

DisplayPort 1.4 RX Subsystem v2.0

Product Guide

PG300 (v2.0) December 5, 2018



Table of Contents

Chapter 1: Introduction.....	4
Features.....	4
IP Facts.....	5
Chapter 2: Overview.....	6
Unsupported Features.....	6
Licensing and Ordering.....	7
Chapter 3: Product Specification.....	8
AXI4-Stream Video Interface.....	8
Native Video Interface.....	10
Subsystem Sub-core Descriptions.....	11
Standards.....	14
Resource Use.....	14
Port Descriptions.....	14
Register Space.....	20
Chapter 4: Designing with the Subsystem.....	40
DisplayPort Overview.....	40
Reduced Blanking.....	48
Pixel Mapping.....	49
AXI4-Stream Interface Color Mapping	56
Clocking.....	56
Resets.....	58
Address Map Example.....	58
Programming Sequence.....	58
Chapter 5: Design Flow Steps.....	59
Customizing and Generating the Subsystem.....	59
Constraining the Subsystem.....	62
Simulation.....	63
Synthesis and Implementation.....	64

Chapter 6: Example Design	65
Available Example Designs	65
Building the Example Design	66
Hardware Setup and Run	77
Display User Console	83
Setting the FMC Voltage to 1.8V	84
Tested Equipment	85
Appendix A: Upgrading	86
Appendix B: Frequently Asked Questions	87
Appendix C: Driver Documentation	88
Appendix D: Debugging	89
Finding Help on Xilinx.com	89
Debug Tools	90
Hardware Debug	91
Software Debug	94
Appendix E: Additional Resources and Legal Notices	96
Xilinx Resources	96
Documentation Navigator and Design Hubs	96
References	97
Revision History	97
Please Read: Important Legal Notices	98

Introduction

The Xilinx[®] DisplayPort 1.4 RX Subsystem is a plug-in solution for serial digital video data reception in large video systems and supports video resolutions of up to Full Ultra HD (FUHD) at 30 fps. The Xilinx DisplayPort subsystem provides ease of use in selecting the required mode with automated customization.

Features

- Support for DisplayPort Sink (RX) capabilities
- Supports multi-stream transport (MST) and single stream transport (SST) at FUHD at 30 fps
- Dynamic lane support (1, 2, or 4 lanes)
- Dynamic link rate support (1.62/2.7/5.4/8.1 Gb/s)
- Dynamic support for 6, 8, 10, 12, or 16 bits per component (BPC)
- Dynamic support for RGB/YCbCr444/YCbCr422 color formats
- Supports 16-bit Video PHY (GT) interface
- Supports 2 to 8 channel audio with 44/48 KHz sample rates
- Supports HDCP 1.3 decryption in SST
- AXI IIC controller for external peripheral programming
- Supports native or AXI4-Stream video input interface
- Supports single audio stream in MST mode
- Supports SDP packet for static HDR mode

IP Facts

LogiCORE™ IP Facts Table	
Subsystem Specifics	
Supported Device Family ⁽¹⁾	UltraScale+™ Families (GTHE4, GTYE4) UltraScale™ Families (GTHE3) Zynq® UltraScale+™ RFSoc
Supported User Interfaces	AXI4-Stream, AXI4-Lite, Native video
Resources	Performance and Resource Use web page
Provided with Subsystem	
Design Files	Hierarchical subsystem packaged with DisplayPort RX core and other IP cores
Example Design	Vivado® IP integrator
Test Bench	Not Provided
Constraints File	IP cores delivered with XDC files
Simulation Model	Not Provided
Supported S/W Driver	Standalone
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	
Provided by Xilinx® at the 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 DisplayPort 1.4 RX Subsystem is a full feature, hierarchically packaged subsystem with a DisplayPort (RX) core ready to use in applications in large video systems.

The DisplayPort 1.4 RX Subsystem, in both AXI4-Stream and native interfaces, operates in the following video modes:

- Single stream transport (SST)
- Multi-stream transport (MST) up to 4 streams



RECOMMENDED: For the Xilinx DisplayPort 1.4 RX Subsystem, use a MegaChip retimer.

The following table shows the UltraScale™ and UltraScale+™ families core support.

Table 1: Core Support

Features	UltraScale (GTHE3)	UltraScale+ (GTHE4, GTYE4)
DisplayPort 1.4 – 8.1 Gb/s (without HDCP)	Yes ⁽¹⁾	Yes
DisplayPort 1.4 – 8.1 Gb/s (with HDCP)	Roadmap ⁽²⁾	Yes

Notes:

1. The use of different implementation strategies/pblocks is required to meet timing at 8.1 Gb/s. To learn more about timing closure techniques, see the *UltraFast Design Methodology Guide for the Vivado Design Suite* ([UG949](#)).
2. This feature is limited to High Bit Rate 2 (HBR2).

Unsupported Features

The following features of the standard are not supported in the subsystem:

- In-band stereo
- Video AXI4-Stream interface is not scalable with dynamic pixel mode selection
- Dual-pixel splitter is not supported in native video mode
- HDCP is not supported in MST mode
- HDCP 2.x
- eDP and iDP

- GTC
- Non-LPCM audio
- DSC and/or FEC
- 16/32 channel audio
- MCCS over DDC/CI
- 420 Colorimetry

Licensing and Ordering

This Xilinx[®] LogiCORE[™] IP module is provided under the terms of the [Xilinx Core License Agreement](#). The module is shipped as part of the Vivado[®] Design Suite. For full access to all core functionalities in simulation and in hardware, you must purchase a license for the core. To generate a full license, visit the [product licensing web page](#). Evaluation licenses and hardware timeout licenses might be available for this subsystem. Contact your [local Xilinx sales representative](#) for information about pricing and availability.

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 subsystem.

For more information about this subsystem, visit the DisplayPort [product web page](#).

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](#).

License Checkers

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

- Vivado Synthesis
- Vivado Implementation
- `write_bitstream` (Tcl command)

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

Product Specification

The subsystem can operate with an AXI4-Stream video interface or a native interface using a variety of sub-cores which are described in the following sections.

AXI4-Stream Video Interface

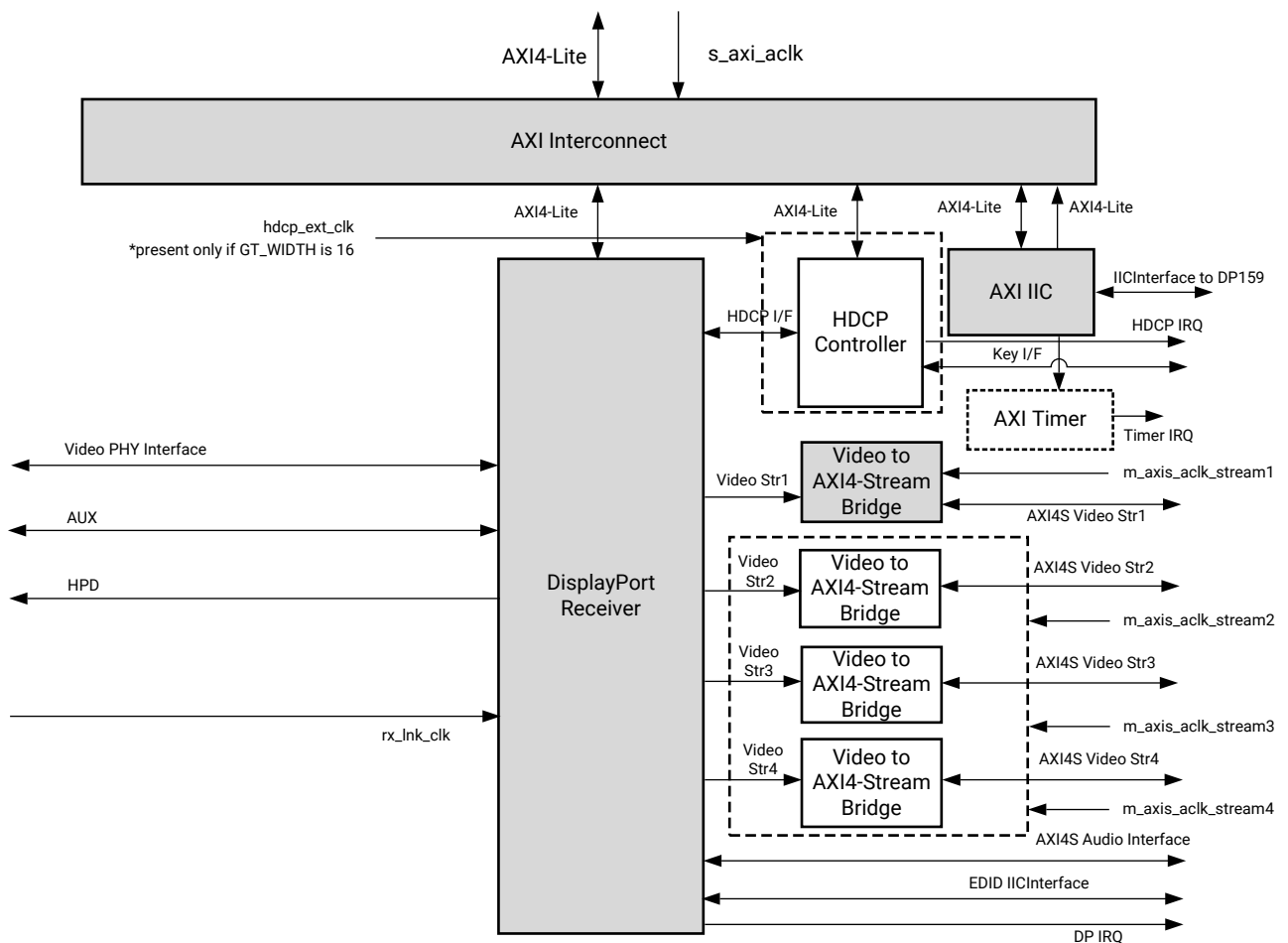
When configured with the AXI4-Stream video interface, the subsystem is packaged with these subcores:

- DisplayPort Receive core
- DisplayPort Video to AXI4-Stream Bridge
- AXI IIC controller
- HDCP core with AXI Timer when HDCP feature is enabled

In MST mode, in addition to the subcores listed in SST, Video to AXI4-Stream Bridge instances increase to the number of video streams.

Because the DisplayPort 1.4 RX Subsystem is hierarchically packaged, you select the parameters and the subsystem creates the required hardware. The subsystem receives video using the DisplayPort v1.4 protocol over a 16-bit video PHY interface. The subsystem works with the Video PHY Controller (*Video PHY Controller LogiCORE IP Product Guide (PG230)*) configured for the DisplayPort protocol. The subsystem outputs multi-pixel video over an AXI4-Stream interface. The following figure shows the architecture of the subsystem assuming MST with four streams.

Figure 1: DisplayPort 1.4 RX Subsystem AXI4-Stream Video Interface Block Diagram



X15190-110518

Related Information

[Pixel Mapping Examples on AXI4-Stream Interface](#)

Native Video Interface

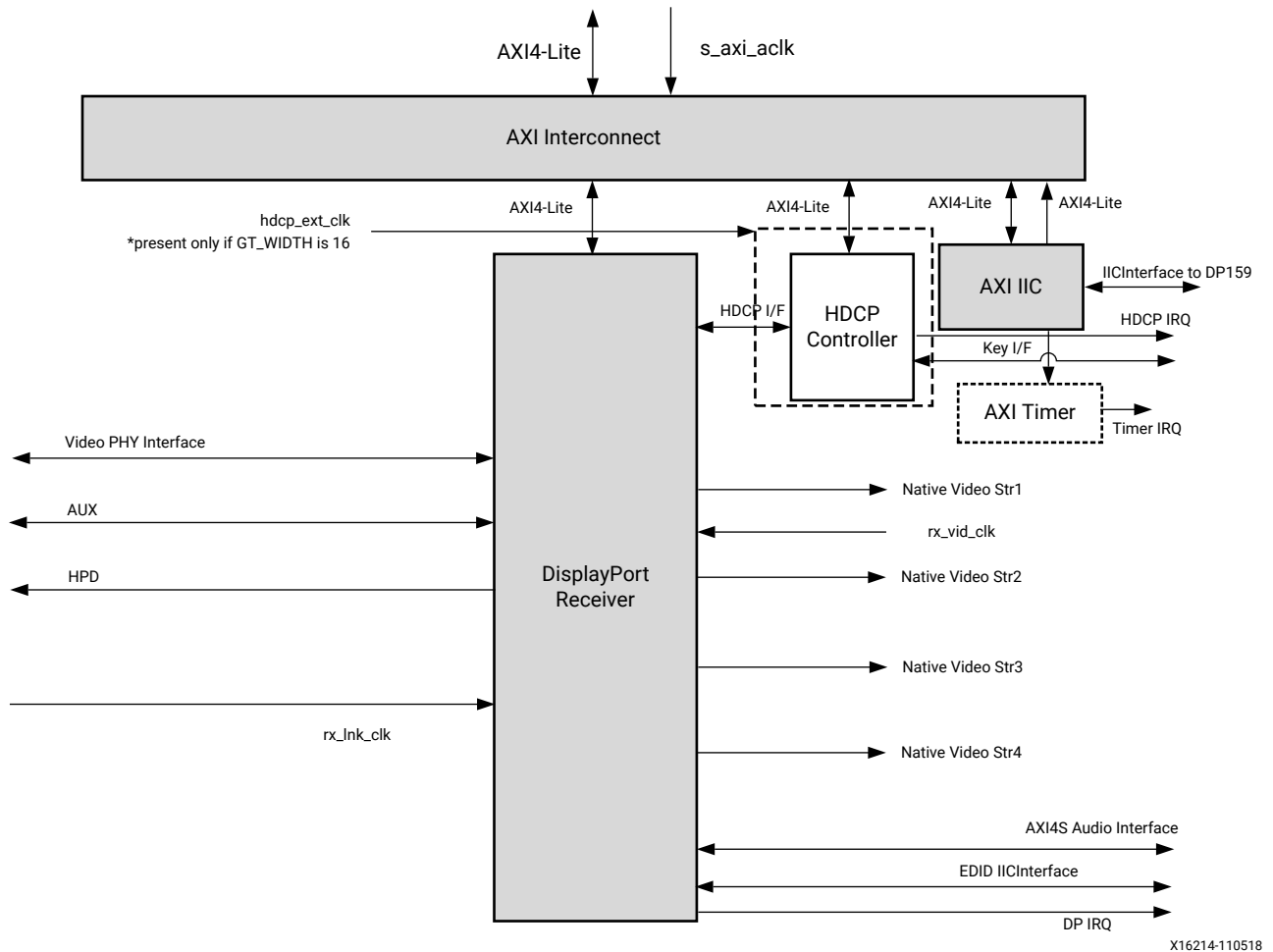
When the native video interface is selected, the subsystem is packaged with two mandatory subcores:

- DisplayPort RX core
- AXI IIC controller

Because the DisplayPort 1.4 RX Subsystem is hierarchically packaged, you select the parameters and the subsystem creates the required hardware. The following figure shows the architecture of the subsystem assuming MST with four native video streams.

The DisplayPort 1.4 RX Subsystem receives the video using the DisplayPort 1.4 protocol over a 16-bit video PHY interface. The subsystem works in conjunction with the Video PHY Controller configured for DisplayPort protocol. The subsystem outputs multi-pixel video to the AXI4-Stream interface.

Figure 2: DisplayPort 1.4 RX Subsystem Native Video Block Diagram



Related Information

[Pixel Mapping on Native Video Interface](#)

Subsystem Sub-core Descriptions

The subsystem is comprised of multiple sub-cores. The following sections provide a brief overview of these sub-cores.

Video to AXI4-Stream Bridge IP Core

A video to AXI4-Stream bridge is used in DisplayPort 1.4 RX Subsystem to convert the video output of the DisplayPort RX IP core to an AXI4-Stream interface. See the *Video In to AXI4-Stream LogiCORE IP Product Guide* ([PG043](#)) for information on this core.

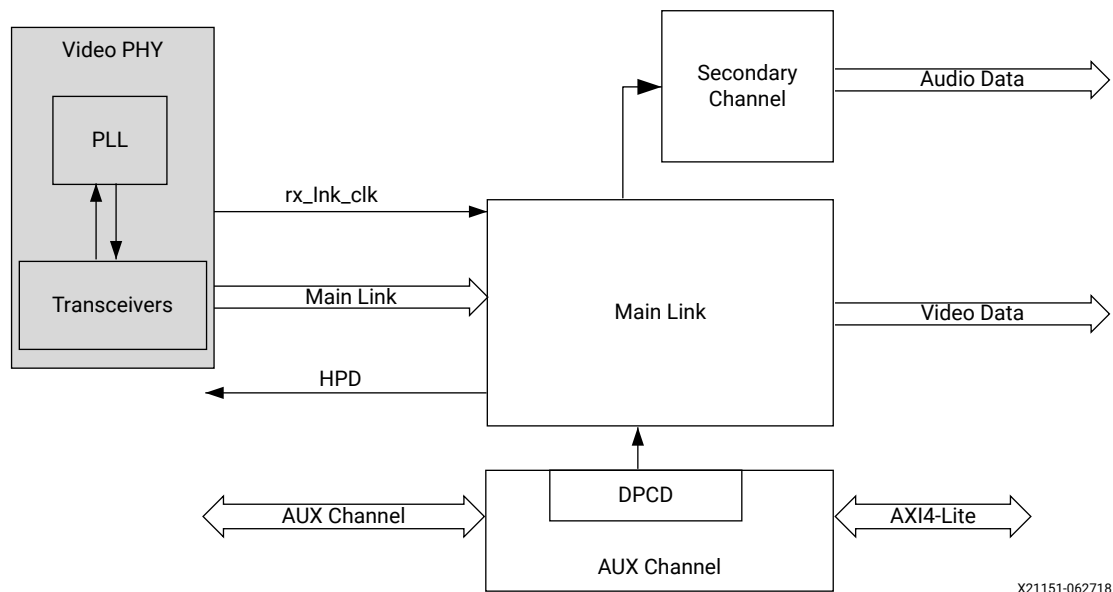
Note: In MST mode, there are N number of bridges in the subsystem, where N = the number of AXI4-Stream outputs to the subsystem.

DisplayPort Receive IP Core

The DisplayPort RX block is delivered as part of the DisplayPort 1.4 RX Subsystem and contains the following components, also shown in the following figure:

- **Main Link:** Provides delivery of the primary video stream.
- **Secondary Channel:** Provides the delivery of audio information from the blanking period of the video stream to an AXI4-Stream interface.
- **AUX Channel:** Establishes the dedicated source to sink communication channel.
- **DPCD:** Contains the set of DisplayPort Configuration Data, which is used to establish the operating parameters of each core.

Figure 3: DisplayPort Receive Core Block Diagram

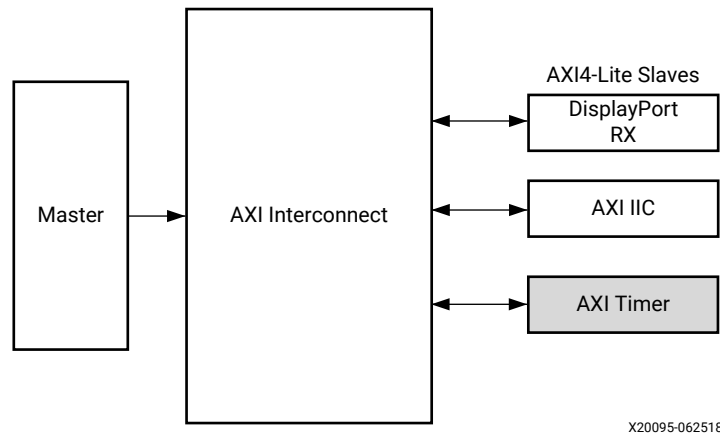


AXI Interconnect IP Core

The subsystem uses the Xilinx® AXI Interconnect IP core, as a crossbar, which contains one AXI4-Lite slave interface and two AXI4-Lite master interfaces.

