CIX P1

Technical Reference Manual

V1.0

May 20, 2026

Legal Notice

Copyright © 2026 CIX Technology Group Co., LTD. All rights reserved.
This document is the property of CIX Technology Group Co., Ltd. This document may only be distributed to (i) a CIX Technology Group party having a legitimate business need for the information contained herein, or (ii) a non-CIX Technology Group party having a legitimate business need for the information contained herein. No license, expressed or implied, under any patent, copyright or trade secret right is granted or implied by the conveyance of this document. No part of this document may be reproduced, transmitted, transcribed, stored in a retrieval system, translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise without the prior written permission of CIX Technology Group Co., Ltd.

Trademarks and Permissions
The CIX Technology Group Logo and all other trademarks indicated as such herein are trademarks of CIX Technology Group Co., Ltd.
PCI Express and PCIe are registered trademarks of PCI-SIG.
Radxa and Orion O6 are trademarks of Radxa Limited.
Other product names used in this publication are for description purposes only and may be trademarks of their respective companies.

Notice
The purchased products, services and features are stipulated by the contract made between CIX Technology and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.
The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure the accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Revision History

Revision	Date	Description
V1.0	May 2026	Initial release

Contents

Revision History i

Acronyms and Abbreviations viii

Chapter 1 System Overview 1-1

1.1 Features 1-1
1.2 Architectural Overview / Block Diagram 1-2
1.3 Memory Map 1-4
1.4 Interrupt Map 1-8
1.5 DMA Map 1-43

Chapter 2 CPU 2-1

2.1 Overview 2-1
2.2 Block Diagram 2-2
2.3 Function Description 2-2

Chapter 3 GPU 3-1

3.1 Overview 3-1
3.1.1 Features 3-1
3.1.2 Standard 3-1
3.2 Function Description 3-2

Chapter 4 VPU 4-1

4.1 Overview 4-1
4.1.1 Video Encoding 4-1
4.1.2 Video Decoding 4-4

Chapter 5 DPU 5-1

5.1 Function Description 5-1
5.1.1 Display Layers 5-1
5.1.2 Display Layer Resolutions 5-1
5.1.3 Pixel Format Summary 5-2
5.1.4 AFBC Decoder 5-5
5.1.5 AFBC Encoder 5-6
5.1.6 Memory Pointers, Stride, and Cropping 5-6
5.1.7 Memory Write-Back 5-7
5.1.8 Rotation, Horizontal and Vertical Flips 5-7
5.1.9 Quality of Service 5-9

5.1.10 Color Space Conversions and Gamma Transformation 5-9
5.1.11 Supported Data Flows 5-10
5.1.12 Multi-layer composition 5-13
5.1.13 Multi-Layer Composition in Tandem 5-13
5.1.14 Side-By-Side Mode 5-14
5.1.15 Image Merge Mode 5-14
5.1.16 Scaling and Image Enhancement 5-15
5.1.17 Software Layer Split 5-16
5.1.18 Display Formats 5-17
5.1.19 Single Display Link (Non-Split Mode) 5-20
5.1.20 Single Link Image Merge Processing 5-20
5.1.21 Dithering 5-21
5.1.22 CRC Signature Generation 5-22
5.1.23 Display Security 5-22

Chapter 6 SMMU 6-1

6.1 Overview 6-1
6.1.1 SMMU PCIE Subsystem 6-2
6.1.2 SMMU MM Subsystem 6-2
6.1.3 SMMU Syshub Subsystem 6-3

Chapter 7 PCIe 7-1

7.1 Overview 7-1
7.2 Block Diagram 7-1
7.3 PCIe Subsystem Features 7-2
7.4 PCIe Gen4 Controller 7-2
7.4.1 Architectural Block Diagram 7-2
7.4.2 Controller PCS 7-3
7.4.3 PHY Logical Layer 7-4
7.4.4 Data Link Layer 7-5
7.4.5 Transaction Layer 7-6
7.4.6 Host Adaptation Layer Operation 7-7
7.4.7 Application Layer 7-7
7.5 PHY Overview 7-8
7.5.1 Features 7-8
7.5.2 PMA Description 7-9
7.5.3 Common Module 7-10
7.5.4 Lane Module 7-12
7.5.5 Analog Test Bus (ATB) 7-18
7.5.6 Link Power States 7-34

Chapter 8 USB 8-1

8.1 Features 8-1
8.2 Block Diagram 8-2

8.2.1 Brief Function Description 8-3
8.3 Address Mapping 8-3
8.4 USB3.2 Gen2 Dual-Role Controller 8-4
8.4.1 Overview 8-4
8.4.2 Architectural Specification 8-5
8.5 USB 2.0 PHY 8-7
8.5.1 Overview 8-7
8.5.2 Features 8-7
8.5.3 Architecture Overview 8-8
8.6 USB3 Type-C PHY 8-9
8.6.1 PHY Architecture 8-10
8.6.2 Top-Level Architecture 8-10
8.6.3 PHY Pin List 8-11
8.7 USB3 PHY 8-12
8.7.1 PHY Architecture 8-13
8.7.2 Top Level Architecture 8-13
8.7.3 PHY Pin List 8-13

Chapter 9 FCH 9-1

9.1 Overview 9-1
9.2 Key Features 9-1
9.3 Block Diagram 9-1
9.4 I2C 9-2
9.4.1 Architecture Specification 9-2
9.5 I3C 9-3
9.5.1 Functional Description 9-3
9.6 XSPI 9-3
9.6.1 Introduction 9-3
9.6.2 Functional Description 9-4
9.7 Timer 9-5
9.7.1 Feature Summary 9-5
9.7.2 Simplified Block Diagram 9-6
9.8 DMA 9-6
9.8.1 Overview 9-6
9.9 UART 9-10
9.9.1 About the UART 9-10
9.9.2 Product Revisions 9-11

Chapter 10 DDR 10-1

10.1 Function Description 10-1
10.2 Block Diagram 10-1
10.3 Features 10-1

Chapter 11 DP 11-1

11.1 Supported Feature List 11-1

Chapter 12 MIPI 12-1

12.1 Overview 12-1
12.1.1 Block Diagram 12-1
12.1.2 Features 12-2
12.2 CSI2RX Controller 12-3
12.3 CSI Bridge 12-4
12.3.1 Overview 12-4
12.3.2 Block Diagram 12-4
12.4 PHY 12-5
12.4.1 Features 12-5

Chapter 13 Debug 13-1

13.1 Overview 13-1
13.1.1 Features 13-1
13.1.2 Debug components address map 13-1
13.2 Block Diagram 13-1
13.3 Function Description 13-1
13.3.1 DAP 13-1
13.4 Interface Description 13-2
13.4.1 Physical Connection to the JTAG-DP 13-2
13.5 Register Description 13-3

List of Figures

Figure 1-1 P1 Block Diagram 1-3
Figure 2-1 Block Diagram 2-2
Figure 6-1 SMMU Block Diagram 6-1
Figure 6-2 SMMU PCIE Subsystem 6-2
Figure 6-3 SMMU MM Subsystem 6-2
Figure 6-4 SMMU Syshub Subsystem 6-3
Figure 7-1 PCIe Block Diagram 7-1
Figure 7-2 PCIe controller IP architecture overview 7-3
Figure 7-3 Controller PCS block diagram 7-4
Figure 7-4 Physical Layer Block Diagram 7-5
Figure 7-5 PCIe PHY Block Diagram 7-9
Figure 7-6 SerDes Macro Block Diagram 7-10
Figure 7-7 Common Module Block Diagram 7-12
Figure 7-8 Lane Module Block Diagram 7-14
Figure 7-9 Transmit Path Block Diagram 7-16
Figure 7-10 Receive Path Block Diagram 7-18
Figure 7-11 ATB Bus Architecture 7-20
Figure 8-1 USB Subsystem Block Diagram 8-2
Figure 8-3 Basic USB2.0 Application Using UTMI+ PHY 8-7
Figure 8-4 Functional Block Diagram 8-9
Figure 8-5 USB Type C PHY 8-10
Figure 8-6 USB3 Type-C PHY Block Diagram 8-11
Figure 8-7 USB3 PHY 8-13
Figure 9-1 FCH Block Diagram 9-2
Figure 9-3 IP6522 XSPI Flash Controller with IP6182/IP6184 PHY Block Diagram 9-4
Figure 9-4 Timer and Main Data Paths 9-6
Figure 9-5 DMA-350 in a System 9-7
Figure 9-6 DMA-350 Top-Level Block Diagram 9-8
Figure 10-1 DDR Block Diagram 10-1
Figure 12-1 MIPI CSI Subsystem Block Diagram 12-1
Figure 12-2 CSI controller Block Diagram 12-4
Figure 12-3 MIPI CSI Bridge Block Diagram 12-5
Figure 13-1 Debug System Structure 13-1
Figure 13-2 DAP JTAG-DP Interface 13-2

List of Tables

Table 1-1 SOC Core Features 1-1
Table 1-2 Memory Map 1-4
Table 1-3 Interrupt Map 1-8
Table 2-1 CPU Configuration 2-2
Table 5-1 Pixel Formats Supported by the Afbc Decoder 5-5
Table 5-2 Afbc Ram Blocks for Default Configurations 5-8
Table 5-3 Summary of Permitted Data Flow Through the Composition Unit for the Default Configurations 5-12
Table 5-4 Allocation of Read Layers during Software Layer Split 5-17
Table 5-5 Bit Precision Reduction on the Output Of Dithering Block 5-21
Table 7-1 PCIe Subsystem Features 7-2
Table 7-2 Equalization Evaluation Interface Signals 7-17
Table 7-3 Common Component and Test Point Definitions 7-20
Table 7-4 Lane Component and Test Point Definitions 7-25
Table 7-5 Link Power States 7-34
Table 8-1 USB Address Mapping 8-3
Table 8-2 USB3 Type-C PHY External Pin / Bump List 8-11
Table 8-3 USB3 Type-C PHY External Pin / Bump List 8-12
Table 8-4 USB3 PHY Power Pin List 8-13
Table 8-5 USB3 PHY External Pin / Bump List 8-14
Table 10-1 Bank Organization 10-2
Table 10-2 Supported CK:WCK Ratio 10-2
Table 16-1External Debugger View 13-1
Table 13-2 JTAG-DP Signal Connections 13-2
Table 13-3 JTAG-DP Interface Description 13-2

Acronyms and Abbreviations

Acronym	Description
ACPI	Advanced Configuration and Power Interface
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
AXI	Advanced eXtensible Interface - a system bus protocol for very high data-rate transfers
DDR	Double Data Rate
DRAM	Dynamic RAM
DVD	Digital Video Disc
DVFS	Dynamic Voltage and Frequency Scaling
EP	(USB) End Point
FIFO	First-in, First-out - a memory component from which data is read in the same order it is written
GPIO	General Purpose Input/Output
I2C	Inter Integrated Circuit
I3C	Improved Inter Integrated Circuit
IP	Intellectual Property - specifically a HDL representation of a design or the design itself available as a reusable component
IPA	Intelligent Power Allocation
IPS	Integrated Protocol Stack - additional hierarchical design layers that include the associated PHY and memory macros
JTAG	Joint Test Access Group. An IEEE standard.
LSB	Least Significant Bit
MB	Mega Byte
MIPI	Mobile Industry Processor Interface
MPS	Max Packet Size
MSI	Message Signaled Interrupts. In-band method for sending interrupts over AXI or another bus protocol, rather than side-band interrupt pins.
MSB	Most Significant Bit
N/A	Not Applicable
PCI	Peripheral Component Interface
PCIe	PCI Express
PIPE	PHY Interface for the PCIe Express Architecture
PLL	Phase Locked Loop
POST	Power-On Self-Test
SLC	System Level Cache
SPI	Serial Peripheral Interface
SS	Super Speed (5Gb/s, USB3.2)
SSP	Super Speed Plus (10Gb/s, USB3.2)

Acronym	Description
TT	Transaction Translator
TWS	True Wireless Stereo
USB	Universal Serial Bus
UTMI	USB 2.0 Transceiver Macrocell Interface
VESA	Video Electronics Standards Association
WDT	Watchdog Timer
xHCI	eXtensible Host Controller Interface

Chapter 1 System Overview

P1 is CIX's first-generation 64-bit Armv9.2-based general-purpose SoC featuring a big.LITTLE CPU architecture, designed for applications such as PCs, automotive cockpits, and Arm-based cluster servers.
P1 is a System-on-Chip (SoC) integrating multiple subsystems, including the CPU, GPU, NPU, VPU, DPU, ISP, HSIO, and LSIO, onto a single chip.
P1 can be deployed in various application scenarios, from high-end computing to entry-level mobile devices, by using different SoC configurations.

1.1 Features

The soc supports numerous features, as listed in Table 1-1.

Table 1-1 SOC Core Features

Item	Descriptions
CPU Cluster	SupportArmv9.2-A A64 Instruction set 4 Hunter Big Core/4 Hunter Middle Core/4 Hayes core DSU, Include L3 Cache, Support data coherency
GPU Cluster	Support Avalon architecture Support OpenCL 3.0, OpenGL ES 1.1,2.0,3.1 Support Graphic Compute Acceleration Support 2D Graphic Support 3D Graphic Support General Purpose Computing on GPU Support 10 GPU cores
NPU	Support General-purpose computing Support AI-Specific processing capability with a different processing unit Support scalar Process Unit Support Vector/Tensor Processing Engine Support AI Fixed Function Accelerator Support FPU
VPU	Support 8-bit and 10-bit Support post-processing and pre-processing Support H.265/H.264/VP8/VP9 encoding Support AV1/H.265/H.264/H.263/AVSMPEG-4/MPEG-2/VP8/VP9/JPEG/MPEG2 Decoding
DPU	Support 1PPC and 2PPC (Pixels Per Cycle) modes Support 2 display pipelines with 4 layers for each pipe Support max display mode 4Kp120Hz Support RGB/ARGB/XRGB/YUV input pixel formats Support AFBC decompression support RGB <-> YUV color conversions(CSC) Support Gamma and Degamma

Item	Descriptions
	Support 90/180/270 degrees Rotation and H/V Flips Support maximum downscaling 1/6 in horizontal and 1/6 in vertical Support maximum upscaling 64 times in horizontal and 64 times in vertical Support pixel write-back for each pipe Support security display
DDR	Support LPDDR5 and LPDDR4X LPDDR4X supports 8 Bank mode, LPDDR supports 16 Bank and BG mode Support 2 Channel or 4 Channel Support 6400Mb/s
PCIE	Support PCIE Gen4 Support RC Mode Support 1 X8 Link Support 1 X4 Link Support 1 X2 Link Support 2 X1 Link
USB	Support 4 USB 2.0 Support 3 USB 3.2(comb DP 1.4) Support 2 USB3.2 Type A DRD
MIPI	Support 4 CSI
DP	Support 3 DP1.4(comb USB3.1) Support 1 DP/eDP 1.4Tx
I2C	Support 13 I2C
SPI	Support SPI/QSPI
I3C	Support 2 I3C
UART	Support 5 UART
GPIO	Support 16~64 GPIO
Audio	Support HDA/I2S-SC/I2S_MC Support 32bit 5-stage HIFI5
Security Management	Support TrustZone Media Protection (TZMP)
Power Management	Support fine-grained and intelligent power management
Debug Management	Support JTAG Debug

1.2 Architectural Overview / Block Diagram

Figure 1-1 shows the overall architecture of P1. P1 includes the CPU cluster, GPU cluster, multimedia subsystem, data fabric, LPDDR5 subsystem, PCIe subsystem, I/O subsystem, system control subsystem, and debug subsystem.

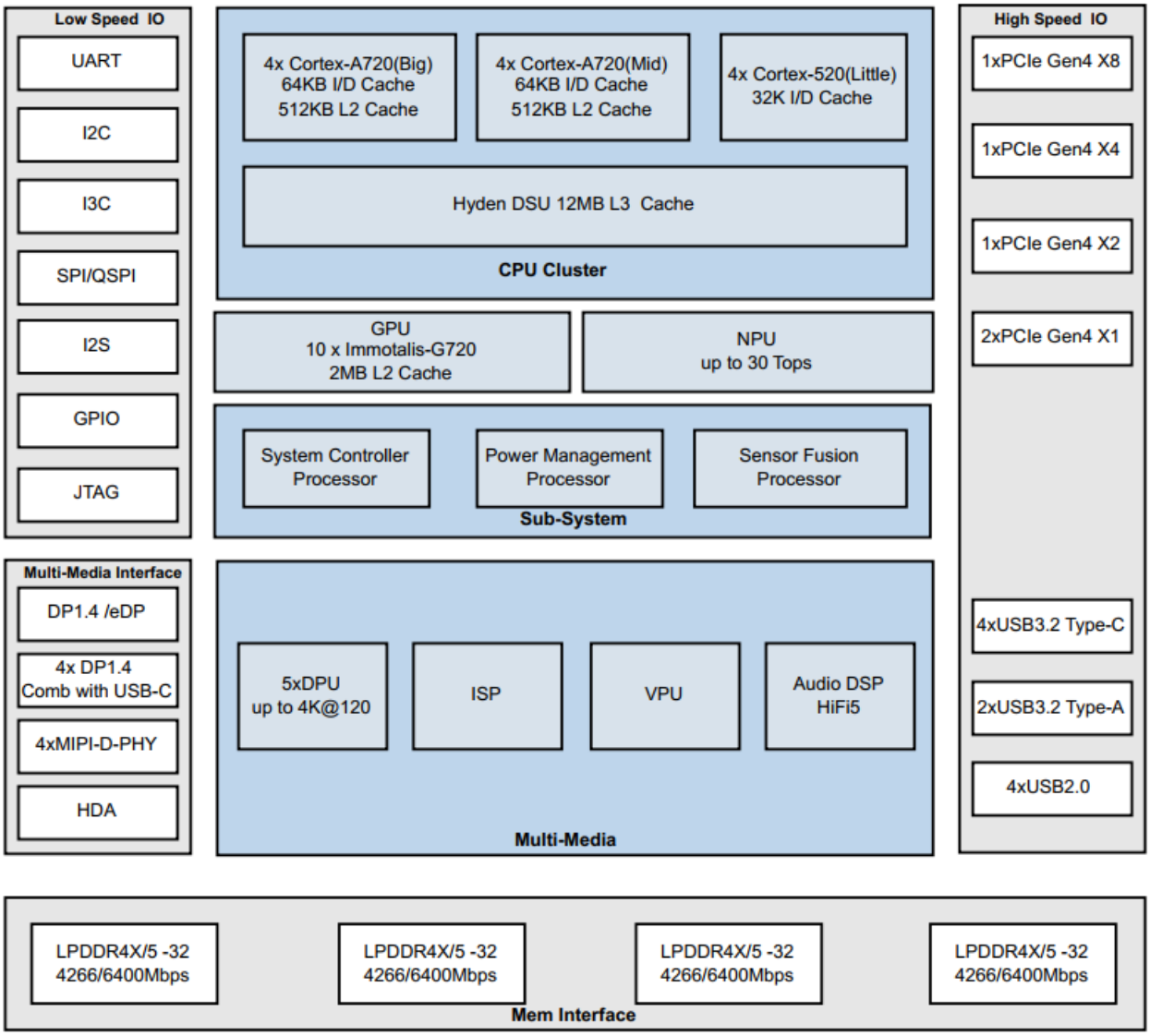


Figure 1-1 P1 Block Diagram

The CPU cluster consists of:

- Four big cores, each with 64 KB instruction cache (I-cache), 64 KB data cache (D-cache), and 512 KB L2 cache;
- Four middle cores, each with 64 KB I-cache, 64 KB D-cache, and 512 KB L2 cache;
- Four little cores, each with 32 KB I-cache and 32 KB D-cache.
- The Hayden DynamIQ Unit, which provides a 12 MB L3 cache, snoop control, and filtering functions.

The GPU cluster consists of 10 G720 GPU cores and four L2 caches, each with a capacity of 512 KB. The multimedia subsystem includes the VPU, DPU, ISP, and NPU modules.
The data fabric supports data coherency and includes a system-level cache. It supports the AMBA 5 CHI protocol, provides a high-bandwidth bus, and supports Quality of Service (QoS) and Reliability, Availability, and Serviceability (RAS) features.
The LPDDR5 subsystem supports LPDDR5 and LPDDR4X. LPDDR4X supports 8-bank mode, while LPDDR5 supports 16-bank and bank-group (BG) modes. The subsystem supports 2-channel and 4-channel configurations, with a maximum data rate of up to 6400 Mb/s.

The PCIe subsystem supports 1 x8 link, 1 x4 link, 1 x2 link, and 2 x1 links, with support for PCIe Gen4.
The I/O subsystem provides USB 2.0, USB 3.0, DP, I2C, GMAC, sensor hub, FCH, and audio processing modules, supporting most common applications.
The Audio subsystem provides audio processing, data transfer, and interface capabilities with support for high-performance DSP processing and multi-channel audio applications. It supports standard audio interfaces such as I2S and HDA, with sample rates up to 192 kHz and multiple channel configurations.
The FCH subsystem provides platform control and management functions to support system-level operations.
The System Control subsystem provides security management and power management functions for the SoC. For security management, it supports secure boot, product lifecycle management, configuration gasket, memory fencing, TRNG generation, key protection, and data encryption and decryption. These features help ensure system security. The power management subsystem supports DVFS for CPU, GPU, and NPU, as well as DFS for DDR. It provides dynamic voltage and frequency scaling, clock scaling, and power gating for selected subsystems under specific scenarios.
It has a JTAG interface for debugging the system, if necessary.
The above paragraph briefly describes the important functions and features of each subsystem. For more details, refer to the relevant chapters in the following sections.

1.3 Memory Map

Table 1-2 Memory Map

Start Address	End Address	64K blocks in size	IP
0000_0000	0000_FFFF	1 x 64K	FCH_LOCAL_RCSU
0001_0000	04FF_FFFF	1279 x 64K	FCH
0500_0000	0500_FFFF	1 x 64K	CSU_SE_LOCAL_RCSU
0501_0000	05FF_FFFF	255 x 64K	CSU_SE
0600_0000	0600_FFFF	1 x 64K	CSU_PM_LOCAL_RCSU
0601_0000	06FF_FFFF	255 x 64K	CSU_PM
0700_0000	0700_FFFF	1 x 64K	Audio_Local_RCSU
0701_0000	07FF_FFFF	255 x 64K	Audio
0800_0000	0800_FFFF	1 x 64K	Sensor_Hub_Local_RCSU
0801_0000	08FF_FFFF	255 x 64K	Sensor Hub
0900_0000	0900_FFFF	1 x 64K	USB-C_0_LOCAL_RCSU
0901_0000	0901_FFFF	1 x 64K	USB-C_0
0902_0000	0902_FFFF	1 x 64K	USB-C_0-2.0_PHY
0903_0000	0906_FFFF	4 x 64K	USB-C_0-PHY
0907_0000	0907_FFFF	1 x 64K	USB-C_1_LOCAL_RCSU
0908_0000	0908_FFFF	1 x 64K	USB-C_1
0909_0000	0909_FFFF	1 x 64K	USB-C_1-2.0_PHY
090A_0000	090D_FFFF	4 x 64K	USB-C_1-PHY
090E_0000	090E_FFFF	1 x 64K	USB-C_2_LOCAL_RCSU
090F_0000	090F_FFFF	1 x 64K	USB-C_2

Start Address	End Address	64K blocks in size	IP
0910_0000	0910_FFFF	1 x 64K	USB-C_2-2.0_PHY
0911_0000	0914_FFFF	4 x 64K	USB-C_2-PHY
0915_0000	0915_FFFF	1 x 64K	USB-C_3_LOCAL_RCSU
0916_0000	0916_FFFF	1 x 64K	USB-C_3
0917_0000	0917_FFFF	1 x 64K	USB-C_3-2.0_PHY
0918_0000	091B_FFFF	4 x 64K	USB-C_3-PHY
091C_0000	091C_FFFF	1 x 64K	USB3_LOCAL_RCSU
091D_0000	091D_FFFF	1 x 64K	USB3_0
091E_0000	091E_FFFF	1 x 64K	USB3_1
091F_0000	091F_FFFF	1 x 64K	USB3_0-2.0_PHY
0920_0000	0920_FFFF	1 x 64K	USB3_1-2.0_PHY
0921_0000	0924_FFFF	4 x 64K	USB3-PHY
0925_0000	0925_FFFF	1 x 64K	USB2_0_LOCAL_RCSU
0926_0000	0926_FFFF	1 x 64K	USB2_0
0927_0000	0927_FFFF	1 x 64K	USB2_0-PHY
0928_0000	0928_FFFF	1 x 64K	USB2_1_LOCAL_RCSU
0929_0000	0929_FFFF	1 x 64K	USB2_1
092A_0000	092A_FFFF	1 x 64K	USB2_1-PHY
092B_0000	092B_FFFF	1 x 64K	USB2_2_LOCAL_RCSU
092C_0000	092C_FFFF	1 x 64K	USB2_2
092D_0000	092D_FFFF	1 x 64K	USB2_2-PHY
092E_0000	092E_FFFF	1 x 64K	USB2_3_LOCAL_RCSU
092F_0000	092F_FFFF	1 x 64K	USB2_3
0930_0000	0930_FFFF	1 x 64K	USB2_3-PHY
0931_0000	0931_FFFF	1 x 64K	ETH_LOCAL_RCSU
0932_0000	0932_FFFF	1 x 64K	ETH0
0933_0000	0933_FFFF	1 x 64K	ETH1
0934_0000	09FF_FFFF	204 x 64K	RSVD
0A00_0000	0A00_FFFF	1 x 64K	PCIe_X8_LOCAL_RCSU
0A01_0000	0A01_FFFF	1 x 64K	PCIe_X8_Ctler
0A02_0000	0A05_FFFF	4 x 64K	PCIe_X8_PHY
0A06_0000	0A06_FFFF	1 x 64K	PCIE_X4211_LOCAL_RCSU
0A07_0000	0A07_FFFF	1 x 64K	PCIe_X4_Ctler
0A08_0000	0A0B_FFFF	4 x 64K	PCIe_X4_PHY
0A0C_0000	0A0C_FFFF	1 x 64K	PCIe_X2_Ctler
0A0D_0000	0A0D_FFFF	1 x 64K	PCIe_X1_0_Ctler
0A0E_0000	0A0E_FFFF	1 x 64K	PCIe_X1_1_Ctler
0A0F_0000	0A12_FFFF	4 x 64K	PCIe_X211_PHY
0A13_0000	0AFF_FFFF	237 x 64K	RSVD
0B00_0000	0B00_FFFF	1 x 64K	SMMU0_LOCAL_RCSU
0B01_0000	0B0C_FFFF	12 x 64K	SMMU0.TCU
0B0D_0000	0B0D_FFFF	1 x 64K	SMMU1_LOCAL_RCSU
0B0E_0000	0B19_FFFF	12 x 64K	SMMU1.TCU

Start Address	End Address	64K blocks in size	IP
0B1A_0000	0B1A_FFFF	1 x 64K	SMMU2_LOCAL_RCSU
0B1B_0000	0B5A_FFFF	64 x 64K	SMMU2.TCU
0B5B_0000	0BFF_FFFF	165 x 64K	RSVD
0C00_0000	0C00_FFFF	1 x 64K	DDR_Ctler0_LOCAL_RCSU
0C01_0000	0C01_FFFF	1 x 64K	DDR Controller0
0C02_0000	0C02_FFFF	1 x 64K	DDR_Ctler1_LOCAL_RCSU
0C03_0000	0C03_FFFF	1 x 64K	DDR Controller1
0C04_0000	0C04_FFFF	1 x 64K	DDR_Ctler2_LOCAL_RCSU
0C05_0000	0C05_FFFF	1 x 64K	DDR Controller2
0C06_0000	0C06_FFFF	1 x 64K	DDR_Ctler3_LOCAL_RCSU
0C07_0000	0C07_FFFF	1 x 64K	DDR Controller3
0C08_0000	0C08_FFFF	1 x 64K	Broadcast_Local_RCSU
0C09_0000	0C09_FFFF	1 x 64K	Broadcast
0C0A_0000	0C0A_FFFF	1 x 64K	TZC400_0_LOCAL_RCSU
0C0B_0000	0C0B_FFFF	1 x 64K	TZC400_0
0C0C_0000	0C0C_FFFF	1 x 64K	TZC400_1_LOCAL_RCSU
0C0D_0000	0C0D_FFFF	1 x 64K	TZC400_1
0C0E_0000	0C0E_FFFF	1 x 64K	TZC400_2_LOCAL_RCSU
0C0F_0000	0C0F_FFFF	1 x 64K	TZC400_2
0C10_0000	0C10_FFFF	1 x 64K	TZC400_3_LOCAL_RCSU
0C11_0000	0C11_FFFF	1 x 64K	TZC400_3
0C12_0000	0CFF_FFFF	238 x 64K	RSVD
0D00_0000	0D00_FFFF	1 x 64K	PCIE_HUB_LOCAL_RCSU
0D01_0000	0D02_FFFF	2 x 64K	PCIe_Hub
0D03_0000	0D03_FFFF	1 x 64K	MMHUB_LOCAL_RCSU
0D04_0000	0D07_FFFF	4 x 64K	MM_Hub
0D08_0000	0D08_FFFF	1 x 64K	System_Hub_LOCAL_RCSU
0D09_0000	0D0C_FFFF	4 x 64K	System_Hub
0D0D_0000	0D14_FFFF	8 x 64K	SMN_Hub
0D15_0000	0D15_FFFF	1 x 64K	SMN_RCSU
0D16_0000	0DFF_FFFF	234 x 64K	RSVD
0E00_0000	0E00_FFFF	1 x 64K	GICD_LOCAL_RCSU
0E01_0000	0E39_FFFF	57 x 64K	GICD/ITS/GCI
0E3A_0000	0EFF_FFFF	198 x 64K	RSVD
0F00_0000	0FC4_FFFF	197 x 64K	DSU_Utility
0FC5_0000	0FC5_FFFF	1 x 64K	Core_PCSM_RCSU_LOCAL
0FC6_0000	0FCB_FFFF	6 x 64K	Core_PCSM_RCSU
0FCC_0000	0FFF_FFFF	52 x 64K	RSVD
1000_0000	1000_FFFF	1 x 64K	CI-700_LOCAL_RCSU
1001_0000	13FF_FFFF	1023 x 64K	CI-700
1400_0000	1400_FFFF	1 x 64K	DPU0_LOCAL_RCSU
1401_0000	1402_FFFF	2 x 64K	DPU0
1403_0000	1404_FFFF	2 x 64K	DPU0_AFBC

Start Address	End Address	64K blocks in size	IP
1405_0000	1405_FFFF	1 x 64K	DP0_LOCAL_RCSU
1406_0000	1406_FFFF	1 x 64K	DP0_TX
1407_0000	1407_FFFF	1 x 64K	DPU1_LOCAL_RCSU
1408_0000	1409_FFFF	2 x 64K	DPU1
140A_0000	140B_FFFF	2 x 64K	DPU1_AFBC
140C_0000	140C_FFFF	1 x 64K	DP1_LOCAL_RCSU
140D_0000	140D_FFFF	1 x 64K	DP1_TX
140E_0000	140E_FFFF	1 x 64K	DPU2__LOCAL_RCSU
140F_0000	1410_FFFF	2 x 64K	DPU2
1411_0000	1412_FFFF	2 x 64K	DPU2_AFBC
1413_0000	1413_FFFF	1 x 64K	DP2_LOCAL_RCSU
1414_0000	1414_FFFF	1 x 64K	DP2_TX2
1415_0000	1415_FFFF	1 x 64K	DPU3_LOCAL_RCSU
1416_0000	1417_FFFF	2 x 64K	DPU3
1418_0000	1419_FFFF	2 x 64K	DPU3_AFBC
141A_0000	141A_FFFF	1 x 64K	DP3_LOCAL_RCSU
141B_0000	141B_FFFF	1 x 64K	DP3_TX
141C_0000	141C_FFFF	1 x 64K	DPU4_LOCAL_RCSU
141D_0000	141E_FFFF	2 x 64K	DPU4
141F_0000	1420_FFFF	2 x 64K	DPU4_AFBC
1421_0000	1421_FFFF	1 x 64K	DP4_LOCAL_RCSU
1422_0000	1422_FFFF	1 x 64K	DP4_TX
1423_0000	1423_FFFF	1 x 64K	VPU_LOCAL_RCSU
1424_0000	1424_FFFF	1 x 64K	VPU
1425_0000	1425_FFFF	1 x 64K	NPU_LOCAL_RCSU
1426_0000	1426_FFFF	1 x 64K	NPU
1427_0000	1427_FFFF	1 x 64K	MIPI0_LOCAL_RCSU
1428_0000	1428_FFFF	1 x 64K	CSI0
1429_0000	1429_FFFF	1 x 64K	CSI1
142A_0000	142A_FFFF	1 x 64K	CSI-DPHY0
142B_0000	142B_FFFF	1 x 64K	CSIDMA0
142C_0000	142C_FFFF	1 x 64K	CSIDMA1
142D_0000	142D_FFFF	1 x 64K	MIPI1_LOCAL_RCSU
142E_0000	142E_FFFF	1 x 64K	CSI2
142F_0000	142F_FFFF	1 x 64K	CSI3
1430_0000	1430_FFFF	1 x 64K	CSI-DPHY1
1431_0000	1431_FFFF	1 x 64K	CSIDMA2
1432_0000	1432_FFFF	1 x 64K	CSIDMA3
1433_0000	1433_FFFF	1 x 64K	ISP_0_LOCAL_RCSU
1434_0000	1434_FFFF	1 x 64K	ISP_0
1435_0000	1435_FFFF	1 x 64K	ISP_1_LOCAL_RCSU
1436_0000	143A_FFFF	5 x 64K	ISP_1
143B_0000	14FF_FFFF	197 x 64K	RSVD

Start Address	End Address	64K blocks in size	IP
1500_0000	1500_FFFF	1 x 64K	GPU_LOCAL_RCSU
1501_0000	15FF_FFFF	255 x 64K	GPU Register
1600_0000	161F_FFFF	32 x 64K	PMCTRL_S5
1620_0000	16FF_FFFF	224 x 64K	RSVD
1700_0000	17FF_FFFF	256 x 64K	RSVD
1800_0000	1BFF_FFFF	1024 x 64K	APBIC_0
1C00_0000	1C00_FFFF	1 x 64K	LDO0
1C01_0000	1C01_FFFF	1 x 64K	LDO1
1C02_0000	1CFF_FFFF	254 x 64K	RSVD
1D00_0000	1FFF_FFFF	768 x 64K	RSVD

1.4 Interrupt Map

Table 1-3 Interrupt Map

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
0	cluster	nCLUSTERERRIRQ	Error indicator for an ECC error that causes potential data corruption or loss of coherency in the L3 RAMs, or AXI or CHI write transactions with a write response error condition.
1		nCLUSTERFAULTIRQ	Fault indicator for a detected 1 bit or 2bit Error Correcting Code(ECC) error in the L3 or snoop filter RAMS.
2		nCLUSTERPMUIRQ	Cluster PMU interrupt request.
3		nCOMPLEXERRIRQ[0]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CX down to each complex
4		nCOMPLEXERRIRQ[1]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CX down to each complex
5		nCOMPLEXERRIRQ[2]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CX down to each complex
6		nCOMPLEXERRIRQ[3]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CX down to each complex

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
7		nCOMPLEXFAULTIRQ[0]	Fault indicator for a detected 1 bit or 2 bit ECC error in the complex RAMs. Bits CX down to 0 are for the shared RAMs in each complex.
8		nCOMPLEXFAULTIRQ[1]	Fault indicator for a detected 1 bit or 2 bit ECC error in the complex RAMs. Bits CX down to 0 are for the shared RAMs in each complex.
9		nCOMPLEXFAULTIRQ[2]	Fault indicator for a detected 1 bit or 2 bit ECC error in the complex RAMs. Bits CX down to 0 are for the shared RAMs in each complex.
10		nCOMPLEXFAULTIRQ[3]	Fault indicator for a detected 1 bit or 2 bit ECC error in the complex RAMs. Bits CX down to 0 are for the shared RAMs in each complex.
11		nCOREERRIRQ[0]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
12		nCOREERRIRQ[1]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
13		nCOREERRIRQ[2]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
14		nCOREERRIRQ[3]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
15		nCOREERRIRQ[4]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
16		nCOREERRIRQ[5]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
17		nCOREERRIRQ[6]	Error indicator for an ECC error that causes potential data corruption or loss

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
18		nCOREERRIRQ[7]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
19		nCOREERRIRQ[8]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
20		nCOREERRIRQ[9]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
21		nCOREERRIRQ[10]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
22		nCOREERRIRQ[11]	Error indicator for an ECC error that causes potential data corruption or loss of coherency. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
23		nCOREFAULTIRQ[0]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
24		nCOREFAULTIRQ[1]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
25		nCOREFAULTIRQ[2]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
26		nCOREFAULTIRQ[3]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
27		nCOREFAULTIRQ[4]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
28		nCOREFAULTIRQ[5]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
29		nCOREFAULTIRQ[6]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
30		nCOREFAULTIRQ[7]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
31		nCOREFAULTIRQ[8]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
32		nCOREFAULTIRQ[9]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
33		nCOREFAULTIRQ[10]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
34		nCOREFAULTIRQ[11]	Fault indicator for a detected 1 or 2 bit ECC error in the core RAMs. Bits CN down to 0 are for the L1 and L2 RAMs in each core.
35		nMPAMSIRQ	Error indicator from MPAM Secure programming.
36		nMPAMNSIRQ	Error indicator from MPAM Non-Secure programming.
37		CTIIRQ[12]	CTI interrupt , 1 cluster interrupts
38		COREPPUIRQ[0]	Core Power Policy Unit (PPU) interrupt indication. Recommand that connect this signal to the component that is programming the PPUs.
39		COREPPUIRQ[1]	Core PPU interrupt indication. Recommand that connect this signal to the component that is programming the PPUs.
40		COREPPUIRQ[2]	Core PPU interrupt indication.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			Recommend that connect this signal to the component that is programming the PPUs.
41		COREPPUIRQ[3]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
42		COREPPUIRQ[4]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
43		COREPPUIRQ[5]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
44		COREPPUIRQ[6]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
45		COREPPUIRQ[7]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
46		COREPPUIRQ[8]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
47		COREPPUIRQ[9]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
48		COREPPUIRQ[10]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
49		COREPPUIRQ[11]	Core PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.
50		CLUSTERPPUIRQ	Cluster PPU interrupt indication. Recommend that connect this signal to the component that is programming the PPUs.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
51	ci700	INTREQPMU_NID68	MPU count overflow interrupt x=HN-T NodeID
52		INTREQERRNS_NID68	Non-Secure error handling interrupt
53		INTREQERRS_NID68	Secure error handling interrupt
54		INTREQFAULTNS_NID68	Non-Secure fault handling interrupt
55		INTREQFAULTS_NID68	Secure fault handling interrupt
56		INTREQMPAMERRNS_NID68	Non-secure MPAM fault handling interrupt
57		INTREQMPAMERRS_NID68	Secure MPAM fault handling interrupt
58		INTREQPPU_NID68	Power state transition complete
59		ext_ci700_rcsu_ci700_rcsu_ppu_irq	ci700 ppu interrupt indication.
60	ni700_mmhub	CLK_0_nPMUINTERRUPT	Active-LOW level sensitive interrupt to indicate a counter, event, or cycle has overflowed
61		PD_0_INTERRUPT	Secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
62		PD_0_NS_INTERRUPT	Non-secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
63		mm_hub_rcsu_ppu_irq	mmhub ppu interrupt indication.
64	ni700_pcie	CLK_0_nPMUINTERRUPT	Active-LOW level sensitive interrupt to indicate a counter, event, or cycle has overflowed
65		PD_0_INTERRUPT	Secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
66		PD_0_NS_INTERRUPT	Non-secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
67		pcie_hub_rcsu_ppu_irq	pciehub ppu interrupt indication.
68	ni700_syshub	CLK_0_nPMUINTERRUPT	Active-LOW level sensitive interrupt to indicate a counter, event, or cycle has overflowed

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
69		PD_0_INTERRUPT	Secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
70		PD_0_NS_INTERRUPT	Non-secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
71	ni700_smn	CLK_0_nPMUINTERRUPT	Active-LOW level sensitive interrupt to indicate a counter, event, or cycle has overflowed
72		PD_0_INTERRUPT	Secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
73		PD_0_NS_INTERRUPT	Non-secure active-HIGH level-sensitive interrupt per power domain that is used to indicate specific conditions(IDM ot non-IDM) within a slave or master interface
74	smmu_pciehub	ext_tcu_event_q_irpt_s	Event queue, Secure interrupt. Asserts a Secure interrupt to indicate that the Event queue is not empty or has overflowed.
75		ext_tcu_event_q_irpt_ns	Event queue, Non-secure interrupt. Asserts a Non-secure interrupt to indicate that the Event queue is not empty or has overflowed.
76		ext_tcu_cmd_sync_irpt_ns	SYNC complete, Non-secure interrupt. Asserts a Non-secure interrupt to indicate that the CMD_SYNC command is complete.
77		ext_tcu_cmd_sync_irpt_s	SYNC complete, Secure interrupt. Asserts a Secure interrupt to indicate that the CMD_SYNC command is complete.
78		ext_tcu_global_irpt_ns	Asserts a global Non-secure interrupt
79		ext_tcu_global_irpt_s	Asserts a global Secure interrupt
80		ext_tcu_ras_fhi	Fault handling RAS interrupt for a contained error

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
81		ext_tcu_ras_cri	Error recovery RAS interrupt for an uncontained error
82		ext_tcu_ras_eri	Critical error interrupt, for an uncontainable uncorrected error
83		ext_tcu_pmu_irpt	Asserts a PMU interrupt.
84		ext_tcu_pri_q_irpt_ns	Asserts a Page Request Interface (PRI) queue interrupt
85		ext_tbu_ras_fhi	Fault handling RAS interrupt for a contained error
86		ext_tbu_ras_eri	Error recovery RAS interrupt for an uncontained error
87		ext_tbu_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
88		ext_tbu_pmu_irpt	PMU interrupt
89	smmu_syshub	ext_tcu_event_q_irpt_s	Event queue, Secure interrupt. Asserts a Secure interrupt to indicate that the Event queue is not empty or has overflowed.
90		ext_tcu_event_q_irpt_ns	Event queue, Non-secure interrupt. Asserts a Non-secure interrupt to indicate that the Event queue is not empty or has overflowed.
91		ext_tcu_cmd_sync_irpt_ns	SYNC complete, Non-secure interrupt. Asserts a Non-secure interrupt to indicate that the CMD_SYNC command is complete.
92		ext_tcu_cmd_sync_irpt_s	SYNC complete, Secure interrupt. Asserts a Secure interrupt to indicate that the CMD_SYNC command is complete.
93		ext_tcu_global_irpt_ns	Asserts a global Non-secure interrupt
94		ext_tcu_global_irpt_s	Asserts a global Secure interrupt
95		ext_tcu_ras_fhi	Fault handling RAS interrupt for a contained error
96		ext_tcu_ras_cri	Error recovery RAS interrupt for an uncontained error
97		ext_tcu_ras_eri	Critical error interrupt, for an uncontainable uncorrected error
98		ext_tcu_pmu_irpt	Asserts a PMU interrupt.
99		ext_tcu_pri_q_irpt_ns	Asserts a Page Request Interface (PRI) queue interrupt

