500 is connected to a Debug APB interconnect. This enables off-chip and on-chip debug agents to access STM-500.

STM-500 uses CoreSight authentication signals to control debug permissions.

The STM-500 trace stream is output through the *Advanced Trace Bus* (ATB) interface and it is integrated with the rest of the CoreSight trace infrastructure.

3.10.2 Features of STM-500

STM-500 has the following features:

- A fully synchronous design with one clock and two resets
  One 64-bit AXI subordinate interface for extended stimulus port inputs

- One hardware event observation interface for tracing 64 hardware events
- One 32-bit debug APB completer interface for configuration and status
- One 64-bit ATB manager interface for trace output
- One DMA peripheral request interface that is compatible with the AMBA DMA Controller DMA-330
- Two depth-configurable FIFO buffers for usage-optimized configurability:
  - Data FIFO
  - Channel information FIFO
- A fully memory-mapped software stimulus supporting 65,536 stimulus ports and 128 managers
- Leading zero data compression
- Full support for guaranteed and invariant timing software stimulus writes
- Support for single-shot and multi-shot triggers with a cross-trigger port, trigger packet insertion, and ATB trace triggers
- An internal and an external source for STPv2 synchronization
- Timestamping of trace events
- Two low-power interfaces

3.11 PCK-600 Power Control Kit

This section gives an overview of the product and its features.

For more information, see the PCK-600 documentation set:

- *Arm® CoreLink™ PCK-600 Power Control Kit Technical Reference Manual*
- *Arm® CoreLink™ PCK-600 Power Control Kit Configuration and Integration Manual*

3.11.1 About the Power Control Kit

The PCK-600 provides a set of configurable RTL components for the creation of SoC clock and power control infrastructure. The components use the Arm Q-Channel and P-Channel low power interfaces.

The PCK-600 consists of the following components:

Low Power Distributor Q-Channel (LPD-Q)

The LPD-Q component distributes a Q-Channel from one Q-Channel controller to up to 32 Q-Channel devices.

Low Power Distributor P-Channel (LPD-P)

The LPD-P component distributes a P-Channel from one P-Channel controller to up to 8 P-Channel devices.

Low Power Combiner Q-Channel (LPC-Q)

The LPC-Q component combines the Q-Channels from multiple Q-Channel controllers to multiple Q-Channel devices with common control requirements.

P-Channel to Q-Channel Converter (P2Q)

The P2Q component converts a P-Channel to a Q-Channel.

Clock Controller (CLK-CTRL)

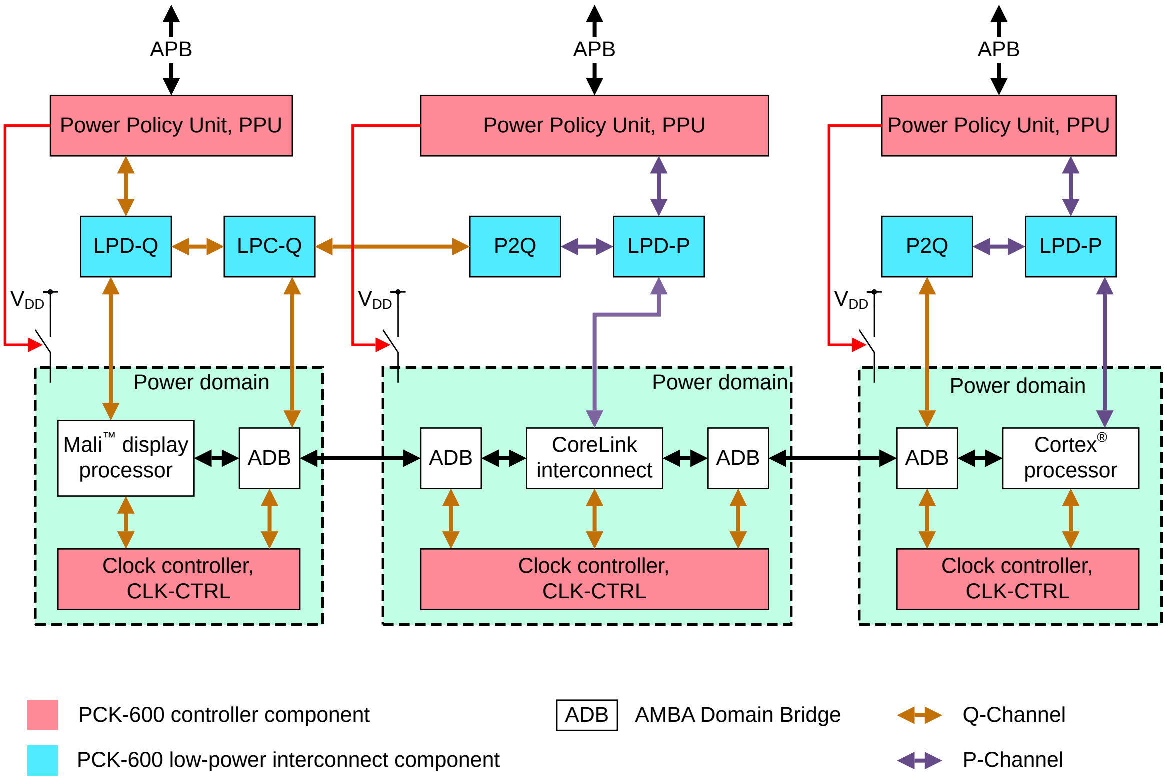
The CLK-CTRL component provides *High-level Clock Gating* (HCG) for a single clock domain.

Power Policy Unit (PPU)

The PPU component is a configurable and programmable P-Channel and Q-Channel power domain controller.

The following figure shows an example system that uses the components to manage three power domains. The components are shown in red and blue.

Figure 3-9: Example system that contains PCK-600



3.12 GFC-200 Generic Flash Controller

This section gives an overview of the product and its features.

