MIPI D-PHY v4.2

LogiCORE IP Product Guide

Vivado Design Suite

PG202 (v4.2) September 7, 2020

Table of Contents

Chapter 1: Introduction **4**
- Features 4
- IP Facts 5

Chapter 2: Overview **6**
- Navigating Content by Design Process 6
- Feature Summary 7
- Applications 7
- Unsupported Features 7
- Licensing and Ordering 8

Chapter 3: Product Specification **9**
- Standards 9
- MIPI D-PHY TX (Master) Core Architecture 9
- MIPI D-PHY RX (Slave) Core Architecture 11
- MIPI D-PHY Splitter Bridge Mode 13
- Performance and Resource Use 14
- Port Descriptions 16
- Register Space 26

Chapter 4: Designing with the Core **32**
- General Design Gudelines 32
- Shared Logic 33
- I/O Planning for UltraScale+ Devices 38
- I/O Planning for Versal Devices 39
- Clocking 41
- Resets 46
- Protocol Description 49

Chapter 5: Design Flow Steps **59**
- Customizing and Generating the Core 59
- Constraining the Core 67

Simulation 69
Synthesis and Implementation 69

Chapter 6: Example Design 70
Overview 70
Simulating the Example Design 71

Chapter 7: Test Bench 72

Appendix A: Verification, Compliance, and Interoperability 74
Hardware Validation 74

Appendix B: Debugging 76
Finding Help on Xilinx.com 76
Debug Tools 77
Simulation Debug 78
Hardware Debug 79
AXI4-Lite Interface Debug 81

Appendix C: Pin and Bank Rules 82
Pin Rules for Zynq UltraScale+ MPSoC Devices 82
Pin Rules for 7 series FPGAs 89

Appendix D: Additional Resources and Legal Notices 90
Xilinx Resources 90
Documentation Navigator and Design Hubs 90
References 90
Revision History 91
Please Read: Important Legal Notices 93

Introduction

The Xilinx® MIPI D-PHY Controller is designed for transmission and reception of video or pixel data for camera and display interfaces. The core is used as the physical layer for higher level protocols such as the Mobile Industry Processor Interface (MIPI) Camera Serial Interface (CSI-2) and Display Serial Interface (DSI).

This product guide provides information about using, customizing, and simulating the core for UltraScale+ and 7 series FPGA families as well as Versal™ ACAP. It also describes the core architecture and provides details on customizing and interfacing to the core.

Features

- Compliant to MIPI Alliance Standard for D-PHY Specification, version 2.0.
- Synchronous transfer at high-speed mode with a bit rate of 80-2936 Mb/s depending on the device family and speed grade. For details about device family supported line rates see the *UltraScale Architecture SelectIO Resources User Guide* (UG571).
- One clock lane and up to four data lanes for TX configuration.
- One clock lane and up to eight data lanes for RX configuration.
- Asynchronous transfer at low-power mode with a bit rate of 10 Mb/s.
- Ultra low-power mode, and high-speed mode for clock lane.
- Ultra low-power mode, high-speed mode, and escape mode for data lane.
- PHY-Protocol Interface (PPI) to connect CSI-2 and DSI applications.
- Optional AXI4-Lite interface for register access.

IP Facts

LogiCORE™ IP Facts Table	
Core Specifics	
Supported Device Family[1]	Versal™ ACAP, UltraScale+™ Families, Zynq® UltraScale+™ MPSoC, Zynq®-7000 SoC, 7 series FPGAs
Supported User Interfaces	PPI, AXI4-Lite
Resources	Performance and Resource Use web page
Provided with Core	
Design Files	Encrypted RTL
Example Design	Verilog
Test Bench	Verilog
Constraints File	Xilinx Design Contraints (XDC)
Simulation Model	Not Provided
Supported S/W Driver	N/A
Tested Design Flows[2]	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Release Notes and Known Issues	Master Answer Records: 54550
All Vivado IP Change Logs	Master Vivado IP Change Logs: 72775
Xilinx Support web page	

Notes:

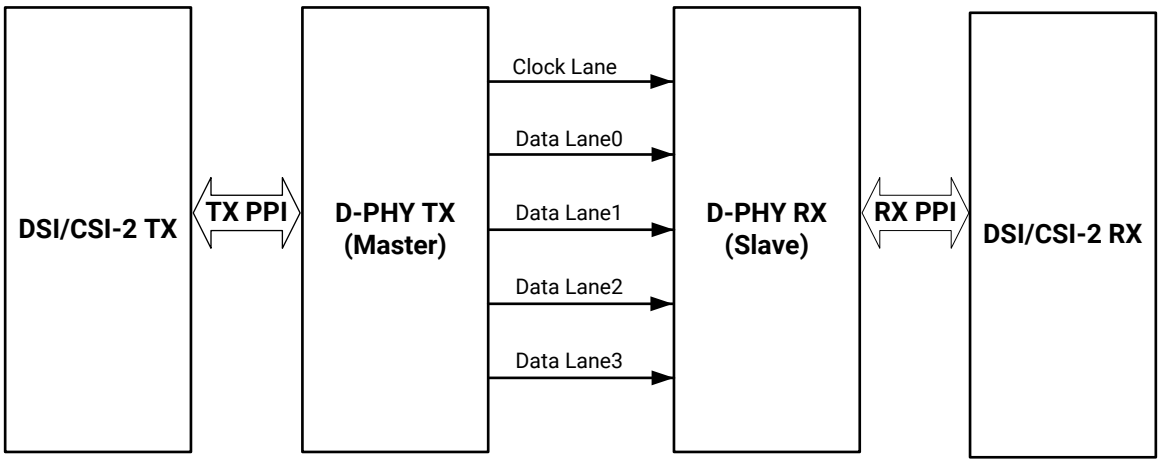
1. For a complete list of supported devices, see the Vivado® IP catalog.
2. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The MIPI D-PHY Controller is a full-featured IP core, incorporating all the necessary logic to properly communicate on this high-speed I/O interface standard. The core supports transmission/reception of camera sensor and video data from/to a standard-format PHY-Protocol Interface (PPI) using the high-speed SelectIO™ interface.

The following figure shows a high-level view of the MIPI D-PHY with all its components:

Figure 1: **D-PHY IP Overview**



Navigating Content by Design Process

Xilinx® documentation is organized around a set of standard design processes to help you find relevant content for your current development task. This document covers the following design processes:

- **Hardware, IP, and Platform Development:** Creating the PL IP blocks for the hardware platform, creating PL kernels, subsystem functional simulation, and evaluating the Vivado® timing, resource use, and power closure. Also involves developing the hardware platform for system integration. Topics in this document that apply to this design process include:
  - Port Descriptions

- Register Space
- Clocking
- Resets
- Customizing and Generating the Core
- Chapter 6: Example Design

Feature Summary

The MIPI D-PHY Controller can be configured as a Master (TX) or Slave (RX). It supports high-speed data transfer up to 2936 Mb/s, and control data can be transferred using Low-Power Data Transfer mode at 10 Mb/s.. The PPI interface allows a seamless interface to DSI and/or CSI IP cores. Using the MIPI D-PHY core Vivado® Integrated Design Environment (IDE)-based I/O planner, you can customize the data lane(s) selection by selecting the I/O bank followed by the clock lane. Optionally, the MIPI D-PHY core provides an AXI4-Lite interface to update the protocol timer values and retrieve the core status for debugging purposes.

Applications

The MIPI D-PHY Controller can be used to interface with the MIPI CSI-2 and DSI controller TX/RX devices. This core allows for seamless integration with higher level protocol layers through the PPI.

Unsupported Features

The following features of the standard are not supported in the MIPI D-PHY Controller:

- Link turnaround (reverse data communication)
- Low-power contention detection
- 8B9B encoding
- Dynamic line rate change

Licensing and Ordering

This Xilinx® LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado® Design Suite under the terms of the Xilinx End User License.

***Note*:** To verify that you need a license, check the License column of the IP Catalog. Included means that a license is included with the Vivado® Design Suite; Purchase means that you have to purchase a license to use the core.

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

Product Specification

The MIPI D-PHY Controller is a physical layer that supports the MIPI CSI-2 and DSI protocols. It is a universal PHY that can be configured as either a transmitter or a receiver. The core consists of an analog front end to generate and receive the electrical level signals, and a digital backend to control the I/O functions.

The MIPI D-PHY Controller provides a point-to-point connection between master and slave, or host and device that comply with a relevant MIPI standard. A typical TX configuration consists of 1 clock lane and 1 to 4 data lanes and a typical RX configuration consists of 1 clock lane and 1 to 8 data lanes. The master/host is primarily the source of data, and the slave/device is usually the sink of data. The D-PHY lanes can be configured for unidirectional lane operation, originating at the master and terminating at the slave. The core can be configured to operate as a master or as a slave. The D-PHY link supports a high-speed (HS) mode for fast data traffic and a low-power (LP) mode for control transactions.

- In HS mode, the low-swing differential signal supports data transfers from 80 Mb/s to 2936 Mb/s.
- In LP mode, all wires operate as a single-ended line capable of supporting 10 Mb/s asynchronous data communications.

Standards

This core is designed to be compatible with the MIPI Alliance D-PHY Specification. For a list of supported devices, see the Vivado® IP catalog.

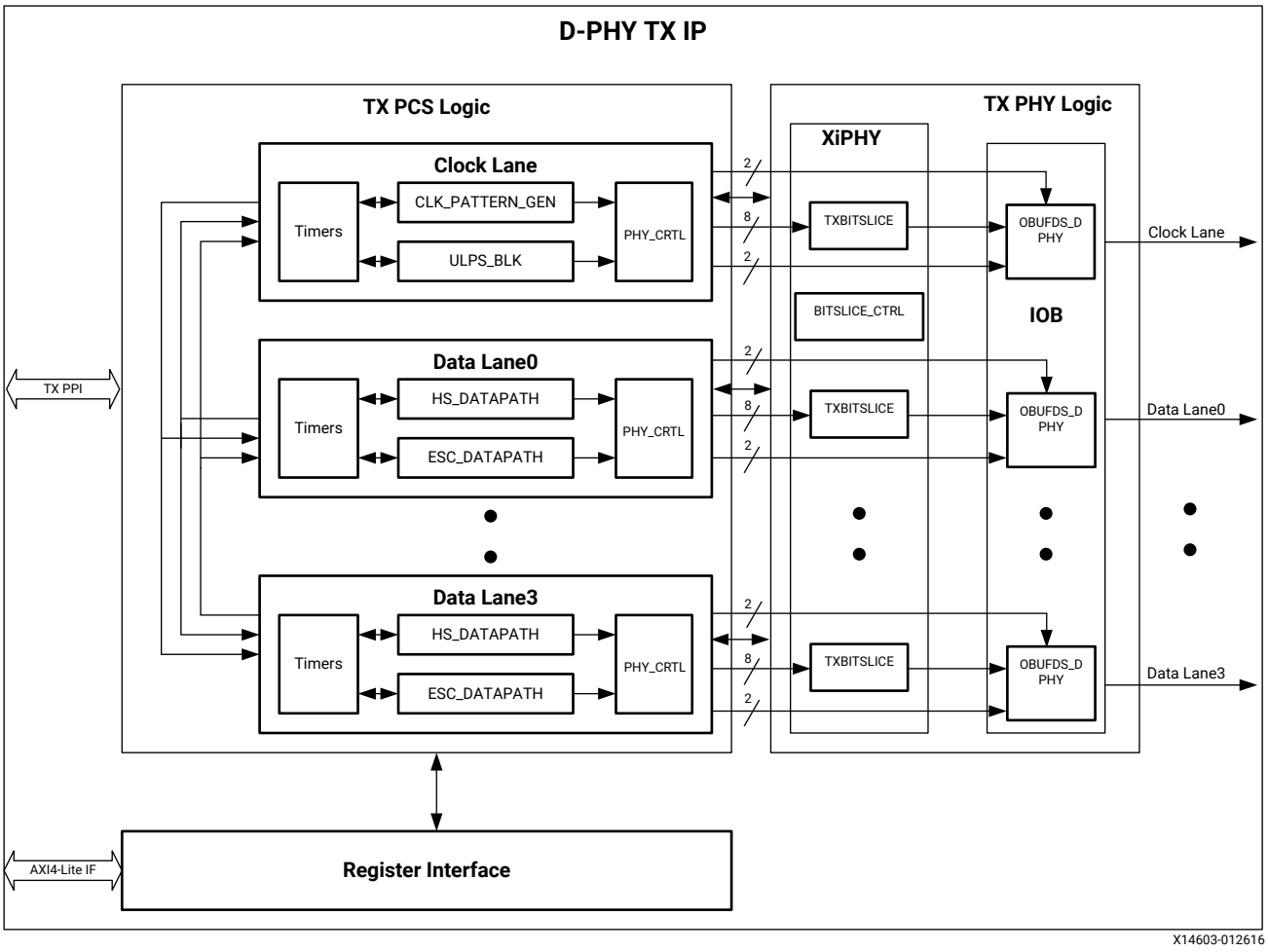
MIPI D-PHY TX (Master) Core Architecture

The following figure shows the MIPI D-PHY TX (Master) core architecture for UltraScale+™ families and Zynq® UltraScale+™ MPSoC devices. The TX core is partitioned into three major blocks:

- **TX Physical Coding Sublayer (PCS) Logic:** Provides the PPI to the core and generates the necessary controls to the PHY for the lane operation. It also generates entry sequences, line switching between low power and high speed, and performs lane initialization.

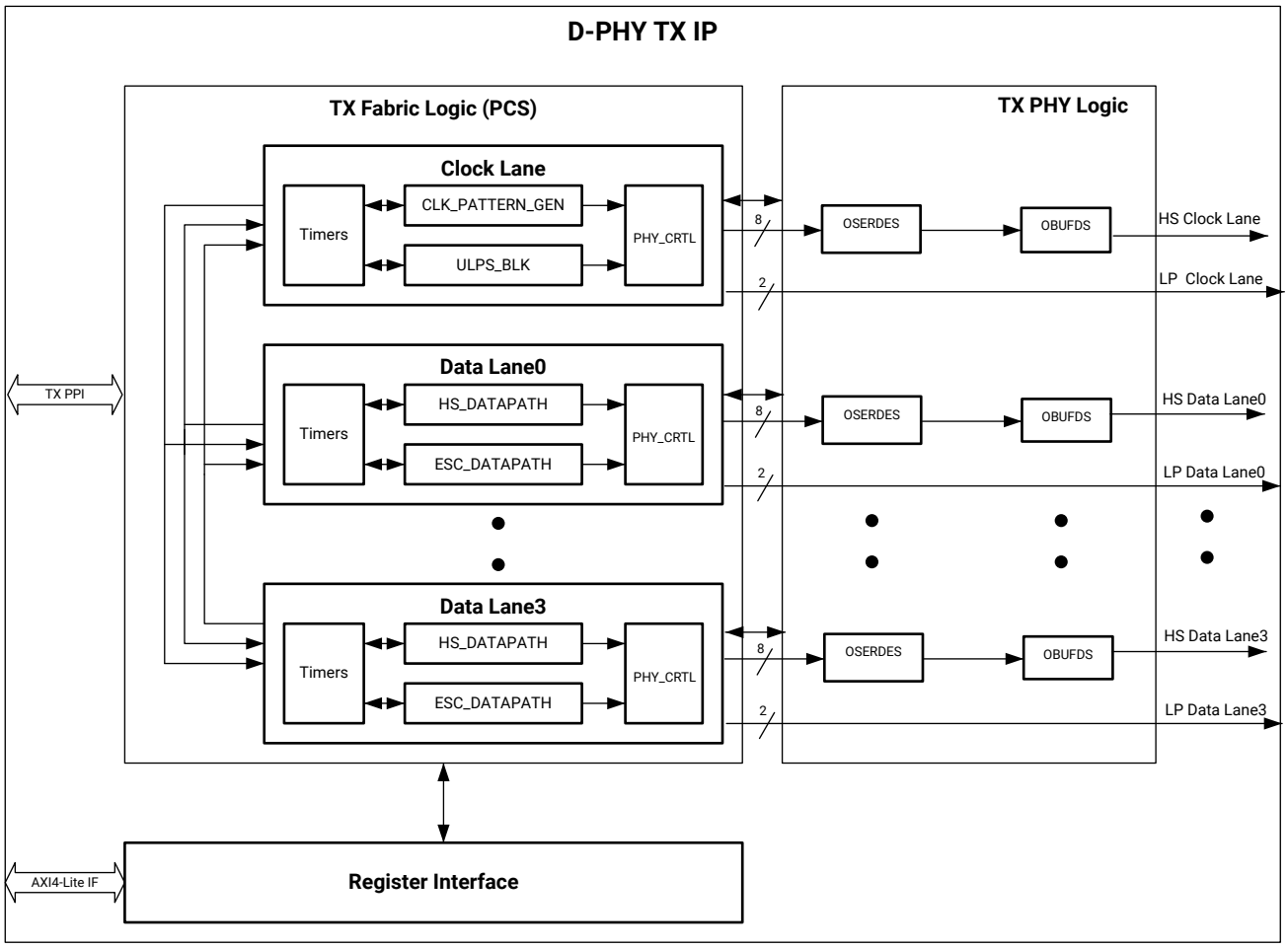
- **TX PHY Logic:** Integrates the BITSLICE_CONTROL and TX_BITSLICE in native mode and D-PHY-compatible I/O block. This block does serialization and has clocking implementation for the PHY.
- **Register Interface:** Optional AXI4-Lite register interface to control mandatory protocol timers and registers.

Figure 3: **MIPI D-PHY TX (Master) Core Architecture for UltraScale+ Families**



The following figure shows the MIPI D-PHY TX (Master) Core Architecture for the 7 series FPGA families.

Figure 4: **MIPI D-PHY TX (Master) Core Architecture for 7 Series FPGA Families**



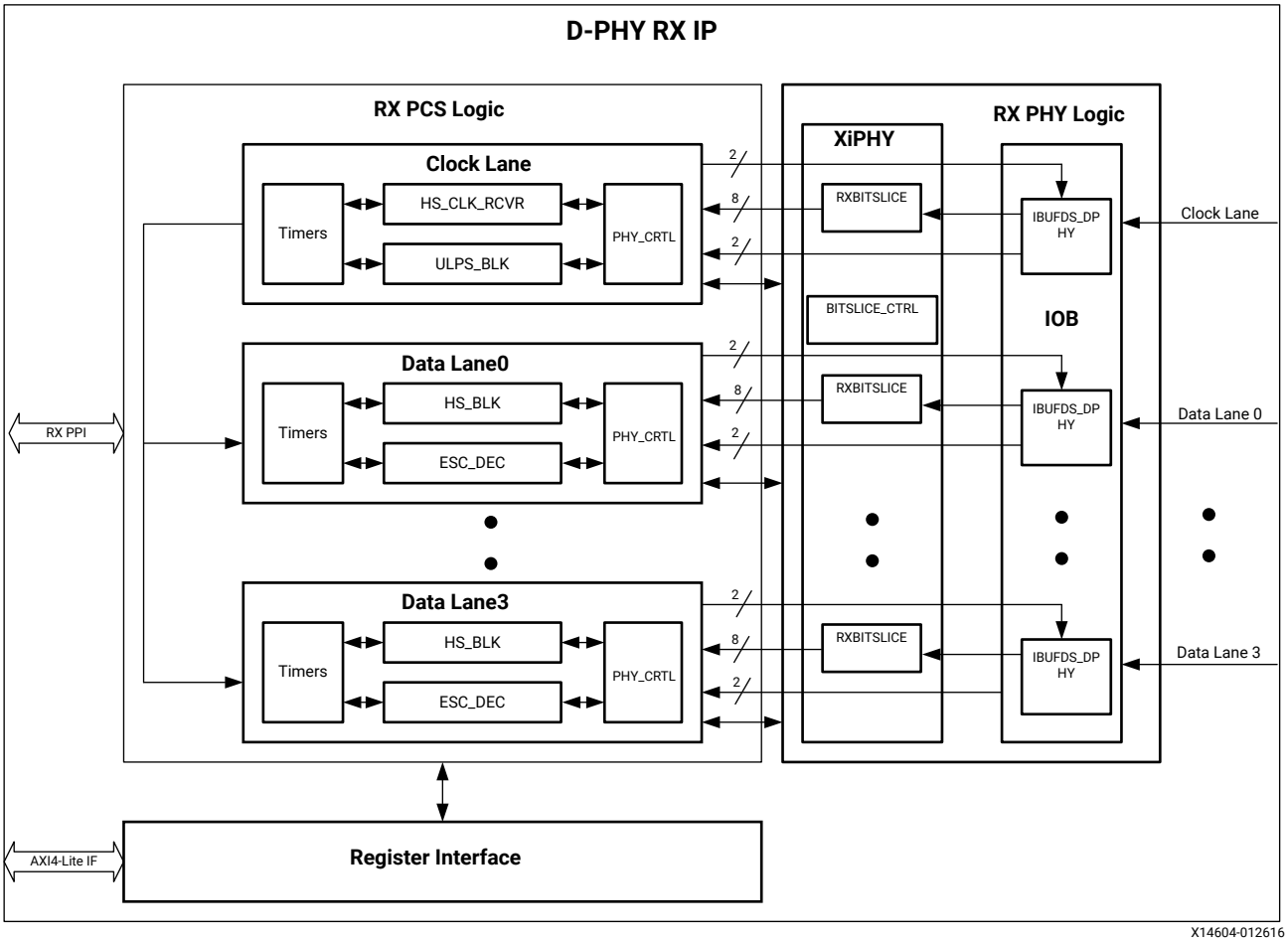
MIPI D-PHY RX (Slave) Core Architecture

The following figure shows the MIPI D-PHY RX (Slave) core architecture for UltraScale+™ families and Zynq® UltraScale+™ MPSoC devices. The RX core is partitioned into three major blocks:

- **RX PCS Logic:** Interfaces with PHY and delivers PHY-Protocol Interface (PPI)-compliant transactions such as High-Speed and Escape mode Low-Power Data Transmission (LPDT) packets. It is also responsible for lane initialization, start-of-transmission (SoT) detection, and clock recovery in escape mode.

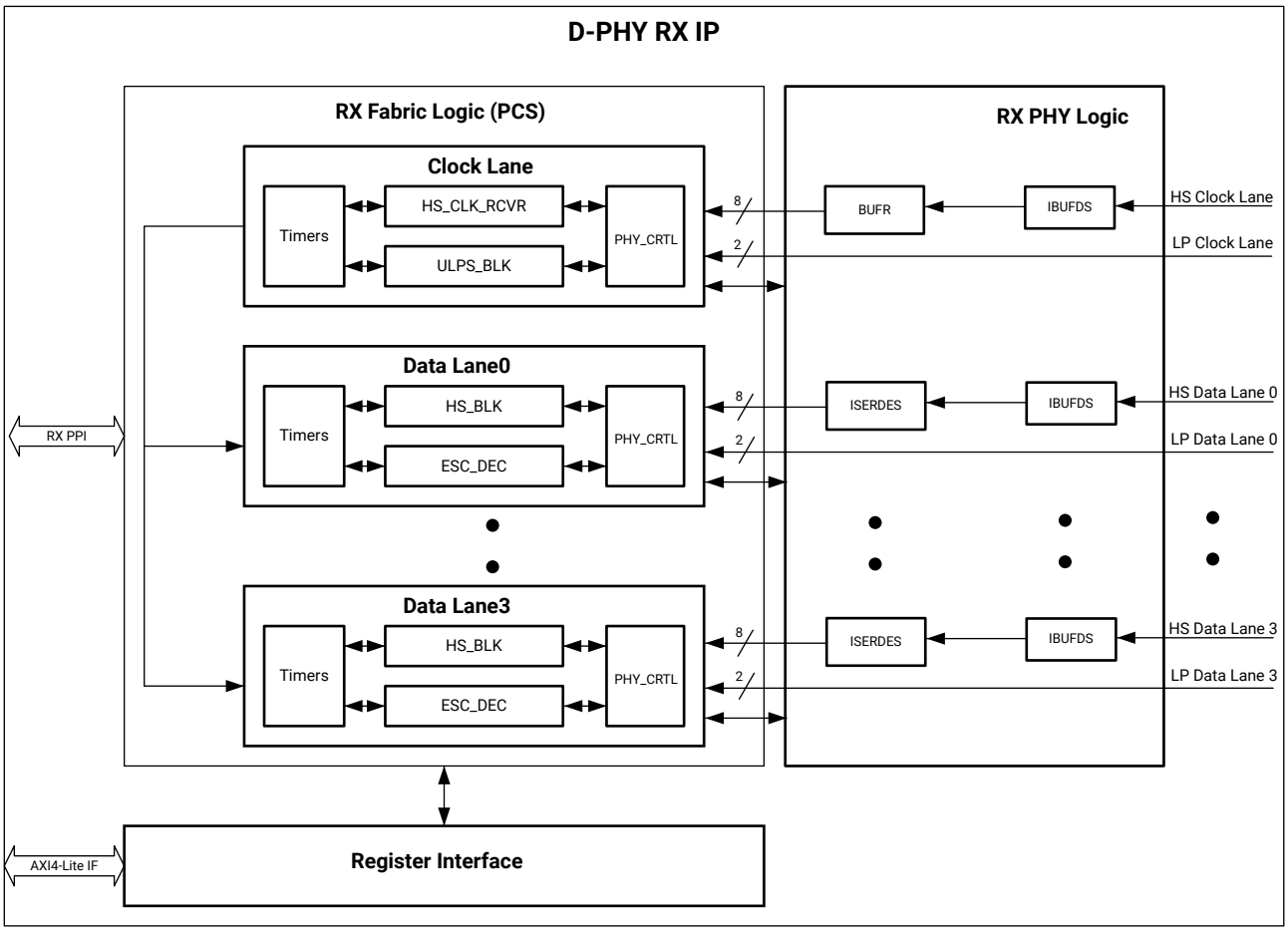
- **RX PHY Logic:** Performs clock recovery in high-speed mode and de-serialization. Integrates the BITSLICE_CONTROL and RX_BITSLICE in native mode and D-PHY compatible I/O block.
- **Register Interface:** Optional AXI4-Lite register interface to control protocol mandatory timers and registers.

Figure 5: **MIPI D-PHY RX (Slave) Core Architecture for UltraScale+ Families**



The following figure shows the MIPI D-PHY RX (Slave) Core Architecture for the 7 series FPGA families.