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
100		ext_tbu_0_ras_fhi	Fault handling RAS interrupt for a contained error
101		ext_tbu_0_ras_eri	Error recovery RAS interrupt for an uncontained error
102		ext_tbu_0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
103		ext_tbu_0_pmu_irpt	PMU interrupt
104		ext_tbu_1_ras_fhi	Fault handling RAS interrupt for a contained error
105		ext_tbu_1_ras_eri	Error recovery RAS interrupt for an uncontained error
106		ext_tbu_1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
107		ext_tbu_1_pmu_irpt	PMU interrupt
108	smmu_mmhub	ext_tcu_event_q_irpt_s	Event queue, Secure interrupt. Asserts a Secure interrupt to indicate that the Event queue is not empty or has overflowed.
109		ext_tcu_event_q_irpt_ns	Event queue, Non-secure interrupt. Asserts a Non-secure interrupt to indicate that the Event queue is not empty or has overflowed.
110		ext_tcu_cmd_sync_irpt_ns	SYNC complete, Non-secure interrupt. Asserts a Non-secure interrupt to indicate that the CMD_SYNC command is complete.
111		ext_tcu_cmd_sync_irpt_s	SYNC complete, Secure interrupt. Asserts a Secure interrupt to indicate that the CMD_SYNC command is complete.
112		ext_tcu_global_irpt_ns	Asserts a global Non-secure interrupt
113		ext_tcu_global_irpt_s	Asserts a global Secure interrupt
114		ext_tcu_ras_fhi	Fault handling RAS interrupt for a contained error
115		ext_tcu_ras_cri	Error recovery RAS interrupt for an uncontained error
116		ext_tcu_ras_eri	Critical error interrupt, for an uncontainable uncorrected error
117		ext_tcu_pmu_irpt	Asserts a PMU interrupt.
118		ext_tcu_pri_q_irpt_ns	Asserts a Page Request Interface (PRI) queue interrupt

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
119		ext_dpu0_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
120		ext_dpu0_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
121		ext_dpu0_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
122		ext_dpu0_tbu0_pmu_irpt	PMU interrupt
123		ext_dpu0_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
124		ext_dpu0_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
125		ext_dpu0_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
126		ext_dpu0_tbu1_pmu_irpt	PMU interrupt
127		ext_dpu0_tbu2_ras_fhi	Fault handling RAS interrupt for a contained error
128		ext_dpu0_tbu2_ras_eri	Error recovery RAS interrupt for an uncontained error
129		ext_dpu0_tbu2_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
130		ext_dpu0_tbu2_pmu_irpt	PMU interrupt
131		ext_dpu1_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
132		ext_dpu1_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
133		ext_dpu1_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
134		ext_dpu1_tbu0_pmu_irpt	PMU interrupt
135		ext_dpu1_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
136		ext_dpu1_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
137		ext_dpu1_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
138		ext_dpu1_tbu1_pmu_irpt	PMU interrupt
139		ext_dpu1_tbu2_ras_fhi	Fault handling RAS interrupt for a contained error
140		ext_dpu1_tbu2_ras_eri	Error recovery RAS interrupt for an uncontained error
141		ext_dpu1_tbu2_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
142		ext_dpu1_tbu2_pmu_irpt	PMU interrupt

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
143		ext_dpu2_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
144		ext_dpu2_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
145		ext_dpu2_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
146		ext_dpu2_tbu0_pmu_irpt	PMU interrupt
147		ext_dpu2_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
148		ext_dpu2_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
149		ext_dpu2_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
150		ext_dpu2_tbu1_pmu_irpt	PMU interrupt
151		ext_dpu2_tbu2_ras_fhi	Fault handling RAS interrupt for a contained error
152		ext_dpu2_tbu2_ras_eri	Error recovery RAS interrupt for an uncontained error
153		ext_dpu2_tbu2_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
154		ext_dpu2_tbu2_pmu_irpt	PMU interrupt
155		ext_dpu3_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
156		ext_dpu3_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
157		ext_dpu3_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
158		ext_dpu3_tbu0_pmu_irpt	PMU interrupt
159		ext_dpu3_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
160		ext_dpu3_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
161		ext_dpu3_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
162		ext_dpu3_tbu1_pmu_irpt	PMU interrupt
163		ext_dpu3_tbu2_ras_fhi	Fault handling RAS interrupt for a contained error
164		ext_dpu3_tbu2_ras_eri	Error recovery RAS interrupt for an uncontained error
165		ext_dpu3_tbu2_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
166		ext_dpu3_tbu2_pmu_irpt	PMU interrupt

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
167		ext_dpu4_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
168		ext_dpu4_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
169		ext_dpu4_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
170		ext_dpu4_tbu0_pmu_irpt	PMU interrupt
171		ext_dpu4_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
172		ext_dpu4_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
173		ext_dpu4_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
174		ext_dpu4_tbu1_pmu_irpt	PMU interrupt
175		ext_dpu4_tbu2_ras_fhi	Fault handling RAS interrupt for a contained error
176		ext_dpu4_tbu2_ras_eri	Error recovery RAS interrupt for an uncontained error
177		ext_dpu4_tbu2_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
178		ext_dpu4_tbu2_pmu_irpt	PMU interrupt
179		ext_npu_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
180		ext_npu_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
181		ext_npu_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
182		ext_npu_tbu0_pmu_irpt	PMU interrupt
183		ext_npu_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
184		ext_npu_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
185		ext_npu_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
186		ext_npu_tbu1_pmu_irpt	PMU interrupt
187		ext_isp_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
188		ext_isp_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
189		ext_isp_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
190		ext_isp_tbu0_pmu_irpt	PMU interrupt

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
191		ext_isp_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
192		ext_isp_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
193		ext_isp_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
194		ext_isp_tbu1_pmu_irpt	PMU interrupt
195		ext_isp_tbu2_ras_fhi	Fault handling RAS interrupt for a contained error
196		ext_isp_tbu2_ras_eri	Error recovery RAS interrupt for an uncontained error
197		ext_isp_tbu2_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
198		ext_isp_tbu2_pmu_irpt	PMU interrupt
199		ext_csi_tbu0_ras_fhi	Fault handling RAS interrupt for a contained error
200		ext_csi_tbu0_ras_eri	Error recovery RAS interrupt for an uncontained error
201		ext_csi_tbu0_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
202		ext_csi_tbu0_pmu_irpt	PMU interrupt
203		ext_csi_tbu1_ras_fhi	Fault handling RAS interrupt for a contained error
204		ext_csi_tbu1_ras_eri	Error recovery RAS interrupt for an uncontained error
205		ext_csi_tbu1_ras_cri	Critical error interrupt, for an uncontainable uncorrected error
206		ext_csi_tbu1_pmu_irpt	PMU interrupt
207	u_umc_wrapper	ddrc_0_controller_int	Interrupt signal from the LPDDR Controller
208		umc_pi_int_ddrp_0	Interrupt signal from the LPDDR PHY
209		tzc400_int_0	Interrupt signal from the TZC400
210		ddrc_1_controller_int	Interrupt signal from the LPDDR Controller
211		umc_pi_int_ddrp_1	Interrupt signal from the LPDDR PHY
212		tzc400_int_1	Interrupt signal from the TZC400
213		ddrc_2_controller_int	Interrupt signal from the LPDDR Controller
214		umc_pi_int_ddrp_2	Interrupt signal from the LPDDR PHY
215		tzc400_int_2	Interrupt signal from the TZC400
216		ddrc_3_controller_int	Interrupt signal from the LPDDR Controller

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
217		umc_pi_int_ddrp_3	Interrupt signal from the LPDDR PHY
218		tzc400_int_3	Interrupt signal from the TZC400
219	audio_sub	irq_o_i2s_sc0	HDA irq High active High active
220		irq_o_i2s_sc1	mailbox1_int_callee High active
221		irq_o_i2s_sc2	mailbox0_int_caller High active
222		irq_o_i2s_mc2a	audio sw irq High active
223		irq_o_i2s_mc2b	dma irq High active
224		irq_o_i2s_mc5a	wdt irq High active
225		irq_o_i2s_mc5b	i2s_mc5e irq High active
226		irq_o_i2s_mc5c	i2s_mc5d irq High active
227		irq_o_i2s_mc5d	i2s_mc5c irq High active
228		irq_o_i2s_mc5e	i2s_mc5b irq High active
229		irq_o_wdt	i2s_mc5a irq High active
230		irq_o_ap_dma	i2s_mc2b_irq High active
231		irq_o_audio_sw	i2s_mc2a_irq High active
232		irq_o_mailbox0_caller	i2s_sc2_irq High active
233		irq_o_mailbox1_callee	i2s_sc1_irq High active
234		irq_o_hda	i2s_sc0_irq High active
235	gpu_wrapper	ext_vdci_gpu_vddgpu_vddsoc_CTIIRQ_vdci_less2more_out[0:0]	for DFD, CTI interrupt
236		ext_vdci_gpu_vddgpu_vddsoc_CTIIRQ_vdci_less2more_out[1:1]	for DFD, CTI interrupt
237		ext_vdci_gpu_vddgpu_vddsoc_IRQJOB_vdci_less2more_out	Signals the completion or failure of work running on the GPU.
238		ext_vdci_gpu_vddgpu_vddsoc_IRQMMU_vdci_less2more_out	Memory management causes these exceptions. These exceptions are potentially recoverable. The application processor interprets the interrupts that the GPU generates. To determine what must be done to handle the interrupt, the software queries the states of the: • GPU shared memory for job interrupts. • Registers in the CSF GPU manager For MMU and GPU interrupts. The GPU does not need to wait for the result of the interrupt. Instead, it can start executing the next set of jobs.
239		ext_vdci_gpu_vddgpu_vddsoc_IRQGPU_vdci_less2more_out	Exceptions that are not associated with specific jobs. These exceptions are not recoverable. Other control events are: • Reset complete

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			• Powerup and powerdown complete • Cache-flush complete
240	usb_wrapper	u2_host0_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
241		u2_host0_otgirq	Dual mode control interrupt request.
242		u2_host0_host_system_error	A sideband signaling that is active when a catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.
243		u2_host1_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
244		u2_host1_otgirq	Dual mode control interrupt request.
245		u2_host1_host_system_error	A sideband signaling that is active when catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.
246		u2_host2_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
247		u2_host2_otgirq	Dual mode control interrupt request.
248		u2_host2_host_system_error	A sideband signaling that is active when a catastrophic system error occurs. It reflects the value of USBSTS.HSE status bit.
249		u2_host3_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
250		u2_host3_otgirq	Dual mode control interrupt request.
251		u2_host3_host_system_error	A sideband signaling that is active when a catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.
252		u3_typea_ctl0_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
253		u3_typea_ctl0_otgirq	Dual mode control interrupt request.
254		u3_typea_ctl0_host_system_error	A sideband signaling that is active when a catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.
255		u3_typea_ctl0_itp	Active in Device mode only. A one clock-period pulse on this port indicates that an ITP packet has been received. (Not available in HOST USB2.0 configuration only)
256		u3_typea_sd0803_phy_interrupt_ln_0	PHY interrupt output (per lane): Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
257		u3_typea_ctl1_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
258		u3_typea_ctl1_otgirq	Dual mode control interrupt request.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
259		u3_typea_ctl1_host_system_error	A sideband signaling that is active when a catastrophic system error occurs. It reflects the value of USBSTS.HSE status bit.
260		u3_typea_ctl1_itp	Active in Device mode only. A one clock-period pulse on this port indicates that an ITP packet has been received. (Not available in HOST USB2.0 configuration only)
261		u3_typea_sd0803_phy_interrupt_ln_1	PHY interrupt output (per lane): Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
262		u3_typec_drd_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
263		u3_typec_drd_otgirq	Dual mode control interrupt request.
264		u3_typec_drd_host_system_error	A sideband signaling that is active when catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.
265		u3_typec_drd_itp	Active in Device mode only. A one clock-period pulse on this port indicates that an ITP packet has been received. (Not available in HOST USB2.0 configuration only)
266		u3_typec_drd_phy_interrupt_dp	PHY interrupt output for DP link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
267		u3_typec_drd_phy_interrupt_usb	PHY interrupt output for USB link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			the associated link's interrupt status register.
268		u3_typec_host0_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
269		u3_typec_host0_otgirq	Dual mode control interrupt request.
270		u3_typec_host0_host_system_error	A sideband signaling that is active when catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.
271		u3_typec_host0_itp	Active in Device mode only. A one clock-period pulse on this port indicates that an ITP packet has been received. (Not available in HOST USB2.0 configuration only)
272		u3_typec_host0_phy_interrupt_dp	PHY interrupt output for DP link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
273		u3_typec_host0_phy_interrupt_usb	PHY interrupt output for USB link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
274		u3_typec_host1_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
275		u3_typec_host1_otgirq	Dual mode control interrupt request.
276		u3_typec_host1_host_system_error	A sideband signaling that is active when catastrophic system error occurs. It reflects value of USBSTS.HSE status bit.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
277		u3_typec_host1_itp	Active in Device mode only. A one clock-period pulse on this port indicates that an ITP packet has been received. (Not available in HOST USB2.0 configuration only)
278		u3_typec_host1_phy_interrupt_dp	PHY interrupt output for DP link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
279		u3_typec_host1_phy_interrupt_usb	PHY interrupt output for USB link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
280		u3_typec_host2_irq	A bit of the Interrupt request bus is driven to a high logic level to indicate that the USBSSP is initiating a legacy interrupt request. For standard xHCI driver only interrupter 0 is supported and hence only interrupt_req[0] is used. This is an active high-level sensitive interrupt.
281		u3_typec_host2_otgirq	Dual mode control interrupt request.
282		u3_typec_host2_host_system_error	A sideband signaling that is active when a catastrophic system error occurs. It reflects the value of USBSTS.HSE status bit.
283		u3_typec_host2_itp	Active in Device mode only. A one clock-period pulse on this port indicates that an ITP packet has been received. (Not available in HOST USB2.0 configuration only)
284		u3_typec_host2_phy_interrupt_dp	PHY interrupt output for DP link: Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
285		u3_typec_host2_phy_interrupt_usb	PHY interrupt output for USB link: Asserted high when a power state or

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			data rate change does not complete within a timeout. Cleared upon reading the associated link's interrupt status register.
286	fch_top	intr_i2c0	I2C0 interrupt to host. When active high indicates that the I2C requires host intervention.
287		intr_i2c1	I2C1 interrupt to host. When active high indicates that the I2C requires host intervention.
288		intr_i2c2	I2C2 interrupt to host. When active high indicates that the I2C requires host intervention.
289		intr_i2c3	I2C3 interrupt to host. When active high indicates that the I2C requires host intervention.
290		intr_i2c4	I2C4 interrupt to host. When active high indicates that the I2C requires host intervention.
291		intr_i2c5	I2C5 interrupt to host. When active high indicates that the I2C requires host intervention.
292		intr_i2c6	I2C6 interrupt to host. When active high indicates that the I2C requires host intervention.
293		intr_i2c7	I2C7 interrupt to host. When active high indicates that the I2C requires host intervention.
294		intr_spi0	SPI0 interrupt to the CPU. Active high, level sensitive.
295		intr_spi1	SPI1 interrupt to the CPU. Active high, level sensitive.
296		intr_uart0	uart0 interrupt, active HIGH. A single combined interrupt generated as an OR function of several internal interrupts.
297		intr_uart1	uart1 interrupt, active HIGH. A single combined interrupt generated as an OR function of several internal interrupts.
298		intr_uart2	uart2 interrupt, active HIGH. A single combined interrupt generated as an OR function of several internal interrupts.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
299		intr_uart3	uart3 interrupt, active HIGH. A single combined interrupt generated as an OR function of several internal interrupts.
300		intr_i3c0	I3C0 interrupt, indicates that immediate attention from host is required.
301		intr_i3c1	I3C1 interrupt, indicates that immediate attention from host is required.
302		intr_xspi	xSPI external interrupt.
303		intr_dmac	common non-secure channel interrupt for functional interrupt sources only.
304		intr_gpio0	GPIO0 interrupt indicating input pin event.
305		intr_gpio1	GPIO1 interrupt indicating input pin event.
306		intr_gpio2	GPIO2 interrupt indicating input pin event.
307		intr_gpio3	GPIO3 interrupt indicating input pin event.
308		intr_timer[0]	counter0 interrupt bit[0].
309		intr_timer[1]	counter1 interrupt bit[0].
310		intr_timer[2]	counter2 interrupt bit[0].
311		intr_timer[3]	counter3 interrupt bit[0].
312		intr_timer[4]	counter4 interrupt bit[0].
313		intr_timer[5]	counter5 interrupt bit[0].
314		intr_timer[6]	counter6 interrupt bit[0].
315		intr_timer[7]	counter7 interrupt bit[0].
466		intr_timer[8]	counter1 interrupt bit[1]|bit[2]|bit[3]|bit[4].
467		intr_timer[9]	counter1 interrupt bit[5].
468		intr_timer[10]	counter3 interrupt bit[5].
469		intr_timer[11]	counter5 interrupt bit[5].
470		intr_timer[12]	counter7 interrupt bit[5].
471		intr_timer[13]	counter5 interrupt bit[6].
472		intr_timer[14]	counter7 interrupt bit[6].
316	dpu	ext_u_dpu0_IRQ0	This interrupt is from the global control unit and provides interrupts for the layer processing units, the composition units, and display output units.
317		ext_u_dpu0_IRQ1	This interrupt is from the AFBC DMA unit for the subsystems for AFBC encode and DMA.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
318		ext_u_dpu1_IRQ0	This interrupt is from the global control unit and provides interrupts for the layer processing units, the composition units, and display output units.
319		ext_u_dpu1_IRQ1	This interrupt is from the AFBC DMA unit for the subsystems for AFBC encode and DMA.
320		ext_u_dpu2_IRQ0	This interrupt is from the global control unit and provides interrupts for the layer processing units, the composition units, and display output units.
321		ext_u_dpu2_IRQ1	This interrupt is from the AFBC DMA unit for the subsystems for AFBC encode and DMA.
322		ext_u_dpu3_IRQ0	This interrupt is from the global control unit and provides interrupts for the layer processing units, the composition units, and display output units.
323		ext_u_dpu3_IRQ1	This interrupt is from the AFBC DMA unit for the subsystems for AFBC encode and DMA.
324		ext_u_dpu4_IRQ0	This interrupt is from the global control unit and provides interrupts for the layer processing units, the composition units, and display output units.
325		ext_u_dpu4_IRQ1	This interrupt is from the AFBC DMA unit for the subsystems for AFBC encode and DMA.
326	vpu_top	ext_u_vpu_core_IRQ	The interrupt signal. This signal is active-HIGH level. The typical interrupt rate is one interrupt per frame
327	npu_wrapper	ext_u_npu_core_IRQO	The interrupt signal. This signal is active-HIGH level.
328	isp_wrapper	ext_u_isp_core_o_isp_int	refer to Table 3-3 Interrupt Source List of <Arm China LINLON-i7 Processor Technical Reference Manual> p.32
329		ext_u_isp_core_o_isp_int_abnormal	refer to Table 3-3 Interrupt Source List of <Arm China LINLON-i7 Processor Technical Reference Manual> p.32
330		ext_u_isp_core_o_gdc_int	refer to Table 3-3 Interrupt Source List of <Arm China LINLON-i7 Processor Technical Reference Manual> p.32

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
331		ext_u_isp_core_o_gdc_int_abnormal	refer to Table 3-3 Interrupt Source List of <Arm China LINLON-i7 Processor Technical Reference Manual> p.32
332	dp_top	dp0_apb_int	This signal is or of dp_apb_int_dptx , dp_apb_int_dsc , dp_apb_int_phy
333		dp1_apb_int	This signal is or of dp_apb_int_dptx , dp_apb_int_dsc , dp_apb_int_phy
334		dp2_apb_int	This signal is or of dp_apb_int_dptx , dp_apb_int_dsc , dp_apb_int_phy
335		dp3_apb_int	This signal is or of dp_apb_int_dptx , dp_apb_int_dsc , dp_apb_int_phy
336		dp4_apb_int	This signal is or of dp_apb_int_dptx , dp_apb_int_dsc , dp_apb_int_phy
337	sys_dfd_top	soc_etr_bufintr	The memory controlled by etr is full
338		soc_catu_addrerr	Error interrupt indicating failed address translation
339		gic_o_trig0_EVENT	dfd connect bridge which also connect to GIC GIC should send signal to bridge
340		gic_o_trig1_EVENT	dfd connect bridge which also connect to GIC GIC should send signal to bridge
341		gic_o_trig2_EVENT	dfd connect bridge which also connect to GIC GIC should send signal to bridge
342		gic_o_trig3_EVENT	dfd connect bridge which also connect to GIC GIC should send signal to bridge
343	u_gmac_sub_wrapper	gmac0_intr_q0	gmac0 queue0 intr
344		gmac0_intr_q1	gmac0 queue1 intr
345		gmac0_intr_q2	gmac0 queue2 intr
346		gmac0_intr_q3	gmac0 queue3 intr
347		gmac0_intr_q4	gmac0 queue4 intr
348		gmac0_intr_q5	gmac0 queue5 intr
349		gmac0_intr_q6	gmac0 queue6 intr
350		gmac0_intr_q7	gmac0 queue7 intr
351		gmac1_intr_q0	gmac1 queue0 intr
352		gmac1_intr_q1	gmac1 queue1 intr
353		gmac1_intr_q2	gmac1 queue2 intr
354		gmac1_intr_q3	gmac1 queue3 intr
355		gmac1_intr_q4	gmac1 queue4 intr
356		gmac1_intr_q5	gmac1 queue5 intr
357		gmac1_intr_q6	gmac1 queue6 intr
358		gmac1_intr_q7	gmac1 queue7 intr
359	csu_pm_wrapper	int_callee_mailbox1	Mailbox 1 intr
360		int_callee_mailbox3	Mailbox 3 intr

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
361		int_callee_mailbox5	Mailbox 5 intr
362		int_callee_mailbox7	Mailbox 7 intr
363		int_caller_mailbox0	Mailbox 0 intr
364		int_caller_mailbox2	Mailbox 2 intr
365		int_caller_mailbox4	Mailbox 4 intr
366		int_caller_mailbox6	Mailbox 6 intr
367		pm_i2c_intr0	I2c 0 intr
368		pm_i2c_intr1	I2c 1 intr
369		pm_i2c_intr2	I2c 2 intr
370		pm_i2c_intr3	I2c 3 intr
371		prochot_intr	Chip temperature is higher than warning line, or of sensor 0~12
372	s5_ss_top	u0_cdnsgpio gpio_int	AP gpio intr
373		u1_cdnsgpio gpio_int	AP gpio intr
374		u2_cdnsgpio gpio_int	SF gpio intr
375		pm_intr	wakeup from str/sd
376		syswdt_intr	sys wdt l0 intr
377	csu_se_top	intr_star_lockup_vdci_out	se output to gic during star lockup
378		intr_mailb0_caller	Mailbox0 caller, Used for SE to AP virtualization1
379		intr_mailb1_callee	Mailbox1 callee, Used for AP to SE virtualization1
380		intr_mailb2_caller	Mailbox2 caller, Used for SE to AP virtualization0
381		intr_mailb3_callee	Mailbox3 callee, Used for AP to SE virtualization0
382		intr_mailb4_caller	Mailbox4 caller, Used for SE to AP ATF
383		intr_mailb5_callee	Mailbox5 callee, Used for AP to SE ATF
384		intr_mailb6_caller	Mailbox6 caller, Used for SE to AP kernel
385		intr_mailb7_callee	Mailbox7 callee, Used for AP to SE kernel
386		intr_uart1	Uart1 interrupt
387		intr_uart0	Uart0 interrupt
388		csu_se_wdg_wdogint	se watchdog interrupt first level, used for tips
389		csu_se_star_CTIIRQ_vdci_out[1]	star CTI interrupt to AP.
390		csu_se_star_CTIIRQ_vdci_out[0]	star CTI interrupt to AP.
473		csu_se_s0_dummy_out[2]	jiayu interrupt to AP
474		csu_se_s0_dummy_out[3]	jiayu interrupt to AP
391	sensor_fusion	intr_mailb0_callee	intr_mailb0_callee interrupt to AP.
392		intr_mailb1_caller	intr_mailb1_caller interrupt to AP.
393		intr_wdogres	watchdog timeout reset interrupt to AP.

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
394		intr_spi	SPI interrupt to the CPU. Active high, level sensitive.
395		intr_i3c1	I3C1 interrupt, indicate that immediate attention from host is required.
396		intr_i3c0	I3C0 interrupt, indicate that immediate attention from host is required.
397		intr_i2c1	I2C1 interrupt to host. When active high indicates that the I2C requires host intervention.
398		intr_i2c0	I2C0 interrupt to host. When active high indicates that the I2C requires host intervention.
399		CTIIRQ[1]	star CTI interrupt to AP.
400		CTIIRQ[0]	star CTI interrupt to AP.
401		intr_sw_define	software interrupt to AP.
402	pcie_wrapper	x8_correctable_error_out	The client may activate this input for one cycle to indicate a correctable error detected within the client logic that needs to be reported as an internal error through the PCI Express Advanced Error Reporting mechanism. In response, the Controller sets the Corrected Internal Error Status bit in the AER Correctable Error Status Register of all enabled Functions. This error is not considered Function-specific.
403		x8_fatal_error_out	masked. In multi-Function cores, this is the logical OR of the fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
404		x8_non_fatal_error_out	not masked. In multi-Function cores, this is the logical OR of the non-fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
405		x8_local_interrupt	Local Error and Status Register Interrupt. This is an active high, level interrupt. This will be asserted till the Local Error and Status Register is cleared by software. The interrupt_signals_sideband can be looked at to determine which Event triggered the Interrupt
406		x8_phy_interrupt_out	This is an active high, level interrupt output. This output is asserted by the Controller in the RP mode to signal one of the following link training-related events: The link bandwidth changed as a result of the change in the link width or operating speed and the change was initiated locally (not by the link partner), without the link going down. This interrupt is enabled by the Link Bandwidth Management Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Bandwidth Management Status bit of the Link Status Register. The link bandwidth changed autonomously as a result of the change in the link width or operating speed and the change was initiated by the remote node. This interrupt is enabled by the Link Autonomous Bandwidth Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Autonomous Bandwidth Status bit of the Link Status Register. The phy_interrupt_out output is not active when the Controller is configured as an Endpoint.
407		x8_intx_out[0]	

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
408		x8_intx_out[1]	When the core is configured as RP, this output emulates the PCI legacy interrupts. Bit 0= INTA, Bit 1= INTB, Bit2= INTC, Bit 3= INTD Note: Only INTA is used in SKY1
409		x8_intx_out[2]	
410		x8_intx_out[3]	
411		x8_phy_state_chg_sts_p_0	PHY interrupt output (per lane): Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
412		x4_correctable_error_out	The client may activate this input for one cycle to indicate a correctable error detected within the client logic that needs to be reported as an internal error through the PCI Express Advanced Error Reporting mechanism. In response, the Controller sets the Corrected Internal Error Status bit in the AER Correctable Error Status Register of all enabled Functions. This error is not considered Function-specific.
413		x4_fatal_error_out	masked. In multi-Function cores, this is the logical OR of the fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
414		x4_non_fatal_error_out	not masked. In multi-function cores, this is the logical OR of the non-fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
415		x4_local_interrupt	Local Error and Status Register Interrupt. This is an active high, level interrupt. This will be asserted till the Local Error and Status Register is cleared by software. The interrupt_signals_sideband can be looked at to determine which Event triggered the Interrupt
416		x4_phy_interrupt_out	This is an active high, level interrupt output. This output is asserted by the Controller in the RP mode to signal one of the following link training-related events: The link bandwidth changed as a result of the change in the link width or operating speed and the change was initiated locally (not by the link partner), without the link going down. This interrupt is enabled by the Link Bandwidth Management Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Bandwidth Management Status bit of the Link Status Register. The link bandwidth changed autonomously as a result of the change in the link width or operating speed and the change was initiated by the remote node. This interrupt is enabled by the Link Autonomous Bandwidth Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Autonomous Bandwidth Status bit of the Link Status Register. The phy_interrupt_out output is not active when the Controller is configured as an Endpoint.
417		x4_intx_out[0]	When the core is configured as RP, this output emulates the PCI legacy interrupts.
418		x4_intx_out[1]	
419		x4_intx_out[2]	

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
420		x4_intx_out[3]	Bit 0= INTA, Bit 1= INTB, Bit2= INTC, Bit 3= INTD
421		x4_phy_state_chg_sts_p_0	PHY interrupt output (per lane): Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
422		x2_correctable_error_out	The client may activate this input for one cycle to indicate a correctable error detected within the client logic that needs to be reported as an internal error through the PCI Express Advanced Error Reporting mechanism. In response, the Controller sets the Corrected Internal Error Status bit in the AER Correctable Error Status Register of all enabled Functions. This error is not considered Function-specific.
423		x2_fatal_error_out	masked. In multi-Function cores, this is the logical OR of the fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
424		x2_non_fatal_error_out	not masked. In multi-Function cores, this is the logical OR of the non-fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			Reporting is enabled in the Root Error Command register.
425		x2_local_interrupt	Local Error and Status Register Interrupt. This is an active high level interrupt. This will be asserted till the Local Error and Status Register is cleared by software. The interrupt_signals_sideband can be looked at to determine which Event triggered the Interrupt
426		x2_phy_interrupt_out	This is an active high, level interrupt output. This output is asserted by the Controller in the RP mode to signal one of the following link training-related events: The link bandwidth changed as a result of the change in the link width or operating speed and the change was initiated locally (not by the link partner), without the link going down. This interrupt is enabled by the Link Bandwidth Management Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Bandwidth Management Status bit of the Link Status Register. The link bandwidth changed autonomously as a result of the change in the link width or operating speed and the change was initiated by the remote node. This interrupt is enabled by the Link Autonomous Bandwidth Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Autonomous Bandwidth Status bit of the Link Status Register. The phy_interrupt_out output is not active when the Controller is configured as an Endpoint.
427		x2_intx_out[0]	When the core is configured as RP, this output emulates the PCI legacy interrupts. Bit 0= INTA, Bit 1= INTB,
428		x2_intx_out[1]	
429		x2_intx_out[2]	
430		x2_intx_out[3]	

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			Bit2= INTC, Bit 3= INTD
431		x1_0_correctable_error_out	The client may activate this input for one cycle to indicate a correctable error detected within the client logic that needs to be reported as an internal error through the PCI Express Advanced Error Reporting mechanism. In response, the Controller sets the Corrected Internal Error Status bit in the AER Correctable Error Status Register of all enabled Functions. This error is not considered Function-specific.
432		x1_0_fatal_error_out	masked. In multi-Function cores, this is the logical OR of the fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
433		x1_0_non_fatal_error_out	not masked. In multi-Function cores, this is the logical OR of the non-fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
434		x1_0_local_interrupt	Local Error and Status Register Interrupt. This is an active high level interrupt. This will be asserted till the Local Error and Status Register is cleared by software. The interrupt_signals_sideband can be looked at to determine which Event triggered the Interrupt

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
435		x1_0_phy_interrupt_out	This is an active high, level interrupt output. This output is asserted by the Controller in the RP mode to signal one of the following link training-related events: The link bandwidth changed as a result of the change in the link width or operating speed and the change was initiated locally (not by the link partner), without the link going down. This interrupt is enabled by the Link Bandwidth Management Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Bandwidth Management Status bit of the Link Status Register. The link bandwidth changed autonomously as a result of the change in the link width or operating speed and the change was initiated by the remote node. This interrupt is enabled by the Link Autonomous Bandwidth Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Autonomous Bandwidth Status bit of the Link Status Register. The phy_interrupt_out output is not active when the Controller is configured as an Endpoint.
436		x1_0_intx_out[0]	When the core is configured as RP, this output emulates the PCI legacy interrupts. Bit 0= INTA, Bit 1= INTB, Bit2= INTC, Bit 3= INTD
437		x1_0_intx_out[1]	
438		x1_0_intx_out[2]	
439		x1_0_intx_out[3]	
440		x1_1_correctable_error_out	The client may activate this input for one cycle to indicate a correctable error detected within the client logic that needs to be reported as an internal error through the PCI Express Advanced Error Reporting mechanism. In response, the Controller sets the Corrected Internal

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			Error Status bit in the AER Correctable Error Status Register of all enabled Functions. This error is not considered Function-specific.
441		x1_1_fatal_error_out	masked. In multi-Function cores, this is the logical OR of the fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
442		x1_1_non_fatal_error_out	not masked. In multi-Function cores, this is the logical OR of the non-fatal error status bits in the Device Status Registers of all Functions. In the RP mode, this output is activated on detection of a local fatal error, when its reporting is not masked. This signal also gets activated in response to an error message received from the link if Fatal Error Reporting is enabled in the Root Error Command register.
443		x1_1_local_interrupt	Local Error and Status Register Interrupt. This is an active high level interrupt. This will be asserted till the Local Error and Status Register is cleared by software. The interrupt_signals_sideband can be looked at to determine which Event triggered the Interrupt
444		x1_1_phy_interrupt_out	This is an active high, level interrupt output. This output is asserted by the Controller in the RP mode to signal one of the following link training-related events: The link bandwidth changed as a result of the change in the link width or operating speed and the change was initiated locally (not by the link partner), without the link going down. This

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			interrupt is enabled by the Link Bandwidth Management Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Bandwidth Management Status bit of the Link Status Register. The link bandwidth changed autonomously as a result of the change in the link width or operating speed and the change was initiated by the remote node. This interrupt is enabled by the Link Autonomous Bandwidth Interrupt Enable bit in the Link Control Register. The status of this interrupt can be read from the Link Autonomous Bandwidth Status bit of the Link Status Register. The phy_interrupt_out output is not active when the Controller is configured as an Endpoint.
445		x1_1_intx_out[0]	When the core is configured as RP, this output emulates the PCI legacy interrupts. Bit 0= INTA, Bit 1= INTB, Bit2= INTC, Bit 3= INTD
446		x1_1_intx_out[1]	
447		x1_1_intx_out[2]	
448		x1_1_intx_out[3]	
449		x2_1_1_phy_state_chg_sts_p_0	PHY interrupt output (per lane): Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
450		x2_1_1_phy_state_chg_sts_p_1	PHY interrupt output (per lane): Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
451		x2_1_1_phy_state_chg_sts_p_2	PHY interrupt output (per lane):

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			Asserted high when a power state or data rate change does not complete within a timeout. Cleared upon reading the associated lane's interrupt status register. Only valid for the master lane of the link. Slave lanes should be ignored
465		x8_pcie_ep_int	When the EP receives the CfgWr to change the power state of EP, the PCIe Controller will report the event through pcie0_ep_int
452	u_csi_wrapper	irq0	Information type Interrupt status (non-error conditions)
453		err_irq0	Datapath error interrupt status. Provides information about data path errors. The host processor can read the interrupt status register to identify the root cause of the event, typically after that the csi2rx_err_irq interrupt line is raised
454		irq1	Information type Interrupt status (non-error conditions)
455		err_irq1	Datapath error interrupt status. Provides information about data path errors. The host processor can read the interrupt status register to identify the root cause of the event, typically after that the csi2rx_err_irq interrupt line is raised
456		irq2	Information type Interrupt status (non-error conditions)
457		err_irq2	Datapath error interrupt status. Provides information about data path errors. The host processor can read the interrupt status register to identify the root cause of the event, typically after that the csi2rx_err_irq interrupt line is raised
458		irq3	Information type Interrupt status (non-error conditions)
459		err_irq3	Datapath error interrupt status. Provides information about data path errors. The host processor can read the interrupt status register to identify the root cause

SPI/PPI Num	IP BLOCK	Interrupt Signal Name on wrapper	Interrupt Description
			of the event, typically after that the csi2rx_err_irq interrupt line is raised
460		csi_brig_err_irq[0]	csi dma 0 interrupt
461		csi_brig_err_irq[1]	csi dma 1 interrupt
462		csi_brig_err_irq[2]	csi dma 2 interrupt
463		csi_brig_err_irq[3]	csi dma 3 interrupt
464	io_top_s0	PROCHOT_L	protect hot interrupt

1.5 DMA Map

Handshaking Signal	Function	Description
0	UART0 TX	
1	UART0 RX	
2	UART1 TX	
3	UART1 RX	
4	UART2 TX	
5	UART2 RX	
6	UART3 TX	
7	UART3 RX	
8	STM trigger	

Chapter 2 CPU

2.1 Overview

The P1 has a cluster with 8 Cortex-A720 and quad-core Cortex-A520 of which the cores are all single-threaded. Cortex-A520 processor, which is a mid-range, low-power processor that implements the ARMv9-A architecture. And the Cortex-A720 core is a high-performance and low-power Arm product that implements the Armv9-A architecture.

The processor includes the following features:

2.1.1.1 Core-features

- Implementation of the Armv9.2 A A64 instruction-set
- AArch64 Execution state at all Exception levels, EL0 to EL3
- Memory Management Unit (MMU)
- 40-bit Physical Address (PA) and 48-bit Virtual Address (VA)
- Generic Interrupt Controller (GIC) CPU interface to connect to an external interrupt distributor
- Generic Timers interface that supports 64-bit count input from an external system counter
- Implementation of the Reliability, Availability, and Serviceability (RAS) Extension
- Implementation of the Scalable Vector Extension (SVE) with a 128-bit vector length and the Scalable Vector Extension 2 (SVE2)
- Integrated execution unit with Advanced Single Instruction Multiple Data (SIMD) and floating point support
- Support for the optional Cryptographic Extension
- Activity Monitoring Unit (AMU)

2.1.1.2 Cache features

- Separate L1 data and instruction caches
- Private, unified data and instruction L2 cache
- error protection with parity or Error Correcting Code (ECC) allowing Single Error Correction and Double Error Detection (SECDED) on L1 instruction and data caches, L2 cache, and L2 Translation Lookaside Buffer (TLB)
- Support for Memory System Resource Partitioning and Monitoring (MPAM)

2.1.1.3 Debug features

- Armv9.2-A debug logic
- Performance Monitoring Unit (PMU)
- Embedded Trace Macrocell (ETM) with support for Embedded Trace Extension (ETE)
- Trace Buffer Extension (TRBE)
- implementation of the Statistical Profiling Extension (SPE)
- Embedded Logic Analyzer (ELA)

The configuration details are shown in the following table.

Table 2-1 CPU Configuration

number of little core(A520)	4
number of middle core(A720)	4
number of big core(A720)	4
L1 I cache size of A520	32K
L1 D cache size of A520	32K
L1 I cache size of A720	64K
L1 D cache size of A720	64K
L2 cache size of A720	512K
L3 cache size	12M
CPU cache protection	yes
DSU L3 cache protection	yes
Bus master interface	CHI
NEON and SVE2	Yes
Cryptography extension	yes

2.2 Block Diagram

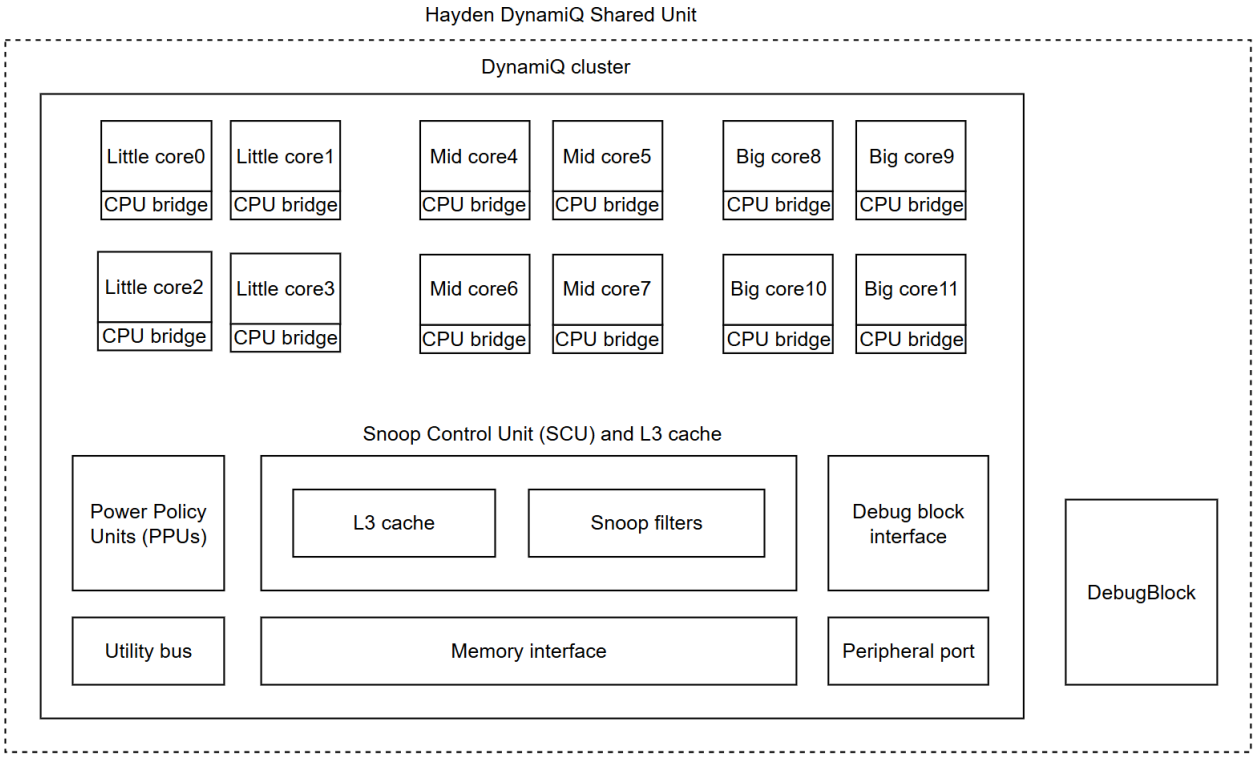


Figure 2-1 Block Diagram

2.3 Function Description

Please refer to the document cortex_a720_r0p1_trm.pdf, cortex_a520_r0p1_trm.pdf and dsu_r2p0_trm.pdf for the detailed function description.

Chapter 3 GPU

3.1 Overview

The Immortalis-G720 GPU is a graphics acceleration platform that is based on open standards. It supports 3D graphics and General Purpose computing on GPU (GPGPU).

3.1.1 Features

The Arm Immortalis- G720 GPU has significant features and properties.

