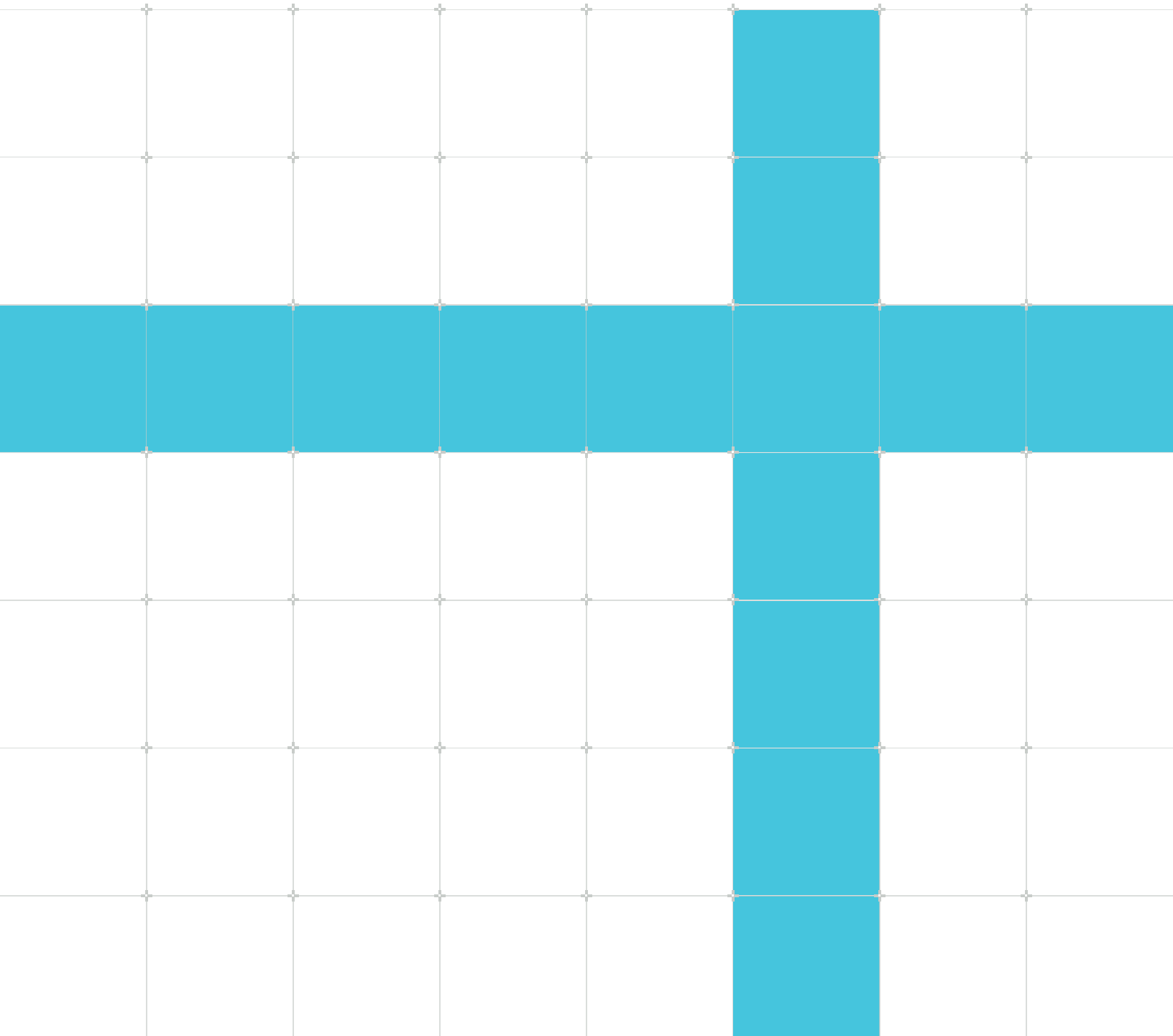
Arm® Cortex®-A5 DesignStart™

Revision: r1p0

Getting Started Guide

Non-confidential

Issue 00
101857_0100_00_en



Release Information

Issue	Date	Confidentiality	Change
0100-00	20 September 2019	Non-Confidential	New document for r1p0

Proprietary Notice

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.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. 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.

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 for an Alpha product, that is a product under development.

Web Address

www.arm.com

Contents

1 Introduction

1.1 Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this manual, 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.

1.2 Intended audience

The Getting Started Guide describes the different parts of the Arm® Cortex®-A5 DesignStart™ product. Read this guide if you are new to Arm IP or want an overview of how DesignStart can be used as the starting point for creating your own Internet of Things (IoT) application.

Conventions

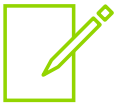
The following subsections describe conventions used in Arm documents.

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

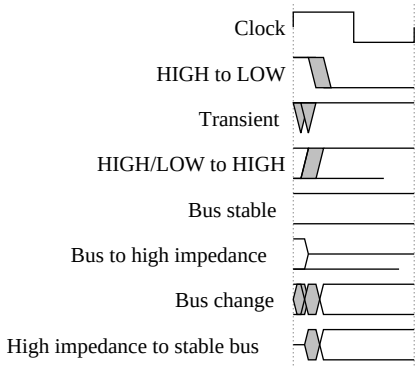
Convention	Use
italic	Introduces special terminology, denotes cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
`monospace italic`	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
`monospace`	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace bold`	Denotes language keywords when used outside example code.
`monospace underline`	Denotes 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	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**.
 Caution	
 Warning	
 Note	

Timing diagrams

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

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

Figure 1: Key to timing diagram conventions



Signals

The signal conventions are:

Signal level

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

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

Lowercase n

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

1.4 Additional reading

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

See https://developer.arm.com for Arm documents.