Figure 6: **MIPI D-PHY RX (Slave) Core Architecture for 7 Series FPGA Families**

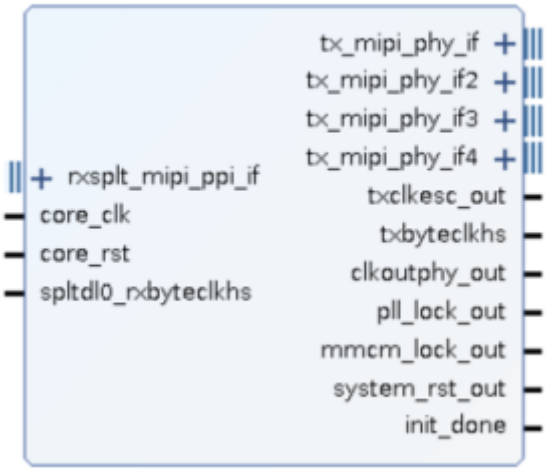


X17793-062320

MIPI D-PHY Splitter Bridge Mode

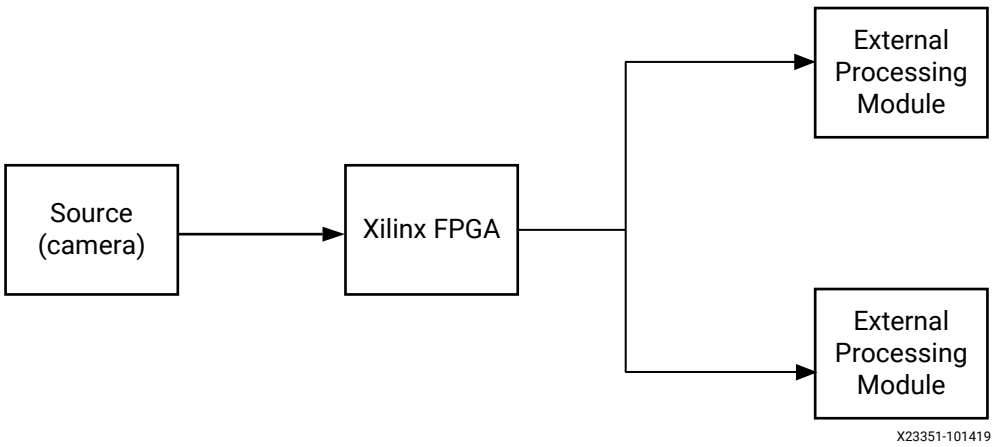
Enabling this mode allows the received PPI RX input data to be sent as MIP TX Data duplicated on multiple DPHY TX Interfaces. You can select up to a maximum of 4 TX Interfaces as shown in the following figure. GUI allows to select IO for each interface. You need to ensure that the IOs are exclusive across interfaces. The IOs of each interface can be same or different banks.

Figure 7: **MIPI D-PHY – D-PHY Splitter Bridge**



This mode is best suited for cases where same camera data need to be processed by multiple external processing modules. In such cases, Xilinx FPGA receives MIPI stream from external source (camera) and replicates on multiple output MIPI stream interfaces for further processing by external modules.

Figure 8: **D-PHY Splitter Bridge use case**



Performance and Resource Use

For full details about performance and resource use, visit the Performance and Resource Use web page.

Maximum Frequencies

The maximum frequency of the core operation is dependent on the supported line rates and the speed grade of the devices.

Latency

The MIPI D-PHY TX core latency is measured from the `requesths` signal of the data lane assertion to the `readyhs` signal assertion.

The MIPI D-PHY RX core latency is the time from the start-of-transmission (SoT) pattern on the serial lines to the `activehs` signal assertion on the PPI. The following table provides the latency numbers for various core configurations.

***Note*:** To calculate the throughput for higher lanes, multiply the existing throughput by the configured number of lanes.

Table 1: **Latency for D-PHY Core Configurations**

Line Rate (Mb/s)	LPX (ns)	Device Family	Lanes	Latency (in byteclkhs[1] cycles)	Data Flow Mode
250	50	UltraScale+	1	10	D-PHY TX (Master)
500	50	UltraScale+	1	18	D-PHY TX (Master)
1,000	50	UltraScale+	1	33	D-PHY TX (Master)
1,250	50	UltraScale+	1	43	D-PHY TX (Master)
1,500	50	UltraScale+	1	51	D-PHY TX (Master)
2,000	50	UltraScale+	1	67	D-PHY TX (Master)
2,500	50	UltraScale+	1	84	D-PHY TX (Master)
250	50	UltraScale+	1	6	D-PHY RX (Slave)
500	50	UltraScale+	1	6	D-PHY RX (Slave)
1,000	50	UltraScale+	1	6	D-PHY RX (Slave)
1,250	50	UltraScale+	1	6	D-PHY RX (Slave)
1,500	50	UltraScale+	1	6	D-PHY RX (Slave)
2,000	50	UltraScale+	1	6	D-PHY RX (Slave)
2,500	50	UltraScale+	1	6	D-PHY RX (Slave)
250	50	7 series	1	16	D-PHY TX (Master)
500	50	7 series	1	24	D-PHY TX (Master)
1,000	50	7 series	1	39	D-PHY TX (Master)
1,250	50	7 series	1	48	D-PHY TX (Master)
250	50	7 series	1	5	D-PHY RX (Slave)
500	50	7 series	1	5	D-PHY RX (Slave)
1,000	50	7 series	1	5	D-PHY RX (Slave)

Table 1: **Latency for D-PHY Core Configurations** *(cont'd)*

Line Rate (Mb/s)	LPX (ns)	Device Family	Lanes	Latency (in byteclkhs[1] cycles)	Data Flow Mode
1,250	50	7 series	1	5	D-PHY RX (Slave)

Notes:

1. Frequency of byteclkhs (MHz) = line rate in Mb/s divided by 8.
2. Latency is dependent on line rate, LPX period , `HSPREPARE` time, and `HSZERO` time.

Throughput

The MIPI D-PHY TX core throughput varies based on line rate, number of data lanes, clock lane mode (continuous or non-continuous) and D-PHY protocol parameters. Throughput is measured from the clock lane `txrequesths` signal assertion to the clock lane `txrequesths` signal deassertion by transferring a standard 640x480 resolution image as frame data on the PPI. In this measurement, the number of bytes transfered from the start to the end are taken into account. Data lane `txrequesths` and `txreadyhs` assertion is considered as one-byte transfer. The following table provides the throughput numbers for various core configurations.

Table 2: **Throughput for MIPI D-PHY TX Core Configurations**

Line Rate (Mb/s)	LPX (ns)	Device Family	Lanes	Throughput (Mb/s)	Data Flow Mode
250	50	UltraScale+	1	239	D-PHY TX (Master)
500	50	UltraScale+	1	462	D-PHY TX (Master)
1,000	50	UltraScale+	1	879	D-PHY TX (Master)
1,250	50	UltraScale+	1	1075	D-PHY TX (Master)
1,500	50	UltraScale+	1	1261	D-PHY TX (Master)
2000	50	UltraScale+	1	1661	D-PHY TX (Master)
2500	50	UltraScale+	1	2002	D-PHY TX (Master)
2936	50	Versal™ ACAP	1	2212	D-PHY TX (Master)
250	50	7 series	1	231	D-PHY TX (Master)
500	50	7 series	1	462	D-PHY TX (Master)
1,000	50	7 series	1	879	D-PHY TX (Master)
1,250	50	7 series	1	1066	D-PHY TX (Master)

Port Descriptions

The external interface of the core is PPI, and the AXI4-Lite interface is optionally available for register programming.

PPI Signals

The MIPI D-PHY core provides PPI signaling for clock lane and data lane operation. The signal ports are listed in the following tables. In these tables <n> is the configurable data lane number (0 to 3).

Table 3: **Common PPI Control Signals**

Signal	Direction	Clock Domain	Description
cl_ stopstate, dl<n>_stopstate	Output	Async	Lane is in Stop state. This active-High signal indicates that the Lane module (TX or RX) is currently in the Stop state. Also, the protocol can use this signal to indirectly determine if the PHY line levels are in the LP-11 state. ***Note*:** This signal is asynchronous to any clock in the PPI.
cl_enable, dl<n>_enable	Input	Async	Enable Lane Module. This active-High signal forces the lane module out of “shutdown”. All line drivers, receivers, terminators, and contention detectors are turned off when Enable is Low. When Enable is Low, all other PPI inputs are ignored and all PPI outputs are driven to the default inactive state. Enable is level sensitive and does not depend on any clock.
cl_ulpsactivenot, dl<n>_ulpsactivenot	Output	Async	ULP State (not) Active. This active-Low signal is asserted to indicate that the Lane is in the ULP state. For a receiver, this signal indicates that the Lane is in the Ultra Low Power (ULP) state. At the beginning of the ULP state, ulpsactivenot is asserted together with rxulpsesc, or rxclkulpsnot for a clock lane. At the end of the ULP state, this signal becomes inactive to indicate that the Mark-1 state has been observed. Later, after a period of time (Twakeup), the rxulpsesc (or rxclkulpsnot) signal is deasserted.

Table 4: **D-PHY TX Clock Lane High-Speed PPI Signal**

Signal	Direction	Clock Domain	Description
cl_txrequesths	Input	txbyteclkhs	High-Speed Transmit Request and Data Valid. For clock lanes, this active-High signal causes the lane module to begin transmitting a high-speed clock.
cl_txclkactivehs	Output	txbyteclkhs	This active-High signal indicates that the clock is being transmitted on the clock lane.

Table 5: **D-PHY TX Clock Lane Escape Mode PPI Signals**

Signal	Direction	Clock Domain	Description
cl_txulpsclk	Input	core_clk	Transmit Ultra-Low Power State on Clock Lane. This active-High signal is asserted to cause a clock lane module to enter the ULP state. The lane module remains in this mode until txulpsclk is deasserted.

Table 5: **D-PHY TX Clock Lane Escape Mode PPI Signals** *(cont'd)*

Signal	Direction	Clock Domain	Description
cl_txulpsexit	Input	core_clk	Transmit ULP Exit Sequence. This active-High signal is asserted when the ULP state is active and the protocol is ready to leave the ULP state. The PHY leaves the ULP state and begins driving Mark-1 after txulpsexit is asserted. The PHY later drives the Stop state (LP-11) when txrequestesc is deasserted. txulpsexit is synchronous to txclkesc. This signal is ignored when the lane is not in the ULP state.

Table 6: **D-PHY TX Data Lane High-Speed PPI Signals**

Signal	Direction	Clock Domain	Description
txbyteclkhs	Output	N/A	High-Speed Transmit Byte Clock. This is used to synchronize PPI signals in the high-speed transmit clock domain. Xilinx recommends that all transmitting data lane modules share one txbyteclkhs signal. The frequency of txbyteclkhs is exactly 1/8 the high-speed bit rate.
dl<n>_txdatahs[7:0]	Input	txbyteclkhs	High-Speed Transmit Data. Eight-bit high-speed data to be transmitted. The signal connected to txdatahs[0] is transmitted first. Data is captured on rising edges of txbyteclkhs.
dl<n>_txrequesths	Input	txbyteclkhs	High-Speed Transmit Request and Data Valid. A Low-to-High transition on txrequesths causes the Lane module to initiate a SoT sequence. A High-to-Low transition on txrequest causes the lane module to initiate an EoT sequence. For data lanes, this active-High signal also indicates that the protocol is driving valid data on txdatahs to be transmitted. The lane module accepts the data when both txrequesths and txreadyhs are active on the same rising txbyteclkhs clock edge. The protocol always provides valid transmit data when txrequesths is active. After asserted, txrequesths remains High until the data has been accepted, as indicated by txreadyhs. txrequesths is only asserted while txrequestesc is Low.
dl<n>_txreadyhs	Output	txbyteclkhs	High-Speed Transmit Ready. This active-High signal indicates that txdatahs[7:0] is accepted by the Lane module to be serially transmitted. txreadyhs is valid on rising edges of txbyteclkhs.
dl<n>_txskewcalhs	Input	txbyteclkhs	High-Speed Transmit Skew Calibration. A low-to-high transition on TxSkewCalHS causes the lane module to initiate a deskew calibration. A high-to-low transition on TxSkewCalHS causes the lane module to stop deskew pattern transmission and initiate an EoT sequence

Table 7: **D-PHY TX Data Lane Control Interface PPI Signal**

Signal	Direction	Clock Domain	Description
dl<n>_forcetxstopmode	Input	Aync	Force Lane to Generate Stop State. This signal allows the protocol to force a lane module into the Stop state during initialization or following an error situation, such as an expired timeout. When this signal is High, the lane module state machine is immediately forced into the Stop state.

Table 8: **D-PHY TX Data Lane Escape Mode PPI Signals**

Signal	Direction	Clock Domain	Description
txclkesc	Input	N/A	Escape Mode Transmit Clock. This clock is directly used to generate escape sequences. The period of this clock determines the phase times for low-power signals as defined in the D-PHY specification.
dl<n>_txrequestesc	Input	txclkesc	Escape Mode Transmit Request. This active-High signal, asserted together with exactly one of txlpdtesc, txulpsesc, or one bit of txtriggeresc, is used to request entry into escape mode. When in escape mode, the lane stays in escape mode until txrequestesc is deasserted. txrequestesc is only asserted by the protocol while txrequesths is Low. txrequesths has highest priority than txrequestesc.
dl<n>_txlpdtesc	Input	txclkesc	Escape Mode Transmit Low-Power Data. This active-High signal is asserted with txrequestesc to cause the lane module to enter low-power data transmission mode. The Lane module remains in this mode until txrequestesc is deasserted. txulpsesc and all bits of txtriggeresc[3:0] are Low when txlpdtesc is asserted.
dl<n>_txulpsexit	Input	txclkesc	Transmit ULP Exit Sequence. This active-High signal is asserted when the ULP state is active and the protocol is ready to leave the ULP state. The PHY leaves the ULP state and begins driving Mark-1 after txulpsexit is asserted. The PHY later drives the Stop state (LP-11) when txrequestesc is deasserted. txulpsexit is synchronous to txclkesc. This signal is ignored when the lane is not in the ULP state.
dl<n>_txulpsesc	Input	txclkesc	Escape Mode Transmit Ultra-Low Power State. This active-High signal is asserted with txrequestesc to cause the lane module to enter the ultra-low power state. The lane module remains in this mode until txrequestesc is deasserted. txlpdtesc and all bits of txtriggeresc[3:0] are Low when txulpsesc is asserted.

Table 8: **D-PHY TX Data Lane Escape Mode PPI Signals** *(cont'd)*

Signal	Direction	Clock Domain	Description
dl<n>_txtriggeresc[3:0]	Input	txclkesc	Escape Mode Transmit Trigger 0-3. One of these active-High signals is asserted with txrequestesc to cause the associated trigger to be sent across the lane interconnect. In the receiving lane module, the same bit of rxtriggeresc is then asserted and remains asserted until the lane interconnect returns to the Stop state, which happens when txrequestesc is deasserted at the transmitter. Only one bit of txtriggeresc[3:0] is asserted at any given time, and only when txlpdtesc and txulpsesc are both Low. The following mapping is done by the D-PHY TX module: • Reset-Trigger → txtriggeresc[3:0] = 4'b0001 • Unknown-3 → txtriggeresc[3:0] = 4'b0010 • Unknown-4 → txtriggeresc[3:0] = 4'b0100 • Unknown-5 → txtriggeresc[3:0] = 4'b1000
dl<n>_txdataesc[7:0]	Input	txclkesc	Escape Mode Transmit Data. This is the eight-bit Escape mode data to be transmitted in low-power data transmission mode. The signal connected to txdataesc[0] is transmitted first. Data is captured on rising edges of txclkesc.
dl<n>_txvalidesc	Input	txclkesc	Escape Mode Transmit Data Valid. This active-High signal indicates that the protocol is driving valid data on txdataesc[7:0] to be transmitted. The lane module accepts the data when txrequestesc, txvalidesc, and txreadyesc are all active on the same rising txclkesc clock edge.
dl<n>_txreadyesc	Output	txclkesc	Escape Mode Transmit Ready. This active-High signal indicates that txdataesc[7:0] is accepted by the lane module to be serially transmitted. txreadyesc is valid on rising edges of txclkesc.

Table 9: **D-PHY RX Clock Lane PPI Signals**

Signal	Direction	Clock Domain	Description
cl_rxclkactivehs	Output	Async	Receiver Clock Active. This asynchronous, active-High signal indicates that a clock lane is receiving a Double Data Rate (DDR) clock signal.
cl_rxulpsclknot	Output	Asynch	Receiver Ultra-Low Power State on Clock Lane. This active-Low signal is asserted to indicate that the clock lane module has entered the ultra-low power state. The lane module remains in this mode with rxulpsclknot asserted until a Stop state is detected on the lane interconnect.

Table 10: **D-PHY RX Data Lane High-Speed PPI Signals**

Signal	Direction	Clock Domain	Description
rxbyteclkhs	Output	N/A	High-Speed Receive Byte Clock. This is used to synchronize signals in the high-speed receive clock domain. The rxbyteclkhs is generated by dividing the received High-Speed DDR clock. ***Note*:** This clock is not continuous and is only available for sampling when the RX clock lane is in high-speed mode.
dl<n>_rxdatahs[7:0]	Output	rxbyteclkhs	High-Speed Receive Data. Eight-bit high-speed data received by the lane module. The signal connected to rxdatahs[0] was received first. Data is transferred on rising edges of rxbyteclkhs.
dl<n>_rxvalidhs	Output	rxbyteclkhs	High-Speed Receive Data Valid. This active-High signal indicates that the lane module is driving data to the protocol on the rxdatahs[7:0] output. There is no rxreadyhs signal, and the protocol is expected to capture rxdatahs[7:0] on every rising edge of rxbyteclkhs where rxvalidhs is asserted. There is no provision for the protocol to slow down (throttle) the receive data.
dl<n>_rxactivehs	Output	rxbyteclkhs	High-Speed Reception Active. This active-High signal indicates that the lane module is actively receiving a high-speed transmission from the lane interconnect.
dl<n>_rxsynchs	Output	rxbyteclkhs	Receiver Synchronization Observed. This active-High signal indicates that the Lane module has seen an appropriate synchronization event. rxsynchs is High for one cycle of rxbyteclkhs at the beginning of a high-speed transmission when rxactivehs is first asserted.
dl<n>_rxskewcalhs	Output	rxbyteclkhs	High-Speed Receive Skew Calibration. This active-High signal indicates that the high speed deskew burst is being received. ***Note*:** This pin is only available for line rate >1500 Mb/s configuration.

Table 11: **D-PHY RX Data Lane PPI Control Interface Signal**

Signal	Direction	Clock Domain	Description
dl<n>_forcerxmode	Input	Async	Force Lane Module to Re-Initialization. This signal allows the protocol to initialize a Lane module and should be released, that is, driven Low, only when the Dp and Dn inputs are in the Stop state for a time T_INIT, or longer. ***Note*:** Assert this signal when the RX Data Lane is in stopstate. Asserting this signal in the middle of High-Speed data reception will result in data integrity failures.

Table 12: **D-PHY RX Data Lane Escape Mode PPI Signals**

Signal	Direction	Clock Domain	Description
dl<n>_rxclkesc	Output	N/A	Escape Mode Receive Clock. This signal is used to transfer received data to the protocol during escape mode. This clock is generated from the two low-power signals in the lane interconnect. Because of the asynchronous nature of escape mode data transmission, this clock cannot be periodic.
dl<n>_rxlpdtesc	Output	rxclkesc	Escape Low-Power Data Receive Mode. This active-High signal is asserted to indicate that the lane module is in low-power data receive mode. While in this mode, received data bytes are driven onto the rxdataesc[7:0] output when rxvalidesc is active. The lane module remains in this mode with rxlpdtesc asserted until a Stop state is detected on the lane interconnect.
dl<n>_rxulpsesc	Output	Async	Escape Ultra-Low Power (Receive) Mode. This active-High signal is asserted to indicate that the lane module has entered the ultra-low power state. The lane module remains in this mode with rxulpsesc asserted until a Stop state is detected on the lane interconnect.
dl<n>_rxtriggeresc[3:0]	Output	Async	Escape Mode Receive Trigger 0-3. These active-High signals indicate that a trigger event has been received. The asserted rxtriggeresc[3:0] signal remains active until a Stop state is detected on the lane interconnect. The following mapping is done by the D-PHY RX module: • Reset-Trigger → rxtriggeresc[3:0] = 4'b0001 • Unknown-3 → rxtriggeresc[3:0] = 4'b0010 • Unknown-4 → rxtriggeresc[3:0] = 4'b0100 • Unknown-5 → rxtriggeresc[3:0] = 4'b1000
dl<n>_rxdataesc[7:0]	Output	rxclkesc	Escape Mode Receive Data. This is the eight-bit escape mode low-power data received by the lane module. The signal connected to rxdataesc[0] is received first. Data is transferred on rising edges of rxclkesc.
dl<n>_rxvalidesc	Output	rxclkesc	Escape Mode Receive Data Valid. This active-High signal indicates that the lane module is driving valid data to the protocol on the rxdataesc[7:0] output. There is no rxreadyesc signal, and the protocol is expected to capture rxdataesc[7:0] on every rising edge of rxclkesc where rxvalidesc is asserted. There is no provision for the protocol to slow down (throttle) the receive data.

Table 13: **D-PHY RX Data Lane PPI Error Signals**

Signal	Direction	Clock Domain	Description
dl<n>_ errsoths	Output	rxbyteclkhs	Start-of-Transmission (SoT) Error. If the high-speed SoT leader sequence is corrupted, but in such a way that proper synchronization can still be achieved, this active-High signal is asserted for one cycle of rxbyteclkhs. This is considered to be a soft error in the leader sequence and confidence in the payload data is reduced.
dl<n>_errsotsynchs	Output	rxbyteclkhs	Start-of-Transmission Synchronization Error. If the high-speed SoT leader sequence is corrupted in a way that proper synchronization cannot be expected, this active-High signal is asserted for one cycle of rxbyteclkhs.
dl<n>_erresc	Output	Async	Escape Entry Error. If an unrecognized escape entry command is received, this active-High signal is asserted and remains asserted until the next change in line state.
dl<n>_errsyncesc	Output	Async	Low-Power Data Transmission Synchronization Error. If the number of bits received during a low-power data transmission is not a multiple of eight when the transmission ends, this active-High signal is asserted and remains asserted until the next change in line state.
dl<n>_errcontrol	Output	Async	Control Error. This active-High signal is asserted when an incorrect line state sequence is detected. For example, if a turn-around request or escape mode request is immediately followed by a Stop state instead of the required Bridge state, this signal is asserted and remains asserted until the next change in line state.

Clocking and Reset Signals

Included in the example design sources are circuits for clock and reset management. The following table shows the ports on the core that are associated with system clock and reset.

Table 14: **Clocking and Reset Signals**

Signal	Direction	Clock Domain	Description
core_clk	Input	N/A	A stable core clock used for control logic.
core_rst	Input	core_clk	An active-High reset signal.
system_rst_out	Output	core_clk	An active-High system reset output to be used by the example design level logic. This port is available when **Shared Logic is in the Core** is selected.
mmcm_lock_out	Output	Async	MMCM lock indication. This port is not available when shared logic in the core is selected in D-PHY TX Configuration.
pll_lock_out	Output	Async	PLL lock indication. This port is available when **Shared Logic is in the Core** is selected. This port is available for UltraScale+ families.
system_rst_in	Input	core_clk	System level reset. This port is available when **Shared Logic is in Example Design** is selected in D-PHY TX configuration.

Table 14: **Clocking and Reset Signals** *(cont'd)*

Signal	Direction	Clock Domain	Description
pll_lock_in	Input	Async	PLL lock indication, This port is available when **Shared Logic is in Example Design** is selected. This port is available for UltraScale+ families.
ssc_byteclkhs_in	Input	N/A	SSC enabled clock input when the example design is in the core and the line rate is greater than 2500 Mb/s.
spltdl<>_rxbyteclkhs	Input	N/A	Clock for input splitter interface. ***Note*:** This pin is only available when splitter bridge mode is enabled.
init_done	Output	core_clk	An active-High signal which indicates lane initialization is done.

I/O Interface Signals

The example design includes circuits for PHY management and D-PHY compatible I/O connectivity. The following table shows the core ports that are associated with the I/O interface.

Table 15: **D-PHY TX I/O Interface**

Signal	Direction	Clock Domain	Description
clk_txp	Output	N/A	Positive differential serial data output pin for clock lane. Available only for UltraScale+ families.
clk_txn	Output	N/A	Negative differential serial data output pin for clock lane. Available only for UltraScale+ families.
data_txp[<n-1>:0][1]	Output	N/A	Positive differential serial data output pin for data lane(s). Available only for UltraScale+ families.
data_txn[<n-1>:0][1]	Output	N/A	Negative differential serial data output pin for data lane(s). Available only for UltraScale+ families.
clk_hs_txp	Output	N/A	High-Speed positive differential serial data output pin for clock lane. Available only for 7 series FPGA families.
clk_hs_txn	Output	N/A	High-Speed negative differential serial data output pin for clock lane. Available only for 7 series FPGA families.
clk_lp_txp	Output	N/A	Low-Power positive serial data output pin for clock lane. Available only for 7 series FPGA families.
clk_lp_txn	Output	N/A	Low-Power negative serial data output pin for clock lane. Available only for 7 series FPGA families.
data_hs_txp[<n-1>:0][1]	Output	N/A	High-Speed positive differential serial data output pin for data lane(s). Available only for 7 series FPGA families.
data_hs_txn[<n-1>:0][1]	Output	N/A	High-Speed negative differential serial data output pin for data lane(s). Available only for 7 series FPGA families.
data_lp_txp[<n-1>:0][1]	Output	N/A	Low-Power positive serial data output pin for data lane(s). Available only for 7 series FPGA families.
data_lp_txn[<n-1>:0][1]	Output	N/A	Low-Power negative serial data output pin for data lane(s). Available only for 7 series FPGA families.

Notes:

1. <n> is the data lane number.