The following figure shows the AXI slave structure within the DisplayPort 1.4 RX Subsystem. For more details on the AXI crossbar functionality, see the [AXI Interconnect LogiCORE IP Product Guide \(PG059\)](#).

Figure 4: **AXI4-Lite Interconnect within DisplayPort 1.4 RX Subsystem**



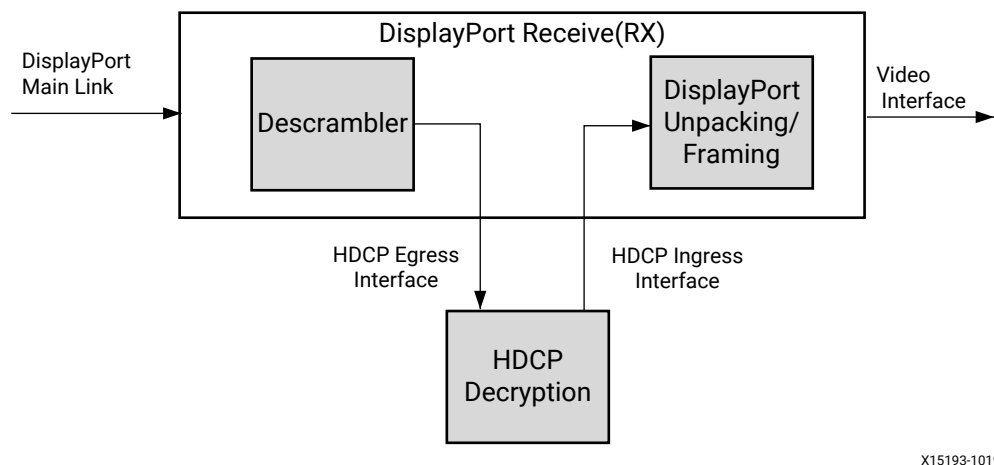
Related Information

[Address Map Example](#)

HDCP Controller IP Core

The HDCP v1.3 protocol specifies a secure method of transmitting audiovisual content. Further, the audiovisual content can be transmitted over a DisplayPort interface. The HDCP Controller IP core is used for data decryption along with the DisplayPort Receive IP core in the DisplayPort 1.4 RX Subsystem.

The following figure shows the DisplayPort 1.4 RX Subsystem with the HDCP controller. For more details on HDCP, see the *HDCP 1.x Product Guide* ([PG224](#)).



AXI Timer IP Core

A 32-bit AXI Timer IP core is used in the DisplayPort 1.4 RX Subsystem. The AXI Timer can be accessed through the AXI4 master interface for basic timer functionality in the system.

AXI IIC IP Core

The AXI IIC Bus Interface IP core is used in DisplayPort 1.4 RX Subsystem to configure any external active devices such as the Retimer.

Standards

The DisplayPort 1.4 RX Subsystem is compatible with the DisplayPort v1.4 Standard, IIC, as well as the AXI4-Lite and AXI4-Stream interfaces.



IMPORTANT! Xilinx® DisplayPort subsystems have passed compliance certification. If you are interested in accessing the compliance report or seeking guidance for the compliance certification of your products, contact your local Xilinx sales representative.

Resource Use

For full details about performance and resource use, visit the [Performance and Resource Use web page](#).

Port Descriptions

The DisplayPort 1.4 RX Subsystem ports are described here.

AXI4-Lite Interface

Table 2: AXI4-Lite Interface

Port Name	I/O	Description
s_axi_aclk	I	AXI Bus clock
s_axi_aresetn	I	AXI reset. Active-Low.
s_axi_awaddr[13:0]	I	Write address

Table 2: AXI4-Lite Interface (cont'd)

Port Name	I/O	Description
s_axi_awprot[2:0]	I	Protection Type
s_axi_awvalid	I	Write address Valid
s_axi_awready	O	Write address Ready
s_axi_wdata[31:0]	I	Write data
s_axi_wstrb[3:0]	I	Write Strobe
s_axi_wvalid	I	Write data valid
s_axi_wready	O	Write data ready
s_axi_bresp[1:0]	O	Write response
s_axi_bvalid	O	Write response valid
s_axi_bready	I	Write response ready
s_axi_araddr s_axi_araddr[13:0]	I	Read address
s_axi_arprot[2:0]	I	Read protection type
s_axi_arvalid	I	Read address valid
s_axi_arready	O	Read address ready
s_axi_rdata[31:0]	O	Read data
s_axi_rresp[1:0]	O	Read data response
s_axi_rvalid	O	Read data valid
s_axi_rready	I	Read data ready

HDCP Key Interface

Table 3: Interrupt Interface

Port Name	I/O	Description
hdcp_ext_clk	I	HDCP external clock
hdcp_key_aclk	I	HDCP key clock
hdcp_key_aresetn	I	Key Interface reset. Active-Low
hdcp_key_tdata[63:0]	I	AXI4-Stream Key Tdata
hdcp_key_last	I	AXI4-Stream Key Tlast
hdcp_key_tready	O	AXI4-Stream Key Tready
hdcp_key_tuser[7:0]	I	AXI4-Stream Key TUSER. KMB should send the Key number from 0 to 41. 0 corresponds to KSV and 1 to 40 are the HDCP Keys count.
hdcp_key_tvalid	I	AXI4-Stream Key TValid
reg_key_sel[2:0]	O	To select the one of the eight sets of 40 keys.
start_key_transmit	O	An Active-High pulse that is used to start key transmit.

DisplayPort Video PHY Sideband Status

Table 4: DisplayPort Video PHY Sideband Status

Port Name	I/O	Description
s_axis_phy_rx_sb_status_tdata[15:0]	I	Video PHY status input
s_axis_phy_rx_sb_status_tready	O	Ready to video PHY for status
s_axis_phy_rx_sb_status_tvalid	I	Video PHY status valid

DisplayPort Video PHY Sideband Control

Table 5: DisplayPort Video PHY Sideband Control

Port Name	I/O	Description
m_axis_phy_rx_sb_control_tdata[7:0]	O	Control output to video PHY
m_axis_phy_rx_sb_control_tvalid	O	Control output valid to video PHY
m_axis_phy_rx_sb_control_tready	I	Control data ready input

DisplayPort Link Clock Interface

Table 6: DisplayPort Link Clock Interface

Port Name	I/O	Description
rx_lnk_clk	I	Link clock

DisplayPort Video PHY Main Link [Lane0–Lane3]

Table 7: DisplayPort Video PHY Main Link [Lane0–Lane3]

Port Name	I/O	Description
s_axis_lnk_rx_lane0_tdata[31:0]	I	Main link data for lane0
s_axis_lnk_rx_lane0_tvalid	I	Main link data valid for lane0
s_axis_lnk_rx_lane0_tready	O	Main link data ready for lane0
s_axis_lnk_rx_lane0_tuser[11:0]	I	Main link user data for lane0
s_axis_lnk_rx_lane1_tdata[31:0]	I	Main link data for lane1
s_axis_lnk_rx_lane1_tvalid	I	Main link data valid for lane1
s_axis_lnk_rx_lane1_tready	O	Main link data ready for lane1
s_axis_lnk_rx_lane1_tuser[11:0]	I	Main link user data for lane1
s_axis_lnk_rx_lane2_tdata[31:0]	I	Main link data for lane2
s_axis_lnk_rx_lane2_tvalid	I	Main link data valid for lane2
s_axis_lnk_rx_lane2_tready	O	Main link data ready for lane2

Table 7: DisplayPort Video PHY Main Link [Lane0–Lane3] (cont'd)

Port Name	I/O	Description
s_axis_lnk_rx_lane2_tuser[11:0]	I	Main link user data for lane2
s_axis_lnk_rx_lane3_tdata[31:0]	I	Main link data for lane3
s_axis_lnk_rx_lane3_tvalid	I	Main link data valid for lane3
s_axis_lnk_rx_lane3_tready	O	Main link data ready for lane3
s_axis_lnk_rx_lane3_tuser[11:0]	I	Main link user data for lane3

DisplayPort Receive Video Interface

Table 8: DisplayPort Receive Video Interface

Port Name	I/O	Description
rx_vid_clk	I	DisplayPort 1.4 RX video clock
rx_vid_rst	I	DisplayPort 1.4 RX video reset

AXI4-Stream Video Stream 1 Interface

This interface is enabled when the AXI4-Stream interface is selected.

Table 9: DisplayPort 1.4 RX Subsystem AXI4-Stream Video Stream 1 Interface

Port Name	I/O	Description
m_axis_aclk_stream1	I	Stream1 video clock input
m_axis_video_stream1_tdata[191:0]	O	Stream1 video data. Maximum width of 192.
m_axis_video_stream1_tlast	O	Stream1 video last data, End of line pixel.
m_axis_video_stream1_tready	I	Stream1 video data read
m_axis_video_stream1_tuser	O	Stream1 video user data
m_axis_video_stream1_tvalid	O	Stream1 video data valid

Native Video Stream 1 Interface

Table 10: DisplayPort 1.4 RX Subsystem Native Video Stream 1 Interface

Port Name	I/O	Description
rx_vid_stream1_tx_vid_enable	O	User data video enable
rx_vid_stream1_tx_vid_hsync	O	Horizontal sync pulse. Active on the rising edge.
rx_vid_stream1_tx_vid_oddeven	O	Indicates an odd (1) or even (0) field polarity. If not used, this pin should be connected to 0.
rx_vid_stream1_tx_vid_pixel0[47:0]	O	Video data
rx_vid_stream1_tx_vid_pixel1[47:0]	O	Video data

Table 10: DisplayPort 1.4 RX Subsystem Native Video Stream 1 Interface (cont'd)

Port Name	I/O	Description
rx_vid_stream1_tx_vid_pixel2[47:0]	O	Video data
rx_vid_stream1_tx_vid_pixel3[47:0]	O	Video data
rx_vid_stream1_tx_vid_vsync	O	Vertical sync pulse. Active on the rising edge.
rx_bpc[2:0]	O	Bits per component and derived from MISC0 and MISC1 fields
rx_vid_pixel_mode[2:0]	O	User pixel width/mode
rx_cformat[2:0]	O	Colorimetry indicator field and derived from MISC0 and MISC1 fields

DisplayPort MST Stream

Table 11: DisplayPort 1.4 RX Subsystem MST Stream

Port Name	I/O	Description
m_axis_aclck_stream<n>	I	Stream<n> video clock input
m_axis_video_stream<n>_tdata[191:0]	O	Stream<n> video data. Maximum width of 192.
m_axis_video_stream<n>_tlast	O	Stream<n> video last data, End of line pixel.
m_axis_video_stream<n>_tready	I	Stream<n> video data read
m_axis_video_stream<n>_tuser	O	Stream<n> video user data
m_axis_video_stream<n>_tvalid	O	Stream<n> video data valid
rx_vid_stream<n>_tx_vid_enable	O	User data video enable.
rx_vid_stream<n>_tx_vid_hsync	O	Horizontal sync pulse. Active on the rising edge.
rx_vid_stream<n>_tx_vid_oddeven	O	Indicates an odd (1) or even (0) field polarity. If not used, this pin should be connected to 0.
rx_vid_stream<n>_tx_vid_pixel0[47:0]	O	Video data.
rx_vid_stream<n>_tx_vid_pixel1[47:0]	O	Video data.
rx_vid_stream<n>_tx_vid_pixel2[47:0]	O	Video data.
rx_vid_stream<n>_tx_vid_pixel3[47:0]	O	Video data.
rx_vid_stream<n>_tx_vid_vsync	O	Vertical sync pulse. Active on the rising edge.

Notes:

1. <n> = stream number 2 to 4.

AUX I/O Interface – Internal Bidirectional IOB Ports

Table 12: AUX I/O Interface – Internal Bidirectional IOB

Port Name	I/O	Description
aux_rx_io_p	I/O	Bidirectional AUX IO- p
aux_rx_io_n	I/O	Bidirectional AUX IO- n

AUX I/O Interface – Internal Unidirectional IOB

Table 13: AUX I/O Interface – Internal Unidirectional IOB

Port Name	I/O	Description
aux_rx_channel_in_p	I	Unidirectional AUX channel in - p
aux_rx_channel_in_n	I	Unidirectional AUX channel in - n
aux_rx_channel_out_p	O	Unidirectional AUX channel out - p
aux_rx_channel_out_n	O	Unidirectional AUX channel out- n

AUX IP Interface – External IOB

Table 14: AUX IP Interface – External IOB

Port Name	I/O	Description
aux_rx_data_in	I	External AUX data input
aux_rx_data_out	O	External AUX data output
aux_rx_data_en_out_n	O	External AUX data enable out. Active-Low.

HPD Interface

Table 15: HPD Interface Ports

Port Name	I/O	Description
rx_hpd	O	HPD from DisplayPort 1.4 RX

EDID IIC Interface

Table 16: EDID IIC Interface

Port Name	I/O	Description
edid_iic_sci_i	I	EDID IIC SCL input
edid_iic_sci_o	O	EDID IIC SCL output
edid_iic_sci_t	O	EDID IIC SCL enable. IIC SCL enable is Active-Low.
edid_iic_sda_i	I	EDID IIC SDA input
edid_iic_sda_o	O	EDID IIC SDA output
edid_iic_sda_t	O	EDID IIC SDA enable. IIC SDA enable is Active-Low.

IIC Interface

Table 17: IIC Interface

Port Name	I/O	Description
ext_iic_sci_i	I	IIC SCL input
ext_iic_sci_o	O	IIC SCL output
ext_iic_sci_t	O	IIC SCL enable
ext_iic_sda_i	I	IIC SDA input
ext_iic_sda_o	O	IIC SDA output
ext_iic_sda_t	O	IIC SDA enable
ext_rst	O	IIC reset through AXI IIC controller GPIO port0

Interrupts

Table 18: Interrupts

Port Name	I/O	Description
dprxss_dp_irq	O	DisplayPort 1.4 RX IP interrupt out
dprxss_iic_irq	O	AXI IIC IP interrupt out
dprx_timer_irq	O	AXI Timer interrupt out

Register Space

This section details registers available in the DisplayPort 1.4 RX Subsystem. The address map is split into following regions:

- DisplayPort RX IP
- AXI IIC
- HDCP Controller
- AXI Timer

The subsystem address propagation in the Vivado® IP integrator assigns the maximum addresses based on full featured configuration. Ensure the following steps are taken:

1. Confirm that the DisplayPort 1.4 RX Subsystem IP is mapped to a base address where the 14th bit in the address is 0. For example, `0x44A00000` is correct and `0x44A02000` causes errors.
2. Ensure that all 14 bits of the address range are reserved for the DisplayPort 1.4 RX Subsystem IP.

DisplayPort Registers

The DisplayPort Configuration Data is implemented as a set of distributed registers which can be read or written from the AXI4-Lite interface. These registers are considered to be synchronous to the AXI4-Lite domain and asynchronous to all others.

For parameters that might change while being read from the configuration space, two scenarios might exist. In the case of single bits, either the new value or the old value is read as valid data. In the case of multiple bit fields, a lock bit might be used to prevent the status values from being updated while the read is occurring. For multi-bit configuration data, a toggle bit is used indicating that the local values in the functional core should be updated.

Any bits not specified in the following tables are considered reserved and returns 0 upon read. The power on reset values of all the registers are 0 unless it is specified in the definition. Only address offsets are listed and the base addresses are configured by the AXI Interconnect.

Receiver Core Configuration

Table 19: Receiver Core Configuration

Offset	Access Type	Description
0x000	R/W	LINK_ENABLE. Enable the receiver 1 - Enables the receiver core. Asserts the HPD signal when set.
0x004	R/W	AUX_CLOCK_DIVIDER. Contains the clock divider value for generating the internal 1 MHz clock from the AXI4-Lite host interface clock. The clock divider register provides integer division only and does not support fractional AXI4-Lite clock rates (for example, set to 75 for a 75 MHz AXI4-Lite clock). [27:24] - Valid values are 0-8. Non-zero value in this field issues defers as per programmed value to DPCD read of LANE0_1_STATUS register. This functionality is needed to extend the clock recovery period from default. [15:8] - The number of AXI4-Lite clocks (defined by the AXI4-Lite clock name: s_axi_aclk) equivalent to the recommended width of AUX pulse. Allowable values include: 0 (default), 8, 16, 24, 32, 40, and 48. As per the DisplayPort protocol specifications, AUX Pulse Width range = 0.4 to 0.6 μ s. [7:0] - Clock divider value For example, for AXI4-Lite clock of 50 MHz (= 20 ns), the filter width, when set to 24, falls in the allowable range as defined by the protocol spec. (20 \times 24 = 480) Program a value of 24 in this register.
0x008	R/W	RX_LINE_RESET_DISABLE. RX line reset disable. This register bit can be used to disable the end of line reset to the internal video pipe for reduced blanking video support. [3] - End of line reset disable to the MST video stream 4 [2] - End of line reset disable to the MST video stream 3 [1] - End of line reset disable to the MST video stream 2 [0] - End of line reset disable to the SST video stream or MST video stream 1
0x00C	R/W	DTG_ENABLE. Enables the display timing generator in the user interface. [0] - DTG_ENABLE: Set to 1 to enable the timing generator. The DTG should be disabled when the core detects the no-video pattern on the link.

Table 19: Receiver Core Configuration (cont'd)

Offset	Access Type	Description
0x010	R/W	<p>USER_PIXEL_WIDTH. Configures the number of pixels output through the user data interface. The Sink controller programs the pixel width to the active lane count (default). You can override this by writing a new value to this register. Use quad pixel mode in MST.</p> <p>[2:0]</p> <ul style="list-style-type: none"> 1 = Single pixel wide interface 2 = Dual pixel output mode. Valid for designs with 2 or 4 lanes. 4 = Quad pixel output mode. Valid for designs with 4 lanes only.
0x014	R/W	<p>INTERRUPT_MASK. Masks the specified interrupt sources from asserting the axi_init signal. When set to a 1, the specified interrupt source is masked. This register resets to all 1s at power up.</p> <p>[31] - Mask for Cable disconnect/unplug interrupt</p> <p>[30] - CRC test start interrupt</p> <p>[29] - Mask MST Act sequence received interrupt</p> <p>[28] - Mask interrupt generated when DPCD registers 0x1C0, 0x1C1 and 0x1C2 are written for allocation/de-allocation/partial deletion</p> <p>[27] - Audio packet FIFO overflow interrupt</p> <p>[18] - Training pattern 3 start interrupt</p> <p>[17] - Training pattern 2 start interrupt</p> <p>[16] - Training pattern 1 start interrupt</p> <p>[15] - Bandwidth change interrupt</p> <p>[14] - TRAINING_DONE</p> <p>[13] - DOWN_REQUEST_BUFFER_READY</p> <p>[12] - DOWN_REPLY_BUFFER_READ</p> <p>[11] - VC Payload De-allocated</p> <p>[10] - VC Payload Allocated</p> <p>[9] - EXT_PKT_RXD: Set to 1 when extension packet is received</p> <p>[8] - INFO_PKT_RXD: Set to 1 when info packet is received</p> <p>[6] - VIDEO: Set to 1 when valid video frame is detected on main link. Video interrupt is set after a delay of eight video frames following a valid scrambler reset character.</p> <p>[4] - TRAINING_LOST: Training has been lost on any one of the active lanes</p> <p>[3] - VERTICAL_BLANKING: Start of the vertical blanking interval</p> <p>[2] - NO_VIDEO: The no-video condition has been detected after active video received</p> <p>[1] - POWER_STATE: Power state change, DPCD register value 0x00600</p> <p>[0] - VIDEO_MODE_CHANGE: Resolution change, as detected from the MSA fields</p>
0x018	R/W	<p>MISC_CONTROL. Allows the host to instruct the receiver to pass the MSA values through unfiltered.</p> <p>[2] - When set to 1, I2C DEFERS is sent as AUX DEFERS to the source device.</p> <p>[1] - When set to 1, the long I2C write data transfers are responded to using DEFER instead of Partial ACKs.</p> <p>[0] - USE_FILTERED_MSA: When set to 0, this bit disables the filter on the MSA values received by the core. When set to 1, two matching values must be detected for each field of the MSA values before the associated register is updated internally.</p>
0x01C	WO	<p>SOFTWARE_RESET_REGISTER.</p> <p>[8] - Soft reset control to external HDCP FIFOs</p> <p>[7] - AUX Soft Reset: When set, AUX logic resets</p> <p>[0] - Soft Video Reset: When set, video logic resets. Reads return zeros.</p>

AUX Channel Status

Table 20: AUX Channel Status

Offset	Access Type	Description
0x020	RO	AUX_REQUEST_IN_PROGRESS. Indicates the receipt of an AUX Channel request [0] - 1 indicates a request is in progress.
0x024	RO	REQUEST_ERROR_COUNT. Provides a running total of errors detected on inbound AUX Channel requests. [7:0] - Error count, a write to register address 0x28 clears this counter.
0x028	R/W	REQUEST_COUNT. Provides a running total of the number of AUX requests received. [7:0] - Total AUX request count, a write to register 0x28 clears this counter.
0x02C	WO	HPD_INTERRUPT. Instructs the receiver core to assert an interrupt to the transmitter using the HPD signal. A read from this register always returns 0x0. [31:16] - HPD_INTERRUPT_LENGTH: Default value is 0. This field defines the length of the HPD pulse. The value should be given in microsecond units. For example for 750 μ s, program 750 in the register. [0] - Set to 1 to send the interrupt through the HPD signal. The HPD signal is brought low for 750 μ s to indicate to the source that an interrupt has been requested.
0x030	RO	REQUEST_CLOCK_WIDTH. Holds the half period of recovered AUX clock. [9:0] - Indicates the number of AXI_CLK cycles between sequential edges during the SYNC period of the most recent AUX request.
0x034	RO	REQUEST_COMMAND. Provides the most recent AUX command received. [3:0] - Provides the command field of the most recently received AUX request.
0x038	RO	REQUEST_ADDRESS. Contains the address field of the most recent AUX request. [19:0] - The twenty-bit address field from the most recent AUX request transaction is placed in this register. For I2C over AUX transactions, the address range is limited to the seven LSBs.
0x03C	RO	REQUEST_LENGTH. The length of the most recent AUX request is written to this register. The length of the AUX request is the value of this register plus one. [3:0] - Contains the length of the AUX request. Transaction lengths from 1 to 16 bytes are supported. For address only transactions, the value of this register will be 0.