Table 1: Arm publications

Document Name	Document ID	Licensee only
Arm® Cortex®-A5 DesignStart™ Technical Overview	177957708	No
ARM® Architecture Reference Manual ARMv7-A and ARMv7-R edition	DDI 0406C.D	No
Arm® Debug Interface Architecture Specification ADI v5.0 to ADI v5.2	IHI 0031E	No
ARM® CoreSight SoC-400 Technical Reference Manual	100536	No
Arm® Server Base System Architecture 5.0 Platform Design Document	DEN 0029B	No
ARM® PrimeCell UART (PL011) Technical Reference Manual	DDI 0183G	No
ARM® PrimeCell Real Time Clock (PL031) Technical Reference Manual	DDI 0224C	No
ARM® PrimeCell General Purpose Input/Output (PL061) Technical Reference Manual	DDI 0190B	No
ARM® TrustZone® True Random Number Generator Technical Reference Manual.	100976	No
ARM® Cortex®-A5 MPCore Technical Reference Manual	DDI 0434C	No
ARM® CoreSight ETM-A5 Technical Reference Manual	DDI 0435C	No
ARM® CoreLink™ Level 2 Cache Controller L2C-310 Technical Reference Manual	DDI 0246H	No
ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual	DDI 0475G	No

Document Name	Document ID	Licensee only
Arm® Debug Interface Architecture Specification ADI v5.0 to ADI v5.2	IHI 0031E	No
ARM® PrimeCell Infrastructure AMBA™ 3 AXI Internal Memory Interface (BP140) Technical Overview	DTO 0009A	No
ARM® PrimeCell Synchronous Serial Port (PL022) Technical Reference Manual	DDI 0194H	No

1.5 Feedback

Arm welcomes feedback on this product and its documentation.

1.5.1 Feedback on this product

Information about how to give feedback on the 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.

1.5.2 Feedback on content

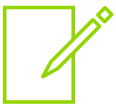
Information about how to give feedback on the content.

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

- The title Arm® Cortex®-A5 DesignStart™ Getting Started Guide.
- The number 101857_0100_00_en.
- If applicable, the page number(s) to which your comments refer.

- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.



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.

2 Overview of Cortex-A5 DesignStart

Cortex-A5 DesignStart includes a comprehensive hardware and software package to enable you to develop custom chip designs and build Linux capable silicon.

Cortex-A5 DesignStart is the ideal starting point for designs aimed at rich embedded and IoT applications and devices.

This Cortex-A5 based product builds on the success of DesignStart packages for Cortex-M0 and Cortex-M3, which are targeted at developing constrained applications.

Example Cortex-A5 DesignStart applications include smart homes, gateways, and smart IP cameras. Devices may also support complex decision making and machine learning.

For the most up-to-date information, see, Cortex-A5 DesignStart.

2.1 What is included in DesignStart

Depending on the maturity of the product, DesignStart can include:

The Cortex-A5 DesignStart example subsystem reference design

A Cortex-A5 based subsystem that includes other Arm IP, such as CoreLink NIC-400 Network Interconnect and CoreSight SoC-400 components.

An example integration layer, that gives an example of how to integrate the subsystem into a full SoC design.

See Cortex-A5 DesignStart subsystem reference design.

Arm IP

Bundles of Cortex-A5 and System IP for implementing a SoC based on the reference design.

See Subsystem IP and components and Example integration layer IP and components

Simulation models and software

A Fast model and an integrated software stack, that enable Linux-based software development before hardware availability.

Cycle models that are compiled directly from Arm RTL and retain complete functional and cycle accuracy, so you can make architectural decisions, optimize performance, or develop bare metal software.

See Modeling your system and Software stack.

Development tools

Arm CoreLink Socrates™ for configuring and building Arm IP (fixed term license).

Arm Development Studio for software development (fixed term license).

Global support

Dedicated support from a global team of experienced Arm engineers.

2.2 The Cortex-A5 DesignStart documents

Arm delivers a documentation set with each of its products.

The documents for Cortx-A5 DesignStart are:

Arm® Cortex™-A5 DesignStart™ Getting Started Guide

The Getting Started Guide describes the different parts of the Cortex-A5 DesignStart product. Read this guide if you are new to Arm IP or want an overview of how DesignStart can be used to create your own embedded or IoT application.

Arm® Cortex™-A5 DesignStart™ Technical Overview

The Technical Overview describes the Arm IP in Cortex-A5 DesignStart and outlines the main features.

The other documents for Arm IP referred to in this guide are in their respective IP bundles.

Technical information in this guide supersedes information in the Technical Overview v1.0.

3 Cortex-A5 DesignStart subsystem reference design

The subsystem and integration layer are the starting point for your custom SoC design. The topology, functional design blocks, and IP give a top-level overview of the reference design.

The reference design includes:

A configurable Cortex-A5 based subsystem

The subsystem implements major functionality of a system that is able to support embedded Linux.

The subsystem is designed using Arm IP and components, and is partitioned into the following functional blocks:

- Processor block
- Debug block
- Interconnect block
- Peripheral block
- Clock and reset block

An example integration layer

The integration layer gives an example of how additional IP can be integrated around the subsystem to form a complete SoC. It consists of example Arm IP which are not suitable for direct silicon implementation and wrappers for third party IP required by a SoC.

Figure 2: Cortex-A5 DesignStart reference design topology on page 16 shows the IP, components, and connections in the reference design.