For more information, see the GFC-200 documentation set:

- *Arm® CoreLink™ GFC-200 Generic Flash Controller Technical Reference Manual*
- *Arm® CoreLink™ GFC-200 Generic Flash Controller Configuration and Integration Manual*

3.12.1 About the GFC-200

The GFC-200 comprises the generic part of a Flash controller in a *System-on-Chip* (SoC). The GFC-200 enables an embedded Flash macro to be integrated easily into any system.

An eFlash macro enables a Flash controller to access eFlash memory. The eFlash macros produced by different foundries and processes can have different interfaces, timings, signal names, protocols, and features that are determined by the foundry processes that produced the eFlash memory.

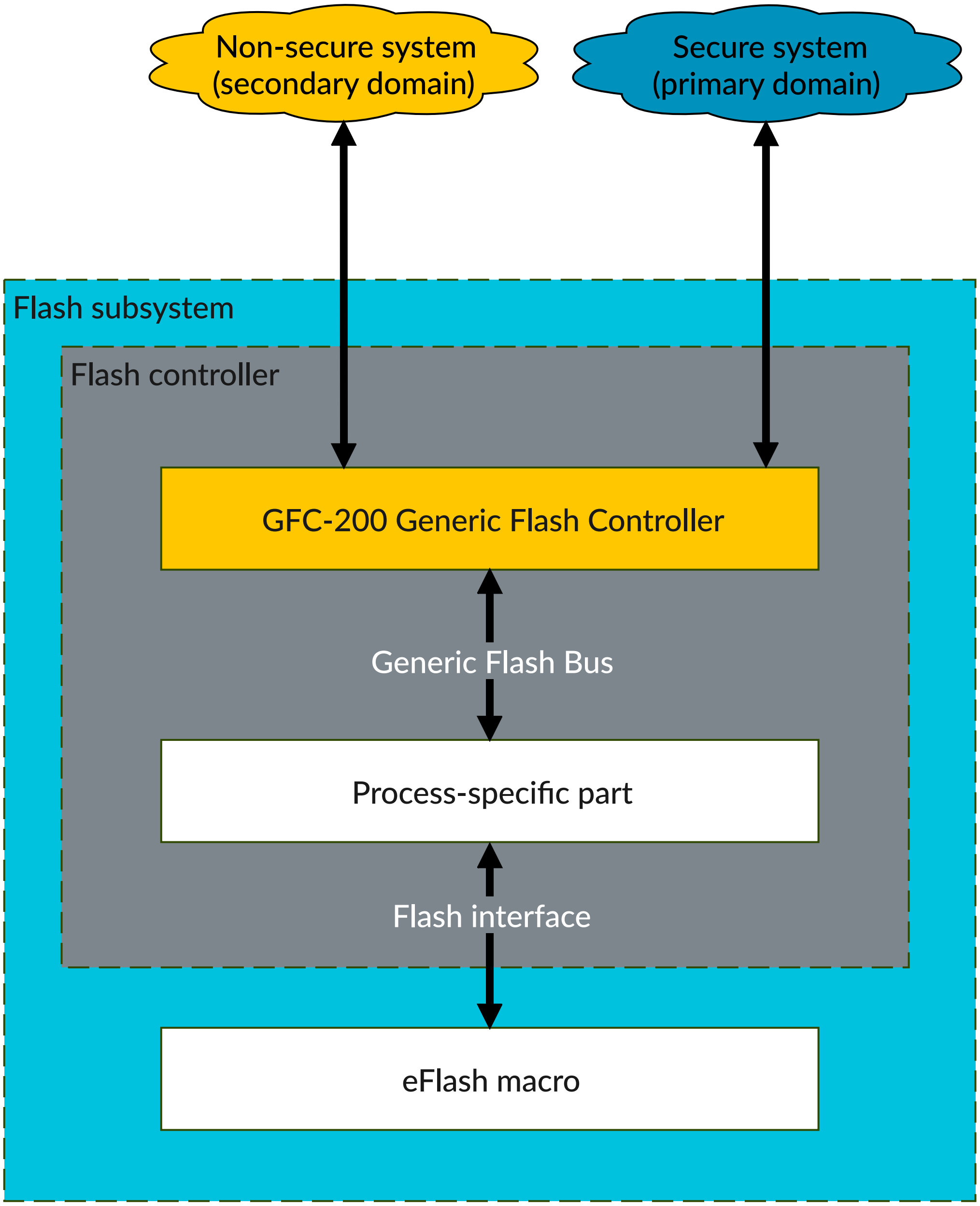
The GFC-200 provides functions that relate only to services for the system side of the Flash controller. The GFC-200 cannot communicate directly with the eFlash macro. Therefore, the GFC-200 must be integrated with a process-specific part that connects to, and communicates with, the eFlash macro.

The process-specific part of the Flash controller is part of the Flash subsystem in your SoC. It communicates directly with the eFlash macro through a Flash interface.

The GFC-200 supports accesses from two managers that can operate in separate domains such as a Non-secure domain and a Secure domain. Communication between the system and eFlash memory is through a *Generic Flash Bus* (GFB) supplied with GFC-200.

The following figure shows how the GFC-200 is used in a Flash controller implementation.

Figure 3-10: GFC-200 in a Flash controller implementation


Non-secure system (secondary domain)
Secure system (primary domain)
Flash subsystem
Flash controller
GFC-200 Generic Flash Controller
Generic Flash Bus
Process-specific part
Flash interface
eFlash macro

3.12.2 Features

The GFC-200 provides several interfaces and features.