Table 20: AUX Channel Status (cont'd)

Offset	Access Type	Description
0x040	RC	<p>INTERRUPT_CAUSE. Indicates the cause of a pending host interrupt. A read from this register clears all values. Write operation is illegal and clears the values.</p> <p>[31] - Cable disconnect/unplug interrupt</p> <p>[30] - CRC test start interrupt</p> <p>[29] - MST Act sequence received interrupt</p> <p>[28] - Interrupt generated when DPCD registers 0x1C0, 0x1C1, and 0x1C2 are written for allocation/de-allocation/partial deletion</p> <p>[27] - Audio packet FIFO overflow interrupt</p> <p>[18] - Training pattern 3 start interrupt</p> <p>[17] - Training pattern 2 start interrupt</p> <p>[16] - Training pattern 1 start interrupt</p> <p>[15] - Bandwidth change interrupt</p> <p>[14] - TRAINING_DONE: Set to 1 when training is done</p> <p>[13] - DOWN_REQUEST_BUFFER_READY: Set to 1 indicating availability of Down request</p> <p>[12] - DOWN_REPLY_BUFFER_READ: Set to 1 for a read event from Down Reply Buffer by upstream source</p> <p>[11] - VC Payload De-allocated: Set to 0 when de-allocation event occurs in controller</p> <p>[10] - VC Payload Allocated: Set to 1 when allocation event occurs in controller</p> <p>[9] - EXT_PKT_RXD: Set to 1 when extension packet is received</p> <p>[8] - INFO_PKT_RXD: Set to 1 when info packet is received</p> <p>[7] - Reserved</p> <p>[6] - VIDEO: Set to 1 when a valid video frame is detected on main link</p> <p>[5] - Reserved</p> <p>[4] - TRAINING_LOST: This interrupt is set when the receiver has been trained and subsequently loses clock recovery, symbol lock or inter-lane alignment, in any of the lanes</p> <p>[3] - VERTICAL_BLANKING: This interrupt is set at the start of the vertical blanking interval as indicated by the VerticalBlanking_Flag in the VB-ID field of the received stream</p> <p>[2] - NO_VIDEO: Receiver has detected the no-video flags in the VBID field after active video has been received</p> <p>[1] - POWER_STATE: Transmitter has requested a change in the current power state of the receiver core</p> <p>[0] - VIDEO_MODE_CHANGE: A change has been detected in the current video mode transmitted on the DisplayPort link as indicated by the MSA fields. The horizontal and vertical resolution parameters are monitored for changes.</p>

Table 20: AUX Channel Status (cont'd)

Offset	Access Type	Description
0x044	R/W	<p>INTERRUPT_MASK_1: Masks the specified interrupt sources from asserting the axi_init signal. When set to a 1, the specified interrupt source is masked. This register resets to all 1s at power up.</p> <p>[31] - TPS4 Interrupt (DPCD Wr) [30] - Access TP Lane Set Interrupt (DPCD Wr) [29] - Access Link Qual Pattern Interrupt (DPCD Wr) [28] - Access Symbol Error Counter Interrupt (DPCD Rd) [17] - Video Interrupt - Stream 4 [16] - Vertical Blanking Interrupt - Stream4 [15] - No Video Interrupt - Stream 4 [14] - Mode Change Interrupt - Stream 4 [13] - Info Packet Received - Stream 4 [12] - Ext Packet Received - Stream 4 [11] - Video Interrupt - Stream 3 [10] - Vertical Blanking Interrupt - Stream 3 [9] - No Video Interrupt - Stream 3 [8] - Mode Change Interrupt - Stream 3 [7] - Info Packet Received - Stream 3 [6] - Ext Packet Received - Stream 3 [5] - Video Interrupt - Stream 2 [4] - Vertical Blanking Interrupt - Stream 2 [3] - No Video Interrupt - Stream 2 [2] - Mode Change Interrupt - Stream 2 [1] - Info Packet Received - Stream 2 [0] - Ext Packet Received - Stream 2</p>

Table 20: AUX Channel Status (cont'd)

Offset	Access Type	Description
0x048	RC	<p>INTERRUPT_CAUSE_1: Indicates the cause of a pending host interrupt. A read from this register clears all values. A write operation would be illegal and would clear all values as well. These bits have the same function as those described in the Interrupt Cause register of stream 1. Reserved bits return 0. See offset 0x040 for more description on each interrupt.</p> <p>[31] - TPS4 Interrupt (DPCD Wr)</p> <p>[30] - Access TP Lane Set Interrupt (DPCD Wr)</p> <p>[29] - Access Link Qual Pattern Interrupt (DPCD Wr)</p> <p>[28] - Access Symbol Error Counter Interrupt (DPCD Rd)</p> <p>[17] - Video Interrupt - Stream 4</p> <p>[16] - Vertical Blanking Interrupt - Stream4</p> <p>[15] - No Video Interrupt - Stream 4</p> <p>[14] - Mode Change Interrupt - Stream 4</p> <p>[13] - Info Packet Received - Stream 4</p> <p>[12] - Ext Packet Received - Stream 4</p> <p>[11] - Video Interrupt - Stream 3</p> <p>[10] - Vertical Blanking Interrupt - Stream 3</p> <p>[9] - No Video Interrupt - Stream 3</p> <p>[8] - Mode Change Interrupt - Stream 3</p> <p>[7] - Info Packet Received - Stream 3</p> <p>[6] - Ext Packet Received - Stream 3</p> <p>[5] - Video Interrupt - Stream 2</p> <p>[4] - Vertical Blanking Interrupt - Stream 2</p> <p>[3] - No Video Interrupt - Stream 2</p> <p>[2] - Mode Change Interrupt - Stream 2</p> <p>[1] - Info Packet Received - Stream 2</p> <p>[0] - Ext Packet Received - Stream 2</p>
0x050	R/W	<p>HSYNC_WIDTH. The display timing generator control logic outputs a fixed length, active-High pulse for the horizontal sync. The timing of this pulse might be controlled by setting this register appropriately. The default value of this register is 0x0F0F.</p> <p>[15:8] - HSYNC_FRONT_PORCH: Defines the number of video clock cycles to place between the last pixel of active data and the start of the horizontal sync pulse.</p> <p>[7:0] - HSYNC_PULSE_WIDTH: Specifies the number of clock cycles the horizontal sync pulse is asserted. The vid_hsync signal is High for the specified number of clock cycles.</p>
0x058	R/W	<p>VSYNC_WIDTH. The display timing generator control logic outputs a fixed length of 63 RX video clocks, active-High pulse for the vertical sync pulse. The timing of this pulse might be controlled by setting this register appropriately. The default value of this register is 0x003F.</p> <p>[15:0] - VSYNC_WIDTH: Defines the number of RX video clock cycles the vertical sync pulse is asserted. The minimum value to be programmed in this register is 0x003F. User can configure the register based on actual VSYNC duration based on MSA.</p>
0x060	R/W	<p>[7:0] - FAST_I2C_DIVIDER. Fast I2C mode clock divider value. Set this value to (AXI4-Lite clock frequency/10) - 1. Valid only for DPCD 1.4.</p>
0x064	R/W	<p>[31] - Set to override Training Pattern 1 (TP1) score.</p> <p>[14:0] - Training pattern1 (TP1) score to override.</p>
0x068	R/W	<p>[31] - Set to override the Training Pattern2/3 scores.</p> <p>[12:0] - Training pattern2/3 (TP2/TP3) score to override.</p>
0x06C	RO	<p>Provides the contents of DPCD registers 0x1C0, 0x1C1, and 0x1C2.</p> <p>[21:16] - Time slot count</p> <p>[13:8] - Starting time slot</p> <p>[5:0] - VC Payload ID</p>

Table 20: AUX Channel Status (cont'd)

Offset	Access Type	Description
0x074	R/W	[5] - Enable CRC support. The CRC has to be calculated outside the DisplayPort IP and the values have to be provided in 0x078, 0x07C, and 0x080. [3:0] - CRC change count to be configured by SW
0x078	R/W	CRC for Red color
0x07C	R/W	CRC for Green color
0x080	R/W	CRC for Blue color

DPCD Fields

Table 21: DPCD Fields

Offset	Access Type	Description
0x084	R/W	LOCAL_EDID_VIDEO. Indicates the presence of EDID information for the video stream. [0] - Set to 1 to indicate to the transmitter through the DPCD registers that the receiver supports local EDID information.
0x088	R/W	LOCAL_EDID_AUDIO. Indicates the presence of EDID information for the audio stream. [0] - Set to 1 to indicate to the transmitter through the DPCD registers that the receiver supports local EDID information
0x08C	R/W	REMOTE_COMMAND. General byte for passing remote information to the transmitter. [7:0] - Remote data byte.
0x090	R/W	DEVICE_SERVICE_IRQ. Indicates DPCD DEVICE_SERVICE_IRQ_VECTOR state. [4] - Set to 1 to indicate a new DOWN Reply Buffer Message is ready. [1] - Reflects SINK_SPECIFIC_IRQ state of DPCD 0x201 register. This bit is RO. [0] - Set to 1 to indicate a new command. Indicates a new command present in the REMOTE_COMMAND register. A Write of 0x1 to this register sets the DPCD register DEVICE_SERVICE_IRQ_VECTOR (0x201), REMOTE_CONTROL_PENDING bit. A write of 0x0 to this register has no effect. Refer to DPCD register section of the standard for more details. Reads from this register reflect the state of DPCD register.
0x094	R/W	VIDEO_UNSUPPORTED. DPCD register bit to inform the transmitter that video data is not supported. [0] - Set to 1 when video data is not supported.
0x098	R/W	AUDIO_UNSUPPORTED. DPCD register bit to inform the transmitter that audio data is not supported [0] - Set to 1 when audio data is not supported.
0x09C	R/W	Override LINK_BW_SET. This register can be used to override LINK_BW_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. LINK_BW_SET corresponds to the 00100h and 0x0001h DPCD address space. <ul style="list-style-type: none"> [4:0] - Link rate override value for DisplayPort Standard v1.4 designs [3:0] - Link rate override value for DisplayPort Standard v1.4 designs 0x6 = 1.62 Gb/s 0xA = 2.7 Gb/s 0x14 = 5.4 Gb/s 0x1E = 8.1 Gb/s

Table 21: DPCD Fields (cont'd)

Offset	Access Type	Description
0x0A0	R/W	Override LANE_COUNT_SET. This register can be used to override LANE_COUNT_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. LANE_COUNT_SET corresponds to the 00101h and 0x00002h DPCD address space. [7] - ENHANCED_FRAME_CAP: Capability override [6] - TPS3_SUPPORTED: Capability override for DisplayPort Standard v1.4 protocol designs only. Reserved for v1.1a protocol. [4:0] - Lane count override value (1, 2, or 4 lanes).
0x0A4	R/W	Override TRAINING_PATTERN_SET. This register can be used to override TRAINING_PATTERN_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. TRAINING_PATTERN_SET corresponds to the 00102h DPCD address space. [15:8] - TRAINING_AUX_RD_INTERVAL (the values are based on DisplayPort Standard v1.4 protocol). [7:6] - SYMBOL ERROR COUNT SEL Override [5] - SCRAMBLING_DISABLE Override [4] - RECOVERED_CLOCK_OUT_EN Override [3:0] - Training Pattern Select
0x0A8	R/W	Override TRAINING_LANE0_SET. This register can be used to override TRAINING_LANE0_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. TRAINING_LANE0_SET corresponds to the 00103h DPCD address space. [7:6] - Reserved [5] - MAX_PRE-EMPHASIS_REACHED override [4:3] - PRE-EMPHASIS_SET override [2] - MAX_SWING_REACHED override [1:0] - VOLTAGE SWING SET override
0x0AC	R/W	Override TRAINING_LANE1_SET. This register can be used to override TRAINING_LANE1_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. Same as Override TRAINING_LANE0_SET. TRAINING_LANE1_SET corresponds to the 00104h DPCD address space.
0x0B0	R/W	Override TRAINING_LANE2_SET. This register can be used to override TRAINING_LANE2_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. Same as Override TRAINING_LANE0_SET. TRAINING_LANE2_SET corresponds to the 00105h DPCD address space.
0x0B4	R/W	Override TRAINING_LANE3_SET. This register can be used to override TRAINING_LANE3_SET in the DPCD register set. Register 0x0B8 (direct_dpcd_access) must be set to 1 to override DPCD values. Same as Override TRAINING_LANE0_SET. TRAINING_LANE3_SET corresponds to the 00106h DPCD address space.
0x0B8 *	R/W	Override DPCD Control Register. Setting this register to 0x1 enables AXI/APB write access to DPCD capability structure.
0x0BC	R/W	Override DPCD DOWNSPREAD control field. Register 0x0B8 must be set to 1 to override DPCD values. [0] - MAX_DOWNSPREAD Override
0x0C0	R/W	Override DPCD LINK_QUAL_LANE0_SET field for DPCD1.4 version only. Register 0x0B8 must be set to 1 to override DPCD values. [2:0] - LINK_QUAL_LANE0_SET override
0x0C4	R/W	Override DPCD LINK_QUAL_LANE1_SET field for DPCD1.4 version only. Register 0x0B8 must be set to 1 to override DPCD values. [2:0] - LINK_QUAL_LANE1_SET override

Table 21: DPCD Fields (cont'd)

Offset	Access Type	Description
0x0C8	R/W	Override DPCD LINK_QUAL_LANE2_SET field for DPCD1.4 version only. Register 0x0B8 must be set to 1 to override DPCD values. [2:0] - LINK_QUAL_LANE2_SET override
0x0CC	R/W	Override DPCD LINK_QUAL_LANE3_SET field for DPCD1.4 version only. Register 0x0B8 must be set to 1 to override DPCD values. [2:0] - LINK_QUAL_LANE3_SET override
0x0D0	R/W	[8] - Clears the VCPayload Table contents. This bit is auto cleared. [7:5] - Unused [4] - VCPayload Table update bit. SW writes this bit with which the core updates the DPCD register 0x2C0 bit to 1. This bit is used when VC Payload table is controlled through SW [2] - Set to 1 to override act trigger. Usually, the VCPayload active buffer updates on receiving ACT trigger sequence. This bit can be set when the VC payload table is in SW control. This bit is for advanced users only. [1] - Set to 1 to enable SW control over VCpayload table. This provides SW write access to offset 0x800-0x8FF. This bit is for advanced users only. [0] - MST CAPABILITY: Enable or Disable MST capability. Set to 1 to enable MST capability. This bit should be set during the configuration programming stage only.
0x0D4	R/W	[6:0] - Sink device count: Recommended to be programmed during initialization of the Sink device. In SST mode, the value should be 1.
0x0E0	R/W	GUID word 0. Allows you to set up GUID if required from host interface. Valid for DPCD1.4 version only. [31:0] - Lower 4 bytes of GUID DPCD field
0x0E4	R/W	GUID word 1. Allows you to set up GUID if required from host interface. Valid for DPCD1.4 version only. [31:0] - Bytes 4 to 7 of GUID DPCD field
0x0E8	R/W	GUID word 2. Allows you to set up GUID if required from host interface. Valid for DPCD1.4 version only. [31:0] - Bytes 8 to 11 of GUID DPCD field
0x0EC	R/W	GUID word 3. Allows you to set up GUID if required from host interface. Valid for DPCD1.4 version only. [31:0] - Bytes 12 to 15 of GUID DPCD field
0x0F0	R/W	GUID Override. [0]: When set to 0x1, the GUID field of the DPCD reflects the data written in GUID Words 0 to 3. Valid for DPCD1.4 version only. When this register is set to 0x1, GUID field of DPCD becomes read only and source-aux writes are NACK-ed.