Figure 2: Cortex-A5 DesignStart reference design topology

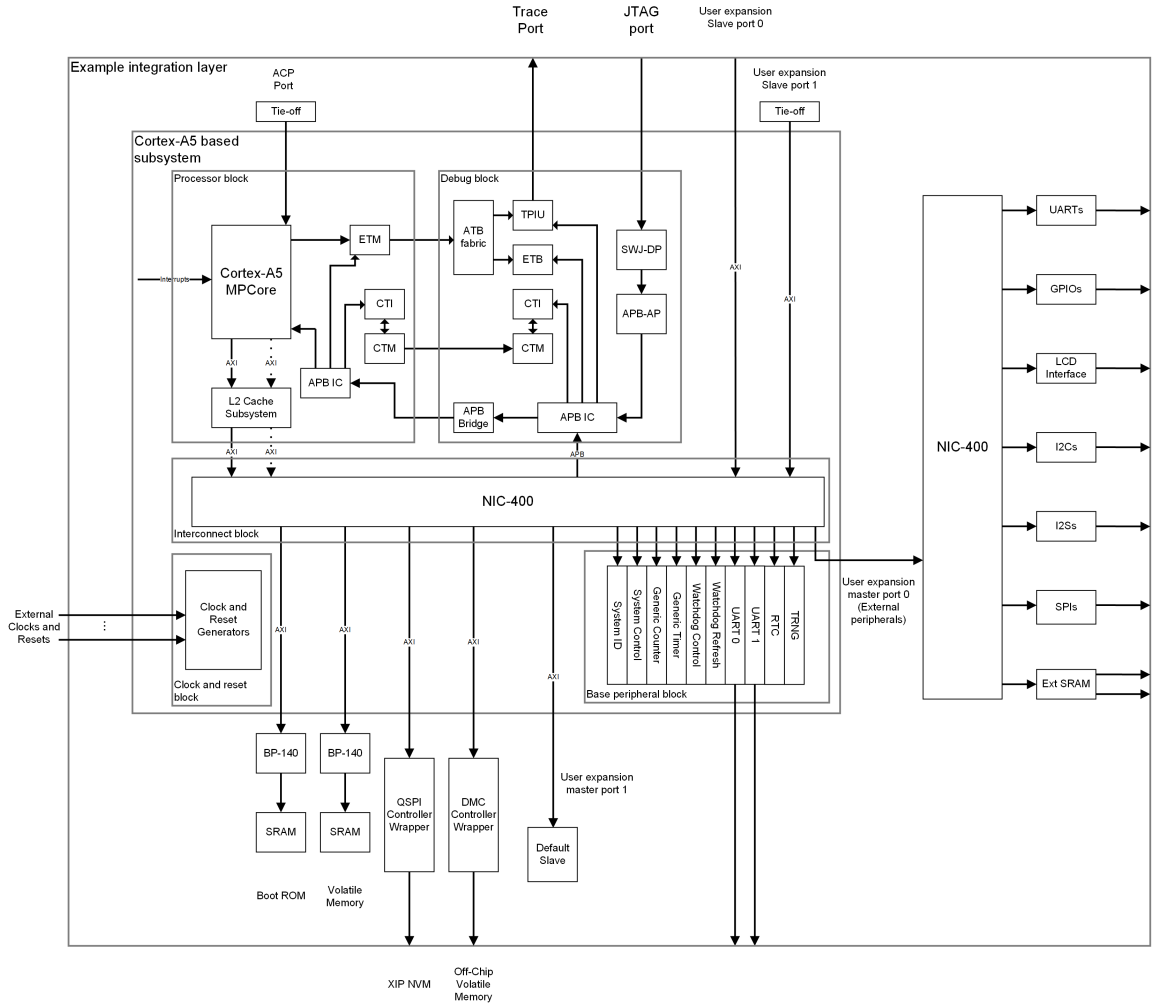


Table 2: Abbreviations used in Figure 2: Cortex-A5 DesignStart reference design topology on page 16

Abbreviation	Description
APB IC	Advanced Peripheral Bus Interconnect Signals
APB-AP	An optional component of the Debug Access Port that provides an AHB interface to an SoC
ATB	An AMBA bus protocol for trace data
BP-140	PrimeCell Infrastructure AMBA 3 AXI Internal Memory Interface
CTI	Cross Trigger Interface
CTM	Cross Trigger Matrix
DMC	Dynamic Memory Controller
ETB	Embedded Trace Buffer
ETM	Embedded Trace Macrocell
I2C	Inter-Integrated Circuit controller
I2S	Inter-IC Sound Bus Controller
QSPI	Quad Serial Port Interface
RTC	Real Time Clock
SPIs	Shared Peripheral Interrupts
SWJ-DP	Combined JTAG-DP and SW-DP
TPIU	Trace Port Interface Unit
TRNG	True Random Number Generator

3.1 Subsystem IP and components

The subsystem consists of the following Arm IP:

Cortex-A5 MPCore processor with optional FPU and NEON extensions

The Cortex-A5 MPCore processor is a high-performance, low-power, Arm macrocell with an L1 cache subsystem that provides full virtual memory capabilities. Up to four individual cores can be linked in a cache-coherent cluster, under the control of a Snoop Control Unit (SCU), that maintains L1 data cache coherency for memory marked as shared.

CoreLink Level 2 Cache Controller L2C-310

The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a recognized method of improving the performance of ARM-based systems

when the processor generates significant memory traffic. The cache controller is a unified, physically addressed, physically tagged cache with up to 16 ways.

CoreSight SoC-400 solution for debug and trace

CoreSight SoC-400 is a comprehensive library of components for the creation of debug and trace functionality within a system.

CoreSight ETM-A5 Embedded Trace Macrocell for Cortex-A5

The ETM-A5 is a real-time trace module providing instruction and data tracing of the Cortex-A5 microprocessor.

CoreLink NIC-400 Network Interconnect

The NIC-400 enables you to create a complete high performance, optimized, and AMBA®-compliant network infrastructure that can range from a single bridge component to a complex interconnect of up to 128 masters and 64 slaves of AMBA protocols.

True Random Number Generator (CC003)

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

PrimeCell Real Time Clock (PL031)

The Real Time Clock (RTC) is an AMBA slave 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.

PrimeCell UART (PL011)

The UART is an Advanced Microcontroller Bus Architecture (AMBA) compliant SoC peripheral that is developed, tested, and licensed by Arm. The UART is an AMBA slave module that connects to the Advanced Peripheral Bus (APB). The UART includes an Infrared Data Association (IrDA) Serial InfraRed (SIR) protocol ENcoder/ DECoder (ENDEC).

PrimeCell General Purpose Input/Output (PL061)

The PrimeCell GPIO is an Advanced Microcontroller Bus Architecture (AMBA) bus slave that connects to the AMBA Advanced Peripheral Bus (APB). It provides eight programmable inputs or outputs that you can control in software or hardware modes.

Generic timer and counter

The subsystem includes a Generic Timer and Generic Counter as specified in the Arm ARM. Refer to Arm Architecture Reference Manual ARMv7-A and ARMv7-R edition *Arm Architecture Reference Manual ARMv7-A and ARMv7-R edition* Section B8 and Appendix E for programmers model of Generic Timers and Counter.

Generic watchdog

If the watchdog is not refreshed within a certain time it raises an interrupt signal. The watchdog uses the Generic Timer system counter as the time base against which the decision to trigger an interrupt is made.

3.2 Example integration layer IP and components

The integration layer shows how additional IP and components can be integrated around the subsystem to form a complete system or SoC.