Table 16: **D-PHY RX I/O Interface**

Signal	Direction	Clock Domain	Description
clk_rxp	Input	N/A	Positive differential serial data input pin for clock lane. Available only for UltraScale+ families.
clk_rxn	Input	N/A	Negative differential serial data input pin for clock lane. Available only for UltraScale+ families.
data_rxp[<n-1>:0][1]	Input	N/A	Positive differential serial data input pin for data lane(s). Available only for UltraScale+ families.
data_rxn[<n-1>:0][1]	Input	N/A	Negative differential serial data input pin for data lane(s). Available only for UltraScale+ families.
bg<x>_pin<y>_nc	Input	N/A	Inferred bitslice ports. The core infers bitslice0 of a nibble for strobe propagation within the byte group; <x> indicates byte group (0,1,2,3); <y> indicates bitslice0 position (0 for the lower nibble, 6 for the upper nibble.) • RTL Design: There is no need to drive any data on these ports. • IP Integrator: These ports must be brought to the the top level of the design in order for the constraints to be properly applied. ***Note*:** Pins are available only for UltraScale+ families.
clk_hs_rxp	Input	N/A	High-Speed positive differential serial data input pin for clock lane. Available only for 7 series FPGA families.
clk_hs_rxn	Input	N/A	High-Speed negative differential serial data input pin for clock lane. Available only for 7 series FPGA families.
clk_lp_rxp	Input	N/A	Low-Power positive serial data input pin for clock lane. Available only for 7 series FPGA families.
clk_lp_rxn	Input	N/A	Low-Power negative serial data input pin for clock lane. Available only for 7 series FPGA families.
data_hs_rxp[<n-1>:0][1]	Input	N/A	High-Speed positive differential serial data input pin for data lane(s). Available only for 7 series FPGA families.
data_hs_rxn[<n-1>:0][1]	Input	N/A	High-Speed negative differential serial data input pin for data lane(s). Available only for 7 series FPGA families.
data_lp_rxp[<n-1>:0][1]	Input	N/A	Low-Power positive serial data input pin for data lane(s). Available only for 7 series FPGA families.
data_lp_rxn[<n-1>:0][1]	Input	N/A	Low-Power negative serial data input pin for data lane(s). Available only for 7 series FPGA families.

Notes:

1. <n> is the data lane number.

AXI4-Lite Interface Signals

The AXI4-Lite signals (`s_axi_*`) are described in the *Vivado Design Suite: AXI Reference Guide* (UG1037).

7 Series FPGA Families Calibration Logic Signals

D-PHY RX IP includes calibration logic for 7 series FPGA families. The following table lists ports associated with the calibration logic.

Table 17: **7 Series FPGA Families Calibration Logic Signals**

Signal	Direction	Clock Domain	Description
dlyctrl_rdy_out	Output	N/A	Ready signal output from IDEALYCTRL, stating delay values are adjusted as per vtc changes.

Active Lane Support Signals

D-PHY TX IP supports active lanes. The following table lists ports associated with active lane support.

Table 18: **Active Lane Support Signal**

Signal	Direction	Clock Domain	Description
active_lanes_in[<n-1>:0][1]	Input	core_clk	Input to specify active lanes. This feature is available for D-PHY TX multi-lane configuration. Bits from LSB to MSB corresponds to TX Data lane 0 to 3.

Notes:

1. <n> is the data lane number.

Register Space

The MIPI D-PHY core register space is shown in the following table. This register interface is optional and allows you to access the general interconnect states. It also provides control to program protocol timing parameters, such as INIT, and the protocol watchdog timers.

IMPORTANT! *This memory space must be aligned to an AXI 32-bit word boundary.*

Endianness Details

All registers are in little endian format, as shown in the following table.

Table 19: **32-bit Little Endian Example**

Byte	Address Offset	Bit Boundaries
Byte 1	0x0	[7:0]
Byte 1	0x1	[15:8]

Table 19: **32-bit Little Endian Example** *(cont'd)*

Byte	Address Offset	Bit Boundaries
Byte 2	0x2	[23:16]
Byte 3	0x3	[31:24]

Table 20: **MIPI D-PHY Core Register Space**

Offset	Name	Width	Access	Description
0x0	CONTROL	32-bit	R/W	Enable and soft reset control for PHY.
0x4	IDELAY_TAP_VALUE	32-bit	R/W	To program the tap values in fixed mode of calibration in 7 series D-PHY RX configuration for lanes 1 to 4.
0x8	INIT	32-bit	R/W	Initialization timer.
0xC	Reserved	32-bit	N/A	N/A
0x10	HS_TIMEOUT	32-bit	R/W	Watchdog timeout in high-speed mode. Time from SoT to EoT is taken into account for the timer elapse. This register is available if the **Enable HS and ESC Timeout Counters/ Registers** checkbox is selected in the Vivado IDE. HS_RX_TIMEOUT is used for RX (slave) HS_TX_TIMEOUT is used for TX (master)
0x14	ESC_TIMEOUT	32-bit	R/W	Protocol specific. In escape mode, if line stays in LP-00 longer than this time period the core generates a timeout and goes to Stop state. This register is available if the **Enable HS and ESC Timeout Counters/Registers** checkbox is selected in the Vivado IDE. This register is used as Escape Mode Timeout in RX, and Escape Mode Silence Timeout in TX. Escape Mode Timeout should be greater than Escape Mode Silence Timeout.
0x18	CL_STATUS	32-bit	RO	Status register for PHY error reporting for clock Lane.
0x1C to 0x28	DL0_STATUS	32-bit	RO	Status registers for PHY error reporting for data lanes 1 to 4.
	DL1_STATUS	32-bit	RO	
	DL2_STATUS	32-bit	RO	
	DL3_STATUS	32-bit	RO	
0x30	HS_SETTLE	32-bit	R/W	HS_SETTLE timing control for lane 1.
0x34 to 0x44	Reserved	32-bit	N/A	N/A
0x48 to 0x60	HS_SETTLE	32-bit	R/W	HS_SETTLE timing control for lanes 2 to 8.
0x64 to 0x70	DL4_STATUS	32-bit	RO	Status registers for PHY error reporting for data lanes 5 to 8.
	DL5_STATUS	32-bit	RO	
	DL6_STATUS	32-bit	RO	
	DL7_STATUS	32-bit	RO	
0x74	IDELAY_TAP_VALUE	32-bit	R/W	To program the tap values in fixed mode of calibration in 7 series D-PHY RX configuration for lanes 5 to 8.

CONTROL Registers

The following table shows the CONTROL register (0x0 offset) bit mapping and description. Writing a 1 to SRST resets the MIPI D-PHY core. For the soft reset impact on the MIPI D-PHY core, see Reset Coverage table. The MIPI D-PHY core functions only when the DPHY_EN bit is set to 1 (by default).

Table 21: **CONTROL Register Bit Description**

Bits	Name	Access	Default Value	Description
31:2	Reserved	RO	0	Reserved.
1	DPHY_EN	R/W	1	Enable bit for D-PHY. 1: D-PHY controller is enabled. 0: D-PHY controller is disabled.
0	SRST	R/W	0	Soft reset for D-PHY Controller. If 1 is written to this bit, the D-PHY controller fabric logic and status registers are reset.

IDELAY_TAP_VALUE for Lanes 1 to 4

The IDELAY Tap Value register (0x4 Offset) is used to configure the idelay tap values in fixed mode for 7 series families. The tap values are programed dynamically during the core operation. The core need not be disabled to program a different tap value. The initial tap value for all lanes is same as the GUI parameter C_IDLY_TAP. The following table shows the Idelay Tap Value register bit description.

Table 22: **IDELAY_TAP VALUE Bit Description**

Bits	Name	Access	Default	Description
31:29	Reserved	RO	0	Reserved
28:24	Tap value for lane3	R/W	IDELAY tap value from GUI	Programs the IDELAY tap value for lane3
23:21	Reserved	RO	0	Reserved
20:16	Tap value for lane2	R/W	IDELAY tap value from GUI	Programs the IDELAY tap value for lane2
15:13	Reserved	RO	0	Reserved
12:8	Tap value for lane1	R/W	IDELAY tap value from GUI	Programs IDELAY Tap value for lane1
7:5	Reserved	RO	0	Reserved
4:0	Tap value for lane0	R/W	IDELAY tap value from GUI	Programs IDELAY Tap value for lane0

Notes:

1. All lanes tap values are available for R/W irrespective of the GUI configuration for number of lanes.

INIT Register

The INIT register (0x8 offset) is used for lane initialization. The following table shows the register bit description.

RECOMMENDED: *Xilinx® recommends that you use one millisecond or longer as INIT_VAL for the MIPI D-PHY TX core, and 500 µs for the MIPI D-PHY RX core.*

Table 23: **INIT Register Bit Description**

Bits	Name	Access	Default Value	Description
31:0	INIT_VAL	R/W	RX D-PHY IP:100 µs (32'h186A0) TX D-PHY IP:1 ms (32'hF4240)	Initialization timer value in ns.

HS_TIMEOUT Register

The HS_TIMEOUT register (0x10 offset) is used as a watchdog timer in high-speed mode. This register is used as HS_TX_TIMEOUT (MIPI D-PHY TX core) or as HS_RX_TIMEOUT (MIPI D-PHY RX core). The following table shows the HS_TIMEOUT register bit description.

Table 24: **HS_TIMEOUT Register Bit Description**

Bits	Name	Access	Default Value	Description
31:0	HS_RX_TIMEOUT/HS_TX_TIMEOUT	R/W	65,541	Maximum frame length in bytes. Valid range is 1,000 to 65,541. Timeout occurs for HS_RX_TIMEOUT/D-PHY_LANES at the RX data lanes in high speed mode. Timeout occurs for HS_TX_TIMEOUT/D-PHY_LANES at the TX data lanes in high speed mode.

ESC_TIMEOUT Register

The ESC_TIMEOUT register (0x14 offset) is used for the watchdog timer in escape mode. The following table shows the ESC_TIMEOUT register bit description.

Table 25: **ESC_TIMEOUT Register Bit Description**

Bits	Name	Access	Default Value	Description
31:0	ESC_TIMEOUT	R/W	25,600 ns	Escape timeout period in ns. Timeout occurs for the data lanes in escape mode.

CL_STATUS Register

CL_STATUS register (0x18 offset) provides clock lane status and state machine control. The following table provides CL_STATUS register bit description.

Table 26: **CL_STATUS Register Bit Description**

Bits	Name	Access	Default Value	Description
31:6	Reserved	RO	0	Reserved
5	ERR_CONTROL	RO	0	Clock lane control error. This bit is applicable only for the MIPI D-PHY RX core. This bit is asserted when D-PHY RX clock lane receives erroneous High-Speed entry sequence or ULPS entry sequence or ULPS exit sequence. This bit is cleared when D-PHY RX clock lane receives stopstate on the serial lines.
4	STOP_STATE	RO	0	Clock lane is in the Stop state.
3	INIT_DONE	RO	0	Set after the lane has completed initialization.
2	ULPS	RO	0	Set to 1 when the core in ULPS (ULP State) mode.
1:0	MODE	RO	0	2'b00: Low Power Mode (Control Mode) 2'b01: High Speed Mode 2'b10: Escape Mode

DL_STATUS Register

The DL_STATUS register (0x1C to 0x28, 0x64 to 0x70 offset) provides data lane status and state machine control. The following table provides the DL_STATUS register bit description.

Table 27: **DL_STATUS Register Bit Description**

Bits	Name	Access	Default Value	Description
31:16	PKT_CNT	RO	0	Number of packets received or transmitted on the data lane. This field is updated using the rxbyteclkhs clock and the RX clock lane must be in high-speed mode when reset is applied to the D-PHY RX IP. Otherwise, this value does not get reset for MIPI D-PHY RX IP configuration.
15:7	Reserved	RO	0	Reserved.
6	STOP_STATE	RO	0	Data lane is in the Stop state.
5	ESC_ABORT	R/W1C	0	This bit is set after the Data Lane Escape Timeout (Escape Mode Timeout in case of RX, or Escape Mode Silence Timeout in case of TX) is elapsed. Write-to-1 clears this bit.
4	HS_ABORT	R/W1C	0	Set after the Data Lane High-Speed Timeout (HS_TX_TIMEOUT or HS_RX_TIMEOUT) has elapsed. Write to 1 clears this bit.
3	INIT_DONE	RO	0	Set after the lane has completed initialization.
2	ULPS	RO	0	Set to 1 when the core is in ULPS mode.

Table 27: **DL_STATUS Register Bit Description** *(cont'd)*

Bits	Name	Access	Default Value	Description
1:0	MODE	RO	0	2'b00: Low Power mode (control mode). 2'b01: High Speed mode 2'b10: Escape mode.

HS_SETTLE Register

The HS_SETTLE register (0x30 offset, 0x48 to 0x60 offset) provides control to update the HS_SETTLE timing parameter for RX data lanes. The following table provides the HS_SETTLE register bit description.

Table 28: **HS_SETTLE Register Bit Description**

Bits	Name	Access	Default Value	Description
31:9	Reserved	RO	0	Reserved
8:0	HS_SETTLE_NS	R/W	135 + 10 UI	HS_SETTLE timing parameter (ns). This value will be applied for all data lanes and will only be applicable for D-PHY RX configuration. ***Note*:** UI is unit interval.

IDELAY_TAP_VALUE for Lanes 5 to 8

The IDELAY Tap Value register (0x74 Offset) is used to configure the IDELAY tap values in fixed mode for 7 series FPGAs. The tap values are programed dynamically during the core operation. The core need not be disabled to program a different tap value. The initial tap value for all lanes is same as the GUI parameter C_IDLY_TAP. The following table shows the Idelay Tap Value register bit description.

Table 29: **IDELAY_TAP VALUE Bit Description**

Bits	Name	Access	Default	Description
31:29	Reserved	RO	0	Reserved
28:24	Tap value for lane7	R/W	IDELAY tap value from GUI	Programs the IDELAY tap value for lane7
23:21	Reserved	RO	0	Reserved
20:16	Tap value for lane6	R/W	IDELAY tap value from GUI	Programs the IDELAY tap value for lane6
15:13	Reserved	RO	0	Reserved
12:8	Tap value for lane5	R/W	IDELAY tap value from GUI	Programs IDELAY Tap value for lane5
7:5	Reserved	RO	0	Reserved
4:0	Tap value for lane4	R/W	IDELAY tap value from GUI	Programs IDELAY Tap value for lane4

Notes:

1. All lanes tap values are available for R/W irrespective of the GUI configuration for number of lanes.

XILINX

Chapter 4

Designing with the Core

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

General Design Gudelines

This section describes the steps required to turn a MIPI D-PHY core into a fully functioning design with user-application logic.

IMPORTANT! *Not all implementations require all of the design steps listed here. Follow the logic design guidelines in this manual carefully.*

Use the Example Design

Each instance of the MIPI D-PHY v4.2 core created by the Vivado design tool is delivered with an example design that can be implemented in a device and then simulated. This design can be used as a starting point for your own design or can be used to sanity-check your application in the event of difficulty. See the Example Design content for information about using and customizing the example designs for the core.

Know the Degree of Difficulty

The MIPI D-PHY core design is challenging to implement in any technology, and the degree of difficulty is further influenced by:

- Maximum system clock frequency
- Targeted device architecture
- Nature of the user application

All MIPI D-PHY core implementations require careful attention to system performance requirements. Pipelining, logic mappings, placement constraints, and logic duplications are all methods that help boost system performance.

Registering Signals

To simplify timing and increase system performance in a programmable device design, keep all inputs and outputs registered between the user application and the core. This means that all inputs and outputs from the user application should come from, or connect to, a flip-flop. While registering signals might not be possible for all paths, it simplifies timing analysis and makes it easier for the Xilinx® tools to place and route the design.

Recognize Timing Critical Signals

The constraints provided with the example design identify the critical signals and timing constraints that should be applied.

Make Only Allowed Modifications

You should not modify the core. Any modifications can have adverse effects on system timing and protocol compliance. Supported user configurations of the core can only be made by selecting the options in the customization IP dialog box when the core is generated.

I/O Placement

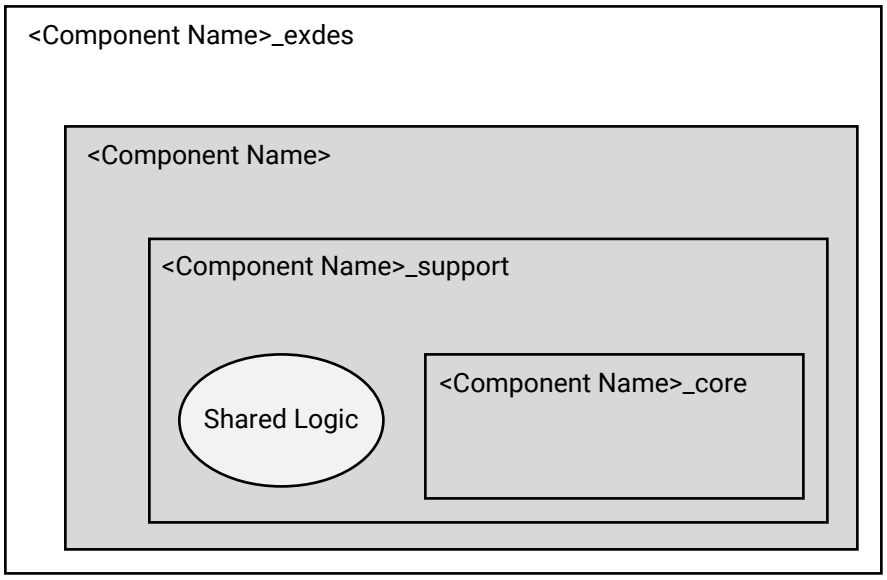
The MIPI D-PHY protocol supports the MIPI_DPHY_DCI I/O standard, and this I/O standard is supported only in an HP I/O bank in the UltraScale+™, Zynq® UltraScale+™ MPSoC and XPIO Bank in Versal families. It is recommended that you use consecutive bit slices for data lanes starting from the clock lane BITSLICE. All I/O placements should be restricted to the same I/O bank.

Shared Logic

Shared Logic provides a flexible architecture that works both as a stand-alone core and as part of a larger design with one of more core instances. This minimizes the amount of HDL modifications required, but at the same time retains the flexibility of the core.

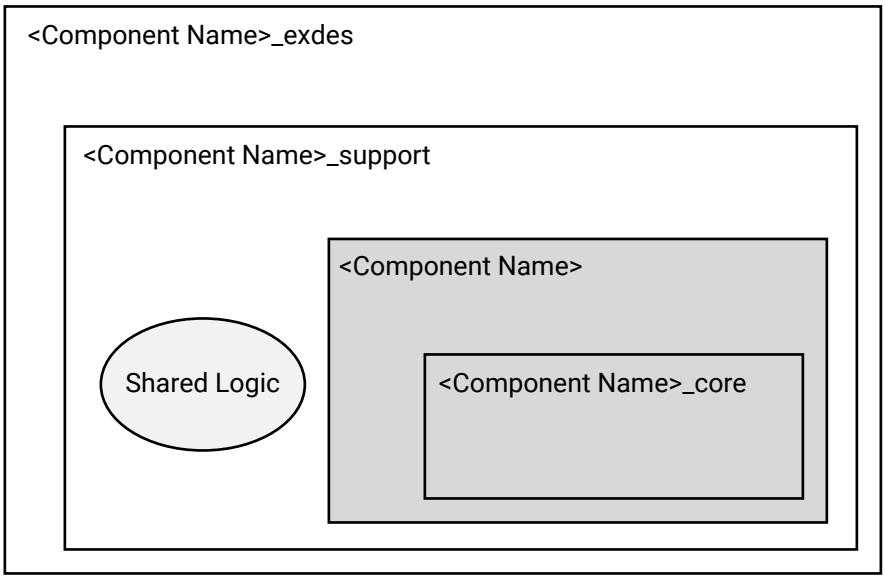
There is a level of hierarchy called <component_name>_support. The following figures show two hierarchies where the shared logic is either contained in the core or in the example design. In these figures, <component_name> is the name of the generated core. The difference between the two hierarchies is the boundary of the core. It is controlled using the Shared Logic option in the Vivado IDE Shared Logic tab for the MIPI D-PHY Controller.

Figure 9: **Shared Logic Included in Core**



X15949-020916

Figure 10: **Shared Logic Included in Example Design**



X15948-020916

The shared logic comprises an MMCM, a PLL and some BUFGs (maximum of 4).

Shared Logic in Core

Select Include Shared Logic in core if:

- You do not require direct control over the MMCM and PLL generated clocks
- You want to manage multiple customizations of the core for multi-core designs
- This is the first MIPI D-PHY core in a multi-core system

These components are included in the core, and their output ports are also provided as core outputs.

Shared Logic in Example Design

Select Include Shared Logic in example design if:

- This is the second MIPI D-PHY core in a multi-core design
- You only want to manage one customization of the MIPI D-PHY core in your design
- You want direct access to the input clocks

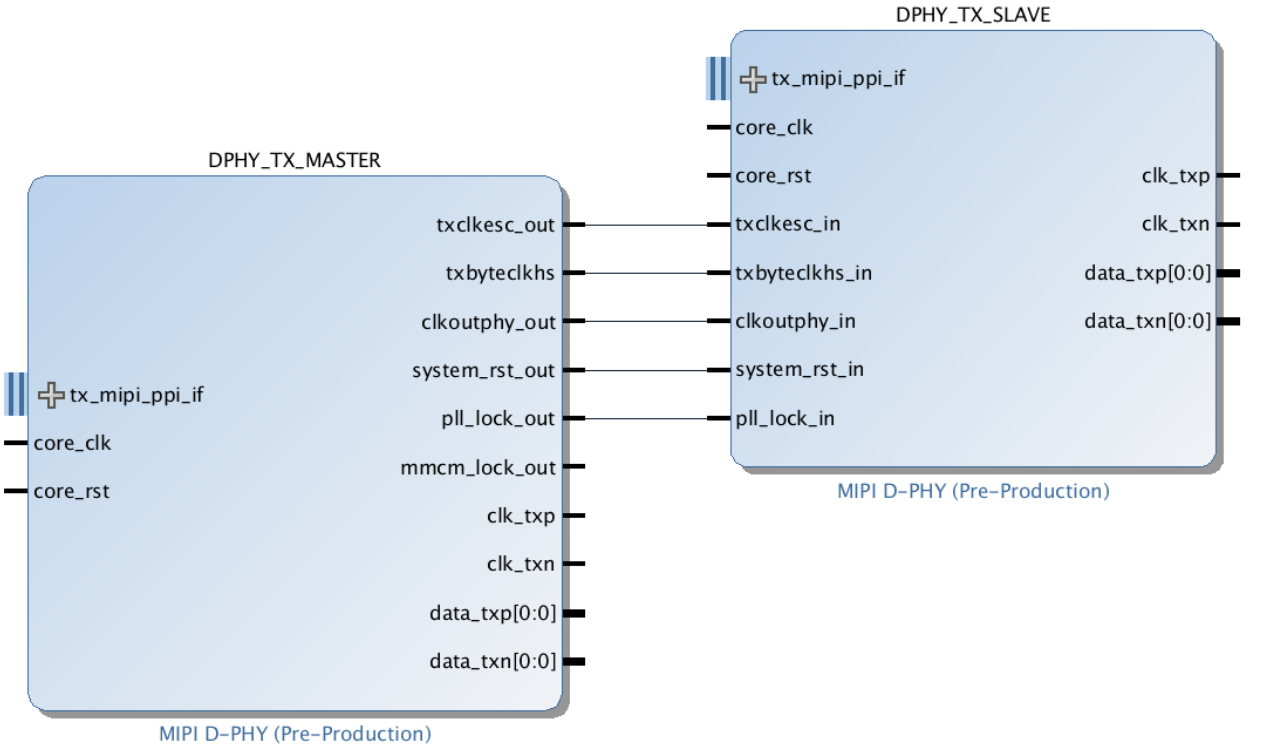
To fully utilize the MMCM and PLL, customize one MIPI D-PHY core with shared logic in the core and one with shared logic in the example design. You can connect the MMCM/PLL outputs from the first MIPI D-PHY core to the second core.

If you want fine control you can select **Include shared logic in example design** and base your own logic on the shared logic produced in the example design.

Case 1: UltraScale+ Device MIPI D-PHY TX Core

The following figure shows the sharable resource connections from the MIPI D-PHY TX core with shared logic included (DPHY_TX_MASTER) to the instance of another MIPI D-PHY TX core without shared logic (DPHY_TX_SLAVE).

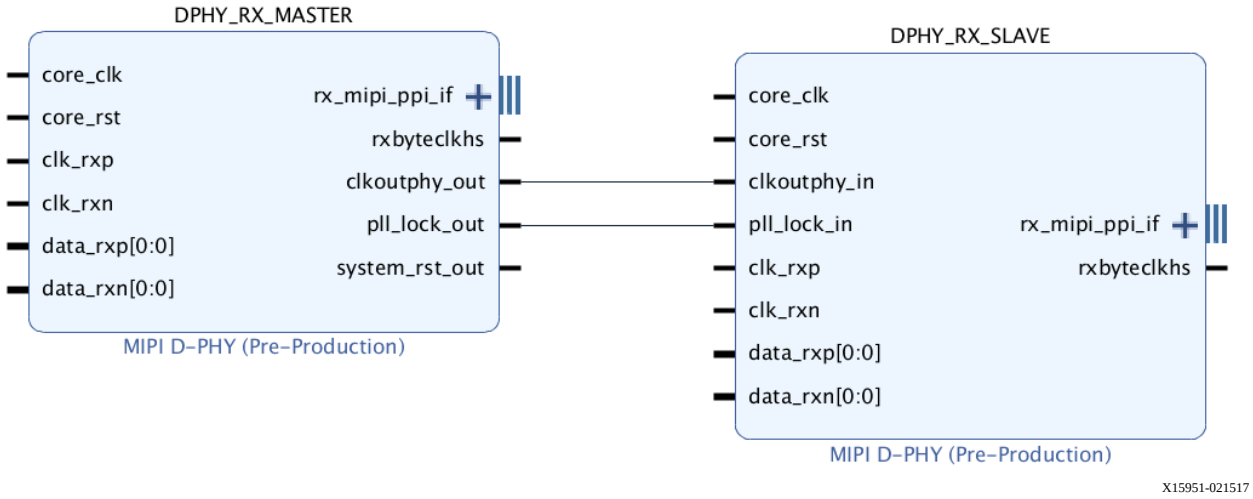
Figure 11: **Shared Logic Example for MIPI D-PHY TX Core**



Case 2: UltraScale+ Device MIPI D-PHY RX Core

The following figure shows the sharable resource connections from the MIPI D-PHY RX core with shared logic included (DPHY_RX_MASTER) to the instance of another MIPI D-PHY RX core without shared logic (DPHY_RX_SLAVE).

