Arm® CoreLink™ SDK-101 System Design Kit

Revision: r0p1

Technical Overview

arm

101147_0001_00_en

Arm® CoreLink™ SDK-101 System Design Kit

Technical Overview

Release Information

Document History

Issue	Date	Confidentiality	Change
0000-00	13 October 2017	Non-Confidential	First release for r0p0
0001-00	05 October 2018	Non-Confidential	First release for r0p1

Non-Confidential Proprietary Notice

LES-PRE-20349

Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

Product Status

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

Web Address

http://www.arm.com

Contents
Arm® CoreLink™ SDK-101 System Design Kit Technical Overview

Preface

This preface introduces the *Arm® CoreLink™ SDK-101 System Design Kit Technical Overview*.

It contains the following:

About this book

This book is for the Arm® CoreLink™ SDK-101 System Design Kit (SDK-101). It describes the hardware and software for the system.

Product revision status

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

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

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

Intended audience

This book is written for hardware or software engineers who want an overview of the functionality in the CoreLink™ SDK-101 System Design Kit.

Using this book

This book is organized into the following chapters:

Chapter 1 SDK-101 overview
This chapter introduces the Arm CoreLink SDK-101 System Design Kit (SDK-101).

Chapter 2 Functional overview
This chapter describes the IP products included in the SDK-101 license.

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

Glossary

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

See the *Arm® Glossary* for more information.

Typographic conventions

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

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

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

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

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

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

`<and>`

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

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

SMALL CAPITALS

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

Additional reading

This section lists publications by Arm and by third parties.

See *Infocenter*, for access to Arm documentation.

Arm publications

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

- *Arm® CoreLink™ SSE-050 Subsystem for Embedded Technical Reference Manual* (Arm 100918).
- *Arm® Cortex®-M3 Processor Technical Reference Manual* (Arm 100165).
- *Arm® Cortex®-M System Design Kit Technical Reference Manual* (Arm DDI 0479).
- *Arm® CoreLink™ CG092 AHB Flash Cache Technical Reference Manual* (Arm DDI 0569).
- *Arm® PrimeCell Real Time Clock (PL031) Technical Reference Manual* (Arm DDI 0224).
- *Arm® TrustZone® True Random Number Generator Technical Reference Manual* (Arm 100976).
- *Arm® CoreLink™ SDK-101 Release Note* (Arm EPM 136363).
- *Arm® AMBA® APB Protocol Specification* (Arm IHI 0024).
- *Arm® AMBA® Generic Flash Bus Protocol Specification* (Arm IHI 0083).

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

- *Arm® CoreLink™ SSE-050 Subsystem for Embedded Configuration and Integration Manual* (Arm 100919).
- *Arm® Cortex®-M0 and M0+ System Design Kit Example System Guide* (Arm DUI 0559).
- *Arm® Cortex®-M System Design Kit Example System Guide* (Arm DUI 0594).
- *Arm® CoreLink™ GFC-100 Generic Flash Controller Technical Reference Manual* (Arm 101059).
- *Arm® CoreLink™ GFC-100 Generic Flash Controller Configuration and Integration Manual* (Arm 101060).

Note

- 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.

Feedback

Feedback on this product

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

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

Feedback on content

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

- The title *Arm CoreLink SDK-101 System Design Kit Technical Overview*.
- The number 101147_0001_00_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note

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

Chapter 1
SDK-101 overview

This chapter introduces the Arm CoreLink SDK-101 System Design Kit (SDK-101).

It contains the following sections:

1.1 About the SDK-101

The SDK-101 provides a subsystem architecture, a reference platform, and a collection of IP products that can be used to create an IoT endpoint system. To provide this functionality, the SDK-101 product grants licenses to the following subsystem and component IP products:

CoreLink SSE-050 Subsystem for Embedded

The SSE-050 provides a starting point for a product in the *Internet of Things* (IoT) and embedded market segments. The SSE-050 provides a process and technology agnostic reference, pre-integrated, validated, hardware and software subsystem for Arm Cortex-M3 processors that can be extended to provide an IoT endpoint system.

Cortex-M System Design Kit

The Cortex-M System Design Kit provides an example system for the Arm Cortex-M processors and reusable AMBA AHB-Lite and APB components for low-power designs.

Cortex-M0 and M0+ System Design Kit

The Cortex-M0 and Cortex-M0+ System Design Kit provides an example system-level design for the Arm Cortex-M0 and Cortex-M0+ processors and reusable AMBA components for system-level development.

CoreLink CG092 AHB Flash Cache

The CG092 is an instruction cache that is designed to be instantiated between the bus interconnect and the *embedded Flash* (eFlash) controller.

PrimeCell Real Time Clock

The *Real Time Clock* (RTC) is an AMBA slave module that connects to the *Advanced Peripheral Bus* (APB).