Flash memory partitioning:

- Ability to divide the available Flash memory space into several partitions and perform access control on a per partition basis

- Dynamically configurable access rights to partitions
- A configuration parameter controls the size of the partitions

AMBA® AHB-Lite interface:

- Read-only access to the embedded Flash
- Configurable data width
- Burst support
- Low latency

Primary APB completer interface:

- Write and erase access to the embedded Flash
- Debug read access to the embedded Flash
- Control port for GFC-200 and the eFlash macro
- Interrupt capability for long running commands
- Access to internal registers and the control registers in the process-specific part

Secondary APB completer interface:

- Write and erase access to the embedded Flash
- Debug read access to the embedded Flash
- Control port for GFC-200
- Interrupt capability for long running commands
- Access to internal registers

APB register requester interface:

- Enables access to the registers in the process-specific part

Q-Channel interface:

- Control port for system power
- Control port for the system clock

P-Channel controller interface:

- Control port for power to the process-specific part

Generic Flash Bus (GFB):

- Enables GFC-200 accesses to embedded Flash
- Simple command-based protocol
- Synchronous with the AHB clock
- Simplifies communication between GFC-200 and the attached process-specific part

3.13 GFC-100 Generic Flash Controller

This section gives an overview of the product and its features.

For more information, see the GFC-100 documentation set:

- *Arm® CoreLink™ GFC-100 Generic Flash Controller Technical Reference Manual*
- *Arm® CoreLink™ GFC-100 Generic Flash Controller Configuration and Integration Manual*

3.13.1 About GFC-100

The GFC-100 comprises the generic part of a Flash controller in a *System-on-Chip* (SoC). GFC-100 enables an embedded Flash macro to be integrated easily into any system.

An eFlash macro enables a Flash controller to access eFlash memory. The eFlash macros produced by different foundries and processes can have different interfaces, timings, signal names, protocols and features that are determined by the foundry processes that produced the eFlash memory.

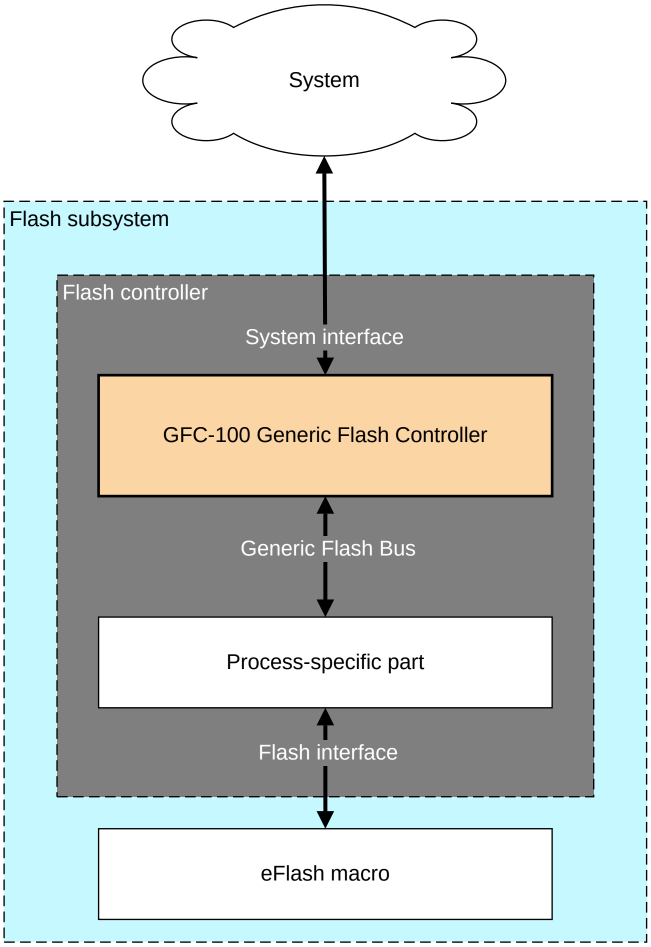
GFC-100 provides the functions that relate only to services for the system side of the Flash controller. GFC-100 cannot communicate directly with the eFlash macro. Therefore, GFC-100 must be integrated with a process-specific part that connects to, and communicates with, the eFlash macro.

The process-specific part of the Flash controller is part of the Flash subsystem in your SoC. It communicates directly with the eFlash macro through a Flash interface.

Communication between the system and eFlash memory is through a *Generic Flash Bus* (GFB) supplied with GFC-100.

The following figure shows how GFC-100 is used in a Flash controller implementation.

Figure 3-11: GFC-100 in a Flash controller implementation



3.13.2 Features

GFC-100 provides several interfaces and test features.

Advanced High-performance Bus (AHB-Lite) interface:

- Read access to the main and extended areas of embedded Flash
- Burst support
- Low latency

Advanced Peripheral Bus (APB) completer interface:

- Write and erase access to the main and extended areas of embedded Flash
- Debug read access to the main and extended areas of embedded Flash
- Control port for GFC-100 and the eFlash macro
- Interrupt capability for long running commands

- Access to internal and external registers

APB register requester interface:

- Control port for attached process-specific registers

Q-Channel interface:

- Control port for system power
- Control port for the system clock

P-Channel controller interface:

- Control port for power to the attached process-specific part

Generic Flash Bus (GFB):

- Enables GFC-100 accesses to embedded Flash
- Simple command-based protocol
- Synchronous with the AHB clock
- Simplifies communication between GFC-100 and the attached process-specific part

3.14 Corelink AHB Flash Cache

This section gives an overview of the product and its features.

For more information, see the AHB Flash Cache documentation set:

- *Arm® CoreLink™ CG092 AHB Flash Cache Technical Reference Manual*
- *Arm® CoreLink™ CG092 AHB Flash Cache Configuration and Integration Manual*

3.14.1 About AHB Flash Cache

The AHB Flash Cache is an instruction cache that is instantiated between the bus interconnect and the eFlash controller.