Core ID

Table 22: Core ID

Offset	Access Type	Description
0x0FC	RO	<p>CORE_ID. Returns the unique identification code of the core and the current revision level.</p> <p>[31:24] - DisplayPort protocol major version</p> <p>[23:16] - DisplayPort protocol minor version</p> <p>[15:8] - DisplayPort protocol revision</p> <p>[7:0] - Core mode of operation</p> <ul style="list-style-type: none"> 0x00: Transmit 0x01: Receive <p>The CORE_ID value for the protocol and core is DisplayPort Standard v1.4 with a Receive core: 32'h01_04_00_01.</p>
0x110	RO	<p>USER_FIFO_OVERFLOW. This status bit indicates an overflow of the user data FIFO of pixel data. This event might occur if the input pixel clock is not fast enough to support the current DisplayPort link width and link speed.</p> <p>[11] - Video Timing FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the Video Timing FIFO is overflowed for stream4.</p> <p>[10] - Video Timing FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the Video Timing FIFO is overflowed for stream3.</p> <p>[9] - Video Timing FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the Video Timing FIFO is overflowed for stream2.</p> <p>[8] - Video Timing FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the Video Timing FIFO is overflowed.</p> <p>[7] - Video Unpack FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the Video unpack FIFO is overflowed for stream4.</p> <p>[6] - Video Unpack FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the Video unpack FIFO is overflowed for stream3.</p> <p>[5] - Video Unpack FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the Video unpack FIFO is overflowed for stream2.</p> <p>[4] - Video Unpack FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the Video unpack FIFO is overflowed.</p> <p>[3] - FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the internal FIFO has detected an overflow condition for Stream 4. This bit clears upon read.</p> <p>[2] - FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the internal FIFO has detected an overflow condition for Stream 3. This bit clears upon read.</p> <p>[1] - FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the internal FIFO has detected an overflow condition for Stream 2. This bit clears upon read.</p> <p>[0] - FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the internal FIFO has detected an overflow condition for Stream 1. This bit clears upon read.</p>
0x114	RO	<p>USER_VSYNC_STATE. Provides a mechanism for the host processor to monitor the state of the video datapath. This bit is set when vsync is asserted.</p> <p>[3] - State of the vertical sync pulse for Stream 4.</p> <p>[2] - State of the vertical sync pulse for Stream 3.</p> <p>[1] - State of the vertical sync pulse for Stream 2.</p> <p>[0] - State of the vertical sync pulse for Stream 1.</p>

PHY Configuration and Status

Table 23: PHY Configuration and Status

Offset	Access Type	Description
0x208	RO	<p>PHY_STATUS. Provides status for the receiver core PHY.</p> <p>[31:30] - RX buffer status, lane 3</p> <p>[29:28] - RX buffer status, lane 2</p> <p>[27:26] - RX buffer status, lane 1</p> <p>[25:24] - RX buffer status, lane 0</p> <p>[23] - RXBYTEISALIGNED status of PHY, lane 3</p> <p>[22] - RXBYTEISALIGNED status of PHY, lane 2</p> <p>[21] - RXBYTEISALIGNED status of PHY, lane 1</p> <p>[20] - RXBYTEISALIGNED status of PHY, lane 0</p> <p>[15:14] - RX voltage low, lanes 2 and 3</p> <p>[13:12] - RX voltage low, lanes 0 and 1</p> <p>[11:10] - PRBS error, lanes 2 and 3</p> <p>[9:8] - PRBS error, lanes 0 and 1</p> <p>[7] - Receiver Clock locked</p> <p>[6] - FPGA general interconnect clock PLL locked</p> <p>[5] - PLL for lanes 2 and 3 locked (Tile 1)</p> <p>[4] - PLL for lanes 0 and 1 locked (Tile 0)</p> <p>[3:2] - Reset done for lanes 2 and 3 (Tile 1)</p> <p>[1:0] - Reset done for lanes 0 and 1 (Tile 0)</p>

Table 23: PHY Configuration and Status (cont'd)

Offset	Access Type	Description
0x214	R/W	<p>MIN_VOLTAGE_SWING. Some DisplayPort implementations require the transmitter to set a minimum voltage swing during training before the link can be reliably established. This register is used to set a minimum value which must be met in the TRAINING_LANEX_SET DPCD registers. The internal training logic forces training to fail until this value is met.</p> <p>Note: It is not recommended to change this register value.</p> <p>[31] - PHY Test Mode (Advanced. Used to test PHY.) [23:14] - PREEMP_TABLE (only for Advanced users) 15:14: Iteration 1 pre-emp request level 17:16: Iteration 2 pre-emp request level 19:18: Iteration 3 pre-emp request level 21:20: Iteration 4 pre-emp request level 23:22: Iteration 5 pre-emp request level [13:12] - SET_PREEMP (only for Advanced users) [11:10] - Channel Equalization options (only for Advanced users)</p> <ul style="list-style-type: none"> 00: Default (pre-emphasis adjust request is incremented one by per iteration until maximum pre-emphasis limit (SET_PREEMP) is reached) 01: Hold pre-emphasis adjust request to SET_PREEMP for all iterations 10: Not applicable 11: Pick values from PREEMP_TABLE <p>[9:8] - SET_VSWING (only for Advanced users). Default value is 0x0000. [6:4] - VSWING_SWEEP_CNT (only for Advanced users) [3:2] - Clock Recovery options (only for Advanced users)</p> <ul style="list-style-type: none"> 00: Default (Voltage swing adjust request will get incremented one by every iteration) 01: Increment adjust request every 4 or VSWING_SWEEP_CNT iterations 10: Hold adjust request to SET_VSWING value for all iterations 11: Not applicable <p>[1:0] - The minimum voltage swing setting matches the values defined in the DisplayPort Standard for the TRAINING_LANEX_SET register.</p>
0x21C	R/W	<p>CDR_CONTROL_CONFIG.</p> <p>[30] - Disable Training Timeout [19:0] - Controls the CDR tDLOCK timeout value. The counter is run using the AXI4-Lite clock in the PHY Module. Default value is 20'h1388.</p>
0x220	R/W	<p>BS_IDLE_TIME. Blanking start symbol idle time. Default is 0x1312D00 (200 ms) considering 100 MHz AXI4-Lite clock frequency. The value is based on AXI4-Lite clock frequency and you are expected to update as needed.</p> <p>[31:0] - The value written in this register is used in DisplayPort Sink to detect cable disconnect or unplug event. DisplayPort sink checks the Blanking Start symbol over the link for the specified period and generates cable unplug interrupt. The timeout counter is loaded with this register value which is working with AXI clock. Cable unplug counter is a free running counter and it reloads with BS_IDLE_TIME as and when it receives the BS character over link or when it's time out by reaching maximum count.</p>

DisplayPort Audio

Table 24: DisplayPort Audio

Offset	Access Type	Description
12'h300	R/W	RX_AUDIO_CONTROL. This register enables audio stream packets in main link. [5:4]: MST Audio Stream no. N/A for SST [3]: Audio Enable for STREAM 4 in MST. N/A for SST [2]: Audio Enable for STREAM 3 in MST. N/A for SST [1]: Audio Enable for STREAM 2 in MST. N/A for SST [0]: Audio Enable for SST. In MST, Audio Enable for STREAM 1
12'h304	RO	RX_AUDIO_INFO_DATA [31:0] Word formatted as per CEA 861-C Info Frame. Total of eight words should be read. 1st word <ul style="list-style-type: none"> [31:24] = HB3 [23:16] = HB2 [15:8] = HB1 [7:0] = HB0 2nd word - DB3,DB2,DB1,DB0 8th word - DB27,DB26,DB25,DB24 The data bytes DB1...DBN of CEA Info frame are mapped as DB0-DBN-1. Info frame data is copied into these registers (read only).
12'h324	RO	RX_AUDIO_MAUD. M value of audio stream as decoded from Audio time stamp packet by the sink (read only). [31:24] - Reserved [23:0] - MAUD
12'h328	RO	RX_AUDIO_NAUD. N value of audio stream as decoded from Audio time stamp packet by the sink (read only). [31:24] - Reserved [23:0] - NAUD
12'h32C	RO	RX_AUDIO_STATUS. [9] - Extension Packet Received. Resets automatically after all words (9) are read. Blocks new packet until host reads the data. [8:3] - Reserved [2:1] - RS Decoder Error Counter. Used for debugging purpose. [0] - Info Packet Received. Resets automatically after all info words (eight) are read. Blocks new packet until host reads the data.

Table 24: DisplayPort Audio (cont'd)

Offset	Access Type	Description
12'h330 to 12'h350	RO	<p>RX_AUDIO_EXT_DATA</p> <p>[31:0] - Word formatted as per extension packet described in protocol standard. Packet length is fixed to 32 bytes in Sink controller.</p> <p>You should convey this information to Source using the vendor fields and ensure proper packet size transmission is done by the Source controller. Total of nine words should be read. Extension packet address space can be used to hold the audio copy management packet/ISRC packet/VSC packets.</p> <p>1st word</p> <ul style="list-style-type: none"> [31:24] = HB3 [23:16] = HB2 [15:8] = HB1 [7:0] = HB0 <p>2nd word - DB3,DB2,DB1,DB0</p> <p>....</p> <p>9th word - DB31,DB30,DB29,DB28</p> <p>Extension packet data is copied into these registers (read only). This is a key-hole memory. So, nine reads from this address space is required.</p>

DPCD Configuration Space

For detailed descriptions of these registers, see the *VESA DisplayPort Standard v1.4* ([VESA website](http://www.vesa.org/displays/vdp14)) standard.

Table 25: DPCD Configuration Space

Offset	Access Type	Description
0x400	RO	<p>DPCD_LINK_BW_SET. Link bandwidth setting.</p> <p>[7:0] - Set to 0x0A when the link is configured for 2.7 Gb/s or 0x06 when configured for 1.62 Gb/s or 0x14 when link is configured for 5.4 Gb/s.</p>
0x404	RO	<p>DPCD_LANE_COUNT_SET. Number of lanes enabled by the transmitter.</p> <p>[4:0] - Contains the number of lanes that are currently enabled by the attached transmitter. Valid values fall in the range of 1-4.</p>
0x408	RO	<p>DPCD_ENHANCED_FRAME_EN. Indicates that the transmitter has enabled the enhanced framing symbol mode.</p> <p>[0] - Set to 1 when enhanced framing mode is enabled.</p>
0x40C	RO	<p>DPCD_TRAINING_PATTERN_SET. Current value of the training pattern registers.</p> <p>[1:0] - TRAINING_PATTERN_SET: Set the link training pattern according to the 2-bit code:</p> <ul style="list-style-type: none"> 000 = Training not in progress 001 = Training pattern 1 010 = Training pattern 2 011 = Training pattern 3 111 = Training pattern 4

Table 25: DPCD Configuration Space (cont'd)

Offset	Access Type	Description
0x410	RO	DPCD_LINK_QUALITY_PATTERN_SET. Current value of the link quality pattern field of the DPCD training pattern register. [1:0] - transmitter is sending the link quality pattern: <ul style="list-style-type: none"> 00 = Link quality test pattern not transmitted 01 = D10.2 test pattern (unscrambled) transmitted 10 = Symbol Error Rate measurement pattern 11 = PRBS7 transmitted
0x414	RO	DPCD_RECOVERED_CLOCK_OUT_EN. Value of the output clock enable field of the DPCD training pattern register. ixia_locid="41">[0] - Set to 1 to output the recovered receiver clock on the test port.
0x418	RO	DPCD_SCRAMBLING_DISABLE. Value of the scrambling disable field of the DPCD training pattern register. By default, scrambling is disabled. [0] - Set to 1 when the transmitter has disabled the scrambler and transmits all symbols.
0x41C	RO	DPCD_SYMBOL_ERROR_COUNT_SELECT. Current value of the symbol error count select field of the DPCD training pattern register. [1:0] - SYMBOL_ERROR_COUNT_SEL: <ul style="list-style-type: none"> 00 = Disparity error and illegal symbol error 01 = Disparity error 10 = Illegal symbol error 11 = Reserved
0x420	RO	DPCD_TRAINING_LANE_0_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. [5] - MAX_PRE-EMPHASIS_REACHED: Set to 1 when the maximum pre-emphasis setting is reached. [4:3] - PRE-EMPHASIS_SET <ul style="list-style-type: none"> 00 = Training Pattern 2 without pre-emphasis 01 = Training Pattern 2 with pre-emphasis level 1 10 = Training Pattern 2 with pre-emphasis level 2 11 = Training Pattern 2 with pre-emphasis level 3 [2] - MAX_SWING_REACHED: Set to 1 when the maximum driven current setting is reached. [1:0] - VOLTAGE_SWING_SET <ul style="list-style-type: none"> 00 = Training Pattern 1 with voltage swing level 0 01 = Training Pattern 1 with voltage swing level 1 10 = Training Pattern 1 with voltage swing level 2 11 = Training Pattern 1 with voltage swing level 3
0x424	RO	DPCD_TRAINING_LANE_1_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. The fields of this register are identical to DPCD_TRAINING_LANE_0_SET.
0x428	RO	DPCD_TRAINING_LANE_2_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. The fields of this register are identical to DPCD_TRAINING_LANE_0_SET.
0x42C	RO	DPCD_TRAINING_LANE_3_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. The fields of this register are identical to DPCD_TRAINING_LANE_0_SET.
0x430	RO	DPCD_DOWNSPREAD_CONTROL. The transmitter uses this bit to inform the receiver core that downspreading has been enabled. [0] - SPREAD_AMP: Set to 1 for 0.5% spreading or 0 for none.

Table 25: DPCD Configuration Space (cont'd)

Offset	Access Type	Description
0x434	RO	DPCD_MAIN_LINK_CHANNEL_CODING_SET. 8B/10B encoding can be disabled by the transmitter through this register bit. [0] - Set to 0 to disable 8B/10B channel coding. The default is 1.
0x438	RO	PCD_SET_POWER_STATE. Power state requested by the source core. On reset, power state is set to power down mode. [1:0] - Requested power state <ul style="list-style-type: none"> 00 = Reserved 01 = state D0, normal operation 10 = state D3, power down mode 11 = Reserved
0x43C	RO	DPCD_LANE01_STATUS. Value of the lane 0 and lane 1 training status registers. Write-Only Access [31] - Override Lane Set Registers [27:26] - Override Pre-Emphasis Level [25:24] - Override Voltage Level Read-Only Access [15:14] - Lane 1 Adjust Pre-Emphasis Value [13:12] - Lane 1 Adjust Voltage Value [11:10] - Lane 0 Adjust Pre-Emphasis Value [9:8] - Lane 0 Adjust Voltage Value [6] - LANE_1_SYMBOL_LOCKED. Set to 1 when, the 8B10B pattern is locked and TP2/3 pattern is locked on LANE_1. [5] - LANE_1_CHANNEL_EQ_DONE. Set to 1, when the lane alignment is done on LANE_1. [4] - LANE_1_CLOCK_RECOVERY_DONE. Set to 1, when D10.2 symbol is locked on LANE_1. [2] - LANE_0_SYMBOL_LOCKED. Set to 1 when, the 8B10B pattern is locked and TP2/3 pattern is locked on LANE_0. [1] - LANE_0_CHANNEL_EQ_DONE. Set to 1, when the lane alignment is done on LANE_0. [0] - LANE_0_CLOCK_RECOVERY_DONE. Set to 1, when D10.2 symbol is locked on LANE_0.
0x440	RO	DPCD_LANE23_STATUS. Value of the lane 2 and lane 3 training status registers. Read-Only Access [15:14] - Lane 3 Adjust Pre-Emphasis Value [13:12] - Lane 3 Adjust Voltage Value [11:10] - Lane 2 Adjust Pre-Emphasis Value [9:8] - Lane 2 Adjust Voltage Value [6] - LANE_3_SYMBOL_LOCKED. Set to 1 when, the 8B10B pattern is locked and TP2/3 pattern is locked on LANE_3. [5] - LANE_3_CHANNEL_EQ_DONE. Set to 1, when the lane alignment is done on LANE_3. [4] - LANE_3_CLOCK_RECOVERY_DONE. Set to 1, when D10.2 symbol is locked on LANE_3. [2] - LANE_2_SYMBOL_LOCKED. Set to 1 when, the 8B10B pattern is locked and TP2/3 pattern is locked on LANE_2. [1] - LANE_2_CHANNEL_EQ_DONE. Set to 1, when the lane alignment is done on LANE_2. [0] - LANE_2_CLOCK_RECOVERY_DONE. Set to 1, when D10.2 symbol is locked on LANE_2.
0x444	RO	SOURCE_OUI_VALUE. Value of the Organizationally Unique Identifier (OUI) as written by the transmitter via the DPCD register AUX transaction. [23:0] - Contains the value of the OUI set by the transmitter. This value might be used by the host policy maker to enable special functions across the link.

Table 25: DPCD Configuration Space (cont'd)

Offset	Access Type	Description
0x448	RC/RO	SYM_ERR_CNT01. Reports symbol error counter of lanes 0 and 1. The lane 0 and lane 1 error counts are cleared when this register is read. [31] - Lane 1 error count valid [30:16] - Lane 1 error count [15] - Lane 0 error count valid [14:0] - Lane 0 error count
0x44C	RC	SYM_ERR_CNT23. Reports symbol error counter of lanes 2 and 3. The lane 2 and lane 3 error counts are cleared when this register is read. [31] - Lane 3 error count valid [30:16] - Lane 3 error count [15] - Lane 2 error count valid [14:0] - Lane 2 error count
0x452	RO	DPCD Value written by DP Source [31:0] - {5'd0, Link_Qual_Lane3, 5'd0, Link_Qual_Lane2, 5'd0, Link_Qual_Lane2, 5'd0, Link_Qual_Lane0}
0x45C	R/W	[31:16] - PRBS Error Counter - Lane 1 {Valid, 15-bit counter value} [15:0] - PRBS Error Counter - Lane 0 {Valid, 15-bit counter value}
0x460	R/W	[31:16] - PRBS Error Counter - Lane 3 {Valid, 15-bit counter value} [15:0] - PRBS Error Counter - Lane 2 {Valid, 15-bit counter value}

MSA Values

Table 26: MSA Values

Offset	Access Type	Description
0x500	RO	MSA_HRES. The horizontal resolution detected in the Main Stream Attributes. [15:0] - Represents the number of pixels in a line of video
0x504	RO	MSA_HSPOL. Horizontal sync polarity. [0] - Indicates the polarity of the horizontal sync as requested by the transmitter
0x508	RO	MSA_HSWIDTH. Specifies the width of the horizontal sync pulse. [14:0] - Specifies the width of the horizontal sync in terms of the recovered video clock
0x50C	RO	MSA_HSTART. This main stream attribute is the number of clock cycles between the leading edge of the horizontal sync and the first cycle of active data. [15:0] - Number of blanking cycles before active data
0x510	RO	MSA_HTOTAL. Tells the receiver core how many video clock cycles occur between leading edges of the horizontal sync pulse. [15:0] - Total number of video clocks in a line of data
0x514	RO	MSA_VHEIGHT. Total number of active video lines in a frame of video. [15:0] - Vertical resolution of the received video
0x518	RO	MSA_VSPOL. Specifies the vertical sync polarity requested by the transmitter. [0] - A value of 1 in this register indicates an active-High vertical sync. A 0 indicates an active-Low vertical sync.

Table 26: MSA Values (cont'd)

Offset	Access Type	Description
0x51C	RO	MSA_VSWIDTH. The transmitter uses this value to specify the width of the vertical sync pulse in lines. [14:0] - Specifies the number of lines between the leading and trailing edges of the vertical sync pulse.
0x520	RO	MSA_VSTART. This main stream attribute specifies the number of lines between the leading edge of the vertical sync pulse and the first line of active data. [15:0] - Number of blanking lines before the start of active data
0x524	RO	MSA_VTOTAL. Total number of lines between sequential leading edges of the vertical sync pulse. [15:0] - Total number of lines per video frame is contained in this value
0x528	RO	MSA_MISC0. Contains the value of the MISC0 attribute data. [7:5] - COLOR_DEPTH: Number of bits per color/component. [4] - YCbCr_COLOR: Set to 1 (ITU-R BT709-5) or 0 (ITU-R BT601-5). [3] - DYNAMIC_RANGE: Set to 1 (CEA range) or 0 (VESA range). [2:1] - COMPONENT_FORMAT: <ul style="list-style-type: none"> 00 = RGB 01 = YCbCr 4:2:2 10 = YCbCr 4:4:4 11 = Reserved [0] - CLOCK_MODE: <ul style="list-style-type: none"> 0 = Asynchronous clock mode 1 = Synchronous clock mode
0x52C	RO	MSA_MISC1. Contains the value of the MISC1 attribute data. [7] - Implements the attribute information contained in the DisplayPort MISC1 register described in section 2.2.4 of the standard [6:3] - RESERVED: Bits are always set to 0 [2:1] - STEREO_VIDEO: Used only when stereo video sources are being transmitted. See the <i>VESA DisplayPort Standard v1.4</i> (VESA website) section 2.24 for more information. [0] - INTERLACED_EVEN: 1 indicates that the number of lines per frame is an even number
0x530	RO	MSA_MVID. This attribute value is used to recover the video clock from the link clock. The recovered clock frequency depends on this value as well as the CLOCK_MODE and MSA_NVID registers. [23:0] - MVID: Value of the clock recovery M value.
0x534	RO	MSA_NVID. This attribute value is used to recover the video clock from the link clock. The recovered clock frequency depends on this value as well as the CLOCK_MODE and MSA_MVID registers. [23:0] - NVID: Value of the clock recovery N value.
0x538	RO	MSA_VBID. The most recently received VB-ID value is contained in this register. [7:0] - VBID: See Table 2-3 (pg. 44) in the <i>VESA DisplayPort Standard v1.4</i> (VESA website) for more information. The default value is 0x19.