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 is achieved by use of a 1Hz clock input to the RTC.

TrustZone True Random Number Generator

The *True Random Number Generator* (TRNG) provides an assured level of entropy (as analyzed by Entropy Estimation logic). The output from the TRNG can be used to seed deterministic random bit generators.

GFC-100 Generic Flash Controller

The GFC-100 Generic Flash Controller enables embedded Flash macros to be integrated easily into any system.

1.1.1 Using the SDK IP products

The SDK-101 licensed IP can be used in the following ways:

- Use the CoreLink SSE-050 subsystem as a verified foundation for your own IoT solution that is based around the Cortex-M3 processor.
- Use the *Cortex-M System Design Kit* (CMSDK) and the example system as a starting point for your own IoT solution that is based around the Cortex-M0, Cortex-M0+, Cortex-M3, or Cortex-M4 processors.
- Use the Cortex-M0 and Cortex-M0+ System Design Kit and the example system as a foundation for your own IoT solution that is based around the Cortex-M0 or Cortex-M0+ processors.
- Use the system IP provided with the SSE-050, CMSDK, CG092, and your own IP to create a custom solution. You can use the example systems and software libraries as a reference for your system solution.

Note

- The SSE-050 and CMSDK build scripts include interconnections to a processor, but a processor must be separately licensed and installed.
- See the *Arm® CoreLink™ SDK-101 Release Note* for details about how to download and install the SDK-101 components that you require.

1.2 Product deliverables

The CoreLink SDK-101 product bundle does not have hardware or software deliverables. Its subsystems and IP component products include these deliverables.

The hardware deliverables must be downloaded separately for the following IP products that are included in the SDK-101 license:

- CoreLink SSE-050 Subsystem for Embedded (CG063).
- Cortex-M System Design Kit (BP210).
- Cortex-M0 and M0+ System Design Kit (BP200).
- CoreLink CG092 AHB Flash Cache (CG092).
- PrimeCell Real Time Clock (PL031).
- TrustZone True Random Number Generator (CC003).
- GFC-100 Generic Flash Controller (CG090).

See the *Arm® CoreLink™ SDK-101 Release Note* for the component versions.

1.3 Compliance

See the *Technical Reference Manuals* for more details of the product's compliance to the following specifications:

- Arm architecture.
- CoreSight™ debug.
- Interrupt controller architecture.
- Advanced Microcontroller Bus Architecture.
- AMBA Generic Flash Bus.

1.4 Documentation

The following documents are supplied with the CoreLink SDK-101 product bundle:

Technical Overview

The *Technical Overview* (TO) describes the functionality of the SDK-101 System Design Kit.

Release Note

The *Release Note* describes download and installation instructions for the IP products included in the SDK-101.

——— **Note** ———

- The separately downloaded product bundles also contain documentation such as *Technical Reference Manual*s or *Configuration and Integration Manual*s.
- See the individual product bundles for details of what documentation is provided for that IP bundle.

———————————

Chapter 2
Functional overview

This chapter describes the IP products included in the SDK-101 license.

It contains the following sections:

2.1 Supported processors

The following table lists the processors supported by the products in the CoreLink SDK-101 bundle.

Table 2-1 Supported processors

CoreLink SDK-101 products							
Processor	**CM0SDK**	**CMSDK**	**AHB Flash Cache**	**SSE-050 Subsystem**	**RTC**	**TRNG**	**GFC-100**
Cortex-M0	Yes	-	Yes	-	Yes	Yes	Yes
Cortex-M0+	Yes	-	Yes	-	Yes	Yes	Yes
Cortex-M23	-	Yes	Yes	-	Yes	Yes	Yes
Cortex-M3	-	Yes	Yes	Yes	Yes	Yes	Yes
Cortex-M4	-	Yes	Yes	-	Yes	Yes	Yes

2.2 CoreLink SSE-050 Subsystem for Embedded

The Arm CoreLink SSE-050 Subsystem for Embedded is a subsystem that provides a starting point for a product in the *Internet of Things* (IoT) and embedded market segments.

The SSE-050 subsystem delivers a process and technology agnostic reference, pre-integrated, validated, hardware and software subsystem that can be extended to provide an IoT endpoint system.

The SSE solution consists of hardware, software, and software tools to enable the rapid development of IoT *System on Chip* (SoC) solutions.

The SSE-050 Subsystem for Embedded contains the following components:

- A Cortex-M3 processor:
  - Bit-banding enables using standard instructions to read or modify individual bits. The default implementation does not include bit banding.
  - Eight MPU regions (optional).
  - NVIC providing deterministic, high-performance interrupt handling with a configurable number of interrupts.
  - *Wakeup Interrupt Controller* (WIC) with configurable number of WIC lines (optional). This is a latch-based WIC implementation, and not the standard Cortex-M3 WIC. See the *Arm® CoreLink™ SSE-050 Subsystem Configuration and Integration Manual* for more information.
  - Little-endian memory addressing only (for compatibility with the eFlash cache).