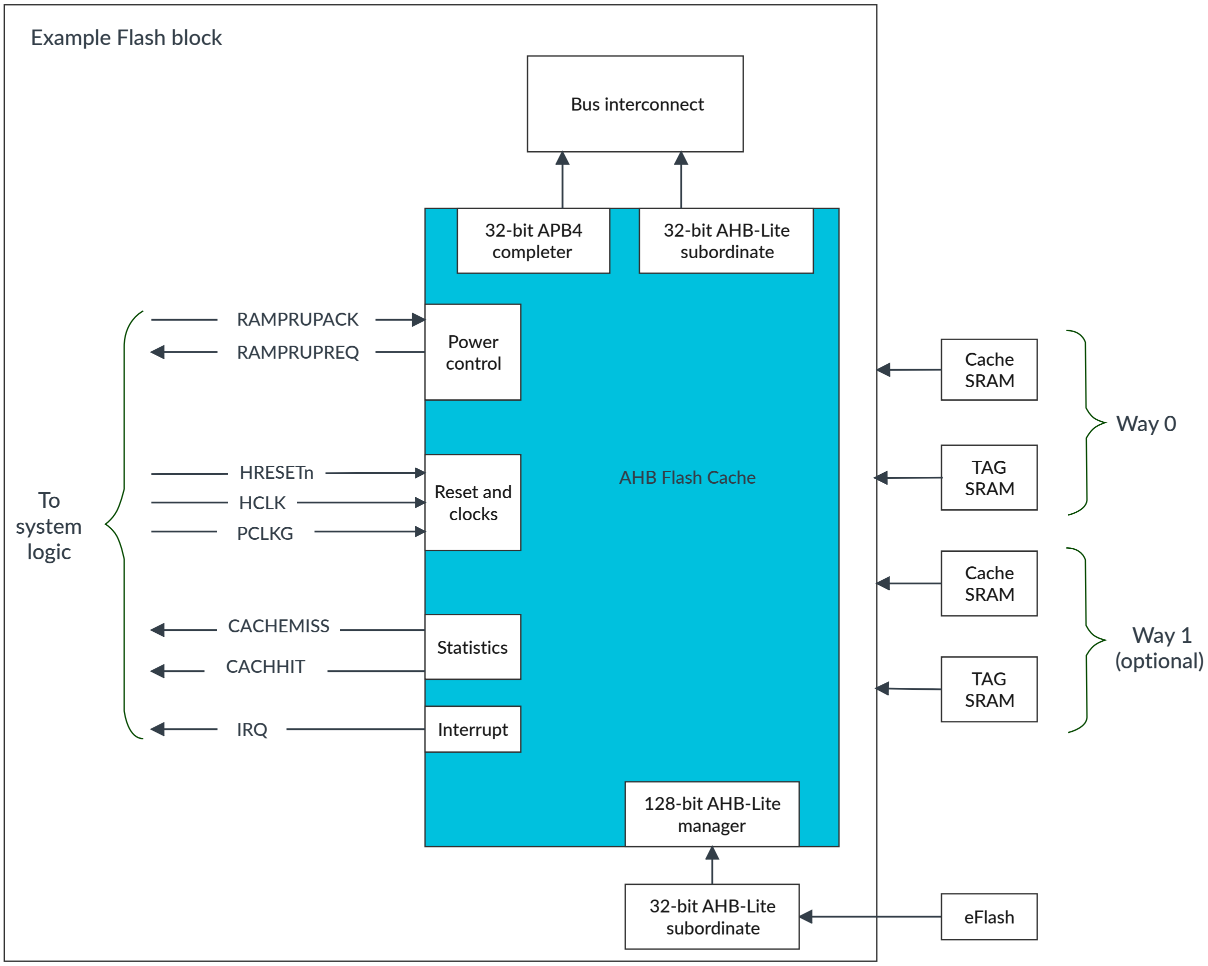
The AHB Flash Cache is a simple cache for on-chip *embedded Flash* (eFlash). The CG092 design is optimized for fetching Cortex®-M3 or Cortex®-M4 instructions directly from an eFlash. The main benefit of the AHB Flash Cache is improved power efficiency, but there are also improvements in code fetching performance.

Note

The AHB Flash Cache can also be used with external eFlash if the Flash controller is modified accordingly.

The following figure shows the connections in a typical Flash subsystem.

Figure 3-12: Example eFlash implementation



3.14.2 Features of AHB Flash Cache

The AHB Flash Cach is an instruction cache designed to be instantiated between the bus interconnect and the eFlash controller.

The CG092 has the following features:

- Configurable cache size (minimum 256 bytes/way).
- Four words per cacheline.
- Supports 2-way set associative cache, or 1-way fully associative cache.
- Configurable address bus size (based on flash memory size) so that tag memory size can be minimized.
- SRAM power-control handshaking to an external power management unit.

- Supports automatic and manual SRAM power up and power down (with simple handshaking). If valid data is in the powered-down cache because the cache is in a low-power state, the cache contents should not be invalidated on wake up. The software can therefore save energy by avoiding invaliding the cache RAMs on wake up.
- Supports automatic or manual cache invalidate in the enabling sequence. This behavior can be overridden.
- 32 bit AHB subordinate interface to the AHB manager in the system processor.
- 32 bit APB completer interface to the memory-mapped registers of the CG092.
- 128-bit AHB manager interface to the eFlash.
- Interrupt request generated on SRAM power or manual invalidation errors.
- Optional run-time support for prefetch to improve performance when executing a sequence of code that has not been read before.
  The prefetching performance impact is application dependent and might have a negative impact on eFlash power consumption.
- Optional compile-time support configurable performance counters that measure cache hits and misses.
  Exported cache hit and cache miss status signals can be used by performance measurement logic implemented at SoC level.

Note

An eFlash controller is not part of the CG092 component.

3.15 AXI Internal Memory Interface (BP140)

This section gives an overview of the product and its features.

For more information, see the AXI Internal Memory Interface (BP140) documentation set:

- *Arm PrimeCell Infrastructure AMBA 3 AXI Internal Memory Interface (BP140) Technical Overview*
- *Arm PrimeCell Infrastructure AMBA 3 AXI Internal Memory Interface (BP140) Design Manual*

3.15.1 About the internal memory interface

You can use the `IntMemAxi` component to interface to a synthesized SRAM. It can also be connected to ROM.