The example integration layer includes:

- Static Memory Controller (SMC) from PrimeCell Infrastructure AMBA 3 AXI Internal Memory Interface (BP140) (non-programmable component).
- ROM and RAM.
- Quad Serial Port Interface (QSPI) controller wrapper which, depending on your requirements, can be replaced with a third-party QSPI controller.
- Dynamic Memory Controller (DMC) controller wrapper which, depending on your requirements, can be replaced with a third-party DMC controller.
- External interconnect, that is implemented by NIC-400.
- External peripherals:
  - Inter-Integrated Circuit controller (I^2C).
  - Inter-IC Sound Bus Controller (I^2S).
  - PrimeCell GPIO (PL061) general purpose I/O device.
  - PrimeCell UART (PL011).
  - Color LCD (CLCD) interface.
  - SPI from PrimeCell Synchronous Serial Port (PL022).

4 Modeling your system

There are different models that are available for designing and testing your system. These models enable you to develop software ahead of hardware availability.

Cortex-A5 DesignStart currently includes two types of models, that offer a trade-off between precision and operating speed:

Fast Model

Arm Fast Models are fast, functionally accurate models suited to software development or high-level modeling. The Cortex-A5 DesignStart Fast Model is delivered as a pre-built Fixed Virtual Platform (FVP) which is quick and easy to setup and start your software development. The FVP works with the software stack to give out-of-box Linux support, see Software stack.

Cycle Model

For cycle accurate simulation, you can use Arm Cycle Models, which are derived directly from RTL and are 100% cycle accurate. The Cortex-A5 DesignStart Cycle Model is delivered as a pre-built Cycle Performance Analysis Kit (CPAK) and is ideal for performance analysis of low-lever software.

Configurable versions of both model types can be purchased from Arm or selected distributors.

The models are provided with a high level of instrumentation and support debug access by source level debuggers such as Arm Development Studio and a variety of others from Arm ecosystem partners. Tools for customizing the models can be purchased separately.

All models are based on the Arm Versatile Expres memory map. For a full description of the VE memory map for Cortex-A series, see the Fast Models Reference Manual Version 11.8.

For more details and to download the models, visit https://developer.arm.com/tools-and-software/simulation-models/designstart-simulation-models/cortex-a5-designstart-model.

5 Software

An extensible software stack is available that provides a reference Linux Operating System to accelerate your application on Cortex-A5 DesignStart.

The software stack represents a fully integrated and operational model of the Cortex-A5 DesignStart system.

The reference stack can be modified and configured to meet your specific requirements.

For more details and to download the software, visit https://community.arm.com/developer/tools-software/oss-platforms/w/docs/447/cortex-a5-designstart.

Appendix A Programmers model

The Programmers Model describes the reference design subsystem and integration layer memory regions and registers, and provides information on how to program a SoC that contains an implementation of the reference design.

This appendix provides enough information to enable you to program a model without having a Cortex-A5 DesignStart license.

The following information applies to all registers:

- Reserved locations follow these guidelines:
  - If a peripheral or device does not occupy all the memory allocated to it, the unoccupied region is Reserved.
  - Locations which are marked as Reserved are treated as **RAZ/WI** and must generate an error.
  - Attempting to access reserved or unused address locations can result in unpredictable behavior.
- Unless otherwise stated in the accompanying text:
  - Do not modify undefined register bits.
  - Ignore undefined register bits on reads.
  - Unless otherwise specified, all register bits are reset to a logic 0 by a system or power up reset.
- The following describes the access type:
  - Read and write (RW)
  - Read-only (RO)
  - Write-only (WO)
  - Read-As-Zero and Write-Ignored (**RAZ/WI**)

A.1 Memory map

All peripherals and devices in the subsystem are allocated memory units of at least 64KB to allow the use of any page granule size defined by *ARM Architecture Reference Manual ARMv7-A and ARMv7-R* edition.

A.1.1 Subsystem memory map

The subsystem memory map describes the base address and region information for the subsystem.

Table 3: Subsystem memory map

Base address	Size	Component	Notes
0x0000_0000	64KB	SRAM0 for ROM	Default implementation region
0x0001_0000	960KB	SRAM0 for ROM	Extendable region, any unmapped section of this region is aliased
0x0010_0000	15MB	Reserved	-
0x0100_0000	16MB	Reserved	-
0x0200_0000	2MB	SRAM1	Default implementation region
0x0220_0000	94MB	SRAM1	Extendable region, any unmapped section of this region is aliased
0x0800_0000	40MB	QSPI flash	Default implementation region
0x0A80_0000	88MB	QSPI flash	Extendable region, any unmapped section of this region is aliased
0x1000_0000	160MB	Debug	See Self-hosted debug memory map
0x1A00_0000	32MB	Base peripherals	See Base peripherals memory map
0x1C00_0000	64KB	A5 Peripherals	See *Arm® Cortex®-A5 MPCore Technical Reference Manual*
0x1C01_0000	64KB	L2CC Peripherals	See *Arm® CoreLink™ Level 2 Cache Controller L2C-310 Technical Reference Manual*
0x1C02_0000	575MB	Reserved	-
0x4000_0000	512MB	Master expansion port 0	Used in the integration layer for connecting external peripherals. See Integration layer memory map.
0x6000_0000	512MB	Master expansion port 1	-
0x8000_0000	2GB	DDR	-

A.1.1.1 Self-hosted debug memory map

The self-hosted debug memory map describes the offset of the subsystem debug components relative to the Debug region base address (0x1000_0000).

Table 4: Self hosted debug memory map

Offset	Size	Component	Notes
0x1000_0000	64KB	System Debug ROM	See *ARM® CoreSight SoC-400 Technical Reference Manual*
0x1001_0000	64KB	Replicator	
0x1002_0000	64KB	TPIU	
0x1003_0000	64KB	ETB	
0x1004_0000	64KB	System CTI	
0x1005_0000	704KB	Reserved	-
0x1010_0000	128KB	CPU Debug	See CPU debug memory map
0x1012_0000	896KB	Reserved	-
0x1020_0000	158MB	Reserved	-

A.1.1.1.1 CPU debug memory map

The CPU debug memory map lists the address offset of the CPU debug components relative to the CPU Debug region base address in:

- Table 4: Self hosted debug memory map on page 24 (0x1010_0000).
- Table 7: Debug memory map on page 26 (0x8010_0000).