  For more information, see the *Arm® Cortex®-M3 Processor Technical Reference Manual*.

  The Cortex-M3 processor has a *Processor Integration Layer* (PIL) to simplify integration of the SSE-050 Subsystem into a multiprocessor system with a SoC-level CoreSight subsystem.
- Configurable Debug and Trace as either:
  - Standalone system with a *Trace Port Interface Unit* (TPIU) and an SWJ-DP.
  - Full CoreSight integration over a DAP and the ATB buses.
- Multilayer AMBA AHB-Lite interconnect:
  - Low-latency interconnect bus matrix.
  - Two AHB-Lite initiator expansion ports for external AHB masters.
  - Two AHB-Lite target expansion ports for external AHB slaves.
  - 11 APB4 target expansion ports (each with 4KB address space) to connect APB peripherals.
- Memory system, consisting of:
  - Placeholder for eFlash controller and optional cache.
  - Static memory (configurable as one to four 32KB banks) is provided in the example integration layer.
- Two APB timers:
  - Interrupt generation when the counter reaches 0.
  - Each timer has an **TIMERnEXTIN** signal that can be used as an enable or external clock.
  - Configurable privileged access mode.

The following figure shows the SSE-050 Subsystem for Embedded, with other IP, in an example design.

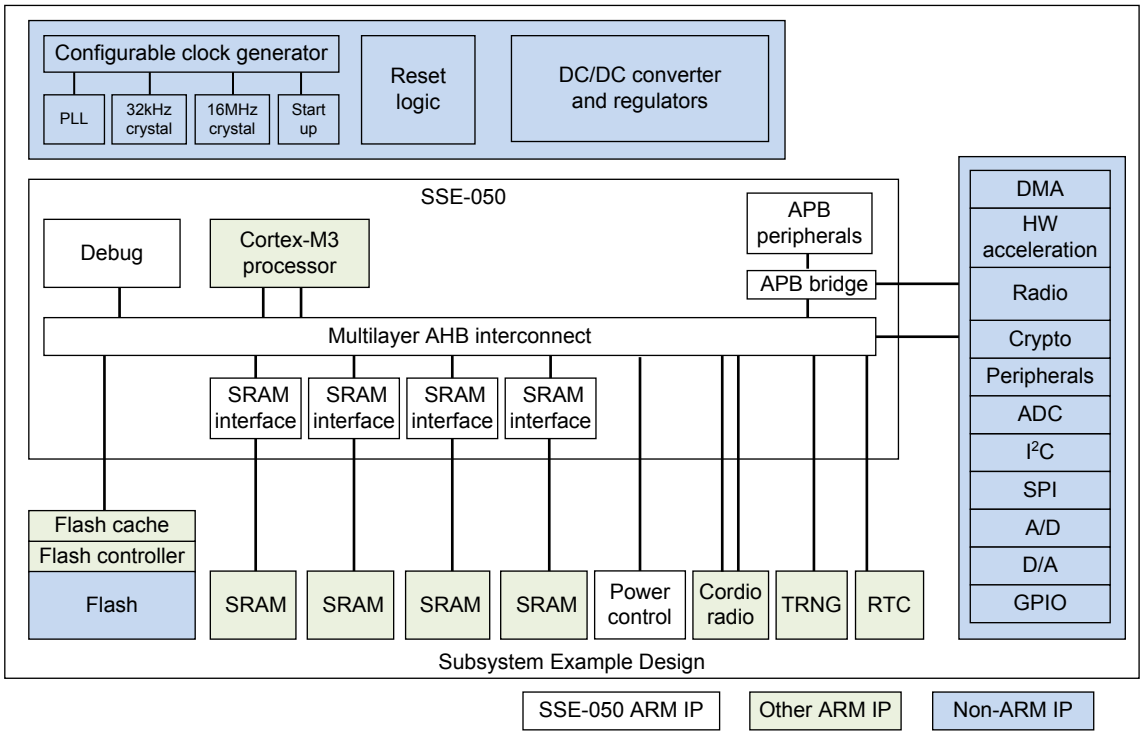


Figure 2-1 SSE-050 Subsystem for Embedded example design

2.2.1 Subsystem for Embedded software

Application processor firmware, which is available separately, consists of the code that is required to boot the subsystem up to the point where the OS execution starts. Contact your Arm representative for details on the software and its location.