- Support of multiple clock and voltage domains, allowing you to maximize performance within a given power budget.
- API feature set with support for shader-based and fixed-function graphics APIs.
- Anti-aliasing capabilities.
- Effective cores for General Purpose computing on GPU (GPGPU) applications.
- High memory bandwidth and low power consumption for 3D graphics content.
- Performance leading 3D graphics.
- AMBA 4 AXI-Lite slave interface for GPU configuration.
- 256-bit AMBA 4 ACE master interface for external memory access.
- Easy integration.
- Latency tolerance.
- Compressed texture formats.
- Configurable per-core power management for enabling the optimal power and performance combination for each application.
- Coherency-aware interconnects for system memory and resource sharing.
- Arm Frame Buffer Compression (AFBC) 1.3.
- 8-bit, 10-bit, and 16-bit YUV input and output formats.
- Secure processing of DRM-protected content.
- Support Serial Wire debug for embedded MCU
- Deferred Vertex Shading (DVS)
- Dynamic Voltage and Frequency Scaling (DVFS)
- Dynamic Core Scaling
- Support for larger tiles of up to 64 x 64
- Increased blending throughput for Multi-Sampling Anti-Aliasing (MSAA) use cases
- Increased texture-mapper throughput for 64bpp content
- Ability to power gate the Ray-Tracing Unit (RTU)
- Alignment of clock, reset, and power domains for shader stacks
- Increased depth of the compute task queue (from 4 to 16 tasks)
- Support for Vulkan dynamic buffer descriptors
- Support for subbanked RAMs

3.1.2 Standard

The GPU hardware and software support compute standards and graphics API standards.
The GPU supports these compute API standards:

- OpenCL 3.0 Full Profile

The GPU supports these graphics API standards:

- OpenGLES 1.1, 2.0, and 3.2
- Vulkan 1.4

3.2 Function Description

The GPU is a modular system that contains the shader processor cores and associated blocks. These blocks include the tiler, the Memory Management Unit (MMU), and the Level 2 (L2) cache memory subsystem.
The GPU comprises the following functional block types:

- Command Stream Frontend (CSF)
  This block issues and sends commands to the specific functional blocks in the GPU.
- Shader core
  These blocks form the main processing engine of the GPU. The shader cores carry out all rendering and computation operations.
- Tiler
  This block creates lists of all the objects in a scene, so that the shader cores process the objects efficiently.
- MMU
  This block translates virtual addresses to physical addresses for external bus interface access.
- L2 cache memory subsystem
  This block provides caching for all internal manager blocks. It also supports cache coherency for caches inside and outside of the GPU. The L2 cache sits between the internal manager blocks and the external interconnect. All transactions go through the cache. If the cache cannot service an internal transaction, it performs an external transaction to read or write data.

Chapter 4 VPU

4.1 Overview

The VPU includes the following features:

4.1.1 Video Encoding

The video processor supports several video encoding formats.

4.1.1.1 General Encoding Features

- The VPU supports the following features:
- You can configure Arm Frame Buffer Compression (AFBC) 1.0 or 1.2 input.
- Support for YUV422 and YUV420 AFBC block split for 16x16.
- Stride support. This is not applicable to AFBC input formats.
- Horizontal and vertical mirroring. This is not applicable to AFBC input formats.
- Support for optional rotation of the source frame in 90-degree steps before encoding. This is not applicable to AFBC input formats.

Note:
If YUV422 is rotated by 90 degrees or 270 degrees and not converted to YUV420, the result is encoded as YUV440.

The VPU supports encoding the following source-frame input formats:

- 1-plane YUV422, scan-line format, interleaved in YUYV or UYVY order.

Note:

YUV422 inputs can be converted to YUV420 format.

- 1-plane RGB, 8-bit, byte address order RGBA, BGRA, ARGB, or ABGR.
- 3-plane RGB, 8-bit, plane order RGB.
- 2-plane YUV420, scan-line format, chroma interleaved in UV or VU order.
- 3-plane YUV420, scan-line format.
- 3-plane YUV420 10-in-16 bit scan line format.

Note:
Support for 3-plane formats is included to help you with testing. Do not use these for maximum performance.

- AFBC YUV422.
- AFBC YUV420.
- YUV420 10-bit, P010, little endian, MSB aligned.

4.1.1.2 Supported Codecs

The following codec standards are supported for encoding:

- HEVC (H.265) Main, Main Still Profile
- HEVC (H.265) Main 10 Profile
- H.264 Baseline Profile (BP)
- H.264 Main Profile (MP)

- H.264 High Profile (HP)
- H.264 High 10 Profile
- VP8
- VP9 Profile 0
- VP9 Profile 2 at 10-bit depth
- JPEG/ MJPEG, baseline sequential, YUV420 and YUV422

Note:

Only the JPEG codec can encode YUV422.

4.1.1.3 HEVC (H.265) Encoding Features

- The encoded bitstream complies with the HEVC (H.265) Main and Main 10 Profiles.
- Encoding speed of:
  - 4320p30 for four cores at 1200MHz.
- Bitrates up to:
  - 240MBit/s for four cores at 1200MHz.
- Maximum frame width of 8,192 pixels.
- Maximum frame height of 8,192 pixels.
- 8-bit or 10-bit sample depth with I, P and B frames.
- Progressive encoding with 64 x 64 CTU size.
- Tiled made, up to four tiles, horizontal splits only.
- Wavefront parallel encoding.
- The Motion Estimation (ME) search window is +128 pixels horizontally, +64 pixels vertically.
- ME search down to Quarter Picture ELement (QPEL) resolution.
- 8 x 8, 16 x 16 and 32 x 32 luma intra-modes.
- 4 x 4, 8 x 8 and 16 x 16 chroma intra-modes.
- 8 x 8, 16 x 16 and 32 x 32 inter-modes.
- 4 x 4, 8 x 8, 16 x 16 and 32 x 32 transform size for luma.
- 4 x 4, 8 x 8 and 16 x 16 transform size for chromas.
- Skipped CUs, Merge modes.
- Deblocking.
- Sample Adaptive Offset (SAO).
- Constrained intra-prediction selectable.
- Fixed Quantization Parameters (QP) operation or rate-controlled operation.
- Rate controlled based on bitrate and buffer size settings. This is also known as leaky bucket.
- Long term reference frame.
- Selectable intra-frame refresh interval.
- Slice insertion on a CTU row granularity.
- Possible to limit the search window and the split options.
- Encoders do not prevent the output from exceeding the maximum number of bits per CTU.
- Gradual Decoding Refresh (GDR).
- Scalable Video Coding (SVC) with Temporal Layer 3.
- Scene change detection with adaptive intra block.

4.1.1.4 H.264 Encoding Features

- The encoded bitstream complies with the Baseline, Main, High and High 10 Profiles.
- Encoding speed of 4320p30, four cores at 1200MHz.

- Bitrates up to 240MBit/s, four cores at 1200MHz.
- Maximum frame width of 8,192 pixels.
- Maximum frame height of 8,192 pixels.
- I, P and B frames.
- Progressive encoding.
- Context Adaptive Binary Arithmetic Coding (CABAC) or Context Adaptive Variable Length Coding (CAVLC) entropy coding.

Note:

B frames are not supported with CAVLC entropy coding.

- The Motion Estimation (ME) search window is ±128 pixels horizontally, ±64 pixels vertically.
- ME search down to Quarter Picture Element (QPEL) resolution.
- All 4 x 4, 8 x 8, 16 x 16 luma and 8 x 8 chroma intra-modes evaluated.
- 8 x 8 and 16 x 16 inter-modes.
- 4 x 4 and 8 x 8 transform.
- Skipped macroblocks.
- Deblocking.
- Constrained intra-prediction selectable.
- Fixed QP operation or rate-controlled operation.
- Rate controlled based on bitrate and buffer size settings. This is also known as leaky bucket.
- Long term reference frame.
- Selectable intra-frame refresh interval.
- Slice insertion on 32-pixel high row granularity.
- Possible to limit the search window and the macroblock split options.
- Escaping to prevent Network Abstraction Layer (NAL) unit start code emulation. This is always enabled and is independent of the NAL packet format setting.
- Gradual Decoding Refresh (GDR).
- Scalable Video Coding (SVC) with Temporal Layer 3.
- Scene change detection with adaptive intra block.

Note:

Encoders do not prevent the output from exceeding the maximum number of bits per macroblock.

4.1.1.5 VP8 Encoding Features

- Encoding speed of 1080p360, four cores at 1200MHz.
- Bitrates up to 50MBit/s per 1080p30 session.
- Maximum frame width of 2,048 pixels.
- Maximum frame height of 2,048 pixels.
- land P frames.
- Progressive encoding.
- The ME search window is ±128 pixels horizontally, ±64 pixels vertically.
- ME search down to QPEL resolution.
- All 4 x 4 and 16 x 16 luma and 8 x 8 chroma intra-modes evaluated.
- 8 x 8 and 16 x 16 inter-modes.
- Skipped macroblocks.
- Deblocking.
- Fixed QP operation or rate-controlled operation.

- Rate controlled based on bitrate and buffer size settings. This is also known as leaky bucket.
- Selectable intra-frame refresh interval.
- Possible to limit the search window and the macroblock split options.

4.1.1.6 VP9 encoding features

- The encoded bitstream complies with the VP9 Profile 0 at 8-bit depth, or Profile 2 at 10-bit depth.
- Encoding speed of 4320p30, four cores at 1200MHz.
- Bitrates up to 240MBit/s, four cores at 1200MHz.
- Maximum frame width of 8,192 pixels.
- Maximum frame height of 8,192 pixels.
- 8-bit or 10-bit sample depth. I and P frames.
- Progressive encoding.
- Tiled rows and columns.
- The Motion Estimation (ME) search window is ± 128 pixels horizontally and ± 64 pixels vertically.
- ME search down to Quarter Picture ELement (QPEL) resolution.
- 8 x 8, 16 x 16 and 32 x 32 luma intra-modes.
- 4 x 4, 8 x 8 and 16 x 16 chroma intra-modes.
- 8 x 8, 16 x 16 and 32 x 32 inter-modes.
- 8 x 8, 16 x 16 and 32 x 32 transform size for luma.
- 4 x 4, 8 x 8 and 16 x 16 transform size for chroma.
- Skipped superblocks.
- Deblocking.
- Fixed QP operation or rate-controlled operation.
- Rate controlled based on bitrate and buffer size settings, also known as a leaky bucket.
- Selectable intra-frame refresh interval.
- Implicit or explicit probability update, using delayed contexts.

4.1.1.7 JPEG Encoding Features

- Encoding speed of 4320p40 for 4:2:2, four cores at 1200MHz.
- Bitrates up to 450MBit/s for 4:2:2, four cores at 1200MHz.
- 4:2:0 and 4:2:2 baseline sequential.
- Programmable full set of Huffman symbols, separate for luma and chroma.
- Programmable full set of quantization matrices, separate for all color components.
- Programmable DC restart interval.
- Maximum frame width of 16384 pixels.
- Maximum frame height of 16384 pixels.

4.1.2 Video Decoding

The video processor supports several video codecs for decoding.

4.1.2.1 General Decoding Features

Supported the following source frame output formats:

- 2-plane YUV420 scan line format, chroma interleaved in UV or VU order.
- 3-plane YUV420 scan line format.

Note:

Support for 3-plane formats is included to help you with testing. Do not use these for maximum performance.

- YUV420.10-bit, P010, little endian, MSB aligned.

Note:

The YUV buffer and stride must have correct alignment for maximum performance.

- YUV420 AFBC format, 8-bit, and 10-bit.
  You can configure AFBC.1.0 or AFBC 1.2 output.
- 1-plane RGB 8-bit format, byte address order RGB.
- 3-plane RGB 8-bit format, plane order RGB.

Note:

The RGB 8bit format is converted from YUV420, do not use these for maximum performance.

- Stride support for scan-line formats only.
- Support for rotation of the decoded frame in 90-degree steps before output. This is not applicable for AFBC output formats.
- Support for downscaling the decoded frame horizontally and vertically by a factor of 2 or 4. For AFBC formats only a factor of 2 is supported.
- Support for downscaling the decoded frame horizontally and vertically by any factor individually. No AFBC formats are supported for the downscaling output. The smallest resolution can be 2x2 pixels.
- 2x2 downscaling is supported with AFBC output.
- Support for output of average frame luminance and chrominance for each 32x32 pixel block, for every output display frame.

4.1.2.2 Supported Codecs

The following codec standards are supported for decoding:

- AV1 Main
- HEVC (H.265) Main/Main10.
- H.264 Baseline, Main, High, and High10 Progressive Profile.
- VP8.
- VP9 Profile 0 and VP9 Profile 2 at 10-bit depth, not 12-bit.
- JPEG, baseline sequential, YUV420 and YUV422.

Note

Only the JPEG codec can decode YUV422.

- MPEG4 SP/ASP.
- MPEG2 MP.
- H.263 Profile 0.

4.1.2.3 AV1 Decoding Features

- The decoder is fully compliant to the Main Profiles.
- Decoding speed equals 4320p60, four cores at 1200MHz.
- Average bitrates up to 480MBits/s, four cores at 1200MHz.
- Maximum frame width is 8,192 pixels.
- Maximum frame height is 8,192 pixels.
- Error concealment is performed in case of bit errors.
- Stream parameter information is output.

4.1.2.4 HEVC (H.265) Decoding Features

- The decoder is fully compliant to the Main and Main10 Profiles.
- Decoding speed equals 4320p60, four cores at 1200MHz.
- Average bitrates up to 480MBit/s, four cores at 1200MHz.
- Maximum frame width is 8,192 pixels.
- Maximum frame height is 8,192 pixels.

- Error concealment is performed in case of bit errors.
- Stream parameter information is output.

4.1.2.5 H.264 Decoding Features

- The decoder is fully compliant to H.264 Baseline, Main, High, and High 10 progressive Profiles.
- If Flexible Macroblock Ordering (FMO) or Arbitrary Slice Ordering (ASO) are used for baseline streams, the decoding speed is WVGA at 30fps for a single core at 400MHz.
- If FMO and ASO are not used, the decoding speeds are:
  - 4320p60 for four cores at 1200MHz.
  - 1080i720 for four cores at 1200MHz.
- For progressive streams
- Average bitrates up to 480MBit/s for four cores at 1200MHz.
- Maximum frame width is 8,192 pixels.
- Maximum frame height is 8,192 pixels.
- For interlaced streams
  - Average bitrates up to 360MBit/s for four cores at 1200MHz.
  - A maximum frame width of 2,048 pixels.
- Indent, this is for interlaced streams.
- Error concealment is performed in case of bit errors.
- Stream parameter information is output.

Escaping to prevent NAL unit start code emulation is always expected. This is independent of the NAL packet format setting.

4.1.2.6 VP8 Decoding Features

- The decoder is fully compliant to the VP8 Specification.
- Decoding speed equals 1080p360, four cores at 1200MHz.
- Average bitrates up to 50MBit/s per 1080p30 session, four cores at 1200MHz.
- Maximum frame width is 2,048 pixels.
- Maximum frame height is 2,048 pixels.
- Error concealment is performed in case of bit errors.

4.1.2.7 VP9 Decoding Features

- The decoder is fully compliant to Profile 0 and Profile 2, 10-bit. Profile 2 at 12-bit is not supported.
- Decoding speed equals 4320p60, four cores at 1200MHz assuming no non-visible, no AltRef frames.
- Decoding speed equals 4320p40, four cores at 1200MHz assuming an AltRef distance of 4.
- Average bitrates up to 480MBit/s, four cores at 1200MHz.
- Maximum frame width is 8,192 pixels.
- Maximum frame height is 8,192 pixels.
- Error concealment is performed in case of bit errors.
- Stream parameter information is output.

4.1.2.8 JPEG Decoding Features

- Decoding speed equals 4320p30, four cores at 1200MHz.
- Average bitrates up to 240MHz, four cores at 1200MHz.

- Support for 4:2:0 output by the video encoder, baseline sequential YYYYCbCr interleaved MCU format only.
- Support for 4:2:2 output by the video encoder, baseline sequential YYCbCr interleaved MCU format only.
- Maximum frame width is 32768 pixels.
- Maximum frame height is 32768 pixels.
- Error concealment is performed in case of bit errors.

4.1.2.9 MPEG4 Decoding Features

- The decoder is compliant to MPEG4 Simple Profile and Advanced Simple Profile.
- Global Motion Compensation (GMC) is limited to a single warp point.
- Decoding speed equals 1080p360, or 1080i720, four cores at 1200MHz.
- Average bitrates up to 50MBit/s, four cores at 1200MHz.
- Maximum frame width is 2,048 pixels.
- Maximum frame height is 2,048 pixels.
- Error concealment is performed in case of bit errors.

4.1.2.10 MPEG2 Decoding Features

- The decoder is compliant to MPEG2 Main Profile.
- Decoding speed equals 1080p360, or 1080i720, four cores at 1200MHz.
- Average bitrates up to 50MBit/s, four cores at 1200MHz.
- Maximum frame width is 4,096 pixels. A maximum frame width of 2048 pixels for interlaced streams.
- Maximum frame height is 4,096 pixels.
- Error concealment is performed in case of bit errors.

4.1.2.11 H.263 Decoding Features

- The decoder is compliant to H.263 Profile 0.
- Decoding speed equals 1080p360, four cores at 1200MHz.
- Average bitrates up to 50Mbit/s, four cores at 1200MHz.
- Maximum frame width is 2,048 pixels.
- Maximum frame height is 2,048 pixels.
- Error concealment is performed in case of bit errors.

Chapter 5 DPU

5.1 Function Description

5.1.1 Display Layers

The display processor supports read accesses from and write accesses to the system for various graphic and video content types.
The display processor reads and writes each layer from and to a separate frame buffer in system memory. The display processor supports various RGB and YUV pixel formats.
The layer processing units in the display processor can each read four layers at the same time from system memory. Each read layer pipeline provides:

- Pixel unpacking from the 128-bit AXI words.
- RGB to RGB transform.
- Inverse transform (gamma) operation.
- Forward transform (gamma) operation. The categories of layers are:

5.1.1.1 Rich Layer

A rich layer supports both RGB and YUV pixel formats. The rich layer can implement YUV to RGB conversion.

5.1.1.2 Simple Layer

A simple layer only supports RGB pixel formats.
The display processor provides one write-back layer pipeline for each layer processing unit.
The AFBC DMA unit supports read access of most of the uncompressed RGB and YUV layer formats that the layer processing unit supports, and writes back in common AFBC formats.

5.1.2 Display Layer Resolutions

For each display layer, the display processor supports RGB and YUV pixel formats at various resolutions.
The display processor uses arbitrary resolutions in the layer processing units for real-time DMA transfers.
The display processor provides arbitrary cropping of layers through a combination of:

- Address pointer.
- Horizontal and vertical size.
- AFBC decoder cropping controls.

The AFBC DMA unit supports these resolutions:

- 16 byte-aligned addressing and reading of layers of arbitrary resolution (that are multiples of 16 bytes for a given pixel format) with aligned addressing.
- An arbitrary cropping (for multiples of 16 bytes for a given pixel format) of the layers, through a combination of:
    - Address pointer.
    - Horizontal and vertical size.

5.1.3 Pixel Format Summary

The display processor supports pixel formats for various RGB and YUV data.
There are various pixel formats in the display processor that support different resolutions.

Note
All format groups and codes that are not listed are reserved.

ARGB 32 bits per pixel, 10-bit sample, 2bit alpha

FORMAT[5:3] = 0b000
Support on all read and write layers of the layer processing unit. Support for non-compressed formats.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b000	ARGB_2101010	N/A	N/A	N/A	Read
0b001	ABGR_2101010	16 by 16 32 by 32	Optional	Always	Read, write
0b010	RGBA_1010102	N/A	N/A	N/A	Read
0b011	BGRA_1010102	N/A	N/A	N/A	Read

ARGB 32 bits per pixel, 8-bit sample, 8-bit alpha

FORMAT[5:3] = 0b001
Support on all read and write layers of the layer processing unit. Support for non-compressed formats.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b000	ARGB_8888	N/A	N/A	N/A	Read
0b001	ABGR_8888	16 by 16 32 by 32	Optional	Always	Read, write
0b010	RGBA_8888	N/A	N/A	N/A	Read
0b011	BGRA_8888	N/A	N/A	N/A	Read

XRGB 32 bits per pixel, 8-bit sample, wasted byte

FORMAT[5:3] = 0b010
Support on all read and write layers of the layer processing unit. Support for non-compressed formats.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b000	XRGB_8888	N/A	N/A	N/A	Read
0b001	XBGR_8888	N/A	N/A	N/A	Read, write
0b010	RGBX_8888	N/A	N/A	N/A	Read
0b011	BGRX_8888	N/A	N/A	N/A	Read

RGB 24 bits per pixel

FORMAT[5:3] = 0b011
Support on all read and write layers of the layer processing unit. Support for non-compressed formats.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b000	RGB_888	N/A	N/A	N/A	-
0b001	BGR_888	16 by 16 32 by 32	Optional	Always	-

RGB 16 bits per pixel

FORMAT[5:3] = 0b100
Support on all read layers of the layer processing unit. Support for non-compressed formats.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b000	RGBA_5551	N/A	N/A	N/A	Read
0b001	ABGR_1555	16 by 16 32 by 32	Never	Never	Read, write
0b010	RGB_565	N/A	N/A	N/A	Read
0b011	BGR_565	16 by 16 32 by 32	Never	Never	Read, write

YUV 422, YUV 420, 8-bit sample

FORMAT[5:3] = 0b101 Support on layer 0 and layer 2 in the layer processing unit. For YUV_420_p2_8, support for layer 0, layer 2, and layer write-back. Support for non-compressed data in all formats except YUV_422_P2_8.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b001	YUV_422_P2_8 (AFBC only)	16 by 16	Never	N/A	Write
0b010	VYUY_422_P1_8	N/A	N/A	N/A	Read
0b011	YVYU_422_P1_8	N/A	N/A	N/A	Read
0b110	YUV_420_P2_8	16 by 16	Never	N/A	Read, write
0b111	YUV_420_P3_8	N/A	N/A	N/A	Read

YUV 422, YUV 420, 10-bit sample

FORMAT[5:3] = 0b110
Support on layer 0 and layer 2 in the layer processing unit. Support for non-compressed formats.

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b110	YUV_420_P1_10 (Y0L2)	N/A	N/A	N/A	-

FORMAT[2:0]	Format Name	AFBC Super Block Size	AFBC Block Split (16 by 16)	AFBC Block Split (32 by 8)	AFBC DMA Unit
0b111	YUV_420_P2_10 (P010)	16 by 16	Never	N/A	Read, write

5.1.3.1 Input Pixel Formats for Rich Layers

The display processor supports various RGB and YUV pixel formats in the layer processing unit. The rich layers in the layer processing unit support:

- ABGR 2-10-10-10, ARGB 2-10-10-10, RGBA 10-10-10-2, BGRA 10-10-10-2 (32bpp).
- ABGR 8-8-8-8, ARGB 8-8-8-8, RGBA 8-8-8-8, BGRA 8-8-8-8 (32bpp).
- XBGR 8-8-8-8, XRGB 8-8-8-8, RGBX 8-8-8-8, BGRX 8-8-8-8 (32bpp).
- RGB 8-8-8, BGR 8-8-8 (24bpp).
- RGBA 5-5-5-1, ABGR 1-5-5-5, RGB 5-6-5, BGR 5-6-5 (16bpp).
- 8-bit YUV 4:2:2 2-plane (AFBC only).
- 8-bit YUV 4:2:2 1-plane (VYUY, YVYU).
- 8-bit YUV 4:2:0 2-plane.
- 8-bit YUV 4:2:0 3-plane.
- 10-bit YUV 4:2:0 1-plane (ARM Y0L2).
- 10-bit YUV 4:2:0 2-plane (P010).

5.1.3.2 Input Pixel Formats for Simple Layers

The display processor supports various RGB and YUV pixel formats in the layer processing unit. The simple layers in the layer processing unit support:

- ABGR 2-10-10-10, ARGB 2-10-10-10, RGBA 10-10-10-2, BGRA 10-10-10-2 (32bpp).
- ABGR 8-8-8-8, ARGB 8-8-8-8, RGBA 8-8-8-8, BGRA 8-8-8-8 (32bpp).
- XBGR 8-8-8-8, XRGB 8-8-8-8, RGBX 8-8-8-8, BGRX 8-8-8-8 (32bpp).
- RGB 8-8-8, BGR 8-8-8 (24bpp).
- RGBA 5-5-5-1, ABGR 1-5-5-5, RGB 5-6-5, BGR 5-6-5 (16bpp).

5.1.3.3 Input Pixel Formats for the AFBC DMA Unit

The display processor supports various RGB and YUV pixel formats in the AFBC DMA unit. The AFBC DMA unit supports reading:

- ABGR 2-10-10-10, ARGB 2-10-10-10, RGBA 10-10-10-2, BGRA 10-10-10-2 (32bpp).
- ABGR 8-8-8-8, ARGB 8-8-8-8, RGBA 8-8-8-8, BGRA 8-8-8-8 (32bpp).
- XBGR 8-8-8-8, XRGB 8-8-8-8, RGBX 8-8-8-8, BGRX 8-8-8-8 (32bpp).
- RGBA 5-5-5-1, ABGR 1-5-5-5, RGB 5-6-5, BGR 5-6-5 (16bpp).
- 8-bit YUV 4:2:2 1-plane (VYUY, YVYU).
- 8-bit YUV 4:2:0 2-plane.
- 8-bit YUV 4:2:0 3-plane.
- 10-bit YUV 4:2:0 2-plane (P010).

5.1.3.4 Output Pixel Formats for Write-Back Layers

The display processor write-back layer supports various RGB and YUV formats. The write-back layer in the layer processing unit supports:

- ABGR 2-10-10-10, ARGB 2-10-10-10, RGBA 10-10-10-2, BGRA 10-10-10-2 (32bpp)
- ABGR 8-8-8-8, ARGB 8-8-8-8, RGBA 8-8-8-8, BGRA 8-8-8-8 (32bpp)
- XBGR 8-8-8-8, XRGB 8-8-8-8, RGBX 8-8-8-8, BGRX 8-8-8-8 (32bpp)
- RGB 8-8-8, BGR 8-8-8 (24bpp)
- YUV 4:2:0 2-plane 8-bit.

5.1.3.5 Output Pixel Formats for the AFBC DMA Unit

The display processor write-back layer supports various RGB and YUV formats. The AFBC DMA unit write-back supports:

- BGRA 10-10-10-2 (32bpp)
- ABGR 8-8-8-8
- XBGR 8-8-8-8,
- BGR 8-8-8 (24bpp)
- ABGR 1-5-5-5, BGR 5-6-5 (16bpp).
- 8-bit YUV 4:2:2 2-plane.
- 8-bit YUV 4:2:0 2-plane.
- 10-bit YUV 4:2:0 2-plane (P010).

5.1.4 AFBC Decoder

The layer processing unit of the display processor integrates an *Arm Frame Buffer Compression* (AFBC) decoder subsystem. This subsystem integrates two AFBC decoder cores.

The AFBC decoding subsystem has these features:

- Decompression possible for all four display layers. The display processor can decompress:
  - Up to two YUV layers and up to four RGB layers.
- Supports custom resolutions:
  - From 4 by 4 up to 2048 by 4096 pixels for YUV layers.
  - From 4 by 4 up to 4096 by 4096 pixels for RGB layers.
- Supported superblock layouts:
  - Superblock layout 0, 16 by 16 tiled RGB formats with up to 2048 by 4096 pixels.
  - Superblock layout 1, 16 by 16 tiled planar YUV 4:2:0 formats with up to 2048 by 4096 pixels.
  - Superblock layout 2, 16 by 16 planar YUV 4:2:2: up to 2048 by 4096 pixels.
  - Superblock layout 3, 32 by 8 tiled RGB formats with 17 or more bpp:
    - with up to 4096 by 4096 pixels.
  - Superblock layout 4, 32 by 8 tiled RGB formats with 16 or less bpp:
    - with up to 4096 by 4096 pixels.
  - Multiple RGB and YUV pixel formats. The table lists the support for pixel formats:

Table 5-1 Pixel Formats Supported by the Afbc Decoder

Format name	Format code	Block split support
ABGR_2101010	6'h01	Yes
ABGR_8888	6'h09	Yes
BGR_888	6'h19	Yes
ABGR_1555	6'h21	No
BGR_565	6'h23	No
YUV_422_P2_8	6'h29	No
YUV_420_P2_8	6'h2E	No

Format name	Format code	Block split support
YUV_420_P2_10	6'h37	No

- Block split mode support for RGB 32bpp formats.
- Payload fetching in 16×8 blocks and 32 by 4 pixels for RGB 32bpp formats.
- Payload fetching in 16×16 blocks for RGB 16bpp, YUV 4:2:2, and YUV 4:2:0 formats.
- Support for rotation by 90º, 180º, and 270º, and horizontal and vertical flip of the decoded layers.
- Support for cropping of decoded layers.

The AFBC decoder does not support de-interlacing (line replication) on the output. For AFBC compressed interlaced video layers, use the scaling engine to perform de-interlacing.
The maximum read issuing capability of the AFBC decoder is 64.

5.1.5 AFBC Encoder

The display processor integrates an *Arm Frame Buffer Compression* (AFBC) DMA Unit. This unit is for the compression of uncompressed layers, that require 90º or 270º rotation before composition, to achieve optimum system memory access.
The display processor implements memory-to-memory AFBC encoding functionality. This functionality improves the behavior of system memory access for real-time DMAs implemented in the layer processing units. The AFBC DMA unit features:

- Compression of a single layer at a time.
- Superblock layouts:
  - Superblock layout 0, 16 by 16 tiled RGB formats.
  - Superblock layout 1, 16 by 16 tiled planar YUV 4:2:0 formats.
  - Superblock layout 2, 16 by 16 planar YUV 4:2:2.
- No support for block split mode.
- Multiple RGB and YUV pixel formats.
- Payload is always produced in 16x16 blocks for all formats.
- Does not support rotation or flip of input pixels; the layers are rotated while reading from the layer processing units.
- Cropping is limited to AXI words, does not support cropping of pixels within 128-bit AXI words, and does support cropping of AXI words.
- Custom resolutions from 4 by 4, up to 4096 by 4096 pixels in each configuration. The maximum read issuing and the maximum write issuing capability is 16.

5.1.6 Memory Pointers, Stride, and Cropping

Each display layer has a preserved 40-bit base address register, which indicates the start of data for the layer.
For layers in the layer processing unit, the base address does not have to be aligned to a 128-bit data word and layer cropping within an AXI word is supported.
For the read layer in the AFBC DMA unit, the base address must be aligned to a 128-bit data word and layer cropping within an AXI word is not supported.
The software driver can program the memory pointers. The software driver can change the pointers in each frame through validation of a new configuration.
In addition to the memory pointer of each layer, there is a programmable 17-bit memory stride. This stride represents the offset in bytes from the base address corresponding to one line. The memory

stride must align to a 128-bit data word (16 bytes). The 17-bit memory stride is interpreted as a value in twos complement representation.
When layer cropping is required, you must consider the layer memory pointer, memory stride, input horizontal, and vertical sizes. The layer processing units support arbitrary combinations of these parameters to achieve any custom cropping. The only limitations are:

- For YUV 4:2:0 and YUV 4:2:2 pixel formats, where you must consider the chroma layout.
- For YUV 4:2:2 formats, horizontal cropping must be a multiple of two pixels.
- For YUV 4:2:0 formats, horizontal and vertical cropping must both be a multiple of two pixels.

5.1.7 Memory Write-Back

The display processor permits composition results to be written back to system memory.
The display processor implements functionality to write composition results back to system memory. To enable this functionality, software must configure:

- The composition unit, to pass the composition results to the layer processing unit.
- The layer processing unit, to pack pixel streams into memory words and write this data back to system memory.

The composition unit supports downscaling of composition results before passing the result to the layer processing unit. For composition results with horizontal resolution beyond 2048 pixels, enable the internal split functionality.
The internal precision of the write-back layer pipeline in the layer processing unit is 12 bits. The write- back layer pipeline supports color space conversion from RGB to YUV 4:2:0. For YUV formats, the processor subsamples every other sample to provide decimation of UV samples.
The display processor supports a software programmable layer alpha for formats with alpha. Software programs this layer through the LALPHA bitfield of the LW_CONTROL register. The field is 8-bits wide. When software programs an RGB 32bpp format with 2-bit alpha, the 2-bit alpha must be in the two MSBs of the LALPHA bitfield.
Software can enable the write-back functionality in one frame mode through the OFM bit in the LW_CONTROL register of the layer processing unit. In this mode, the display processor writes the composition result back to system memory only when software programs a new scene.
The trusted layer cannot be written back to memory. The composition unit always passes composition results back to the layer processing unit before it composes the trusted layer.

5.1.7.1 Memory Write-Back Limitations

The display processor permits composition results to be written back to system memory. Write-back has limitations that depend on the configuration.
The display processor functionality to write composition results back to system memory is limited:

- The display processor cannot write the trusted layer back to memory. The composition unit always passes composition results back to the layer processing unit before it composes the trusted layer.

5.1.8 Rotation, Horizontal and Vertical Flips

The display processor can read AFBC compressed layers in any supported rotation and flip combination when encoded in superblock layout 0, 1, and 2. The display processor integrates an AFBC DMA unit to pre-encode uncompressed layers with AFBC before 90º or 270º rotation.
The display processor provides rotation by 180º, and horizontal and vertical flips for uncompressed layers through the memory subsystem and latency buffers subsystem. The display processor can read:

- AFBC compressed layers in superblock layout 3 and 4.

- AFBC uncompressed layers that are rotated by 180º.

The display processor can support flip horizontally and vertically on the fly.
The AFBC decoder block performs rotation and flips of AFBC compressed layers.
Rotation angle, or layer orientation on the active screen, is programmable in the design and can be different for each display layer.
The AFBC decoder subsystem decodes, writes, and rotates the layers using an output buffer (on-chip SRAM memory). This buffer consists of four instances and is distributed between the display layers. The processor can rotate and AFBC decode all layers simultaneously. Support for any given size layer depends on format and configuration options.

5.1.8.1 AFBC Memory Configurations

The display processor can read AFBC compressed layers in any supported rotation and flip combination when encoded in superblock layout 0, 1, and 2. The display processor requires certain AFBC memory block sizes.
The AFBC output buffer memory for each layer processing unit is fixed at 256KB . These memory sizes constrain the maximum line sizes of the decoded or rotated layers.
The table shows the AFBC SRAM sizes to enable buffering of 4, 8, or 16 pixel lines, after AFBC decoding and, optionally, rotation.

Table 5-2 Afbc Ram Blocks for Default Configurations

Layers	AFBC RAM Size Per Layer	Format	Superblock Layout	Lines Stored	Maximum Horizontal Size
LPU0 Layer 0 LPU0 Layer 2 LPU1 Layer 0 LPU1 Layer 2	64KB	RGB 32bpp (or 24bpp)	SL-0 (16x16)	8	2048
		RGB 32bpp (or 24bpp)	SL-3 (32x8)	4	4096
		RGB 16bpp	SL-0 (16x16)	16	2048
		RGB 16bpp	SL-4 (32x8)	8	4096
		YUV 420 (8 bit or 10 bit)	SL-1 (16x16)	16	2048
		YUV 422	SL-2 (16x16)	16	2048
LPU0 Layer 1 LPU0 Layer 3 LPU1 Layer 1 LPU1 Layer 3	64KB	RGB 32bpp (or 24bpp)	SL-0 (16x16)	8	2048
		RGB 32bpp (or 24bpp)	SL-3 (32x8)	4	4096
		RGB 16bpp	SL-0 (16x16)	16	2048
		RGB 16bpp	SL-4 (32x8)	8	4096
		YUV 420 (8 bit or 10 bit)	SL-1 (16x16)	X	X
		YUV 422	SL-2 (16x16)	X	X

5.1.9 Quality of Service

The display processor handles *Quality of Service* (QoS) using the **ARQOS**[3:0], **AWQOS**[3:0] signals.

The **ARQOS**[3:0] and **AWQOS**[3:0] signals are part of the AXI read and write address channels. These signals are driven in-line with AXI transactions.

For layer processing units, software can program the value that is carried on **ARQOS[3:0]**. Software sets the value:

- In the LPU_RAXI_CONTROL register, for the layer processing unit.
- In the DS_AXI_CONTROL register, for the AFBC DMA unit.

For layer processing units, software can program the value that is carried on **AWQOS[3:0]**. Software sets the value:

- In the LPU_WAXI_CONTROL register, for the layer processing unit.
- In the AES_AXI_CONTROL register, for the AFBC DMA unit.

5.1.10 Color Space Conversions and Gamma Transformation

The display processor implements a range of color space conversions. The display can perform forward and inverse gamma transforms to convert between linear and non-linear RGB samples.

The display processor supports pre-composition processing of layers. Pre-composition processing can perform:

- YUV to RGB color space interpolation and conversion in the rich read layer pipelines of the layer processing unit with:
  - 4:2:2, 4:2:0 to 4:4:4 chroma subsampling schemes.
  - Implementation of several chroma interpolation methods.
  - Programmable coefficients for the conversion matrix.
- RGB to YUV conversion in the write-back layer pipeline of the layer processing unit with:
  - 4:4:4 to 4:2:0 chroma decimation.
  - Programmable coefficients for the conversion matrix.
- Inverse gamma transformation in all read layer pipelines of the layer processing unit. The processor can linearize gamma-corrected RGB samples to linear RGB or to sRGB space for processing in the rest of the processing pipeline. For inverse gamma transformation:
  - There are two 65 entry lookup tables that are shared between all read layers.
  - The lookup tables have pre-defined coefficient values that correspond to the sRGB to linear transform and the ST.2084 to sRGB transform.
  - Software can reprogram the coefficients when none of the layers use a look-up table.
- Color space conversion and adjustment, with programmable coefficients for the conversion matrix. The display processor only applies this function for layers 1 and 3.
- Forward gamma transformation.
  The display processor only applies this function for layers 1 and 3.

The processor can apply gamma correction to convert linear RGB samples to non-linear samples for the common composition gamma RGB space. For forward gamma transformation:

  - There is a 65 entry lookup table that is shared between all supported layers.
  - The table has pre-defined coefficient values that correspond to the linear to sRGB.
  - Software can reprogram the coefficients when none of the layers use a look-up table.

- All read layer pre-processing functions operate on 12-bit input samples and generate 12-bit samples on the output.
- The display processor can support a mixture of content with:

  - ITU-R BT.709 format.
  - BT.1886 (sRGB) format.
  - ST.2084 (HDR) format.
  - HLG format.
  - The display processor can support other standards through appropriate use of coefficients in the lookup tables.

The display processor supports post-processing of the composition output before the output goes to the display panel. The postprocessing can provide:

- Forward gamma transformation. The processor can apply gamma correction to convert linear RGB samples to non-linear samples for display output color space. For forward gamma correction
  - There is a 65 entry lookup table that is dedicated for each individual display output interface.
  - The lookup tables have pre-defined coefficient values that correspond to the linear to the Rec.709 transform.
  - Software can reprogram the coefficients when a display output interface is in idle mode or when gamma transformation is disabled.
- Output color adjustment, for white balance or white point correction, with programmable RGB conversion coefficients.
- RGB to YUV conversion in the display output unit with:
  - 4:4:4 to 4:2:2 chroma decimation.
  - 4:4:4 to 4:2:0 chroma decimation.
  - Programmable coefficients for the conversion matrix.
- Dithering of RGB components using the ordered noise method. The display processor supports dithering from 12 bits down to 8 bits or 10 bits.

5.1.11 Supported Data Flows

The display processor enables different data flows from the layer processing units through the composition units.

The display processor provides flexible use of the functional blocks and the layer data can flow through the different blocks depending on the required flow. The data flow can support software layer split, layer merge, and side-by-side processing. The Z-order of layers is programmable. The main processing data flows are:

5.1.11.1 Single Display Processing

In single display processing, only one of the display output units is active. Possible dataflows for single display processing:

- Layer data moves through layer processing unit 0 and composition unit 0. Display output unit 0 is in single link mode. Output data appears on link 0. Other units are not active.
- Layer data moves through layer processing units 0 and 1 to composition units 0 and 1. Composition unit 1 passes composite data to composition unit 0. The resultant compositedata from composition unit 0 moves to display output unit 0. Display output unit 0 is in single link mode. Output data appears on link 0.
- Layer data moves through layer processing units 0 and 1 to composition units 0 and 1. Composition unit 1 passes composite data to composition unit 0. The resultant compositedata from composition unit 0 moves to display output unit 0. Display output unit 0 is in single link mode. Output data appears on link 0. Composition unit 0 writes composite databack to memory.
- Layer data moves through layer processing units 0 and 1 to composition units 0 and 1. Composition unit 1 passes composite data to composition unit 0. The resultant compositedata from composition unit

0 moves to display output unit 0. Display output unit 0 is in dual-link mode. Output data appears on link 0. Composition unit 0 writes compositedata back to memory.

This dataflow can process up to eight alpha blended layers. The diagram shows an example of single display processing.

5.1.11.2 Side-By-Side Display Processing

In side-by-side display processing, both of the display output units are active and the output appears only on one output unit. Possible dataflows for side-by-side processing:

- Layer data moves through layer processing units 0 and 1 and composition units 0 and 1.. Display output unit 1 is active and passes composition data to display output unit 0. Output data from display output unit 0 appears on link 0. Composition unit 0 writes composite data back to memory for the left side of composition. Composition unit 1 writes composite data back to memory for the right side of composition.
- Layer data moves through layer processing units 0 and 1 and composition units 0 and 1.. Display output unit 0 is active and passing data to display output unit 1. Output data from display output unit 1 appears on link 0. Composition unit 1 writes composite data back to memory for the left side of composition. Composition unit 0 writes composite data back to memory for the right side of composition.

5.1.11.3 Image Merge Display Processing

In image merge display processing, both of the display output units are active and the output appears only on one output unit. Possible dataflows for image merge processing:

- Layer data moves through layer processing units 0 and 1 and composition units 0 and 1. Display output unit 0 is in single link mode. Display output unit 1 is active and passes composition data to display output unit 0. Output data from display output unit 0 is the left part of one frame, and output data from display output unit 1 is the right part of the frame. They are merged together into one frame, and output only on link 0 of display output unit 0. Composition unit 0 writes composite data back to memory for the left side of composition. Composition unit 1 writes composite data back to memory for the right side of composition.
- Layer data moves through layer processing units 0 and 1 and composition units 0 and 1. Display output unit 1 is in single link mode. Display output unit 0 is active and passing data to display output unit 1. Output data from display output unit 1 is the left part of one frame, and output data from display output unit 0 is the right part of the frame. They are merged together into one frame, and output only on link 0 of display output unit 1. Composition unit 1 writes composite data back to memory for the left side of composition. Composition unit 0 writes composite data back to memory for the right side of composition.

5.1.11.4 Composition

- Layer data is received and the compositor blends the layers.
- The composite data goes to the composition unit output.
- The composition unit does not write the data back to memory.