You can use the internal memory behavioral component, `IntMemBhavAxi`, as a memory subordinate for behavioral testbenches.

3.15.2 Features of the internal memory interface

The AXI internal memory interface, `IntMemAxi`, has the following features:

- It provides a single-port memory interface configurable for synchronous SRAM or ROM
- The HDL code is supplied as Verilog
- The memory footprint is that of the connected SRAM or ROM
- It accepts a single address for each of the read and write channels
- It uses round-robin arbitration between read and write transactions, the default is read
- For write transactions, the first data transfer can take place 2 cycles after the address is accepted. Subsequent data transfers can complete in consecutive cycles.
- For read transactions:
  - With one wait state: The first data transfer can take place 2 cycles after the address is accepted. Subsequent data transfers can complete in consecutive cycles.
  - With zero wait states: The first data transfer can take place 3 cycles after the address is accepted. Subsequent data transfers take 2 cycles.
- It supports:
  - All AMBA AXI channels except the low power channel
  - All AXI burst types
  - Aligned and unaligned transfer types
- You can configure the following parameters:
  - Data width of 64 bits or 32 bits
  - ID width
  - Wait states, single or none, for SRAM and ROM reads
  - Whether the interface is to be used with SRAM or ROM
  - Memory read access to be zero or one wait-state
  - `MEM_ADDR_WIDTH`, `MEM_INIT_FILE_0`, and `MEM_INIT_FILE_1` for `IntMemBhavAxi`

3.16 XHB-500 bridge

This section gives an overview of the product and its features.

For more information, see the XHB-500 Bridge documentation set:

- *Arm CoreLink XHB-500 Technical Reference Manual AXI5 to AHB5 bridge and AHB5 to AXI5 bridge*
- *Arm CoreLink XHB-500 Configuration and Integration Manual AXI5 to AHB5 bridge and AHB5 to AXI5 bridge*

3.16.1 About the XHB-500 bridges

The product provides an AMBA® AXI5 to AHB5 bridge and an AHB5 to AXI5 bridge.

The AXI5 to AHB5 bridge translates AXI5 transactions into the corresponding AHB transfers. The bridge has an AXI5 subordinate interface and an AHB5 manager interface.

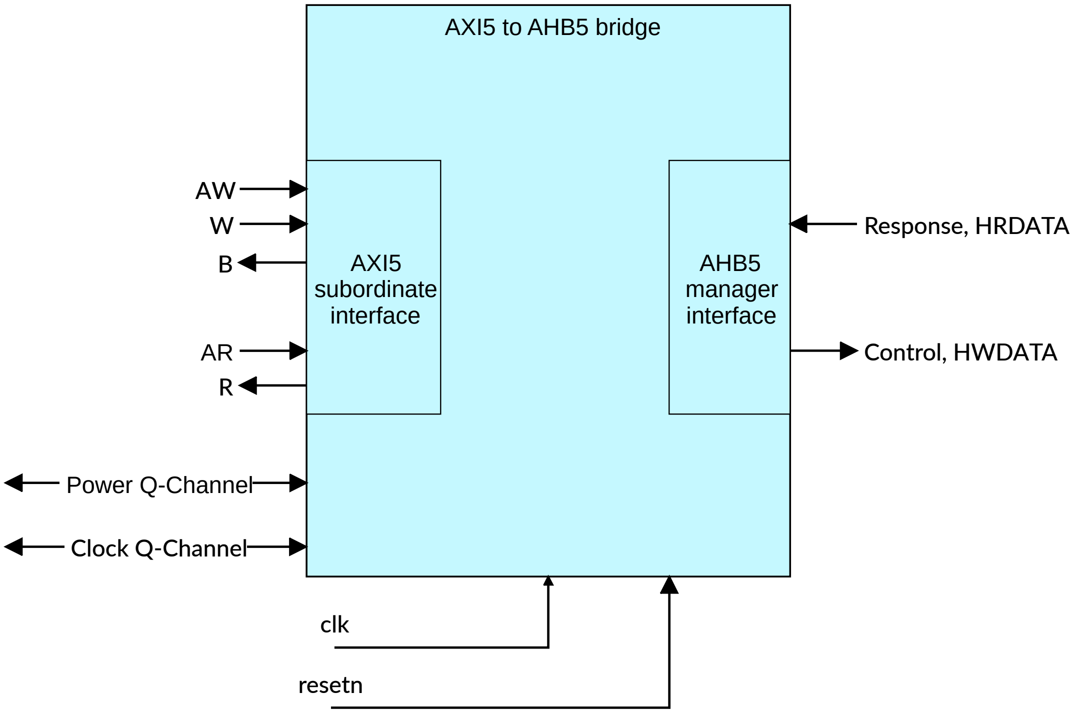
The AHB5 to AXI5 translates AHB5 transfers into the corresponding AXI transactions. The bridge has an AHB5 subordinate interface and an AXI5 manager interface.

AXI5 to AHB5 overview

The AHB5 is a low-latency bridge that performs no transaction buffering.

The following figure shows the interfaces of the AHB5.