The firmware contains:

- *Cortex® Microcontroller Software Interface Standard* (CMSIS) compliant drivers.
- Flash programming support code, which is separate from Mbed OS.
- Mbed OS ported onto the SSE-050 system.
- Mbed OS driver support and code, for example I/O peripherals.
- Code that is required to load Mbed from boot media and set up the initial security environment.

2.3 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.

  These components are used in the CMSDK example system, but you can also use the components to create your own custom system.
- An example system for supported processor products.
- Example synthesis scripts for the example system.
- 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).
- Documentation including:
  - *Arm® Cortex®-M System Design Kit Technical Reference Manual*.
  - *Arm® Cortex®-M0 and Cortex®-M0+ System Design Kit Example System Guide*.
  - *Arm® Cortex®-M System Design Kit Example System Guide*.

For details of the CMSDK components, see the *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

This section contains the following subsections:

- *2.3.1 Example system* on page 2-19.
- *2.3.2 Components* on page 2-20.
- *2.3.3 Cortex-M Software Design Kit software* on page 2-20.

2.3.1 Example system

The following figure shows the block diagram of the CMSDK example system.

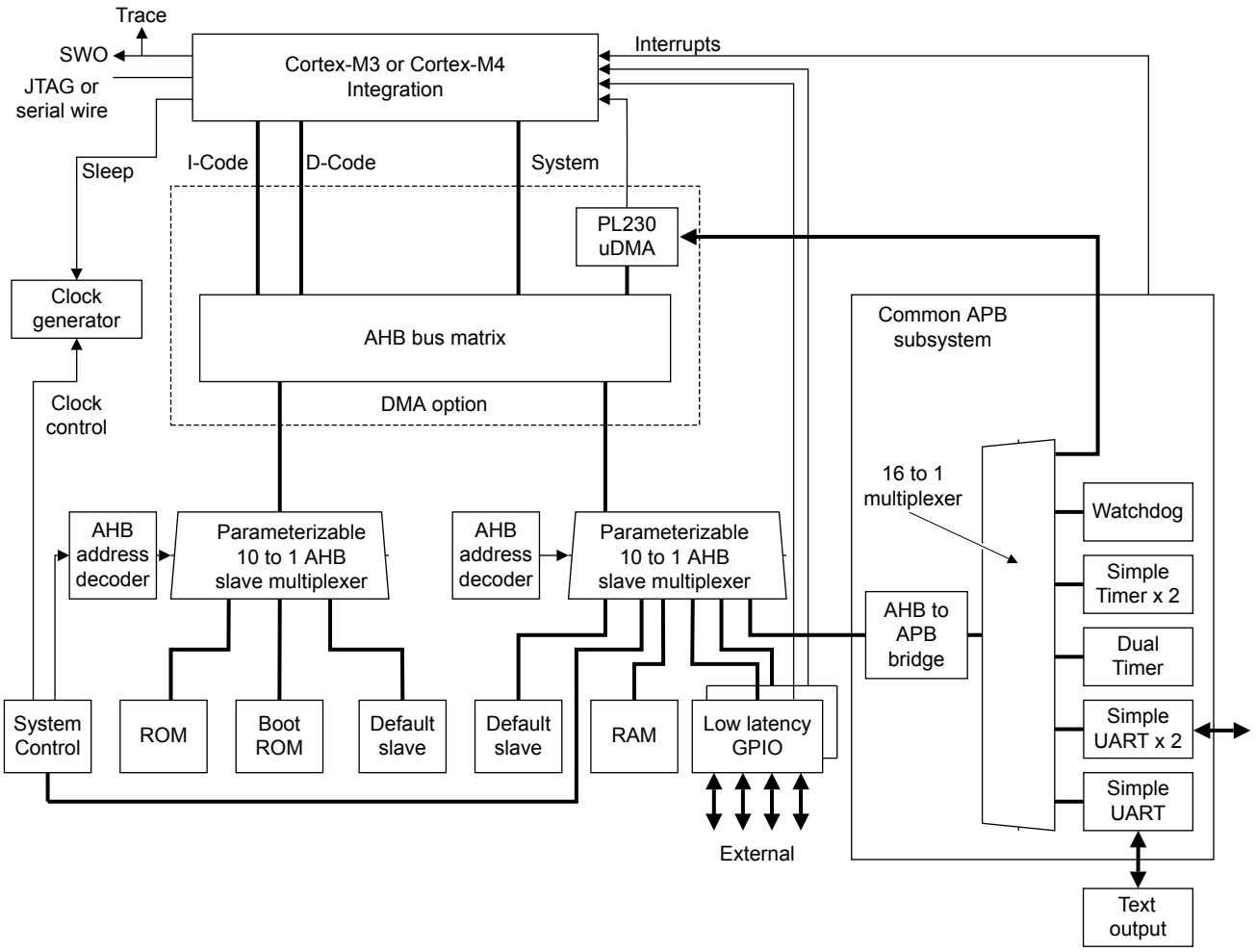


Figure 2-2 CMSDK example system

Note

The µDMA Controller (PL230) is not included in the SDK-101 license and, if instantiated, must be licensed separately. See the *Arm® PrimeCell µDMA Controller (PL230) Technical Reference Manual* for more information.