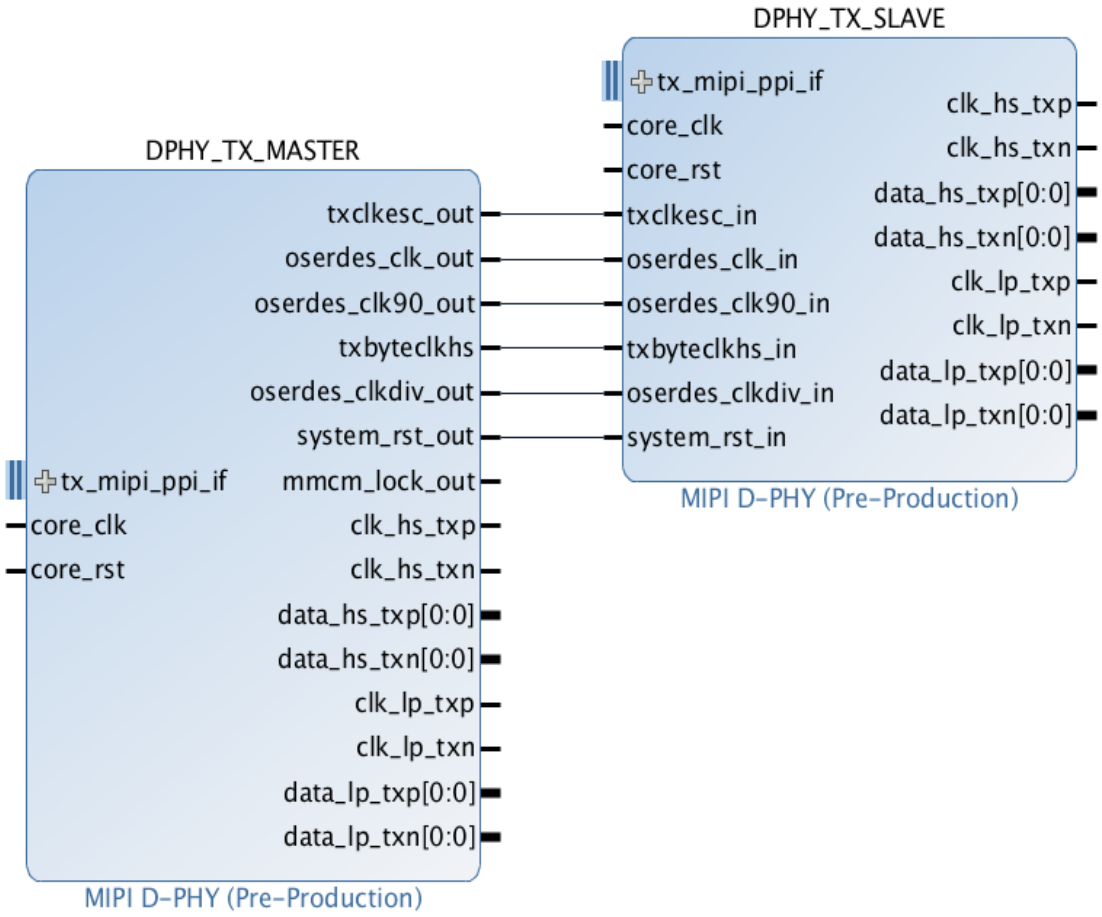
Figure 12: **Shared Logic Example for MIPI D-PHY RX Core**



Case 3: 7 Series FPGAs MIPI D-PHY TX Core

The following figure shows the sharable resource connections from the MIPI D-PHY TX core with shared logic included (DPHY_TX_MASTER) to the instance of another MIPI D-PHY TX core without shared logic (DPHY_TX_SLAVE).

Figure 13: **Shared Logic Example for 7 Series FPGAs MIPI D-PHY TX Core**



I/O Planning for UltraScale+ Devices

The MIPI D-PHY Controller provides an I/O planner feature for I/O selection. You can select any I/O for the clock and data lanes in the TX core configuration for the selected HP I/O bank.

For the RX core configuration, dedicated byte clocks (DBC) or quad byte clocks (QBC) are listed for the clock lane for the selected HP I/O bank. For the QBC clock lane all of the I/O pins are listed for data lane I/O selection but for the DBC clock lane only byte group I/O pins are listed for data lane I/O selection in the RX core configuration.

Eight D-PHY IP cores can be implemented per IO bank due to BITSLICE and BITSLICE_CONTROL instances in UltraScale+ devices.

IMPORTANT! *If the RX data lane I/O pins are selected non-contiguously then an additional one, two, or three I/O pins (RX_BITSLICE) are automatically used for clock/Strobe propagation. Therefore, it is recommended that you select adjacent I/O pins for the RX configuration to make efficient use of the I/O. The propagation of strobes to the RX data pins follows the inter-byte and inter-nibble clocking rules given in the UltraScale Architecture SelectIO Resources User Guide (UG571).*

I/O Planning for Versal Devices

The MIPI D-PHY GUI does not have I/O Assignment tab for Versal devices. Instead you need to use consolidated I/O planning in the main Vivado IDE Planning that is nibble planner. You can select any I/O for the clock and data lanes in the TX core configuration for the selected XPIO bank.

For the RX core configuration, select the clock capable pin that is 0th pin of a nibble for the clock lane for the selected XPIO bank.

Detailed steps on how to use the Vivado IDE Planning is detailed under section "I/O Planning for Versal Advanced IO Wizard" in *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320).

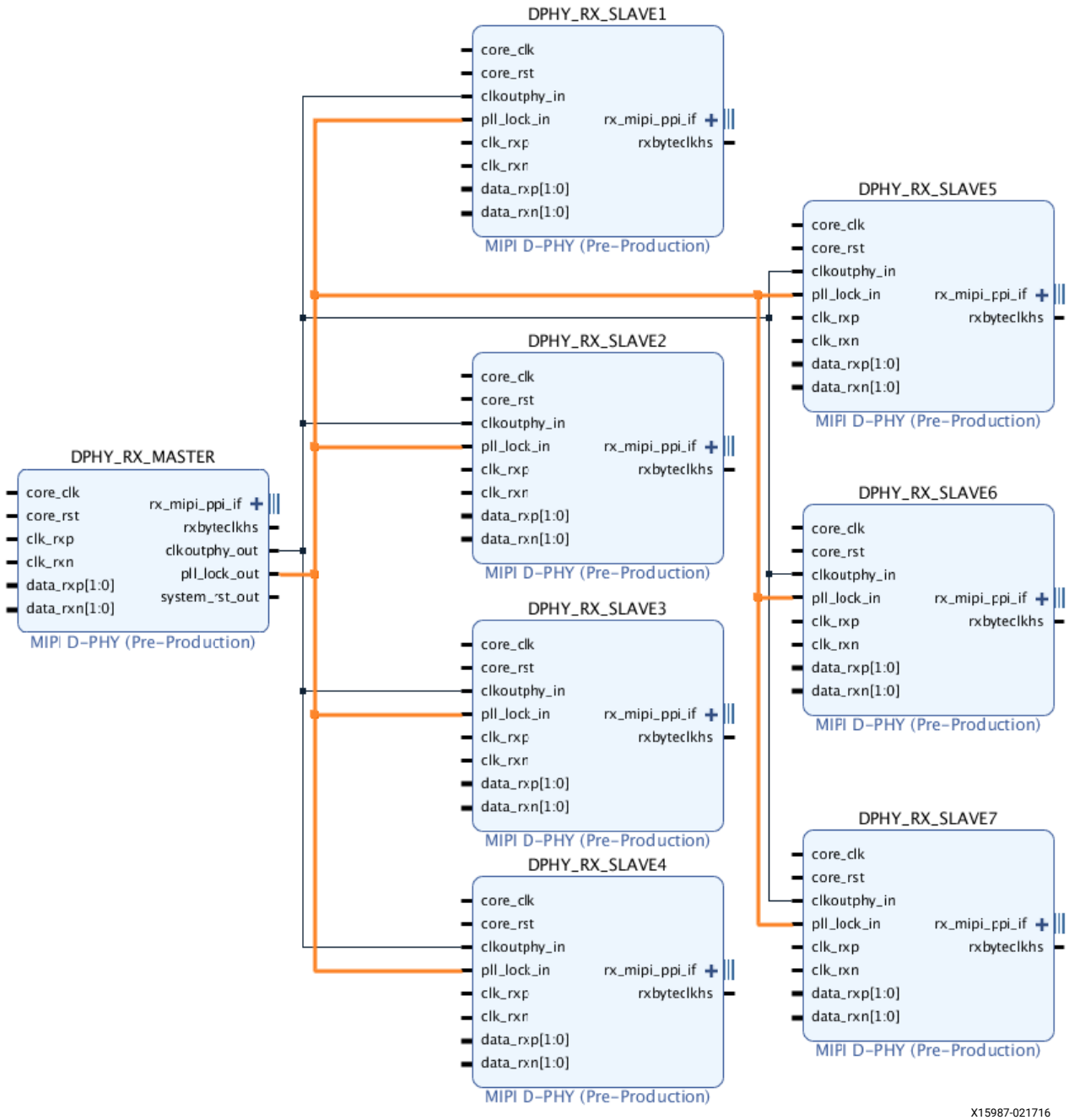
While selecting the IOs in a bank across nibbles, users need to ensure the inter-nibble and inter-byte clock guidelines are followed. Refer "Clocking" section in *Versal ACAP SelectIO Resources Architecture Manual* (AM010).

The following figure shows the eight MIPI D-PHY RX cores configured with one clock lane and two data lanes and implemented in a single I/O bank.

The DPHY_RX_MASTER is configured with **Include Shared Logic in core** option and the remaining cores are configured with **Include Shared Logic in example design**. The constant `clkoutphy` signal is generated within the PLL of the DPHY_RX_MASTER core irrespective of the line rate and shared with all other slave IP cores (DPHY_RX_SLAVE1 to DPHY_RX_SLAVE7) with different line rates. The `pll_lock` signal connection is required for slave IP initialization.

***Note*:** The master and slave D-PHY RX cores can be configured with the different line rate (less than 1500) when sharing `clkoutphy` within an I/O bank.

Figure 14: **MIPI D-PHY RX Core Shared Logic Use Case for Single I/O Bank**



Clocking

The MIPI D-PHY Controller requires a 200 MHz free running clock (`core_clk`). This clock is used as input to the Mixed-Mode Clock Manager (MMCM), and the required clocks are generated based on IP configurations.

IMPORTANT! *`core_clk` should be either coming from the on-board oscillator or the single MMCM or the PLL from target FPGA device. `core_clk` should not be generated from the cascaded MMCM blocks.*

The following figures show the MIPI D-PHY Controller clock diagrams for UltraScale+ families. The MIPI D-PHY TX core takes `core_clk` as an input and generates the necessary clocks from the MMCM. The `clkoutphy` signal from the PLL is used in the BITSLICE_CONTROL of the PHY block in native mode.

Figure 15: **MIPI D-PHY Core TX Clocking for Versal™ Families**

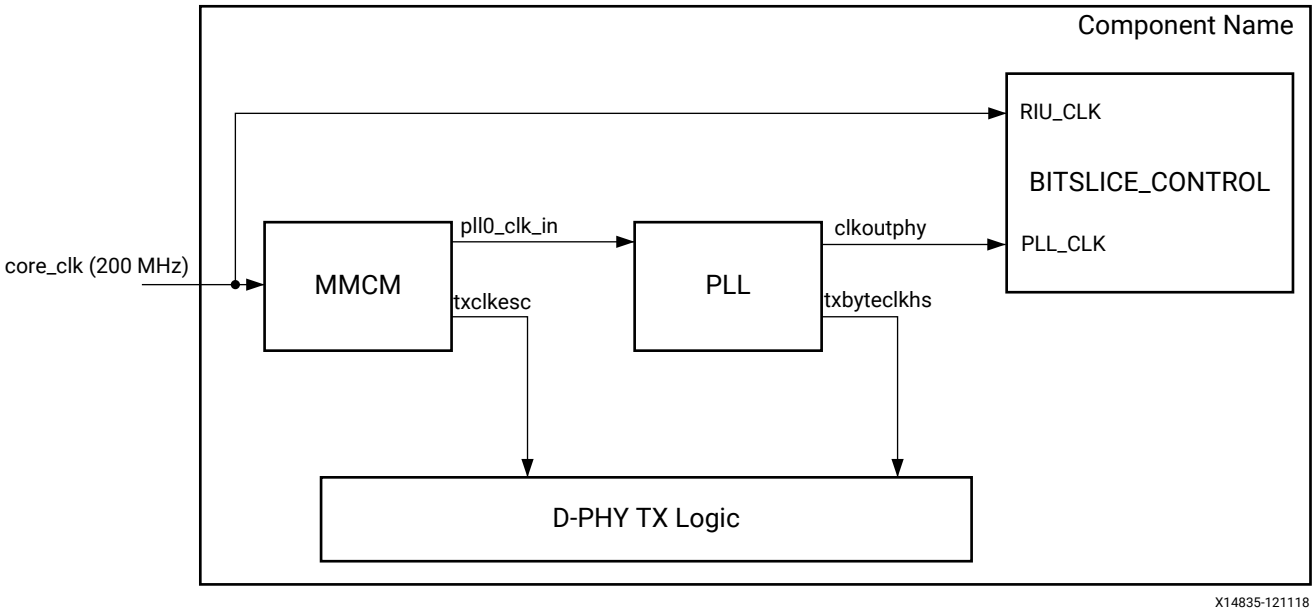


Figure 16: **MIPI D-PHY Core TX Clocking for UltraScale+™ Families**

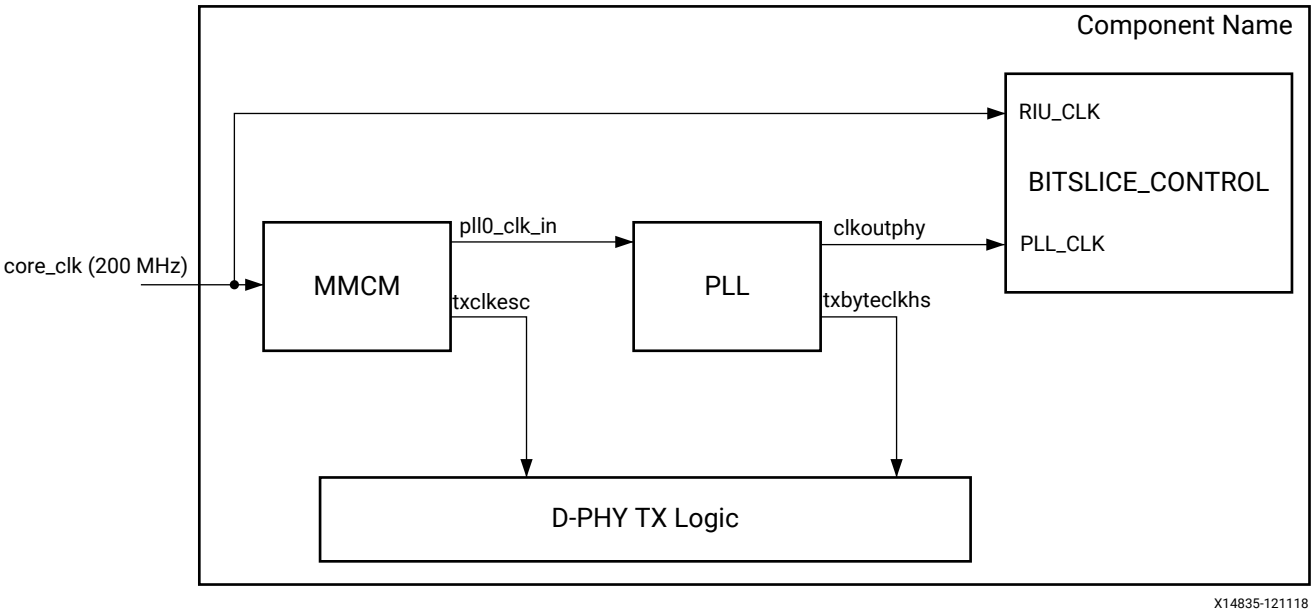
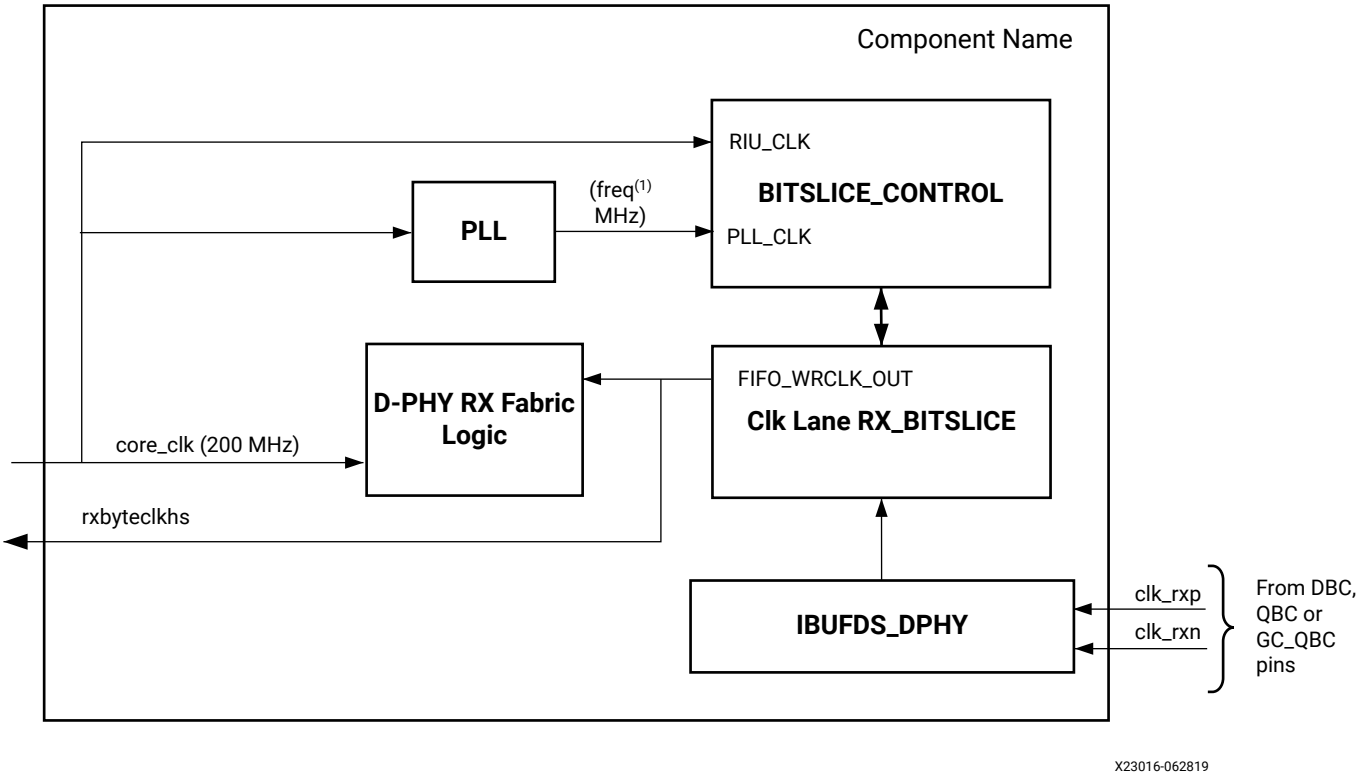


Figure 17: **MIPI D-PHY Core RX Clocking for UltraScale+ Families where Line Rates ≤ 1500 Mb/s**



***Note*:** For line rates less than or equal to 1500 Mb/s, the frequency value is 1500. When line rates are greater than 1500 Mb/s and deskew is disabled, the frequency value is equal to the line rate.

The following figures show the MIPI D-PHY Controller clock diagrams for 7 series FPGA families. The MIPI D-PHY Controller takes `core_clk` as an input and generates the necessary clocks from the MMCM for D-PHY TX IP. MMCM is *not* used in the D-PHY RX IP when the line rate is less than or equal to 1500 Mb/s; for line rates greater than 1500 Mb/s, MMCM is used.

Figure 18: **MIPI D-PHY Core TX Clocking for 7 Series FPGA Families**

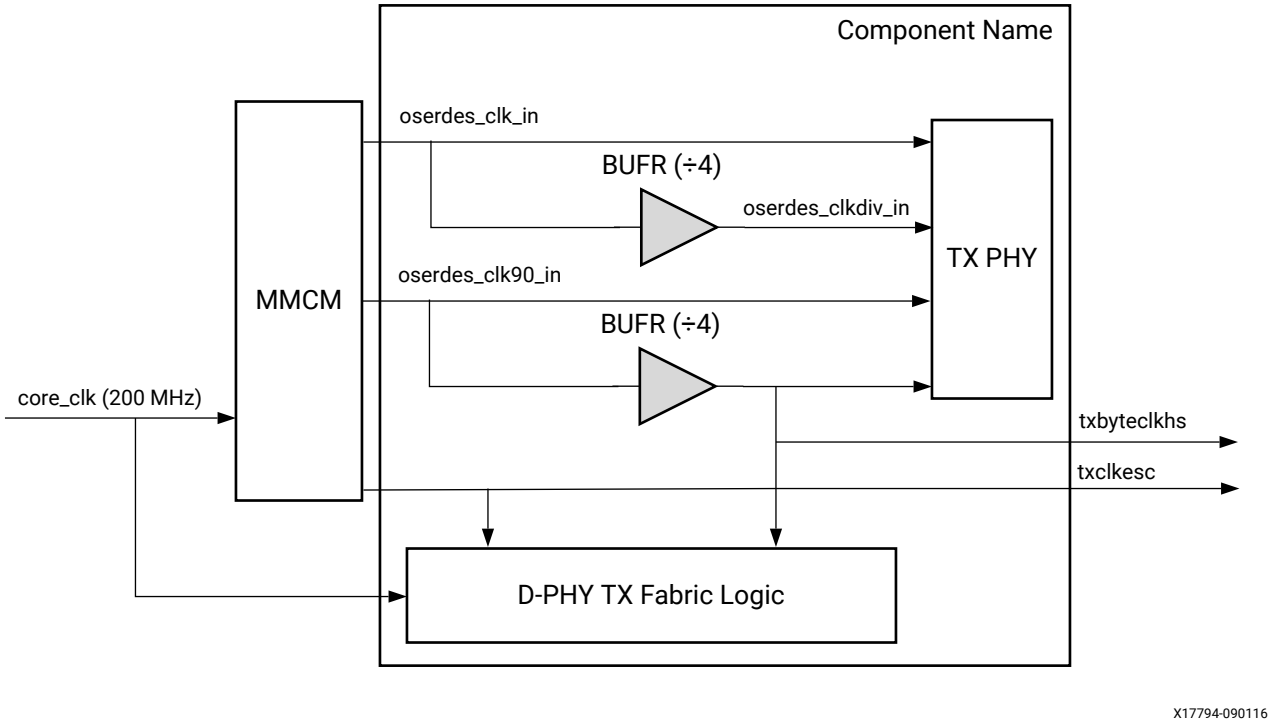
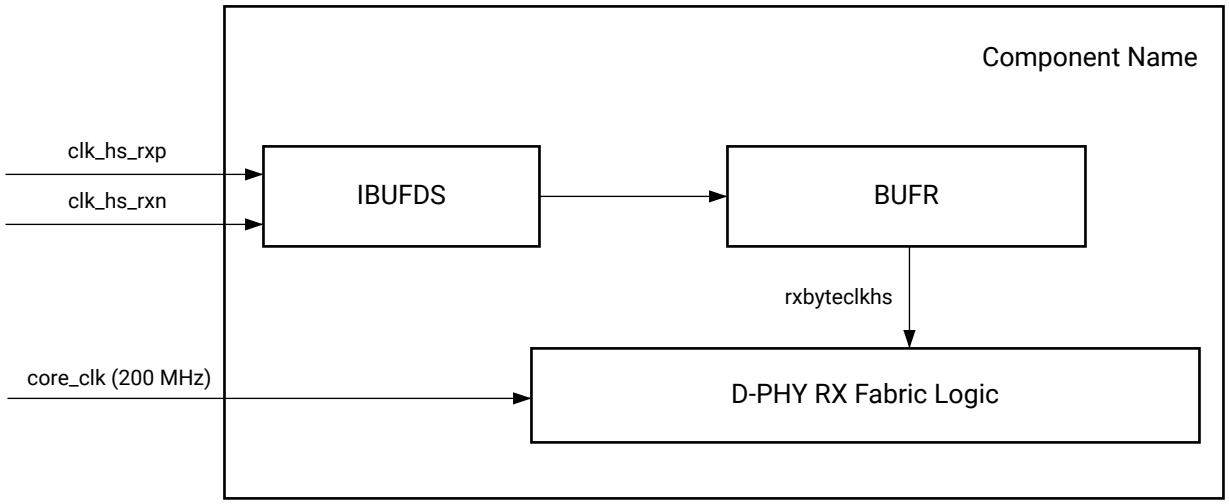


Figure 19: **MIPI D-PHY Core RX Clocking for 7 series FPGA Families**



The following table provides details about the core clocks.

Table 30: **MIPI D-PHY Clocking Details**

Clock	Frequency	IP Configuration	Notes
core_clk	200.000 MHz	All	Used for control logic and input to MMCM.
txbyteclkhs[1]	10.000–187.500 MHz Derived from the line rate divided by 8	MIPI D-PHY TX core Shared Logic in Core	Input to PHY and used to transmit high-speed data. This clock is generated from oserdes_clk90_out as source for 7 series FPGAs.
xiphy_byteclk_out[1]	75.000–187.500 MHz Line rate divided by ratio[2]	MIPI D-PHY TX core Shared Logic in Core Line rate < 600 Mb/s	Input to PHY and used to transmit high-speed data. This clock is not available for the 7 series FPGA families.
clkoutphy_out[1]	Line rate	Shared Logic in Core	PHY serial clock. This clock is not available for the 7 series FPGA families.
txclkesc_out	10.000–20.000 MHz	MIPI D-PHY TX core Shared Logic in Core	Clock used for Escape mode operations
txbyteclkhs_in[1]	10.000–187.500 MHz Derived from the line rate divided by 8.	MIPI D-PHY TX core Shared Logic in Example Design	Input to PHY and used to transmit high-speed data. This clock should be generated from oserdes_clk90_in as source for the 7 series FPGA families.
xiphy_byteclk_in[1]	75.000–187.500 MHz Line rate divided by ratio[2]	MIPI D-PHY TX core Shared Logic in Example Design Line rates < 600 Mb/s.	Input to PHY and used to transmit high-speed data. This clock is not available for the 7 series FPGA families.
clkoutphy_in[1]	Line rate	Shared Logic in Example Design	PHY serial clock. This clock is not available for the 7 series FPGA families.
txclkesc_in	10.000–20.000 MHz	MIPI D-PHY TX core Shared Logic in Example Design	Clock used for Escape mode operations
rxbyteclkhs	10.000–187.500 MHz Derived from the line rate divided by 8.	MIPI D-PHY RX core	Clock received on RX clock lane and used for high-speed data reception

Table 30: **MIPI D-PHY Clocking Details** *(cont'd)*

Clock	Frequency	IP Configuration	Notes
oserdes_clk_out	line rate/2	7 series FPGA families and Shared Logic is in the core and D-PHY TX configuration	Used to connect the CLK pin of TX clock lane OSERDES
oserdes_clk90_out	line rate/2	7 series FPGA families and Shared Logic is in the core and D-PHY TX configuration	Used to connect the CLK pin of TX data lane OSERDES. It has 90 degree phase shift relationship with oserdes_clk_out
oserdes_clkdiv_out	line rate/8	7 series FPGA families and Shared Logic is in the core and D-PHY TX configuration	Used to connect the CLKDIV pin of TX clock lane OSERDES and generated from oserdes_clk_out as source
oserdes_clk_in	line rate/2	7 series FPGA families and Shared Logic is in the Example Design and D-PHY TX configuration	Used to connect the CLK pin of TX clock lane OSERDES
oserdes_clk90_in	line rate/2	7 series FPGA families and Shared Logic is in the Example Design and D-PHY TX configuration	Used to connect the CLK pin of TX data lane OSERDES and should have 90 degree phase shift with oserdes_clk_in
oserdes_clkdiv_in	line rate/8	7 series FPGA families and Shared Logic is in the Example Design and D-PHY TX configuration	Used to connect the CLKDIV pin of TX clock lane OSERDES and should be generated from oserdes_clk_in as source
cl_tst_clk_in	line rate/2	7 series FPGA families D-PHY TX configuration and Shared Logic is in the Example Design and Infer OBUFTDS option is selected	Used for TX clock lane IO buffer tristate signal synchronization
dl_tst_clk_in	line rate/2	7 series FPGA families D-PHY TX configuration and Shared Logic is in the Example Design and Infer OBUFTDS option is selected	Used for TX data lane IO buffer tristate signal synchronization
cl_tst_clk_out	line rate/2	7 series FPGA families D-PHY TX configuration and Shared Logic is in the core and Infer OBUFTDS option is selected	Used for TX clock lane IO buffer tristate signal synchronization

Table 30: **MIPI D-PHY Clocking Details** *(cont'd)*

Clock	Frequency	IP Configuration	Notes
dl_tst_clk_in	line rate/2	7 series FPGA families D-PHY TX configuration and Shared Logic is in the core and Infer OBUFTDS option is selected	Used for TX data lane IO buffer tristate signal synchronization

Notes:

1. The txbyteclkhs and xiphy_byteclk clocks should be generated from same clock source or PLL.
2. Ratios for various line rate range are as follows:
   - 4 for 300 to 599 Mb/s.
   - 2 for 150 to 299 Mb/s.
   - 1 for 80 to 149 Mb/s.

