MIPI CSI-2 Receiver Subsystem v1.0

Product Guide

Vivado Design Suite

PG232 November 18, 2015

XILINX
ALL PROGRAMMABLE™

Table of Contents

IP Facts

Chapter 1: Overview

Sub-Core Details 6
Applications 8
Unsupported Features 8
Licensing and Ordering Information 8

Chapter 2: Product Specification

Standards 10
Resource Utilization 10
Port Descriptions 10
Register Space 11

Chapter 3: Designing with the Subsystem

General Design Guidelines 22
Clocking 23
Resets 23
Protocol Description 23

Chapter 4: Design Flow Steps

Customizing and Generating the Subsystem 25
Constraining the Subsystem 27
Simulation 28
Synthesis and Implementation 28

Appendix A: Debugging

Finding Help on Xilinx.com 29
Hardware Debug 31
Interface Debug 31

Appendix B: Application Software Development

Appendix C: Additional Resources and Legal Notices

Xilinx Resources 34
References 34
Revision History 35
Please Read: Important Legal Notices 35

Introduction

The Mobile Industry Processor Interface (MIPI) Camera Serial Interface (CSI-2) RX subsystem implements a CSI-2 receive interface according to the MIPI CSI-2 standard, v1.1 [Ref 1]. The subsystem captures raw images from MIPI CSI-2 camera sensors and outputs AXI4-Stream data ready for image processing. The subsystem allows fast selection of the top level parameters and automates most of the lower level parameterization. The AXI4-Stream interface allows a seamless interface to other AXI4-Stream-based subsystems.

Features

- Support for 1 to 4 D-PHY lanes
- Line rates ranging from 80 to 1500 Mb/s
- Multiple Data Type support (RAW, RGB, YUV)
- AXI IIC support for Camera Control Interface (CCI)
- Filtering based on Virtual Channel Identifier
- Support for 1, 2 or 4 pixels per sample at the output as defined in the Xilinx *AXI4-Stream Video IP and System Design Guide* (UG934) [Ref 2] format
- AXI4-Lite interface for register access to configure different subsystem options
- Dynamic selection of active lanes from configured lanes
- Interrupt generation to indicate subsystem status information
- Internal D-PHY allows direct connection to image sources

IP Facts Table	
Subsystem Specifics	
Supported Device Family(1)	UltraScale+™ Families Zynq® UltraScale+ MPSoC
Supported User Interfaces	AXI4-Lite, AXI4-Stream
Resources	Performance and Resource Utilization web page
Provided with Subsystem	
Design Files	Encrypted RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Synthesis Constraints File: XDC
Simulation Model	Not Provided
Supported S/W Driver(2)	Standalone
Tested Design Flows(3)	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx at the Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. Standalone driver details can be found in the SDK directory (<*install_directory*>/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from the Xilinx Wiki page.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The MIPI CSI-2 RX subsystem allows you to quickly create systems based on the MIPI protocol. It interfaces between MIPI-based image sensors and an image sensor pipe. An internal high speed physical layer design, D-PHY, is provided that allows direct connection to image sources. The top level customization parameters select the required hardware blocks needed to build the subsystem. Figure 1-1 shows the subsystem architecture.

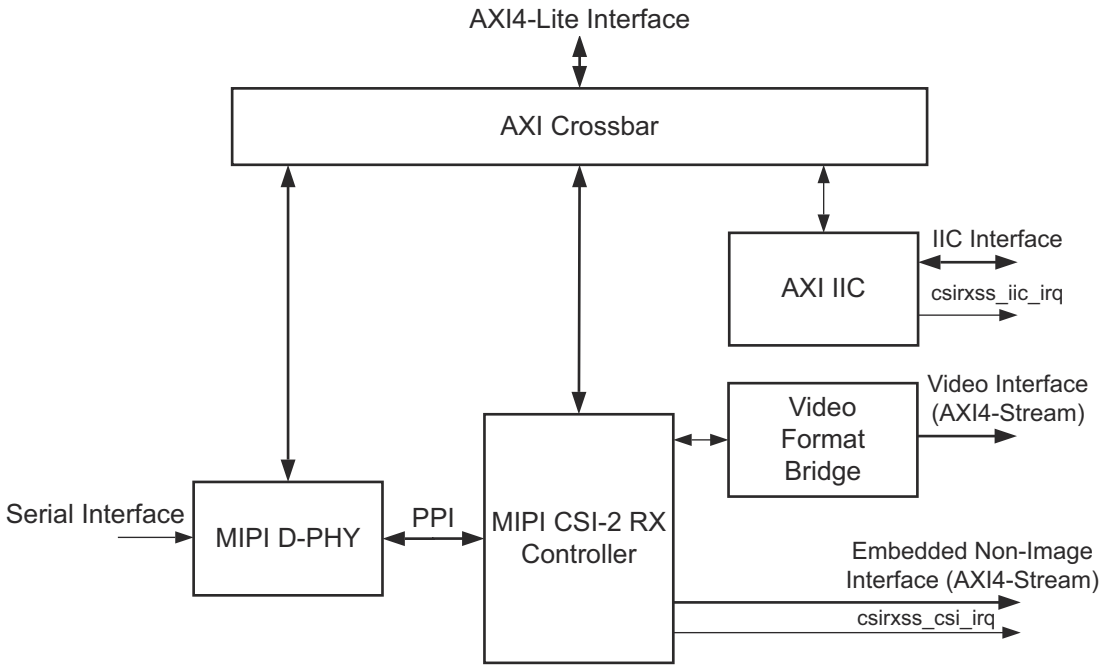

Figure 1-1: **Subsystem Architecture**

The subsystem consists of the following sub-cores:

- MIPI D-PHY
- MIPI CSI-2 RX Controller
- AXI Crossbar
- Video Format Bridge
- AXI IIC

Sub-Core Details

MIPI D-PHY

See the *MIPI D-PHY v1.0 LogiCORE IP Product Guide (*PG202*)* [Ref 3] for details.