Table 5: CPU debug memory map

Offset	Size	Component	Notes
0x0_0000	4KB	Cortex-A5 PIL ROM table	-
0x0_1000	60KB	Reserved	-
0x1_0000	4KB	Processor0 Debug	-
0x1_1000	4KB	Processor0 PMU	-
0x1_2000	4KB	Processor1 Debug	Implemented if using 2 or more Cortex-A5 cores. Otherwise reserved.
0x1_3000	4KB	Processor1 PMU	
0x1_4000	4KB	Processor2 Debug	Implemented if using 3 or more Cortex-A5 cores. Otherwise reserved.
0x1_5000	4KB	Processor2 PMU	

Offset	Size	Component	Notes
0x1_6000	4KB	Processor3 Debug	Implemented if using 4 or more Cortex-A5 cores. Otherwise reserved.
0x1_7000	4KB	Processor3 PMU	
0x1_8000	4KB	CTI0	-
0x1_9000	4KB	CTI1	Implemented if using 2 or more Cortex-A5 cores. Otherwise reserved.
0x1_A000	4KB	CTI2	Implemented if using 3 or more Cortex-A5 cores. Otherwise reserved.
0x1_B000	4KB	CTI3	Implemented if using 4 or more Cortex-A5 cores. Otherwise reserved.
0x1_C000	4KB	ETM0	-
0x1_D000	4KB	ETM1	Implemented if using 2 or more Cortex-A5 cores. Otherwise reserved.
0x1_E000	4KB	ETM2	Implemented if using 3 or more Cortex-A5 cores. Otherwise reserved.
0x1_F000	4KB	ETM3	Implemented if using 4 or more Cortex-A5 cores. Otherwise reserved.

A.1.1.2 Base peripherals memory map

The base peripherals memory map describes the base address and region information for the peripherals.

Table 6: Base peripherals memory map

Base address	Size	Component	Notes
0x1A00_0000	64KB	System ID	See System ID registers
0x1A01_0000	64KB	System control	See System control registers
0x1A02_0000	64KB	Generic counter control	See *Arm® Server Base System Architecture 5.0 Platform Design Document*
0x1A03_0000	64KB	Generic counter read	
0x1A04_0000	64KB	Generic timer control	
0x1A05_0000	64KB	Generic timer 0	
0x1A06_0000	640KB	Reserved	-
0x1A10_0000	64KB	Watchdog control	See *Arm® Server Base System Architecture 5.0 Platform Design Document*
0x1A11_0000	64KB	Watchdog refresh	
0x1A12_0000	896KB	Reserved	-

Base address	Size	Component	Notes
0x1A20_0000	64KB	UART0	See *ARM® PrimeCell UART (PL011) Technical Reference Manual*
0x1A21_0000	64KB	UART1	
0x1A22_0000	64KB	RTC	See *ARM® PrimeCell Real Time Clock (PL031) Technical Reference Manual*
0x1A23_0000	64KB	TRNG	See *ARM TrustZone True Random Number Generator Technical Reference Manual*
0x1A24_0000	29MB	Reserved	-

A.1.1.3 External debug memory map

When the debug subsystem is accessed by an external debugger, bit 31 of the APB address bus must be set high. This bit is used by the SoC-400 components to distinguish between external debug access and self-hosted debug access. Therefore, the external debugger memory map starts at base address 0x8000_0000.

Table 7: Debug memory map

Base address	Size	Component	Notes
0x8000_0000	64KB	System Debug ROM	See *ARM® CoreSight SoC-400 Technical Reference Manual*
0x8001_0000	64KB	Replicator	
0x8002_0000	64KB	TPIU	
0x8003_0000	64KB	ETB	
0x8004_0000	64KB	System CTI	
0x8005_0000	704KB	Reserved	
0x8010_0000	128KB	CPU Debug	See CPU debug memory map
0x8012_0000	896KB	Reserved	-
0x8020_0000	158MB	Reserved	-

A.2 Integration layer memory map

The integration layer uses one expansion master port to connect to the NIC-400. The integration layer memory map contains memory information for all the peripherals in the example integration layer.

Table 8: Integration layer memory map

Base address	Size	Component	Notes
0x4000_0000	64KB	UART2	See *ARM® PrimeCell UART (PL011) Technical Reference Manual*
0x4001_0000	64KB	UART3	
0x4002_0000	64KB	UART4	
0x4003_0000	64KB	UART5	
0x4004_0000	64KB	GPIO0	See *ARM® PrimeCell General Purpose Input/Output (PL061) Technical Reference Manual*
0x4005_0000	64KB	GPIO1	
0x4006_0000	64KB	GPIO2	
0x4007_0000	64KB	GPIO3	
0x4008_0000	64KB	GPIO4	
0x4009_0000	64KB	GPIO5	
0x400A_0000	64KB	GPIO6	
0x400B_0000	64KB	GPIO7	
0x400C_0000	256KB	Reserved	-
0x4010_0000	64KB	I2S0	See I2S
0x4011_0000	64KB	I2S1	
0x4012_0000	64KB	I2C0	See I2C
0x4013_0000	64KB	I2C1	
0x4014_0000	64KB	I2C2	
0x4015_0000	64KB	I2C3	
0x4016_0000	64KB	I2C4	
0x4017_0000	64KB	SPI0	See *ARM® PrimeCell Synchronous Serial Port (PL022) Technical Reference Manual*
0x4018_0000	64KB	SPI1	
0x4019_0000	64KB	SPI2	
0x401A_0000	64KB	CLCD Config Reg	See CLCD interface
0x401B_0000	320KB	Reserved	-
0x4020_0000	1MB	Ethernet	See Ethernet and USB
0x4030_0000	1MB	USB	

A.3 Register Descriptions

The registers described in this section complement those described in the *ARM Architecture Reference Manual ARMv7-A and ARMv7-R*edition edition. You will also need to refer to the technical reference manuals of the included Arm IP and any model you want to use.

A.3.1 System ID registers

System ID registers describe system configuration and identification information.