2.3.2 Components

The CMSDK example system consists of the following components and models:

- Basic AHB-Lite components.
- APB components.
- Advanced AHB-Lite components.
- Behavioral memory models.
- Verification components.

2.3.3 Cortex-M Software Design Kit software

The Cortex-M System Design Kit includes the following software:

- CMSIS-compliant drivers.
- Device-specific header files, startup code, and example drivers including retargeting code for the `printf()` and `puts()` functions.

- Platform hardware adaptation layer code that is required in addition to the open-source code and generic Cortex-M processor header files.
- Mbed OS driver support.

  Extra Cortex-M code is available on the Mbed website.
- Shell scripts to sync, build, and run the software.

2.4 Cortex-M0 and M0+ System Design Kit

The Cortex-M0 and Cortex-M0+ System Design Kit provides:

- An example system-level design for the Arm Cortex-M0 and Cortex-M0+ processors.
- Reusable AMBA components for system-level development from the CMSDK.

For information on the AMBA components that the design kit uses, see the *Arm® Cortex®-M System Design Kit Technical Reference Manual*.

This section contains the following subsections:

- *2.4.1 About the example system* on page 2-22.
- *2.4.2 Cortex-M0 and Cortex-M0+ software* on page 2-23.

2.4.1 About the example system

The *Arm® Cortex®-M0 and M0+ System Design Kit Example System Guide* describes an example system for the Cortex-M0 and Cortex-M0+ processors.

The following figure shows the block diagram for the example system.