For example, the xiphy_byteclk frequency is 125.000 MHz for 500 Mb/s line rate.

IMPORTANT! *All the input clocks supplied to the MIPI D-PHY core should have ±100 PPM difference and violating this results in either data corruption or data duplication.*

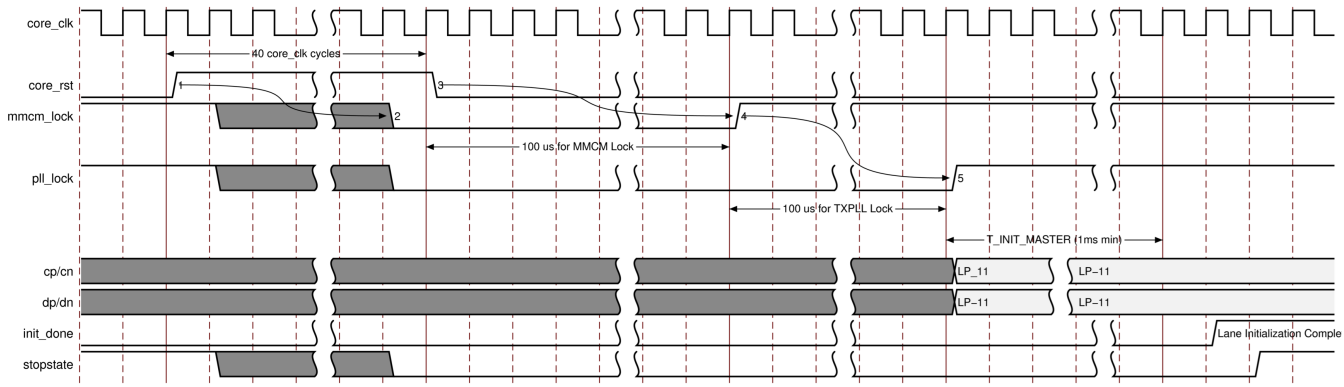
Resets

The active-High reset signal `core_rst` is used in the MIPI D-PHY Controller.

The following figure shows the power-on reset behavior for the MIPI D-PHY Controller.

1. The `core_rst` signal is asserted for forty `core_clk` cycles. Forty clock cycles are required to propagate the reset throughout the system.
2. The `mmcm_lock` and `pll_lock` signals go Low due to `core_rst` assertion.
3. The `mmcm_lock` signal is asserted within 100 µs after `core_rst` deassertion and generates the input clock for the PLL.
4. The `pll_lock` signal is asserted within 100 µs after `mmcm_lock` assertion.
5. LP-11 is driven on the lines for T_INIT or longer. This helps the MIPI D-PHY core complete the lane initialization. Lane initialization is indicated by the `init_done` internal status signal in the waveform.
6. After LPX_PERIOD of LP-11 assertion, `stopstate` is asserted.

Figure 20: **Power on Reset Sequence for the MIPI D-PHY Core**



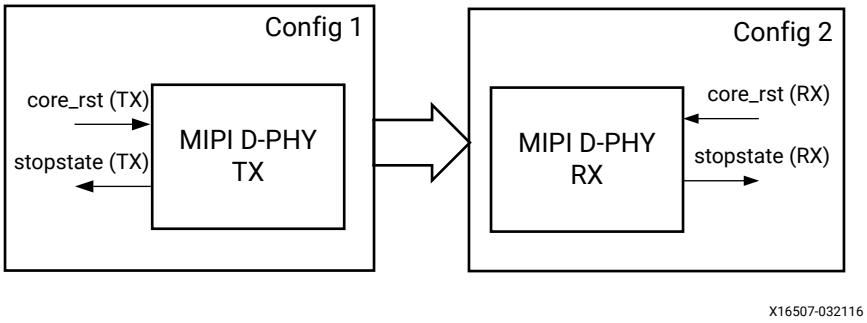
The following table summarizes all resets available to the MIPI D-PHY Controller and the components affected by them.

Table 31: **Reset Coverage**

Functional Block	core_rst	DPHY_EN (Core Enable from Register)	SRST (Soft Reset from Register)	s_axi_aresetn
TX/RX PCS	Yes	Yes	Yes	No
TX/RX PHY	Yes	Yes	No	No
Registers	Yes	Yes	Yes	Yes
Lane Initialization	Yes	Yes	No	No

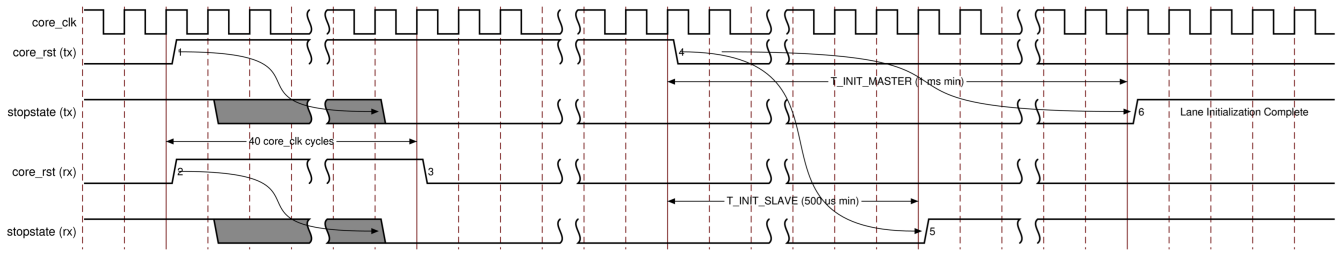
The following figure shows the MIPI D-PHY TX IP and MIPI RX IP connected in a system. Config 1 and Config 2 can be in the same or multiple device(s)/board(s).

Figure 21: **MIPI D-PHY TX and RX System**



The following figure shows the reset assertion sequence for MIPI D PHY Core:

Figure 22: **Reset Assertion Sequence for MIPI D-PHY Core**



Resetting the MIPI D-PHY TX and RX Core

To reset the MIPI D-PHY TX and RX core in a system, perform the following procedure:

1. Assert `core_rst` of MIPI D-PHY RX IP core for minimum 40 `core_clk` cycles.
2. Assert `core_rst` of MIPI D-PHY TX IP core.
3. Assert the MIPI D-PHY RX `core_rst` signal for a minimum 40 `core_clk` cycles.
4. Release the MIPI D-PHY RX `core_rst` signal.

   Note: When there are multiple instances of D-PHY within the same bank, or when there are TX and RX in same bank, perform the reset removal at same time.
5. Release the MIPI D-PHY TX `core_rst` signal.

   Note: When there are multiple instances of D-PHY within the same bank, or when there are TX and RX in same bank, perform the reset removal at same time.
6. The MIPI D-PHY RX IP core initialization happens after a T_INIT_SLAVE time of 500 µs and is indicated by the assertion of `stopstate`.
7. The MIPI D-PHY TX IP core initialization happens after a T_INIT_MASTER time if 1 ms and is indicated by `stopstate` assertion.
8. At this point, the MIPI D-PHY TX IP core is ready to accept data from the TX PPI interface.

Note: The impact of the assertion of `core_rst` on the MIPI D-PHY core is the same as the assertion of the DPHY_EN bit of the CONTROL register.

Resetting TX-Only Designs

1. Assert the MIPI D-PHY TX IP `core_rst`.
2. Hold reset signals for a minimum of 40 `core_clk` cycles.
3. Deassert the MIPI D-PHY TX `core_rst` signal.
4. The MIPI D-PHY TX IP core initialization completes after a `T_INIT_MASTER` time of 1 ms and is indicated by the assertion of the `stopstate` signal.

5. At this point, the MIPI D-PHY RX IP core is ready to accept, and the MIPI D-PHY TX IP core is ready to send, data fed from the TX PPI interface.

Resetting RX-Only Designs

1. Assert the MIPI D-PHY RX IP `core_rst`.
2. Hold the reset signals for a minimum of 40 `core_clk` cycles.
3. Deassert the MIPI D-PHY RX `core_rst` signal.
4. The MIPI D-PHY RX IP core initialization completes after a `T_INIT_SLAVE` time of 500 µs. This is indicated by the assertion of the `stopstate` signal.
5. At this point, the MIPI D-PHY RX IP core is ready to accept MIPI D-PHY serial data from the TX partner.

Protocol Description

A high-speed clock is generated from the clock lane and is used for high-speed operations. The line status is detected based on low-power signals. During normal operation, the Lane module is always in the control mode or high-speed mode. High-speed operations happen in bursts, and start from and end in the Stop state (LP-11).

IMPORTANT! *A low-power line state of less than 20 ns is ignored by the MIPI D-PHY RX core.*

The following sections describe the features in detail for the MIPI D-PHY Controller.

Initialization

After power-up, the slave side PHY is initialized when the master PHY drives a Stop state for a period longer than T_INIT. The first Stop state (LP-11) that is longer than the specified T_INIT is called the Initialization period.

***Note*:** T_INIT is considered a protocol-dependent parameter which must be longer than 100 µs.

High Speed Transfer

High-speed signaling is used for fast data traffic. High-speed data communication appears in bursts with an arbitrary number of payload data bytes.

High Frequency Clock Transmission

The clock lane transmits a low-swing, differential high-speed DDR clock from the master to the slave for high-speed data transmission. It is controlled by the protocol through the clock lane PPI. The clock signal has quadrature-phase with a toggling bit sequence on the data lane.

Escape Mode

The low-power (LP) functions include single-ended transmitters (LP-TX), receivers (LP-RX), and Low-Power Contention-Detectors (LP-CD). Because this core supports only unidirectional communication, contention detector logic is not required. Low-power functions are always present in pairs as these are single-ended functions operating on each of the two interconnect wires individually.

Remote Triggers

The MIPI D-PHY Controller defines four types of trigger commands. In escape mode, the MIPI D-PHY applies Spaced-One-Hot bit encoding for asynchronous communication. Therefore, operation of a data lane in this mode does not depend on the clock lane. Trigger signaling is the mechanism to send a flag to the protocol at the receiving side, on request of the protocol on the transmitting side. So, data received after the trigger command is not interpreted by the core.

Low Power Data Transmission

Low-Power Data Transmission (LPDT) data can be communicated by the protocol at low speed, while the lane remains in low-power mode. Data is encoded on the lines with the Spaced-One-Hot code. The data is self-clocked by the applied bit encoding and does not rely on the clock lane. The core supports a maximum data transfer of 10 Mb/s in low-power (LP) mode.

***Note*:** The maximum clock frequency is 20 MHz in LPDT.

Ultra-Low Power State

This is one type of escape mode and is supported by both the clock lane and data lane. You can exit from the ultra-low power state by the wakeup timer, which is governed by the T_WAKEUP protocol timing parameter.

Interfaces

The MIPI D-PHY Controller has a PPI interface and an AXI4-Lite interface.

PPI Interface

The following section explains the PPI timing through a series of examples.

Example 1: High-Speed Transmit from D-PHY TX (Master) Side

This section describes a high-speed transmission by the D-PHY TX (Master) IP.

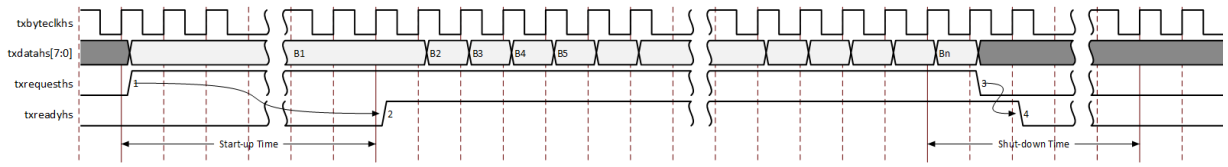
1. While `txrequesths` is Low, the lane module ignores the value of `txdatahs[7:0]`. To begin transmission, the protocol drives the `txdatahs` signal with the first byte of data and asserts the `txrequesths` signal.
2. This data byte is accepted by the D-PHY on the first rising edge of `txbyteclkhs` with `txreadyhs` also asserted. Now, the protocol logic drives the next data byte onto `txdatahs`. After every rising clock cycle with `txreadyhs` active, the protocol supplies a new valid data byte or ends the transmission.
3. After the last data byte has been transferred to the lane module, `txrequesths` is driven Low to cause the lane module to stop the transmission and enter Stop state.
4. The `txreadyhs` signal is driven Low after `txrequesths` goes Low.

The minimum number of bytes transmitted can be as small as one.

***Note*:** The `txrequesths` signal of the TX clock lane must be asserted to start the high-speed data transfer.

The following figure shows the high-speed transmission by the D-PHY TX (Master) IP.

Figure 23: **High-Speed Mode Data Transfer from D-PHY TX (Master)**



The start-up time can be calculated using.

`2*LPX_TIME + HS_PREPARE_TIME + HS_ZERO_TIME + CDC_DELAY.`

Where `HS_PREPARE` and `HS_ZERO` are D-PHY protocol timing parameters and maximum values used in the IP. You cannot control the`HS_PREPARE` and `HS_ZERO` values as they are automatically calculated based on the line rate. You can configure LPX using the Vivado® IP Catalog. `CDC_DELAY` will be 30 ns + 2 txbyteclkhs.

Example 2: Low-Power Data Transfer from D-PHY TX (Master) Side

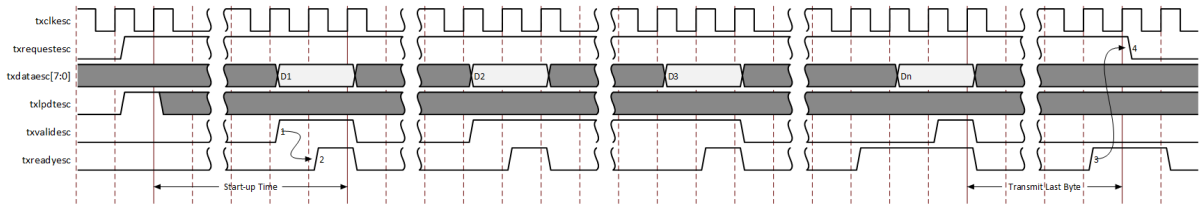
This section describes low-power data transmission operation.

1. For low-power data transmission, the `txclkesc` signal is used. The PPI directs the data lane to enter low-power data transmission escape mode by asserting `txrequestesc` and setting `txlpdtesc` High.

2. The low-power transmit data is transferred on the `txdataEsc[7:0]` when `txvalidesc` and `txreadyesc` are both active at a rising edge of `txclkesc`. The byte is transmitted in the time after the `txdataesc` is accepted by the MIPI D-PHY TX core (`txvalidesc` and `txreadyesc` are High) and therefore the `txclkesc` continues running for some minimum time after the last byte is transmitted.
3. The PPI knows the byte transmission is finished when `txreadyesc` is asserted.
4. After the last byte has been transmitted, the PPI deasserts `txrequestesc` to end the low-power data transmission. This causes `txreadyesc` to return Low, after which the `txclkesc` clock is no longer needed.

The following figure shows the low-power data transmission operation.

Figure 24: **Low-Power Data Transfer from D-PHY TX (Master)**



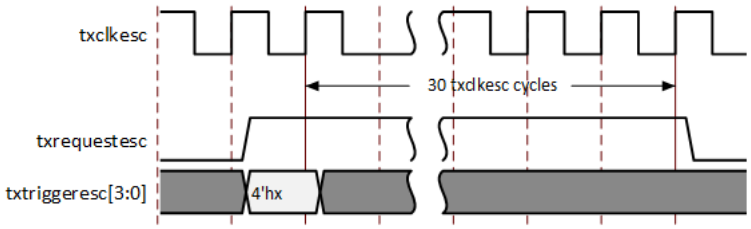
Example 3: Trigger Command Transmission from D-PHY TX (Master) Side

This section describes trigger transmission operation.

1. `txrequestesc` is asserted along with the trigger value in `txtriggeresc[3:0]`.
2. Because the PPI does not have a handshake signal to report back the trigger transmission on the serial line, `txrequestesc` is driven Low after 30 `txclkesc` clock cycles. The 30 clock cycles ensures that the MIPI D-PHY TX core transfers the trigger command on the serial line.

The following figure shows the trigger transmission operation.

Figure 25: **Trigger Command Transmission from D-PHY TX (Master)**



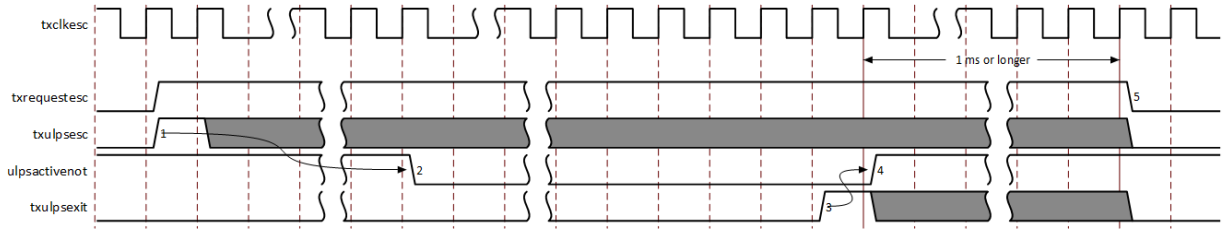
Example 4: D-PHY TX (Master) Data Lane ULPS Operation

This section describes a TX data lane ULPS operation.

1. The PPI drives `txrequestesc` High to initiate the ULPS entry request. The `txulpsesc` signal is asserted for one `txclkesc` cycle.
2. The MIPI D-PHY TX core drives the data lane `ulpsactivenot` (active-Low) to Low which indicates that the ULPS command is transmitted on the serial lines.
3. The PPI drives the `txulpsexit` pulse to start the ULPS exit operation.
4. The MIPI D-PHY TX core responds by deasserting the `ulpsactivenot` signal and starts transmitting MARK-1 on the line for T_WAKEUP time.
5. The PPI deasserts the `txrequestesc` after T_WAKEUP time has elapsed following the deassertion of the `ulpsactivenot` signal.

The following figure shows TX data lane ULPS operation.

Figure 26: **D-PHY TX (Master) ULPS Mode Operation for Data Lane**



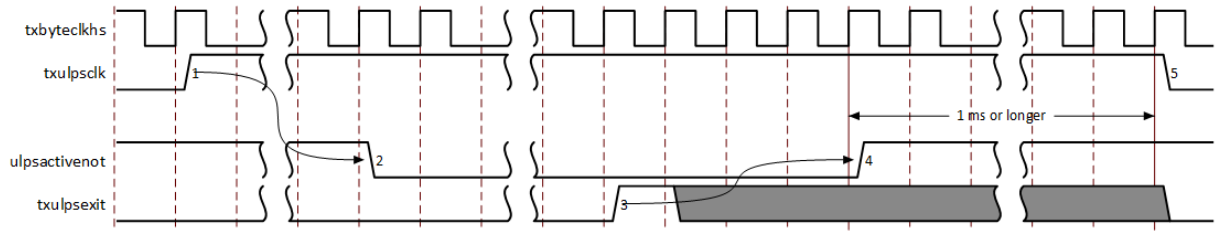
Example 5: D-PHY TX (Master) Clock Lane ULPS Operation

This section describes a TX clock lane ULPS operation.

1. The PPI drives `txulpsclk` to initiate the clock lane ULPS mode.
2. The MIPI D-PHY TX core drives the clock lane `ulpsactivenot` (active-Low) to Low after the ULPS entry sequence is transmitted on the serial line.
3. The PPI asserts the `txulpsexit` signal to exit from ULPS.
4. The MIPI D-PHY TX core drives the `ulpsactivenot` High and drives MARK-1 on the serial lines.
5. The PPI deasserts the `txrequestesc` after T_WAKEUP time has elapsed following deassertion of the `ulpsactivenot` signal.

The following figure shows the TX clock lane ULPS operation.

Figure 27: **D-PHY TX (Master) ULPS Mode Operation for Clock Lane**



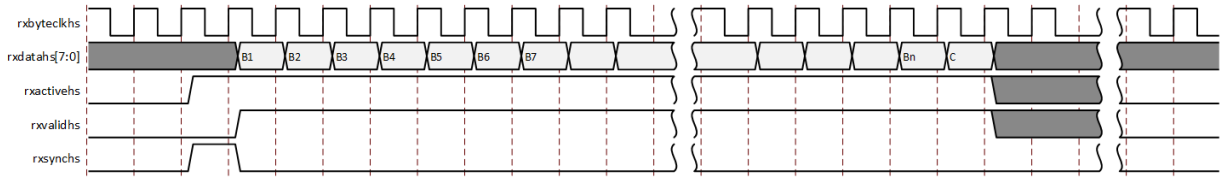
Example 6: High-Speed Receive at D-PHY RX (Slave) Side

This section describes a high-speed reception at the slave side PPI. This behavior is shown in the following figure.

The `rxactivehs` signal indicates that a receive operation is occurring. A normal reception starts with a pulse on `rxsynchs` followed by valid receive data on subsequent cycles of `rxbyteclkhs`. Note that the protocol is prepared to receive all of the data. There is no method for the receiving protocol to pause or slow data reception.

Because end-of-transmission (EoT) processing is not performed in the PHY, one or more additional bytes are presented after the last valid data byte. The first of these additional bytes, shown as byte “C” in the following figure, is either all 1s or all 0s. Subsequent bytes might or might not be present and can have any value. The `rxactivehs` and `rxvalidhs` signals transition Low simultaneously sometime after byte “C” is received. After these signals have transitioned Low, they remain Low until the next high-speed data reception begins.

Figure 28: **High-Speed Mode Data Receive at the D-PHY RX (Slave)**

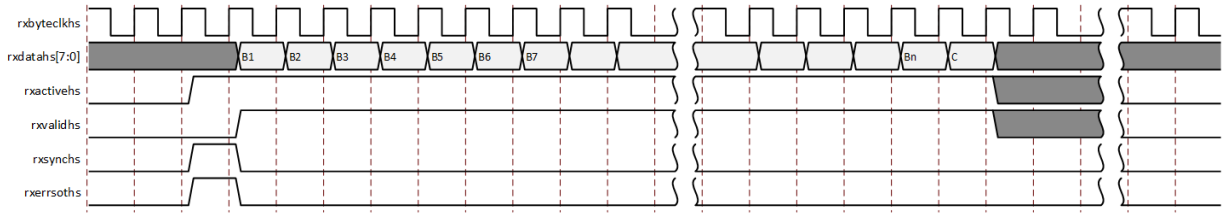


Note: D-PHY RX data lanes operate independently and the received high-speed data, from the serial lines, is passed to the higher layers through PPI. MIPI D-PHY RX IP does not perform any byte alignment or inter-lane skew between RX data lanes. It is the responsibility of the higher layer protocol cores. MIPI CSI-2 Receiver Subsystem compensates up to two `rxbyteclkhs` clock cycles between RX data lanes PPI High-Speed data.

Example 7: High-Speed Receive with Synchronization Error at D-PHY RX (Slave) Side

The MIPI D-PHY RX core can detect a start-of-transmission (SoT) pattern with single-bit error. It is reported by the assertion of `rxerrsoths` for one clock cycle of `rxbytehs` along with the `rxsynchs` pulse. This behavior is shown in the following figure.

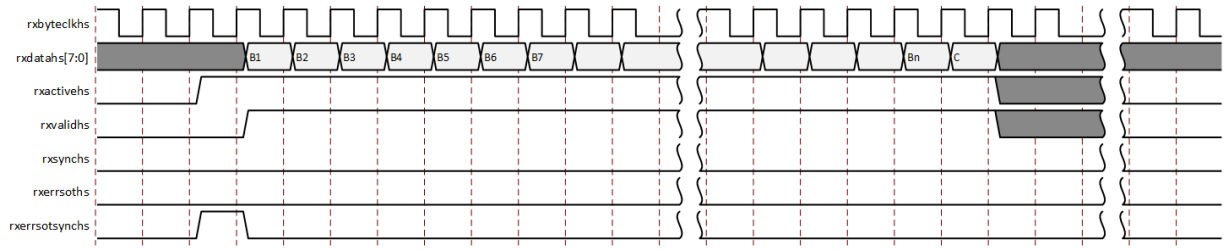
Figure 29: **High-Speed Mode Data Receive with Synchronization Error at the D-PHY RX (Slave)**



Example 8: High-Speed Mode Data Receive with Loss of Synchronization at D-PHY RX (Slave) Side

The MIPI D-PHY RX core reports the multi-bit error on the SoT pattern by asserting `rxerrsotsynchs` for one clock cycle of `rxbyteclkhs`. This scenario indicates that the SoT pattern is corrupted. Note that `rxsynchs` is not asserted. Received payload is passed on to the PPI. This behavior is shown in the following figure.