MIPI CSI-2 RX Controller

The MIPI CSI-2 RX Controller core receives 8-bit data per lane, with support for up to 4 lanes, from the MIPI D-PHY core through the PPI. This data is then processed to extract the real image information. The final extracted image is made available to the user/processor interface using the AXI4-Stream protocol.

Features of this core include:

- 1–4 lane support, with register support to select active lanes (the actual number of available lanes to be used)
- Short and long packets with all word count values supported
- Primary and many secondary video formats supported
- Data Type (DT) interleaving
- Virtual Channel Identifier (VC) interleaving
- Combination of Data Type and VC interleaving
- Multilane interoperability
- Error Correction Code (ECC) for 1-bit error correction and 2-bit error detection in packet header
- CRC check for payload data
- Maximum data rate of 1.5 Gb/s
- Pixel byte packing based on data format
- AXI4-Lite interface to access core registers
- Low power state detection
- Error detection (D-PHY Level Errors, Packet Level Errors, Protocol Decoding Level Errors)
- AXI4-Stream interface with 32/64-bit TDATA width support to offload pixel information externally
- Interrupt support for indicating internal status/error information

As shown in Table 1-1 the embedded non-image AXI4-Stream interface data width is selected based on the Data Type selected.

Table 1-1: **Embedded Non-Image AXI4-Stream Interface TDATA Widths**

Data Type (DT)	AXI4-Stream Interface TDATA Width
RAW6	32
RAW7	32
RAW8	32
RAW10	64
RAW12	64
RAW14	64
All RGB	64
YUV 422 8bit	64

Abrupt termination events such as a soft reset, disabling a core while a packet is being written to the line buffer, or a line buffer full condition results in early termination. The termination is implemented by assertion of EOL on the video interface or TLAST and TUSER[1] on the embedded non-image interface, based on the current long packet being processed.

AXI Crossbar

See the *AXI Interconnect LogiCORE IP Product Guide* [Ref 4] for details.

Video Format Bridge

The Video Format Bridge core uses the user-selected VC and Data Type information to filter only the required AXI4-Stream data beats. This AXI4-Stream data is further processed based on the Data Type information and the output is based on the requested number of pixels per beat. The output interface adheres to the protocol defined in the *AXI4-Stream Video IP and System Design Guide* (UG934) [Ref 2].

For unaligned transfers there is no way to specify the partial final output (TKEEP) for the output interface. Ensure that you take this into consideration and discard the unintended bytes in the last beats when there are un-aligned transfers.

AXI IIC

See the *AXI IIC Bus Interface v2.0 LogiCORE IP Product Guide (*PG090) [Ref 5] for details.

Applications

The Xilinx MIPI CSI-2 RX controller implements a Camera Serial Interface between a camera sensor and a programmable device performing baseband processing. Bandwidth requirement for the camera sensor interface has gone up due to the development of higher resolution cameras. Traditional parallel interfaces require an increasing number of signal lines resulting in higher power consumption. The new high speed serial interfaces, such as MIPI CSI specifications, address these expanding bandwidth requirements without sacrificing power. MIPI is a group of protocols defined by the mobile industry group to standardize all interfaces within mobile platforms such as mobile phones and tablets. However the large volumes and the economies of scale of the mobile industry is forcing other applications to also adopt these standards. As such MIPI-based camera sensors are being increasingly used in applications such as driver assistance technologies in automotive applications, video security surveillance cameras, video conferencing and emerging applications such as virtual and augmented reality.

Unsupported Features

- Some YUV Data Types (YUV 420 (8-bit and 10-bit), YUV 422 10-bit) are not supported.

Licensing and Ordering Information

License Checkers

If the IP requires a license key, the key must be verified. The Vivado® design tools have several license checkpoints for gating licensed IP through the flow. If the license check succeeds, the IP can continue generation. Otherwise, generation halts with error. License checkpoints are enforced by the following tools:

- Vivado synthesis
- Vivado implementation
- write_bitstream (Tcl command)

IMPORTANT: *IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.*

License Type

This Xilinx module is provided under the terms of the Xilinx Core License Agreement. The module is shipped as part of the Vivado® Design Suite. For full access to all core functionalities in simulation and in hardware, you must purchase a license for the core. Contact your local Xilinx sales representative for information about pricing and availability.

For more information, visit the MIPI CSI-2 RX Subsystem product web page.

Information about other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

Standards

- MIPI Alliance Standard for Camera Serial Interface CSI-2 [Ref 1]
- MIPI Alliance Physical Layer Specifications, D-PHY Specification [Ref 6]
- Processor Interface, AXI4-Lite: see the *Vivado Design Suite: AXI Reference Guide* (UG1037) [Ref 7]
- Output Pixel Interface: see the *AXI4-Stream Video IP and System Design Guide* (UG934) [Ref 2]

Resource Utilization

For full details about performance and resource utilization, visit the Performance and Resource Utilization web page.

Port Descriptions

The MIPI CSI-2 RX Subsystem I/O signals are described in Table 2-1.

Table 2-1: **Port Descriptions**

Signal name	Direction	Description
lite_aclk	Input	AXI clock
lite_aresetn	Input	AXI reset. Active-Low
S00_AXI*		AXI4-Lite interface, defined in the *Vivado Design Suite: AXI Reference Guide* (UG1037) [Ref 7]
dphy_clk_200M(1)	Input	Clock for D-PHY core. Must be 200 MHz.
clk_lane_rxp	Input	Differential serial data input pin for D-PHY RX clock lane
clk_lane_rxn		

Table 2-1: **Port Descriptions *(Cont'd)***

Signal name	Direction	Description
data_lane_rxp[3:0]	Input	Differential serial data input pin for D-PHY RX data lane(s)
data_lane_rxn[3:0]		
video_aclk	Input	Subsystem clock
video_aresetn	Input	Subsystem reset. Active-Low.
video_*		AXI4-Stream interface with the Xilinx *AXI4-Stream Video IP and System Design Guide* (UG934) [Ref 2] compatibility
emb_nonimg		AXI4-Stream interface for embedded non image packets
csirxss_csi_irq	Output	Interrupt from CSI-2 RX Controller
csirxss_iic_irq	Output	Interrupt from AXI IIC

Notes:

1. The active-High reset for the MIPI D-PHY core is generated internally by negating the external active-Low reset (`video_aresetn`).