Table 9: System ID register summary

Offset	Name	Type	Reset	Width	Description
0x000	Reserved	RAZ/WI	-	-	Reserved
0x004	Reserved	RAZ/WI	-	-	Reserved
0x008	SYS_CFG2	RO	0x0022_0000	32-bit	System configuration 2
0x040	SOC_ID	RO	CFG_DEF	32-bit	SoC identification
0xFC8	IIDR	RO	IMPDEF	32-bit	Implementer identification register
0xFD0	CNTPIDR4	RO	0x04	32-bit	Peripheral ID 4
0xFD4	Reserved	**RAZ/WI**	-	-	-
0xFD8	Reserved	**RAZ/WI**	-	-	-
0xFDC	Reserved	**RAZ/WI**	-	-	-
0xFD4	Reserved	**RAZ/WI**	-	-	-
0xFE0	CNTPIDR0	RO	0xB0	32-bit	Peripheral ID 0
0xFE4	CNTPIDR1	RO	0xB8	32-bit	Peripheral ID 1
0xFE8	CNTPIDR2	RO	0x0B	32-bit	Peripheral ID 2
0xFEC	CNTPIDR3	RO	0x00	32-bit	Peripheral ID 3
0xFF0	CNTCIDR0	RO	0x0D	32-bit	Component ID 0
0xFF4	CNTCIDR1	RO	0xF0	32-bit	Component ID 1
0xFF8	CNTCIDR2	RO	0x05	32-bit	Component ID 2
0xFFC	CNTCIDR3	RO	0xB1	32-bit	Component ID 3

A.3.1.1 SYS_CFG2, System Configuration 2

SYS_CFG2 contains read-only information about the number of expansion master interfaces and slave interfaces.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Functional group

System ID registers

Address offset

0x0000_0008

Bit descriptions

Table 10: SYS_CFG2 bit descriptions

Bits	Types	Reset	Name	Description
31:24	RO	0x00	RESERVED	Reserved
23:20	RO	0x2	NUM_EXP_MST	Number of expansion master interfaces, 0x2 = 2 expansion master interfaces. All other values are reserved.
19:16	RO	0x2	NUM_EXP_SLV	Number of expansion slave interfaces, 0x2 = 2 expansion slave interfaces. All other values are reserved.
15:0	RO	0x0000	RESERVED	Reserved

A.3.1.2 SOC_ID, SoC identification

SOC_ID contains information to identify a SoC that has been derived from the CA5DS subsystem, and the implementer of the SoC. The value of this register is driven by incoming signals that are external to the subsystem.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Functional group

System ID registers

Address offset

0x0000_0040

Bit descriptions

Table 11: SOC_ID bit descriptions

Bits	Types	Reset	Name	Description
31:20	RO	CFG_DEF	PRODUCT_ID	SoC product identification.
19:16	RO	CFG_DEF	VARIANT	Variant of SoC.
15:12	RO	CFG_DEF	REVISION	Minor Revision of SoC.
11:0	RO	CFG_DEF	IMPLEMENTER	JEP106 Code of company implementing SoC.

A.3.1.3 IIDR, Implementer identification

IIDR contains information to identify the implementer of the reference design architecture.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Functional group

System ID registers

Address offset

0x0000_0FC8

Bit descriptions

Table 12: IIDR bit descriptions

Bits	Types	Reset	Name	Description
31:20	RO	0x7F0	PRODUCT_ID	Subsystem product identification
19:16	RO	0x0	VARIANT	Variant of the subsystem
15:12	RO	0x0	REVISION	Revision of the subsystem
11:0	RO	0x43B	IMPLEMENTER	JEP106 Code of company implementing the subsystem • [11:8] - JEP106 Continuation Code • [7] - Always 0 • [6:0] - JEP106 identify code

A.3.2 System control registers

The system control registers describe control information in the subsystem.

Table 13: System control register summary

Offset	Name	Type	Reset	Width	Description
0x000	RESERVED	RW	-	32-bit	Reserved
0x010	PE0_CONFIG	RW	0x0000_0000	32-bit	Processing Element 0 Static Config
0x020	PE1_CONFIG	RW	0x0000_0000	32-bit	Processing Element 1 Static Config
0x030	PE2_CONFIG	RW	0x0000_0000	32-bit	Processing Element 2 Static Config
0x040	PE3_CONFIG	RW	0x0000_0000	32-bit	Processing Element 3 Static Config

Offset	Name	Type	Reset	Width	Description
0x200	SYS_RST_SYN	RO	See individual bit resets	32-bit	System reset syndrome
0x330	SYS_RST_CTL	WO	0x0000_0000	32-bit	System reset control
0xFD0	CNTPIDR4	RO	0x04	32-bit	Peripheral ID 4
0xFD4	Reserved	**RAZ/WI**	-	-	-
0xFD8	Reserved	**RAZ/WI**	-	-	-
0xFDC	Reserved	**RAZ/WI**	-	-	-
0xFD4	Reserved	**RAZ/WI**	-	-	-
0xFE0	CNTPIDR0	RO	0xB1	32-bit	Peripheral ID 0
0xFE4	CNTPIDR1	RO	0xB8	32-bit	Peripheral ID 1
0xFE8	CNTPIDR2	RO	0x0B	32-bit	Peripheral ID 2
0xFEC	CNTPIDR3	RO	0x00	32-bit	Peripheral ID 3
0xFF0	CNTCIDR0	RO	0x0D	32-bit	Component ID 0
0xFF4	CNTCIDR1	RO	0xF0	32-bit	Component ID 1
0xFF8	CNTCIDR2	RO	0x05	32-bit	Component ID 2
0xFFC	CNTCIDR3	RO	0xB1	32-bit	Component ID 3

A.3.2.1 PE0_CONFIG, Processing Element 0 Static Config

PE0_CONFIG is used to control the configuration options of the Processing Element or core 0 of the processor cluster.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Functional group

System control registers

Address offset

0x0000_0010

Bit descriptions

Table 14: PE0_CONFIG bit descriptions

Bits	Types	Reset	Name	Description
31:3	RO	0x000_0000	RESERVED	Reserved
2	RW	1'b0	VINITHI	Location of the exception vectors at reset. • 1'b0 - Vector starts at 0x0000_0000 • 1'b1 - Vector starts at 0xFFFF_0000
1	RW	1'b0	TEINIT	Enabling Thumb® exceptions. • 1'b0 - Exceptions taken in Arm(v7) state • 1'b1 - Exceptions taken in Thumb state
0	RW	1'b0	CFGEND	Endianness configuration. • 1'b0 - Little Endian • 1'b1 - Big Endian

A.3.2.2 PE1_CONFIG, Processing Element 1 Static Config

PE1_CONFIG is used to control the configuration options of the Processing Element or core 1 of the processor cluster.