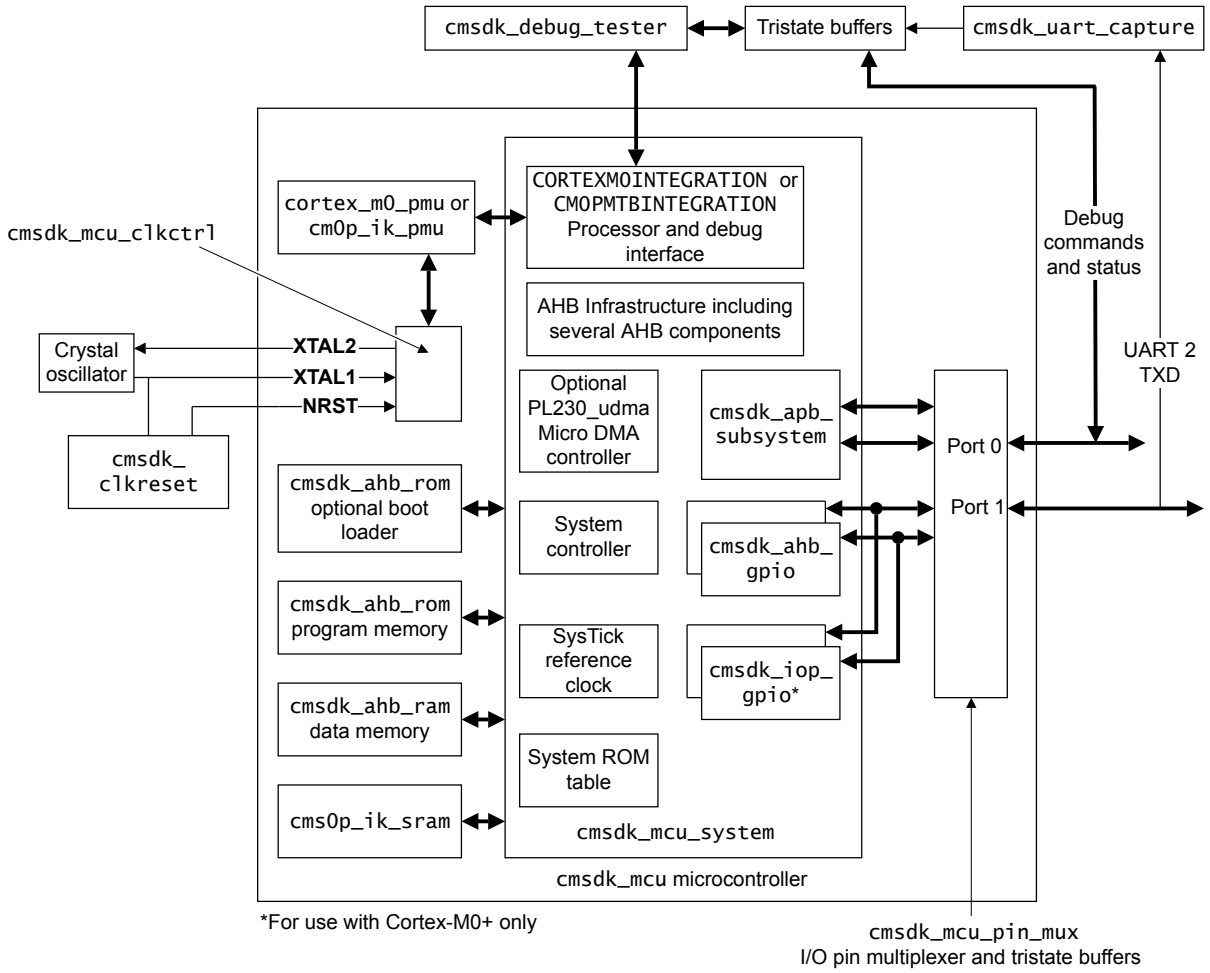


Figure 2-3 CMSDK example system

The example system is a simple microcontroller design that contains the following:

- A single Cortex-M0 or Cortex-M0+ processor.
- Internal program memory.

- SRAM data memory.
- Boot loader.
- The following peripherals:
  - Several timers.
  - *General-Purpose input/output* (GPIO).
  - *Universal Asynchronous Receiver Transmitter* (UART).
  - Watchdog timer.
- Debug connection.

Note

The optional µDMA Controller (PL230) is not included in the SDK-101 license and, if instantiated, must be licensed separately. See the *Arm® PrimeCell µDMA Controller (PL230) Technical Reference Manual* for more information.

2.4.2 Cortex-M0 and Cortex-M0+ software

The Cortex-M0 and M0+ System Design Kit products include the following software:

- CMSIS-compliant drivers.
- Device-specific header files, startup code, and example drivers including retargeting code for the `printf()` and `puts()` functions.
- Platform hardware adaptation layer code that is required in addition to the open-source code and generic Cortex-M processor header files.
- Mbed OS driver support.

  Extra Cortex-M0 and Cortex-M0+ code is available on the Mbed website.
- Shell scripts to sync, build, and run the software.

2.5 CoreLink CG092 AHB Flash Cache

The CG092 AHB Flash Cache is an instruction cache that is instantiated between the bus interconnect and the eFlash controller.

The CG092 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 CG092 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.