Register Space

This section details registers available in the MIPI CSI-2 RX Subsystem. The address map is split into following regions:

- MIPI CSI-2 RX Controller core
- AXI IIC core
- MIPI D-PHY core

Each IP core is given an address space of 64K. Example offset addresses from the system base address when the AXI IIC and MIPI D-PHY registers are enabled are shown in Table 2-2.

Table 2-2: **Sub-Core Address Offsets**

IP Cores	Offset
MIPI CSI-2 RX Controller	0x0_0000
AXI IIC	0x1_0000
MIPI D-PHY	0x2_0000(1)

Notes:

1. When the AXI IIC core is not present, the MIPI D-PHY offset moves up and starts at 0x1_0000. The software driver handles this seamlessly.

MIPI CSI-2 RX Controller Core Registers

Table 2-3 specifies the name, address, and description of each firmware addressable register within the MIPI CSI-2 RX controller core.

Table 2-3: **MIPI CSI-2 RX Controller Core Registers**

Address Offset	Register Name	Description
0x00	Core Configuration	Core configuration options
0x04	Protocol Configuration	Protocol configuration options
0x08	Reserved(1)	
0x0C	Reserved	
0x10	Core status	Internal status of the core
0x14	Reserved	
0x18	Reserved	
0x1C	Reserved	
0x20	Global Interrupt Enable	Global interrupt enable registers
0x24	Interrupt status	Interrupt status register
0x28	Interrupt enable	Interrupt enable register
0x2C	Reserved	
0x30	Generic short packet	Short packet data
0x34	Reserved	
0x38	Reserved	
0x3C	Clock Lane information	Clock lane status information
0x40	Lane0 Information	Lane 0 status information
0x44	Lane1 Information	Lane 1 status information
0x48	Lane2 Information	Lane 2 status information
0x4C	Lane3 Information	Lane 3 status information
0x50	Reserved	
0x54	Reserved	
0x58	Reserved	
0x5C	Reserved	
0x60	Image Information 1 for VC0	Image information 1 of the current processing packet with VC of 0
0x64	Image Information 2 for VC0	Image information 2 of the current processing packet with VC of 0
0x68	Image Information 1 for VC1	Image information 1 of the current processing packet with VC of 1
0x6C	Image Information 2 for VC1	Image information 2 of the current processing packet with VC of 1

Table 2-3: **MIPI CSI-2 RX Controller Core Registers *(Cont'd)***

Address Offset	Register Name	Description
0x70	Image Information 1 for VC2	Image information 1 of the current processing packet with VC of 2
0x74	Image Information 2 for VC2	Image information 2 of the current processing packet with VC of 2
0x78	Image Information 1 for VC3	Image information 1 of the current processing packet with VC of 3
0x7C	Image Information 2 for VC3	Image information 2 of the current processing packet with VC of 3

Notes:

1. Access type and reset value for all the reserved bits in the registers is read-only with value 0.
2. Register accesses should be word aligned and there is no support for a write strobe. WSTRB is not used internally.
3. Only the lower 7 bits (6:0) of the read and write address of the AXI4-Lite interface are decoded. This means that accessing address 0x00 and 0x80 results in reading the same address of 0x00.
4. Reads and writes to addresses outside this table do not return an error.

Core Configuration Register

The Core Configuration register is described in Table 2-4 and allows you to enable and disable the MIPI CSI-2 RX Controller core and apply a soft reset during core operation.

Table 2-4: **Core Configuration Register**

Bits	Name	Reset Value	Access	Description
31–2	Reserved	N/A	N/A	Reserved
1	Soft Reset	0x0	R/W	1: Resets the core 0: Takes core out of soft reset All registers reset to their default value (except for this bit, Core Enable and Active lanes configuration). In addition to resetting registers when this bit is set to 1: • Shut down port is not asserted on the PPI lanes • Internal FIFOs (PPI, Packet, Generic Short Packet) are flushed • Control Finite State Machine (FSM) stops processing current packet. Any partially written packet to memory is marked as errored. This packet, when made available through the AXI4-Stream interface, reports the error on TUSER[1].

Table 2-4: **Core Configuration Register *(Cont'd)***

Bits	Name	Reset Value	Access	Description
0	Core Enable	0x1	R/W	1: Enables the core to receive and process packets 0: Disables the core for operation When disabled: • Shuts down port assertion on the PPI lanes • Internal FIFOs (PPI, Packet, Generic Short Packet) are flushed • Control FSM stops processing current packet Any partially written packet to memory is marked as errored. This packet, when made available through the AXI4-Stream interface, reports the error on TUSER[1].

Notes:

1. The short packet and line buffer FIFO full conditions take a few clocks to reflect in the register clock domain from the core clock domain due to Clock Domain Crossing (CDC) blocks.

Protocol Configuration Register

The Protocol Configuration register is described in Table 2-5 and allows you to configure protocol specific options such as the number of lanes to be used.

Table 2-5: **Protocol Configuration Register**

Bits	Name	Reset Value	Access	Description
31–5	Reserved	N/A	N/A	Reserved
4–3	Maximum Lanes(1)	Number of lanes configured during core generation	R	Maximum lanes of the core 0x0—1 Lane 0x1—2 Lanes 0x2—3 Lanes 0x3—4 Lanes
2	Reserved	N/A		Reserved
1–0	Active Lanes	Number of lanes configured during core generation	R(2)/W	Active lanes in the core(3) 0x0—1 Lane 0x1—2 Lanes 0x2 —3 Lanes 0x3—4 Lanes

Notes:

1. Maximum Lanes cannot exceed the number of lanes as set by the **Serial Data Lanes** parameter at generation time.
2. A read from this register reflects the current number of lanes being used by core. This is useful when dynamically updating the active lanes during core operation to ensure that the core is using the new active lanes information. See Chapter 3, Designing with the Subsystem for more information.
3. Active Lanes cannot exceed the Maximum Lanes as set in the Protocol Configuration register setting of bits 4–3.