5.1.11.5 Composition with Write-Back

- Layer data is received and the compositor blends the layers.
- The composite data goes to the composition unit output.

- The composition unit sends output data to the layer processing unit to write back to system memory.

5.1.11.6 Scaling and Composition

- Layer data undergoes scaling and the compositor blends the scaled layers.
- The composite data goes to the composition unit output.
- The composition unit does not write the data back to memory.

5.1.11.7 Scaling and Composition with Write-Back

- Layer data undergoes scaling and the compositor blends the scaled layers.
- The composite data goes to the composition unit output.
- The composition unit sends output data to the layer processing unit to write back to system memory.
- When the composition output data horizontal size is wider than 2048 pixels, the display processor uses internal-split functionality to support the dataflow.

5.1.11.8 Composite Scaling with Write-Back

- Layer data bypasses the scaling subsystem and the compositor blends the layers.
- The composite data goes to the composition unit output.
- The compositor also feeds the data back to the scaling subsystem.
- The scaling subsystem scales the data and passes it to composition unit.
- The composition unit sends scaled output data to layer processing unit to write back to system memory.
- When the composition output data horizontal size is wider than 2048 pixels, the display processor uses internal-split functionality to support the dataflow.

5.1.11.9 Scaling and Composition with Composite Scaling and Write-Back

- All layer data except one bypasses the scaling subsystem and passes to the compositor.
- One layer data undergoes scaling and then passes to the compositor.
- The compositor blends the layers.
- The composite data goes to the composition unit output.
- The compositor feeds the composite data back to the unused scaling engine.
- The scaling engine scales the data and passes it to composition unit.
- The composition unit sends scaled output data to the layer processing unit to write back to system memory.

Table 5-3 Summary of Permitted Data Flow Through the Composition Unit for the Default Configurations

Input to composition unit	No scaling		Layer scaling		Composite scaling with write- back	
	No write-back	With write-back	No write-back	With write-back	No layer scaling	With layer scaling
Layer 0, 1, 2, 3 individually	Supported	Supported	Supported	Supported	Supported	Supported
Layer 0&2, 1&2, 0&3, 1&3	Supported	Supported	Supported	Supported	Supported	-

Input to composition unit	No scaling		Layer scaling		Composite scaling with write- back	
	No write-back	With write-back	No write-back	With write-back	No layer scaling	With layer scaling
with software layer split						

5.1.11.10 Data Flow Support in MP2_4096 Default Configuration Only

Layer 2 scale before composition. Layer 3 scale before composition.
Layer 0 and layer 2 scale before composition. Layer 1 and layer 2 scale before composition. Layer 0 and layer 3 scale before composition. Layer 1 and layer 3 scale before composition.
Layer 2 scale before composition and composition write-back to memory. Layer 3 scale before composition and composition write-back to memory.
Layer 0 and layer 2 scale before composition and composition write-back to memory. Layer 1 and layer 2 scale before composition and composition write-back to memory. Layer 0 and layer 3 scale before composition and composition write-back to memory. Layer 1 and layer 3 scale before composition and composition write-back to memory.
Composition scale write-back to memory only, using a single scaling engine. Downscaling only.
Composition scale for memory write-back only with internal split and both scaling engines. Downscaling only.
Layer 0 scale before composition and composition scale for memory write-back only. Layer 1 scale before composition and composition scale for memory write-back only. Layer 2 scale before composition and composition scale for memory write-back only. Layer 3 scale before composition and composition scale for memory write-back only.

5.1.12 Multi-layer composition

The display processor enables hierarchical display layer composition including a background color layer.
Each composition unit in the display processor supports up to five input layers, layers 0-4. The composition unit supports one, extra, background color layer. The background color layer is always the bottom of the layer hierarchy. The diagram shows the hierarchy and order of composition. Apart from the background color layer, the other layers can be in arbitrary input order. The Z-order is programmable.
The composition unit composes the layers with an alpha blend operation. The inputs to the layers can be:

- One of the layers from the layer processing unit that is connected to the given composition unit.
- The composition result from the other composition unit.
- A scaled layer from the scaling subsystem of the given composition unit.

The composition units can compose up to five layers to produce a single composited output.

5.1.13 Multi-Layer Composition in Tandem

The display processor enables hierarchical display layer composition including a background color layer.
The display processor supports the use of composition units in tandem. The diagram shows the processing hierarchy for an eight layer composition. The composition result from composition unit n

forms an input to composition unit m (DOm ACTIVE mode), where n can be 0 or 1 and m can be 1 or 0.
Input from one composition unit can pass to the input of another composition unit with any Z-order. You can program any of the CU_INPUT_ID registers with the OUTPUT_ID of the other composition unit. Particular input layers are enabled in the composition unit with the EN (enable) bit in the respective CU_INPUT{0-4}_CONTROL register. The software driver must program the correct, valid input identifier for the CU_INPUT_ID{0-4} register.

5.1.14 Side-By-Side Mode

The display processor can process left and right composition halves in parallel for a merge mode display.
The display processor supports the parallel combination of layer processing units, composition units, and display output units. These streams pass data in parallel from system memory to the display output. Side-by-side processing is only available:

- When the display processor controls a single display.

In side-by-side mode, the display processor operates on one half of the composition scene through layer processing unit 0, composition unit 0, and display output unit 0. The processor operates on the other half of the scene through layer processing unit 1, composition unit 1, and display output unit 1. The display processor uses one display output unit as a master output and the other display output unit as a slave. The master output unit has the active display output link. The slave output unit passes its data to the master output unit. The master output unit sends out its data on link 0 of the output interface.
Side-by-side mode can provide up to four layer composition. The software driver must configure identical image-processing parameters for the layer processing units, the composition units, and the display output units. The software driver must program the necessary offsets to read left and right halves of the composition layers.
In side-by-side mode, software can configure each half of the composition to provide half of the maximum supported resolution. Therefore, scaling with software layer split and internal composition split options are not required, and are therefore not supported.
When the processor scales a layer on the left and the right sides of the pipelines in parallel, the resultant image might have visual artifacts on the boundary between the left and right halves.

5.1.15 Image Merge Mode

The display processor can process left and right composition halves in parallel for a single link display.
The display processor supports the parallel combination of layer processing units, composition units, and display output units to provide single link streams. These streams pass data in parallel from system memory to the display output. Image merge processing is only available:

- When the display processor controls a single display in single link mode.

In image merge mode, the display processor operates on one half of the composition scene through layer processing unit 0, composition unit 0, and display output unit 0. The processor operates on the other half of the scene through layer processing unit 1, composition unit 1, and display output unit 1. The display processor uses one display output unit as a master output and the other display output unit as a slave. The master output unit has the active display output link. The slave output unit

passes its data to the master output unit. The master output unit merges its data and the slave data from the slave output unit into one frame, and sends out merged data on link 0 of the master output interface.
Image merge mode can provide up to four layer composition. The software driver must configure identical image-processing parameters for the layer processing units, the composition units, and the display output units. The software driver must program the necessary offsets to read left and right halves of the composition layers.
In image merge mode, software can configure each half of the composition to provide half of the maximum supported resolution. Therefore, scaling with software layer split and internal composition split options are not required, and are therefore not supported.
When the processor scales a layer on the left and the right sides of the pipelines in parallel, the resultant image might have visual artifacts on the boundary between the left and right halves. Therefore, the left and the right sides have an overlapped area to solve this problem. In addition, the overlapped area can be cropped in the master display output unit.

5.1.16 Scaling and Image Enhancement

In the display processor, each composition unit can perform scaling. The number of scaling engines depends on the display processor configuration.
The scaling subsystem has these features:

- For the input of layers from the layer processing unit to the composition units:
- Scale the composition result for memory write-back.
- Scale input and output size from 16 by 16, to 2048 by 4096 pixels for each scaling engine in the scaling subsystem.
- Support for scaling of the composition result split across two scaling engines in parallel.
- Support for parallel scaling of two input layers for the software layer split capability.
- Scale in an arbitrary ratio for horizontal and vertical dimensions. This ratio is programmable for input and output resolutions with single pixel accuracy. The maximum downscaling ratio the processor supports is six times in the horizontal direction and six times in the vertical direction. The maximum upscaling ratio is 64 times in the horizontal direction and 64 times in the vertical direction.
- Scaling method that is software programmable between the nearest neighbor method or the polyphase filtering method.
- 12-bit internal processing precision.
- Polyphase filtering:
    - Six taps and 16 phases.
    - Five pre-defined, hardcoded filter coefficients tables.
    - Two register-programmable look-up-tables for user-defined filter coefficients, separately for each dimension, horizontal and vertical.
    - Programmable initial phases, separately for each dimension, horizontal and vertical.
- Embedded image enhancement engine:
    - High-pass filtering in both horizontal and vertical directions, using a 3 by 3 pixel window.
    - Edge detection that is based on up to eight neighboring pixels.
    - Programmable edge detection levels.

  - Programmable filter coefficients.

- Scaling of 8-bit alpha component.
- Separate programmable scaling methods for alpha and for RGB components.

5.1.17 Software Layer Split

The display processor uses software layer split to process layers that are larger than it can accommodate in a single layer. Software layer split caters for use cases where a single layer pipeline cannot support the content, or a single scaling engine cannot scale the input content.
The display processor can read a single frame buffer in external memory with a software layer split technique. To make this access, the processor uses the resources in the layer processing and composition units across two layers. Software layer split access is necessary because the display processor cannot read certain resolutions in a single layer.

- Video layers in YUV 4:2:0 or YUV 4:2:2 formats of horizontal resolution greater than 2048 pixels.
- Graphics layers of AFBC SL0 and SL1 of horizontal resolution greater than 2048 pixels. The maximum horizontal values that are permitted in rich layers are:

RGB format and non-rotated scan = 4096 pixels,
YUV 4:2:2/4:2:0 format and non-rotated scan = 2048 pixels.
AFBC RGB format and SL3 or SL4 = 4096 pixels,
AFBC RGB or YUV 4:2:2/4:2:0 format and SL0 = 2048 pixels. Before or after scaling per single scaling engine = 2048 pixels.
The maximum horizontal values that are permitted in simple layers are:
RGB format and non-rotated scan = 4096 pixels,.
AFBC RGB format and SL3 or SL4 = 4096 pixels.
AFBC RGB and SL0 = 2048 pixels.
Before or after scaling per single scaling engine = 2048 pixels.
When you use software layer split for scaling, the maximum horizontal size of content before the split is 4032 pixels.
The software driver might have to compose scenes that contain rich or simple layers with sizes that exceed the maximum values per layer pipeline. In these circumstances, the driver must enable two layers to access the large frame buffer.
To handle the scene with full resolution in the composition unit, the software must correctly handle:

- Horizontal size.
- Vertical size.
- Horizontal offset.
- Vertical offset.
- Memory stride.
- Memory pointer.
- Alpha-blending.
- Layer processing.
- AFBC cropping.

The scaling subsystem in each composition unit can implement up to two scaling engines (depending on the configuration). This arrangement makes it possible to read one large frame buffer

using two rich or simple layers. The processor can pass these two layers to the two scaling engines, scale them, merge them, and then compose them in the composition subsystem.
This approach is recommended, for example, when downscaling of 4K video layer in YUV 4:2:0 or 4:2:2 format to 1920 by 1080 displays.
Software layer split has constraints on which layers can access the large frame buffer. The table shows the arrangements of layers for software splits.

Table 5-4 Allocation of Read Layers during Software Layer Split

Layer Pair	Pixel Formats	Left Layer	Right Layer	Scaling Engine 0	Scaling Engine 1
Layer 0 and Layer 2	RGB, YUV	Layer 0	Layer 2	Layer 0 (left)	Layer 2 (right)
Layer 1 and Layer 3	RGB	Layer 1	Layer 3	Layer 1 (left)	Layer 3 (right)
Layer 0 and Layer 3	RGB	Layer 0	Layer 3	Layer 0 (left)	Layer 3 (right)
Layer 1 and Layer 2	RGB	Layer 1	Layer 2	Layer 1 (left)	Layer 2 (right)
Layer 0 and Layer 1	RGB	Layer 0	Layer 1	Not permitted during software layer split	
Layer 2 and Layer 3	RGB	Layer 2	Layer 3	Not permitted during software layer split	

It is possible that pixel artifacts occur at the boundary of the scaled layers. To eliminate these artifacts in the composition result, there must be some pixel overlap on the left and right side. When reading the left and right half of the 4K layer, the two layers in the layer processing unit must fetch extra overlapping pixels from each side. The composition unit eliminates the pixel overlap, after the scaling operation, through correct programming of pixel cropping values.
The figure shows an example of access to a 3840 by 2160 video frame buffer in YUV 4:2:0 format. The processor uses two rich layers from one layer processing unit. In this example, the processor composes the 4K video layer with two other simple layers in ARGB_8888 pixel format.

5.1.18 Display Formats

The display processor provides various display formats that support industry standards. The display processor provides:

- An arbitrary active region resolution.
  - From 64 by 64 pixels up to 4096 by 4096 pixels.
- Parallel display output interface with **HSYNC** and **VSYNC** signals.
- Frame rates for standards such as VESA, CEA, and HDMI.
- Programmable display timing parameters:
  - HFRONTPORCH.
  - VFRONTPORCH.

  - VBACKPORCH.
  - HBACKPORCH.
  - HSYNCWIDTH.
  - VSYNCWIDTH.
  - HACTIVE.
  - VACTIVE.
- Programmable polarity of HSYNC and VSYNC.
- Up to 10-bit displays.
- Side-by-side operation.
- Single link image merge operation.
- RGB, YUV 4:4:4, 4:2:2 output.
- YUV 4:2:0 output.
- **DISPREADY** input signal. Supports asynchronous MIPI DSI transmitters. The **DISPREADY** signal supports different backpressure modes:
  - Regular, suspend mode.
  - Halt mode.
- **FRAMESTART** output signal. Provides information about the beginning of a new frame that is sent through the display output interface.
- Data enable signal. Blanking.
- General purpose, 32-bit **DISPUSER** signal.
- Command mode.
- Support for offline composition.
- Support for Cyclic Redundancy Checks (CRC) on the display output.
- **TRANSFACTIVE** output signal. Provides information about the active frame transfer.

The table lists limitations of the display timing parameters:

Display Timing parameter	Permitted Values in Video Mode	Permitted Values in Command Mode	Description
HACTIVE	64-4096	64-4096	Horizontal size of active region.
HFRONTPORCH	0–4095	8	Horizontal front porch, number of pixels. When dual link is enabled in video mode, then the minimum HFRONTPORCH value is 8 pixels.
HSYNCWIDTH	1-1023	4	Horizontal synchronization pulse width, number of pixels. When dual link is enabled in video mode, then the minimum value of HSYNCWIDTH + HBACKPORCH is 8 pixels. When side-by-side processing or image merge processing is enabled in video mode, then the minimum value of HSYNCWIDTH + HBACKPORCH is 12 pixels. When side-by-side processing or image merge processing is enabled in video mode and the cropping area horizontal size is odd, then the

Display Timing parameter	Permitted Values in Video Mode	Permitted Values in Command Mode	Description
			minimum value of HSYNCWIDTH + HBACKPORCH is 13 pixels.
HSP	0 or 1	0 or 1	Horizontal synchronization pulse polarization: 0 = negative, 1 = positive
HBACKPORCH	0-1023	4	Horizontal back porch, number of pixels. When dual link is enabled in video mode, then the minimum value of HSYNCWIDTH + HBACKPORCH is 8 pixels. When side-by-side processing or image merge processing is enabled in video mode, then the minimum value of HSYNCWIDTH + HBACKPORCH is 12 pixels. When side-by-side processing or image merge processing is enabled in video mode and the cropping area horizontal size is odd, then the minimum value of HSYNCWIDTH + HBACKPORCH is 13 pixels.
VACTIVE	64-4096	64-4096	Vertical size of active region, number of lines.
VFRONTPORCH	0-16383	4	Vertical front porch, number of lines.
VSYNCWIDTH	1-255	1	Vertical synchronization pulse width, number of lines. In Video Mode, the minimum total value of VFRONTPORCH + VSYNCWIDTH + VBACKPORCH must be at least 3 lines.
VSP	0 or 1	0 or 1	Vertical synchronization pulse polarization: 0 = negative, 1 = positive. In Video Mode, the minimum total value of VFRONTPORCH + VSYNCWIDTH + VBACKPORCH must be at least 3 lines.
VBACKPORCH	0-255	1	Vertical back porch, number of lines. In Video Mode, the minimum total value of VFRONTPORCH + VSYNCWIDTH + VBACKPORCH must be at least 3 lines.
PREFETCH_LINE	1-16893	1-5	Prefetch line number. The maximum permitted value is: VFRONTPORCH + VSYNCWIDTH + VBACKPORCH
PROG_LINE0	0-20989	0-4101	Programmable line 0 number. The maximum permitted value is equal to the maximum vertical total time.

Display Timing parameter	Permitted Values in Video Mode	Permitted Values in Command Mode	Description
PROG_LINE1	0-20989	0-4101	Programmable line 1 number. The maximum permitted value is the maximum vertical total time.
DRIFT_TIMEOUT	8-4095	8-4095	Dual link drift timeout (number of PXL_CLK clock cycles). If dual link mode is enabled, then drift timeout must be >= HFRONTPORCH.
FRAME_TIMEOUT	N/A	0-16777215	Frame timeout (number of PXL_CLK clock cycles).
T0_INTERVAL	N/A	0-16777215	TE window interval 0 (number of PXL_CLK clock cycles).
T1_INTERVAL	N/A	0	TE window interval 1 (number of PXL_CLK clock cycles).
T2_INTERVAL	N/A	0	TE window interval 2 (number of PXL_CLK clock cycles).
TE_TIMEOUT	N/A	0	TE timeout (number of PXL_CLK clock cycles).

5.1.19 Single Display Link (Non-Split Mode)

Each output unit of the display processor has the potential for two output links. The default arrangement is to use one link.

Each display output unit includes link 0 with a dedicated set of signals:

HSYNC

VSYNC

DATAEN

PXLDATAx

The display processor transfers all pixel data to the output only through link 0.

In single link operation, all display timing parameters HFP, HSYNC, HBP, VFP, VSYNC, VBP, and active region sizes HACTIVE, VACTIVE, refer to the full frame. The figures show display output interface timings for a single link at positive polarization of the **HSYNC** and negative polarization of the **VSYNC** signals.

The **DISPREADYL10** input to the display processor must be tied to 0.

5.1.20 Single Link Image Merge Processing

Single link image merge processing enables the display processor to support high-resolution output through a single. Single link image merge processing is enabled with the IM (image merge enable) bit and SBS (side by side processing enable) in the IPS_CONTROL register. When enabled, the display processor transfers the output pixel data through one link, but outputs two pixels per clock cycle. In single link image merge processing operation, display timing parameters HFP, HSYNC,

HBP, VFP, VSYNC, VBP, and active region size VACTIVE, refer to the full frame. HACTIVE is half of the full frame when the display processor is configured to output two pixels per clock cycle.

5.1.21 Dithering

The display processor can apply dither to the output data. The display processor uses the ordered dithering algorithm.
The software driver enables dithering with the DITH (dithering enable) bitfield of the IPS_CONTROL register.
The input pixel format for dithering can be RGB or YUV 4:4:4, with an input color depth of 12 bits per component.
The software driver must define the output bit precision (after dithering). Software sets the value in the OUT_DEPTH bitfield of the IPS_DEPTH control register. The valid values for output precision are 8 and 10. The same output depth precision is applied for each pixel component.
The width of R, G, and B output channels from the dithering block is always 10 bits. The 8-bit dithering result is always aligned to the 8 MSBs of the 10-bit channel. When dithering is disabled:

- The 12-bit input RGB or YUV samples propagate to the dither block output dropping the 2 LSBs.
- The display processor does not apply rounding.

When the input pixel format is YUV 4:4:4, the processor only applies ordered dithering to the Y component. The processor rounds up the LSB, to reduce the precision of the chrominance components, and then discards 4 bits or 2 bits. For example, when the processor performs dithering from 12 bits to 8 bits, the processor checks the value on the four LSBs of the chrominance sample. If the value is equal to or greater than 0x8, then the processor increments eight MSBs by one. The new value must stay in range. The processor applies the same dithering matrix for all three RGB pixel components.
The table shows four ways to reduce the precision of the RGB or Y output samples from the dithering block. These reductions depend on the OUT_DEPTH bitfield in IPS_DEPTH register and DITH bit in IPS_CONTROL register. Processing in subsequent blocks in the processor maintains the bit precision of RGB or YUV. The bit precision persists for chrominance decimation and clamping. The display output interfaces (DISP0 and DISP1) always align the 8-bit samples to MSB.

Table 5-5 Bit Precision Reduction on the Output of Dithering Block

IPS_CONTROL.DITH	IPS_DEPTH.OUT_DEPTH	Description
0	10	The processor truncates 2 LSBs and reduces the precision of RGB, or YUV output samples from 12 bits to 10 bits.
0	8	The processor truncates 4 LSBs and reduces the precision of RGB, YUV output samples from 12 bits to 8 bits.
1	10	The processor uses the ordered dithering algorithm to reduce the precision of RGB or Y output samples from 12 bits to 10 bits. The

IPS_CONTROL.DITH	IPS_DEPTH.OUT_DEPTH	Description
		processor rounds up the values of and samples to 10 bits and then truncates 2 LSBs.
1	8	The processor uses the ordered dithering algorithm to reduce the precision of RGB output samples from 12 bits to 8 bits. The processor rounds up the values of and samples to 8 bits and then truncates 2 LSBs.

5.1.22 CRC Signature Generation

The display processor supports CRC signature generation.
The display processor computes per frame *Cyclic Redundancy Check* (CRC) data. The processor bases the CRC on the final pixel data that is sent out on the display output interface. Each display output unit integrates two CRC blocks, and each block derives the 64-bit CRC for the frame. Software can use the CRC, for example, to detect that content of a frame has not changed since the last refresh.
In the single link output, the display processor uses only one CRC block.
Software enables CRC calculation with the CRC bit (CRC enable) in the BS_CONTROL register. The registers are updated with the calculated CRC every frame. Software can read the CRC signatures through the BS_CRCn_LOW and BS_CRCn_HIGH control registers (n is 0 or 1). The _LOW register holds the lower 32 bits of the CRC, and the _HIGH register holds the higher 32 bits of the CRC.
The processor writes to BS_CRC0_LOW and BS_CRC0_HIGH. The CRC algorithm simulates a linear-feedback shift register with a 64-bit polynomial, 0xC96C5795D7870F42. The display processor implements this algorithm with a CRC coefficient table of 64 by 64-bit size. The CRC uses the 8-bit or 10-bit parallel pixel samples that are transmitted to the output to form the 64-bit words for the CRC block. The bit precision on the output is contained in the OUT_DEPTH bitfield of the IPS_DEPTH control register. Each bit of the CRC is computed as a parallel XOR of the input data with a corresponding entry in the coefficient table. The CRC computed for the current 64-bit input word is used to derive a 64-bit mask for the next 64-bit input data word. This mask is then used during the XOR of the input word with the CRC coefficient table entry. The algorithm is reset at the beginning of each frame.
When the display output unit applies dithering to the output, the CRC uses 8-bit precision. CRC calculation is not supported for offline composition mode.

5.1.23 Display Security

The display processor provides different levels of security. The processor is compatible with the Arm Trusted Base System Architecture specification, and TrustZone Media Protection Architecture versions 1 and 2 requirements.
The display processor provides these security levels:

Protected mode

Protected mode is for the display of content that is protected for DRM. The display processor accesses the content in protected system memory.

Trusted layer operation

Trusted layer operation is for composition with a trusted user interface layer. The trusted layer is stored in the trusted frame buffer, in the secure memory. This layer might display, for example a secure banking application layer or secure watermark logo.

Normal display mode (non-protected and non-trusted)

Normal display mode is for normal processing that does not require secure operation.

Protected display mode is transparent to trusted layer operation in the display processor because these two security modes use different hardware and software mechanisms. The display processor can operate in different security modes in the different layer processing units, when using multiple display outputs.

Chapter 6 SMMU

6.1 Overview

The major functional blocks of the SMMU are the Translation Buffer Unit (TBU), Translation Control Unit (TCU), and Distributed Translation Interface (DTI) interconnect.
The following figure shows the block diagram of the SMMU.

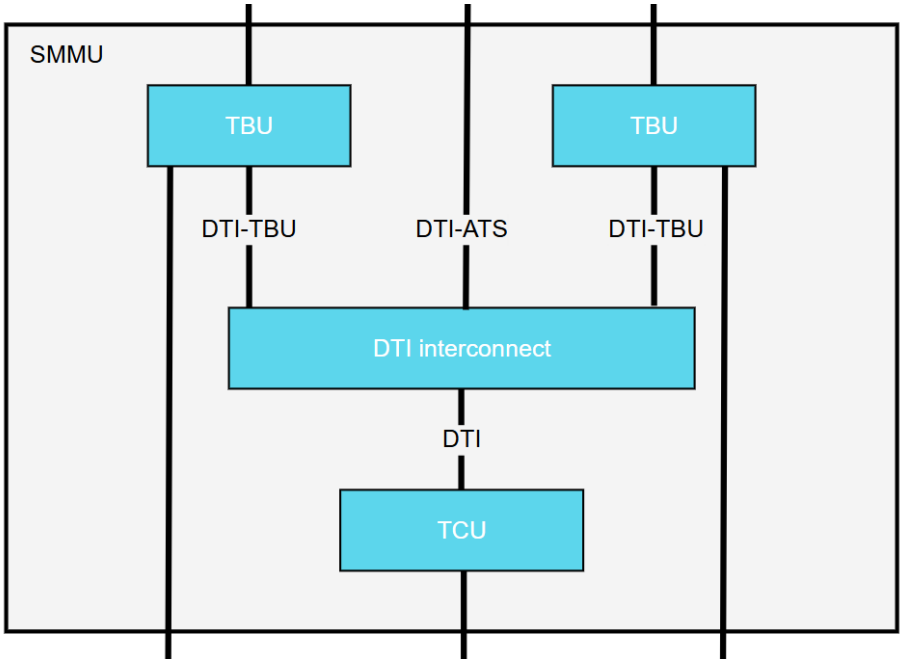


Figure 6-1 SMMU Block Diagram

The SMMU contains the following key components:

Translation Buffer Unit (TBU)

The TBU contains Translation Lookaside Buffers (TLBs) that cache translation tables. The SMMU implements a TBU that can be connected to single master or multiple masters. It is also possible to connect multiple TBUs to a single master to improve performance. These TBUs are local to the corresponding master and are implemented as ACE-Lite TBU.

Translation Control Unit (TCU)

The TCU controls and manages the address translations. The SMMU implements a single TCU. In SMMU-based systems, the AMBA® DTI protocol defines the standard for communicating with the TCU. See the AMBA® DTI Protocol Specification.

DTI interconnect

The DTI interconnect connects multiple TBUs to the TCU.
See the Arm® CoreLink™ MMU-700 System Memory Management Unit Technical Reference Manual for more information: https://developer.arm.com/documentation/101542/latest/.

6.1.1 SMMU PCIE Subsystem

SMMU in the PCIe subsystem.

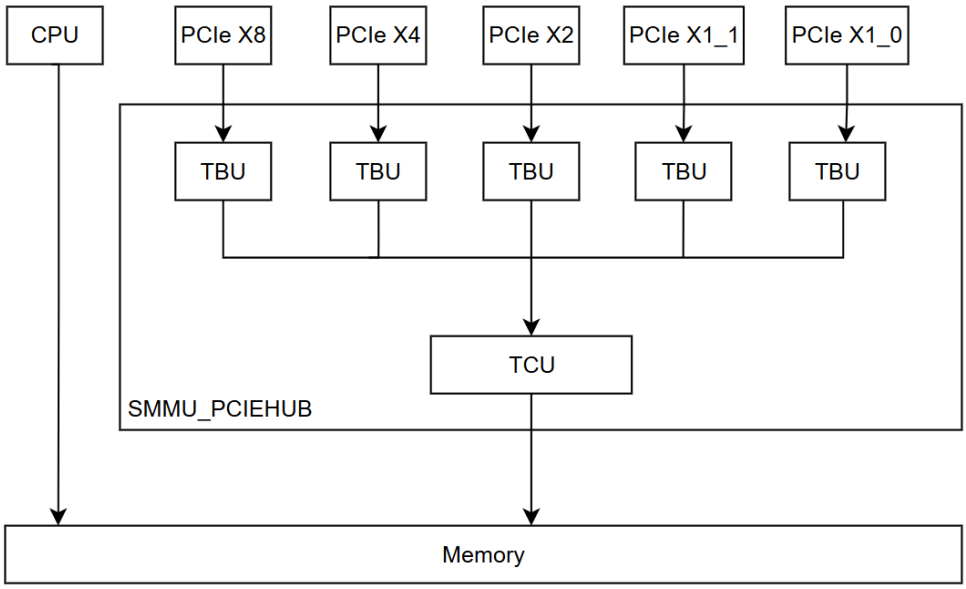


Figure 6-2 SMMU PCIE Subsystem

6.1.2 SMMU MM Subsystem

SMMU in the multi-Media subsystem.

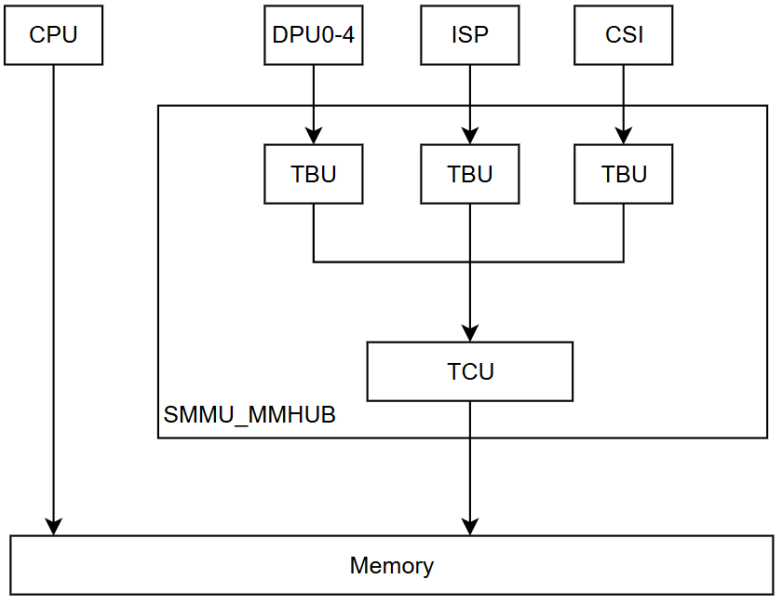


Figure 6-3 SMMU MM Subsystem

6.1.3 SMMU Syshub Subsystem

SMMU in the system control sub-system.

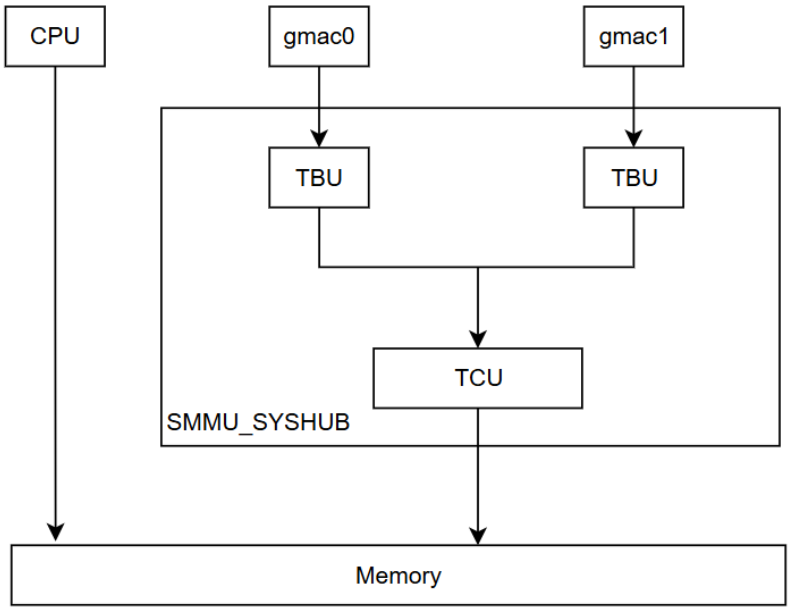


Figure 6-4 SMMU Syshub Subsystem

Chapter 7 PCIe

Note:
The PCIe subsystem does not support Endpoint (EP) mode. EP-related registers and settings are neither configurable nor applicable. The subsystem operates only in Root Port (RP) mode.

7.1 Overview

The PCIe subsystem consists of several IP blocks, as illustrated in the Figure 7-1. It includes:

- 5 PCIe cores, with a total of 16 lanes.
- 2 PCIe X4 PHY and 1 PCIe X8 PHY for physical layer connectivity.
- One NI-700, configured as a PCIe Hub, managing upstream and downstream traffic.
- One TBU/TCU module, handling address translation.
- One ITS (Interrupt Translation Service), converting message-based interrupts for the GIC (Generic Interrupt Controller).

7.2 Block Diagram

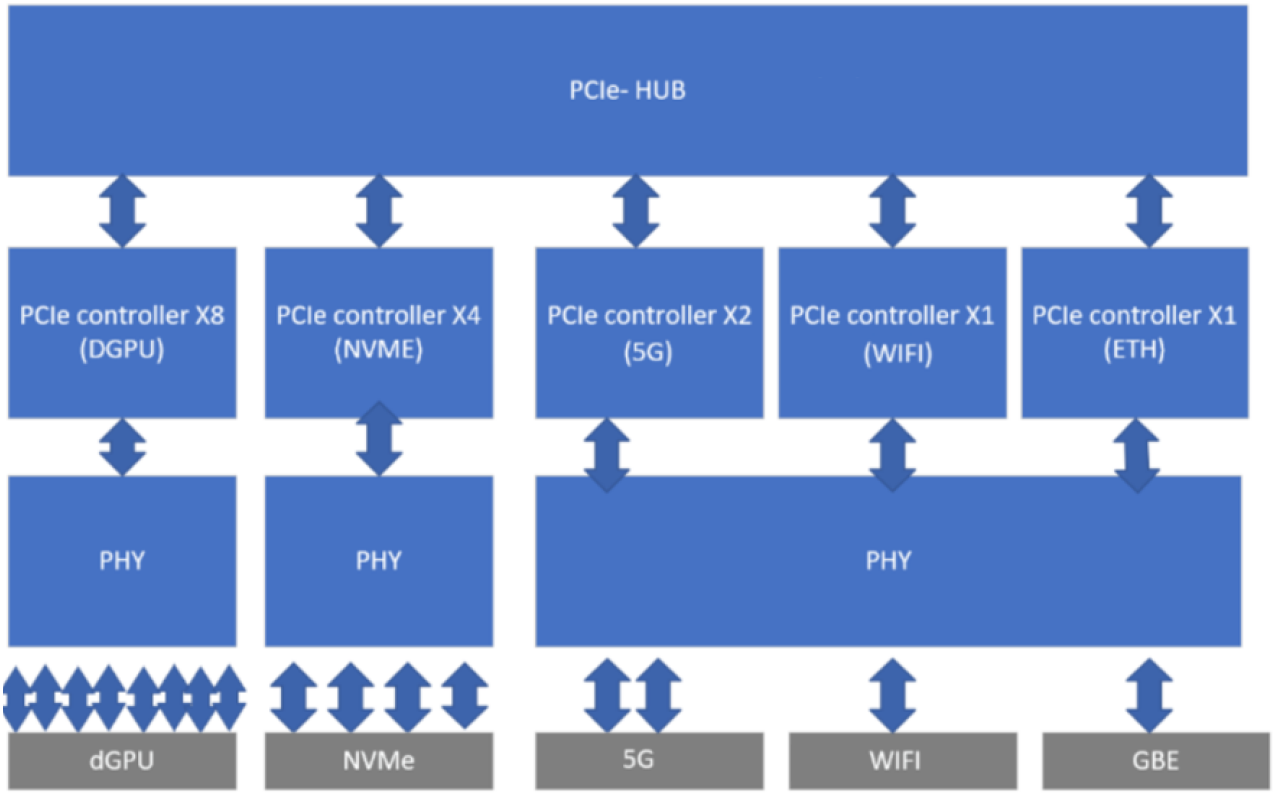


Figure 7-1 PCIe Block Diagram

7.3 PCIe Subsystem Features

Table 7-1 PCIe Subsystem Features

PCIe Link	X1_0	X1_1	X2	X4	X8
Pin Name	PCIE3	PCIE4	PCIE2	PCIE1	PCIE0
RC Mode	Y	Y	Y	Y	Y
EP Mode	N	N	N	N	N
Traffic Type	WIFI	Ethernet	5G	NVMe	GPU
Power Gating	N	N	N	N	Y
Highest Generation	Gen4	Gen4	Gen4	Gen4	Gen4
Max Lane Number	1	1	2	4	8

7.4 PCIe Gen4 Controller

7.4.1 Architectural Block Diagram

This section describes the operation of internal functional blocks and the external interfaces of the PCIe core. The Controller IP consists of the following key layers:

- CPCS (Controller PCS)
- PLL (PHY Logical Layer)
- DLL (Data Link Layer)
- TL (Transaction Layer)
- AL (Application Layer)

The PCIe Controller IP HPA (High Performance Architecture) architecture is optimized to handle multiple Transaction Layer packets and multiple Data Link Layer packets in parallel on both the receive and the transmit paths. The key feature of this architecture is to optimize latency in the design by reducing the amount of data realignment that is done between different internal blocks of the IP.
The PCIe Controller IP has two key Interface Boundaries:

- The PIPE interface which connects to the PMA/PCS
- The AXI4 1/F which connects to the application logic.

The PCIe Controller IP uses industry standard specification for these interfaces.

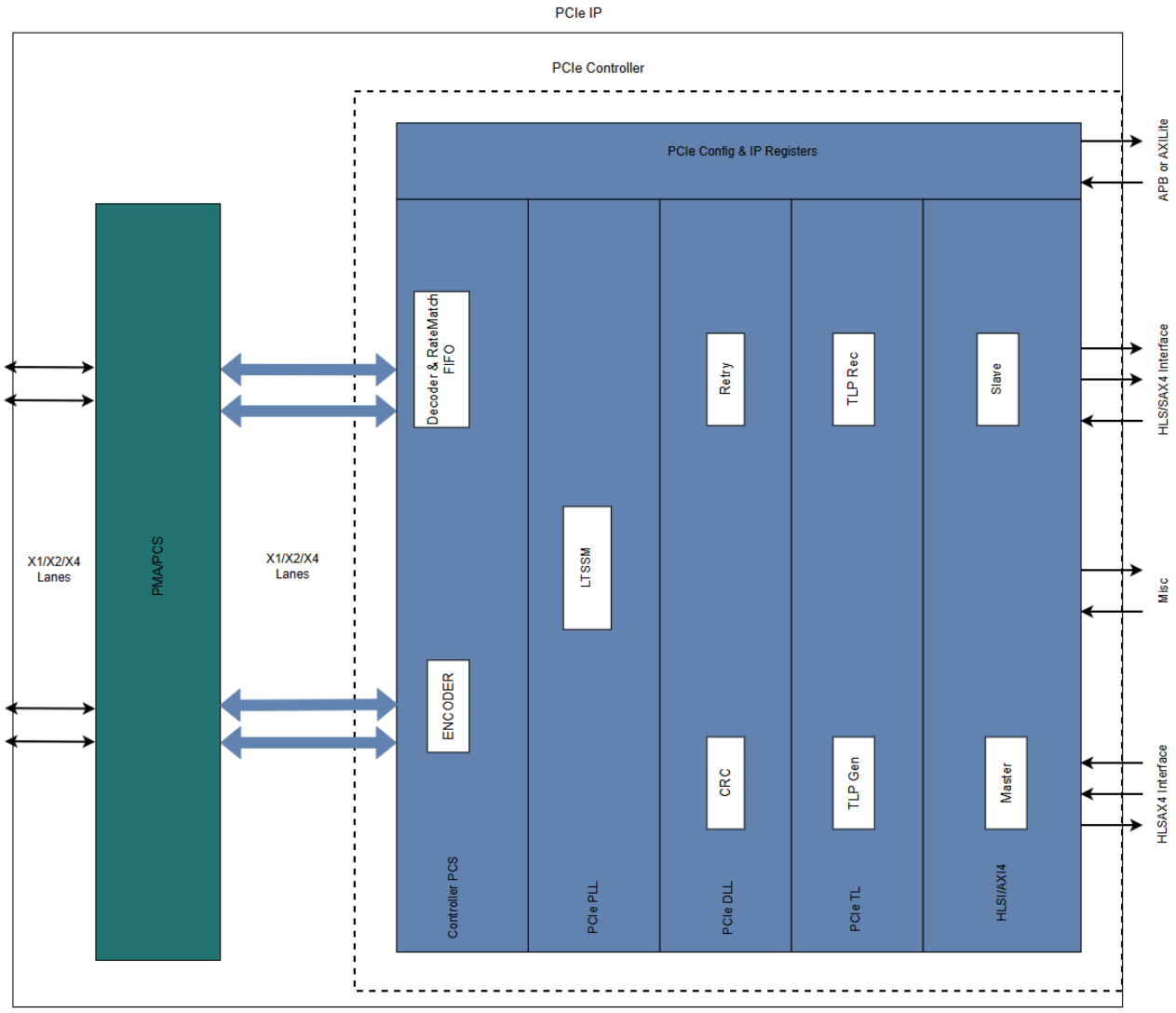


Figure 7-2 PCIe controller IP architecture overview

7.4.2 Controller PCS

The controller PCS IP is implemented to support PCIe PIPE Specification 5.2 (SerDes Architecture with Low Pin Count). The logic is implemented per lane and also supports multi-link feature. This module interacts with a PIPE 5.2 compliant PHY on one side and the data transfer between Controller IP internally using custom PIPE 4.4.1Interface on the other side. The CPCS interfaces to the PHY at 20bit for Gen1/2 and 40bit interface for Gen3/4. On the PIPE 4.4.1the interface width is fixed at 32bits. This module supports nominal empty RM (Rate Match) FIFO for reducing latency. It also supports test mode for internal loopback option from receiver to transmitter.

Figure 7-3 shows the block diagram of Controller PCS.