Configurations

This register is available in configurations with more than one core.

Attributes

Width

32-bit

Functional group

System control registers

Address offset

0x0000_0020

Bit descriptions

Table 15: PE1_CONFIG bit descriptions

Bits	Types	Reset	Name	Description
31:3	RO	0x000_0000	RESERVED	Reserved
2	RW	1'b0	VINITHI	Location of the exception vectors at reset. • 1'b0 - Vector starts at 0x0000_0000 • 1'b1 - Vector starts at 0xFFFF_0000
1	RW	1'b0	TEINIT	Enabling Thumb® exceptions. • 1'b0 - Exceptions taken in Arm(v7) state • 1'b1 - Exceptions taken in Thumb state
0	RW	1'b0	CFGEND	Endianness configuration. • 1'b0 - Little Endian • 1'b1 - Big Endian

A.3.2.3 PE2_CONFIG, Processing Element 2 Static Config

PE2_CONFIG is used to control the configuration options of the Processing Element or core 2 of the processor cluster.

Configurations

This register is available in configurations with more than two cores.

Attributes

Width

32-bit

Functional group

System control registers

Address offset

0x0000_0030

Bit descriptions

Table 16: PE2_CONFIG bit descriptions

Bits	Types	Reset	Name	Description
31:3	RO	0x000_0000	RESERVED	Reserved
2	RW	1'b0	VINITHI	Location of the exception vectors at reset. • 1'b0 - Vector starts at 0x0000_0000 • 1'b1 - Vector starts at 0xFFFF_0000
1	RW	1'b0	TEINIT	Enabling Thumb® exceptions. • 1'b0 - Exceptions taken in Arm(v7) state • 1'b1 - Exceptions taken in Thumb state
0	RW	1'b0	CFGEND	Endianness configuration. • 1'b0 - Little Endian • 1'b1 - Big Endian

A.3.2.4 PE3_CONFIG, Processing Element 3 Static Config

Processing element static configuration register is used to control the configuration options of the Processing Element or core 3 of the processor cluster.

Configurations

This register is available in configurations with more than three cores.

Attributes

Width

32-bit

Functional group

System control registers

Address offset

0x0000_0040

Bit descriptions

Table 17: PE3_CONFIG bit descriptions

Bits	Types	Reset	Name	Description
31:3	RO	0x000_0000	RESERVED	Reserved
2	RW	1'b0	VINITHI	Location of the exception vectors at reset. • 1'b0 - Vector starts at 0x0000_0000 • 1'b1 - Vector starts at 0xFFFF_0000
1	RW	1'b0	TEINIT	Enabling Thumb® exceptions. • 1'b0 - Exceptions taken in Arm(v7) state • 1'b1 - Exceptions taken in Thumb state
0	RW	1'b0	CFGEND	Endianness configuration. • 1'b0 - Little Endian • 1'b1 - Big Endian

A.3.2.5 SYS_RST_SYN, System Reset Syndrome

SYS_RST_SYN stores information on the cause of most recent reset of the system.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Functional group

System control registers

Address offset

0x0000_0200

Bit descriptions

Table 18: SYS_RST_SYN bit descriptions

Bits	Types	Default	Name	Description
31:4	RO	0x000_0000	RESERVED	Reserved
3	RO	1'b0	SYSRSTREQ	When HIGH, indicates that the last reset was caused by SW programmed SYSRSTREQ being asserted.
2	RO	1'b0	nSRST	When HIGH, indicates that the last reset was caused by nSRST pin being asserted.
1	RO	1'b0	SYSWDIRQ	When HIGH, indicates that the last reset was caused by system watchdog second expiry signal being asserted.
0	RO	1'b1	PORESETn	When HIGH, indicates that the last reset was caused by PORESETn pin being asserted

A.3.2.6 SYS_RST_CTL, System Reset Control

SYS_RST_CTL provides a software programmable mechanism to reset the system.

Configurations

This register is available in all configurations.

Attributes

Width

32-bit

Functional group

System control registers

Address offset

0x0000_0330

Bit descriptions

Table 19: SYS_RST_CTL bit descriptions

Bits	Types	Reset	Name	Description
31:2	RO	0x000_0000	RESERVED	Reserved
1	RW	1'b0	RST_REQ	System reset request. • 1'b0 - No reset requested. • 1'b1 - Reset requested.
0	RO	1'b0	RESERVED	Reserved

A.3.3 Integration layer registers

The integration layer registers describe information for peripherals in the example integration layer.

Table 20: Integration layer register summary

Name	Type	Reset	Width	Description
I2C	RO, WO	-	-	Inter-Integrated Circuit controller
I2S	RO, WO, RW, RAZ/WI	-	-	Inter-IC Sound Bus Controller
CLCD Interface	RW	-	-	CLCD Interface

A.3.3.1 I2C, Inter-Integrated Circuit controller

A serial bus module is provided to enable the system to use an APB interface to communicate with the I^2C interface.

Table 21: I2C controller bit descriptions

Offset	Name	Type	Reset	Width	Description
0x000	SB_CONTROL	RO			Read serial control bits: • Bit [0] is SCL • Bit [1] is SDA
0x000	SB_CONTROLS	WO			Set serial control bits: • Bit [0] is SCL • Bit [1] is SDA

Offset	Name	Type	Reset	Width	Description
0x004	SB_CONTROLC	WO			Clear serial control bits: • Bit [0] is SCL • Bit [1] is SDA

A.3.3.2 I2S, Inter-IC Sound Bus Controller

An I^2S controller module is provided to enable the system to use an APB interface to communicate with the I^2S interface.