Vendor-Specific DPCD

Table 27: Vendor-Specific DPCD

Offset	Access Type	Description
0xE00 to 0xEFC	R/W	<p>SOURCE_DEVICE_SPECIFIC_FIELD. Access to Source specific field of DPCD address space. AXI accesses are all word-based (32 bits).</p> <ul style="list-style-type: none"> 0xE00 to 0xE02: Read Only (IEEE OUI Value Programmed by Source) 0xE03 to 0xEFF: Write/Read
0xF00 to 0xFFC	R/W	<p>SINK_DEVICE_SPECIFIC_FIELD. Access to Sink specific field of DPCD address space. AXI accesses are all word-based (32 bits).</p> <ul style="list-style-type: none"> 0xF00 to 0xF02: Read Only (IEEE OUI Value from GUI) 0xF03 to 0xFFF: Write/Read

AXI IIC Registers

For details about the AXI IIC registers, see the *AXI IIC Bus Interface LogiCORE IP Product Guide* ([PG090](#)).

AXI Timer Registers

For details about the AXI Timer registers, see the *AXI Timer LogiCORE IP Product Guide* ([PG079](#)).

HDCP Registers

For details about the HDCP registers, see the *HDCP 1.x Product Guide* ([PG224](#)).

Designing with the Subsystem

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

DisplayPort Overview

The Sink core requires a series of initialization steps before it begins receiving video. These steps include bringing up the Physical Interface (PHY) and setting the internal registers for the proper management of the AUX channel interface.

The Sink policy maker in the example design provides the basic steps for initialization. The following Sink registers are recommended to program after power up:

- Override LINK_BW_SET
- Override LANE_COUNT_SET
- Override DPCD DOWNSPREAD
- Sink Device Count

These values indicate key DPCD capabilities of the Sink.

The DisplayPort link Hot Plug Detect signal is tied directly to the state of the receiver core enable bit. Until the core is enabled, the receiver does not respond to any AUX transactions or main link video input.

While the Display Timing Generator might be enabled at any time, Xilinx[®] recommends keeping the DTG disabled until the receiver core policy maker detects the start of active video. This condition can be detected initially through the assertion of the MODE_INTERRUPT which detects the change in the vertical and horizontal resolution values.

Upon receipt of the interrupt, the receiver policy maker should verify the values of the Main Stream Attributes (offset 0x500–0x530) to ensure that the requested video mode is within the range supported by the Sink device. If these values are within range, the Display Timing Generator should be enabled to begin passing valid video frames through your data interface.

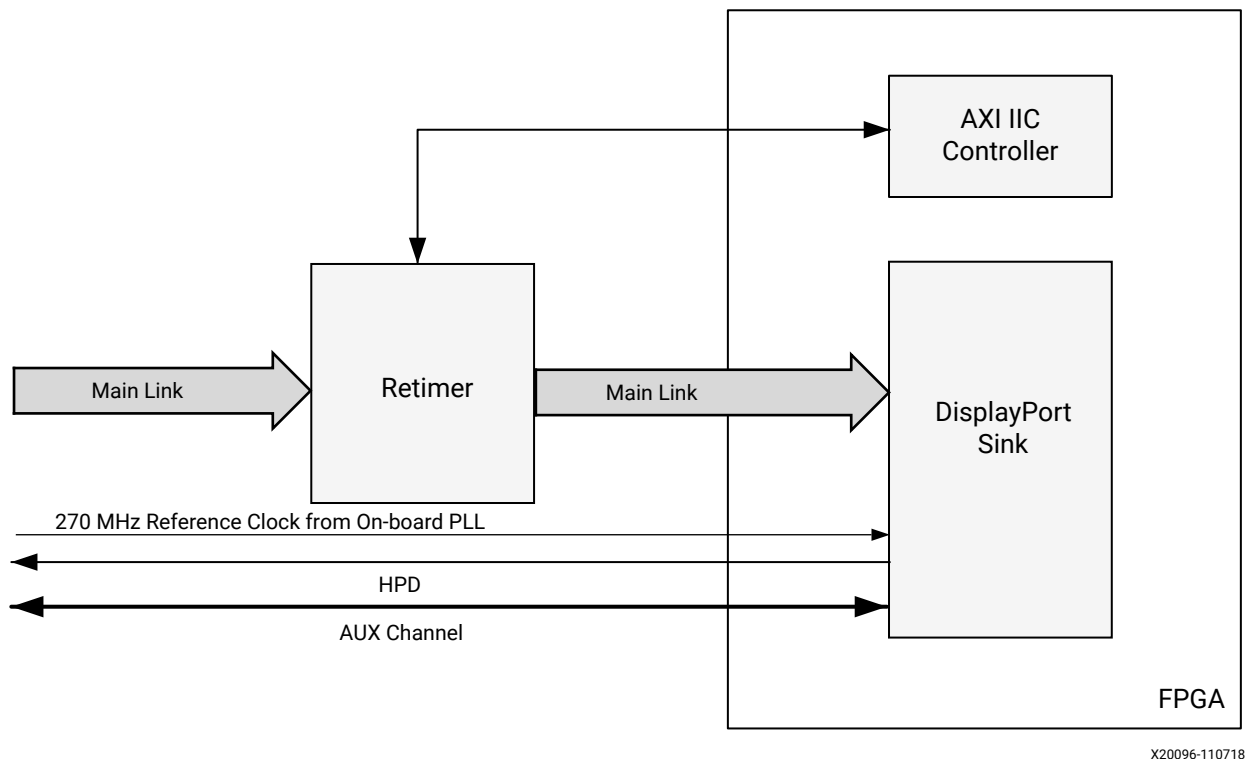
MegaChips Retimer

Xilinx® expects the use of the MCDP6000 Retimer along with the DisplayPort 1.4 RX Subsystem solution. MCDP6000 as a retimer provides better SI features. As a retimer, the MCDP6000 removes the random and ISI jitter from video source. The MCDP6000 configuration is controlled through an I2C interface. The DisplayPort 1.4 RX Subsystem design needs an external I2C controller to configure the Retimer.

Note: The IIC controller must work at 400 kHz.

For details on reference clock requirements and its connectivity, see the related information.

Figure 5: MCDP6000 Retimer



X20096-110718

Because the DisplayPort software driver handles the required MCDP6000 configuration, you do not have to control the MCDP6000 retimer. If the MCDP6000 is managed by a control other than the DisplayPort software driver, contact MegaChips (mca_support@megachips.com) for detailed programming information. The product page is <http://www.megachips.com/products/displayport/MCDP60x0>.

Related Information

[Clocking](#)

Link Training

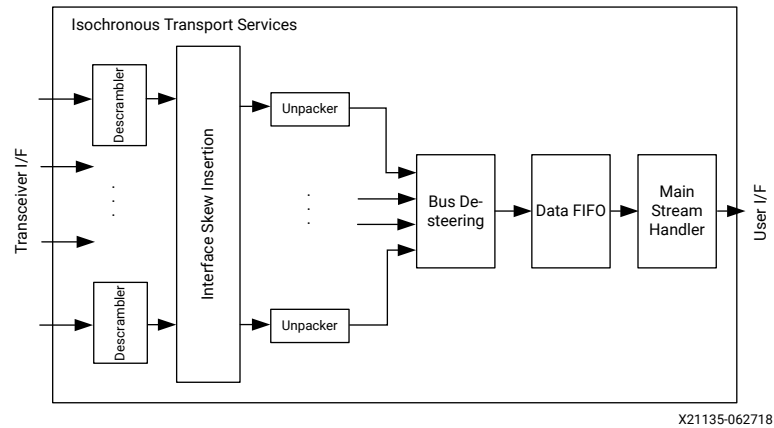
The link training commands are passed from the DPCD register block to the link training function. When set into the link training mode, the functional datapath is blocked, and the link training controller monitors the PHY and detects the specified pattern. Care must be taken to place the Sink core into the proper link training mode before the source begins sending the training pattern. Otherwise, unpredictable results might occur.

The link training process is specified in the *VESA DisplayPort Standard v1.4* ([VESA website](http://www.vesa.org/displayport/vesa-displayport-standard-v1-4)).

The Main Link for the Sink core drives a stream of video data for you. Using horizontal and vertical sync signals for framing, this user interface matches the industry standard for display controllers and plugs in to existing video streams with little effort. Though the core provides data and control signaling, you are still expected to supply an appropriate clock. This clock can be generated with the use of M and N values provided by the core. Alternatively, you might want to generate a clock by other means. The core underflow protection allows you to use a fast clock to transfer data into a frame buffer.

You can specify one, two, or four pixel-wide data through a register field. The bit width and format is determined from the Main Stream Attributes, which are provided as register fields.

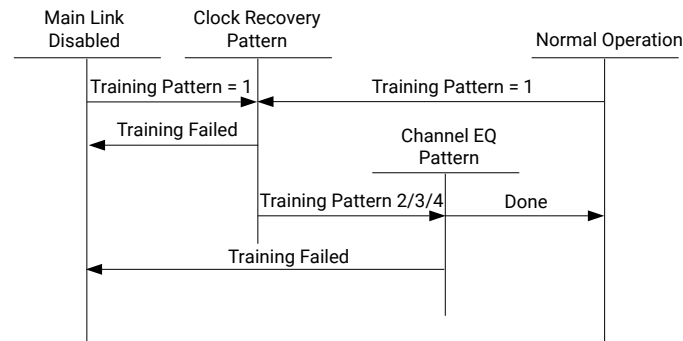
Figure 6: Sink Main Link Datapath



X21135-062718

The following figure shows the flow diagram for link training. For details, see the VESA DisplayPort Standard v1.4 ([VESA website](http://www.vesa.org/displayport/)).

Figure 7: Link Training States



X20174-062518

Receiver Clock Generation

This section describes the frame buffer and non-frame buffer designs.

Frame Buffer

With a frame buffer, you can generate a clock that is equal to or faster than the video clock to clock the user interface into a frame buffer.

This is the Xilinx® implemented solution as it does not require any external clock management.

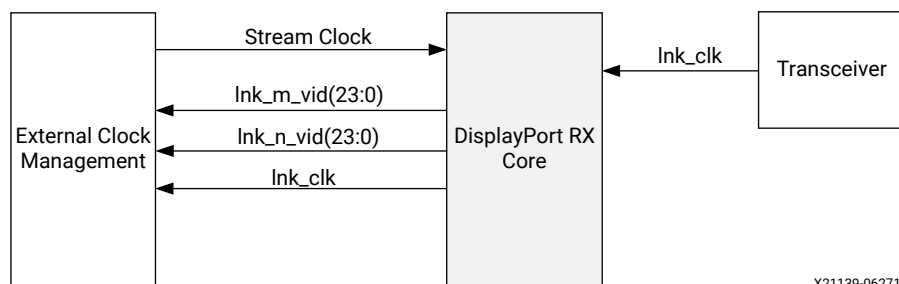
Non-Frame Buffer

For non-frame buffer designs, the DisplayPort RX core requires the generation of a video stream using the M and N values within the Main Stream Attributes to reconstruct an accurate stream clock. The DisplayPort RX core places this information on dedicated signals and provides an update flag to signal a change in these values. The following figure shows how to use the M and N values from the core to generate a clock. For more details, see the *VESA DisplayPort Standard v1.4* ([VESA website](http://www.vesa.org/displayport/DisplayPortStandard14)).



RECOMMENDED: The Xilinx® MMCM is not accurate enough to be used to regenerate the necessary clock for non-frame buffer design. You need to use an external PLL that meets the requirements of the DisplayPort standard. For more details, see the *VESA DisplayPort Standard v1.4* ([VESA website](http://www.vesa.org/displayport/DisplayPortStandard14)).

Figure 8: Receiver Clock Generation



X21139-062718

Common Event Detection

In certain applications, the detection of some events might be required. This section describes how to detect these events.

Transition from Video to No Video

In the course of operation, the Source core might stop sending video as detected by the NO_VIDEO interrupt. During this time, you should not rely on any MSA values.

Transition from No Video to Video

The transmission of video after a NO_VIDEO interrupt can be detected by the VERTICAL_BLANKING interrupt. Upon the reception of a VERTICAL_BLANKING interrupt, if disabled, you might then re-enable the display timing generator.

Mode Change

A mode change can be detected by the MODE_CHANGE interrupt. You must either read the new MSA values from register space or use the dedicated ports provided on the Main Link to properly frame the video data.

Cable is Unplugged, Lost Training

When a cable becomes unplugged or training is lost for any other reason, the TRAINING_LOST interrupt occurs. At that point, video data and MSA values should not be relied on.

After the cable becomes plugged in again, no action is required from you; the core properly resets itself and applies HPD. In a scenario, where the cable is plugged-in but the training is lost, the software is expected to assert a HPD upon the occurrence of a TRAINING_LOST interrupt, so that the Source can retrain the link.

Link is Trained

You can determine that the core is properly training by reading from the LANE_STATUS register and observing lane alignment and symbol lock on all active lanes. Additionally, it is advisable to ensure that the PLL is locked (per the Video PHY Controller) and reset is complete, which is also part of the PHY_STATUS register.

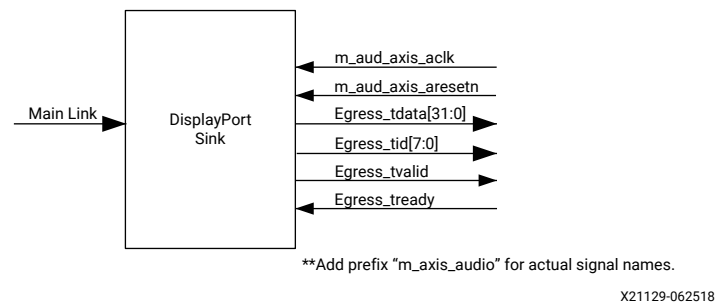
Secondary Channel Operation

The current version of the DisplayPort core supports eight-channel Audio. The DisplayPort Audio IP core is offered as modules to provide flexibility to modify the system as needed.

As shown in the following figure, the Audio interface to the DisplayPort core is defined using the AXI4-Stream interface.

Audio data and secondary packets are received from the main link and stored in internal buffers of the DisplayPort Sink core. The AXI4-Stream interface of the DisplayPort core transfers audio samples along with control bits. The DisplayPort Sink should never be back pressured.

Figure 9: Audio Data Interface of DisplayPort Sink System



Multi-Channel Audio

The DisplayPort RX captures the audio data received over the link and sends it over AXI4-Stream interface, along with the channel ID (`TID[3:0]`) based on the number of channels and the speaker allocation. The Stream ID received over the Info frame is also sent over the `TID[7:4]`. Samples for unallocated channels are dropped in the DisplayPort RX.

Audio Management

This section contains the procedural tasks required to achieve audio communication.

Programming DisplayPort Sink

1. Disable Audio by writing `0x00` to the `RX_AUDIO_CONTROL` register. The disable bit also flushes the buffers in the DisplayPort Sink. When there is a change in video/audio parameters, Xilinx® recommends following this step.
2. Enable audio by writing `0x01` to the `RX_AUDIO_CONTROL` register.
3. To read the Info packet, poll the `RX_AUDIO_STATUS[0]` register, and when asserted, read all eight words.
4. MAUD and NAUD are available as output ports and also in registers. Use these values per the design clocking structure. For example, in software, a poll routine can be used to detect a change and trigger a PLL-M and N value programming.

Re-Programming Sink Audio

1. Look for MUTE status by polling VB-ID.
2. When MUTE bit is set, Disable Audio in DisplayPort Receiver.
3. Wait for some time (in μ s) or wait until MUTE bit is removed.
4. Enable Audio in DisplayPort Receiver.
5. Wait for some time (in μ s).

Reading Info/Ext Packet

These packets can be read using poll mode or interrupt mode.

Poll Mode

1. Read `RX_AUDIO_STATUS` register until Info/Ext packet bit is set.
2. Based on Info/Ext bit setting, read respective buffers immediately. New packets get dropped if buffer is not read.
3. The status bit automatically gets cleared after reading packet.

Interrupt Mode

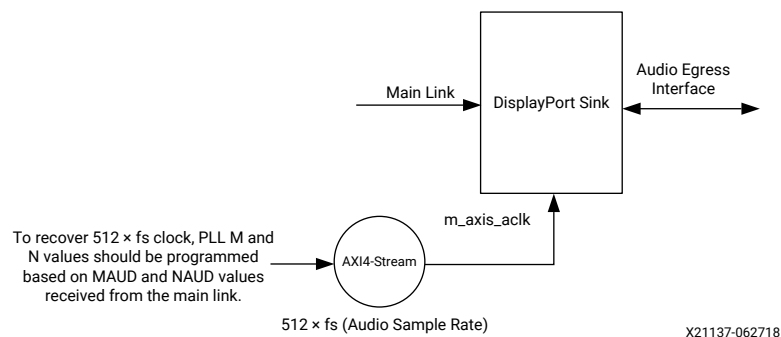
1. Ensure EXT_PKT_RXD/INFO_PKT_RXD interrupt is enabled by setting proper mask.
2. Wait for interrupt, Read interrupt cause register to check if EXT_PKT_RXD or INFO_PKT_RXD is set.
3. Based on interrupt status, read packet from appropriate buffer immediately.

Audio Clocking (Recommendation)

DisplayPort Sink device receives MAUD and NAUD values from the upstream source device. These values are accessible to the system through the output ports and registers.

The system should have a clock generator (preferably programmable) to generate $512 \times fs$ (Audio Sample Rate) clock frequency based on MAUD and NAUD values. External clock source is preferred for better precision.

Figure 10: Audio Clocking for Sink



Sampling Frequencies

The DisplayPort 1.4 RX Subsystem with a GT data width of 16-bit mode supports up to eight channels of audio with maximum supported sampling frequency of 192 KHz for all link rates.

Programming the Core in MST Mode

This section includes details about programming the Sink core in MST mode.

Enabling MST

The following steps are recommended to enable MST functionality:

To enable MST functionality, perform link bring-up and enable MST capability in MST Capability register (0x0D0). The Source device enables the MST after payload allocation and ACT event process is done.

MST AUX Messaging

1. Wait for the DOWN_REQUEST_BUFFER_READY status interrupt, and read from DOWN_REQUEST_BUFFER. Continue to collect side-band messages as per VESA DisplayPort Standard v1.4 ([VESA website](#)). After a complete sideband message is received, the software processes the message and writes the reply to DOWN_REPLY_BUFFER.
2. After the response is written, set DOWN Reply Buffer Message to 1 in the Remote Command New register.
3. Wait for DOWN_REPLY_BUFFER_READ status in interrupts and continue writing responses.

During the MST AUX messaging phase, the required PBN (available BW) is calculated and sent to the source. The source then sends allocation requests based on available bandwidth. Internally, the Sink hardware updates the VC payload table by monitoring the AUX transactions. Alternatively, if you are an advanced user, you can use software to control the VC payload table (setting bit 1 in 0x0D0 enables it); the software maintains a VC payload manager (by monitoring the interrupt bit 28 of 0x014 register and 0x06C register) and writes the resulting stream allocations to 0x800-0x8FF. When the software finishes writing to the VC payload table, the software has to set bit 4 in 0x0D0.

For an interrupt event, read both Interrupt Cause and Interrupt Cause 1 registers.



IMPORTANT! The software is required to form appropriate LINK_ADDRESS sideband reply as per the Message Transaction protocol given in Section 2.11.2 of the VESA DisplayPort Standard v1.4 ([VESA website](#)). The LINK_ADDRESS reply helps the source to identify the topology of the sink. For example, if the sink core is configured for four MST streams, it receives the multi-stream input from the DisplayPort RX and outputs four individual streams in native video format. In this case, the LINK_ADDRESS_REPLY can be modeled to contain one input and four output logical ports, with their DisplayPort device plug status set as 1 and Peer Device Type set as 3.