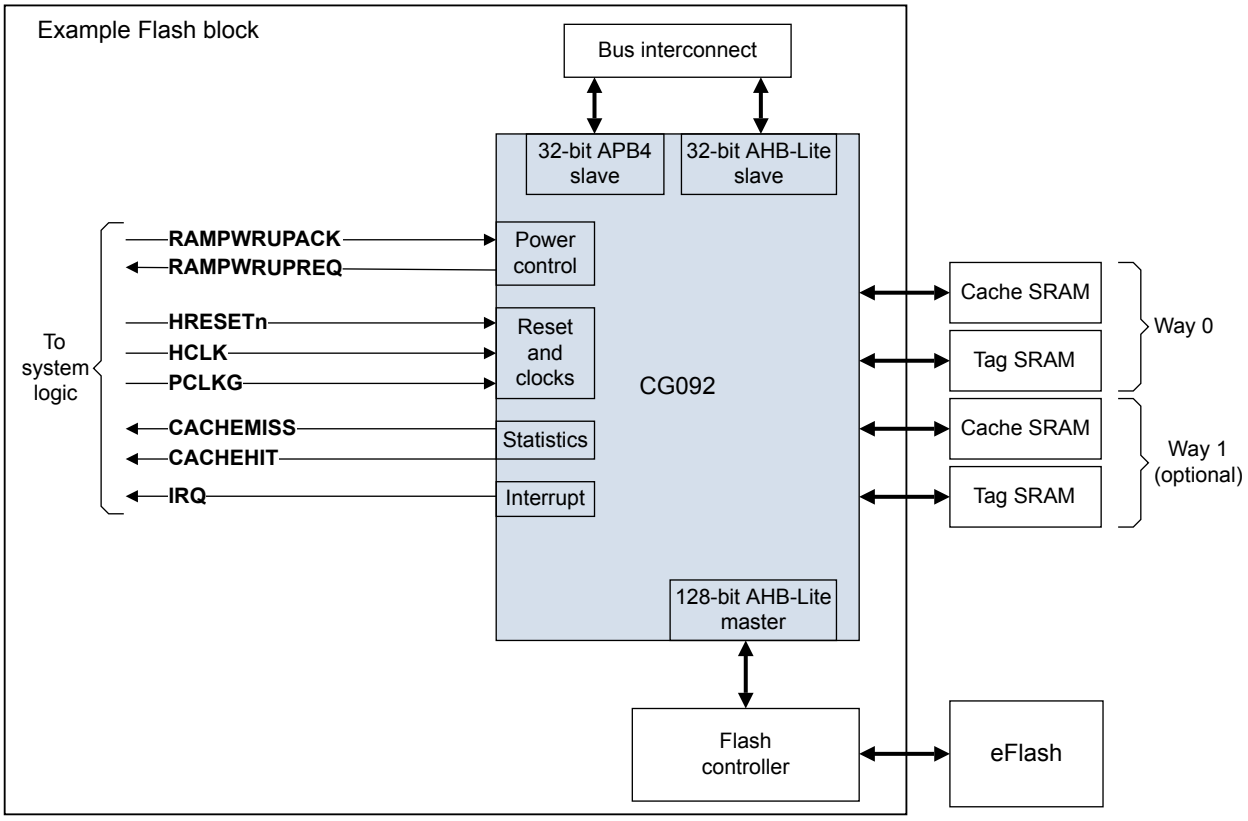


Figure 2-4 Example eFlash implementation

2.6 GFC-100 Generic Flash Controller

The Arm CoreLink 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.

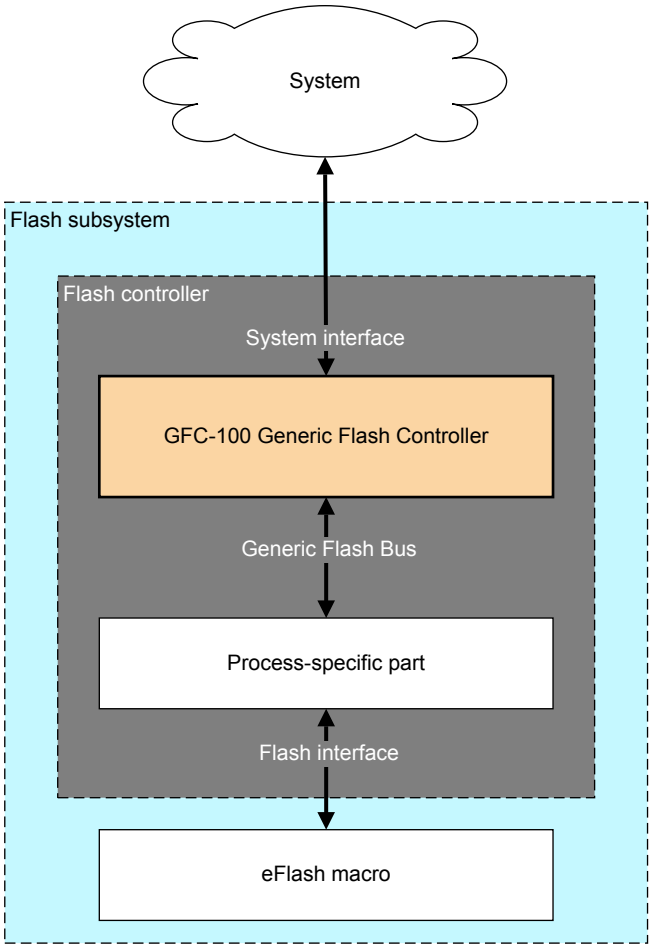
An eFlash macro enables a Flash controller to access eFlash memory. The eFlash macros produced by different foundries and processes can have different interfaces, timings, signal names, protocols and features that are determined by the foundry processes that produced the eFlash memory.

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.



Note

The process-specific part is not included and must be either licensed or designed separately.

Figure 2-5 GFC-100 in a Flash controller implementation

GFC-100 provides several interfaces and test features:

- *Advanced High-performance Bus* (AHB) interface.
- *Advanced Peripheral Bus* (APB) slave interface.
- APB register master interface.
- Q-Channel interface.
- P-Channel master interface.
- *Generic Flash Bus* (GFB)

2.7 Real Time Clock

The *Real Time Clock* (RTC) is an AMBA slave module that connects to the *Advanced Peripheral Bus* (APB).

The following figure shows the RTC block diagram.