Table 22: I2S controller register summary

Offset	Name	Type	Reset	Width	Description
0x000	CONTROL	RW	-	-	Control Register
0x004	STATUS	RO	-	-	Status Register
0x008	ERROR	RW	-	-	Error Status Register
0x00C	DIVIDE	RW	-	-	Clock Divide ratio register
0x010	TXBUF	WO	-	-	Transmit Buffer FIFO Data Register
0x014	RXBUF	RO	-	-	Receive Buffer FIFO Data Register
0x300	ITCR	RW	-	-	Integration Test Control Register
0x304	ITIP1	RO	-	-	Integration Test Input Register 1
0x308	ITOP1	RW	-	-	Integration Test Output Register 1
0xFD0	CNTPIDR4	RO	-	-	Peripheral ID 4
0xFD4	-	RAZ/WI	-	-	Reserved
0xFD8	-	RAZ/WI	-	-	Reserved
0xFDC	-	RAZ/WI	-	-	Reserved
0xFD4	-	RAZ/WI	-	-	Reserved
0xFE0	CNTPIDR0	RO	-	-	Peripheral ID 0
0xFE4	CNTPIDR1	RO	-	-	Peripheral ID 1
0xFE8	CNTPIDR2	RO	-	-	Peripheral ID 2
0xFEC	CNTPIDR3	RO	-	-	Peripheral ID 3
0xFF0	CNTCIDR0	RO	-	-	Component ID 0

Offset	Name	Type	Reset	Width	Description
0xFF4	CNTCIDR1	RO	-	-	Component ID 1
0xFF8	CNTCIDR2	RO	-	-	Component ID 2
0xFFC	CNTCIDR3	RO	-	-	Component ID 3

A.3.3.3 CLCD Interface

The Color LCD (CLCD) interface is a custom peripheral that provides an interface to a STMicroelectronics STMPE811QTR Port Expander with Advanced Touch Screen Controller on the Keil® MCBSTM32C display board. The Keil display board contains an AM240320LG display panel and uses a Himax HX8347-D LCD controller.

Table 23: CLCD interface

Offset	Type	Name	Description
0x000	RW	Write Command	A write to this address will cause a write to the LCD commands register. A read from this address will cause a read from the LCD busy register.
0x004	RW	Write data RAM, Read data RAM	A write to this address will cause a write to the LCD data register. A read to this address will cause a read from the LCD data register.
0x008	RW	Interrupt	Bit 0 indicates Access Complete. Write 0 to Bit 0 to clear. This bit is set if read data is valid. Bits [31:1] should be ignored.

A.3.3.4 Ethernet and USB

An AHB static memory controller has been provided for the system to communicate to an ethernet controller device (SMSC LAN9220) and a Hi-Speed USB OTG controller device (ISP1763) via two static memory interface correspondingly.

A.4 Interrupt map

The reference design uses hardware and software interrupts.

The following types of interrupts are supported:

Software Generated Interrupts (SGIs)

This interrupt is generated explicitly by software by writing to a dedicated distributor register, the Software Generated Interrupt Register.

Private Peripheral Interrupts (PPIs)

This interrupt is generated by a peripheral that is private to an individual core.

Shared Peripheral Interrupts (SPIs)

This interrupt is generated by a peripheral that the Interrupt Controller can route to more than one core.

A.4.1 Subsystem interrupt map

The following table describes the interrupts in the subsystem.

Table 24: Subsystem interrupt map

Interrupt ID	Description	Type	Edge or Level	Notes
0-15	Software generated	SGI	-	-
16-26	Reserved	PPI	-	-
27	Global timer	PPI	Edge	-
28	Legacy nFIQ	PPI	Level	-
29	Private timer	PPI	Edge	-
30	Watchdog timers	PPI	Edge	-
31	Legacy nIRQ	PPI	Level	-
32	System watchdog	SPI	Level	-
33	Reserved	SPI	-	-
34	System timer - physical	SPI	Level	-
35-37	Reserved	SPI	-	-
38	Real-time clock	SPI	Level	-
39	True random number generator	SPI	Level	-
40	UART0 combined	SPI	Level	-
41	UART1 combined	SPI	Level	-
42	Debug Interrupt 0	SPI	Level	-
43	Debug Interrupt 1	SPI	Level	-
44	L2CC combined interrupt	SPI	Level	-
45	CPU0 debug comm TX	SPI	Level	-

Interrupt ID	Description	Type	Edge or Level	Notes
46	CPU0 debug comm RX	SPI	Level	-
47	CPU0 PMU counter overflow	SPI	Level	-
48	CPU0 CTI trigger	SPI	Level	-
49	CPU1 debug comm TX	SPI	Level	Implemented if using 2 or more Cortex-A5 cores. Otherwise reserved.
50	CPU1 debug comm RX	SPI	Level	
51	CPU1 PMU counter overflow	SPI	Level	
52	CPU1 CTI trigger	SPI	Level	
53	CPU2 debug comm TX	SPI	Level	Implemented if using 3 or more Cortex-A5 cores. Otherwise reserved.
54	CPU2 debug comm RX	SPI	Level	
55	CPU2 PMU counter overflow	SPI	Level	
56	CPU2 CTI trigger	SPI	Level	
57	CPU3 debug comm TX	SPI	Level	Implemented if using 4 or more Cortex-A5 cores. Otherwise reserved.
58	CPU3 debug comm RX	SPI	Level	
59	CPU3 PMU counter overflow	SPI	Level	
60	CPU3 CTI trigger	SPI	Level	
61-63	Reserved	SPI	-	-

A.4.2 Example integration layer interrupt map

The following table describes the interrupts in the integration layer.

Table 25: Integration layer interrupt map

ID	Description	Type	Trigger	Notes
64	UART2 combined	SPI	Level	-
65	UART3 combined	SPI	Level	-
66	UART4 combined	SPI	Level	-
67	UART5 combined	SPI	Level	-
68	I2S0	SPI	Level	-
69	I2S1	SPI	Level	-
70	SPI0 combined	SPI	Level	-
71	SPI1 combined	SPI	Level	-
72	SPI2 combined	SPI	Level	-
73	CLCD	SPI	Level	-

ID	Description	Type	Trigger	Notes
74	USB	SPI	Level	-
75	Ethernet	SPI	Level	-
76	GPIO0 combined	SPI	Level	-
77	GPIO1 combined	SPI	Level	-
78	GPIO2 combined	SPI	Level	-
79	GPIO3 combined	SPI	Level	-
80	GPIO4 combined	SPI	Level	-
81	GPIO5 combined	SPI	Level	-
82	GPIO6 combined	SPI	Level	-
83	GPIO7 combined	SPI	Level	-
84	FPGA_generic	SPI	Level	-
85-255	Reserved	SPI		-

Appendix B Revisions

B.1 Revisions

This appendix describes changes between released issues of this book.

Table 26: Issue 0100-00

Change	Location	Affects
First release	-	-