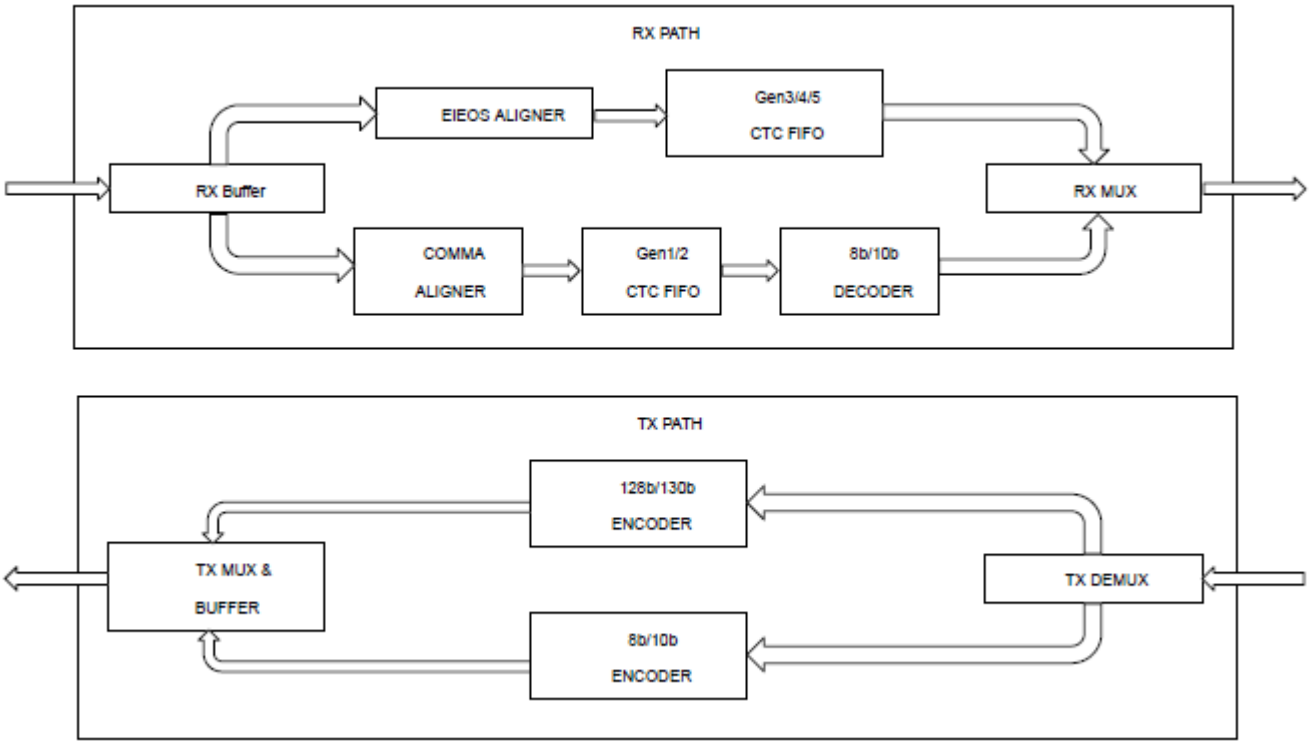


Figure 7-3 Controller PCS block diagram

This block module supports:

- **Fixed Core Clock mode:** In this operation mode, the Controller PCS requires a fixed Core Clock input that is provided to the core blocks irrespective of the current speed of operation on the PIPE interface. This mode adds various clock transfer FIFOs hence, adding latency in the design.

7.4.3 PHY Logical Layer

The receive operation of the Physical layer is as follows:

1. Data arrives on the PIPE Interface from the CPCS block through one or more lanes, over a 32-bit interface per lane.
2. The data from each lane is de-scrambled independently and goes through the logic to detect link power state transitions.
3. The lanes are then de-skewed by passing the data through FIFOs that are aligned on SKP Ordered Set (Gen1/2) and SOS Ordered Set (Gen3 and higher). The lanes are aligned as a single unit.
4. The de-skewed data is decoded by a frame decoder.
5. The frame decoder removes the SOP/EOP framing delimiters from the packet and aligns them on the internal data path. The frame decoder is designed to deal with varying link widths and all possible alignments of packets on the lanes.
6. The decoded data is passed to the Data Link Layer with indications such as packet-type and errors detected (if any).
7. The received data from each lane is also routed to a Link Training Receive State Machine, which is responsible for detecting and decoding training sequences received from the lane.

8 Each of these state machines passes on information extracted from the training sequences to the LTSSM.

The transmit operation of the Physical layer is as follows:

1 Data arrives from the Data Link Layer.

2 The data passes through a frame encoder, which adds SOP and EOP delimiters to the packets and stripes the data on the lanes. The frame encoder is capable of dealing with varying link widths and all legal alignments of packets on the lanes.

3 The outgoing packets from the frame encoder are multiplexed with training sequences generated by the LTSSM.

4 The multiplexer disables the data path from the frame encoder during link training. It allows the LTSSM to control the lanes. Each lane has a separate scrambler to transform its data before entering the PIPE interface.

5 The precoding block performs the receive and transmit precoding as defined in the PCIe Specification. The outgoing PIPE interface is configured for 32bits per lane for Gen1/2/3/4.

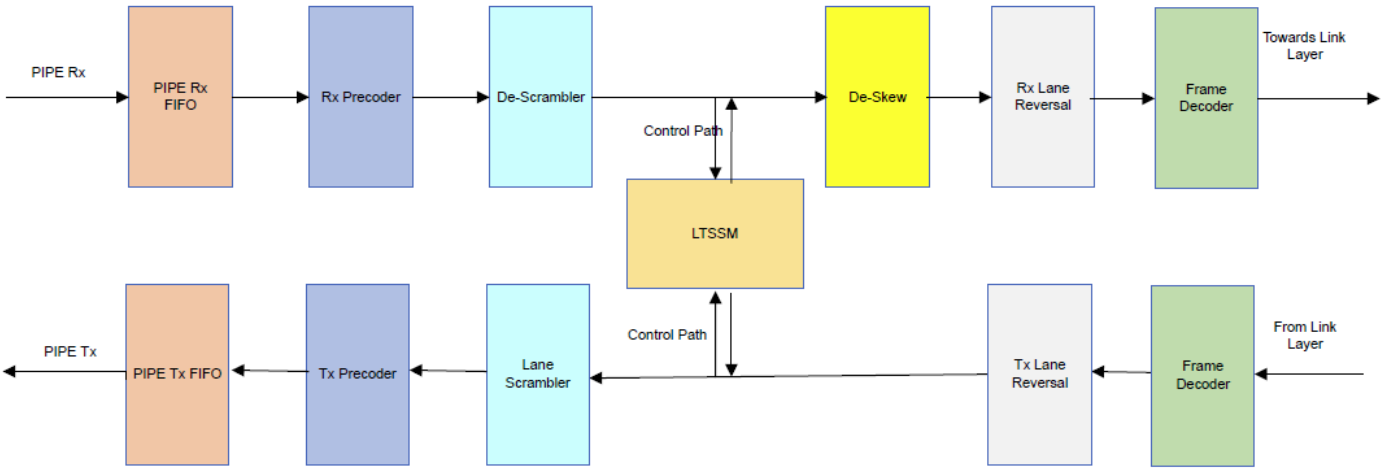


Figure 7-4 Physical Layer Block Diagram

7.4.4 Data Link Layer

The receive operation of the DLL is as follows:

1 The data from the link passes through a decoder state machine for each link.

2 The decoders verify the integrity of packets by matching the received CRC with the generated CRC, and comparing their sequence numbers with the expected values (for TLPs).

3 After comparing and removing the link-layer CRC, the DLLP decoders pass on the received packet to its target module based on the packet type.

4 All received TLPs with their sequence number and LCRC fields-removed are passed on to the Transaction Layer.

5 Data Link Layer acknowledgements ACKs and NAKs are processed within the Data link layer itself, and credit DLLPs are sent to the flow control module.

6 After the CRC check, the received Data Link Layer acknowledgements are sent to the transmit side of the Data Link Layer for processing.

7 The logic on the transmit side matches the acknowledgements with the outstanding packets and deal with all the error conditions.

8 The ACKs and NAKs for the received TLPs are sent to the transmit side of the Data Link Layer, where they are multiplexed with the outgoing TLPs.

The transmit operation of the DLL is as follows:

1. The received data from the Transaction Layer is formatted for transmission to the Physical Layer by inserting a sequence number and CRC.
2. The formatted TLPs are multiplexed with outgoing DLLPs such as ACKs/NAKs and credit packets, and sent to the Physical Layer over a data path shared by both links of the core. The transmit side also contains the replay buffer associated with the link, which is responsible for re-transmitting packets when needed. The replay buffer uses an external single-port RAM for storing the packets. The single-port RAM is used as a replay buffer. It can either be written or read. It never writes and reads simultaneously. There is also an internal pointer RAM for keeping track of packets stored in the replay buffer.

The Data Link Layer also contains logic to generate Power Management DLLPs to facilitate transitions of the link to L1 and L2 states as part of the PCIe Power Management.

7.4.5 Transaction Layer

Transaction Layer (TL) handles TLP buffering, Ordering, Credit gating, PCIe tag check and error checks.

7.4.5.1 TL Receive Side Operation

The following are high level details of operations done on Transaction layer receive side.

1. TLPs are decoded and checked for malformed errors, length mismatches, ECRC errors.
2. Order of received TLP is stored. It is used to enforce ordering while packets are read from receiver buffer and transferred to application logic.
3. TLPs are stored in store-and-forward RAM. The 128bit data segments are used to efficiently utilize RAM space if small TLPs are received in a cycle. There is a separate receive buffer RAM for Posted, Non-Posted and Completion packets.
4. The TLPs are split into the following separate streams and forwarded for further processing:
   - Posted.
   - Non-Posted.
   - Completion.
5. The receive flow control parameters of the core (payload and header credit for the three types of packets: Posted, Non-Posted and Completion) are set based on the available space in the receive buffers (stored in RAM). The flow control protocol ensures that the receive buffer do not overflow. The receive buffers communicate their state to the flow control module so that when the packet is forwarded out of the receive buffers, the corresponding credit becomes available and can be advertised to the link.
6. Based on the configuration, up to 8 OW aligned TLPs are possible to receive from PCIe link side. Transaction layer converts those packets into 128164bit aligned 1/2/4 TLPs per cycle based on the selected configuration.

7.4.5.2 TL Transmit Side Operation

1. Posted, Non-Posted and Completion (SC) packets arrive from the Application logic over separate interfaces.

2. Logic within the Transaction Layer multiplexes the packets, inserts ECRC if enabled, and forwards them to the Data link layer.
3. The Completions and messages generated within the core are also multiplexed on the same data path to the Data link layer.
4. Credit gating logic blocks the TLPs if the sufficient credits are not available from the remote device.
5. PCIe Ordering of TLPs are enforced from HLS (Posted, Non-posted, Completion) and interrupt interface towards PCIe link in this layer.

7.4.6 Host Adaptation Layer Operation

The following features in Host Adaptation Layer (HAL) eases application integration with PCIe IP:

1 Targeted PF is decoded after comparing all the BAR registers and the TLP address.

2 PCIe tags from outbound request towards PCIe link is stored and compared with completions for error checks. These are also used to generate completion timeouts, if completions are not received from PCIe link within a programmed time.

3 Endpoints are expected to accept all completions from PCIe link. The Throttling logic present in this layer stops outbound Non-Posted TLPs, if space for receive completion buffer is insufficient. This is a useful feature, if application logic back-pressure TLPs from PCIe-link.

4 All read and write requests to configuration registers are routed to the register set of the function addressed by the request, and returns completion packets back to the link.

5 All interrupt related messages are processed by a separate interrupt processing module, which controls the Interrupt interface of the core. The error messages are processed by an error handling module. There are also blocks to generate messages internally based on programmable registers.

6 Logic to generate UR Completion towards PCIe link for unsupported requests detected in IP.

7.4.7 Application Layer

The Application Layer of the core provides interface to the application logic implemented outside the IP, insulating you from the complexity of the PCI Express protocols. The Application layer provides the following key interfaces:

7.4.7.1 AXI4 Interface

The AXI4 interface connects to the existing HLS interface through an AXI4 Bridge logic. This interface is based on the AMBA AXI4 specification. This bridge implements all the PCIe and AXI4 ordering rules, any exceptions to this are explicitly stated.

7.4.7.2 APB

The IP provides access to the register's implemented PCIe core through a 32-bit user selectable APB bus. Through this bus, a local processor may access and modify the contents of the IP and access the PCIe Specification Configuration's registers associated with the PCIe Functions. The core adheres to protocol as defined in the *AMBA®APB Protocol Specification or AMBA®AX/ and ACE Protocol Specification Issue E.*

7.5 PHY Overview

The PHY macro is an implementation of a PIPE compliant PCS, supporting components (uController, TAP controller, APB interface, PHY power control and Clock/Reset controller) and Sierra7 PMA. Figure 7-5 illustrates a top-level block diagram for the PHY macro, and its interfaces. The following are high level points worth noting in this figure.

- The PIPE interface is the interface which connects to PIPE-based controllers.
- The PHY and PMA isolation and control modules serve two purposes.
  - They provide for two levels of isolation testing. These are PMA only and full PHY macro. When in isolation mode, PHY and PMA signals are under the control of CDB registers. The setting/reseting of isolation modes is considered static and this should not be done when in an active operating mode.
  - These modules also contain general control and status registers for the PCS and PMA.
- The Async APB Interface Slave module is part of a two module system that allows the internal CDB registers to be writ-ten and read by either the external APB interface or the integrated TAP interface. In normal mode (non-isolation), both interfaces can access the internal CDB, with the TAP having higher priority. When the PHY internal APB clock is sourced from the TAP clock (bit[1] in PHY macro diagnostic control register), only the TAP can access the internal CDB.
- The PMA boundary scan function is capable of being controlled either by the internal TAP controller, or an external TAP, under the control of the bscan_ext_select pin.

7.5.1 Features

Supported standards:

- PIPE interface supporting: PCIe Gen 1, 2, 3 and 4
- Supported configurations: PCIe x1, x2, and x4 links are supported, with the following restriction:
  - PCIe: For multi-lane links (i.e., x2 and x4), all lanes within a link must be adjacent.
- PIPE PHY interface is compliant with the PCIe PIPE Specification, Revision 4.4.1.
  - Supports source-synchronous PIPE interface, where PIPE outputs are synchronous to the PIPE clock output and PIPE inputs are synchronous to a separate PIPE clock input.
- Supports PCIe Gen4 lane margining through the PIPE message bus.
- Supports a 16-bit AMBA APB interface for configuration and diagnostics.
- Supports the PCI-SIG “L1 PM Substates with CLKREQ” ECN via the PIPE or sideband interface.
- Supports selectable 16-bit or 32-bit PIPE interface data width for PCIe Gen1/2/3/4.
  - Supports compile-time selection of 16-bit or 32-bit internal PCS data path width for PCIe Gen3/4. When 32-bit data path width is selected, only a 32-bit PIPE interface is supported. The 32-bit data path reduces internal clock frequency, while the 16-bit data path optimizes routing resources.
- For automotive safety applications, supports monitoring of PHY power state and data rate transitions, with interrupt generation upon failure.

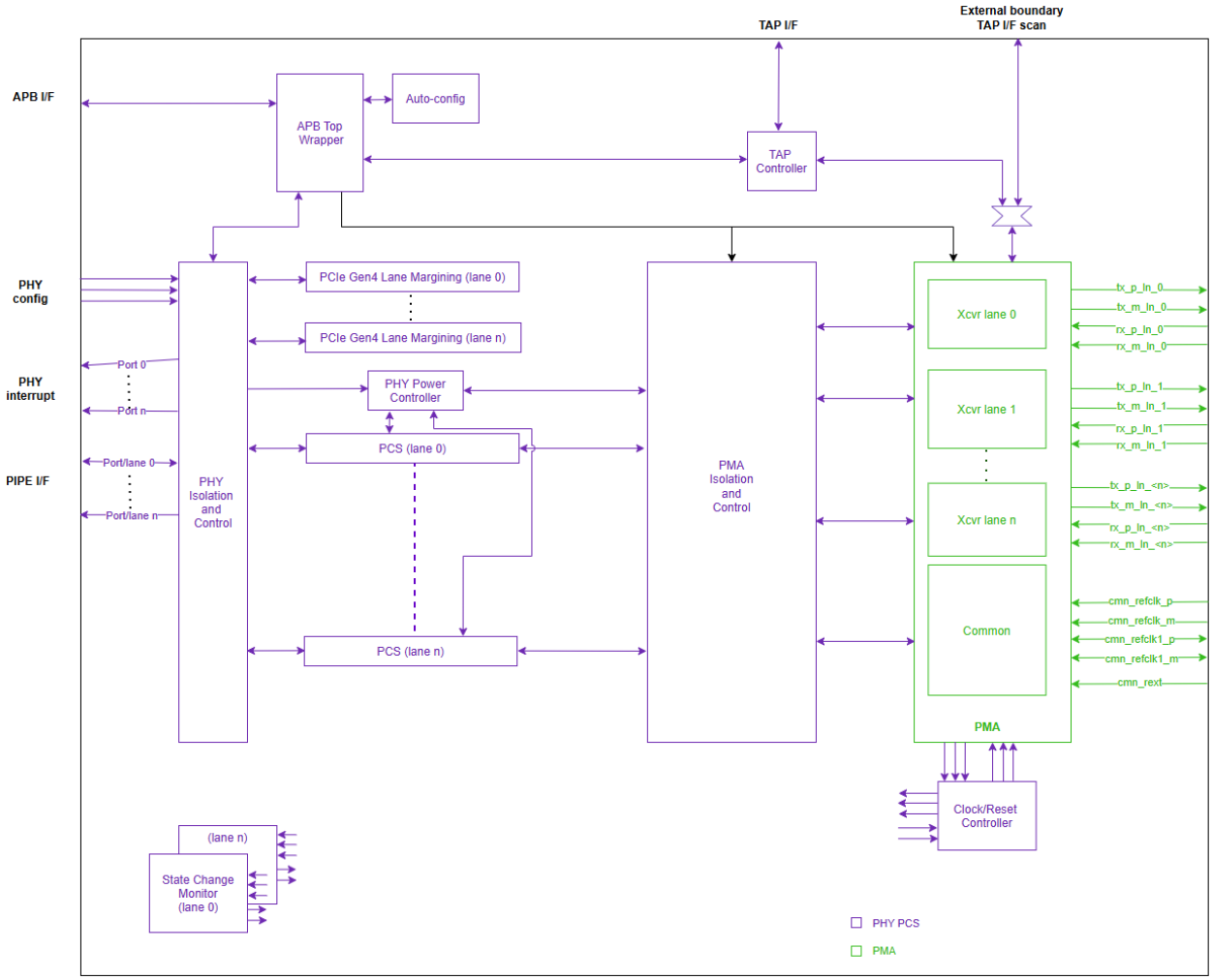


Figure 7-5 PCIe PHY Block Diagram

7.5.2 PMA Description

The macros implement the functionality generally attributed to the PMA sublayer, including:

- High-speed clock generation
- Serialization of parallel word data into a single-bit stream on calibrated termination media pins with adjustable drive levels and optional pre-cursor and post-cursor emphasis capability
- Related media pin support functions such as receiver detection and idle detection.
- Reception of single-bit data streams on calibrated termination media pins with clock recovery and optional 5-tap decision feedback equalization and de-serialization of this data into programmable width words.
- Support features for these functions including bias generation, automatic calibrations, and staged power down levels.

This macro does not include features generally attributed to the PCS sublayer such as encoding / decoding data with various codes such as 8B10B, comma detection, data scrambling, or rate matching FIFOs. These features are available in Cadence PCS Layer IP.

Figure 7-6 SerDes Macro Block Diagram shows an overall block diagram of the SerDes macro. Macros are organized around two main sub mod-ules, the Common and the Lane. The Common sub module includes any resources shared by all the Lanes including con-figuration through the AMBA

TM APB interface, biasing, resistor calibration, and portions of the high-speed clock synthesis. 7.5.3 Common Module provides additional details regarding the Common module.

The Lane sub module houses the link configuration and initialization, final high-speed clock synthesis, and paired transmit and receive paths from parallel data words to serial data and vice-versa. 7.5.4 Lane Module provides details of this module. Multiple Lanes may be included within a particular SD0302 macro configuration, all supported from a single Common sub module. The maximum number of Lane sub modules supported is sixteen per Common. Adjacent Lanes may be may be operated together as a multi-lane link or Lanes may operate independently with some restrictions. There is support to operate multiple standards simultaneously within a single Macro with restrictions. In such a case the differing standards are assigned to different multi-lane Links. Transmit and receive paths are largely isolated from each other within a Lane; however, biasing and clock generation is shared between the two paths as well as various loopback points for testability. Within a given Lane, the nominal data rate for transmit and receive is expected to be the same. The architecture supports simultaneously operating entire Lanes at different rates with some restrictions. The Macro supports low power states defined in the PCIe, and Ethernet standards. Individual lanes within a multi-lane link may be disabled or placed in standby at boot and on the fly (with restrictions The SD0302 series of macros are related by base architecture. Each macro may be differentiated by process, metal stack-up, lane configuration, wire-bond or flip-chip, and by bump pitch.

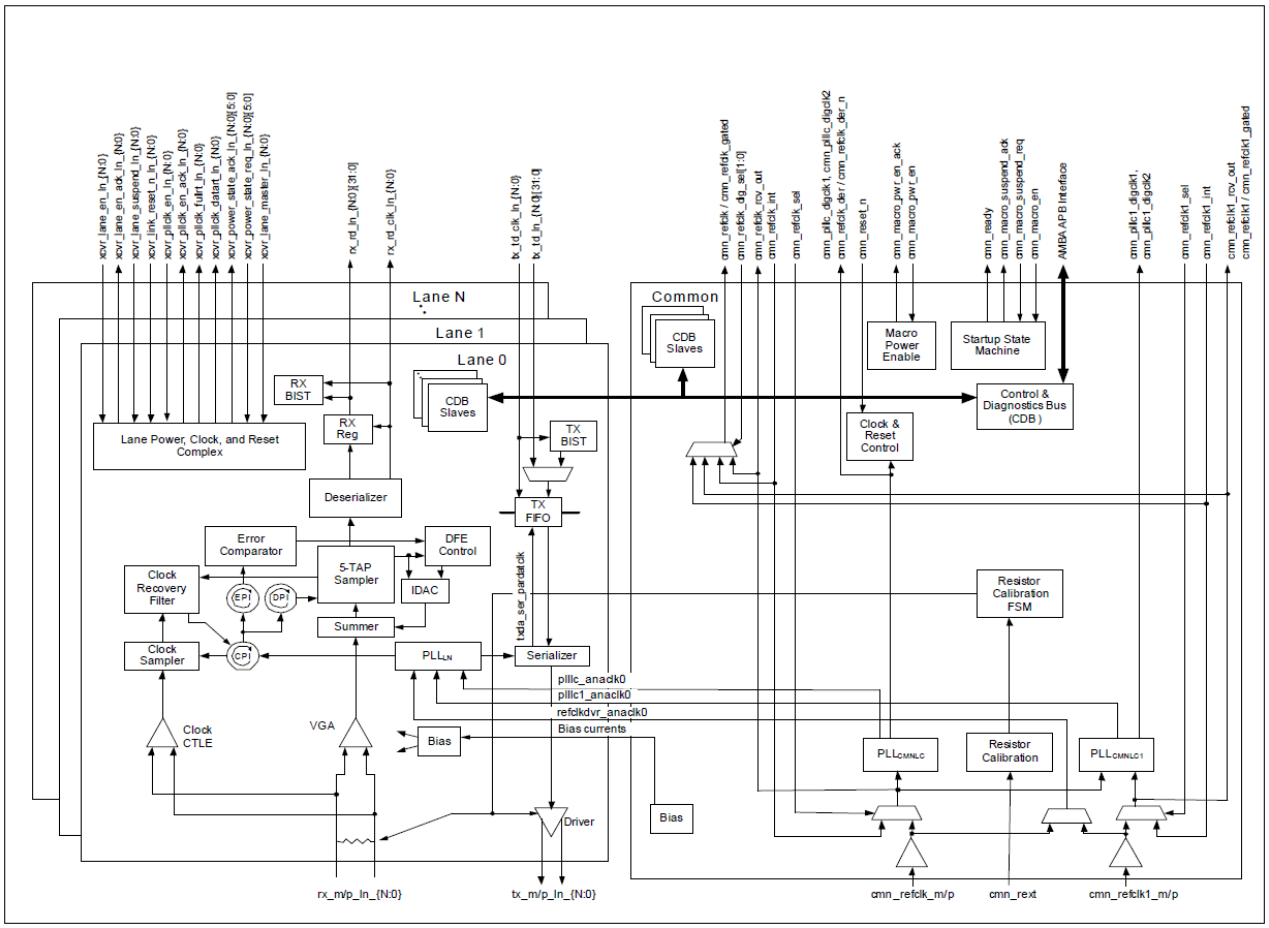


Figure 7-6 SerDes Macro Block Diagram

7.5.3 Common Module

The common module implements functions that are shared by all lanes:

- Macro startup
- Configuration and diagnostics through AMBAAPB
- Bias generation
- Resistance calibration
- Initial clock synthesis
- Analog test bus decode and digital conversion
- Clock gating and reset synchronization and distribution

Figure 7-7 Common Module Block Diagram is a high-level block diagram showing both analog and digital components. The following paragraphs discuss the various submodules shown within this diagram.

The Macro Power Enable State Machine provides for initial hold-off of the internal power gated supplies while the bumped supplies are present and the Macro is dormant. The Startup State Machine then automates the initialization process of the Common submodule functions and provides a means to turn off the internal power islands during a cmn_macro_sus-pend_req. Both the Macro Power Enable State Machine and Startup State Machine control the power islands and any required isolation through the Common Power Island State Machine.

The Configuration and Diagnostic Bus (COB) Controller implements the 16-bit AMBA APB interface for the Macro. One set of related registers of particular interest is the bank of read-only Macro ID registers that identify the macro, and its configuration. These include the *macro_id_type, macro_id_rev,* and *macro_id_mfg* registers.

When enabled under Startup State Machine control, the Biasing and Bandgap circuit provides current to all analog blocks inside the Macro.

Resistor calibration is one of the steps automatically initiated by the Startup State Machine. It is required to trim the termina-tion resistance values in receive and transmit paths of each lane. This function is implemented by the Resistor Calibration Controller, an analog comparator, and a resistor external to the Macro from the cmn_rext bump to ground. If two Macros are used within a single SOC, the resistor calibration from one of them may be shared with the other to eliminate the need for a sec-ond external resistor. In such a case, the slave PMA should set cmn_rescal_insel logic one, and the cmn_res-cal_code_out[S:0] from the master PMA should then be routed within the SOC to the slave PMA's cmn_rescal_code_in[S:0]. If the master PMA would be disabled when the slave is expected to be active, additional SOC core logic outside the PMA is required to retain the master's calibration code during the period it is disabled.

Clock synthesis of internal clocks is performed by two PLLs that operate off a provided reference clock.

The common module implements the analog test bus (ATB) decoding, houses the core_atb0 and core_atb1 analog core-side output pins, as well as providing means to drive the ATB out on cmn_refclk1_m and cmn_refclk1_p Macro bumps. also contains an ADC and related circuitry to provide for the ATB bus values being read over the COB bus. Please refer to 7.5.5 Analog Test Bus (ATB) for more details aboutATB.

Finally, clock gating, and reset resynchronization functions are all implemented in the Common Clock and Reset controller block. Transmit Reset Release pulses are generated and distributed out to the lanes (via the High Speed Reset Release State Machine and High Speed Reset Generator) such that the skew between lanes is minimized.

A process monitor circuit with several ring oscillators is included. This circuit is accessed through the COB registers. A signal detection oscillator is instantiated, supplying 500MHz

clocks to the lanes to be used in digital signal detection filtering, LFPS detection filtering, and LFPS generation.

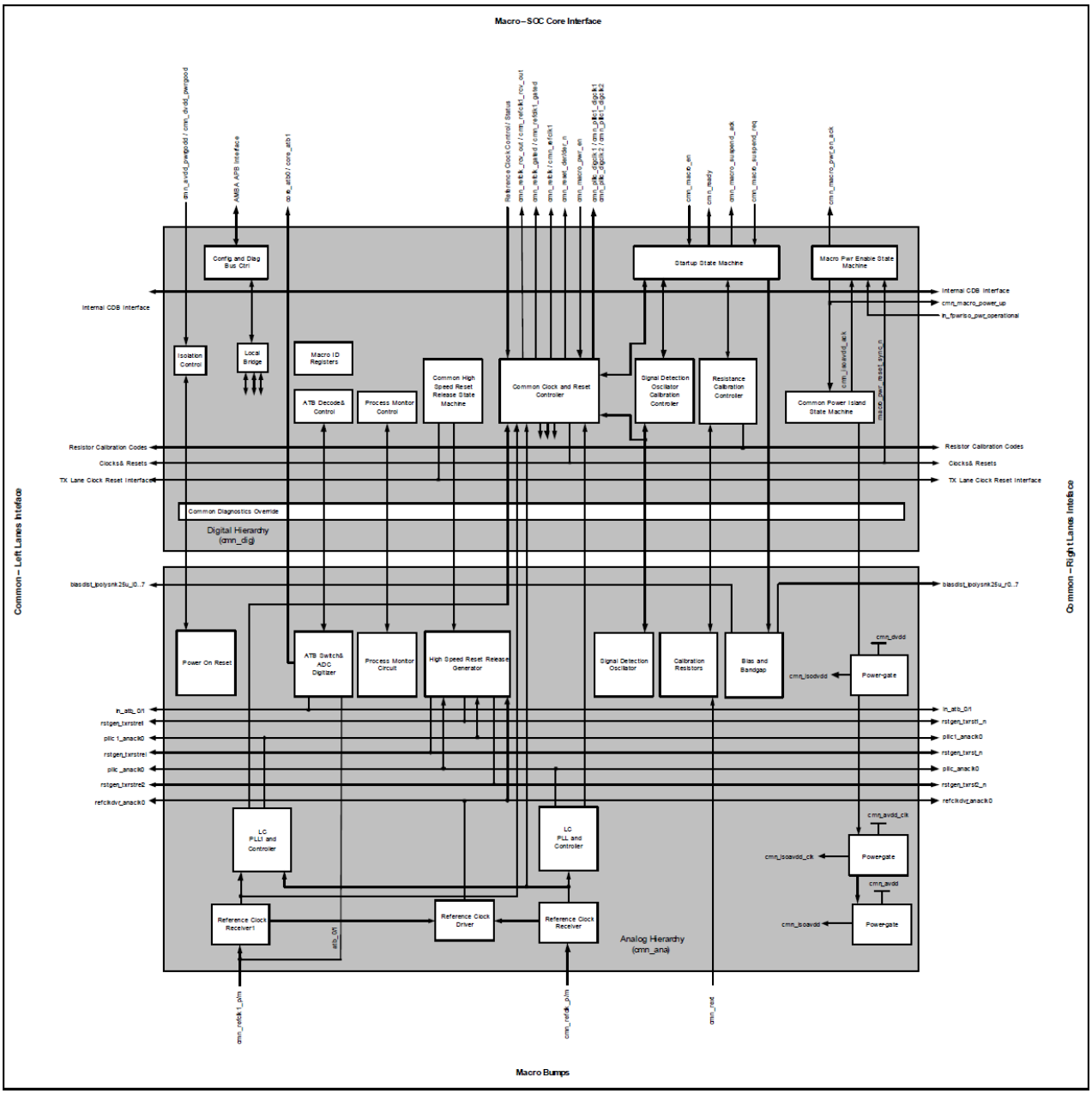


Figure 7-7 Common Module Block Diagram

7.5.4 Lane Module

The lane module contains a transmit serializer and receive deserializer data path. These are described in succeeding sub-sections 7.5.4.1 Transmit Overview and 7.5.4.2 Receive Overview. The following functions found at the lane level of hierarchy:

- Final high-speed clock synthesis
- Clock and reset management
- Configuration and diagnostics through the internal COB bus back to the Common's AMBA APB
- Standard Mode Decoding
- Power-gating and block level power controls
- Loopback Control

- Analog test bus decode

Figure 7-8 is a high-level block diagram showing both analog and digital components. The following paragraphs discuss the various sub-modules shown within this diagram.
Clock synthesis of the full data-rate frequency used in the transmit path, and the datarate frequency used in the receive path is performed by the PLL_{LN} that operates off refclkdvr_anaclk0, plllc_anaclk0, or the plllc1_anaclk0 provided from the Common module. Selection of which reference to PLL_{LN} to use is through *pl/ctrl_anaclk0_sel_mode0/1/23_preg* COB registers and the Link's master Lane xcvr_standard_mode_ln_{3:0}[2:0] value.

This circuitry generates and manages the clocks used within the Lane, including various data-rate clocks and reset signals.

The Lane clock and reset management sub-module supports multiple clock domains and associated reset controls.

It also includes the Transmit Reset State Machine and the High-Speed Reset Generator, which operate under the control of the Power State Machine to coordinate transmit resets across all Link Lanes, ensuring that lane-to-lane skew requirements are met.

Macro pins xcvr_standard_mode_ln_{3:0}[2:0] and xcvr_data_width_ln_{3:0}[2:0] are distributed to the Standard Mode Decoder to provide rate-based control and analog trims when changing between PCIe Gen modes. Controls and trims include:

- PLLLN reference selection, feedback divider ratio and subrate selections.
- PI calibration settings and analog trims.
- Clock-path CTLE, data-path error comparator, data-path sampler trims.
- Standard-based equalization engine selections.

Internal power-gates throughout the Lane are controlled centrally in the Lane Power Controller. The power-gates are normally controlled through signals from the Startup and Power State Machine Complex within the Common module in conjunction with the rx_lane_en_ln_{3:0} and tx_lane_en_ln_{3:0} Macro pins; however, direct control of these switches is also provided by dedicated Macro pin, ln_pwrgate_en_ln_{3:0}.
The Power State Machine in the master Lane provides power-state entry and exit control for all Lanes within the Link. The Lane Power Controller receives signals from the Power State Machine for controlling the individual analog sub-modules based on the selected power-state. To override these assignments, the Lane Power Controller definitions may be redefined through COB access.
The ATB function is documented in 7.5.5 Analog Test Bus (ATB).

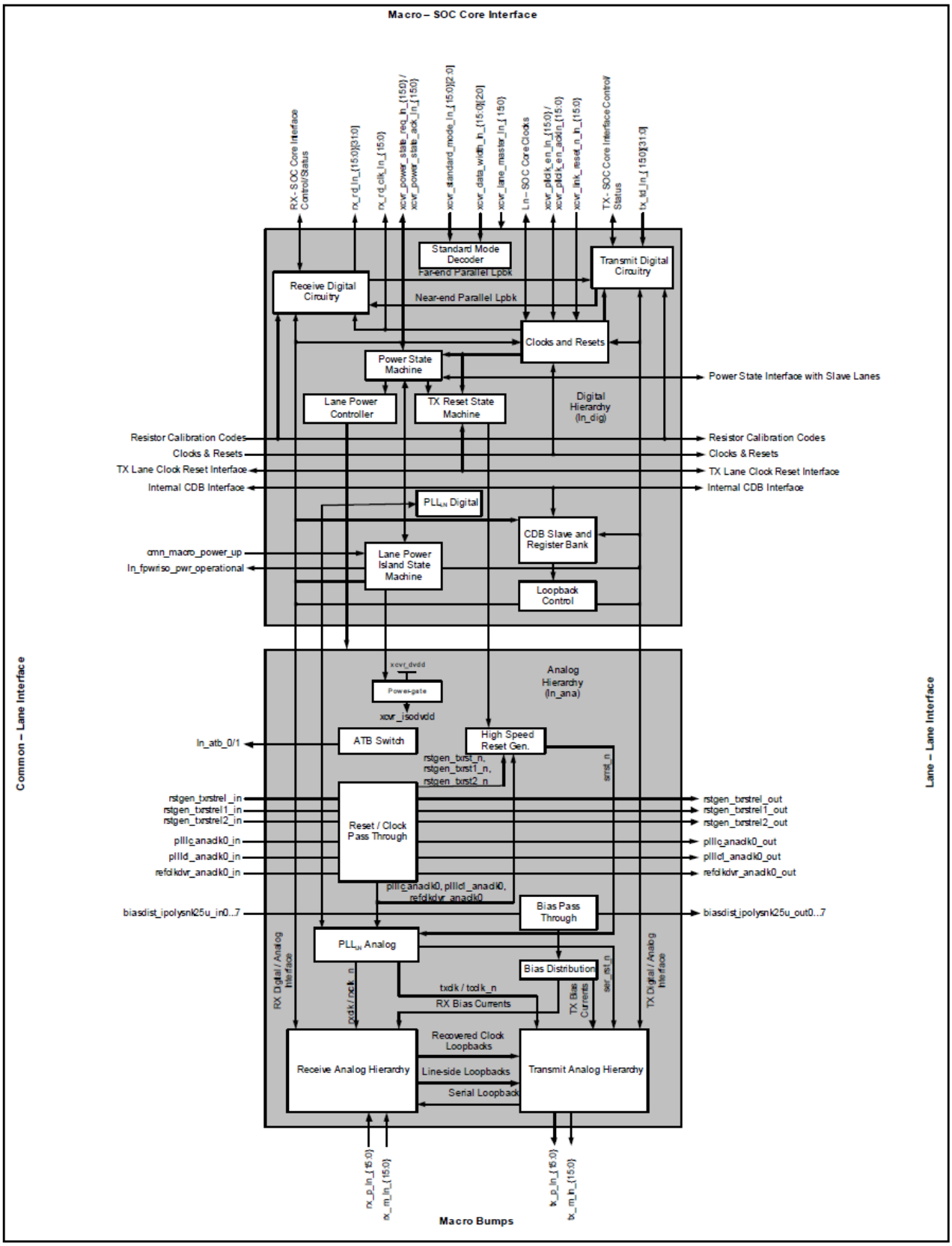


Figure 7-8 Lane Module Block Diagram

7.5.4.1 Transmit Overview

The main function of the transmit path is to serially transmit the data from the input parallel interface, tx_td_ln_{3:0}[31:0] on the tx_p/m_ln_{3:0} bumps. Figure 7-9 shows a block diagram of the transmit path. The supported parallel data widths are 16, 20 and 32 bits, selected at the

Link level through xcvr_data_width_ln_{3:0}[2:0]. The data is transmitted LSB first and its polarity can be inverted or not via the tx_differential_invert_ln_{3:0} pin.
The "Driver Related Signals" in the figure are constituted of several sub-functions. The first of these are the PIPE LocalFS[5:0] and LocalLF[5:0] support signals required for PCIe Gen 3 and Gen 4 modes. tx_local_fs_ln_{3:0}[5:0] and tx_local_lf_ln_{3:0}[5:0] Macro pins provide direct support for these PIPE signals. They are valid only once resistor calibration has been completed as indicated by the assertion of cmn_ready. The assertion of tx_low_power_swing_en_ln_{3:0} may modify these values somewhat if the transmit driver's amplitude boost circuitry had been enabled via *drvctrl_ampboost_en_preg*.
When operating in PCIe Gen 3 and PCI Gen 4 the Macro supports the PIPE's preset to coefficient mapping function with an interface comprised of tx_get_local_preset_coef_ln_{3:0}, tx_local_preset_index_ln_{3:0}[3:0], tx_local_tx_co-ef_valid_ln_{3:0}, and tx_local_tx_preset_coef_ln_{3:0}[17:0]. The latter three signals are reused and along with tx_get_local_init_coef_ln_{3:0} to form a similar interface for returning init coefficients in 10GKR modes of operation.
For PCIe Gen 3, PCIe Gen 4 and the 10GKr modes, tx_deemphasis_ln_{3:0}[17:12] provides the post emphasis setting, tx_deemphasis_ln_{3:0}[11:6] provides the main cursor setting and tx_deemphasis_ln_{3:0}[5:0] provides the pre emphasis setting per the PIPE TxDeemph[17:0] definition. In PCIe Gen3, Gen4 the settings are in the FS bases scale per the standard and boundary conditions for each of the cursor are also per the standard. In 10GKr the settings are in similar units as established by the init coefficients. The following boundary conditions must be maintained for this standard:

$C_0 + |C_{+1}| + |C_{-1}| <= FS$

$C_0 - |C_{+1}| - |C_{-1}| >= 0.1875 * FS$

$C_0 >= 0.5625 * FS$

$|C_{+1}| <= 0.375 * FS$

$|C_{-1}| <= 0.875 * FS$

For standards other than PCIe Gen 3, PCIe Gen 4, the de-emphasis values are calculated based on the resistor calibration to conform to the PIPE TXDeemph[1:0] (tx_deemphasis_ln_{3:0}[1:01)deemphasis selections for PCIE Gen 1 and Gen 2.
The tx_vmargin_ln_{3:0}[2:0] and tx_low_power_swing_en_ln_{3:0} pins are used to control the transmit driver's margin value in PCIe modes per the PIPE's TxMargin[2:0] and TxSwing signals. Note that per the PIPE specification, tx_low_pow-er_swing_en_ln_{3:0} is not expected to be asserted in PCIe Gen 3, Gen 4.
Also provided is the tx_high_z_ln_{3:0} pin which when asserted forces the driver's tx_td_p/m_ln_{3:0} bumps to their high-impedance state. Electrical idle per the PCIe is supported with the assertion of the tx_elec_idle_ln_{3:0} pin. tx_elec_idle_ln_{3:0} has lower priority than the tx_high_z_ln_{3:0} pin.
Receiver detection per the PCIe is supported. The function may be enabled as required within the A2 and A3 power-states by asserting tx_rcv_detect_en_ln_{3:0}. tx_rcv_detect_en_ln_{3:0} should remain asserted until tx_rcv_detect_done_ln_{3:0} transitions high at which point the result, tx_rcv_detected_ln_{3:0}, will be valid and remain valid until tx_rcv_detect_en_ln_{3:0} is deasserted. tx_rcv_detected_ln_{3:0} will be high if a far-end receiver was found; otherwise it will remain low.

Not shown in Figure 7-9 is an alternate mode providing lower data path latency. The alternative mode termed Ultra-low Latency mode, bypasses the TX Sync FIFO altogether; the tx_td_ln_{3:0} data is written directly to the analog serializer. The tx_td_ln_{3:0}[31:0] and tx_elec_idle_ln_{3:0} must be delivered synchronous to the local tx_td_clk_out_ln{3:0} Macro output clock. Only 20-bit data widths are supported for this Ultra low latency mode.