Reduced Blanking

The DisplayPort IP core supports CVT standard RB and RB2 reduced blanking resolutions. As per the CVT specifications, RB/RB2 resolution has $HBLANK \leq 20\% HTOTAL$, $HBLANK = 80/160$ and $HRES \% 8 = 0$.

For the CVT standard, RB/RB2 resolutions end of the line reset need to be disabled by setting the corresponding bit in the Line Reset Disable register (0x008 for the RX). For the Non-CVT reduced blanking resolutions, where HRES is non multiple of 8, end of line reset is required to clear extra pixels in the video path for each line.

The DisplayPort TX knows the resolution ahead of time and reset disable can be done during initialization. In the DisplayPort RX when the video mode change interrupt occurs, the MSA registers can be read to know whether the resolution is reduced blanking or standard resolution and the corresponding bit can be set.

Pixel Mapping

Pixel Mapping on AXI4-Stream Interface

By default, the pixel mode is equal to the lane count during subsystem generation. You can override the pixel width dynamically. For example, if the driver selects a 2 pixel mode as default, you can change the pixel mode to 1.

- For a pixel mode of 1, valid pixels are available only in pixel 0 position.
- For a pixel mode of 2, valid pixels are available only in pixel 0 and pixel 1 position.
- For a pixel mode of 4, valid pixels are available only in pixel 0, pixel 1, pixel 2, and pixel 3 position.

The data width of the AXI4-Stream interface depends on different parameters of the core.

$$\text{Pixel_Width} = \text{MAX_BPC} \times 3$$

$$\text{Interface Width} = \text{Pixel Width} \times \text{LANE_COUNT}$$

For example, if the system is generated using four lanes with MAX_BPC equal to 16, the data width of the AXI4-Stream interface is $16 \times 4 \times 3$ which equals 192.

Pixel Mapping Examples on AXI4-Stream Interface

The following table shows the pixel mapping examples for an AXI4-Stream interface.

Table 28: Pixel Mapping Examples on AXI4-Stream Interface

MAX_BPC	LANES	Pixel Width	Interface Width	Video BPC	Pixel 3			Pixel 2			Pixel 1			Pixel 0		
16	4	48	192	16	191:176	175:160	159:144	143:128	127:112	111:96	95:80	79:64	63:48	47:32	31:16	15:0
16	2	48	96	16	-	-	-	-	-	-	95:80	79:64	63:48	47:32	31:16	15:0
16	1	48	48	16	-	-	-	-	-	-	-	-	-	47:32	31:16	15:0
12	4	36	144	12	143:132	131:120	119:108	107:96	95:84	83:72	71:60	59:48	47:36	35:24	23:12	11:0
12	2	36	72	12	-	-	-	-	-	-	71:60	59:48	47:36	35:24	23:12	11:0
12	1	36	40	12	-	-	-	-	-	-	-	-	-	35:24	23:12	11:0
10	4	30	120	10	119:110	109:100	99:90	89:80	79:70	69:60	59:50	49:40	39:30	29:20	19:10	9:0
10	2	30	64	10	-	-	-	-	-	-	59:50	49:40	39:30	29:20	19:10	9:0
10	1	30	32	10	-	-	-	-	-	-	-	-	-	29:20	19:10	9:0

Table 28: Pixel Mapping Examples on AXI4-Stream Interface (cont'd)

MAX BPC	LANES	Pixel Width	Interface Width	Video BPC	Pixel 3			Pixel 2			Pixel 1			Pixel 0		
8	4	24	96	8	95:88	87:80	79:72	71:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
8	2	24	48	8	-	-	-	-	-	-	47:40	39:32	31:24	23:16	15:8	7:0
8	1	24	24	8	-	-	-	-	-	-	-	-	-	23:16	15:8	7:0
16	4	48	192	12	191:180	175:164	159:148	143:132	127:116	111:100	95:84	79:68	63:52	47:36	31:20	15:4
16	2	48	96	12	-	-	-	-	-	-	95:84	79:68	63:52	47:36	31:20	15:4
16	1	48	48	12	-	-	-	-	-	-	-	-	-	47:36	31:20	15:4
12	4	36	144	10	143:134	131:122	119:110	107:98	95:86	83:74	71:62	59:50	47:38	35:26	23:14	11:2
12	2	36	72	10	-	-	-	-	-	-	71:62	59:50	47:38	35:26	23:14	11:2
12	1	36	40	10	-	-	-	-	-	-	-	-	-	35:26	23:14	11:2
10	4	30	120	8	119:112	109:102	99:92	89:82	79:72	69:62	59:52	49:42	39:32	29:22	19:12	9:2
10	2	30	64	8	-	-	-	-	-	-	59:52	49:42	39:32	29:22	19:12	9:2
10	1	30	32	8	-	-	-	-	-	-	-	-	-	29:22	19:12	9:2
8	4	24	96	6	95:90	87:82	79:74	71:66	63:58	55:50	47:42	39:34	31:26	23:18	15:10	7:2
8	2	24	48	6	-	-	-	-	-	-	47:42	39:34	31:26	23:18	15:10	7:2
8	1	24	24	6	-	-	-	-	-	-	-	-	-	23:18	15:10	7:2
16	4	48	192	10	191:182	175:166	159:150	143:134	127:118	111:102	95:86	79:70	63:54	47:38	31:22	15:6
16	2	48	96	10	-	-	-	-	-	-	95:86	79:70	63:54	47:38	31:22	15:6
16	1	48	48	10	-	-	-	-	-	-	-	-	-	47:38	31:22	15:6
12	4	36	144	8	143:136	131:124	119:112	107:100	95:88	83:76	71:64	59:52	47:40	35:28	23:16	11:4
12	2	36	72	8	-	-	-	-	-	-	71:64	59:52	47:40	35:28	23:16	11:4
12	1	36	40	8	-	-	-	-	-	-	-	-	-	35:28	23:16	11:4
10	4	30	120	6	119:114	109:104	99:94	89:84	79:74	69:64	59:54	49:44	39:34	29:24	19:14	9:4
10	2	30	64	6	-	-	-	-	-	-	59:54	49:44	39:34	29:24	19:14	9:4
10	1	30	32	6	-	-	-	-	-	-	-	-	-	29:24	19:14	9:4
16	4	48	192	8	191:184	175:168	159:152	143:136	127:120	111:104	95:88	79:72	63:56	47:40	31:24	15:8
16	2	48	96	8	-	-	-	-	-	-	95:88	79:72	63:56	47:40	31:24	15:8
16	1	48	48	8	-	-	-	-	-	-	-	-	-	47:40	31:24	15:8
12	4	36	144	6	143:138	131:126	119:114	107:102	95:90	83:78	71:66	59:54	47:42	35:30	23:18	11:6
12	2	36	72	6	-	-	-	-	-	-	71:66	59:54	47:42	35:30	23:18	11:6
12	1	36	36	6	-	-	-	-	-	-	-	-	-	35:30	23:18	11:6
16	4	48	192	6	191:186	175:170	159:154	143:138	127:122	111:106	95:90	79:74	63:58	47:42	31:26	15:10
16	2	48	96	6	-	-	-	-	-	-	95:90	79:74	63:58	47:42	31:26	15:10
16	1	48	48	6	-	-	-	-	-	-	-	-	-	47:42	31:26	15:10

Notes:

1. The padding bits are zeros.

Pixel Mapping Examples on AXI4-Stream Interface (UG934-Compliant)

The following table shows the pixel mapping examples for an AXI4-Stream interface that is UG934-compliant.

Table 29: Pixel Mapping Examples on AXI4-Stream Interface (UG934-Compliant Mode)

MAX_BPC	LANES	Pixel Width	Interface Width	Video Sampling Mode	Pixel 3			Pixel 2			Pixel 1			Pixel 0		
8	4	24	96	444	95:88	87:80	79:72	71:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
	4	16	64	422	63:56	55:48		47:40	39:32		31:24	23:16		15:8	7:0	
	4	8	32	Y-Only	31:24			23:16			15:8			7:0		
	2	24	48	444	-	-	-	-	-	-	47:40	39:32	31:24	23:16	15:8	7:0
	2	16	32	422	-	-		-	-		31:24	23:16		15:8	7:0	
	2	8	16	Y-Only	-			-			15:8			7:0		
	1	24	24	444	-	-	-	-	-	-	-	-	-	23:16	15:8	7:0
	1	16	16	422	-	-		-	-		-	-		15:8	7:0	
	1	8	8	Y-Only	-			-			-			7:0		
10	4	30	120	444	119:112	109:102	99:92	89:82	79:72	69:62	59:52	49:42	39:32	29:22	19:12	9:2
	4	20	80	422	79:72	69:62		59:52	49:42		39:32	29:22		19:12	9:2	
	4	10	40	Y-Only	39:32			29:22			19:12			9:2		
	2	30	60	444	-	-	-	-	-	-	59:52	49:42	39:32	29:22	19:12	9:2
	2	20	40	422	-	-		-	-		39:32	29:22		19:12	9:2	
	2	10	20	Y-Only	-			-			19:12			9:2		
	1	30	30	444	-	-	-	-	-	-	-	-	-	29:22	19:12	9:2
	1	20	20	422	-	-		-	-		-	-		19:12	9:2	
	1	10	10	Y-Only	-			-			-			9:2		
12	4	36	144	444	143:136	131:124	119:112	107:100	95:88	83:76	71:64	59:52	47:40	35:28	23:16	11:4
	4	24	96	422	95:88	83:76		71:64	59:52		47:40	35:28		23:16	11:4	
	4	12	48	Y-Only	47:40			35:28			23:16			11:4		
	2	36	72	444	-	-	-	-	-	-	71:64	59:52	47:40	35:28	23:16	11:4
	2	24	48	422	-	-		-	-		47:40	35:28		23:16	11:4	
	2	12	24	Y-Only	-			-			23:16			11:4		
	1	36	36	444	-	-	-	-	-	-	-	-	-	35:28	23:16	11:4
	1	24	24	422	-	-		-	-		-	-		23:16	11:4	
	1	12	12	Y-Only	-			-			-			11:4		

Table 29: Pixel Mapping Examples on AXI4-Stream Interface (UG934-Compliant Mode)
(cont'd)

MAX_BPC	LANES		Pixel Width	Interface Width	Video Sampling Mode	Pixel 3			Pixel 2			Pixel 1			Pixel 0		
16	4	48	192	444	191:184	175:168	159:152	143:136	127:120	111:104	95:88	79:72	63:56	47:40	31:24	15:8	
	4	32	128	422	127:120	111:104		95:88	79:72		63:56	47:40		31:24	15:8		
	4	16	64	Y-Only	63:56			47:40			31:24			15:8			
	2	48	96	444	-	-	-	-	-	-	95:88	79:72	63:56	47:40	31:24	15:8	
	2	32	64	422	-	-		-	-		63:56	47:40		31:24	15:8		
	2	16	32	Y-Only	-			-			31:24			15:8			
	1	48	48	444	-	-	-	-	-	-	-	-	-	47:40	31:24	15:8	
	1	32	32	422	-	-		-	-		-	-		31:24	15:8		
	1	16	16	Y-Only	-			-			-			15:8			

Notes:

1. Video BPC is 8.

Pixel Mapping on Native Video Interface

The primary interface for user image data has been modeled on the industry standard for display timing controller signals. The port list consists of video timing information encoded in a vertical and horizontal sync pulse and data valid indicator. These single bit control lines frame the active data and provide flow control for the AXI4-Stream video.

Vertical timing is framed using the vertical sync pulse which indicates the end of frame N - 1 and the beginning of frame N. The vertical back porch is defined as the number of horizontal sync pulses between the end of the vertical sync pulse and the first line containing active pixel data. The vertical front porch is defined as the number of horizontal sync pulses between the last line of active pixel data and the start of the vertical sync pulse. When combined with the vertical back porch and the vertical sync pulse width, these parameters form what is commonly known as the vertical blanking interval.

At the trailing edge of each vertical sync pulse, the user data interface resets key elements of the image datapath. This provides for a robust user interface that recovers from any kind of interface error in one vertical interval or less.

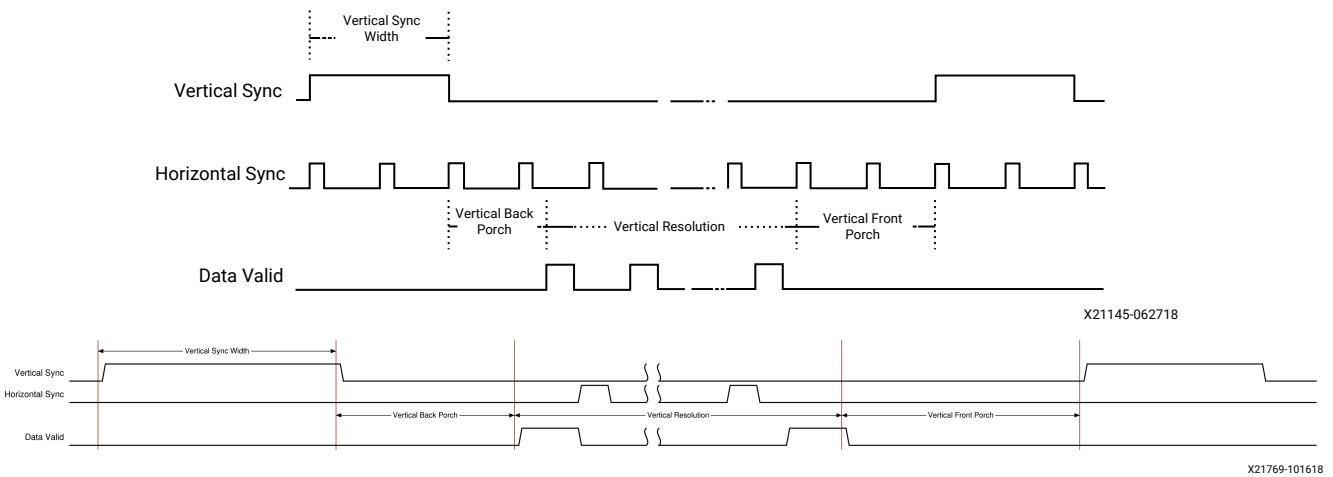
You have the option to use the resolved M and N values from the stream to generate a clock, or to use a sufficiently-fast clock and pipe the data into a line buffer.



RECOMMENDED: Xilinx® recommends using a fast clock and ignoring the M and N values unless you can be certain of the source of these values. Unlike the Source Core, when using a fast clock, the data valid signal might toggle within a scan line.

The following figure shows the typical signaling of a full frame of data.

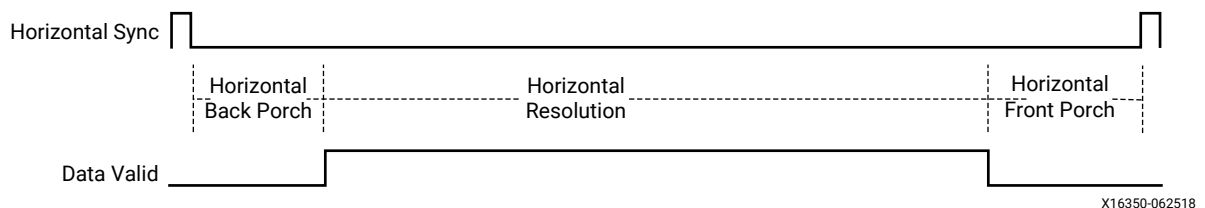
Figure 11: User Interface Vertical Timing



Similarly, the horizontal timing information is defined by a front porch, back porch, and pulse width. The porch values are defined as the number of clocks between the horizontal sync pulse and the start or end of active data. Pixel data is only accepted into the image data interface when the data valid flag is active-High. The following figure is an enlarged version of the previous figure, giving more details on a single scan line. The horizontal sync pulse should be used as a line advance signal. Use the rising edge of this signal to increment the line count.

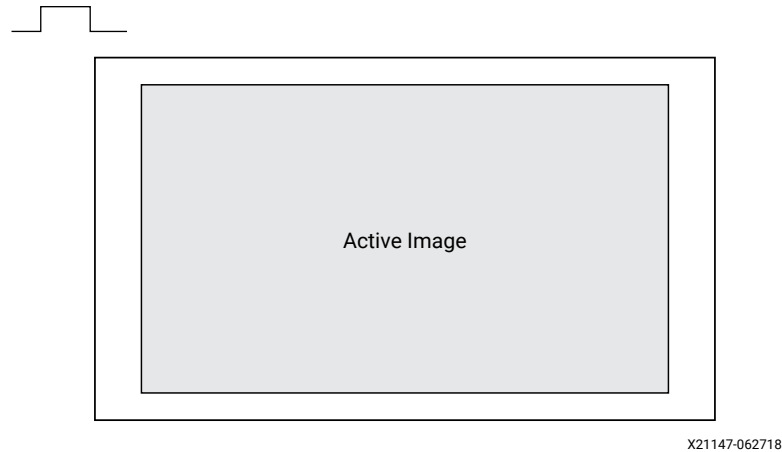
Note: Data Valid might toggle if using a fast clock. Also, Data Valid signal might remain asserted for the duration of a scan line. Dropping the valid signal might result in improper operation.

Figure 12: User Interface Horizontal Timing



In the two-dimensional image plane, these control signals frame a rectangular region of active pixel data within the total frame size. This relationship of the total frame size to the active frame size is shown in here.

Figure 13: Active Image Data



The User Data interface can accept one, two, or four pixels per clock cycle. The second pixel is active only when USER_PIXEL_WIDTH is set and the negotiated number of lanes is > 1. The `vid_pixel` width is always 48 bits, regardless if all bits are used. For pixel mappings that do not require all 48 bits, the convention used for this core is to occupy the MSB bits first and leave the lower bits either untied or driven to zero.

Pixel Mapping Examples on the User Data Interface

The following table shows the correct mapping for all supported data formats.

Table 30: Pixel Mapping Examples on the User Data Interface

Format	BPC/BPP	R	G	B	Cr	Y	Cb	Cr/Cb	Y
RGB	6/18	[47:42]	[31:26]	[15:10]	-	-	-	-	-
RGB	8/24	[47:40]	[31:24]	[15:8]	-	-	-	-	-
RGB	10/30	[47:38]	[31:22]	[15:6]	-	-	-	-	-
RGB	12/36	[47:36]	[31:20]	[15:4]	-	-	-	-	-
RGB	16/48	[47:32]	[31:16]	[15:0]	-	-	-	-	-
YCrCb444	6/18	-	-	-	[47:42]	[31:26]	[15:10]	-	-
YCrCb444	8/24	-	-	-	[47:40]	[31:24]	[15:8]	-	-
YCrCb444	10/30	-	-	-	[47:38]	[31:22]	[15:6]	-	-
YCrCb444	12/36	-	-	-	[47:36]	[31:20]	[15:4]	-	-
YCrCb444	16/48	-	-	-	[47:32]	[31:16]	[15:0]	-	-
YCrCb422	8/16	-	-	-	-	-	-	[47:40]	[31:24]
YCrCb422	10/20	-	-	-	-	-	-	[47:38]	[31:22]

Table 30: Pixel Mapping Examples on the User Data Interface (cont'd)

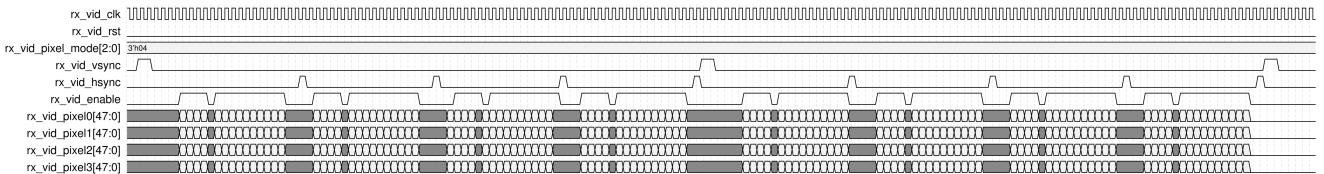
Format	BPC/BPP	R	G	B	Cr	Y	Cb	Cr/Cb	Y
YCrCb422	12/24	-	-	-	-	-	-	[47:36]	[31:20]
YCrCb422	16/32	-	-	-	-	-	-	[47:32]	[31:16]
YONLY	8/8	-	-	-	-	-	-	-	[47:40]
YONLY	10/10	-	-	-	-	-	-	-	[47:38]
YONLY	12/12	-	-	-	-	-	-	-	[47:36]
YONLY	16/16	-	-	-	-	-	-	-	[47:32]

Notes:

- For a YCrCb 4:2:2, the input follows YCr, YCb, YCr, YCb, and so on. This means Cr and Cb are mapped to the same bits on the video output ports of the Sink core.