Figure 30: **High-Speed Mode Data Receive with Loss of Synchronization at the D-PHY RX (Slave)**



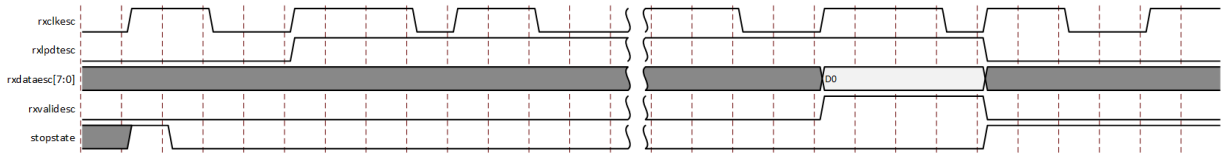
Example 9: Low-Power Receive at D-PHY RX (Slave) Side

The following figure shows a single-byte data reception in low-power mode.

- The `rxclkesc` signal is generated by the MIPI D-PHY RX core from the data lane interconnect.
- The signal `rxlpdtesc` is asserted by the MIPI D-PHY RX core when the LPDT entry command is detected and stays High until the data lane returns to the Stop state, indicating that the LPDT transmission has finished.

- `rxdataesc[7:0]` is valid when `rxvalidesc` is asserted High.

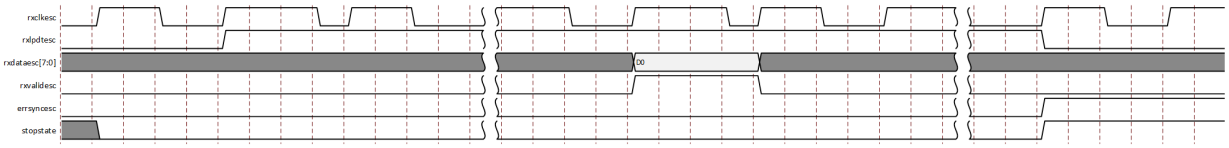
Figure 31: **Low-Power Data Reception at the D-PHY RX (Slave)**



Example 10: Low-Power Data Receive with Synchronization Error at D-PHY RX (Slave) Side

The MIPI D-PHY RX core reports an error to the PPI if the number of received valid bits during LPDT is not a multiple of eight. This is indicated by asserting `errsyncesc` along with `stopstate` and remains asserted until the next change in the serial line state. This behavior is shown in the following figure.

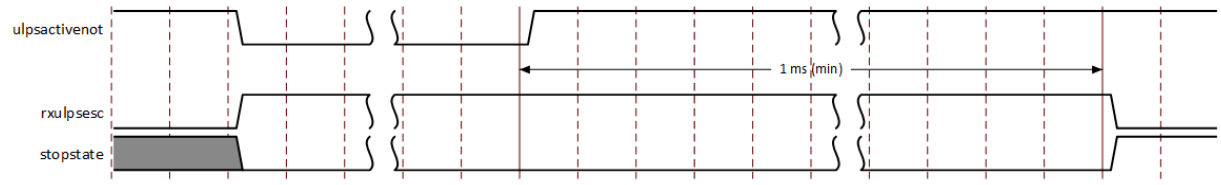
Figure 32: **Low-Power Data Reception with Synchronization Error at the D-PHY RX (Slave)**



Example 11: ULPS Operation at D-PHY RX (Slave) Data Lane

The RX Data lane ULPS entry is indicated by assertion of `rxulpsesc` along with assertion of `ulpsactivenot` (active-Low) signal. ULPS exit is marked by reception of MARK-1 on the line and `ulpsactivenot` is deasserted. After receiving MARK-1 for T_WAKEUP time (1 ms minimum), `rxulpsesc` is deasserted. This behavior is shown in the following figure.

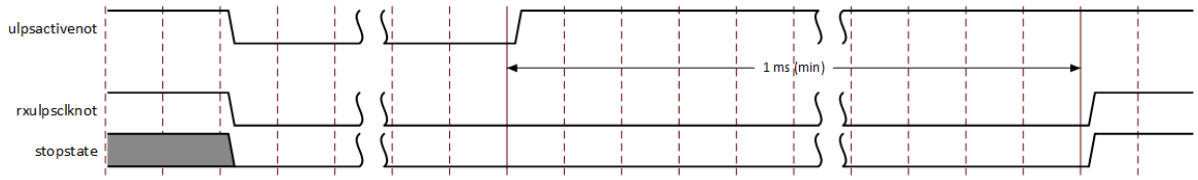
Figure 33: **D-PHY RX (Slave) ULPS Mode Operation for Data Lane**



Example 12: ULPS Operation at D-PHY RX (Slave) Clock Lane

The RX clock lane ULPS entry is indicated by assertion of `rxulpsclknot` (active-Low) along with assertion of `ulpsactivenot` (active-Low) signal. ULPS exit is marked by reception of MARK-1 on the line and `ulpsactivenot` is deasserted. After receiving MARK-1 for T_WAKEUP time (1 ms minimum), `rxulpsclknot` is deasserted. This behavior is shown in the following figure.

Figure 34: **D-PHY RX (Slave) ULPS Mode Operation for Clock Lane**

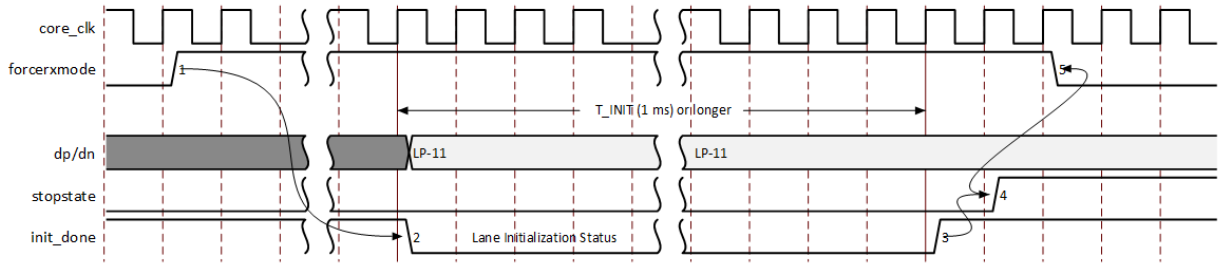


Example 13: RX Data Lane Initialization Using forcerxmode

The RX data lane can be initialized using the forcerxmode signal. This behavior is shown in the following figure.

1. `forcerxmode` is the asynchronous signal and is sampled using `core_clk`.
2. The `forcerxmode` assertion resets the lane initialization status, which is shown as the `init_done` signal in the waveform.
3. LP-11 should be driven on dp/dn serial lines for T_INIT or longer by the MIPI D-PHY TX (Master). This initializes the RX data lane.
4. `Stopstate` is driven High after lane is initialized.
5. `forcerxmode` can be deasserted by sampling `stopstate`.

Figure 35: **RX Data Lane Initialization Using forcerxmode**



***Note*:** Back channel communication is not available from the MIPI D-PHY RX (Slave) to the MIPI D-PHY TX (Master). Hence, you are responsible for making sure that MIPI D-PHY TX drives LP-11 on serial lines after `forcerxmode` is asserted on the MIPI D-PHY RX core module. Otherwise, the MIPI D-PHY RX core does not complete the initialization.

AXI4-Lite Interface

The register interface uses an AXI4-Lite interface, which was selected because of its simplicity. The following figures show typical AXI4-Lite write and read transaction timings.

Figure 36: **AXI4-Lite Write Timing Diagram**

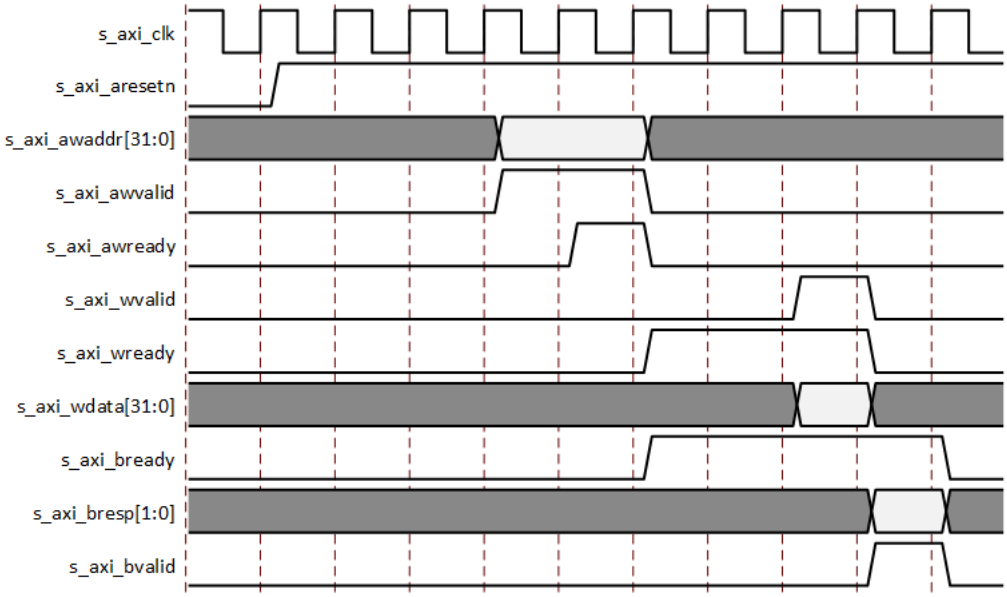
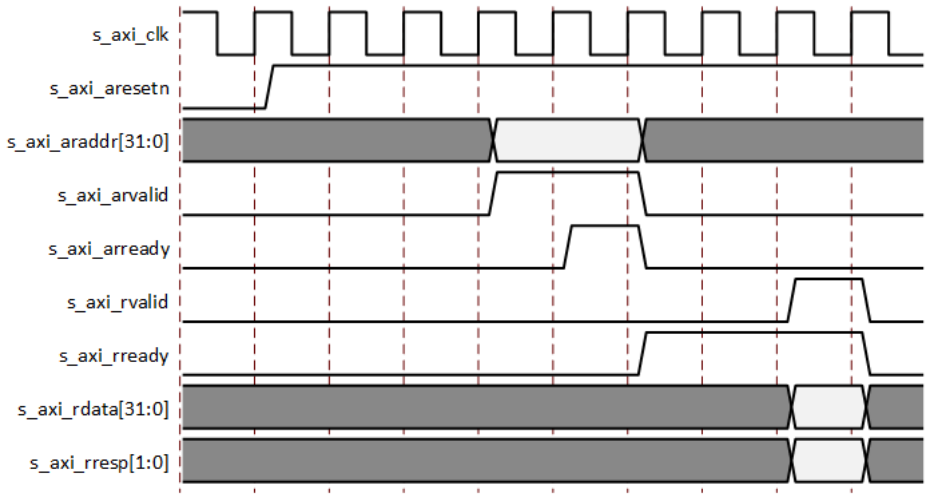


Figure 37: **AXI4-Lite Read Timing Diagram**



Design Flow Steps

This section describes customizing and generating the core, constraining the core, and the simulation, synthesis, and implementation steps that are specific to this IP core. 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)
- *Vivado Design Suite User Guide: Designing with IP* (UG896)
- *Vivado Design Suite User Guide: Getting Started* (UG910)
- *Vivado Design Suite User Guide: Logic Simulation* (UG900)

Customizing and Generating the Core

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

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) 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 IP core using the following steps:

1. Select the IP from the 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) and the *Vivado Design Suite User Guide: Getting Started* (UG910).

Figures in this chapter are illustrations of the Vivado IDE. The layout depicted here might vary from the current version.

Core Configuration Tab

The following figure shows the Core Configuration tab for customizing the MIPI D-PHY Controller.

Figure 38: **Core Configuration Tab for D-PHY TX**

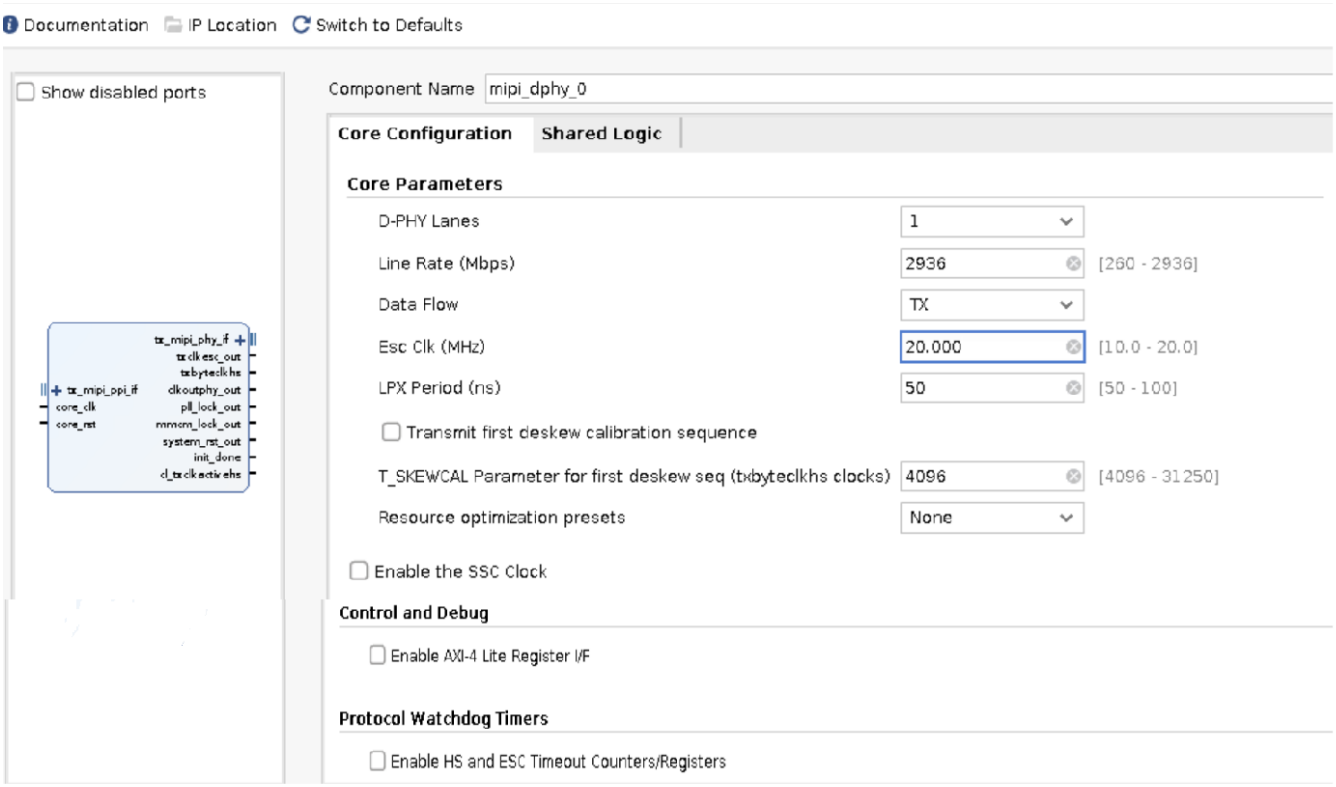
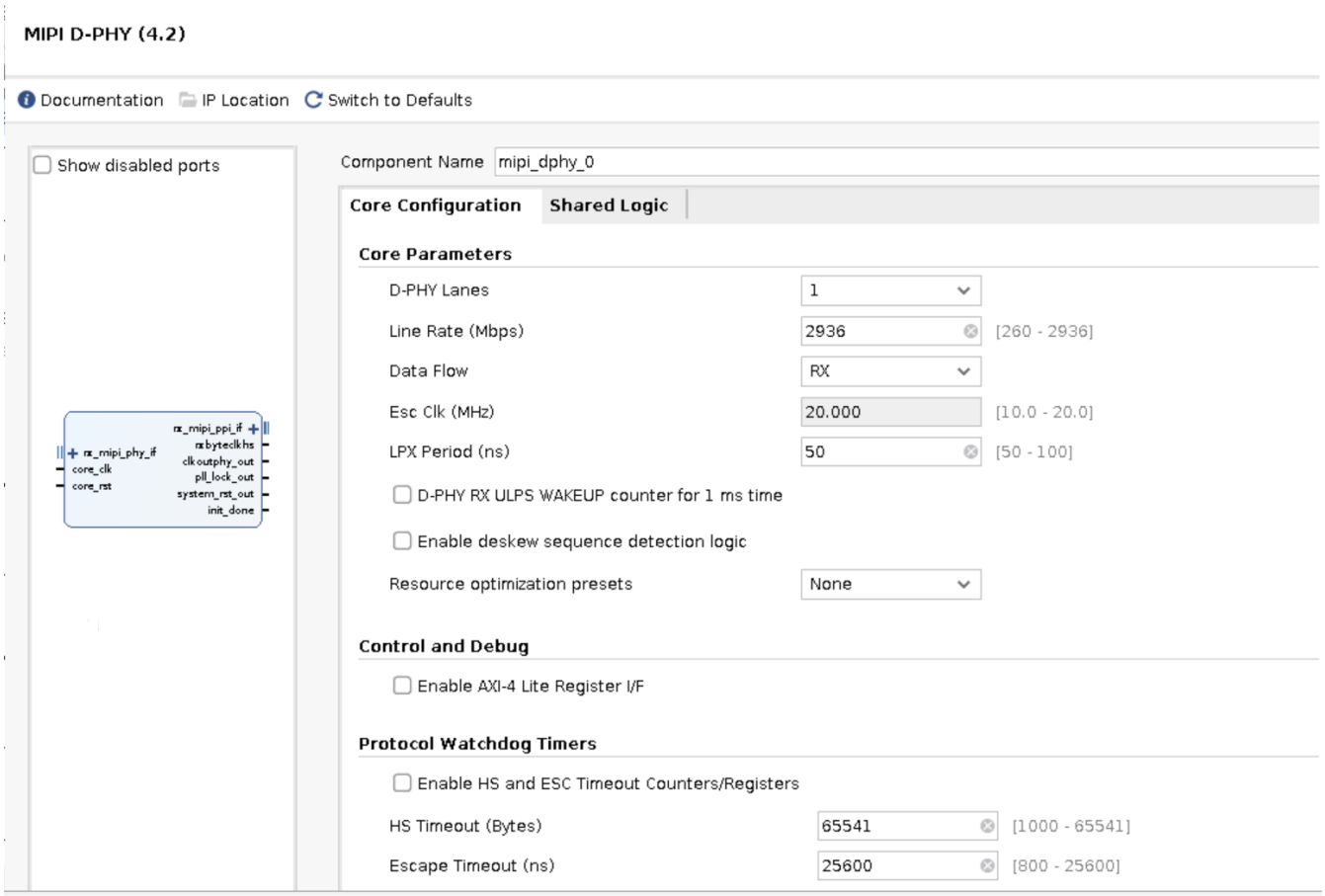


Figure 39: **Core Configuration Tab for D-PHY RX**



Component Name

The Component Name is the base name of the output files generated for this core.

IMPORTANT! *The name must begin with a letter and be composed of the following characters: a to z, A to Z, 0 to 9 and "_".*

Core Parameters

- **D-PHY Lanes:** Select the number of data lanes to be used in the core. The valid range for TX is from 1 to 4, and for RX is from 1 to 8.
- **Line Rate:** Enter a line rate value in megabits per second (Mb/s) within the valid range: 80 to 2936 Mb/s based on the device selected. The Vivado IDE automatically limits the line rates based on the device selected. For details about family/device specific line rate support, refer to the respective device data sheet.
- **Data Flow:** Select the options for the direction of the data transfer. Available options are TX (for Master) and RX (for Slave).

- **Enable Splitter Bridge Mode:** Select this to enable splitter bridge mode. This allows to replicate received MIPI camera stream to multiple (1 to 4) MIPI output streams for further processing by external modules.
- **Number of TX Interfaces:** Select the number MIPI output TX interfaces when Enable Splitter Bridge Mode is selected. You can select up to four TX interfaces. Based on number of TX interfaces selected, the GUI allows I/O configuration for all these interfaces separately. I/Os of each TX interface can be in the same bank or a different bank.

  Note: You must make sure that the all TX I/Os are exclusive and follow IO guidelines.
- **Escape Clk (MHz):** Enter a valid escape clock frequency in MHz into the text box for the MIPI D-PHY Master (TX) core. The valid range is from 10.000 to 20.000 MHz. Applicable only for the MIPI D-PHY TX core.
- **LPX Period (ns):** Enter a valid LPX Period in nanoseconds (ns) into the text box for MIPI D-PHY Master (TX) core. The valid range is from 50 to 100 ns.
- **D-PHY RX ULPS WAKEUP counter for 1 ms time:** Select the option to include 1 ms WAKEUP counter. Otherwise, D-PHY RX IP checks only for the LP-10 transition to exit from the ULPS mode.

  Note: Available only for D-PHY RX configuration.
- **Resource Optimization presets:** By using the mentioned presets, you can reduce the resources depending upon the requirements.

  If preset CSI2RX_XLNX is selected, the ULPS and LPDT features are not supported by the core.

  If preset CSI2RX_XLNX2 is selected, the ULPS and LPDT features, `errorsotsynchs` assertion, and the checking of the LPX period are not supported by the core.

  If preset `CSI2TX_XLNX` is selected, the ULPS and ESC features are not supported by the core; furthermore, the register interface is removed, and the clock and data lane status information is provided through ports.
- **IODELAY_GROUP Name:** This parameter is used to select the IODELAY_GROUP name for the IDELAYCTRL. All core instances in the same bank sharing IDELAYCTRL should have the same name for this parameter. Select a unique name per bank.

  Note: Available only for 7 series D-PHY RX configuration.
- **Enable deskew sequence detection logic:** This parameter is used to enable the deskew detection logic. When a deskew packet is received, D-PHY does the eye centering between clock and data.

  Note: The minimum required length of the periodic calibration pattern is 2^{13} UI.

- **Enable the SSC Clock:** This parameter is used when the SSC feature is required. When this parameter is selected, you must drive the SSC enabled byte clock (`ssc_byteclkhs_in`) to the core when the line rate is greater than 2500 Mb/s and when the shared logic is inside the core.
- **Transmit First Deskew Calibration Sequence:** This parameter is used to enable the initial deskew pattern in D-PHY TX configuration.
- **T_SKEWCAL Parameter for first deskew seq (txbyteclkhs clocks):** This parameter defines length of the initial calibration sequence.
- **Transmit Periodic deskew calibration sequence:** This parameter is used to enable the periodic deskew pattern in D-PHY TX configuration.

  ***Note*:** The length of the periodic pattern depends on the length of `dl<n>_txskewcalhs`.

Control and Debug

- **Infer OBUFTDS for 7 series HS outputs:** Select this option to infer OBUFTDS for HS outputs.

  ***Note*:** This option is available only for 7 series D-PHY TX configuration. It is recommended to use this option for D-PHY compatible solution based on resistive circuit. For details, see *D-PHY Solutions* (XAPP894).

- **Enable Active Lane support:** Select this option to control TX data lanes. Active lanes allows the D-PHY TX to run with lower lanes than IP is configured for. This helps the lane down scaling and disabling any TX data lane by the deasserting corresponding bit in the `active_lanes_in` bus input. It is recommended to update the `active_lanes_in` when all data lanes are in stopstate. HS_TX_TIMEOUT is disabled internally when the `active_lanes_in` feature is exercised. Lane 0 is always enabled (otherwise, `txreadyhs` is not asserted).
- **Enable AXI4-Lite Register I/F:** Select the AXI4-Lite based register interface for control and debug purposes.

Protocol Watchdog Timers

- **Enable HS and ESC Timeout Counters/Registers:** Enable the HS_TX_TIMEOUT/ HS_RX_TIMEOUT and ESC_TIMEOUT counters. Select this option to enable the HS_TIMEOUT and ESC_TIMEOUT registers provided that the AXI4-Lite register interface is enabled.
- **HS Timeout (Bytes):** Enter the maximum transmission or reception length in bytes for High-Speed mode. The valid range is from 1,000 to 65,541 bytes.
- **Escape Timeout (ns):** Enter the maximum transmission or reception length in ns for LPDT escape mode. The valid range is from 800 to 25,600 ns.
- **Calibration Mode:** Select the calibration for 7 series D-PHY RX IP. Available options:
  - None (default selection) - Does not add IDELAYE2 primitive.

  - Fixed - Sets the IDELAYE2 TAP value given in the IDELAY Tap Value.
  - Auto - Adds the IDELAYE2 primitive. IDELAY Tap Value will be configured by D-PHY RX IP based on received traffic and calibration algorithm. IP uses the DIFF_TERM=TRUE setting for input buffers when Calibration mode is set to Auto. Auto algorithm performs a skew calibration on the run time. It usually requires few HS packet reception by D-PHY RX IP to determine the correct IDELAY tap value. This mode is available for line rates above 450 Mb/s.

- **IDELAY Tap Value:** Enter IDELAY TAP value used calibration in fixed mode. The valid range is from 1 (default option) to 31.
- **Include IDELAYCTRL in core:** For multiple D-PHY RX IP cores that are sharing single IO bank, select this option to include IDELAYCTRL in the IP for the auto calibration mode. Only one IDELAYCTRL is available per I/O bank. In case of multiple D-PHY RX cores in single I/O bank, only one D-PHY RX IP core should have this option selected. For the rest of D-PHY RX cores, this option should be unselected.

  ***Note*:** This option is applicable only for 7 series D-PHY RX IP configuration.

- **Enable 300 MHz clock for IDELAYCTRL:** Select this option to connect 300 MHz to IDELAYCTRL and is used in auto calibration mode.