Core Status Register

The Core Status register is described in Table 2-6.

Table 2-6: **Core Status Register**

Bits	Name	Reset Value	Access	Description
31–16	Packet Count	0x0	R	Counts number of long packets written to the line buffer • No roll-over of this counter reported/ supported • Count includes error packets (if any)
15–4	Reserved	N/A	N/A	N/A
3	Short packet FIFO Full	0x0	R	Indicates the current status of short packet FIFO full condition
2	Short packet FIFO not empty	0x0	R	FIFO not empty: Indicates the current status of short packet FIFO not empty condition
1	Stream Line buffer Full	0x0	R	Indicates the current status of line buffer full condition
0	Soft reset/Core disable in progress	0x0	R	Set to 1 by the core to indicate that internal soft reset/core disable activities are in progress

Global Interrupt Enable Register

The Global Interrupt Enable register is described in Table 2-7.

Table 2-7: **Global Interrupt Enable Register**

Bits	Name	Reset Value	Access	Description
31–1	Reserved	N/A	N/A	Reserved
0	Global Interrupt enable	0x0	R/W	Master enable for the device interrupt output to the system 1: Enabled—the corresponding Interrupt Enable register (IER) bits are used to generate interrupts 0: Disabled—Interrupt generation blocked irrespective of IER bits

Interrupt Status Register

The Interrupt Status register (ISR) is described in Table 2-8 and captures the error and status information for the core.

Table 2-8: **Interrupt Status Register**

Bits	Name	Reset Value	Access[1]	Description
31	Frame Received	0x0	R/W1C	Asserted when the Frame End (FE) short packet is received for the current frame
30–22	Reserved	N/A	N/A	N/A
21	Incorrect lane configuration	0x0	R/W1C	Asserted when Active lanes is greater than Maximum lanes in the protocol configuration register
20	Short packet FIFO full	0x0	R/W1C	Active-High signal asserted when the short packet FIFO full condition detected
19	Short packet FIFO not empty	0x0	R/W1C	Active-High signal asserted when short packet FIFO not empty condition detected
18	Stream line buffer full	0x0	R/W1C	Asserts when the line buffer is full[2]
17	Stop state	0x0	R/W1C	Active-High signal indicates that the lane module is currently in Stop state[3]
16	Escape Ultra Low Power mode (RxUlpmEsc)[9]	0x0	R/W1C	Active-High signal asserted to indicate that the lane module has entered the Escape Ultra Low Power mode[3]
15	Escape entry error (ErrEsc)	0x0	R/W1C	Asserted when an unrecognized escape entry command is received[3]
14	Control error	0x0	R/W1C	Asserted when an incorrect line state sequence detected[3]
13	SoT error (ErrSoTHS)	0x0	R/W1C	Indicates Start-of-Transmission (SoT) error detected and corrected[3]
12	SoT sync error (ErrSotSyncHS)	0x0	R/W1C	Indicates SoT synchronization completely failed[3]
11	ECC 2-bit error (ErrEccDouble)	0x0	R/W1C	Asserted when an ECC syndrome is computed and two bit errors detected in the received packet header
10	ECC 1-bit error (Detected and Corrected) (ErrEccCorrected)	0x0	R/W1C	Asserted when an ECC syndrome was computed and a single bit error in the packet header was detected and corrected
9	CRC error (ErrCrc)	0x0	R/W1C	Asserted when the computed CRC code is different from the received CRC code
8	Unsupported Data Type (ErrID)	0x0	R/W1C	Asserted when a packet header is decoded with an unrecognized or un-implemented data ID

Table 2-8: **Interrupt Status Register *(Cont'd)***

Bits	Name	Reset Value	Access[1]	Description
7	Frame synchronization error for VC3 (ErrFrameSync)	0x0	R/W1C	Asserted when an FE is not paired with a Frame Start (FS) on the same virtual channel[4]
6	Frame level error for VC3 (ErrFrameData)	0x0	R/W1C	Asserted after an FE when the data payload received between FS and FE contains errors. The data payload errors are CRC errors.
5	Frame synchronization error for VC2 (ErrFrameSync)	0x0	R/W1C	Asserted when an FE is not paired with a FS on the same virtual channel[4]
4	Frame level error for VC2 (ErrFrameData)	0x0	R/W1C	Asserted after an FE when the data payload received between FS and FE contains errors. The data payload errors are CRC errors.
3	Frame synchronization error for VC1 (ErrFrameSync)	0x0	R/W1C	Asserted when an FE is not paired with a FS on the same virtual channel[4]
2	Frame level error for VC1 (ErrFrameData)	0x0	R/W1C	Asserted after an FE when the data payload received between FS and FE contains errors. The data payload errors are CRC errors.
1	Frame synchronization error for VC0 (ErrFrameSync)	0x0	R/W1C	Asserted when a FE is not paired with a FS on the same virtual channel[4]
0	Frame level error for VC0 (ErrFrameData)	0x0	R/W1C	Asserted after an FE when the data payload received between FS and FE contains errors. The data payload errors are CRC errors.