The design allows use of a faster pixel clock. For example, 150 MHz or higher video clock frequency for all standard video resolutions. DisplayPort 1.4 RX Subsystem supports DMA mode without any internal line buffers for video display. You need to reproduce the exact video timing from the M_VID and N_VID values reported over MSA. The interface timing in this case is shown in the following figure.

Figure 14: RX Pixel Timing



Note: The width of `rx_vid_vsync`, `rx_vid_hsync`, `rx_vid_enable`, and the number of hsync pulses shown in the previous figure above are scaled down to have better visibility. The number of hsync pulses are equal to the number of active lines in a frame. The default widths of `rx_vid_hsync` pulse is 16 and `rx_vid_vsync` pulse is 63. The widths hsync and vsync can be controlled through software as per MSA.

AXI4-Stream Interface Color Mapping

This table shows the color mapping for the AXI4-Stream interface and this complies with the guidelines in the *AXI4-Stream Video IP and System Design Guide* ([UG934](#)).

Table 31: AXI4-Stream Interface Data Mapping

	AXI4-Stream Interface											
	Pixel 3			Pixel 2			Pixel 1			Pixel 0		
	Comp3	Comp2	Comp1	Comp3	Comp2	Comp1	Comp3	Comp2	Comp1	Comp3	Comp2	Comp1
RGB	R	B	G	R	B	G	R	B	G	R	B	G
YCbCr444	Cr	Cb	Y	Cr	Cb	Y	Cr	Cb	Y	Cr	Cb	Y
YCbCr422	-	Cr/Cb	Y	-	Cr/Cb	Y	-	Cr/Cb	Y	-	Cr/Cb	Y
Y-Only	-	-	Y	-	-	Y	-	-	Y	-	-	Y

Notes:

- For component widths, see the Pixel Mapping Examples on AXI4-Stream Interface.

Related Information

[Pixel Mapping Examples on AXI4-Stream Interface](#)

Clocking

This section describes the link clock (`rx_lnk_clk`), video clock (`rx_vid_clk`), and Video Bridge to AXI4-Stream master interface clock. The `rx_vid_clk` should be 270 MHz or higher and the `m_axis_aclk_streamn` can be equal or greater than the `rx_vid_clk`.

The `rx_lnk_clk` is a link clock input to the DisplayPort 1.4 RX Subsystem generated by the Video PHY (GT).

The `hdcp_ext_clk` input has to be driven from external MMCM where it has a frequency requirement of $\text{hdcp_ext_clk} = \text{rx_lnk_clk}/2$ MHz.

The following table shows the clock ranges.

Table 32: Clock Ranges

Clock Domain	Min (MHz)	Max (MHz)	Description
<code>rx_lnk_clk</code>	81	405	Link clock
<code>rx_vid_clk</code>	150	270	Video clock

Table 32: Clock Ranges (cont'd)

Clock Domain	Min (MHz)	Max (MHz)	Description
s_axi_aclk	25	135	Host processor clock

The core uses six clock domains:

- **lnk_clk:** The `rxoutclk` from the Video PHY is connected to the RX subsystem link clock. Most of the core operates in link clock domain. This domain is based on the `lnk_clk_p/n` reference clock for the transceivers. The link rate switching is handled by a DRP state machine in the core PHY later. When the lanes are running at 2.7 Gb/s, `lnk_clk` operates at 135 MHz. When the lanes are running at 1.62 Gb/s, `lnk_clk` operates at 81 MHz. When the lanes are running at 5.4 Gb/s, `lnk_clk` operates at 270 MHz. When the lanes are running at 8.1 Gb/s, `lnk_clk` operates at 405 MHz.

In the DisplayPort Sink core, `lnk_clk` is derived from the recovered clock from the transceiver. When the cable is disconnected this clock becomes unstable.

Note: `lnk_clk = link_rate/20`, when GT-Data width is 16-bit.

- **vid_clk:** This is the primary user interface clock. It has been tested to run as fast as 150 MHz, which accommodates to a screen resolution of 2560×1600 when using two-wide pixels and larger when using the four-wide pixels. Based on the DisplayPort Standard, the video clock can be derived from the link clock using `mvid` and `nvid`.
- **s_axi_aclk:** This is the processor domain. It has been tested to run as fast as 135 MHz. The AUX clock domain is derived from this domain, but requires no additional constraints. In UltraScale™ FPGA, `s_axi_aclk` clock is connected to a free-running clock input. `gtwiz_reset_clk_freerun_in` is required by the reset controller helper block to reset the transceiver primitives. A new GUI parameter is added for AXI_Frequency, when the DisplayPort IP is targeted to UltraScale FPGA. The requirement is `s_axi_aclk ≤ lnk_clk`.
- **aud_clk:** This is the audio interface clock. The frequency will be equal to $512 \times$ audio sample rate.
- **s_aud_axis_aclk:** This clock is used by the source audio streaming interface. This clock should be $= 512 \times$ audio sample rate.
- **m_aud_axis_aclk:** This clock is used by the sink audio streaming interface. This clock should be $= 512 \times$ audio sample rate.

For more information on clocking, see the *Video PHY Controller LogiCORE IP Product Guide* (PG230).

Resets

The subsystem has one reset input for each of the AXI4-Lite, AXI4-Stream, and Video interfaces:

- **s_axi_aresetn**: Active-Low AXI4-Lite reset. This resets all the programming registers.
- **rx_vid_rst**: Active-High video pipe reset. MST with four streams, there are four video resets.
- **mcdp6000_rst**: Active-High soft reset to the MCDP6000 retimer generated through AXI IIC GPIO port. This reset is asserted through AXI IIC register programming for GPIO ports. For more details, see the *AXI IIC Bus Interface LogiCORE IP Product Guide* ([PG090](#)).

Address Map Example

The following table shows an example based on a subsystem base address of 0x44C0_0000 (14 bits). There are no registers in the Video to AXI4-Stream bridge.

Table 33: Address Map Example

	SST	MST
DisplayPort 1.4 RX	0x44C0_0000	0x44C0_0000
AXI IIC Controller	0x44C0_1000	0x44C0_1000
HDCP Controller	0x44C0_2000	0x44C0_2000
AXI Timer	0x44C0_3000	0x44C0_3000

Programming Sequence

For PHY related programming, see the *Video PHY Controller LogiCORE IP Product Guide* ([PG230](#)).

Design Flow Steps

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

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* ([UG994](#))
- *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 Subsystem

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

If you are customizing and generating the subsystem in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* ([UG994](#)) 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 subsystem 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.

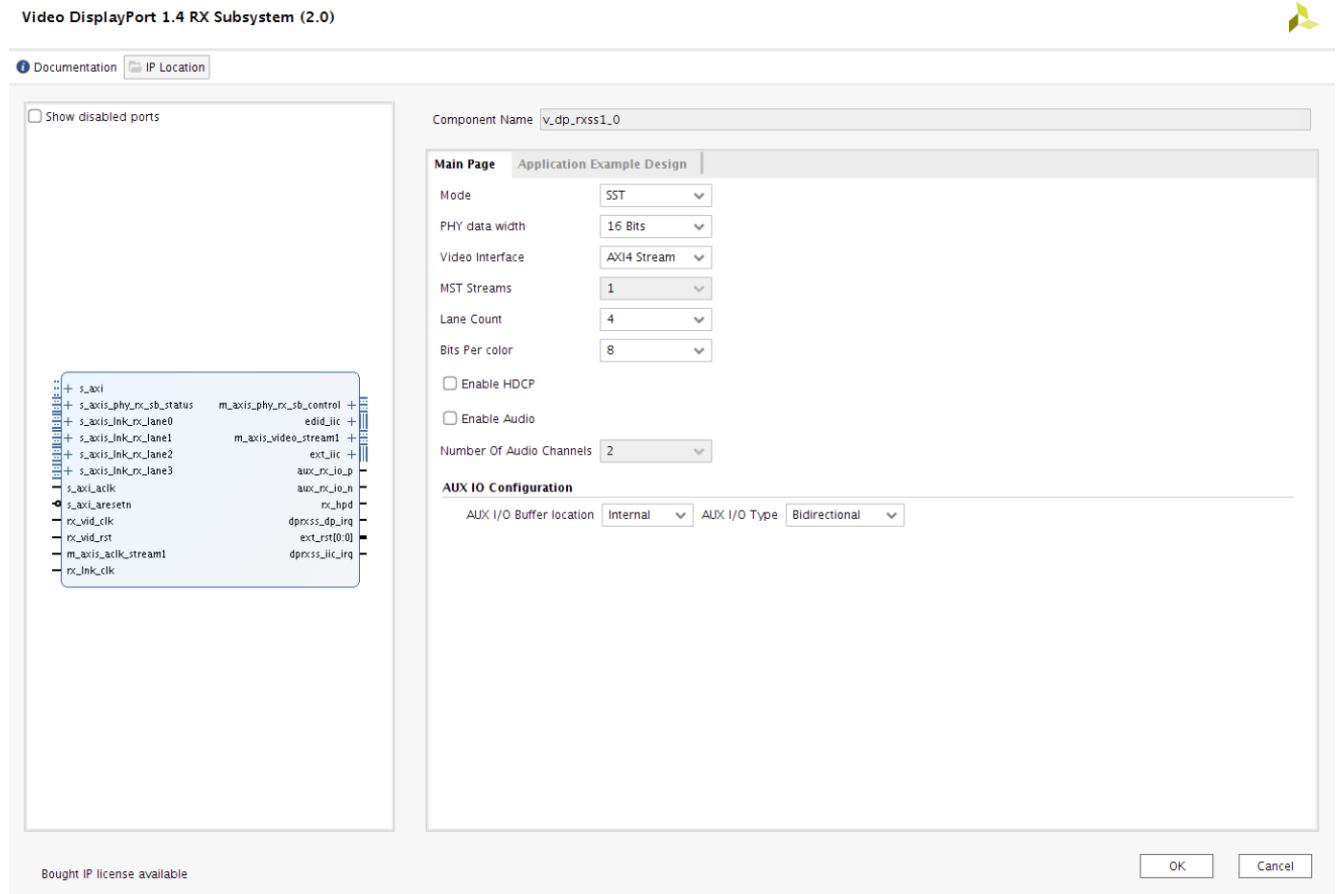
For details, see the *Vivado Design Suite User Guide: Designing with IP* ([UG896](#)) and the *Vivado Design Suite User Guide: Getting Started* ([UG910](#)).

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

Subsystem Configuration Screen

The subsystem configuration screen is shown in the following figure.

Figure 15: Configuration Screen



- **Component Name:** The Component Name is used as the name of the top-level wrapper file for the core. The underlying netlist still retains its original name. Names must begin with a letter and must be composed from the following characters: a through z, 0 through 9, and "_". The name displayport_0 is used as internal module name and should not be used for the component name. The default is v_dp_rxss1_0.
- **Mode:** Selects the desired resolution for the DisplayPort IP. Options are SST or MST.
- **PHY Data Width:** Selects the 16-bit GT data width.
- **Video Interface:** Selects the AXI4-Stream or native for the input video interface.
- **MST Streams:** Selects the maximum number of streams in MST mode.

- **Lane Count:** Selects the maximum number of lanes.
- **Bits Per Color:** Selects the desired maximum bit per component (BPC).
- **Enable HDCP:** Enables the HDCP encryption.
- **Enable Audio:** Enables the audio support.
- **Audio Channels:** Selects the number of audio channels.
- **AUX I/O Buffer location:** Selects the buffer location for AUX channel.
- **AUX I/O Type:** Selects the Bidirectional or Unidirectional buffer type.

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 34: User Parameters

Vivado IDE Parameter/Value	User Parameter/Value	Default Value
Mode	MODE	SST
PHY Data Width	PHY_DATA_WIDTH	16
Video Interface	VIDEO_INTERFACE	AXI4-Stream
MST Streams	NUM_STREAMS	1
Lane Count	LANE_COUNT	4
Bits Per Color	BITS_PER_COLOR	8
Enable HDCP	HDCP_ENABLE	0
Enable Audio	AUDIO_ENABLE	0
Number Of Audio Channels	AUDIO_CHANNELS	2
Pixel Mode	PIXEL_MODE	1/2/4 (Valid only in native mode)
AUX I/O Buffer Location	AUX_IO_LOC	Internal
AUX I/O Type	AUX_IO_TYPE	Bidirectional

Output Generation

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

Constraining the Subsystem

Required Constraints

This section is not applicable for this IP subsystem.

Device, Package, and Speed Grade Selections

See IP Facts in the related information below for details about supported devices.

This section is not applicable for this IP subsystem.

Clock Frequencies

See Clocking for more details about clock frequencies.

Clock Management

This section is not applicable for this IP subsystem.

Clock Placement

This section is not applicable for this IP subsystem.

Banking

For more information on the specific banking constraints, see the *Video PHY Controller LogiCORE IP Product Guide* ([PG230](#)).

Transceiver Placement

Transceiver is external to DisplayPort 1.4 RX Subsystem *Video PHY Controller LogiCORE IP Product Guide* ([PG230](#)).

I/O Standard and Placement

This section is not applicable for this IP subsystem.

AUX Channel

The *VESA DisplayPort Standard v1.4* ([VESA website](#)) describes the AUX channel as a bidirectional LVDS signal. For , hence there are no specific transceiver placement constraints. For more information on the specific transceiver placement constraints, see the 7 series designs, the core uses IOBUFDS (bidirectional buffer) as the default with the LVDS standard. You should design the board as recommended by the VESA DP Protocol Standard. For reference, see the example design XDC file.

For UltraScale+™ and UltraScale™ families supporting HR IO banks, use the following Sink constraints:

```
set_property IOSTANDARD LVDS_25 [get_ports aux_rx_io_p]
set_property IOSTANDARD LVDS_25 [get_ports aux_rx_io_n]
```

For UltraScale+ and UltraScale families supporting HP IO banks, use the following Sink constraints:

```
set_property IOSTANDARD LVDS [get_ports aux_rx_io_p]
set_property IOSTANDARD LVDS [get_ports aux_rx_io_n]
```

HPD

The HPD signal can operate in either a 3.3V or 2.5V I/O bank. By definition in the standard, it is a 3.3V signal.

For UltraScale+ and UltraScale families supporting HR IO banks, use the following constraints:

```
set_property IOSTANDARD LVCMOS25 [get_ports hpd];
```

For UltraScale+ and UltraScale families supporting HP IO banks, use the following constraints:

```
set_property IOSTANDARD LVCMOS18 [get_ports hpd];
```

Board design and connectivity should follow DisplayPort standard recommendations with proper level shifting.

Related Information

[IP Facts](#)

[Clocking](#)

Simulation

There is no example design simulation support for DisplayPort 1.4 RX Subsystem.

Synthesis and Implementation

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

Example Design

Note: All example designs use the Inrevium DP1.4 FMC card.

This chapter contains step-by-step instructions for generating an Application Example Design from the DisplayPort 1.4 RX Subsystem by using the Vivado® Design Suite flow.



RECOMMENDED: For ZCU102/VCU118 (Revision 1.0 or later), you should set up a 1.8V setting after connecting the DisplayPort FMC. For details on the FMC voltage, see the [Related Information](#) link. For more information, see the [ZCU102 System Controller GUI Tutorial \(XTP433\)](#) (registration required) and the [VCU118 System Controller Tutorial \(XTP447\)](#) (registration required).

Related Information

[Setting the FMC Voltage to 1.8V](#)

Available Example Designs

The following table shows the example designs available for the TX and RX DisplayPort 1.4 subsystems.

Table 35: Available Example Designs

GT Type	Topology	Video PHY Config		Hardware	BPC	Processor
		(TXPLL)	(RXPLL)			
GTHE3	Pass-through without HDCP1.3 (1)	QPLL	CPLL	KCU105 + Inrevium DP1.4 FMC	8	MicroBlaze™
GTHE4	RX only	-	CPLL	ZCU102 + Inrevium DP1.4 FMC	10	R5
	TX only	QPLL	-	ZCU102 + Inrevium DP1.4 FMC	10	R5
	FB Pass-through without HDCP1.3	QPLL	CPLL	ZCU102 + Inrevium DP1.4 FMC	10	A53
	FB Pass-through with HDCP1.3	QPLL	CPLL	ZCU102 + Inrevium DP1.4 FMC	10	A53

Table 35: Available Example Designs (cont'd)

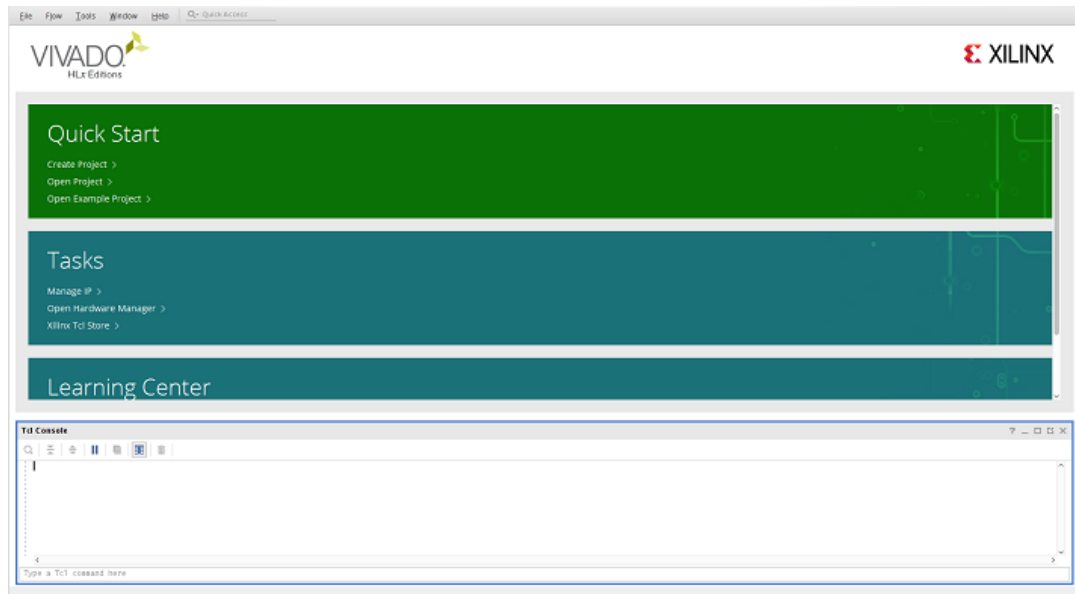
GT Type	Topology	Video PHY Config		Hardware	BPC	Processor
		(TXPLL)	(RXPLL)			
GTYE4	RX only	-	CPLL	VCU118 + Inrevium DP1.4 FMC	10	MicroBlaze
	TX only	QPLL	-	VCU118 + Inrevium DP1.4 FMC	10	MicroBlaze

Notes:

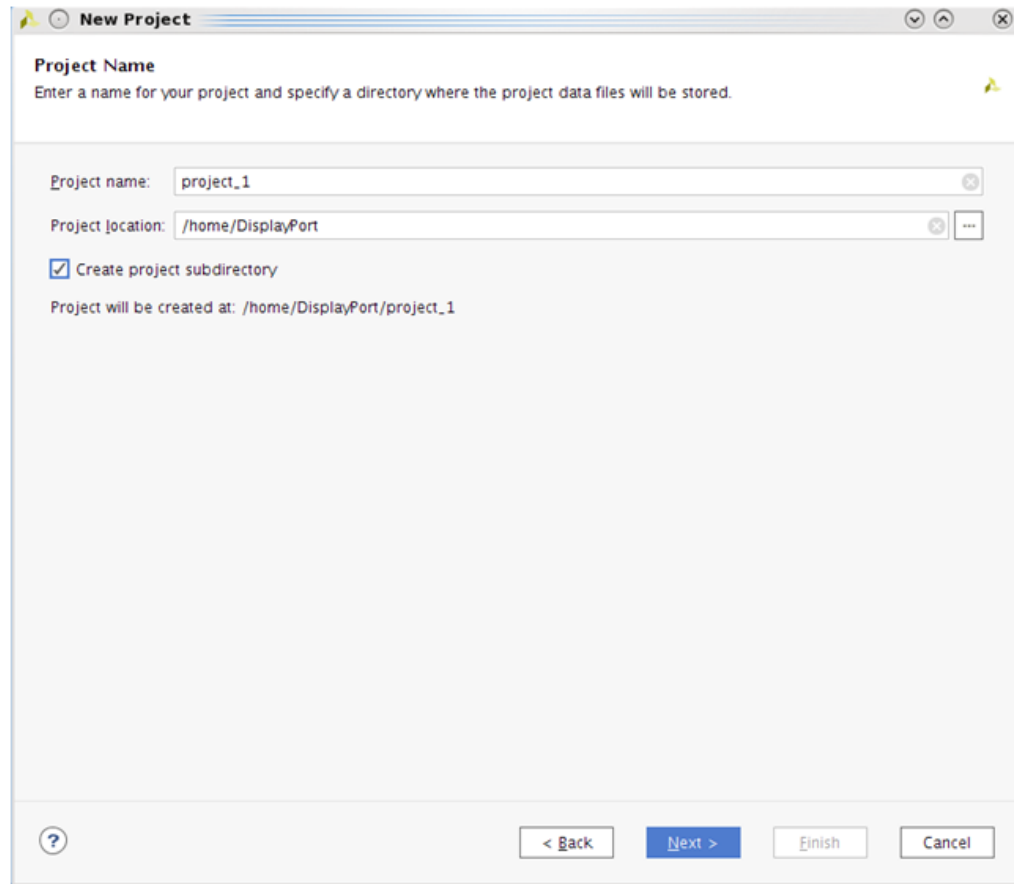
1. Line rate is limited to 5.4 Gb/s for this example design.
2. GT data width is 2 bytes.