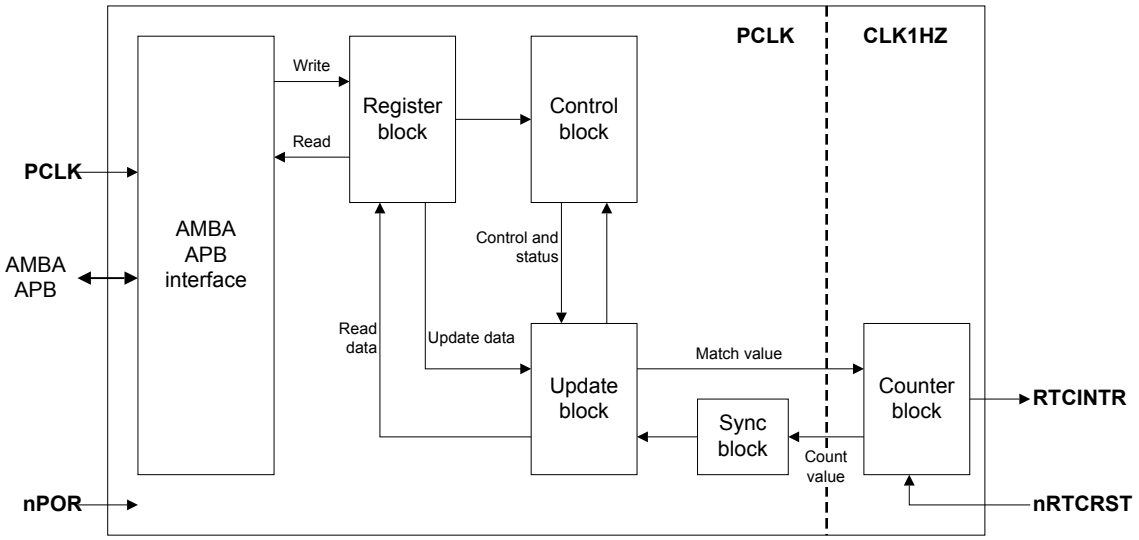


Figure 2-6 RTC block diagram

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.

2.8 True Random Number Generator

The *True Random Number Generator* (TRNG) provides an assured level of entropy (as analyzed by Entropy Estimation logic). The output from the TRNG can be used to seed deterministic random bit generators.

The following figure shows the TRNG.

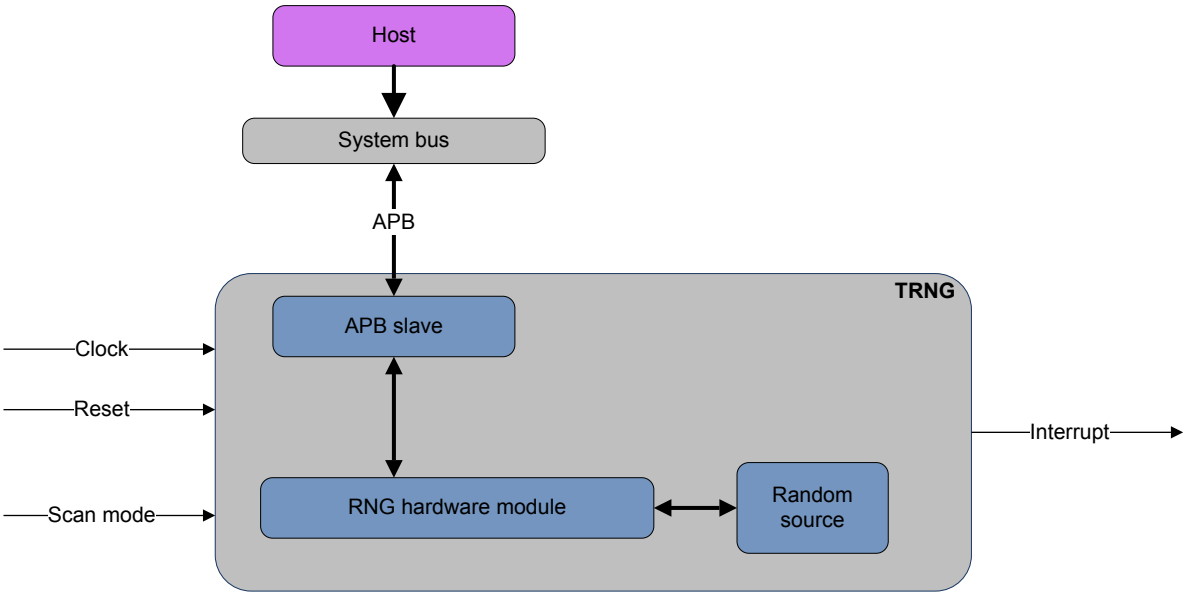


Figure 2-7 TRNG hardware overview

Appendix A
Revisions

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

It contains the following section:

- *A.1 Revisions* on page Appx-A-30.

A.1 Revisions

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

Table A-1 Issue 0000-00

Change	Location	Affects
First release	-	-

Table A-2 Issue 0001-00

Change	Location	Affects
Name of Gereric Flash Controller changed from *Vultan* to *CoreLink™ GFC-100*.	• *1.1 About the SDK-101* on page 1-10 • *1.2 Product deliverables* on page 1-12 • *2.6 GFC-100 Generic Flash Controller* on page 2-25	r1p0 release