Notes:

1. W1C = Write 1 to clear.
2. In a line buffer full condition, reset the core using the external reset, video_aresetn.
3. Reported through the PPI.
4. An ErrSotSyncHS error also generates this error signal.
5. Short packet and line buffer FIFO full conditions take a few clock periods to reflect in the register clock domain from the core clock domain due to Clock Domain Crossing (CDC) blocks.
6. All PPI signals captured in the ISR take a few clock periods to reflect in the register clock domain from the PPI clock domain due to CDC blocks.
7. Frame level errors due to ErrSotSyncHS are mapped to the recent VC processed by the ECC block of the core.
8. Set conditions take priority over the reset conditions for the ISR bits.
9. Signal names in brackets are defined in the *MIPI Alliance Standard for Camera Serial Interface CSI-2* [Ref 1].

VC Mapping

In the event of an `ErrEccDouble` error, the VC is mapped to the VC reported in the current packet header (even if corrupted).

In the event of an `ErrSotSyncHS` error, the VC is mapped to the previous VC processed because in this case the packet header is not available.

Interrupt Enable Register

The Interrupt Enable register (IER) is described in Table 2-9 and allows you to selectively generate an interrupt at the output port for each error/status bit in the ISR. An IER bit set to 0 does not inhibit an error/status condition from being captured, but inhibits it from generating an interrupt.

Table 2-9: **Interrupt Enable Register**

Bits	Name	Reset Value	Access	Description
31	Frame Received	0x0	R/W	Set bits in this register to 1 to generate the required interrupts. Set to 0 to disable the interrupt. For a description of the specific interrupt you are enabling/disabling in this register see the ISR descriptions in Table 2-8.
30–22	Reserved	N/A	N/A	
21	Incorrect lane configuration	0x0	R/W	
20	Short packet FIFO full	0x0	R/W	
19	Short packet FIFO empty	0x0	R/W	
18	Stream line buffer full	0x0	R/W	
17	Stop state	0x0	R/W	
16	Escape Ultra Low Power mode	0x0	R/W	
15	Escape entry error	0x0	R/W	
14	Control error	0x0	R/W	
13	SoT error	0x0	R/W	
12	SoT Sync error	0x0	R/W	
11	ECC 2-bit error	0x0	R/W	
10	ECC 1-bit error (Detected and Corrected)	0x0	R/W	
9	CRC error	0x0	R/W	
8	Unsupported Data Type	0x0	R/W	
7	Frame synchronization error for VC3	0x0	R/W	
6	Frame level error for VC3	0x0	R/W	
5	Frame synchronization error for VC2	0x0	R/W	
4	Frame level error for VC2	0x0	R/W	
3	Frame synchronization error for VC1	0x0	R/W	
2	Frame level error for VC1	0x0	R/W	
1	Frame synchronization error for VC0	0x0	R/W	
0	Frame level error for VC0	0x0	R/W	

Generic Short Packet Register

The Generic Short Packet register is described in Table 2-10. Packets received with generic short packet codes are stored in a 31-deep internal FIFO and are made available through this register. The following conditions reset the FIFO:

- External reset on `video_aresetn`
- Core disable or soft reset through register settings.

Table 2-10: **Generic Short Packet Register**

Bits	Name	Reset Value	Access	Description
31–24	Reserved	N/A	N/A	Reserved
23–8	Data	0x0	R	16-bit short packet data
7–6	Virtual Channel	0x0	R	Virtual channel number
5–0	Data Type	0x0	R	Generic short packet code

Clock Lane Information Register

The Clock Lane Information register is described in Table 2-11. The Stop state and the Ultra Low Power state (rxulpsclknot) are captured in this register.

Table 2-11: **Clock Lane Information Register**

Bits	Name	Reset Value	Access	Description
31–2	Reserved	N/A	N/A	Reserved
1	Stop state	0x0	R	Stop state on clock lane
0	Ultra low power state	0x0	R	Ultra Low Power State on clock lane

Lane<n> Information Registers

The Lane<n> Information register, where n is 0, 1, 2 or 3, is described in Table 2-12 and provides the status of the <n> data lane. This register is reset when any write to the Protocol Configuration register is detected, irrespective of whether the Protocol Configuration register contents are updated or not.

Table 2-12: **Lane 0, 1, 2, 3 Information Register**

Bits	Name	Reset Value	Access	Description[(2)]
31–6	Reserved	N/A	N/A	Reserved
5	Stop state	0x0	R	Detection of Stop state Active-High signal indicates that the lane module is currently in stop state
4	Escape Ultra Low Power mode	0x0	R	Detection of Escape Ultra Low Power mode (RxUlpmEsc) Active-High signal indicates that the lane module has entered the ultra low power mode

Table 2-12: **Lane 0, 1, 2, 3 Information Register *(Cont'd)***

Bits	Name	Reset Value	Access	Description[2]
3	Escape entry error	0x0	R	Detection of Escape Entry Error (ErrEsc) Asserted when an unrecognized escape entry command is received
2	Control error	0x0	R	Detection of Control Error (ErrControl) Asserted when an incorrect line state sequence detected
1	SoT error	0x0	R	Detection of SoT Error (ErrSotHS) Indicates SoT error detected and corrected
0	SoT Sync error	0x0	R	Detection of SoT Synchronization Error (ErrSotSyncHS) Indicates that SoT synchronization failed

Notes:

1. Lane Information registers are present only for the maximum defined number of lanes. Reads to others registers gives 0x0.
2. All bits are reported through the PPI.

Image Information 1 Registers (VC0 to VC3)

The Image Information 1 registers are described in Table 2-13 and provide image information for line count and byte count per VC. The byte count gets updated whenever a long packet (from Data Types 0x18 and above) for the corresponding virtual channel is processed by the control FSM. The line count is updated whenever the packet is written into the line buffer.

Table 2-13: **Image Information 1 Registers**

Bits	Name	Reset Value	Access	Description
31–16	Line count	0x0	R	Number of long packets written to line buffer
15–0	Byte count	0x0	R	Byte count of current packet being processed by the control FSM

Image Information 2 Registers (VC0 to VC3)

The Image Information 2 registers are described in Table 2-14 and provide the image information Data Type. The Data Type is updated whenever a long packet (Data Types 0x18 and above) for the corresponding virtual channel is processed by the control FSM.