  ***Note*:**

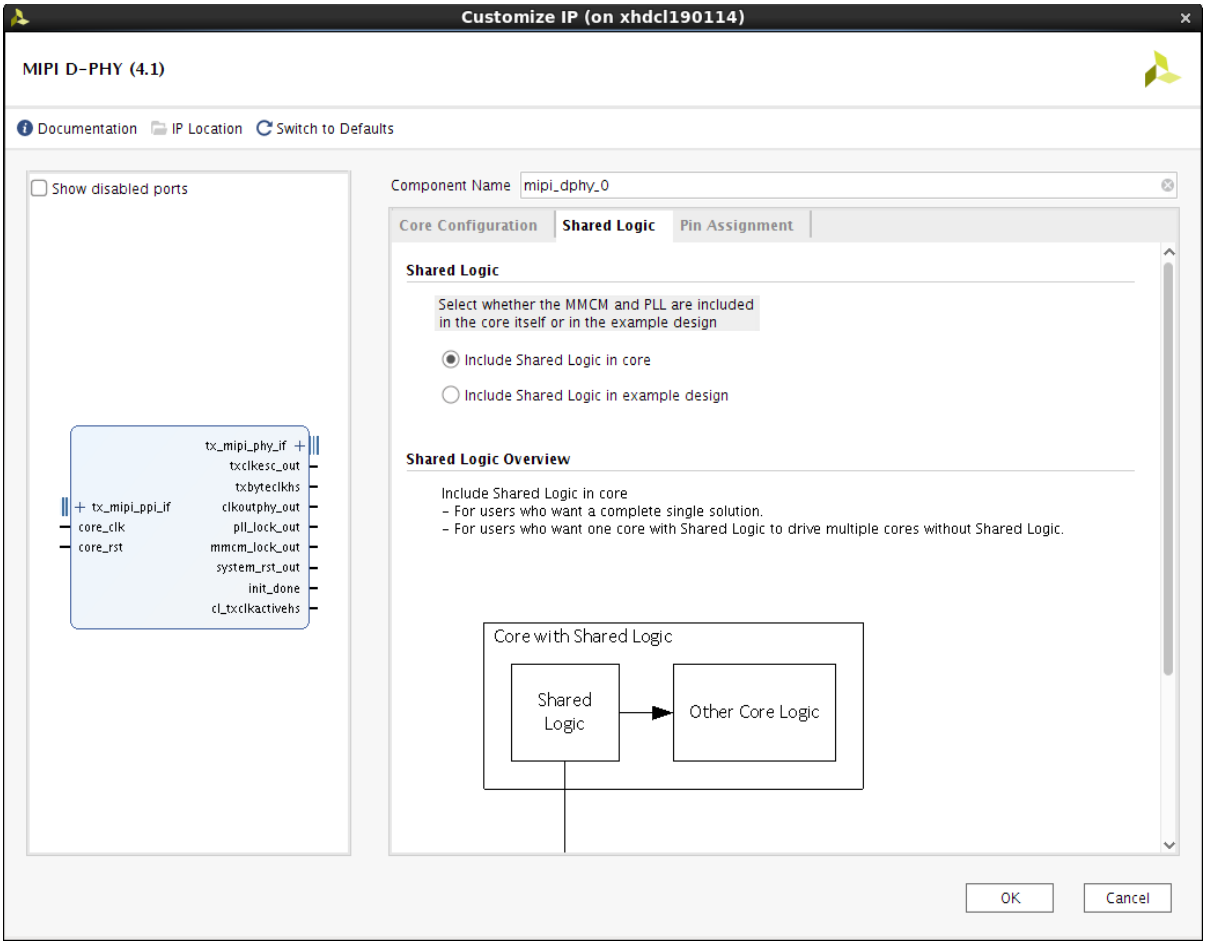
  - This option is applicable only for 7 series D-PHY RX IP configuration.
  - For 7 series in AUTO Mode, when there are multiple instances of DPHY and they share the IDELAY control ready from one DPHY instance to other DPHY instance. The DPHY instance which shares the IDELAY controller ready cannot have theEnable 300 MHz clock for IDELAYCTRL parameter set to true.
  - For 7 series, the IP is tested and validated for max 1250 Mb/s. Line rates higher than 1250 Mb/s are available in the GUI for the users whose setup can scale to higher line rates.

Shared Logic Tab

The following figure shows the Shared Logic tab of the Customize IP interface.

***Note*:** This tab is not available for 7 series D-PHY RX configuration.

Figure 40: **Shared Logic Tab**



This tab allows you to select whether the MMCM and PLL are included in the core or in the example design. Following are the available options:

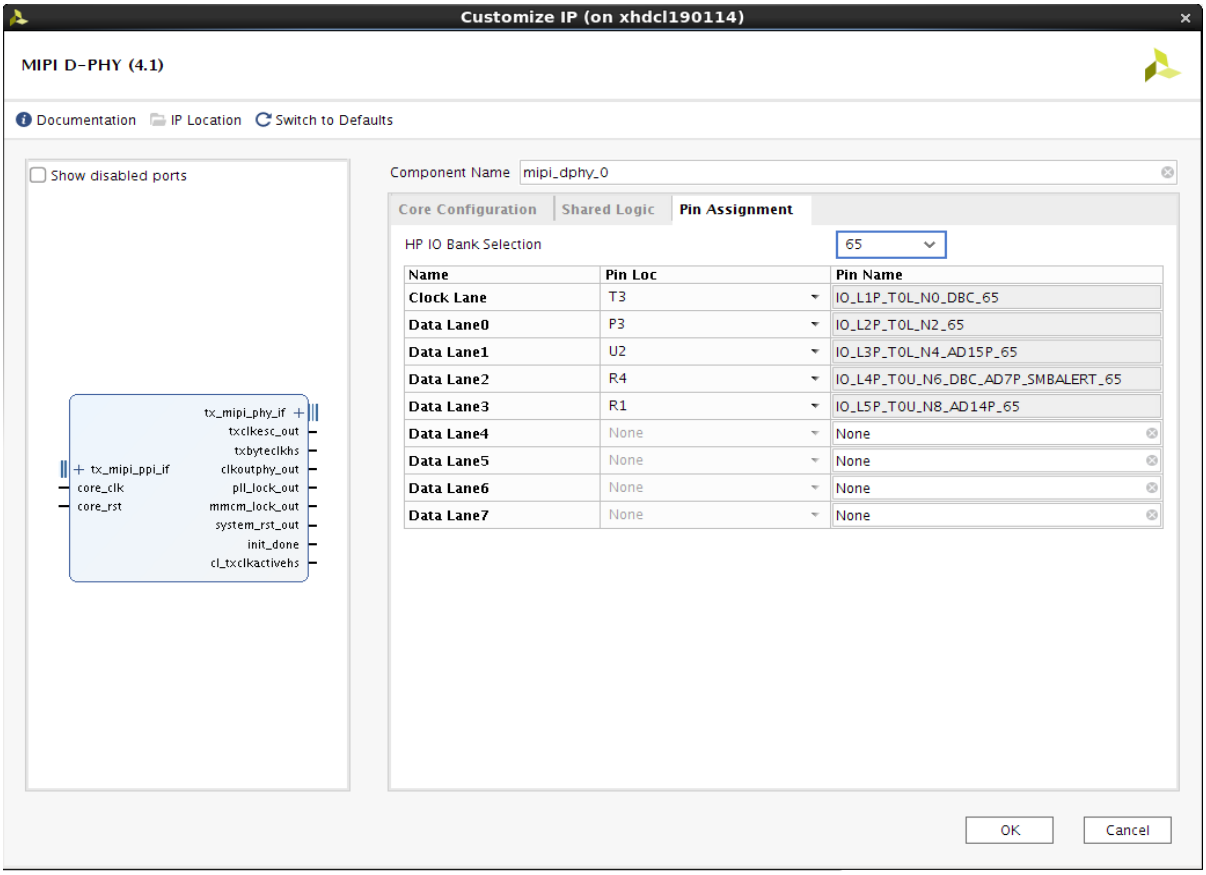
- Include Shared Logic in core
- Include Shared Logic in example design (default selection)

Pin Assignment Tab

The following figure shows the I/O pin parameters for the core. For more information on the optimal IO pin assignment, see the Appendix C, Pin and Bank Rules.

***Note*:** This tab is not available for 7 series and Versal D-PHY RX configuration.

Figure 41: **Pin Assignment Tab**



- **HP IO Bank Selection:** Select the HP I/O bank for clock lane and data lane implementation.

  Note: This option is not available for 7 series FPGAs as D-PHY can be implemented in both HR bank IO and HP bank IO.

- **Clock Lane:** Select the LOC for clock lane. This selection determines the I/O byte group within the selected HP I/O bank.

- **Data Lane 0/1/2/3:** This displays the Data lane 0, 1, 2, and 3 LOC based on the clock lane selection.

User Parameters

The following table shows the relationship between the fields in the Vivado® IDE and the user parameters (which can be viewed in the Tcl Console).

Table 32: **Vivado IDE Parameter to User Parameter Relationship**

Vivado IDE Parameter/Value	User Parameter/Value	Default Value
Core Parameters		
D-PHY Lanes	C_DPHY_LANES	1
Line Rate (Mb/s)	C_LINE_RATE	1,000
Data Flow Mode	C_DATA_FLOW	Master (TX)
Escape Clk (MHz)	C_ESC_CLK_PERIOD	20.000
LPX (ns)	C_LPX_PERIOD	50
D-PHY RX ULPS WAKEUP counter for 1ms time	C_EN_ULPS_WAKEUP_CNT	False
IODELAY_GROUP name	C_IDLY_GROUP_NAME	mipi_dphy_idly_group
Enable the SSC clock	C_EN_SSC	False
Protocol Watchdog Timers		
Enable Deskew Sequence detection Logic	C_RCVE_DESKEW	0
Enable HS and ESC timeout counters/ Registers	C_EN_TIMEOUT_REGS	0
HS Timeout (Bytes)	C_HS_TIMEOUT	65,541
Escape Timeout (ns)	C_ESC_TIMEOUT	25,600
Debug and Control		
Enable Register Interface	C_EN_REGIF	0
OBUFTDS Inference	C_EN_HS_OBUFTDS	0
Active Lane Support	C_EN_ACT_LANES	0
HS_SETTLE Parameter (ns)	C_HS_SETTLE_NS	145

Output Generation

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

Constraining the Core

Required Constraints

This section defines the additional constraint requirements for the core. Constraints are provided with a Xilinx® Design Constraints (XDC) file. An XDC is provided with the HDL example design to give a starting point for constraints for your design.

Device, Package, and Speed Grade Selections

This section is not applicable for this IP core.

Clock Frequencies

`core_clk` should be specified as follows:

```
create_clock -name core_clk -period 5.000 [get_ports core_clk]
```

This constraint defines the frequency of `core_clk` that is supplied to the MMCM and PCS logic.

Clock Management

The MIPI D-PHY Controller uses an MMCM to generate the general interconnect clocks, and the PLL is used to generate the serial clock and parallel clocks for the PHY. The input to the MMCM is constrained as shown in Clock Frequencies. No additional constraints are required for the clock management.

Clock Placement

This section is not applicable for this IP core.

Banking

The MIPI D-PHY Controller provides the Pin Assignment Tab option to select the HP I/O bank. The clock lane and data lane(s) are implemented on the selected I/O bank BITSLICE(s).

***Note*:** Pin assignment is not applicable for 7 series FPGAs and Versal DPHY IP configurations.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

MIPI standard serial I/O ports should use MIPI_DPHY_DCI for the I/O standard in the XDC file for UltraScale+™ families. The LOC and I/O standards must be specified in the XDC file for all input and output ports of the design. UltraScale+ MIPI D-PHY IP generates the IO pin LOC for the pins that are selected during IP customization. No IO pin LOC are provided for 7 series MIPI D-PHY IP designs. You have to manually select the clock capable IO for 7 series RX clock lane and restrict the IO selection within the IO bank for both D-PHY TX and D-PHY RX IP configurations.

It is recommended to select the IO bank with `VRP` pin connected for UltraScale+ MIPI D-PHY TX IP core. If `VRP` pin is present in other IO bank in the same IO column of the device, the following DCI_CASCADE XDC constraint should be used. For example, IO bank 65 has a `VRP` pin and the D-PHY TX IP is using the IO bank 66.

```
set_property DCI_CASCADE {66} [get_iobanks 65]
```

For more information on MIPI_DPHY_DCI IO standard and `VRP` pin requirements, see the *UltraScale Architecture SelectIO Resources User Guide* (UG571).

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).

Synthesis and Implementation

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

Chapter 6

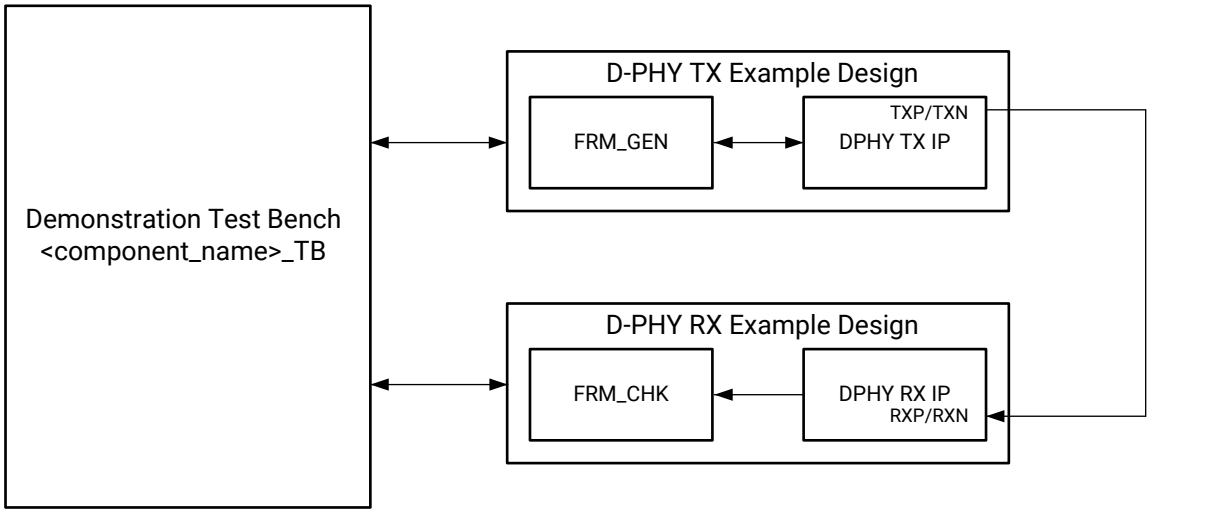
Example Design

This chapter contains information about the example design provided in the Vivado® Design Suite.

Overview

The top module instantiates all components of the core and example design that are needed to implement the design in hardware, as shown in the following figure. This includes the FRM_GEN, DPHY TX IP, FRM_CHK, and the DPHY RX IP modules.

Figure 42: **MIPI D-PHY Core Example Design**



The FRM_GEN module generates user traffic for High-Speed mode and low-power data transmission (LPDT). This module contains a pseudo-random number generator using a linear feedback shift register (LFSR) with a specific initial value to generate a predictable sequence of data.

The FRM_CHK module verifies the integrity of the RX data. This module uses the same LFSR and initial value as the `FRM_GEN` module to generate the expected RX data. The received user data is compared with the locally-generated data and an error is reported if data comparison fails. The example design can be used to quickly get an MIPI D-PHY Controller design up and running on a board, or perform a quick simulation of the module. When using the example design on a board, be sure to edit the `<component name>_exdes.xdc` file to supply the correct pins and clock constraints.

IMPORTANT! *This implementation is used only for reference and as a demonstration of the example test bench.*

Simulating the Example Design

For more information about simulation, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900).

The simulation script performs the following:

1. Compiles the MIPI D-PHY example design and supporting simulation files.
2. Runs the simulation.
3. Runs checks to ensure that it completed successfully.

If the test passes, the following message is displayed:

```
MIPI_D-PHY_TB : INFO: Test Completed Successfully
```

If the test fails, the following message is displayed:

```
MIPI_D-PHY_TB : ERROR: Test Failed
```

If the test hangs, the following message is displayed:

```
MIPI_D-PHY_TB : ERROR: Test did not complete (timed-out)
```

Test Bench

This chapter contains information about the example design provided in the Vivado® Design Suite.

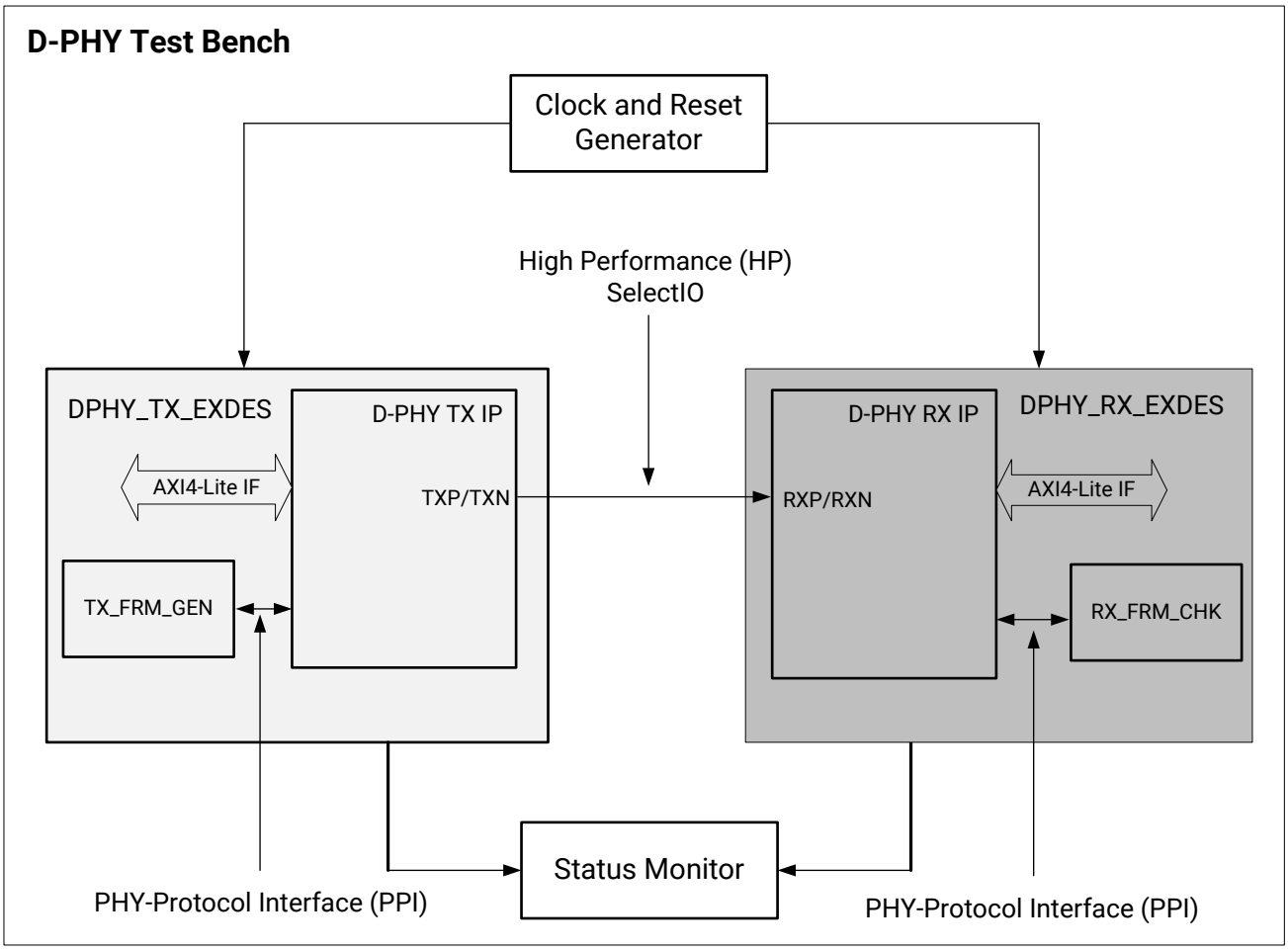
The MIPI D-PHY Controller delivers a demonstration test bench for the example design. This chapter describes the MIPI D-PHY Controller test bench and its functionality. The test bench consists of the following modules:

- Device Under Test (DUT)
- Clock and reset generator
- Status monitor

The example design demonstration test bench is a simple Verilog module to exercise the example design and the core itself. It simulates an instance of the MIPI D-PHY TX example design that is externally looped back to the MIPI D-PHY RX example design. The following figure shows the MIPI D-PHY Controller test bench where `DUT1` is configured as `D-PHY TX`, and `DUT2` is configured as `D-PHY RX`.

The MIPI D-PHY Controller test bench generates all the required clocks and resets, and waits for successful data pattern checking to complete. If it fails to detect successful data pattern checking, it produces an error.

Figure 43: **MIPI D-PHY Test Bench**



Appendix A

Verification, Compliance, and Interoperability

The MIPI D-PHY Controller has been verified using both simulation and hardware testing. A highly parameterizable transaction-based simulation test suite has been used to verify the core. The tests include:

- High-Speed data transmission
- High-Speed data reception
- Low-Power data transmission (LPDT)
- LPDT data reception
- Clock lane Ultra-Low Power State (ULPS) operation
- Data lane ULPS operation
- Triggers and escape mode commands
- Recovery from error conditions
- Register read and write access

Hardware Validation

The MIPI D-PHY Controller is tested in hardware for functionality, performance, and reliability using Xilinx® evaluation platforms. The MIPI D-PHY Controller verification test suites for all possible modules are continuously being updated to increase test coverage across the range of possible parameters for each individual module.

A series of MIPI D-PHY Controller test scenarios are validated using the Zynq® UltraScale+™ MPSoC ZCU102 development board. This board allows the prototyping of system designs where the MIPI D-PHY Controller is used for high-speed serial communication between two boards.

7 series FPGAs do not have native MIPI IOB support: target the HP/HR IO bank for 7 series FPGAs and the XPIO bank for Versal™ devices.

For more information, refer *D-PHY Solutions* (XAPP894)

A series of interoperability test scenarios are listed in the following table that are validated using different core configurations and resolutions.

Table 33: **MIPI CSI2 Sensor Interoperability Testing**

Sensor	Board/Device	Tested Configuration	Resolution
Omnivision OV13850	ZCU102/xczu9eg-ffvb1156-2-i-es2	D-PHY RX 1200 Mb/s 1, 2, 4 Lanes	480p@60fps, 720p@60fps, 1080p@60fps, 4k@30fps
Sony IMX274	ZCU102/xczu9eg-ffvb1156-2-i-es2	D-PHY RX 1440 Mb/s 4 Lanes	All supported modes by sensor
Sony IMX224	ZCU102/xczu9eg-ffvb1156-2-i-es2	D-PHY RX 149 Mb/s, 594 Mb/s 1, 2, 4 Lanes	All-pixel (QVGA) and Window cropping modes
Sony IMX274	ZC702/xc7z020clg484-1	D-PHY RX 576 Mb/s 4 Lanes	1080p@60fps

The following table lists the interoperability test, validated using the MIPI DSI display.

Table 34: **MIPI DSI Display Interoperability Testing**

Sensor	Board/Device	Tested Configuration	Resolution
B101UAN01.7	ZCU102/xczu9eg-ffvb1156-2-e	D-PHY TX 1000 Mb/s 4 Lanes	1920x1200@60fps

XILINX

Appendix B

Debugging

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

Finding Help on Xilinx.com

To help in the design and debug process when using the core, 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. The Xilinx Community Forums are also available where members can learn, participate, share, and ask questions about Xilinx solutions.

Documentation

This product guide is the main document associated with the core. 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 core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use 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 Core

AR 54550.

Technical Support

Xilinx provides technical support on the Xilinx Community Forums for this LogiCORE™ 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.

To ask questions, navigate to the Xilinx Community Forums.

Debug Tools

There are many tools available to address MIPI D-PHY v4.2 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 to interact with the logic debug LogiCORE IP cores, including:

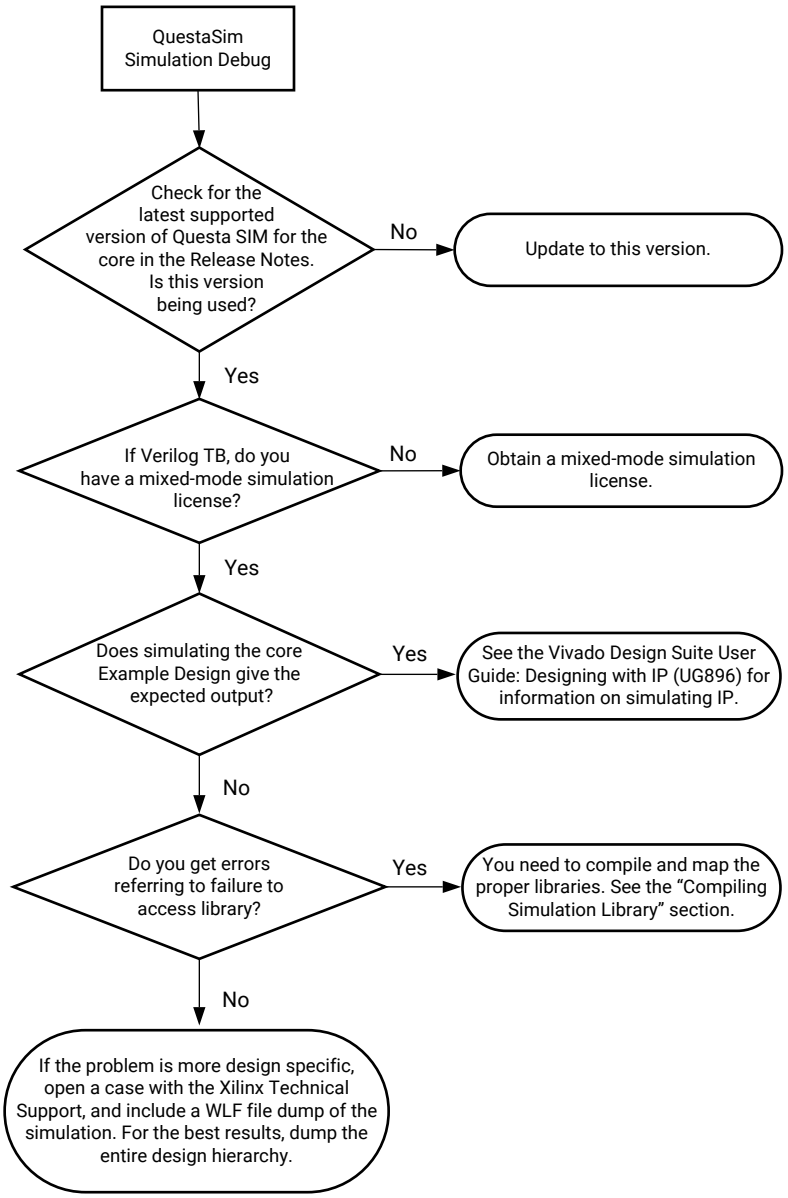
- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

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

Simulation Debug

The simulation debug flow for Mentor Graphics Questa Advanced Simulator is illustrated in the following figure. A similar approach can be used with other simulators.

Figure 44: **Questa Simulation Debug Flow**



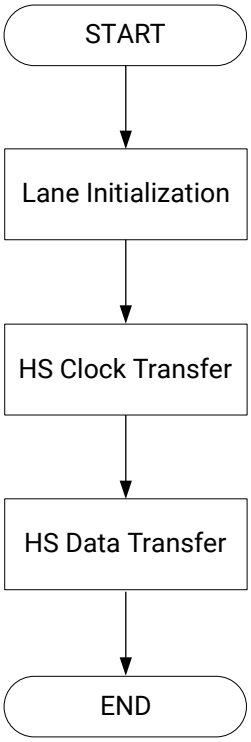
X14842-012616

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.