Figure 3-13: AHB5 interfaces



The main features are:

- Single power domain
- Single clock domain
- Configurable data width
- AXI5 subordinate interface features:
  - AXI5 protocol support

  - AXI4 protocol support
  - Fixed address width
  - Registered or unregistered interface
  - Single Exclusive accesses. Exclusive bursts are not supported
  - Unaligned accesses
  - Conversion of sparse write transactions, when the `HWSTRB_ENABLE` configuration parameter is set to OFF
  - Supports all burst types
- AHB5 manager interface features:
  - AHB5 support
  - AHB-Lite support, which requires several signals to be tied off
  - Fixed address width
  - Registered or unregistered interface
  - Exclusive accesses. For AHB-Lite, extra glue logic is required.
  - Write strobe support using the **hwstrb** signal, when the `HWSTRB_ENABLE` configuration parameter is set to ON. The **hwstrb** signal is not present in the *Arm® AMBA® 5 AHB Protocol Specification*.
- Q-Channel interface for clock control
- Q-Channel interface for power control

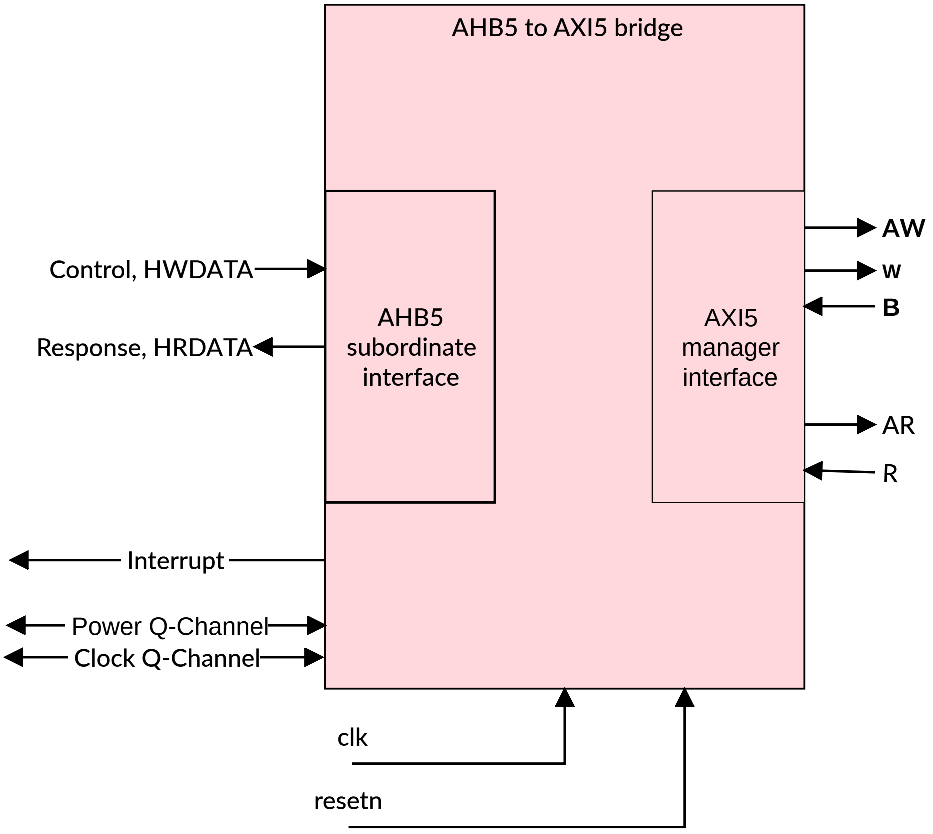
The bridge does not support endian conversion.

AXI5 overview

The AXI5 to AHB5 bridge is a low-latency bridge that performs no transaction buffering.

The following figure shows the interfaces of the AXI5 to AHB5 bridge.

Figure 3-14: AXI5 to AHB5 interfaces



The main features are:

- Single power domain
- Single clock domain
- Configurable data width
- AXI5 subordinate interface features:
    - AXI5 protocol support
    - AXI4 protocol support
    - Fixed address width
    - Registered or unregistered interface
    - Single Exclusive accesses. Exclusive bursts are not supported.
    - Unaligned accesses
    - Conversion of sparse write transactions, when the `HWSTRB_ENABLE` configuration parameter is set to OFF
    - Supports all burst types
- AHB5 manager interface features:

- AHB5 support
- AHB-Lite support, which requires several signals to be tied off
- Fixed address width
- Registered or unregistered interface
- Exclusive accesses. For AHB-Lite, extra glue logic is required.
- Write strobe support using the **hwstrb** signal, when the `HWSTRB_ENABLE` configuration parameter is set to ON. The **hwstrb** signal is not present in the *Arm® AMBA® 5 AHB Protocol Specification*.

- Q-Channel interface for clock control
- Q-Channel interface for power control

The bridge does not support endian conversion.

3.17 UART

This section gives an overview of the product and its features.

For more information, see the UART (PL011) documentation set:

- *PrimeCell UART (PL011) Technical Reference Manual*

3.17.1 About UART

The UART is an Advanced Microcontroller Bus Architecture (AMBA) compliant System-on-Chip (SoC) peripheral that is developed, tested, and licensed by Arm®.

The UART is an AMBA subordinate module that connects to the *Advanced Peripheral Bus* (APB). The UART includes an *Infrared Data Association* (IrDA) *Serial InfraRed* (SIR) protocol *ENcoder/DECoder* (ENDEC).

3.17.2 Features of UART

The UART supports the following features:

- Compliance to the *AMBA Specification* (Rev 2.0) onwards for easy integration into SoC implementation
- Programmable use of UART or IrDA SIR input/output
- Separate 32×8 transmit and 32×12 receive First-In, First-Out (FIFO) memory buffers to reduce CPU interrupts
- Programmable FIFO disabling for 1-byte depth

- Programmable baud rate generator. This enables division of the reference clock by (1×16) to (65535×16) and generates an internal ×16 clock. The divisor can be a fractional number enabling you to use any clock with a frequency >3.6864MHz as the reference clock.
- Standard asynchronous communication bits (start, stop and parity). These are added prior to transmission and removed on reception.
- Independent masking of transmit FIFO, receive FIFO, receive timeout, modem status, and error condition interrupts
- Support for Direct Memory Access (DMA)
- False start bit detection
- Line break generation and detection
- Support of the modem control functions CTS, DCD, DSR, RTS, DTR, and RI
- Programmable hardware flow control
- Fully-programmable serial interface characteristics:
    - data can be 5, 6, 7, or 8 bits
    - even, odd, stick, or no-parity bit generation and detection
    - 1 or 2 stop bit generation
    - baud rate generation, dc up to **UARTCLK**/16
- IrDA SIR ENDEC block providing:
    - programmable use of IrDA SIR or UART input/output
    - support of IrDA SIR ENDEC functions for data rates up to 115200 bps half-duplex
    - support of normal 3/16 and low-power (1.41 - 2.23µs) bit durations
    - programmable division of the **UARTCLK** reference clock to generate the appropriate bit duration for low-power IrDA mode.
- Identification registers that uniquely identify the UART. These can be used by an operating system to automatically configure itself.