Table 2-14: **Image Information 2 Registers**

Bits	Name	Reset Value	Access	Description
31–6	Reserved	N/A	N/A	Reserved
5–0	Data Type	0x0	R	Data Type of current packet being processed by control FSM

AXI IIC Registers

For details about AXI IIC registers, see the *AXI IIC Bus Interface v2.0 LogiCORE IP Product Guide (*PG090) [Ref 5] for details.

MIPI D-PHY Registers

For details about MIPI D-PHY registers, see the *MIPI D-PHY v1.0 LogiCORE IP Product Guide (*PG202*)* [Ref 3].

Designing with the Subsystem

This chapter includes guidelines and additional information to facilitate designing with the subsystem.

General Design Guidelines

The subsystem fits into a image sensor pipe receive path. The input to the subsystem must be connected to an image source such as an image sensor transmitting data that adheres to the MIPI protocol. The output of the subsystem is image data in AXI4-Stream format. Based on the throughput requirement the output interface can be tuned using customization parameters available for the subsystem.

Because the MIPI protocol does not allow throttling on the input interface, the module connected to the output of this subsystem should have sufficient bandwidth for the data generated by the image sensor.

Active lanes used by the subsystem can be dynamically configured using the following guidelines.

1. Program the required lanes in the Protocol Configuration register.
2. The subsystem internally updates the new lanes information after the current packet complete indication is seen (that is, when the current active lanes indicate a Stop state condition) and a subsequent RxByteClkHS signal is seen on the PPI.
3. A read from the Protocol Configuration register reflects the new value after the subsystem has successfully updated the new lanes information internally.
4. Do not send the new updated lanes traffic until the read from Protocol Configuration registers reflects the new value.

Clocking

The subsytem clocks are described in Table 3-1. Clock frequencies should be selected to match the throughput requirement of the downstream video pipe IP cores.

Table 3-1: **Subsystem Clocks**

Clock Name	Description
lite_aclk(1)	AXI4-Lite clock used by the register interface of all IP cores in the subsystem.
video_aclk	Clock used as core clock for all IP cores in the subsystem.
dphy_clk_200M	See the *MIPI D-PHY v1.0 LogiCORE IP Product Guide* (PG202) [Ref 3] for information on this clock.

Notes:
1. The lite_aclk clock should less than or equal to video_aclk.

Resets

The subsystem has two reset ports:

- `lite_aresetn`: Active-Low reset for the AXI4-Lite register interface.
- `video_aresetn`: Active-Low reset for the subsystem blocks.

The duration of `video_aresetn` should be at a minimum two cycles of the slowest clock.

Protocol Description

Programming Sequence

This section contains the programming sequence for the subsystem. Program and enable the components of subsystem in the following order:

1. AXI IIC (if included)
2. MIPI CSI-2 RX Controller
3. MIPI D-PHY (if register interface is enabled)

Address Map Example

Table 3-2 shows an example based on a subsystem base address of 0x44A0_0000 (32 bits) when the AXI IIC core is included and the MIPI D-PHY register interface is enabled.

Table 3-2: **Address Map Example**

Core	Base address
MIPI CSI-2 RX Controller	0x44A0_0000
AXI IIC	0x44A1_0000
MIPI D-PHY	0x44A2_0000(1)

Notes:

1. When the AXI IIC IP core is not present and the MIPI D-PHY register interface is enabled, the base address of the MIPI D-PHY starts with 0x44A1_0000.

AXI IIC IP Core Programming

See the *AXI IIC Bus Interface v2.0 LogiCORE IP Product Guide (*PG090) [Ref 5] for AXI IIC IP core programming details.

MIPI CSI-2 RX Controller Core Programming

MIPI CSI-2 RX Controller Programming Sequence

1. After power on reset (`video_aresetn`), the Core Enable bit is, by default, set to 1 so the core starts processing packets sent on the PPI. Active Lanes is set to Maximum Lanes which is configured in the Vivado IDE using the **Serial Data Lanes** parameter.
2. Disabling and re-enabling the core
   - Disable the core using the Core Configuration Register (set the Core Enable bit to 0 or the Soft reset bit to 1).
   - Wait until the core clears the Soft reset/Core disable in progress bit by reading the Core Status Register.
   - Change the required core settings (for example, active lanes configuration, enabling interrupts)
   - Re-enable the core (set the Core Enable bit to 1 or the Soft Reset bit to 0)

MIPI D-PHY IP Core Programming

See the *MIPI D-PHY v1.0 LogiCORE IP Product Guide (*PG202*)* [Ref 3] for MIPI D-PHY IP core programming details.

Design Flow Steps

This chapter describes customizing and generating the subsystem, constraining the subsystem, and the simulation, synthesis and implementation steps that are specific to this subsystem. More detailed information about the standard Vivado® design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 8]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 9]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 10]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 11]

Customizing and Generating the Subsystem

This section includes information about using Xilinx tools to customize and generate the subsystem in the Vivado Design Suite.

If you are customizing and generating the subsystem in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 8] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl console.

You can customize the IP for use in your design by specifying values for the various parameters associated with the subsystem using the following steps:

1. Select the IP from the Vivado IP catalog.
2. Double-click the selected IP or select the **Customize IP** command from the toolbar or right-click menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 9] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 10].

Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). The layout depicted here might vary from the current version.

The subsystem configuration screen is shown in Figure 4-1.