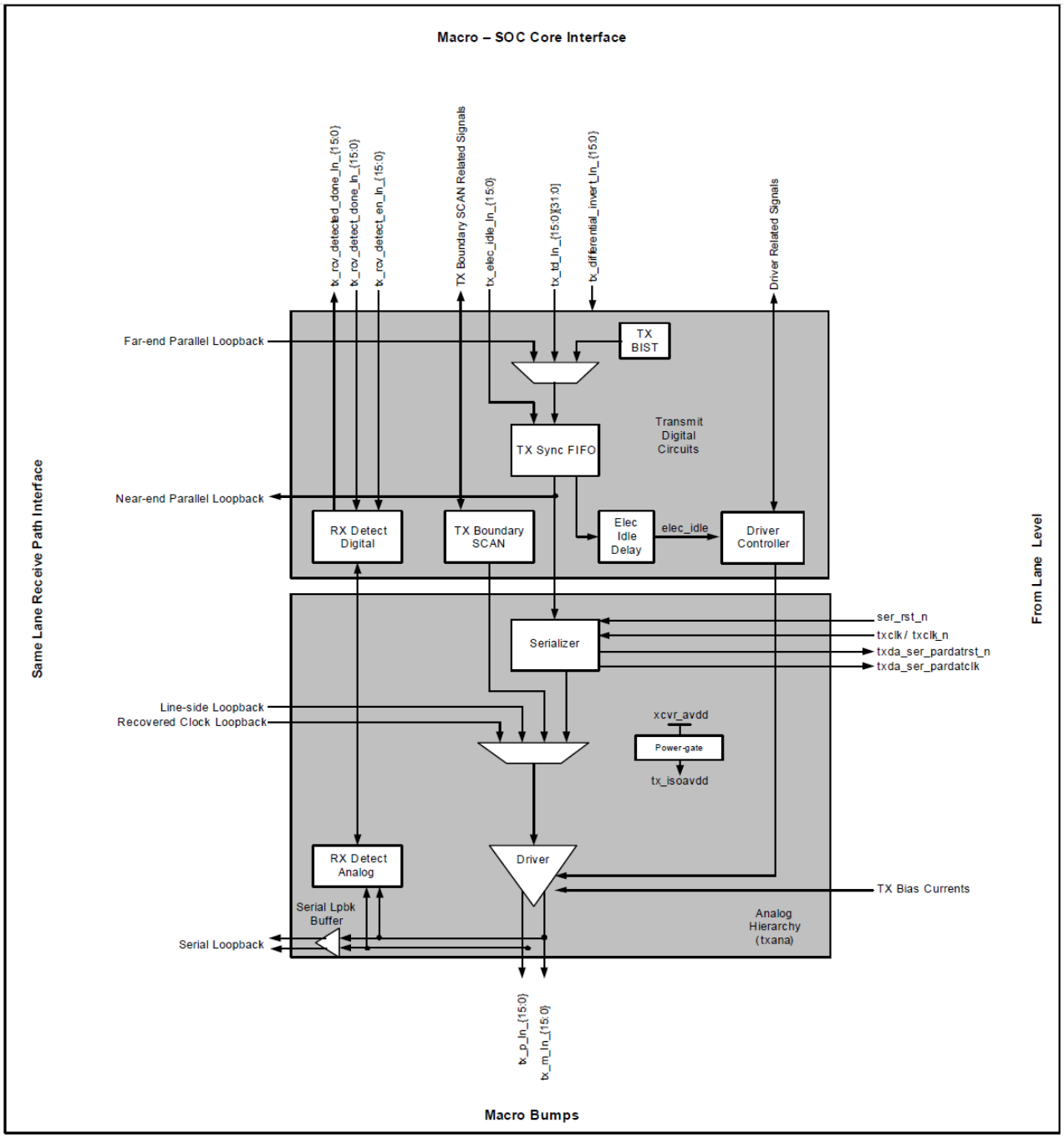


Figure 7-9 Transmit Path Block Diagram

7.5.4.2 Receive Overview

The main function of the receive path is to recover serial data from the rx_p/m_ln_{3:0} bumps and present the data at the rx_rd_ln_{3:0}[31:0] parallel interface. Figure 7-10 shows a block diagram of the receive path. The supported parallel data widths are 16, 20 and 32 bits, selected at the Link level through xcvr_data_width_ln_{3:0}[2:0]. The parallel data is generated LSB first and its polarity can be inverted or not via the rx_differential_invert_ln_{3:0} pin.

PCIe Gen 3, PCIe Gen 4, are considered "closed-eye" standards. They require receive path equalization during training periods which include both near-end receiver equalization and far-end transmitter emphasis optimization. The macro supports these standards through the Equalization Evaluation Interface mentioned in the figure and listed in Table 7-2 below. Equalization evaluation may be rerun after the initial training as directed through the interface.

Table 7-2 Equalization Evaluation Interface Signals

Macro Inputs	Macro Outputs
rx_eq_eval_ln_{3:0}	rx_eq_eval_status_ln_{3:0}
rx_eq_training_data_valid_ln_{3:0}	rx_eq_eval_complete_ln_{3:0}
rx_eq_eval_cnt_rst_ln_{3:0}	rx_link_eval_fb_dir_change_ln_{3:0}[5:0]
rx_invalid_request_ln_{3:0}	

For all other standards equalization is not run, though some basic data path offset and sampling alignment processes are still run initially after clock recovery. These are dubbed "open-eye" standards.

The rx_p/m_ln_{3:0} bump impedance is normally set to a nominal 500 through resistor calibration run during the startup sequence. However when the rx_termination_ln_{3:0} pin is set low, or the receiver is in boundary scan mode, or when if the specific lane or macro as a whole is disabled, any resistor calibration value is ignored and the termination is placed in high impedance.

A signal detection output, rx_signal_detect_ln_{3:0}, is provided for all enabled lanes. This output is asserted on the detection of valid high-speed serial data on the rx_p/m_ln{3:0} bumps. It is valid in A0, A1, and A2 power states and in the A4 power state when cmn_rx_sig_det_en_ext is asserted. By default the level detection and digital glitch filtering are set such that this output may serve as the source for PCIe's idle detection; however COB support exists to adjust these levels.

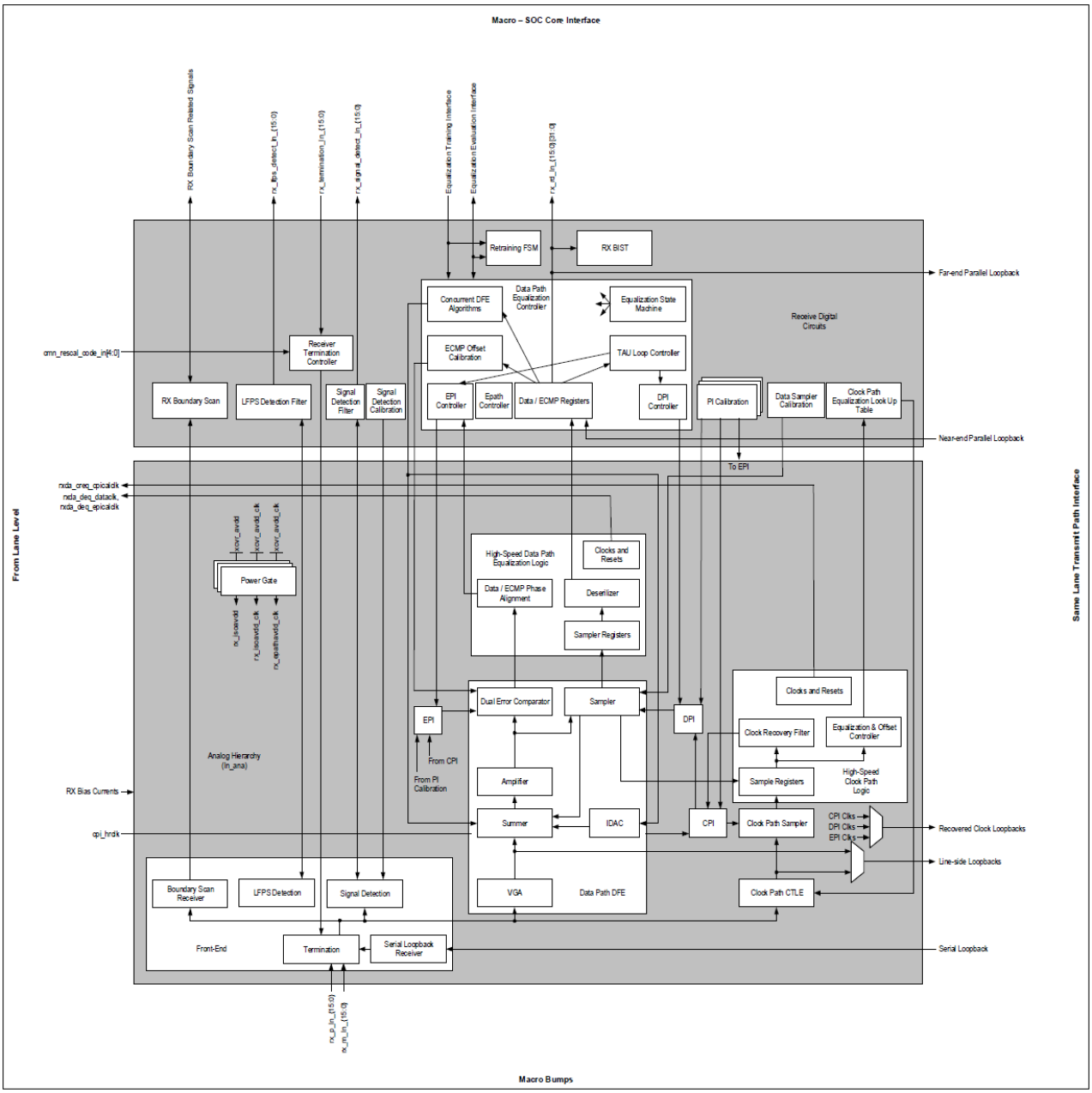


Figure 7-10 Receive Path Block Diagram

7.5.5 Analog Test Bus (ATB)

7.5.5.1 External ATB

An analog test bus (ATB) is available for driving and observing internal analog nodes either on the external cmn_refclk1_m and cmn_refclk1_p bumps, or on core side core_atb0 and core_atb1 analog outputs. Observation or control is provided through a series of transmission gates. Which internal test points may be accessed is controlled through the ATB control register, *cmn_atb_ctrl_preg*. The single bit *cmn_atb_ctrl_region_sel_preg* field selects between points in the common module or from one of the lanes. The four bit *cmn_atb_ctrf_lane_sel_preg* field selects the particular lane; it should be set to 4'd0 if *cmn_atb_ctrf_region_sel_preg* is set low. A particular component within the selected lane or common is selected through the 5 bit *cmn_atb_ctrf_component_sel_preg* field. The *cmn_atb_ctrl_atbsel_preg* four bit field selects a particular test point to be observed within the selected component. Supported components and their associated test points for the common module are defined in Table 7-3 Common

Component and Test Point Definitions. The lane compo-nent and test point definitions are found within Table 7-4 Lane Component and Test Point Definitions.
The ATB control register also has a single-bit field, *cmn_atb_ctrl_atben_preg,* which must be asserted for the test point selection to be activated and the test point switched on to the internal ATB bus. If observing the analog test bus on the cmn_reflck1_m/p bumps, the *cmn_atb_ctrl_bump_con_en_preg* field must also be asserted. If observing the analog test bus on the core_atb0/1 core-side analog output pins, the cmn_core atbesd_en_preg field must also be asserted. If neither cmn_atb_ctrl_bump_con_en_preg nor cmn_core_atbesd_en_preg is asserted, the internal ATB is not present external to the Macro and can only be digitized using the ATB digitizer ATB discussed in the next section.
The ATB bus itself resides solely on the external bump supplies, cmn_avdd, xcvr_avdd_ln_{3:0}. Since the ATB control register resides on the COB bus however, the cmn_isoavdd and rx_isoavdd_dig power-gated must be present as well.
Note: When measuring any ATB points sourced from the digital supplies, cmn_dvdd or xcvr_dvdd_ln_{3:0}, or sourced from internally regulated supplies such as those used by the PLLs, these source supplies can be elevated relative to the ATB supplies, cmn_avdd, and xcvr_avdd_ln_{3:0} to such an extent that ATB transmission gates leak through their body/gate connections and cause erroneous measurements.
The ATB bus has several stages of switches to limit the loading on when active. Each lane has a local atb0/atb1 bus which is isolated from the other lanes and from the common through the ln_ana_atb switch controlled through the active high lnda_atb_en. Any time a test point within a particular lane is selected for observation, that lane's lnda_atb_en must be asserted as well as the individual test point switch. Similarly, the common module's cmn_ana_atb switch isolates a local atb0/atb1 bus from all the lanes. The associated active high enable for this switch, cmnda_atb_en, must be asserted when-ever a test point within the common is selected for observation. Only a single lnda_atb_en or cmn_atb_en may ever be asserted when the ATB is in use. If the ATB bus is to be accessed on the cmn_refclk1_m/p bumps the cmnda_atbesd_en must be asserted. This enable is a direct assignment from the *cmn_atb_ctrl_bump_con_en_preg* ATB control register field. If the ATB bus is to be accessed on the core_atb0/1 core side output pins, the cmnda_core_atbesd_en must be asserted. This enable is a direct assignment from the cmn_core_atbesd_en_preg ATB control register field. Figure 7-11 ATB Bus Architecture shows graphically the organization of the ATB buses internally. To protect from internal floating nodes, the cmnda_atbesd_atbpulldn_n signal is asserted low whenever cmn_atb_ctrl_bump_con_en_preg, cmn_core_atbesd_en_reg, and cmn_atb_ctrl_at-ben_preg are all low, otherwise this signal is deasserted high.
All register fields residing in the *cmn_atb_ctrl_preg* COB address should be removed from scan to prevent damage to ana-log circuits.

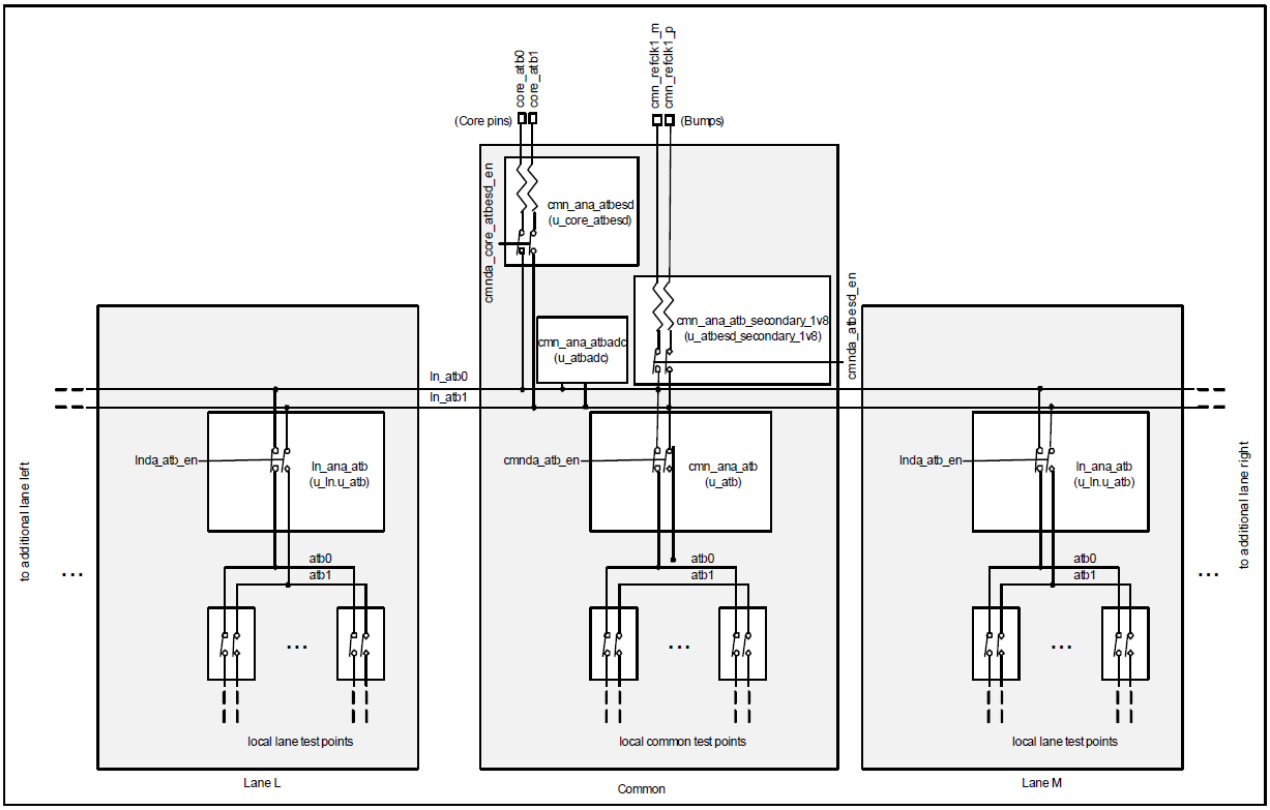


Figure 7-11 ATB Bus Architecture

Table 7-3 Common Component and Test Point Definitions

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
PLLC$_{MNLC}$	5'b0_0000		4'b0000	none	n/a	none	n/a
		3'b010	4'b0001	avdd_reg	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0010	vref	[0.46, 0.54] V	gnd	0V
		3'b010	4'b0011	vpoly300u	[0.46, 0.54] V	gnd	0V
		3'b101	4'b0100	isnk50u	[38, 65] uA	gnd	0V
		3'b010	4'b0101	clk1avdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0110	clk0avdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0111	hfavdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b1000	lfavdd	[0.76, 0.91] V	gnd	0V
		n/a	4'b1001 (used to select bypass mode. pllc_refclk to	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			cmn_plllc_ana-clk0)				
		3'b010	4'b1010	itank	[0.43, 0.57] V	gnd	0V
		3'b010	4'b1011	voff	[0.12, 0.26] V	gnd	0V
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
ATBADC	5'b0_0001	3'b010	4'b0000	avdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0001	rdac_out	[0, 0.170] V	ref_m	0V
		3'b010	4'b0010	ref_2x	[0.238, 0.322] V	ref_m	0V
		3'b010	4'b0011	ref_p/ref_m	[0.153, 0.207] V	ref_m	0V
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Resistor Calibration	5'b0_0010	3'b010	4'b0000	External resistor divide (in_p)	[0.20, 0.50] V	gnd	0V
		3'b010	4'b0001	Potentiometer voltage (in_m)	[0.20, 0.50] V	gnd	0V
		3'b101	4'b0010	cmn_avdd	[0.76, 0.91] V	gnd	0V
			4'b0011	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Bias Generation	5'b0_0011	3'b010	4'b0000	cmn_isoavdd_clk	[0.75, 0.91] V	gnd	0V
		3'b010	4'b0001	vref	[0.47, 0.53] V	gnd	0V
		3'b101	4'b0010	ipolysnk12p5u	[9, 16] uA	gnd	0V
		3'b010	4'b0011	vtempsense	[0.64, 0.94] V	gnd	0V
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
PLLcMNLC1	5'b0_0100		4'b0000	none	n/a	none	n/a
		3'b010	4'b0001	avdd_reg	[0.76, 0.91] V	gnd	0V

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
		3'b010	4'b0010	vref	[0.46, 0.54] V	gnd	0V
		3'b010	4'b0011	vpoly300u	[0.46, 0.54] V	gnd	0V
		3'b101	4'b0100	isnk50u	[38, 65] uA	gnd	0V
		3'b010	4'b0101	clk1avdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0110	clk0avdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0111	hfavdd	[0.76, 0.91] V	gnd	0V
		3'b010	4'b1000	lfavdd	[0.76, 0.91] V	gnd	0V
		n/a	4'b1001 (used to select bypass mode. plllc_refclk to cmn_plllc_ana-clk0)	none	n/a	none	n/a
		3'b010	4'b1010	itank	[0.43, 0.57] V	gnd	0V
		3'b010	4'b1011	voff	[0.12, 0.26] V	gnd	0V
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Process Monitor	5'b0_0101	3'b010	4'b0000	cmn_pmc_isoavdd	[0.75, 0.91] V	gnd	0V
			4'b0001	ro_div256_n	ring osc elk	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Signal Detection Oscillator	5'b0_0110	3'b010	4'b0000	nmos bias voltage	[0.36, 0.56]V	gnd	0V
		3'b010	4'b0001	pmos bias voltage	[0.4, 0.6]V	local power gated supply	[0.75, 0.91] V
			4'b0010	none	n/a	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
sd0302_atb instances at cmn_ana level measuring cmn_isoavdd	5'b0_0111	3'b010	4'b0000	cmn_isoavdd	[0.75, 0.91] V	gnd	0V
		3'b010	4'b0001	cmn_isoavdd	[0.75, 0.91] V	gnd	0V
			4'b0010	none	n/a	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
	5'b0_1000 - 5'b1_1111		4'b0000 - 4'b1111	unused	n/a	unused	n/a
Note: Core supply voltage measurements listed assume 0.85V nominal supply. Please substitute the nominal voltage for the particular process variation used.							

Table 7-4 Lane Component and Test Point Definitions

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
Bias Distribution	5'b0_0000	3'b010	4'b0000	xcvr_avdd_clk	[0.76, 0.91] V	gnd	0V
		3'b101	4'b0001	atb_ipolysrc12p5u	[9, 16] uA	none	n/a
		3'b010	4'b0010	gnd	0V	gnd	0V
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
PLLLN	5'b0_0001		4'b0000	none	n/a	none	n/a
		3'b010	4'b0001	avdd_lf	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0010	biasref	[0.46, 0.54] V	gnd	0V
		3'b010	4'b0011	gnd	0V	gnd	0V
		3'b101	4'b0100	ipolysnk12u5 (to gnd)	[9, 16] uA	gnd	0V
		3'b010	4'b0101	avdd_vco	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0110	avdd_lf	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0111	gmref	[0.325, 0.475] V	gnd	0V
		3'b010	4'b1000	coarse	[0.1,0.7]V	gnd	0V
		3'b010	4'b1001	fine	[0.325, 0.475] V	gnd	0V
		3'b010	4'b1010	avdd_lf	[0.76, 0.91] V	gnd	0V
		3'b010	4'b1011	vring	[0.25, 0.55] V	gnd	0V
		3'b010	4'b1100	vn2_bias	[0.2, 0.6] V	gnd	0V
		3'b010	4'b1101	vp2_bias	[0.4, 0.8] V	gnd	0V
		3'b010	4'b1110	avdd_hf	[0.76, 0.91] V	gnd	0V
			4'b1111	none	n/a	none	n/a
Transmit Analog	5'b0_0010	3'b010	4'b0000	boost bias isoavdd	[0.76, 0.91] V	boost bias ground	0mV
		3'b010	4'b0001	boost bias vbn voltage	[0.390, 0.481] V 448mV @ boost=4	boost bias ground	0mV
		3'b010	4'b0010	boost bias vbp voltage	[0.322, 0.460] V 377mV @ boost=4	boost bias ground	0mV

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
		3'b001, 3'b011	4'b0011	boost bias reference voltage	[0.400, 0.546] V	boost bias feed-back voltage	[0.400, 0.546] V
		3'b010	4'b0100	cml2cmos vbn bias voltage	[0.285, 0.420]	cml2cmos gnd	0V
		3'b010	4'b0101	cml2cmos isoavdd	[0.75, 0.91] V	driver gnd	0V
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
		3'b010	4'b1000	serializer tx_isoavdd	[0.76, 0.91] V	serializer gnd	0V
		3'b010	4'b1001	serializer xcvr_avdd	[0.76, 0.91] V	serializer gnd	0V
			4'b1010	none	n/a	none	n/a
		3'b010	4'b1011	serializer isoavdd_clk	[0.75, 0.91] V	serializer gnd	0V
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
		3'b010	4'b1110	hbridge isoavdd	[0.75, 0.91] V	hbridge isoagnd	0V
		3'b010	4'b1111	xcvr_avdd	[0.76, 0.91] V	gnd	0V
Receive Front End	5'b0_0011	3'b001 3'b011	4'b0000	Signal Detector vrefp	[0.468, 0.572] V	Signal Detector vrefm	[0.423, 0.517] V
		3'b010	4'b0001	Signal Detector vcm	[0.445, 0.550] V	Signal Detector gnd	0V
		3'b010	4'b0010	Signal Detector avdd_int	[0.75, 0.91] V	Signal Detector gnd	0V
		3'b010	4'b0011	Signal Detector vbias	[0.250, 0.350] V	Signal Detector gnd	0V
		3'b101	4'b0100	Signal Detector ipolysnk25u_atb	[17, 33] uA	none	n/a
		3'b101	4'b0101	Signal Detector icomp0src25u	[17, 33] uA	Signal Detector	0V

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
						gnd	
		3'b101	4'b0110	Signal Detector icomp1src25u	[17, 33] uA	Signal Detector gnd	0V
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
		3'b010	4'b1101	xcvr_isodvdd	[0.665, 0.825] V	gnd	0V
		3'b010	4'b1110	xcvr- dvdd	[0.675, 0.825] V	gnd	0V
			4'b1111	none	n/a	none	n/a
Receive CPI (Note the nominal values)	5'b0_0100	3'b001	4'b0000	cmref	[-0.25, -0.13] V (atb0-atb1)	avdd	[0.76, 0.91] V
		3'b101	4'b0001	iprobe (sinking)	[38, 65] uA	none	n/a
		3'b010	4'b0010	rx_isoavdd_clk	[0.75, 0.91] V	gnd	0V
		3'b010	4'b0011	rx_epathavdd_clk	[0.75, 0.91] V	gnd	0V
		3'b010	4'b0100	rx_isoavdd	[0.75, 0.91] V	gnd	0V
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Receive DFEA	5'b0_0101	3'b010	4'b0000	vga - nbias	[0.29, 0.47] V	gnd	0V

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
		3'b010	4'b0001	vga vga1vcm	[0.43, 0.65] V	gnd	0V
		3'b010	4'b0010	vga maingmvcm	[0.43, 0.65] V	gnd	0V
		3'b010	4'b0011	vga vref	[0.43, 0.65] V	gnd	0V
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
		3'b010	4'b1000	amp vcm	[0.43, 0.65] V	gnd	0V
		3'b010	4'b1001	amp nbias	[0.33, 0.54] V	gnd	0V
		3'b010	4'b1010	amp out_m	[0.43, 0.65] V	amp out_p	[0.43, 0.65] V
		3'b101	4'b1011	ampbias sinking iatb (from_rxisoavdd to 10k to gnd)	[4, 12] uA	none	n/a
		3'b010	4'b1100	smp rx_isoavdd_clk	[0.75, 0.91] V	gnd	0V
		3'b010	4'b1101	smp rxisoavdd	[0.75, 0.91] V	gnd	0V
		3'b010	4'b1110	ecmp rx_epathavdd_elk	[0.75, 0.91] V	gnd	0V
		3'b010	4'b1111	idac rx_isoavdd	[0.75, 0.91] V	gnd	0V
Receive DFE B	5'b0_0110	3'b101	4'b0000	idac sinking iatb (from_rxisoavdd to 10k to gnd)	[12, 37] uA	none	n/a
			4'b0001	none	n/a	none	n/a
			4'b0010	none	n/a	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Unused	5'b0_0111		4'b0000	none	n/a	none	n/a
			4'b0001	none	n/a	none	n/a
			4'b0010	none	n/a	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Clock Path CTLEA	5'b0_1000	3'b010	4'b0000	dcbias_vcm (adjustable with ctleclk_lvlshfttrim_preg)	[0.40, 0.63] V	gnd	0V
		3'b010	4'b0001	vcm (stg1 out)	[0.45, 0.66] V	gnd	0V
		3'b010	4'b0010	nbias (stg2)	[0.29, 0.54] V	gnd	0V
		3'b010	4'b0011	pbias (stg2)	[0.29, 0.54] V	gnd	0V
		3'b010	4'b0100	avdd (stg3)	[0.76, 0.91] V	gnd	0V
		3'b010	4'b0101	cmfbout (stg3)	[0.180, 0.560] V	gnd	0V
		3'b010	4'b0110	offsetadj_p (with creq_ctle_off-	[0.345, 0.575] V	gnd	0V

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
				setovr_preg[6:0] all zeros)			
		3'b010	4'b0111	offsetadj_m (with creq_ctle_off-setovr_preg[6:0] all zeros)	[0.345, 0.575] V	gnd	0V
		3'b010	4'b1000	vcm (stg4 out)	[0.400, 0.665] V	gnd	0V
		3'b101	4'b1001	ipolysnk12p5u_atb (adjustable with ctle-clk_mainbias-trim_preg)	[9.9, 17.0] uA	none	n/a
		3'b010	4'b1010	pbias (bias_main)	[0.35, 0.61] V	gnd	0V
		3'b010	4'b1011	pcasc (bias_main)	[0.37, 0.63] V	gnd	0V
		3'b101	4'b1100	ipolysnk12p5u_atb (bias_fixed)	[9.9, 17.0] uA	none	n/a
		3'b010	4'b1101	pbias (bias_fixed)	[0.37, 0.63] V	gnd	0V
		3'b010	4'b1110	pcasc (bias_fixed)	[0.37, 0.63] V	gnd	0V
		3'b010	4'b1111	avdd (bias_fixed)	[0.76, 0.91] V	gnd	0V
Clock Path CTLE B	5'b0_1001	3'b010	4'b0000	nbias(bias_hisupply)	[0.30,0.54]V	gnd	0V
			4'b0001	none	n/a	none	n/a
			4'b0010	none	n/a	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Clock Path Sampler	5'b0_1010	3'b010	4'b0000	rx_isoavdd_clk	[0.75, 0.91] V	gnd	0V
		3'b010	4'b0001	rx_isoavdd	[0.75, 0.91] V	gnd	0V
		3'b010	4'b0010	isrc_clkq02	[0.34, 0.56] V	gnd	0V
		3'b010	4'b0011	isnk_clkq02	[0.3, 0.5] V	gnd	0V
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a
			4'b1111	none	n/a	none	n/a
Receive Line Loopback Multiplexor	5'b0_1110	3'b010	4'b0000	vbn	[0.262, 0.490] V	gnd	0V
		3'b010	4'b0001	vbp	[0.336, 0.624] V	gnd	0V
			4'b0010	none	n/a	none	n/a
			4'b0011	none	n/a	none	n/a
			4'b0100	none	n/a	none	n/a
			4'b0101	none	n/a	none	n/a
			4'b0110	none	n/a	none	n/a
			4'b0111	none	n/a	none	n/a
			4'b1000	none	n/a	none	n/a
			4'b1001	none	n/a	none	n/a
			4'b1010	none	n/a	none	n/a
			4'b1011	none	n/a	none	n/a
			4'b1100	none	n/a	none	n/a
			4'b1101	none	n/a	none	n/a
			4'b1110	none	n/a	none	n/a

Component	component_sel_preg	ADC mode	atbsel_preg	Signal on atb0	Value of signal on atb0 [min, max]	Signal on atb1	Value of signal on atb1 [min, max]
			4'b1111	none	n/a	none	n/a
	5'b0_1110 - 5'b1-1111		4'b0000 - 4'b1111	unused	n/a	unused	n/a
Note: Core supply voltage measurements listed assume 0.85V nominal supply. Please substitute the nominal voltage for the particular process variation used.							

7.5.5.2 ATB Digitizer

An 8-bit ADC has been included to digitize voltages or currents present on the internal In_atb0/In_atb1 buses as programmed through the ATB control register described in the previous section. The ADC operates off a clock derived from the signal detection oscillator clock which requires calibration. Thus the normal startup procedure should be followed at least through cmn_ready assertion.

The digitization process and read back is managed through the cmn_atbadc_ctrl0_preg and cmn_atbadc_ctrl1_preg COB registers. Below are COB pseudo-code sequences for basic a voltage measurement and for a basic current measurement. Additional modes, including averaging voltage measurement, averaging current measurement, test modes, and ramp test modes exist. These modes of operation are described in with details kin a standalone ADC guide document.

Note: When the ATB digitizer is not in use, the *cmn_atbadc_adcmode_preg* field should be set to 4'b000 such that the high impedance is selected, disconnecting the ADC from the In_atb0/In_atb1 buses.

7.5.5.2.1 Basic Voltage Measurement

1. Enable the ADC by writing cmn_atbadc_en_preg high.
2. Enable offset voltage measure mode by cmn_atbadc_adcmode_preg to 4'h6.
3. Write cmn_atbadc_start_preg high.
4. Enable offset correction by writing cmn_atbadc_en_off_cor_preg high.
5. Write cmn_atbadc_start_preg low and poll for cmn_atbadc_datavalid low.
6. Select Desired ATB voltage point from ATB selection table.
7. Write cmn_atbadc_start_preg high.
8. Poll for cmn_atbadc_datavalid assertion, and then read the ADC output from the cmn_atbadc_data COB field.
9. Calculate the voltage using: voltage= (cmn_atbadc_data (in Decimal)/256.0)*0.95.
10. To initiate additional measurements:
    a. ADC must be left enabled to ensure offset value is preserved.
    b. If temperature or voltage change, the offset calibration in step 3 and 4 must be repeated.
    c. Otherwise then repeat steps 5 through 9.

7.5.5.2.2 Basic Current Measurement

11. Enable the ADC by writing *cmn_atbadc_en_preg* high.
12. Enable offset voltage measure mode by *cmn_atbadc_adcmode_preg* to 4'h7.
13. Write *cmn_atbadc_start_preg* high.
14. Enable offset correction by writing cmn_atbadc_en_off_cor_preg high.
15. Write *cmn_atbadc_start_preg* low and poll for *cmn_atbadc_datavalid* low.
16. Select Desired ATB current point from ATB selection table.
17. Write *cmn_atbadc_start_preg* high.
18. Poll for cmn_atbadc_datavalid assertion, and then read the ADC output from the cmn_atbadc_data COB field.
19. Calculate the current using: current= (cmn_atbadc_data (in Decimal)/256.0)*0.6355/10.5K.
20. To initiate additional measurements:
    a. ADC must be left enabled to ensure offset value is preserved.
    b. If temperature or voltage change, the offset calibration in step 3 and 4 must be repeated.

    Otherwise then repeat steps 5 through 9.

7.5.6 Link Power States

The lanes in this macro support multiple power states. For a given Link, the master lane controls the power state for all of the lanes that make up the Link. For further information in configuring Multi-lane Links. The macro implements five power states other than the fully disabled state, delineated as A0 through A4. These are mapped to various states in supported standards as described in Table 7-5 Link Power States. A5 is currently not implemented. Standards which do not have comparable staged power states are expected to operate solely in disabled and A0.

The power state is controlled through a power state machine residing in a Link master lane and impacts all enabled, up converted lanes equivalently. The power state is directed through the master lane's synchronous xcvr_power_state_req_ln_{3:0}[5:0] and xcvr_power_state_ack_ln_{3:0}[5:0] signals on the core-side interface.

PCIe L1.1 and L1.2 substates require additional control sequencing in addition to A4 entry to ensure the published low power levels are achieved.

Table 7-5 Link Power States lists the default operation in each power state and are appropriate for all supported standards. Which circuits are operational and which are shutdown for each state is however controllable through the psc_a0/a1/a2/a3/a4_preg CDB registers.

Table 7-5 Link Power States

Link Power State	PCIe	Default Macro Operation
Disabled	Disabled	Transmitter and receiver lanes not enabled. Serial receiver is in high impedance state.
A0	PIPE P0	High speed serial operation. • PLLLN operational. • TX details • High speed serial path enabled. • Receiver detection (PCIe-only) assumed disabled. • RX details

Link Power State	PCIe	Default Macro Operation
		• High speed serial path enabled. • Termination is in low impedance state. • High-speed signal detection is enabled.
A1	PIPE P0s	Minimal low power state. • PLLLN operational. • TX details • Majority of the High speed serial path is disabled. • Electrical idle maintained. • Receiver detection (PCIe-only) assumed disabled. • RX details • PCIe and EEE: • High speed serial path enabled. • Termination in low impedance state. • High-speed signal detection is enabled. • USB: • High speed data path partially disabled, clocks remain on. • Termination in low impedance state. • High-speed signal detection is disabled.
A2	PIPE P1	Median low power state. • Master Lane PLL_{LN} enabled; Slave Lane(s) PLL_{LN}'s disabled. • TX details • High speed serial path disabled including biasing • Electrical idle maintained*. • Receiver detection enabled as needed. • RX details • High speed serial path disabled including biasing. • Termination in low impedance state. • High-speed signal detection is enabled. **Note**: When entering A2 from idle or from A4 with tx_common_mode_en_ext deasserted, the time to establish electrical idle (Tei_chrg) exceeds the time to acknowledge A2 (Tidle_2_a2 or Ta4_2_a2). Likewise when entering A0 from idle or A4 with tx_common_mode_en_ext deasserted, the time to establish electrical idle exceeds the time to acknowledge A0 (Tidle_2_a0 or Ta4_2_a0).
A3	PIPE P2	Maximum low power state • PLL_{LN} disabled. • TX details • High speed serial path disabled including biasing • Electrical idle maintained to support receiver detection as needed. • RX details • High speed serial path disabled including biasing. • Termination in low impedance state. • High-speed signal detection is disabled.
A4	L1.1 / L1.2	Enhanced median low power state • PLL_{LN} disabled. • TX details • High speed serial path disabled including biasing • Electrical idle disabled. • Receiver detection assumed disabled.

Link Power State	PCIe	Default Macro Operation
		• RX details • High speed serial path disabled including biasing. • Termination in low impedance state. • High-speed signal detection is disabled unless extended through the assertion of rx_sig_det_en_ext. (L1.1 distinguished from L1.2 through the assertion of tx_common_mode_en_ext)
A5	Not Supported	Not Applicable

Chapter 8 USB

8.1 Features

- Support 4 USB2 Host only
  - L1/L2 Support
  - High Speed, Full Speed and Low Speed Supported
  - Up to 16 periodic endpoints supported at a time
  - 16 slots supported
- Support 2 USBSSP DRD Controllers with Type-A interface
  - USB2 L1/L2 Support
  - USB3 U1/U2/U3 Support
  - Host Mode:
    - SuperSpeedPlus(Gen2x1), SuperSpeed(Gen1x1), High Speed, Full Speed and Low Speed Supported
    - Up to 16 periodic endpoints supported at a time
    - 16 slots supported
  - Device Mode:
    SuperSpeedPlus(Gen2x1), SuperSpeed(Gen1x1), High Speed, Full Speed and Low Speed Supported
    Control transfer supported by Endpoint #0
    Up to 15 IN and 15 OUT configurable/programmable endpoints
- Support 1 USBSSP DRD Controller with Type-C interface
  - USB2 L1/L2 Support
  - USB3 U1/U2/U3 Support
  - Host Mode:
    - SuperSpeedPlus(Gen2x1), SuperSpeed(Gen1x1), High Speed, Full Speed and Low Speed Supported
    - Up to 16 periodic endpoints supported at a time
    - 16 slots supported
  - Device Mode:
    - SuperSpeedPlus(Gen2x1), SuperSpeed(Gen1x1), High Speed, Full Speed and Low Speed Supported
    - Control transfer supported by Endpoint #0
    - Up to 15 IN and 15 OUT configurable/programmable endpoints
- Support 3 USBSSP Host Only Controllers with Type-C interface
    - USB2 L1/L2 Support
    - USB3 U1/U2/U3 Support
    - SuperSpeedPlus(Gen2x1), SuperSpeed(Gen1x1), High Speed, Full Speed and Low Speed Supported
    - Up to 16 periodic endpoints supported at a time
    - 16 slots supported
- Application Interface
  - AXI4 Master Interface with 128-bit data and 64-bit address for data path

- APB4 Slave interface for configuration path

8.2 Block Diagram

The following diagram illustrates the microarchitecture of the USB subsystem, highlighting its structural components and primary data paths.

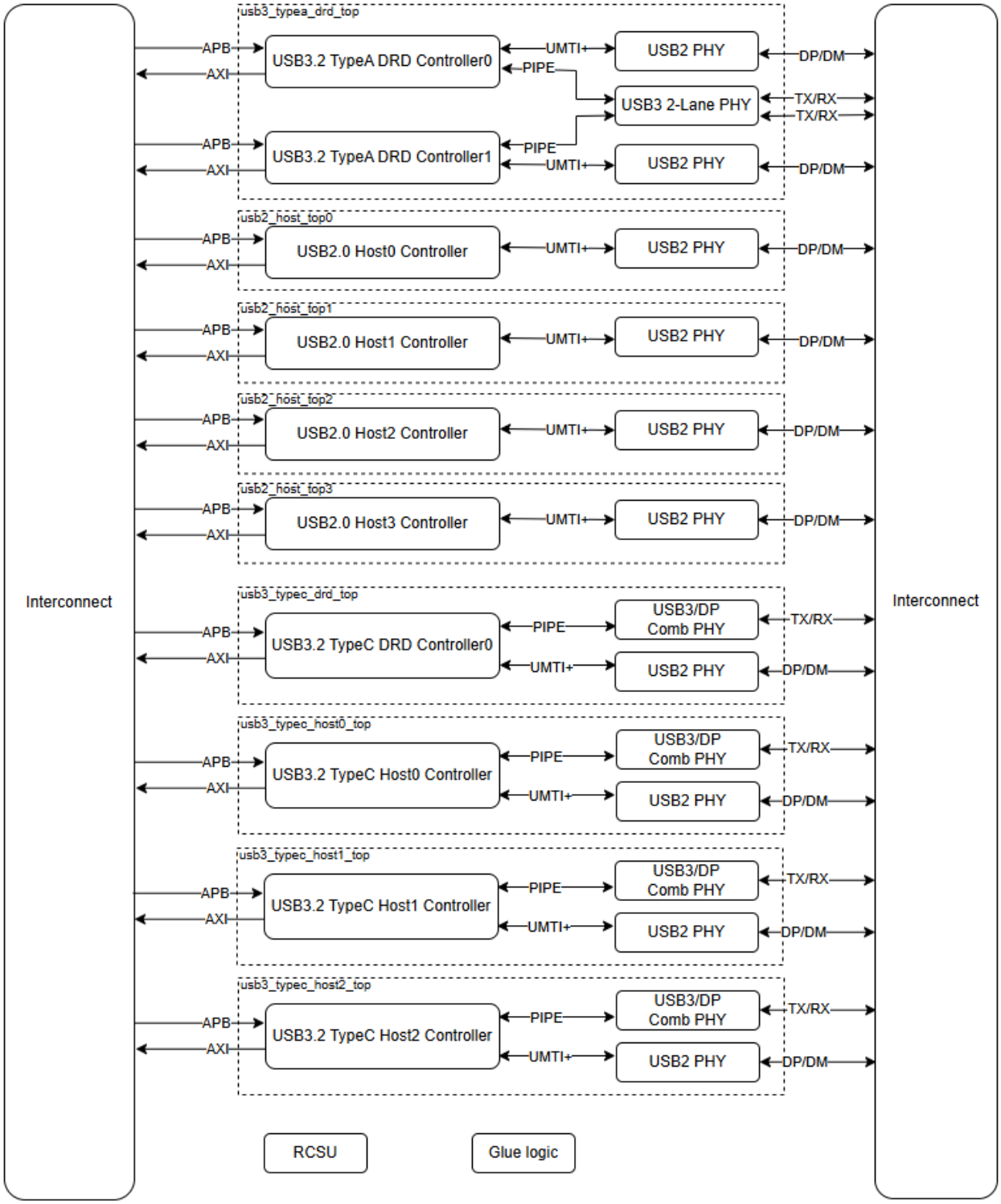


Figure 8-1 USB Subsystem Block Diagram

8.2.1 Brief Function Description

8.2.1.1 usb2_host_top0/1/2/3

USB2 Host Only integrated with one usb2 controller and a USB2PHY.