The following figure shows the steps to perform a hardware debug.

Figure 45: **Debug Flow Chart**



General Checks

- Ensure that all the timing constraints for the core were properly incorporated from the example design and that all constraints were met during implementation.
- Ensure that MMCM and PLL have obtained lock by monitoring `mmcm_lock_out` and `pll_lock_out` ports respectively.
- Verify the IO pin planning and XDC constraints.
- Follow recommended reset sequence.
- Verify all clocks are connected and are with expected frequencies.

- Enable AXI4-Lite based register interface to get core status and control.
- Make sure serial line trace lengths are equal. For PCB Guidelines refer *UltraScale Architecture PCB Design User Guide* (UG583)
- Verify the FMC_VADJ voltage to 1.2V in case of FMC card usage.

D-PHY Protocol Checks

- Ensure that `HS` and `Escape` transactions are initiated when core is in `StopState`.
- Check the enable from PPI is connected and it is active-High during operation.
- Ensure bytes transferred or received are within `HS_TIMEOUT` in case of `HS` mode and `ESC_TIMEOUT` in case of `LPDT`.
- Ensure HS_SETTLE of D-PHY RX matches with the HS_PREPARE + HS_ZERO of D-PHY TX.
- Check received LP transactions are at least of 20 ns duration or more.
- Monitor PPI error signals such as `errsoths` and `errsotsynchs`. Excessive `errsotsynchs` indicates either HS_SETTLE parameter tuning or signal integrity issues on the D-PHY RX link.
- Ensure that there is no skew between different D-PHY RX lanes within same MIPI D-PHY interface. D-PHY RX IP does not perform any inter-lane skew adjustment operations on the received high-speed data. This is left to a higher level protocol layer such as CSI-2 RX.

Lane Initialization

After the assertion of power-on reset, MMCM lock followed by PLL lock should be asserted by the core. Monitor the `mmcm_lock_out` and `pll_lock_out` signals for the lock status. The serial lines of clock lane and data lane(s) should be driven with LP-11 for a period of T_INIT. The T_INIT value of the D-PHY RX should be 50% to 80% of the T_INIT value of the D-PHY TX. Bit 3 of the CL_STATUS or DL_STATUS registers confirm the completion of initialization. When the D-PHY core completes the initialization, `stopstate` is asserted on the PPI. Bit 4 of the CL_STATUS register and bit 6 of the DL_STATUS register indicate the Stop state.

HS Clock Transfer

The high-speed clock is transmitted on the D-PHY TX clock lane. The assertion of `txrequesths` on the TX clock lane starts the clock transmission. A value of 2'b01 in the MODE field of the CL_STATUS register confirms the HS clock transfer. The `cl_rxclkactivehs` PPI signal also can be used to confirm the HS clock reception in the D-PHY RX.

HS Data Transfer

HS data can be transferred as soon as the HS clock transmission has started. The `txrequesths` signal on the TX data lane starts the data transfer. A value of 2'b01 in the MODE field of the DL_STATUS register confirms that the data lane is in HS mode. The PKT_CNT field of the DL_STATUS register provides the numbers of packets transmitted or received by the data lane. The HS mode PPI signals can also be used to monitor the HS data transfer. Each `txrequesths` is counted as one packet in the D-PHY TX and each `rxactivehs` with a `rxsynchs` pulse is considered as one packet in the D-PHY RX. Note that the D-PHY RX also counts erroneous transactions such as `errsoths` and `errsotsynchs`.

You can start with a small number of packets from the D-PHY TX and check whether the PKT_CNT of both the D-PHY TX and D-PHY RX match. Ensure that all of the control mode sequences are captured without any errors and that the `errcontrol` signal of the PPI RX is asserted if any erroneous control sequence is received on the serial lines. The HS_ABORT field in the DL_STATUS register is asserted if the D-PHY RX is receiving more bytes than the HS_TIMEOUT programmed value.

Monitor `errsoths` and `errsotsynchs` and tune the HS_SETTLE of D-PHY RX IP after making sure that there are no signal integrity issues. HS_SETTLE of D-PHY RX can be changed through AXI-4 lite register interface and user can set desired value of HS_SETTLE during IP generation using HS_SETTLE_NS hidden user parameter.

AXI4-Lite Interface Debug

Read from a register that does not have all 0s as a default to verify that the interface is functional. 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 `s_axi_aclk` and `aclk` inputs are connected and toggling.
- The interface is not being held in reset, and `s_axi_areset` is an active-Low reset.
- The interface is enabled, and `s_axi_aclken` is active-High (if used).
- The main core clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or a Vivado Design Suite debug feature captures that the waveform is correct for accessing the AXI4-Lite interface.

XILINX

Appendix C

Pin and Bank Rules

This appendix provides guidelines and recommendations to implement multiple D-PHY interfaces on supported Xilinx® devices.

For more information on pin and bank rules, see the *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320).

Pin Rules for Zynq UltraScale+ MPSoC Devices

This section describes the pin rules for Zynq® UltraScale+™ MPSoC devices.

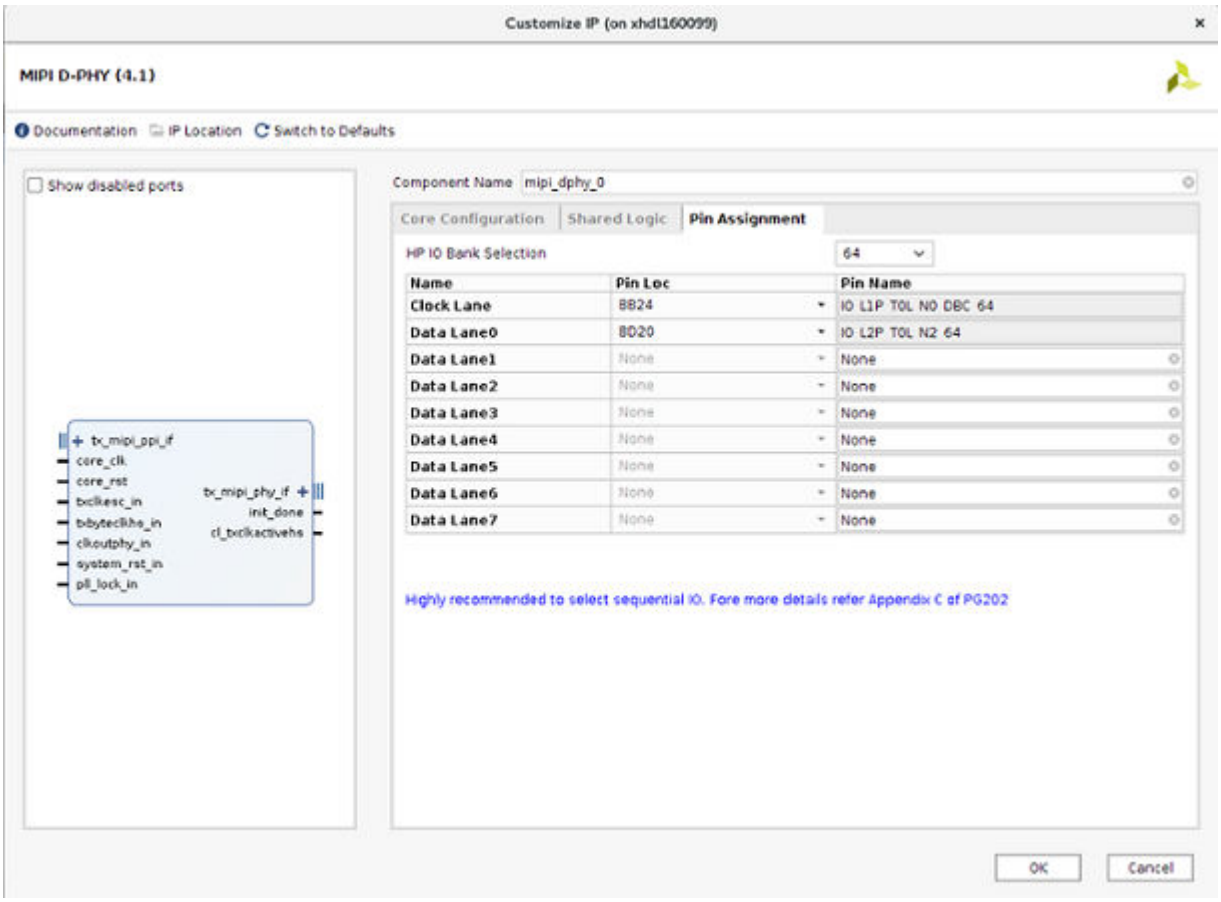
- Clock lane pins are represented with clk_<> and data lane pins are represented with data_<>.
- D-PHY Interface are numbered from if0 to if7.
- Byte lanes in a bank are designed by T0, T1, T2, or T3. Nibbles within a byte lane are distinguished by a "U" or "L" designator added to the byte lane designator (T0, T1, T2, or T3). Thus, they are T0L, T0U, T1L, T1U, T2L, T2U, T3L, and T3U.
- Pins in a byte lane are numbered from 0 to 12.

  ***Note*:** There are two PLLs per bank and a D-PHY uses one PLL in every bank that is being used by the interface.

D-PHY RX Pin Rules

- RX clock lane pins must be DBC, QBC and GC_QBC pins.
- Select the IO pins continuously without leaving any IO pairs in the middle of D-PHY interface. The following figure shows the warning message that appears in the GUI in case of non-continuous pin assignment.

Figure 46: **Warning Message in the GUI for Non-Continuous Pin Assignment**



- D-PHY IP uses IO in Native mode. Using the left out IO's in the nibble is not recommended, in case if its inevitable refer to "Mixing Native and Non-Native Mode I/O in a Nibble" section in *UltraScale Architecture SelectIO Resources User Guide* (UG571).
- HSSIO internally uses few IO under certain IO selection scenarios for Strobe propagation and this can be avoided by selecting IO continuously. Pin(s) used for Strobe propagation will be DBC, QBC or GC_QBC and it will restrict you to implement the multiple D-PHY interfaces.
- D-PHY with two different line rates can be implemented within IO bank and each D-PHY interface will use one PLL.
- All the lanes of a particular MIPI D-PHY instance need to be in the same HP IO bank, which the Pin Assignment tab of XGUI automatically controls for UltraScale+ devices.
- In case of multiple D-PHY instances sharing clocking resources, all such instances also need to be in the same HP IO bank.
- Any IO being placed along with D-PHY interface should have DCI IO standard since D-PHY IO uses MIPI_DPHY_DCI IO Standard.
- IO used for data lanes can be swapped in any order by keeping RX clock lane IO LOC unchanged.

- Initialize all MIPI interfaces in the same HP IO Bank at the same time. For example, multiple D-PHY instances in a system. For more information on implementing multiple interfaces in the same HP IO Bank, see *UltraScale Architecture SelectIO Resources User Guide* (UG571)

The following table shows an example of a four 4-lane D-PHY interface that be implemented in a single HP IO bank.

Table 35: **4x4-lane D-PHY Interface**

Interface	Signal Name	Byte Group	Pin Type
-	-	T3U_12	-
-	-	T3U_11	N
-	-	T3U_10	P
if3	data_rxn[3]	T3U_9	N
if3	data_rxp[3]	T3U_8	P
if3	data_rxn[2]	T3U_7	N
if3	data_rxp[2]	T3U_6	P
if3	data_rxn[1]	T3L_5	N
if3	data_rxp[1]	T3L_4	P
if3	data_rxn[0]	T3L_3	N
if3	data_rxp[0]	T3L_2	P
if3	clk_rxn	T3L_1	N
if3	clk_rxp	T3L_0	P
-	-	T2U_12	-
-	-	T2U_11	N
-	-	T2U_10	P
if2	data_rxn[3]	T2U_9	N
if2	data_rxp[3]	T2U_8	P
if2	data_rxn[2]	T2U_7	N
if2	data_rxp[2]	T2U_6	P
if2	data_rxn[1]	T2L_5	N
if2	data_rxp[1]	T2L_4	P
if2	data_rxn[0]	T2L_3	N
if2	data_rxp[0]	T2L_2	P
if2	clk_rxn	T2L_1	N
if2	clk_rxp	T2L_0	P
-	-	T1U_12	-
-	-	T1U_11	N
-	-	T1U_10	P
if1	data_rxn[3]	T1U_9	N
if1	data_rxp[3]	T1U_8	P
if1	data_rxn[2]	T1U_7	N

Table 35: **4x4-lane D-PHY Interface** *(cont'd)*

Interface	Signal Name	Byte Group	Pin Type
if1	data_rxp[2]	T1U_6	P
if1	data_rxn[1]	T1L_5	N
if1	data_rxp[1]	T1L_4	P
if1	data_rxn[0]	T1L_3	N
if1	data_rxp[0]	T1L_2	P
if1	clk_rxn	T1L_1	N
if1	clk_rxp	T1L_0	P
-	-	T0U_12	-
-	-	T0U_11	N
-	-	T0U_10	P
if0	data_rxn[3]	T0U_9	N
if0	data_rxp[3]	T0U_8	P
if0	data_rxn[2]	T0U_7	N
if0	data_rxp[2]	T0U_6	P
if0	data_rxn[1]	T0L_5	N
if0	data_rxp[1]	T0L_4	P
if0	data_rxn[0]	T0L_3	N
if0	data_rxp[0]	T0L_2	P
if0	clk_rxn	T0L_1	N
if0	clk_rxp	T0L_0	P

The following table shows an example of a eight 2-lane D-PHY interface that be implemented in a single HP IO bank.

Table 36: **8x2-lane D-PHY Interface**

Interface	Signal Name	Byte Group	Pin Type
-	-	T3U_12	-
if7	data_rxn[1]	T3U_11	N
if7	data_rxp[1]	T3U_10	P
if7	data_rxn[0]	T3U_9	N
if7	data_rxp[0]	T3U_8	P
if7	clk_rxn	T3U_7	N
if7	clk_rxp	T3U_6	P
if6	data_rxn[1]	T3L_5	N
if6	data_rxp[1]	T3L_4	P
if6	data_rxn[0]	T3L_3	N
if6	data_rxp[0]	T3L_2	P
if6	clk_rxn	T3L_1	N

Table 36: **8x2-lane D-PHY Interface** *(cont'd)*

Interface	Signal Name	Byte Group	Pin Type
if6	clk_rxp	T3L_0	P
-	-	T2U_12	-
if5	data_rxn[1]	T2U_11	N
if5	data_rxp[1]	T2U_10	P
if5	data_rxn[0]	T2U_9	N
if5	data_rxp[0]	T2U_8	P
if5	clk_rxn	T2U_7	N
if5	clk_rxp	T2U_6	P
if4	data_rxn[1]	T2L_5	N
if4	data_rxp[1]	T2L_4	P
if4	data_rxn[0]	T2L_3	N
if4	data_rxp[0]	T2L_2	P
if4	clk_rxn	T2L_1	N
if4	clk_rxp	T2L_0	P
-	-	T1U_12	-
if3	data_rxn[1]	T1U_11	N
if3	data_rxp[1]	T1U_10	P
if3	data_rxn[0]	T1U_9	N
if3	data_rxp[0]	T1U_8	P
if3	clk_rxn	T1U_7	N
if3	clk_rxp	T1U_6	P
if2	data_rxn[1]	T1L_5	N
if2	data_rxp[1]	T1L_4	P
if2	data_rxn[0]	T1L_3	N
if2	data_rxp[0]	T1L_2	P
if2	clk_rxn	T1L_1	N
if2	clk_rxp	T1L_0	P
-	-	T0U_12	-
if1	data_rxn[1]	T0U_11	N
if1	data_rxp[1]	T0U_10	P
if1	data_rxn[0]	T0U_9	N
if1	data_rxp[0]	T0U_8	P
if1	clk_rxn	T0U_7	N
if1	clk_rxp	T0U_6	P
if0	data_rxn[1]	T0L_5	N
if0	data_rxp[1]	T0L_4	P
if0	data_rxn[0]	T0L_3	N

Table 36: **8x2-lane D-PHY Interface** *(cont'd)*

Interface	Signal Name	Byte Group	Pin Type
if0	data_rxp[0]	T0L_2	P
if0	clk_rxn	T0L_1	N
if0	clk_rxp	T0L_0	P

Strobe Propagation for D-PHY RX

Device architecture within BITSLICE and BITSLICE_CONTROL allows the user to propagate the Strobe between byte groups by using additional IO pin(s) internally. Additional pin usage for Strobe propagation depends on the RX clock lane IO (Strobe) selection along with RX data lane IO selection.

***Note*:** Strobe propagation is not applicable if DBC pin is selected as RX clock lane IO.

The following table provides the scenarios for additional IO realization by HSSIO IP wizard for Strobe propagation. This pin(s) are generated with bg<>_pin<>_nc name. N pins are not shown in the following table for simplicity.

Table 37: **Strobe Propagation for D-PHY RX Interface**

Byte Group	T1L_0 as RX Clock Lane IO	T1U_6 as RX Clock Lane IO	T2L_0 as RX Clock Lane IO	T2U_6 as RX Clock Lane IO
T3U_10	Selecting this IO will force bg2_pin0_nc and bg3_pin0_nc use	Selecting this IO will force bg2_pin6_nc and bg3_pin6_nc use	Selecting this IO will force bg3_pin0_nc use	Selecting this IO will force bg3_pin6_nc use
T3U_8	Selecting this IO will force bg2_pin0_nc and bg3_pin0_nc use	Selecting this IO will force bg2_pin6_nc and bg3_pin6_nc use	Selecting this IO will force bg3_pin0_nc use	Selecting this IO will force bg3_pin6_nc use
T3U_6	Selecting this IO will force bg2_pin0_nc and bg3_pin0_nc use	bg3_pin6_nc will be inferred by using this IO	Selecting this IO will force bg3_pin0_nc use	bg3_pin6_nc will be inferred by using this IO
T3L_4	Selecting this IO will force bg2_pin0_nc and bg3_pin0_nc use	Selecting this IO will force bg2_pin6_nc and bg3_pin6_nc use	Selecting this IO will force bg3_pin0_nc use	Selecting this IO will force bg3_pin6_nc use
T3L_2	Selecting this IO will force bg2_pin0_nc and bg3_pin0_nc use	Selecting this IO will force bg2_pin6_nc and bg3_pin6_nc use	Selecting this IO will force bg3_pin0_nc use	Selecting this IO will force bg3_pin6_nc use
T3L_0	bg3_pin0_nc will be inferred by using this IO	Selecting this IO will force bg2_pin6_nc and bg3_pin6_nc use	bg3_pin0_nc will be inferred by using this IO	Selecting this IO will force bg3_pin6_nc use
T2U_10	Selecting this IO will force bg2_pin0_nc use	Selecting this IO will force bg2_pin6_nc use	Same Byte Group	Same Byte Group
T2U_8	Selecting this IO will force bg2_pin0_nc use	Selecting this IO will force bg2_pin6_nc use	Same Byte Group	Same Byte Group
T2U_6	Selecting this IO will force bg2_pin0_nc use	bg2_pin6_nc will be inferred by using this IO	Same Byte Group	RX Clock Lane IO

Table 37: **Strobe Propagation for D-PHY RX Interface** *(cont'd)*

Byte Group	T1L_0 as RX Clock Lane IO	T1U_6 as RX Clock Lane IO	T2L_0 as RX Clock Lane IO	T2U_6 as RX Clock Lane IO
T2L_4	Selecting this IO will force bg2_pin0_nc use	Selecting this IO will force bg2_pin6_nc use	Same Byte Group	Same Byte Group
T2L_2	Selecting this IO will force bg2_pin0_nc use	Selecting this IO will force bg2_pin6_nc use	Same Byte Group	Same Byte Group
T2L_0	bg2_pin0_nc will be inferred by using this IO	Selecting this IO will force bg2_pin6_nc use	RX Clock Lane IO	Same Byte Group
T1U_10	Same Byte Group	Same Byte Group	Selecting this IO will force bg1_pin0_nc use	Selecting this IO will force bg1_pin6_nc use
T1U_8	Same Byte Group	Same Byte Group	Selecting this IO will force bg1_pin0_nc use	Selecting this IO will force bg1_pin6_nc use
T1U_6	Same Byte Group	RX Clock Lane IO	Selecting this IO will force bg1_pin0_nc use	bg1_pin6_nc will be inferred by using this IO
T1L_4	Same Byte Group	Same Byte Group	Selecting this IO will force bg1_pin0_nc use	Selecting this IO will force bg1_pin6_nc use
T1L_2	Same Byte Group	Same Byte Group	Selecting this IO will force bg1_pin0_nc use	Selecting this IO will force bg1_pin6_nc use
T1L_0	RX Clock Lane IO	Same Byte Group	bg1_pin0_nc will be inferred by using this IO	Selecting this IO will force bg1_pin6_nc use
T0U_10	Selecting this IO will force bg0_pin0_nc use	Selecting this IO will force bg0_pin6_nc use	Selecting this IO will force bg0_pin0_nc and bg1_pin0_nc use	Selecting this IO will force bg0_pin6_nc and bg1_pin6_nc use
T0U_8	Selecting this IO will force bg0_pin0_nc use	Selecting this IO will force bg0_pin6_nc use	Selecting this IO will force bg0_pin0_nc and bg1_pin0_nc use	Selecting this IO will force bg0_pin6_nc and bg1_pin6_nc use
T0U_6	Selecting this IO will force bg0_pin0_nc use	bg0_pin6_nc will be inferred by using this IO	Selecting this IO will force bg0_pin0_nc and bg1_pin0_nc use	bg0_pin6_nc will be inferred by using this IO
T0L_4	Selecting this IO will force bg0_pin0_nc use	Selecting this IO will force bg0_pin6_nc use	Selecting this IO will force bg0_pin0_nc and bg1_pin0_nc use	Selecting this IO will force bg0_pin6_nc and bg1_pin6_nc use
T0L_2	Selecting this IO will force bg0_pin0_nc use	Selecting this IO will force bg0_pin6_nc use	Selecting this IO will force bg0_pin0_nc and bg1_pin0_nc use	Selecting this IO will force bg0_pin6_nc and bg1_pin6_nc use
T0L_0	bg0_pin0_nc will be inferred by using this IO	Selecting this IO will force bg0_pin6_nc use	bg0_pin0_nc will be inferred by using this IO	Selecting this IO will force bg0_pin6_nc and bg1_pin6_nc use

D-PHY TX Pin Rules

- Select the HP IO bank that has the VRP pin. DCI_CASCADE is allowed from HP IO bank of the same IO column in case the VRP pin is grounded for the selected HP IO bank.
- Select the IO pins continuously without leaving any IO pairs in the middle of D-PHY interface.
- Since D-PHY IP is using IO in Native mode, left out IO cannot be used by any other design and it will be unusable.

- D-PHY with two different line rates can be implemented within IO bank and each D-PHY interface will use one PLL.
- All the lanes of a particular MIPI D-PHY instance need to be in the same HP IO bank, which the Pin Assignment Tab of XGUI automatically controls for UltraScale+.
- In a case of multiple MIPI D-PHY instances sharing clock resources, all such instances also need to be in the same HP IO bank.
- IO used for clock lane and data lane(s) can be swapped in any order for D-PHY TX IP.

Pin Rules for 7 series FPGAs

This section describes the pin rules for 7 series FPGAs:

- Non-continuous IO usage is allowed for D-PHY TX and RX interfaces but not recommended.
- Restrict the IO selection within the single IO bank.
- Select SRCC/MRCC pins for D-PHY RX clock lane.

XILINX

Appendix D

Additional Resources and Legal Notices

Xilinx Resources

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

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

Note: For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

References

These documents provide supplemental material useful with this guide:

1. MIPI Alliance D-PHY Specifications
2. *Advanced I/O Wizard LogiCORE IP Product Guide* (PG320)
3. *Vivado Design Suite: AXI Reference Guide* (UG1037)
4. *UltraScale Architecture SelectIO Resources User Guide* (UG571)
5. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
6. *Vivado Design Suite User Guide: Designing with IP* (UG896)
7. *Vivado Design Suite User Guide: Getting Started* (UG910)
8. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
9. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
10. *Vivado Design Suite User Guide: Implementation* (UG904)
11. *D-PHY Solutions* (XAPP894)
12. *UltraScale Architecture PCB Design User Guide* (UG583)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
09/07/2020 Version 4.2	
Appendix C: Pin and Bank Rules	Hot fix to reinstate section.
07/16/2020 Version 4.2	
Core Configuration Tab	Added details of new/updated parameters.
Clocking	Added clarifications about MMCM and line rate.
PPI Signals	Added new signals.
Clocking and Reset Signals	Added new signals.
10/30/2019 Version 4.2	
General Updates	For 7 series fixed mode IDELAY control ready has been incorporated for core operation.
General Updates	Added Versal support.
07/02/2019 Version 4.1	
General Updates	Added 2.5 Gb/s support to the subsystem.

Section	Revision Summary
12/10/2018 Version 4.1	
General Updates	• Extended the RX lane configuration from 4 to 8 lanes. • Extended the RX register space from 4 to 8 lanes. • Figure in Pin Assignment Tab section has been updated. • Figure in D-PHY RX Pin Rules section has been updated.
04/04/2018 Version 4.1	
General Updates	• Added Spartan 7 series support • Added C_IDLY_GROUP_NAME parameter details • Figures in Design Flow Steps chapter have been updated • Added a figure in D-PHY RX Pin Rules section • Added new IDELAY_TAP_VALUE register details • Updated Pin and Bank Rules in Appendix C
10/04/2017 Version 4.0	
Minor Updates	Minor Updates
04/05/2017 Version 3.1	
General Updates	• Updated system_rst_in port details • Updated 7 series calibration ports • Removed calibration register (CAL_REG) • Added new HS_SETTLE register details • Added a new D-PHY RX IP clocking diagram for Zynq® UltraScale+™ due to constant clkoutphy and MMCM removal • Added Appendix C: Pin and Bank Rules
10/05/2016 Version 3.0	
General Updates	• Added 7 series support • Updated Figure 3-12 waveform • Added Active Lane Support in Chapter 4

Section	Revision Summary
04/06/2016 Version 2.0	
General Updates	• Updated D-PHY RX latency numbers • Added PKT_CNT field and updated HS_TIMEOUT/ ESC_TIMEOUT registers • Added Shared Logic feature • Updated I/O planning feature • Updated Clocking section • Added recommended reset sequence for D-PHY in a system • Updated the rxvalidhs signal behavior in Example 6 of High-Speed Receive • Added Hardware Validation in Appendix
11/18/2015 Version 1.0	
Initial Xilinx release	NA

Please Read: Important Legal Notices

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

Copyright