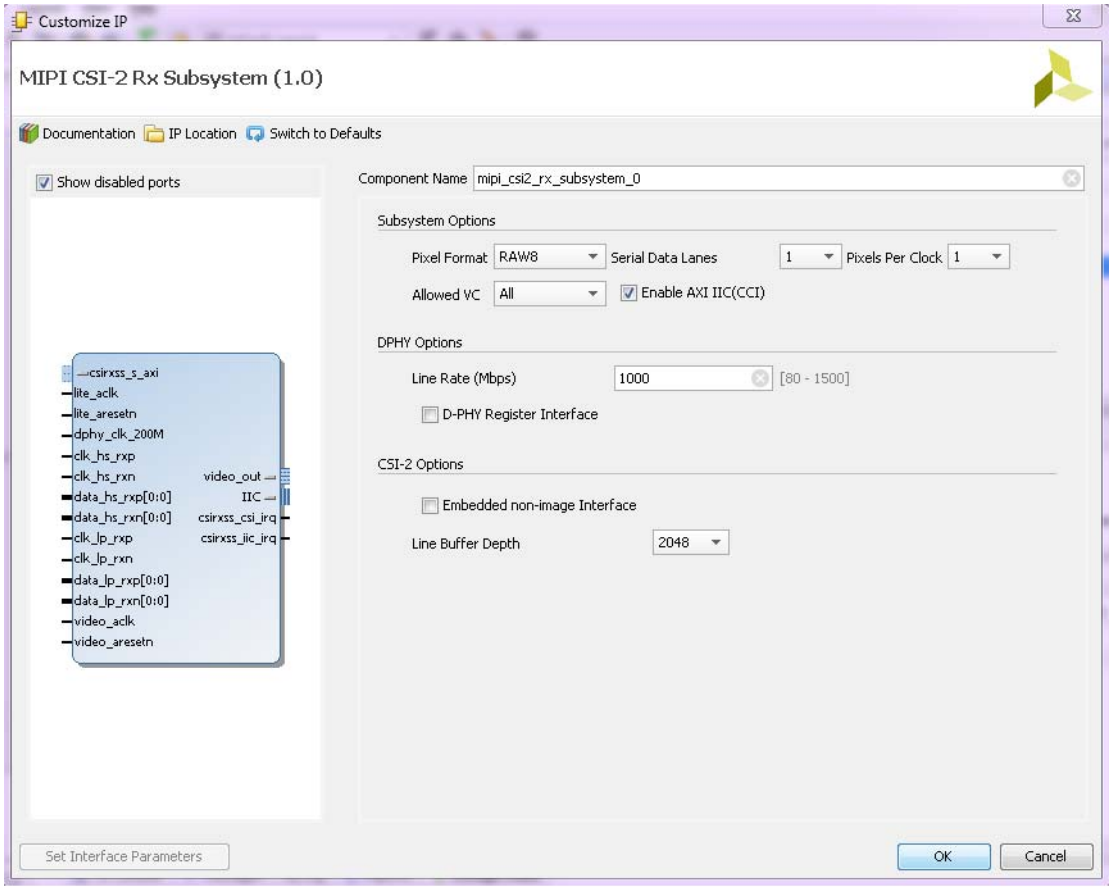

Figure 4-1: **Subsystem Customization Screen**

Component Name: The Component Name is used as the name of the top-level wrapper file for the subsystem. The underlying netlist still retains its original name. Names must begin with a letter and must be composed from the following characters: a through z, 0 through 9, and "_". The default is mipi_csi2_rx_subsystem_0.

Pixel Format: Select Data Type (pixel format) as per the CSI-2 protocol (RAW6, RAW7, RAW8, RAW10, RAW12, RAW14, RGB888, RGB666, RGB565, RGB555, RGB444, YUV422_8bit).

Serial Data Lanes: Select the maximum number of D-PHY lanes for this subsystem instance. Values are 1, 2, 3, or 4.

Pixels Per Clock: Select the number of pixels to output per clock on output interface. Values are 1, 2, or 4.

Allowed VC: Select the VC values to be used to while processing the packets. Values are All, 0, 1, 2, or 3.

Enable AXI IIC: Select to add the AXI IIC core (for CCI support).

Line Rate (Mbps): Select the line rate for the MIPI D-PHY core. Value in the range, 80 to 1500.

D-PHY Register Interface: Select to enable the register interface for the MIPI D-PHY core.

Embedded non-image Interface: Select to process and offload embedded non-image CSI-2 packets using a separate AXI4-Stream interface. If unselected, such packets are not processed and are ignored by the CSI-2 RX controller.

Line Buffer Depth: Depth of internal RAM used to accommodate throttling on the output video interface. Values are 128, 256, 512, 1024, 2048, 4096, 8192, or 16384.

Note: There is no throttling allowed on the input to the PPI.

User Parameters

Table 4-1 shows the relationship between the fields in the Vivado IDE and the User Parameters (which can be viewed in the Tcl Console).

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship**

Vivado IDE Parameter	User Parameter	Default Value
Pixel Format	CMN_PXL_FORMAT	RAW8
Serial Data Lanes	CMN_NUM_LANES	1
Allowed VC	CMN_VC	All
Pixels Per Clock	CMN_NUM_PIXELS	1
Enable AXI IIC(CCI)	CMN_INC_IIC	True
Line Rate (Mbps)	DPY_LINE_RATE	1000
D-PHY Register Interface	DPY_EN_REG_IF	False
Embedded non-image Interface	CSI_EMB_NON_IMG	False
Line Buffer Depth	CSI_BUF_DEPTH	2048

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 9].

Constraining the Subsystem

This section contains information about constraining the subsystem in the Vivado Design Suite.

Required Constraints

The XDC constraints are delivered when the IP core is generated.

Device, Package, and Speed Grade Selections

This section is not applicable for this subsystem.

Clock Frequencies

See Clocking.

Clock Management

This section is not applicable for this subsystem.

Clock Placement

This section is not applicable for this subsystem.

Banking

This section is not applicable for this subsystem.

Transceiver Placement

This section is not applicable for this subsystem.

I/O Standard and Placement

The MIPI D-PHY core `clk_lane_rx*` and `data_lane_rx*` ports need to be mapped to correct package pins based on the device selected.

Simulation

For comprehensive information about Vivado simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 11].

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 9].