8.2.1.2 usb3_typea_drd_top

USBSSP DRD top integrated with two USBSSP controllers, two USB2PHYs and a 2-lane USB3PHY.

8.2.1.3 usb3_typec_drd_top

USBSSP DRD top integrated with a USBSSP controller, a USB2PHY and a USB3COMBPHY which is shared with DP Controller.

8.2.1.4 usb3_typec_host0/1/2_top

USBSSP Host Only integrated with one USBSSP controller, a USB2PHY and a USB3COMBPHY which is shared with DP Controller.

8.2.1.5 RCSU

Provide control registers, status registers and APB Configuration bus.

8.2.1.6 Glue Logical

Clock, reset and some other control logical for USB IP.

8.3 Address Mapping

Table 8-1 USB Address Mapping

Offset	Size	Block	Description
0x0900_0000	64KB	USB3 TypeC DRD RCSU	Address for USB3 TypeC DRD RCSU registers
0x0901_0000	64KB	USB3 TypeC DRD Controller	Address for USB3 TypeC DRD Controller registers
0x0902_0000	64KB	USB3 TypeC DRD USB2PHY	Address for USB3 TypeC DRD USB2PHY registers
0x0903_0000	256KB	USB3 TypeC DRD USB3COMBPHY	Address for USB3 TypeC DRD USB3COMBPHY registers
0x0907_0000	64KB	USB3 TypeC Host0 RCSU	Address for USB3 TypeC Host0 RCSU registers
0x0908_0000	64KB	USB3 TypeC Host0 Controller	Address for USB3 TypeC Host0 Controller registers
0x0909_0000	64KB	USB3 TypeC Host0 USB2PHY	Address for USB3 TypeC Host0 USB2PHY registers
0x090A_0000	256KB	USB3 TypeC Host0 USB3COMBPHY	Address for USB3 TypeC Host0 USB3COMBPHY registers
0x090E_0000	64KB	USB3 TypeC Host1 RCSU	Address for USB3 TypeC Host1 RCSU registers
0x090F_0000	64KB	USB3 TypeC Host1 Controller	Address for USB3 TypeC Host1 Controller registers
0x0910_0000	64KB	USB3 TypeC Host1 USB2PHY	Address for USB3 TypeC Host1 USB2PHY registers
0x0911_0000	256KB	USB3 TypeC Host1 USB3COMBPHY	Address for USB3 TypeC Host1 USB3COMBPHY registers

Offset	Size	Block	Description
0x0915_0000	64KB	USB3 TypeC Host2 RCSU	Address for USB3 TypeC Host2 RCSU registers
0x0916_0000	64KB	USB3 TypeC Host2 Controller	Address for USB3 TypeC Host2 Controller registers
0x0917_0000	64KB	USB3 TypeC Host2 USB2PHY	Address for USB3 TypeC Host2 USB2PHY registers
0x0918_0000	256KB	USB3 TypeC Host2 USB3COMBPHY	Address for USB3 TypeC Host2 USB3COMBPHY registers
0x091C_0000	64KB	USB3 TypeA DRD RCSU	Address for USB3 TypeA DRD RCSU registers
0x091D_0000	64KB	USB3 TypeA DRD0 Controller	Address for USB3 TypeA DRD0 Controller registers
0x091E_0000	64KB	USB3 TypeA DRD1 Controller	Address for USB3 TypeA DRD1 Controller registers
0x091F_0000	64KB	USB3 TypeA DRD0 USB2PHY	Address for USB3 TypeA DRD0 USB2PHY registers
0x0920_0000	64KB	USB3 TypeA DRD1 USB2PHY	Address for USB3 TypeA DRD1 USB2PHY registers
0x0921_0000	256KB	USB3 TypeA DRD USB3PHY	Address for USB3 TypeA DRD USB3PHY registers
0x0925_0000	64KB	USB2 Host0 RCSU	Address for USB2 Host0 RCSU registers
0x0926_0000	64KB	USB2 Host0 Controller	Address for USB2 Host0 Controller registers
0x0927_0000	64KB	USB2 Host0 USB2PHY	Address for USB2 Host0 USB2PHY registers
0x0928_0000	64KB	USB2 Host1 RCSU	Address for USB2 Host1 RCSU registers
0x0929_0000	64KB	USB2 Host1 Controller	Address for USB2 Host1 Controller registers
0x092A_0000	64KB	USB2 Host1 USB2PHY	Address for USB2 Host1 USB2PHY registers
0x092B_0000	64KB	USB2 Host2 RCSU	Address for USB2 Host2 RCSU registers
0x092C_0000	64KB	USB2 Host2 Controller	Address for USB2 Host2 Controller registers
0x092D_0000	64KB	USB2 Host2 USB2PHY	Address for USB2 Host2 USB2PHY registers
0x092E_0000	64KB	USB2 Host3 RCSU	Address for USB2 Host3 RCSU registers
0x092F_0000	64KB	USB2 Host3 Controller	Address for USB2 Host3 Controller registers
0x0930_0000	64KB	USB2 Host3 USB2PHY	Address for USB2 Host3 USB2PHY registers

8.4 USB3.2 Gen2 Dual-Role Controller

8.4.1 Overview

This section specifies the design of the USB 3.2 Gen2 Dual Role Controller IP core, including its optional Integrated Protocol Stack. It is structured as a fully synthesizable softcore (SoC level), surrounded by an integration level (IPS level) which includes examples of Cadence PHY and memory macros. This is subsequently referred to in this document as USBSSP.

USBSSP features dual role capability, enabling connection to external USB devices as either a Host or a Peripheral Device. These modes are mutually exclusive and can be selected via hardware strap

or software configuration. The following modes of operation are supported and compliant with the USB 2.0 and USB 3.2 specifications:

- USB 3.2 xHCI Full or Embedded Host
- USB 3.2 xHCI-derived Peripheral Device

To accommodate a variety of application needs and system constraints, USBSSP is available in multiple controller configurations. These variants share the same core architecture but differ in functionality, area, and power optimization. The Type-A DRD and Type-C DRD controllers are full-featured implementations that support both host and device roles. In contrast, the Type-C Host-only and USB2 Host-only controllers are streamlined versions derived from the full-function controller, offering reduced area and power consumption for applications where dual role support is not required.

8.4.2 Architectural Specification

USBSSP is a dual role USB controller that enables connection to external USB devices as either a Host or a Peripheral Device.

8.4.2.1 Features

The USBSSP supports the following features:

- USB Interface:
  - Compliant with USB 3.2 Specification (1)
  - Compliant with xHCI 1.1 Specification (2)
  - Single USB2.0 Port
  - Single USB3.2 Port
  - USB 3.2 PIPE interface compliant (when no PHY is integrated) (3)
  - USB 2.0 UTMI+ interface compliant (when no PHY is integrated) (4)
  - USB2 L1/L2 Support
  - USB3 U1/U2/U3 Support
- Application Interface:
  - AXI4 Master Interface with 128-bit data and 64-bit address (5)
  - APB4 Slave interface with 32-bit data and address (6)
- Dual Mode Operation:
  - HW selectable default mode selection supporting operation without any SW interaction
  - Programmable runtime mode change
- Host Mode:
  - SuperSpeedPlus (Gen2x1), SuperSpeed (Gen1x1), Hi Speed, Full Speed and Low Speed supported
  - 16 slots supported
  - Up to 16 periodic endpoints supported at a time
  - 256 Primary Streams supported
  - Root Hub functionality implemented
- Peripheral Mode:
  - SuperSpeedPlus (Gen2x1), SuperSpeed (Gen1x1), Hi Speed and Full Speed supported
  - Control transfers supported by Endpoint #0
  - Up to 15 IN and 15 OUT configurable/ programmable endpoints
  - 256 Primary Streams supported
  - Software Interface consistent with the Host Mode based on the xHCI 1.1

- Integrated Protocol Stack
  - PHY integration (where customer applicable)
  - Memory integration
  - Clock, reset, and power management integration

8.4.2.1.1 Clocks and Resets

This interface includes all the clocks and resets sources necessary to support the full USBSSP implementation including:

- AXI Clock: aclk
- APB Clock: pclk
- Low power clock: clk_lpm – always running low power clock used during low power modes of operation that is used to detect connect, disconnect, resume, etc
- Link clocks: clk_pipe (USB3.2) and clk_utmi (USB2.0) – link clocks for the mode of operation in use
- Frame Timer: clk_sof – accurate input clock necessary for frame timing
- Fundamental power up reset input: pwrup_rst_n
- Separate reset input for APB clock domain only: presetn

For more information on start-up and power on reset sequence, refer to USBSSP-DRD Integration Guide (7).

8.4.2.1.2 APB4 Slave Interface

The APB Controller interface allows for S/W programmability of USBSSP including DRD, Host and Device controller logic. It also provides control and operation status information to driver software. Note that within the IPS the PHYs will have their own independent APB interface.

8.4.2.1.3 AXI4 Master Interface

The AXI Master Interface is designed to integrate to an application specific memory controller for providing operational data to USBSSP. This interface is used to gain access to application memory. DMA is used to access data structures and related information during normal operation.

8.4.2.1.4 Interrupt Interface

This interface provides interrupt signals to external CPU for various events detected within USBSSP, including DRD, Host and Peripheral defined interrupts.
Alternatively, Legacy Interrupts can still be used.

8.4.2.1.5 Mode Select Interface

This interface allows an external HW mechanism to determine the default mode of operation, without the need for software to program the DRD logic within USBSSP. This allows an alternative protocol to perform the role negotiation or for the mode to be hardware strapped.

8.4.2.1.6 USB3.2 PHY Interface

If there is no USB3.2 PHY integrated then PIPE interface is employed to attach the USBSSP IP to an external USB3.2 PHY. For interface details, refer to the PIPE Version specification (3). In case of USB3.2 PHY integration, this interface contains USB3.2 PHY related signals.

8.4.2.1.7 USB2.0 PHY Interface

If there is no USB2.0 PHY integrated then UTMI+ interface is employed to attach the USBSSP IP to an external USB2.0 PHY. For interface details, refer to the UTMI+ specification (4). In case of USB2.0 PHY integration, this interface contains USB2.0 PHY related signals.

8.4.2.1.8 Port Power Interface

Port Power Interface provides ability to control and monitor VBUS of the USB port.

8.5 USB 2.0 PHY

8.5.1 Overview

This IP is a complete USB2.0 On-The-Go (OTG) Transceiver. It uses UTMI+ interface to connect to the USB applications namely Host, Peripherals and OTG controllers. The following figure shows the application diagram where the USB2.0 PHY interfaces with the Application controllers in either SoC or FPGA or Custom ASIC.

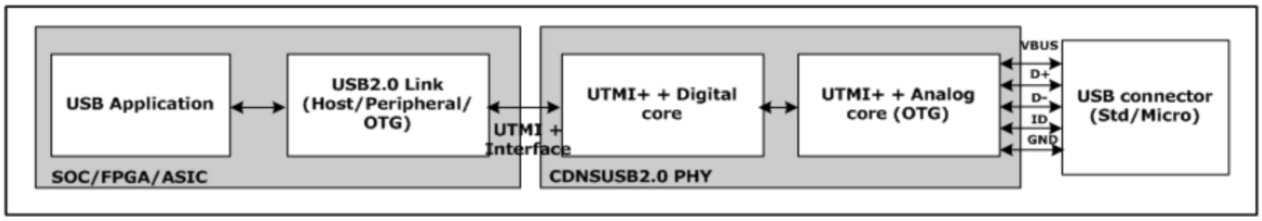


Figure 8-2 Basic USB2.0 Application Using UTMI+ PHY

This IP is fully compliant to UTMI+ Level 3 Specification and it supports High-Speed (480Mbps), Full-Speed (12Mbps) and Low-Speed (1.5Mbps) data rates. It supports both parallel interface (8-bit 60MHz/16-bit 30MHz) and serial interface (fslsserialmode) for data transfers. It uses uni-directional for data transfers. The PHY supports USB low power modes (Suspend/LPM) where the PLL can be shut down to consume only the suspend current as per the USB2.0 Specification. The PHY supports all the USB2.0 Test Modes required for compliance certification.

This IP is fully OTG compliant and implements all the OTG comparators required for supporting SRP and HNP. OTG protocol allows the USB applications (Host and Peripherals) to dynamically switch roles between them without manually changing the cables. This IP shall be dynamically configured to be either Host or Peripheral Transceiver to satisfy the OTG functionality when integrated with OTG devices. For example, a digital camera may connect to a personal computer (PC) as a peripheral for downloading the photos or connect to a printer as a host for printing the photos from the camera memory. Also OTG enabled device shall connect to another OTG enabled to exchange information between them. Consider another example, where a mobile phone connects to another mobile phone in which the Host and peripheral roles shall be interchanged multiple times between them via HNP. All the OTG features are supported by this IP using a single Micro-AB connector.

8.5.2 Features

- Fully Compliant with UTMI+ Level 3 Specification revision 1.0
  - Supports High-Speed (480Mbps), Full-Speed (12Mbps) and Low-Speed (1.5Mbps) data rates
  - Supports Host, Peripherals and OTG (Dual Role Device) Applications
  - Supports 8-bit 60MHz and 16-bit 30MHz parallel interface
  - Supports FSLSSERIAL Mode interface for re-using legacy FS/LS Host Controller IPs
  - Supports generation of long EOP for HS SOFs
  - Supports generation of LS keep-alive strobes
  - Supports FS Preamble for FS Hubs that has LS Device attached
- Supports Battery Charging BC1.2v Specification
- Supports D+/D- Lane reversal for flexible board integration
- Supports USB low power states namely Suspend and Link Power Management (LPM)
- Supports internal comparators for monitoring OTG voltage thresholds

- Supports multiple Reference Clocks for PLL - 9.6, 10, 12, 19.2, 20, 24, 25, 26, 38.4, 40, 48, 50, 52 MHz
- Supports internal PLL for High-Speed (480MHz) Clock and Data Recovery (CDR) operation
- Integrated Termination Resistors (45Ω, 1.5KΩ and 15KΩ)
- Supports Built in Self-Test (BIST) for production testing
- Supports Loopback via UTMI+ Interface
- Supports AMBA3 APB v1.0 interface
- Supports PLL standalone feature
- Supports PLL Bypass feature
- Supports JTAG interface
- Supports Scan At Speed Testing
- 3.3V ESD support on VBUS phy pin

8.5.3 Architecture Overview

This IP is a USB2.0 OTG PHY fully compliant with UTMI+ Level 3 Specification. It includes the following blocks for its operations.

- High-Speed Drivers and Receivers
- Full-Speed/Low-Speed Drivers and Receivers
- VBUS Comparators for OTG operations
- PLL to generate 480MHz clock for HS operations
- High-Speed Clock and Data Recovery
  - Elasticity Buffer to accommodate the ppm differences in HS mode
- Full-Speed/Low-Speed Data Recovery
- SYNC/EOP generator and detector
- NRZI encoder and decoder
- Bit stuffer and unstuffer
- Transmit/Receive Serializer and De-serializer
- Transmit/Receive control state machines
- Interface controller for handling the UTMI + control signal interface
  - Transceiver selection (HS or FS/LS)
  - Termination selection (HS or FS/LS)
  - Operational modes selection
  - Host/Peripheral PHY selection (PullDown Resistors)
  - PLL reference clock source (crystal/external) and reference frequency selection
  - SIE clock generation (60/30 MHz)
  - Data transfer modes selection (8-bit/16-bit uni-directional or FSLSSERIAL Mode)
  - Transmit/Receive data channel
  - Disconnect Detector
  - Test Mode programmability (Fine Tuning)

8.5.3.1 Functional Block Diagram

The following figure shows the functional block diagram of the USB2.0 PHY.

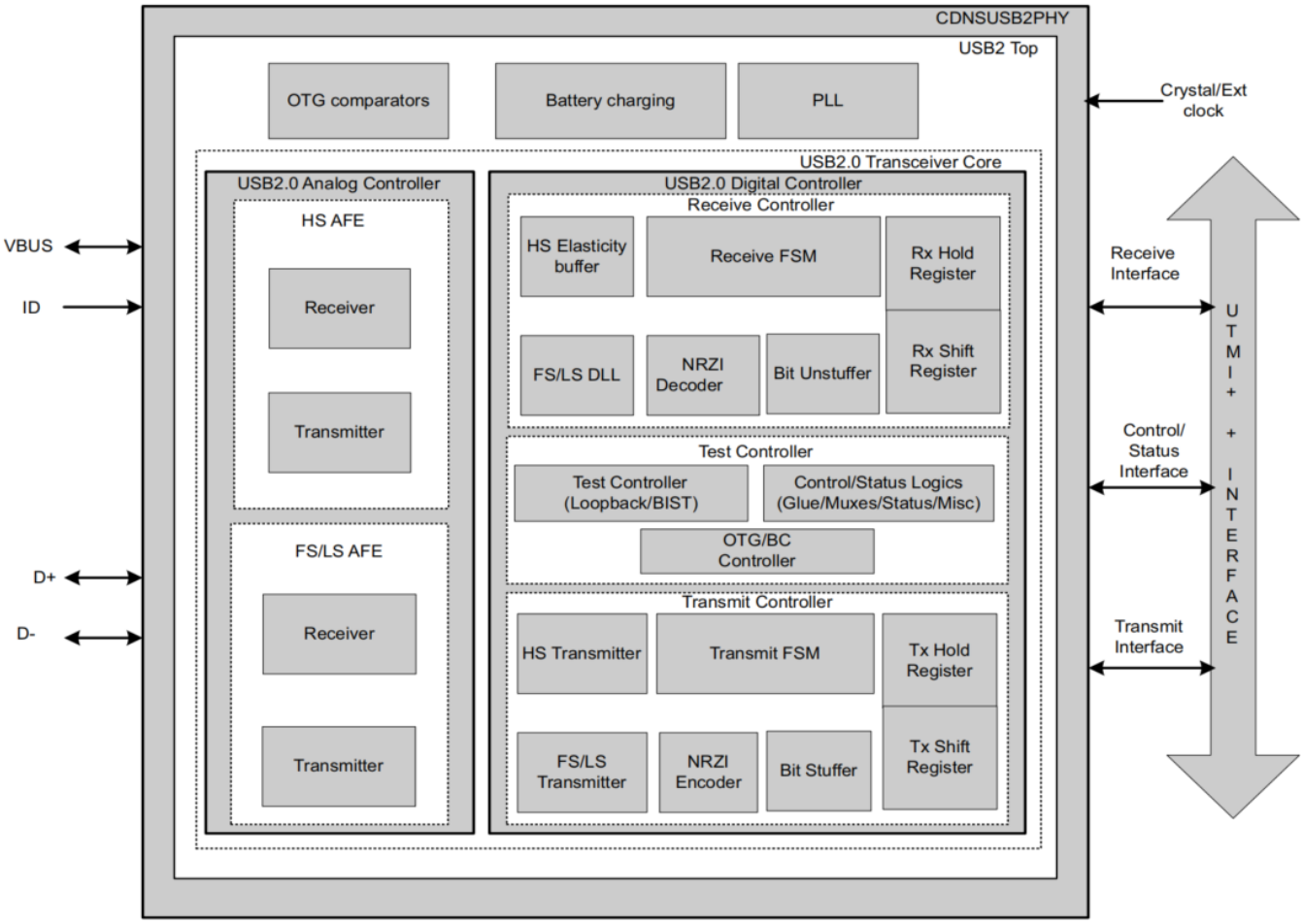


Figure 8-3 Functional Block Diagram

8.6 USB3 Type-C PHY

The USB3 Type-C PHY is a multi-protocol PHY. The PCS layer complies with the PIPE 4.4.1 specification and supports dynamic equalization features required by different protocols. Figure 8-4 shows the architecture and main components of the USB Type-C PHY.

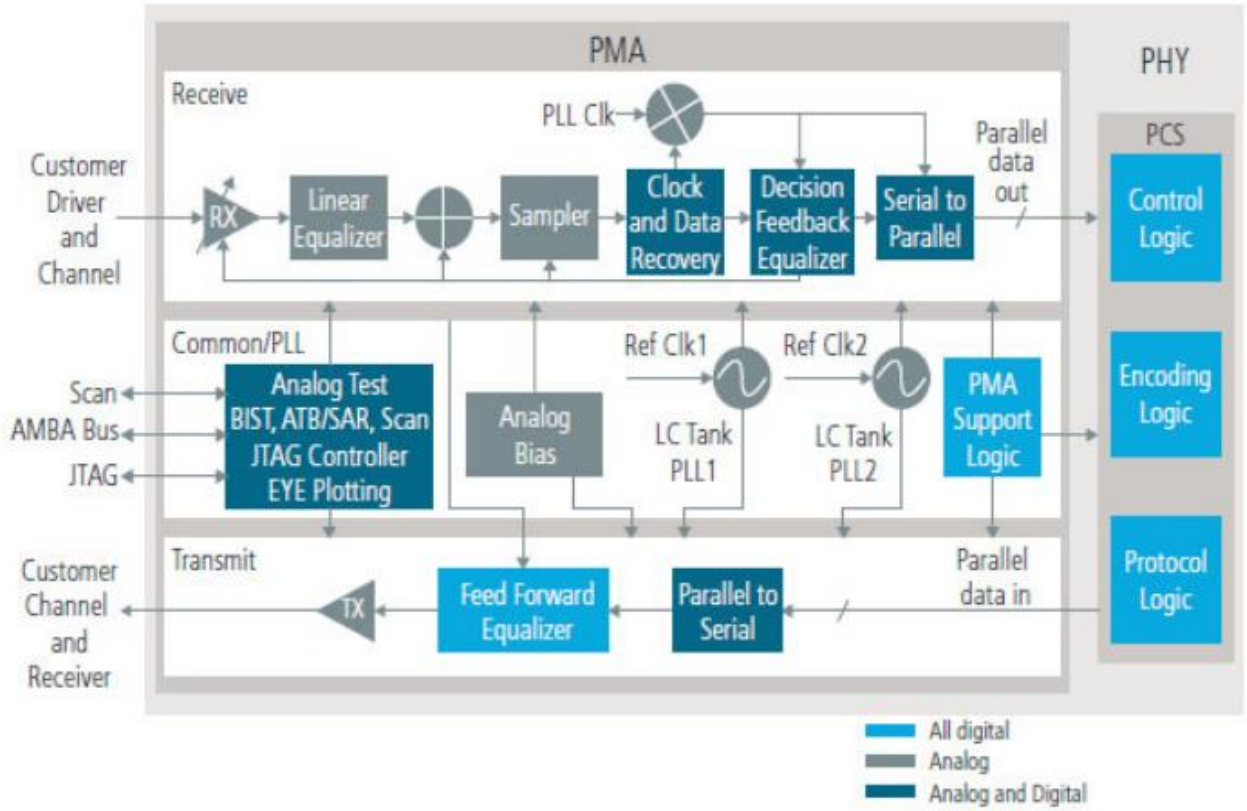


Figure 8-4 USB Type C PHY

8.6.1 PHY Architecture

The USB3 Type-C PHY adopts a lane-based architecture, enabling flexible floorplanning, placement, packaging, and I/O integration.

This architecture provides improved design scalability while maintaining the reliability and ease of integration associated with hard macro implementations.

8.6.2 Top-Level Architecture

The USB3 Type-C PHY consists of a Physical Media Attachment (PMA) layer and a Physical Coding Sublayer (PCS).

The following sections describe the PHY-level and PMA-level block diagrams, along with the key functional building blocks of each layer.

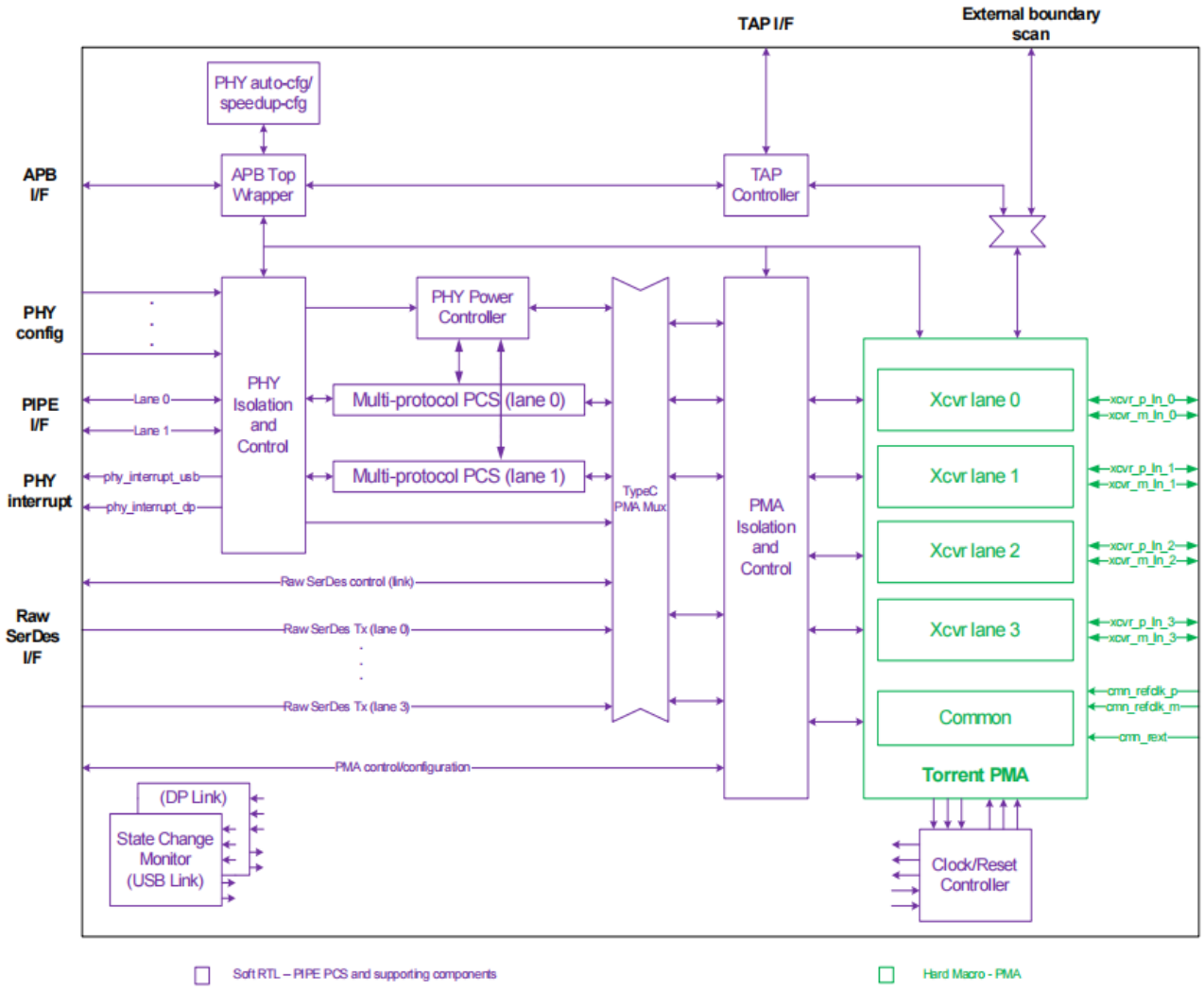


Figure 8-5 USB3 Type-C PHY Block Diagram

8.6.3 PHY Pin List

8.6.3.1 Power Pin List

Table 8-2 USB3 Type-C PHY External Pin / Bump List

Dir.	Signal Name	Type	External Pin / Bump or Internal Core Pin.	Description
input	gnd	power	bump	Macro ground.
input	cmn_avdd	power	bump	PMA common analog non high speed clock power
input	cmn_avdd_clk	power	bump	PMA common analog high speed clock power
input	cmn_avdd_h	power	bump	PMA common analog high voltage power
input	xcvr_avdd_ln_{n:0}	power	bump	PMA transceiver analog and non high speed clock power

Dir.	Signal Name	Type	External Pin / Bump or Internal Core Pin.	Description
input	xcvr_avdd_h_ln_{n:0}	power	bump	PMA transceiver high voltage power
input	xcvr_avdd_clk_ln_{n:0}	power	bump	PMA transceiver analog high speed clock power

8.6.3.2 External Pin / Bump List

Table 8-3 USB3 Type-C PHY External Pin / Bump List

Dir.	Signal Name	Type	Timing	Description
inout	cmn_refclk_p cmn_refclk_m	analog	n/a	PMA external reference clock differential input/output - configurable as reference clock input or as reference clock output. **Note: When configured as an output, a reference clock signal shall not be driven into these pins.**
inout	xcvr_p_ln_{0/1/2/3} xcvr_m_ln_{0/1/2/3}	Diff	n/a	Per lane PMA serial data. Lane0/Lane3 are Tx only (either USB3 Tx or DP Tx) and Lane1/Lane2 are bi-directional (either USB3 Rx or DP Tx).
inout	cmn_rext	analog	n/a	PMA external calibration resistor

8.7 USB3 PHY

The USB3 PHY is a multi-protocol PHY. The PCS complies with the PIPE 4.4.1 standard and supports dynamic equalization features across different protocols.
Figure 8-6 illustrates the architecture and major components of the multi-protocol PHY.

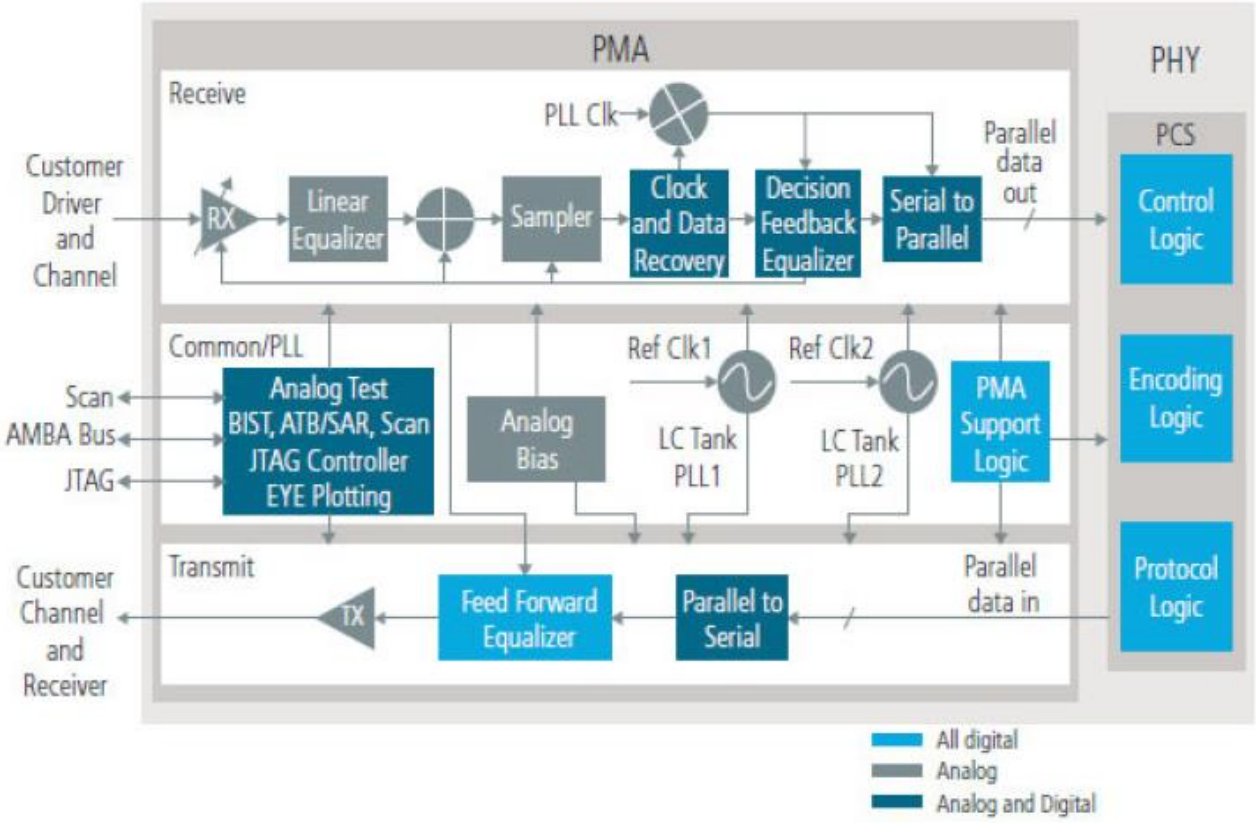


Figure 8-6 USB3 PHY

8.7.1 PHY Architecture

The USB3 PHY is designed with a lane-based architecture, providing greater control over floor-planning, placement, packaging, and I/O integration than other hard PHY solutions, while maintaining reliability and ease of use associated with GDSII macros.

8.7.2 Top Level Architecture

The USB3 PHY consists of a Physical Media Attachment (PMA) layer and a Physical Coding Sublayer (PCS). What follows is PHY level and PMA level block diagram with comprehensive description of their underlying building blocks.

8.7.3 PHY Pin List

The following tables describe the pin list of the USB3 PHY. Note that several of these pins connect directly to the Torrent PMA. In these cases, this will be noted in the description that further description is provided in the PMA specification.

8.7.3.1 Power Pin List

Table 8-4 USB3 PHY Power Pin List

Dir.	Signal Name	Type	External pin / bump or Internal core pin.	Description
input	gnd	power	bump	Macro ground.

Dir.	Signal Name	Type	External pin / bump or Internal core pin.	Description
input	cmn_avdd	power	bump	PMA common analog non high speed clock power
input	cmn_avdd_clk	power	bump	PMA common analog high speed clock power
input	cmn_avdd_h	power	bump	PMA common analog high voltage power
input	xcvr_avdd_in_{n:0}	power	bump	PMA transceiver analog and non high speed clock power
input	xcvr_avdd_clk_in_{n:0}	power	bump	PMA transceiver analog high speed clock power
input	xcvr_avdd_h_in_{n:0}	power	bump	PMA transceiver high voltage power
input	vdd	power	internal	SoC core power to PMA for internal level shifters

8.7.3.2 External Pin / Bump List

Table 8-5 USB3 PHY External Pin / Bump List

Dir.	Signal Name	Type	Timing	Description
inout	cmn_refclk_p cmn_refclk_m	analog	n/a	PMA external reference clock differential input/output - configurable as reference clock input or as reference clock output. Note: When configured as an output, a reference clock signal shall not be driven into these pins.
output	tx_p_in_{n:0} tx_m_in_{n:0}	Diff	n/a	PMA transmitter serial data
input	rx_p_in_{n:0} rx_m_in_{n:0}	Diff	n/a	PMA receiver serial data
inout	cmn_rext	analog	n/a	PMA external calibration resistor

Chapter 9 FCH

9.1 Overview

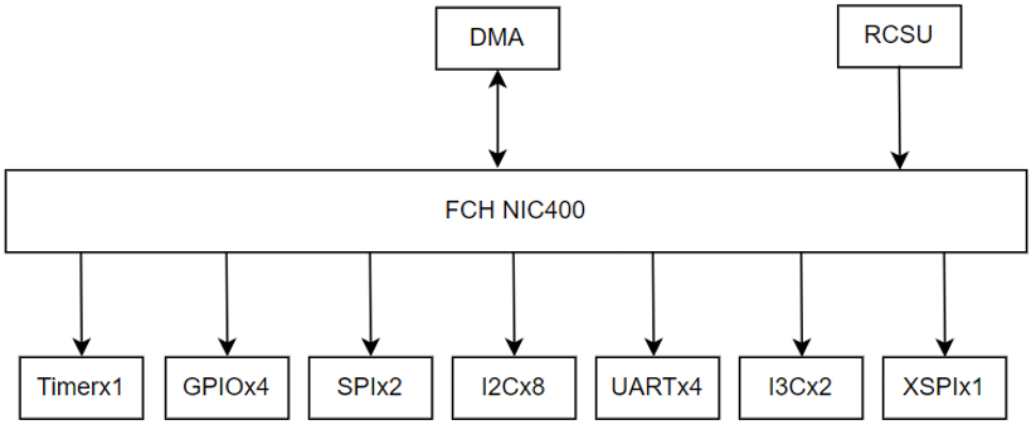
The FCH sub-system includes GPIO, SPI, I²C, UART, I3C, Timer, XSPI and DMA.

9.2 Key Features

- DMA Controller (DMAC) Support
  Provides system-level DMA capability for efficient data transfers between peripherals and DDR memory.
- Multi-Protocol Peripheral Support
  Standard-compliant IPs including:
  - 8x I2C interfaces
  - 2x SPI interfaces
  - 4x UART interfaces
  - 4x GPIO banks
  - 2x I3C interfaces
  - 1x XSPI interface
  - 1xTimer
- Programmable Timer Module
  Dual-function timer supporting:
  - Input capture mode for GPIO pulse measurement (records timestamp counters and generates system interrupts)
  - PWM signal generation
- AXI System Interconnect
  - AXI input interface from SysSMN
  - AXI output interface to SysHub
- Power Domain Architecture
  Entire FCH subsystem operates within the S0 power domain.

9.3 Block Diagram

The following diagram illustrates the FCH's structural architecture and key data flow paths.



RCSU	Register Control Security Unit) is responsible for secure register access control.
DMA Controller	Enables high-speed data transfer between peripherals and DDR
Fabric	CoreLink NIC-400 network interconnect
Timer	Internal counter for FCH operations
GPIO (×4)	General-purpose I/O for external device communication
SPI (×2, Master Mode)	Interfaces with platform sensors (e.g., TPM, fingerprint reader, touch panel)
I2C (×8, Master Mode)	Interfaces with platform sensors
UART (×4)	Debug/logging interface
I3C (×2, Master Mode)	Interfaces with platform sensors
XSPI (Master Mode)	Interfaces with platform sensors

Figure 9-1 FCH Block Diagram

9.4 I2C

9.4.1 Architecture Specification

9.4.1.1 Features

- AMBA APB protocol, version 2.0
- I²C bus specification version 3.0 (only 100kHz, 400kHz and 1MHz modes)
- Programmable for normal (100 kHz), fast (400kHz) and fast plus (1MHz) bus data rates
- Programmable as either a master or slave interface
- Programmable to use normal or extended addressing
- Capable of clock synchronization and bus arbitration
- Fully programmable slave response address
- Optional reversible FIFO with parameterizable depth (same register array for receive and transmit)
- Slave monitor mode when set up as master
- Supports I²C bus hold for slow host service
- Supports combined format transfers both as master and slave
- Slave time out detection with programmable period
- Transfer status interrupts and flags

- System clock over 200 MHz
- Digital Glitch Filter

9.5 I3C

9.5.1 Functional Description

The I3C-Master module is structured into two main modules Host Interface module and I3C I/F module.

All I3C-Master operations on I3C bus are managed by host processor responsible for:

- I3C-Master controller configuration;
- bus initialization;
- management of slaves through Common Command Code;
- data transfers;
- slave requests handling;
- handling mastership handoff and operation in slave mode.

9.6 XSPI

9.6.1 Introduction

The xSPI Flash Controller IP is designed for a wide range of applications and markets, including but not limited to consumer, entertainment, IoT (where storage data is critical for the proper configuration and state of all connected devices), automotive (eg. for driving assistance, such as sign/pedestrian pattern storage, recognition engines and boot-up systems), general communication systems (to mediate in storing critical interconnection configuration), and Mobile applications for which low power and high performance data access and booting is a critical milestone of the entire system requirements.

The xSPI Flash Controller is the first product supporting the JESD216 specification. The most significant change in these specs is the increase of the bandwidth with more data lanes (up to 8) and the introduction of two separate command profiles that provide a consistent standard for device vendors and controller vendors as never before.

The xSPI Flash Controller IP is designed to communicate with Flash Memory Devices supporting JESD216 and JESD251 standards. The architecture of this IP allows the system integration engineers to effectively integrate this product in their full system because of support for mainstream market system interfaces, such as AXI.

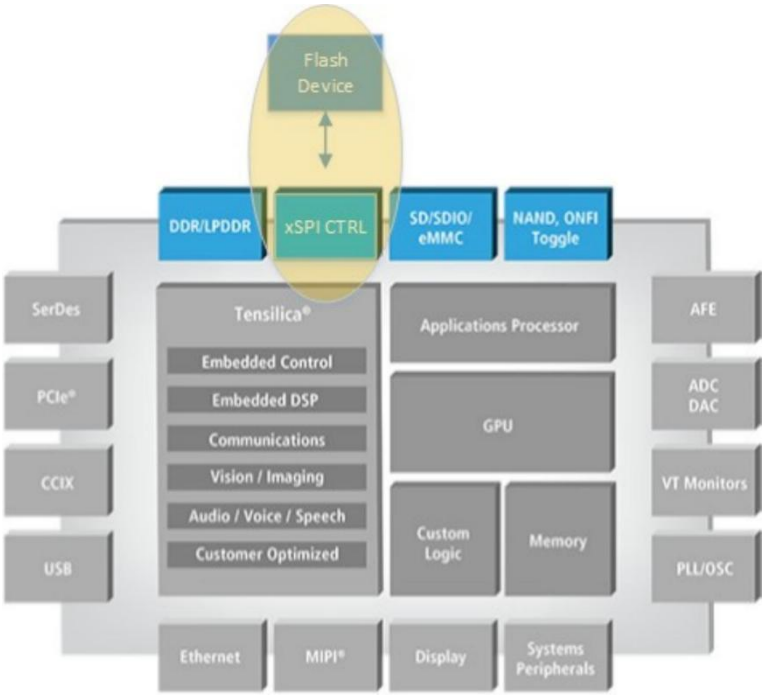


Figure 9-2 IP6522 XSPI Flash Controller with IP6182/IP6184 PHY Block Diagram

The xSPI Flash Controller IP is a highly configurable design solution, able to suit a wide range of applications and use cases while keeping integration efforts low. The core architecture is developed by an experienced team of engineers who have developed many other storage controllers and have come up with a cutting-edge design solution. Somemarket segments in which xSPI Flash Controller IP can be used are listed below:

- IoT
- Automotive
- Entertainment
- Communication
- Mobile

9.6.2 Functional Description

9.6.2.1 Choosing the Work Mode

The user can only select one active work mode at a time and must choose from ACMD, DIRECT or STIG mode to communicate with the attached device. The ACMD or DIRECT are the main modes of operation with STIG used as an accompanying mode to send commands or sequences of commands to the device that may not be directly supported in ACMD or DIRECT mode directly. The STIG mode is a highly flexible mode that can also be used to write or read data to the device and aslo perform any other operation. However, STIG mode is more reliant on host software to construct the necessary command structures. DIRECT mode, ACMD mode (when configured with the slave interface – refer to Selecting which AXI Interface to use in ACMD Mode and STIG mode all require the use of the AXI slave interface. ACMD mode, when used in its most self-reliant form, requires the use of the AXI master interface. If the user does not require to use ACMD mode with the AXI master interface, the AXI master inputs ports can be tied low and the outputs left floating.
ACMD mode requires that software either manages a descriptor table with high level sequences defined within or manages some command registers within the controller to issue those same high-level sequences. The controller can automatically translate high level sequences (RESET, ERASE, ERASE_ALL, PROGRAM, READ) into subsequence's that are tied to the device type.