3.18 Real Time Clock

This section gives an overview of the product and its features.

For more information, see the RTC documentation set:

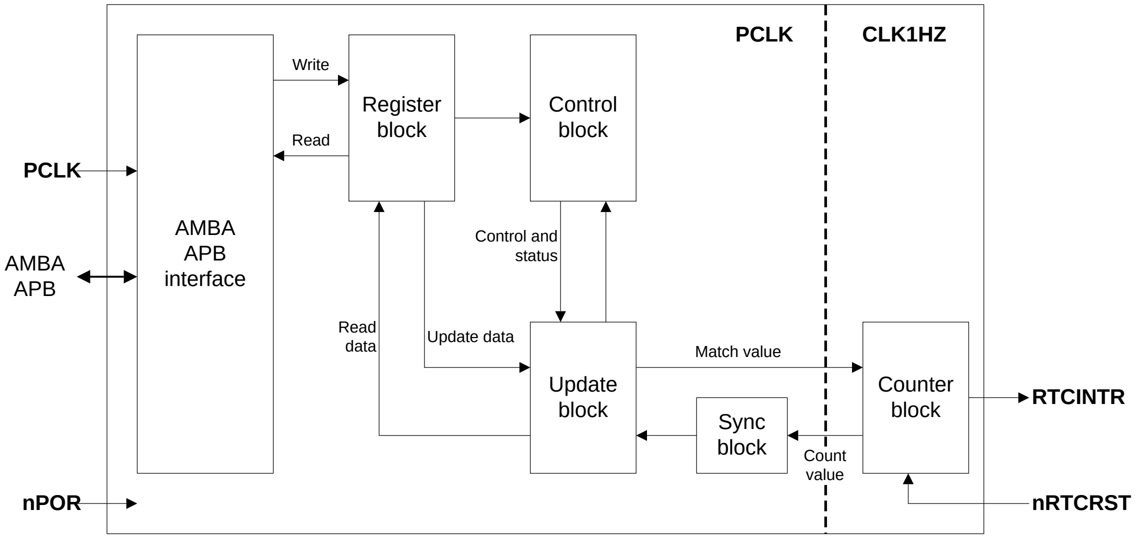
- *Arm® PrimeCell Real Time Clock (PL031) Technical Reference Manual*

3.18.1 About Real Time Clock

The RTC is an AMBA® subordinate module that connects to the *Advanced Peripheral Bus* (APB).

The following figure shows the RTC block diagram.

Figure 3-15: RTC block diagram



The RTC can be used to provide a basic alarm function or long time base counter. This is achieved by generating an interrupt signal after counting for a programmed number of cycles of a real-time clock input. Counting in one second intervals requires a 1Hz clock input to the RTC.

3.18.2 Features of the RTC

The features of the RTC are:

- Compliance to the Arm AMBA Specification (Rev 2.0) onwards for easy integration into SoC implementation
- 32-bit up counter (free-running counter)
- Programmable 32-bit match compare register
- Software maskable interrupt when counter and compare registers are identical

Additional test registers and modes are implemented for functional verification and manufacturing test.

3.19 GIC-400 Generic Interrupt Controller

This section gives an overview of the product and its features.

For more information, see the product name documentation set:

- *Arm® Generic Interrupt Controller Architecture Specification*
- *Arm® CoreLink™ GIC-400 Generic Interrupt Controller Technical Reference Manual*

3.19.1 About the GIC-400

The CoreLink™ GIC-400 is a high-performance, area-optimized interrupt controller with an Advanced Microcontroller Bus Architecture (AMBA) Advanced eXtensible Interface (AXI) interface.

CoreLink™ GIC-400 detects, manages, and distributes interrupts in System on Chip (SoC) configurations.

3.19.2 Features

You can configure the GIC-400 to provide the optimun features, performance, and gate count required for your intended application.

Using the following software-configurable settings of the GIC-400, interrupts can be:

- Enabled or disabled.
- Assigned to one of two groups, Group 0 or Group 1.
- Prioritized.
- Signaled to different processors in multiprocessor implementations.
- Either level-sensitive or edge-triggered.

The GIC-400 implements:

- The GIC Security Extensions, that support:
    - Using Group 0 interrupts as Secure interrupts, and Group 1 interrupts as Non-secure interrupts.
    - Optionally, using the FIQ interrupt request to signal Secure interrupts to a connected processor. The GIC-400 always signals Group 1 interrupts using the IRQ interrupt request.
- The GIC Virtualization Extensions, that provide hardware support for managing virtualized interrupts.

For more information about the GIC Security Extensions and GIC Virtualization Extensions, see the *Arm® Generic Interrupt Controller Architecture Specification*.

You can use the GIC-400 in a multiprocessor system with up to eight processors. The GIC-400 supports systems in which not every processor implements the Arm® Security Extensions or the Arm® Virtualization Extensions. In such cases, each processor uses only the features it is aware of.

For more details, see the *Arm® Generic Interrupt Controller Architecture Specification*.

The GIC-400 implements the following interrupt types:

- 16 Software Generated Interrupts (SGIs).
- Six external Private Peripheral Interrupts (PPIs) for each processor.

Note

Some PPIs have specific purposes.