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

TIP: *If the IP generation halts with an error, there might be a license issue. See License Checkers in Chapter 1 for more details.*

Finding Help on Xilinx.com

To help in the design and debug process when using the MIPI CSI-2 Receiver Subsystem, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the MIPI CSI-2 Receiver Subsystem. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this subsystem can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as:

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the MIPI CSI-2 Receiver Subsystem

AR: 65242

Technical Support

Xilinx provides technical support at the Xilinx Support web page for this IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

Xilinx provides premier technical support for customers encountering issues that require additional assistance.

To contact Xilinx Technical Support, Debug Tools

There are many tools available to address MIPI CSI-2 Receiver Subsystem design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

The Vivado® Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. The debug feature also allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx devices.

The Vivado logic analyzer is used with the logic debug IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 13].

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. The Vivado debug feature is a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the debug feature for debugging the specific problems.

General Checks

- Ensure MIPI DPHY and MIPI CSI-2 RX Controller cores are in the enable state by reading the registers.
- Ensure Incorrect Lane Configuration is not set in the MIPI CSI-2 RX Controller Interrupt Status register.
- Ensure line buffer full condition is not set in the MIPI CSI-2 RX Controller Interrupt Status register.
- Ensure that the PULLUP constraints required for the AXI IIC core pins are set at the system-level XDC when the AXI IIC core is enabled. (See the *AXI IIC Bus Interface v2.0 LogiCORE IP Product Guide* (PG090) [Ref 5] for more information.)

Interface Debug

AXI4-Lite Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. See Figure A-1 for a read timing diagram. Output `s_axi_arready` asserts when the read address is valid, and output `s_axi_rvalid` asserts when the read data/response is valid. If the interface is unresponsive, ensure that the following conditions are met:

- The `lite_aclk` inputs are connected and toggling.
- The interface is not being held in reset, and `lite_aresetn` is an active-Low reset.
- The main subsystem clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or a debug feature capture that the waveform is correct for accessing the AXI4-Lite interface.

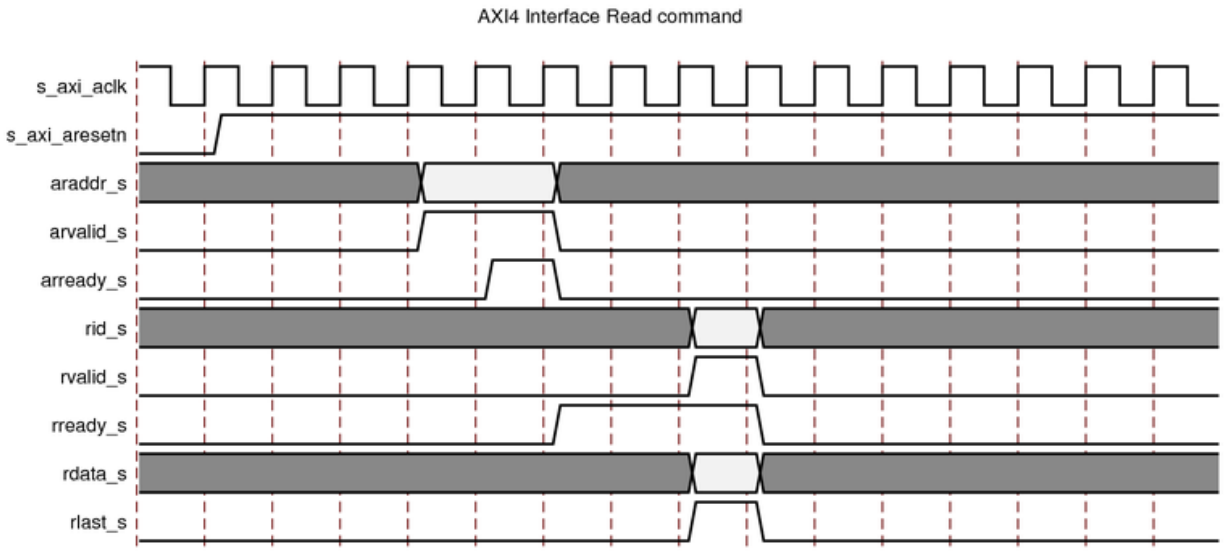

Figure A-1: **AXI4-Lite Timing**

AXI4-Stream Interfaces

If data is not being transmitted or received, check the following conditions:

- If transmit `<interface_name>_tready` is stuck Low following the `<interface_name>_tvalid` input being asserted, the subsystem cannot send data.
- If the receive `<interface_name>_tvalid` is stuck Low, the subsystem is not receiving data.
- Check that the `video_aclk` and `dphy_clk_200M` inputs are connected and toggling.
- Check subsystem configuration.
- Ensure "Stream line buffer full" condition not getting reported in subsystem Interrupt Status register

Application Software Development

Software driver information is not currently available.

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

References

These documents provide supplemental material useful with this product guide:

1. MIPI Alliance Standard for Camera Serial Interface CSI-2: mipi.org/specifications/camera-interface#CSI2
2. *AXI4-Stream Video IP and System Design Guide* (UG934)
3. *MIPI D-PHY v1.0 LogiCORE IP Product Guide (*PG202*)*
4. *AXI Interconnect LogiCORE IP* Product Guide (PG059)
5. *AXI IIC Bus Interface v2.0 LogiCORE IP Product Guide* (PG090)
6. MIPI Alliance Physical Layer Specifications, D-PHY Specification: http://mipi.org/specifications/physical-layer#D-PHY Specification
7. *Vivado Design Suite: AXI Reference Guide* (UG1037)
8. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
9. *Vivado Design Suite User Guide: Designing with IP* (UG896)
10. *Vivado Design Suite User Guide: Getting Started* (UG910)
11. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
12. *ISE to Vivado Design Suite Migration Guide* (UG911)
13. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
14. *Vivado Design Suite User Guide: Implementation* (UG904)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/18/2015	1.0	Initial Xilinx release.
09/30/2015	Early Access Draft	Xilinx Confidential Draft. Approved for external release under NDA only.

Please Read: Important Legal Notices