Note: This mode is only available with the IP6522/IP6522A controller variants.

DIRECT mode, as the name suggests, directly translates reads and writes detected on the AXI slave interface to equivalent low level read and write sub sequences at the device. As such, DIRECT mode can only support READ or PROGRAM operations and is therefore more reliant on STIG mode to perform other operations such as ERASE. It can be used to directly boot from the device if required.

There is no mode that has better overall performance than another. If the choice of which mode to use is not clear, it is recommended to read the rest of this document taking note of any required feature that may only be available in a particular work mode (for example RWDS/DQSM write byte masking, something supported in some HyperRAM or pSRAM devices is currently only available in the DIRECT mode of operation).

9.6.2.1.1 What is XIP Mode

There is often confusion over the term XIP. A commonly used usage mode for this controller is to have the host exe- cute code directly from the attached memory device. This flow is sometimes referred to as **eXecute-In-Place**, where the host has direct access from/to the attached memory device. The controller supports this using the DIRECT work mode; that is, if the desire is to execute code directly, then you enable DIRECT work mode.

There is another usage of the term XIP. Many Flash devices can operate in what their datasheets call “XIP mode” (sometimes also referred to as continuous mode). XIP mode is used by the device to improve read performance by removing the need for the controller to send the read opcode for any read operation. If the read opcode is typically sent over 8 device clock cycles (SDR, serial transmission of the opcode), the read operation latency would be reduced by 8 cycles for that operation. This can significantly improve performance when executing code directly from the device. On the downside, when XIP mode is entered, the device can only perform READ operations. If a different operation is required, XIP mode must be exited. The controller supports XIP mode, but there are steps the host must take to trigger entry to and exit from XIP mode. These steps are different depending on the selected work mode and explained in later sections of this document.

9.7 Timer

9.7.1 Feature Summary

- Supports 8 timers each of 32 bits or 4 timers each of 64 bits. Every two 32-bit timers can be cascaded to form one 64-bit timer.
- Supports capturing one 64 bits free-running counter.
- Supports 8 PWMs each of 32 bits.
- Supports up/down counting modes.
- Supports to capture a value on trigger event.
- Support to count outside pulse edges (rising/falling/both)
- Automatically reload value support.
- Supports to freeze count value.
- Supports enabling and disabling of interrupts.
- Scan test ready.
- Simple APB interface allows easy connection to microprocessor/microcontroller devices

9.7.2 Simplified Block Diagram

The following diagram shows the structure of the Timer and the main data paths.

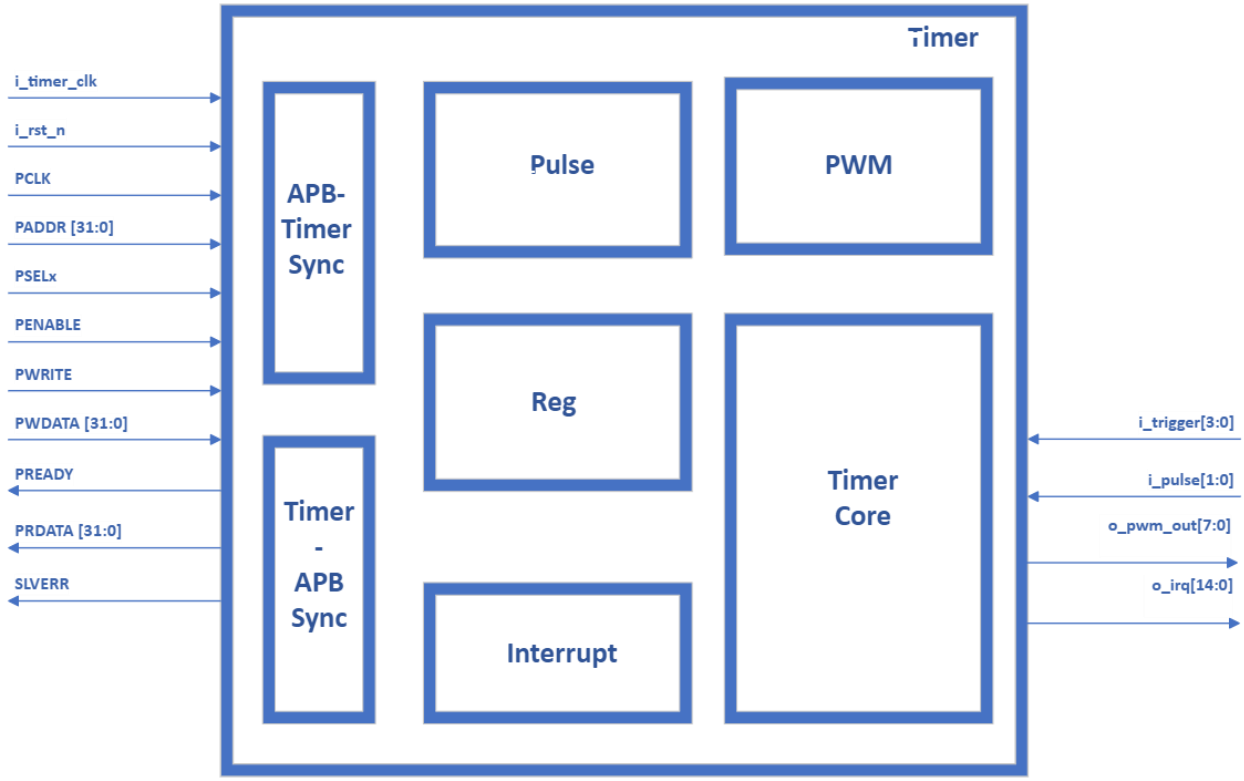


Figure 9-3 Timer and Main Data Paths

9.8 DMA

9.8.1 Overview

The CoreLink DMA-350 is a Direct Memory Access Controller (DMAC) with AMBA® AXI5 interfaces that provides fast memory to memory, peripheral to memory, memory to peripheral copy, and peripheral to peripheral capabilities on multiple channels. The processor in the system can control the DMA channel behavior over an APB4 interface and set security-related settings when Security Extension is supported. The DMAC has configurations for different types of copy, scatter-gather, increment, or two-dimensional image copy operations. It also supports trigger inputs and outputs for flow control and command sequencing capabilities. The DMA-350 adds support for low-power integration through the LPI interfaces for both clock and power.

The DMA-350 can have several channels which can have different properties, like FIFO size, to suit different requirements. Several DMA commands can be combined with the command linking feature. Using the command linking feature, a complex transfer can be set up and the DMAC can perform without the need for interaction with the processor. Each DMA channel can have a stream interface (AXI4-Stream) connected to an external Data Processing Unit. These can perform required data processing tasks on the fetched data before the DMAC writes the data to the destination.

The following figure shows DMA-350 in a system:

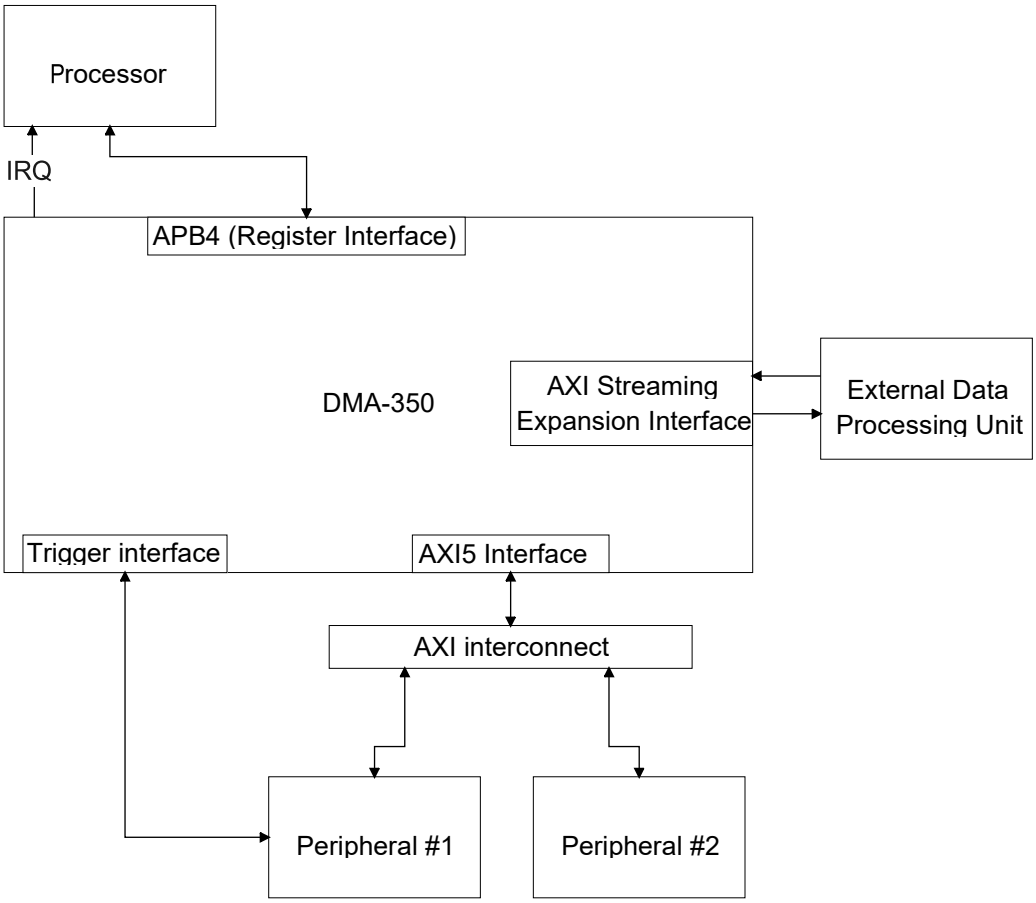


Figure 9-4 DMA-350 in a System

9.8.1.1 Component Overview

The following block diagram shows the top-level components of DMA-350:

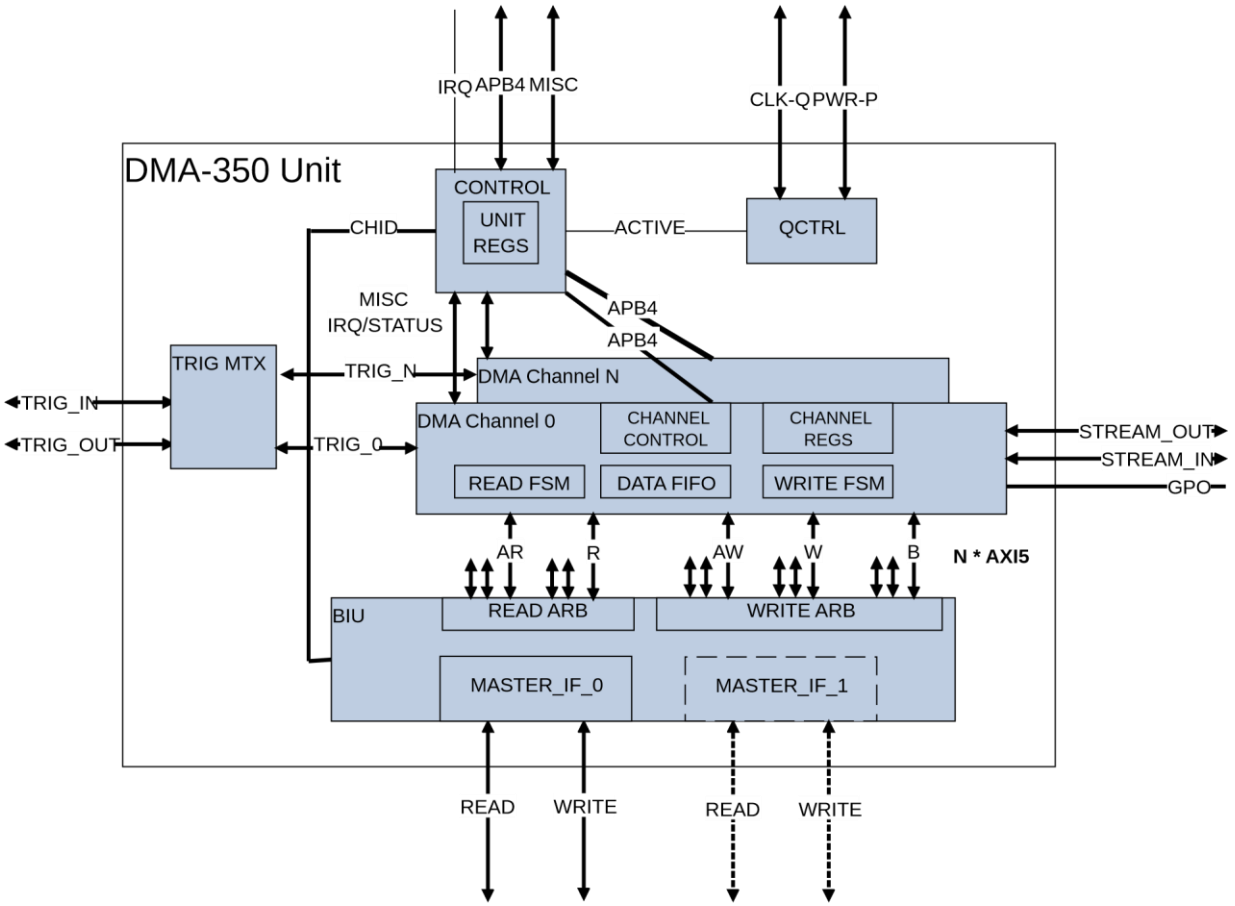


Figure 9-5 DMA-350 Top-Level Block Diagram

DMA-350 has the following main blocks:

9.8.1.1.1 Control

Control Block

- Receives and terminates APB bus from the processor
- Forwards register accesses to each channel
- Implements the common control functions of the DMAC
- Handles security and privilege access right settings of the channels
- Generates interrupts for high-level DMAC events

9.8.1.1.2 DMA Channel

DMA Channel block

- Implements one functional unit of the DMA controller that is capable of generating memory transfers
- Generates AXI transfers that match the settings of the current command
- Terminates the Trigger Interface from peripherals
- Generates interrupts and trigger out indications for internal channel events
- Implements an internal FIFO to separate read and write sides of the channel
- Implements support for an External Data Processing Unit over the AXI-4 stream interface
- Several DMA channels may exist in the DMA-350 determined by a configuration parameter

9.8.1.1.3 BIU

Bus Interface Unit

- Responsible for routing and arbitrating traffic between the DMA channels and either a single or multiple downstream AXI5 manager interfaces

9.8.1.1.4 TRIG MTX

Trigger matrix block

- Implements a selectable connection between the trigger input and output ports of the DMA channels and the top-level trigger input and output ports
- Allows internal triggering between the channels

9.8.1.1.5 QCTRL

Power and Clock management

- Implements the clock and power management features of DMA-350
- Terminates the LPI interfaces and allows quiescence based on the internal activity

9.8.1.2 Compliance

The DMA-350 interfaces are compliant with the following Arm specifications and protocols:

- AMBA AXI5 protocol, see *AMBA® AXI and ACE Protocol Specification*
- AMBA APB protocol, see *AMBA® APB Protocol Specification*
- AMBA AXI4-Stream protocol, see *AMBA® 4 AXI4-Stream Protocol Specification*
- AMBA Low Power Interface protocol, see *AMBA® Low Power Interface Specification*

9.8.1.3 Key Features

The DMA-350 supports the following key features:

- Programmable APB4 configuration registers
- AXI5 data path
- *Low-power interface* (LPI) for clock and power
- 1-8 parallel DMA channels each with a configurable FIFO
- 1D memory copy including increments
- Interrupt capability for each channel and global events
- Flexible command linking capability
- Triggering capabilities for flow control
- Extended memory copy capabilities: 2D/wrap/template
- Expansion interface

9.8.1.4 Configurable Options

The DMA-350 can be configured with the following options to meet specific design requirements:

- AXI5 address width 32-bit or 64-bit
- AXI5 data width of 32-bit, 64-bit, or 128-bit
- Channel identification register width between 0-bit and 16-bit
- General Purpose Output (GPO) width of the channels between 0-bit and 32-bit
- GPO support of a channel
- Stream support of a channel
- FIFO depth of every channel separately to 1,2,4,8
- Extended feature support (2D, wrapping, template) of a channel
- Number of DMA channels between 1 and 8
- Number of trigger input ports between 0 and 32
- Number of trigger output ports between 0 and 32
- Trigger input port synchronization
- Trigger output port synchronization
- Presence of secondary AXI5 manager port
- Presence of Security Extension for TrustZone® support

9.9 UART

9.9.1 About the 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 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).

The features of the UART are described in the following sections:

- Features
- Programmable parameters
- Variations from the 16C650 UART

Note: Because of changes in the programmers model, the UART (PL011) is not backwards compatible with the previous PrimeCell UART (PL010).

9.9.1.1 Features

The UART provides:

- 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.

9.9.1.2 Programmable Parameters

The following key parameters are programmable:

- communication baud rate, integer, and fractional parts
- number of data bits
- number of stop bits
- parity mode
- FIFO enable (32 deep) or disable (1 deep)
- FIFO trigger levels selectable between 1/8, 1/4, 1/2, 3/4, and 7/8
- internal nominal 1.8432MHz clock frequency (1.42-2.12MHz) to generate low-power IrDA mode shorter bit duration
- hardware flow control.

Additional test registers and modes are implemented for integration testing.

9.9.1.3 Variations from the 16C650 UART

The UART varies from the industry-standard 16C650 UART device as follows:

- receive FIFO trigger levels are 1/8, 1/4, 1/2, 3/4, and 7/8
- transmit FIFO trigger levels are 1/8, 1/4, 1/2, 3/4, and 7/8
- the internal register map address space, and the bit function of each register differ
- the deltas of the modem status signals are not available.

The following 16C650 UART features are not supported:

- stop bits (1 or 2 stop bits only are supported)
- independent receive clock.

9.9.2 Product Revisions

This section describes the differences in functionality between product revisions of the UART:

r1p0-r1p1	Contains the following differences in functionality: The Revision field in the *UARTPeriphID2 Register* bits [7:4] now reads back as 0x1.
r1p1-r1p3	Contains the following differences in functionality: The Revision field in the *UARTPeriphID2 Register* bits [7:4] now reads back as 0x2.
r1p3-r1p4	Contains the following differences in functionality: The Revision field in the *UARTPeriphID2 Register* bits [7:4] now reads back as 0x2.
r1p4-r1p5	Contains the following differences in functionality: • The receive and transmit FIFOs are increased to a depth of 32. • The Revision field in the *UARTPeriphID2 Register* bits [7:4] now reads back as 0x3.

Note: See the engineering errata that accompanies the product deliverables, for information about any functional differences that a later revision provides.

Chapter 10 DDR

10.1 Function Description

This section outlines the design implementation of the DDR subsystem.

10.2 Block Diagram

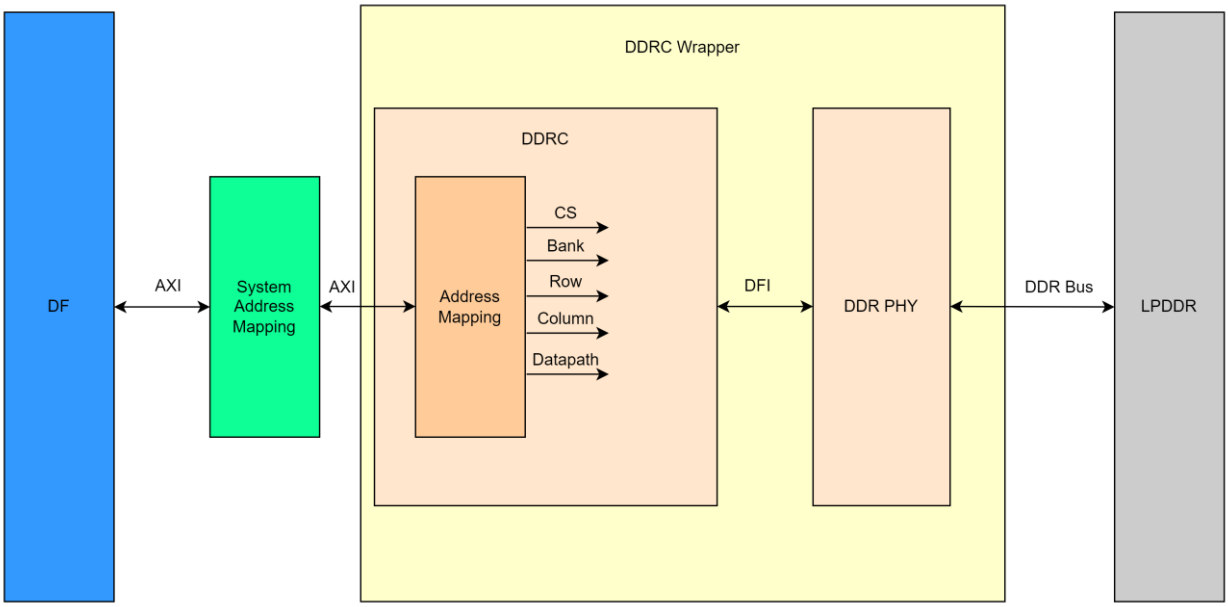


Figure 10-1 DDR Block Diagram

10.3 Features

The DDR controller supports the following features:

- Support LPDDR4X and LPDDR5
- Support commands from command truth table
- Dynamic Frequency Scaling (DFS)
- Built-in Self Test (BIST)
- Different Low power states
- Data retention
- PHY independent boot and training
- DFI interface to PHY
- Data Swap
- Addressing remap

LPDDR5 specific features:

- Bank organization

Support for two bank architectures (BG,16B mode), the maximum data rate depends on the bank architecture

Table 10-1 Bank Organization

Mode	Architecture	Data rate	Burst Length
BG	4 banks, 4 bank groups	>3200Mbps	16
16B	16 banks, no bank group	<=3200Mbps	16

- WCK Clocking
  - LPDDR5 SDRAM needs two clocks for command/address(CK, single data rate) and DQ(WCK, double data rate)
  - CAS with WCK sync command is necessary before READ, WRITE, MRR if DRAM is not synchronized with WCK

Table 10-2 Supported CK:WCK Ratio

CK/WCK	Bank organization	Data Rate	WCK Frequency	CK frequency
1:4	16B mode	<= 3200Mbps	<=1600 MHz	<= 400 MHz
	BG mode	>3200Mbps	>1600 MHz	> 400 MHz

Chapter 11 DP

11.1 Supported Feature List

- Full link rate and lane count support
  - o 1, 2, or 4 lanes
  - o Main link rates from 1.62Gbps up to 8.1Gbps
  - o eDP rates at any multiple of 0.270Gbps up to 8.1Gbps.
- DisplayPort 1.4 functionality
  - o Secondary channel support
  - o SST or MST with up to 2 sources
  - o Deep color to 48 bits per pixel
- Embedded DisplayPort 1.4a
  - o Alternate scrambler reset
  - o Fast link training
  - o Panel self-refresh
  - o Advanced Link Power Management
  - o Forward Error Correction
- Multi-Stream Transport
  - o Up to 2 simultaneous input streams
  - o Full video mode support on all 2 inputs.
  - o Full secondary channel for each virtual source supporting audio.
- Main link – 1, 2, or 4 lanes
  - o Component bit depth 6,8,10,12, and 16 bits per color plane
  - o RGB, YCbCr444/422, Y-Only
- Aux Channel o Host initiated transactions
  - o Hardware request and reply engines
- Content protection interface
  - o External key storage
  - o HDCP 1.3/1.4
  - o HDCP 2.2/2.3
  - o MST and SST modes
- Audio support o I2S
  - o Buffered transmit interface
  - o Automated secondary packet formatting and transmission
- Hot Plug Detection
  - o Connect/Disconnect event detection
  - o Interrupt detection

Chapter 12 MIPI

MIPI CSI subsystem, including MIPI DPHY, CSI controller and the CSI DMA.

12.1 Overview

- The Soc integrates:
  - 4 CSI_DMA instances
  - 4 MIPI-CSI interfaces
  - 2 DPHY units

Each CSI_DMA instance can process only one data stream (input/output) at a time.

- CSI_DMA supports the following pixel formats:
  - RAW
  - RGB888/RGB888x, RGB565
  - YUV422 (10-bit/8-bit), YUV420 (10-bit/8-bit)
- SMMU support:
  - CSI_DMA-1 & CSI_DMA-3: SMMU-capable
  - CSI_DMA-2 & CSI_DMA-4: No SMMU support
- Single MIPI DPHY bandwidth:
  - Max 4 lanes at 2.5 Gbps/lane
  - Total D-PHY throughput: 10 Gbps (2.5 Gbps × 4 lanes)
- Each MIPI DPHY support up to 4 data lanes and 2 cock lanes

12.1.1 Block Diagram

This diagram illustrates the MIPI CSI-2 subsystem architecture, including key components and main data paths.

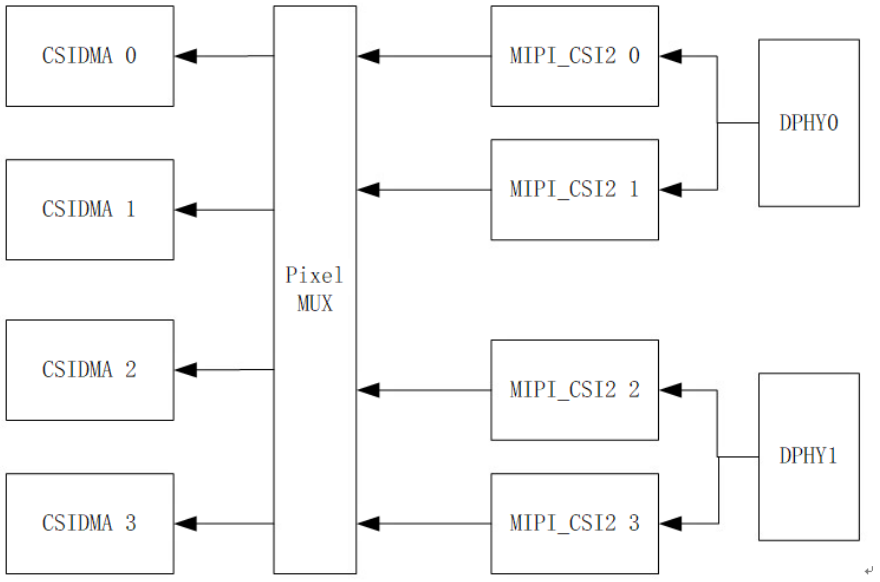


Figure 12-1 MIPI CSI Subsystem Block Diagram

- MIPI DPHYx-RX:
  Supports configuration of up to 4 data lanes and 2 clock lanes. In the application, the DPHY is configured as either 4d1c (4 data lanes and 1 clock lane) for connection with a 4-data-lane camera, or 2×2d1c (two instances of 2 data lanes and 1 clock lane) for connection with two 2-data-lane cameras.
- MIPI-CSIx:
  For CSI0 and CSI2, the hardware is configured with 4 pixel stream interfaces and 4 PPI interfaces. For CSI1 and CSI3, the hardware is configured with 1 pixel stream interface and 2 PPI interfaces.
- CSI_DMA:
  Converts the pixel data interface to the required ISP interface and uses a standard AXI4 interface to transfer pixel data to DDR memory.

12.1.2 Features

Feature Module	Configuration	Technical Details
DPHY Support	2 DPHY units	Configurable as: • 1x4D1C (Single DPHY: 4 data lanes + 1 clock lane) • 2x2D1C (Dual DPHY: 2 data lanes each + 1 clock lane)
CSI Controllers	4 controllers	• CSI0/2: 4 pixel streams + 4 PPI interfaces • CSI1/3: 1 pixel stream + 2 PPI interfaces
CSI DMA Architecture	Dual pixel conversion bridges	• Bridge 1: ISP interface conversion • Bridge 2: AXI4-to-DDR write conversion
RCSU Control Units	2 units	Each contains5 APB interfaces in total: • 1 for DPHY • 2 for CSI controllers • 2 for CSI DMA units
Clock Management	• Dynamic downscaling • Clock gating	• Adjustable CSI system/pixel clocks • APB/pixel/system clocks support gating • FW can disable unused blocks
Debug Features	• ATB debug bus • DPHY clock monitoring	• DMA bridge FSM state observation • HS Rx byte clock monitoring
Performance	Single DPHY max bandwidth	4 lanes @ 2.5Gbps/lane = 10Gbps aggregate bandwidth

12.1.2.1 Key Features

- Hardware-level channel isolation (Split mode supports dual CSI independent operation)
- Pixel format support (RAW/RGB/YUV 8/10bit)
- Standardized debug interface (ATB protocol compliant with Arm CoreSight)

Note:

Specifications comply with MIPI CSI-2 v2.1 standard and MIPI Alliance Specification for DPHY V1.2

12.2 CSI2RX Controller

12.2.1.1 Features

The CSI2RX controller IP handles MIPI CSI-2 protocol, as well as unpacking input data to pixels. ECC/CRC checking & correction is also performed

- Pixel Interface supporting
  - Byte to pixel conversion on pixel interface
  - Direct memory dump using packed byte operation
  - Flow control
- Stream Payload FIFO operation
- Monitoring of frame structure for automatic start/end of line/frame synchronization, when enabled
- LRTE Support – Interpacket Latency Reduction replaces HS-LPS-HS transition with EPDExtended support of RAW24 mode as defined for MIPI CSI-2 v3.0 Specification.

CSI2RX Front handles:

- DPHY lane swapping
- Data lane Byte packing for protocol
- Descrambling
- Clock Lane EPD control
- Discarding of EPD related spacer and filler bytes.

CSI2RX Protocol module handles:

- CSI-2 Protocol (including ECC, CRC, WC, Virtual channel & Data types, Data, Error detection)
- Packed data with associated packet information targeted at IoT applications
- Protocol Based EPD

CSI2RX Stream

Protocol Interface Control module handles:

- Virtual Channel / Data type de-interleaving

Pixel Unpacking module handles:

- Byte to pixel conversion for pixel interface

Monitor module handles:

- Start/End of frame, programmable & truncated frame informational interrupts

Frame Capture Control module handles:

- Selective processing of groups of frames.

Flow Control logic handles:

- Storing data during pushback from the system
- Alignment of data and control outputs

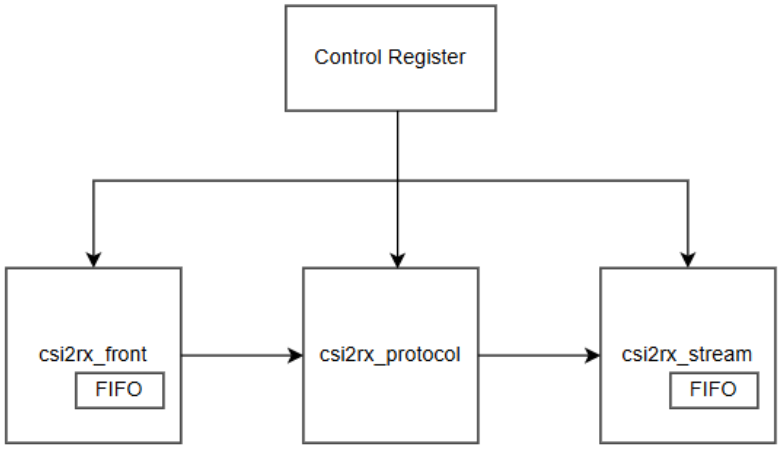


Figure 12-2 CSI controller Block Diagram

12.3 CSI Bridge

12.3.1 Overview

The MIPI CSI Bridge consists of two principal components:

- MIPI CSI IPI ASYNC Bridge
  Implements pixel buffering via FIFO
- MIPI CSI IPI DMA Bridge
  Configures deep FIFO

12.3.2 Block Diagram

The MIPI CSI bridge contains two main paths: one for the ISP use case and another for the DP in and debug use cases. The high-level block diagram for the MIPI CSI bridge is below.

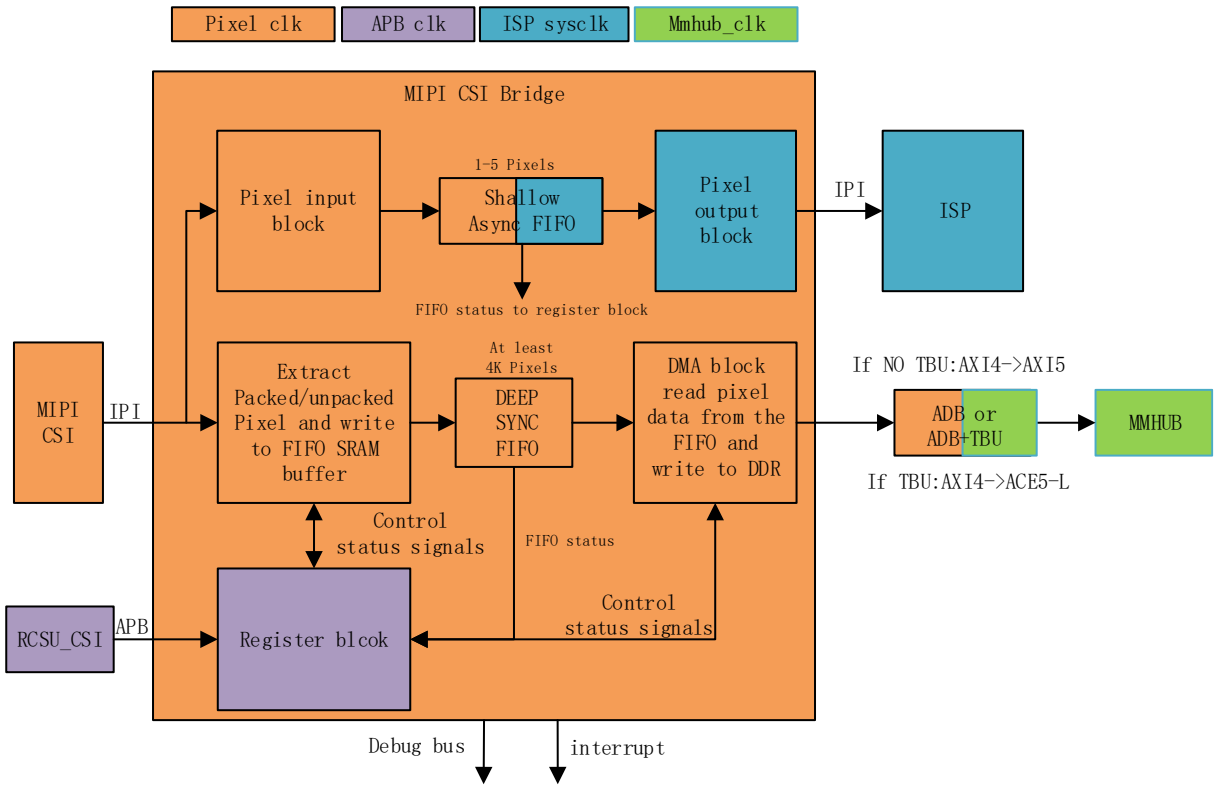


Figure 12-3 MIPI CSI Bridge Block Diagram

12.3.2.1 IPI Async Bridge

The top 3 blocks in the block diagram is IPI Async bridge, it is a bridge between the CSI (IPI interface) and the ISP (DVP interface). It will perform the clock crossing function from the MIPI pixel clock domain to the ISP system clock domain when the IPI ASYNC bridge is enabled. There is some glue logic to transfer the pixel between IPI timing to DVP timing.

12.3.2.2 MIPI CSI IPI DMA Bridge

IPI DMA bridge receives the IPI pixel data for a specific channel and specific data type from MIPI CSI controller and store the corresponding pixel data in the FIFO, then the DMA block will pack the pixel data and write the data to DDR memory according to the pixel format, stride and other settings. This path is mainly for pixel data to DDR memory. It might be used for debugging, ISP DDR path use case, and external DP in (DisplayPort as stream input) case.

12.4 PHY

The Rx PHY macro is an implementation of a Multilink MIPI DPHY Receiver that contains a DPHY Control and Rx PMA.

12.4.1 Features

Some of the important features have been highlighted below:

- High-Speed (HS): maximum up to 2.5Gbps
- Escape mode
- Remote Triggers

- LP-DT: up to 10Mbps
- ULPS mode
- Lane Configuration
- Clock lane is always unidirectional and can operate as:
- Clock RX (CRX): CIL-SCNN (HS-RX, LP-RX)
- Data lane can be unidirectional in LP mode, and can operate as:
- Data RX (DRX): CIL-SFAN (HS-RX, LP-RX)
- PHY Configuration
- RX IP: 4DRX, 2CRX, and CMN
- PPI as defined in the MIPI D-PHY specification.
- APB interface for register read write
- TAP interface for debug and testability
- Testability
- Scan: Stuck-at and At-speed
- BIST using internal loopback
- IDDQ and burn-in support
- Boundary scan to cover analog pins
- Analog Test Bus to probe critical analog signals
- PHY Isolation for HVM testing and diagnostics

Chapter 13 Debug

13.1 Overview

The chip uses Arm CoreSight System-on-chip SoC-600 Technology to support Cortex-A720/Cortex-A520 real-time debug.

13.1.1 Features

- Invasive debug with core halted
- Standard JTAG interface is supported and the Arm Serial Wire interface isn't supported.

13.1.2 Debug components address map

The following table shows the debug component's address in the debug memory map:

Table 16-1External Debugger View

component	Base address	Size
DP romtable	0x1A000000	4KB

13.2 Block Diagram

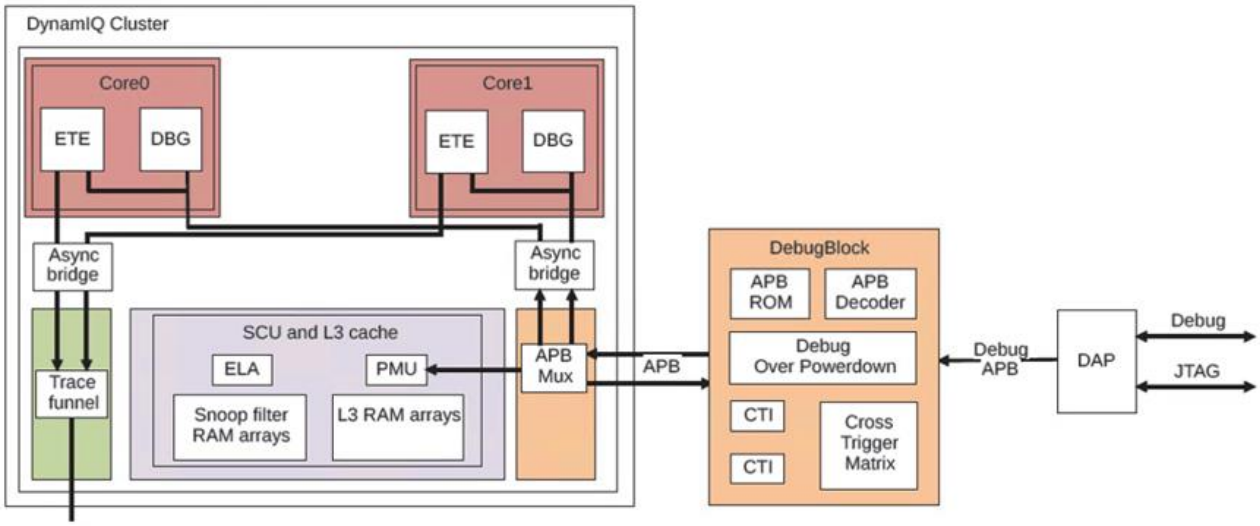


Figure 13-1 Debug System Structure

13.3 Function Description

13.3.1 DAP

The DAP has the following components:

- JTAG Debug Port
- ROM table
- APB Access Port(APB-AP)

DP:

The DP provides a low pin-count interface to DFD for an external debugger via a JTAG interface. Use the DP to provide off-chip debug connectivity into DFD.

ROM Table:

indicates the addresses of other CoreSight™ components to an external debugger.

The APB-AP:

acts as a bridge between the JTAG DP and APB bus, which translates the Debug request to the APB bus.

The DAP provides an internal ROM table connected to the APB interconnect. The Debug ROM table is loaded at address 0x1A000000. The ROM table stores the locations of the components on the Debug APB.

It indicates the addresses of other CoreSight™ components to an External Debugger.

Refer to the document *coresight_soc600_technical_reference_manual_100806_0401_00_en.pdf* for the debug detail description.

13.4 Interface Description

The following figure provides a physical connection view of the DAP JTAG-DP interface.

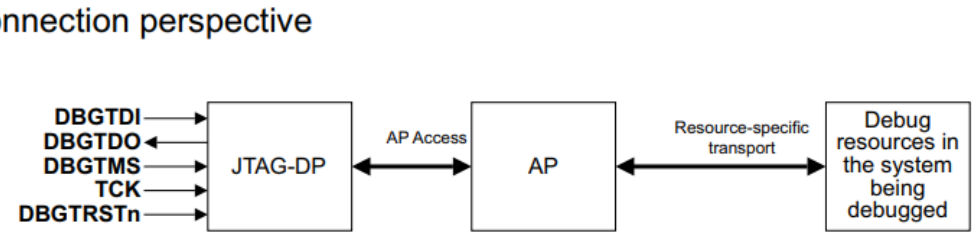


Figure 13-2 DAP JTAG-DP Interface

13.4.1 Physical Connection to the JTAG-DP

The physical connection to the JTAG-DP closely follows the JTAG model. The following table lists the recommended signals for the JTAG-DP physical connection alongside their equivalent JTAG signal names.

Table 13-2 JTAG-DP Signal Connections

JTAG-DP Signal Name	JTAG Equivalent Signal Name	Direction	Required?	Description
DBGTDI	**TDI**	Input	Yes	Debug Data In
DBGTDO	**TDO**	Output	Yes	Debug Data Out
TCK	**TCK**	Input	Yes	Debug Clock
DBGTMS	**TMS**	Input	Yes	Debug Mode Select
DBGTRSTn	**TRST**	Input	Optional	Debug TAP Reset

Table 13-3 JTAG-DP Interface Description

Pin No.	Pin Name	Voltage	Power domain	Pad Type	Direction	Default PU/PD at power up	Pin Description
AD8	JTAG_TCK	1.8V	S5	Push-Pull	I		JTAG Clock

Pin No.	Pin Name	Voltage	Power domain	Pad Type	Direction	Default PU/PD at power up	Pin Description
AA7	JTAG_TDI	1.8V	S5	Push-Pull	I		JTAG DATA IN
AB8	JTAG_TDO	1.8V	S5	Push-Pull	O		JTAG DATA OUT
AD7	JTAG_TMS	1.8V	S5	Push-Pull	I		JTAG TMS
AC7	JTAG_TRST_L	1.8V	S5	Push-Pull	I		JTAG TRST L

13.5 Register Description

For detailed information on debugging, refer to Chapter 10, Programmer's Model, in the document a*rm_coresight_soc600_technical_reference_manual_100806_0401_00_en.pdf*.