- One internal PPI for each processor.
- A configurable number of Shared Peripheral Interrupt (SPIs).

The GIC-400 can assert the following signals to indicate pending interrupts to processors:

Physical interrupts:

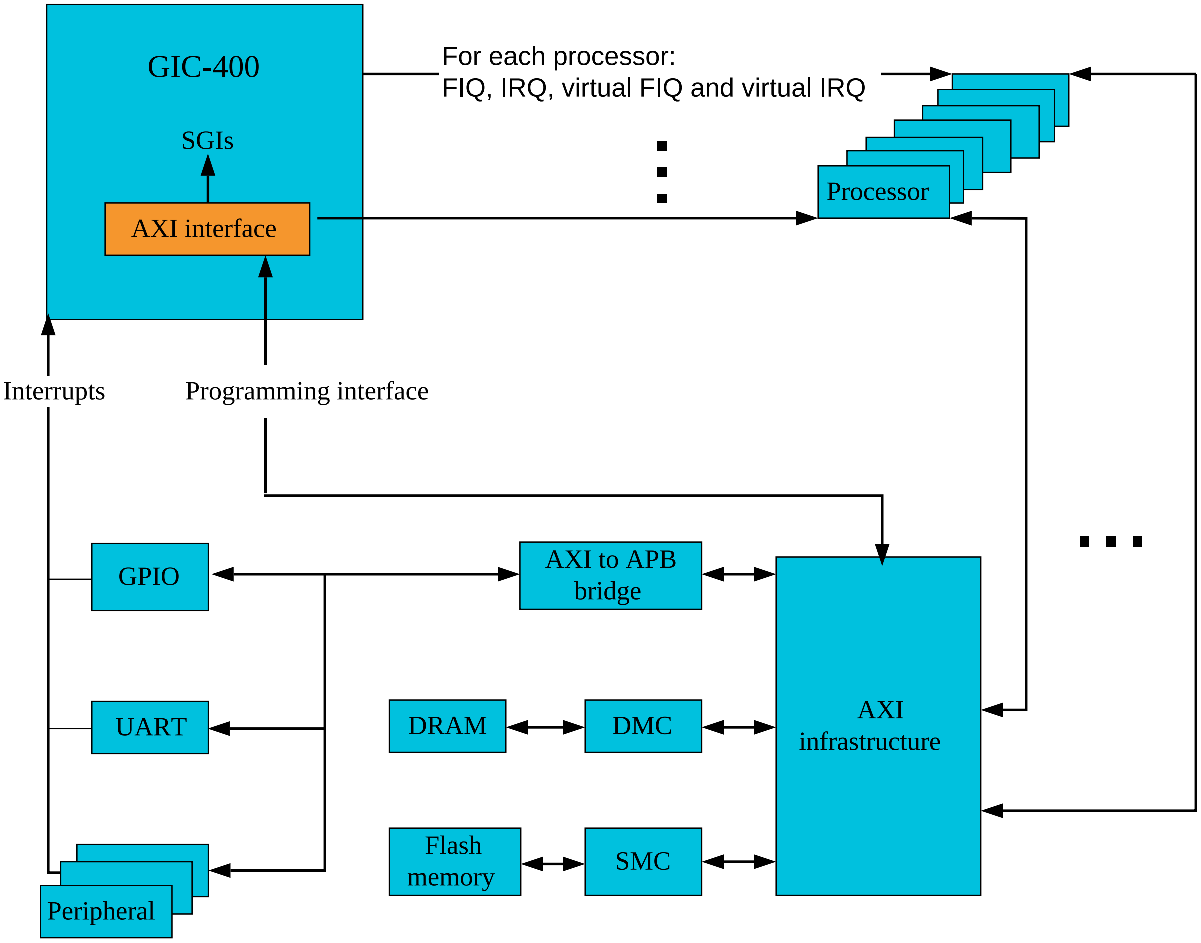
- nFIQCPU[NUM_CPUS–1:0]
- nIRQCPU[NUM_CPUS–1:0

Virtual interrupts

- nVFIQCPU[NUM_CPUS–1:0]
- nVIRQCPU[NUM_CPUS–1:0]

The following diagram provides an overview of the GIC-400 in a multiprocessor system. It shows the interrupts that are sent to the GIC-400 from various sources and the key phases of interrupt-related signaling in the SoC.

Figure 3-16: GIC-400 Overview



The GIC-400 detects PPIs and SPIs from interrupt input signals. There is one signal for each processor for every PPI interrupt ID. There is only one input signal for each SPI interrupt ID, irrespective of the number of processors in the SoC. SGIs do not have input signals and are generated in the GIC-400 using the AXI programming interface

Note

The GIC-400 does not synchronize any inputs. Therefore, all input signals, including the SPI and PPI inputs, must be synchronous to CLK

The GIC-400 notifies each processor of the presence of an interrupt or virtual interrupt by using interrupt output signals. There are also interrupt output signals to provide wakeup functionality to a system power controller.

Virtual interrupts are created and managed by special software that is executing on each processor that is running virtual machines. Such hypervisors are not part of the GIC-400 architecture but are necessary for virtualizing interrupts using the interrupt controller.

For an overview on the hypervisor, see the *Arm® Generic Interrupt Controller Architecture Specification*.

Appendix A Revisions

Changes between released issues of this manual are summarized in tables.

The first table is for the first release. Then, each table compares the new issue of the manual with the last released issue of the manual. Release numbers match the revision history in Release Information on page 2.

Table A-1: Issue 0000-01

Change	Location
First release for EAC	-

Table A-2: Differences between issue 0000-01 and issue 0000-02

Change	Location
References to Arm® Corstone™-710 changed to Arm® Corstone™-1000	Entire document

Table A-3: Differences between issue 0000-02 and issue 0000-03

Change	Location
References to *Master* and *Slave* changed to inclusive terms.	Entire document