Building the Example Design

1. Open the Vivado Design Suite and click **Create Project**.



2. In the **New Project** window, enter a **Project name**, **Project location**, and click **Next** up to the Board/Part selection window.



3. In the **Default Part** window, select the Board as per your requirement. Application Example Designs are available for KCU105, ZCU102, and VCU118.

Default Part

Choose a default Xilinx part or board for your project. This can be changed later.

Select:

Parts

Boards

Filter/ Preview

Vendor:

All

Display Name:

All













Board Rev:

Latest

Reset All Filters

Search:

Q-

Display Name	Vendor	Board Rev	Part
 ZedBoard Zynq Evaluation and Development Kit	em.avnet.com	d	 xc7z020clg484-1
 Artix-7 AC701 Evaluation Platform	xilinx.com	1.1	 xc7a200tbg676-2
 Kintex-7 KC705 Evaluation Platform	xilinx.com	1.1	 xc7k325tffg900-2
 Kintex-UltraScale KCU105 Evaluation Platform	xilinx.com	1.0	 xc7ku040-ffva1156-2-e
 Kintex UltraScale+ KCU116 Evaluation Platform	xilinx.com	1.0	 xc7ku5p-ffvb676-2-e
 Kintex UltraScale KCU1500 Acceleration Development Board	xilinx.com	1.0	 xc7ku115-flvb2104-2-e

Board Connectors

FMC_HPC

FMC_LPC

Target Connections

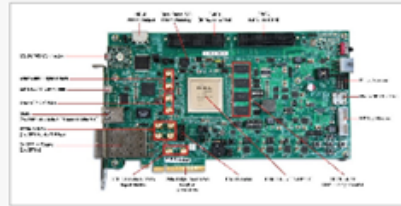
?

< Back

Next >

Finish

Cancel



4. Click **Finish**.

PG300 (v2.0) December 5, 2018
DisplayPort 1.4 RX Subsystem v2.0

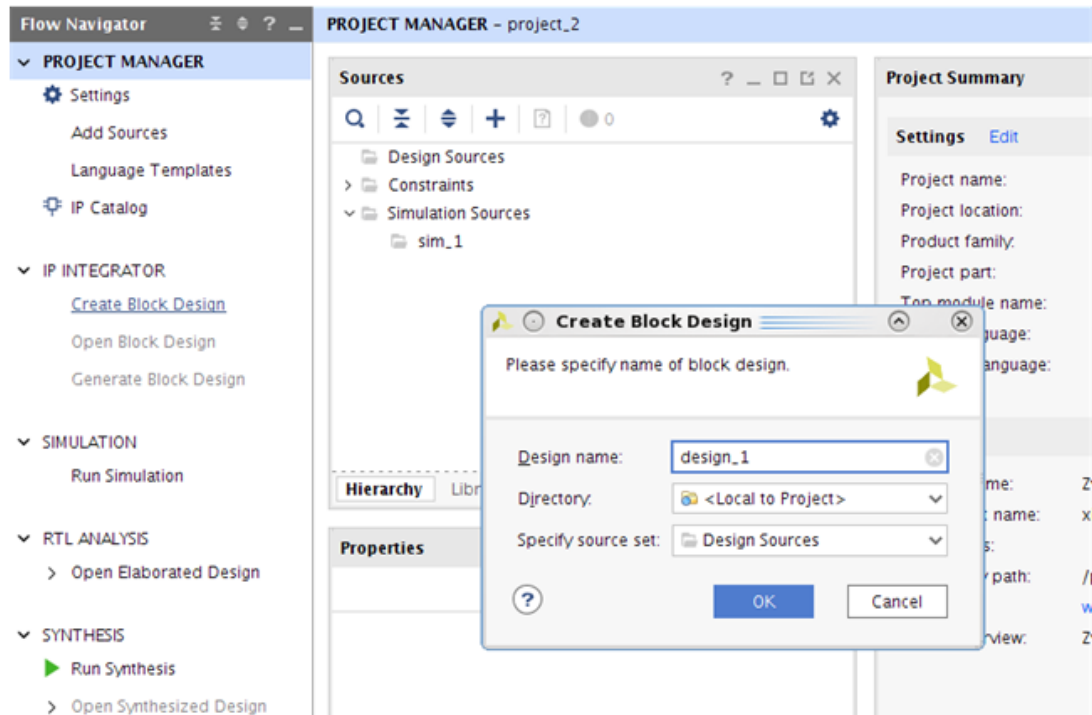
Send Feedback

www.xilinx.com

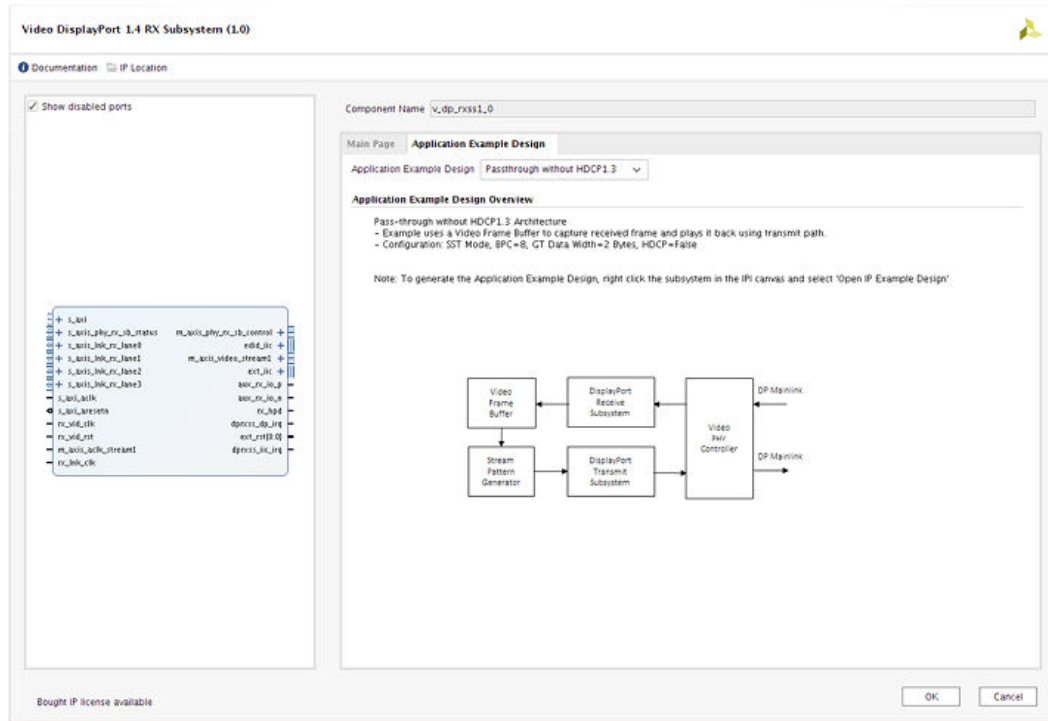
68



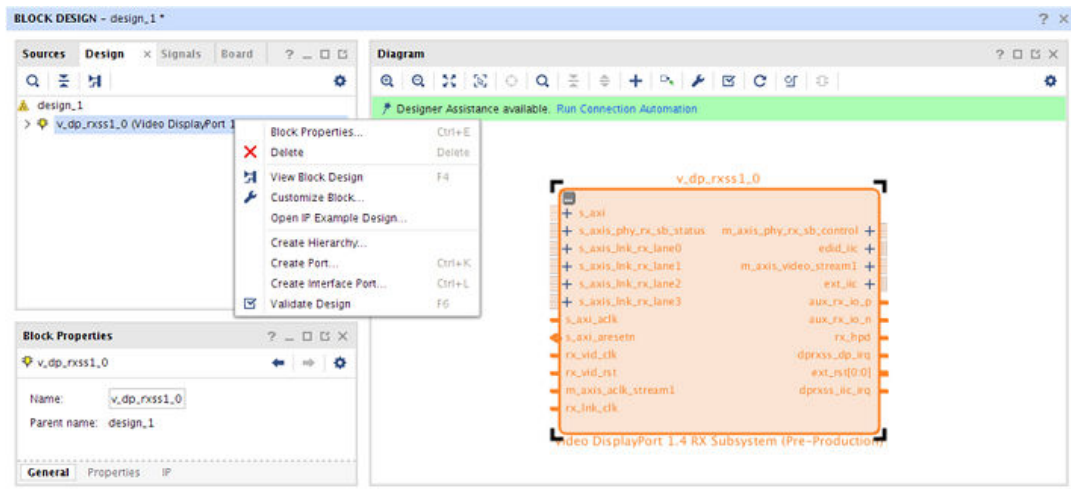
- In the Flow Navigator window, click **Create Block Design (BD)**. Select a name for the Block Design and click **OK**.



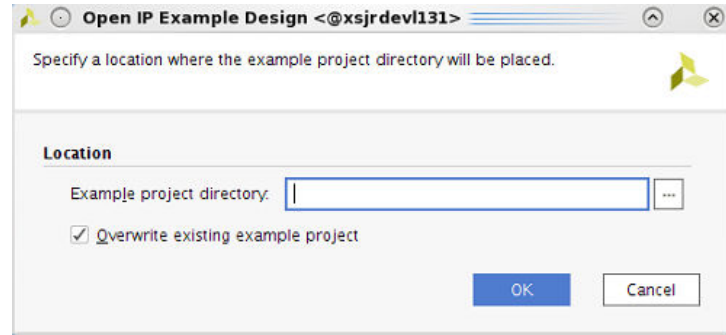
6. Right-click **BD** and click **Add IP**. Search for DisplayPort 1.4 and select either the DisplayPort 1.4 Receiver Subsystem IP (for RX only (ZCU102, VCU118), Pass-through (KCU105, ZCU102) designs) or the DisplayPort 1.4 Transmitter Subsystem IP (for TX only (ZCU102, VCU118), or Pass-through (KCU105) designs).
7. Double-click the **IP** and go to the **Application Example Design** tab in the **Customize IP** window. Select the supported topology in the **Application Example Design** drop-down box. Click **OK** and **Save** the block design.



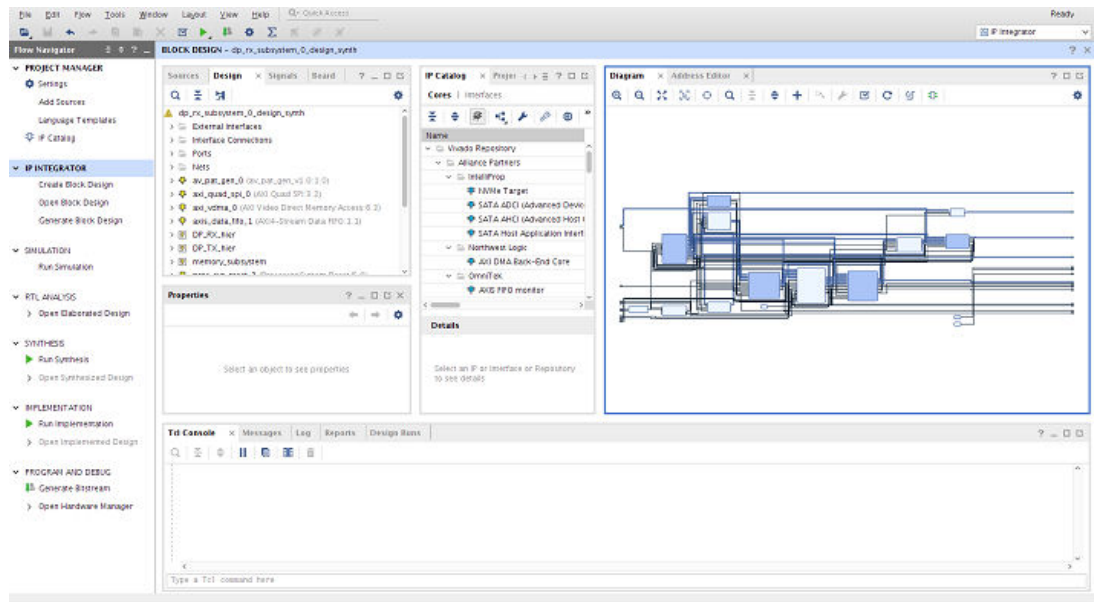
8. Right-click the **DisplayPort Subsystem** IP under Design source in the **Design** tab and click **Open IP Example Design**.



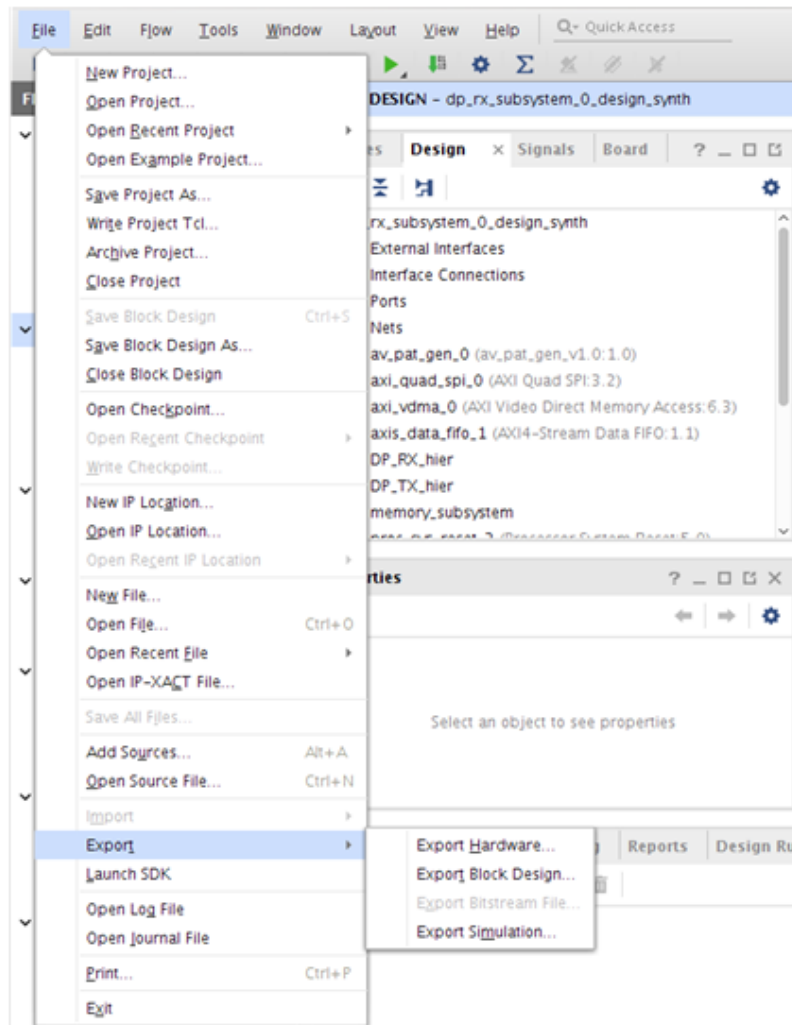
9. Choose **Example project directory** and click **OK**.



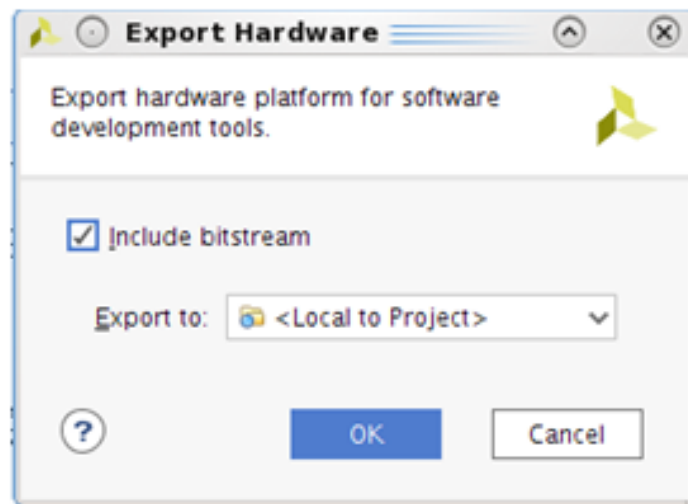
10. The following figure shows the Vivado IP integrator design. Choose the **Generate Bitstream**.



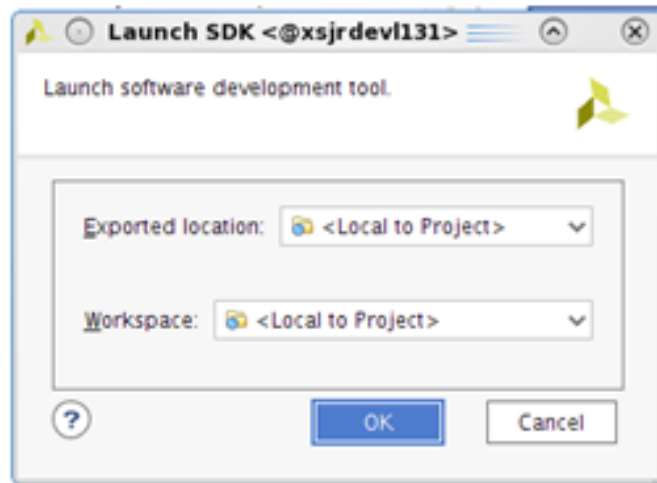
11. Export the hardware to SDK. Click **File** → **Export** → **Export Hardware**.



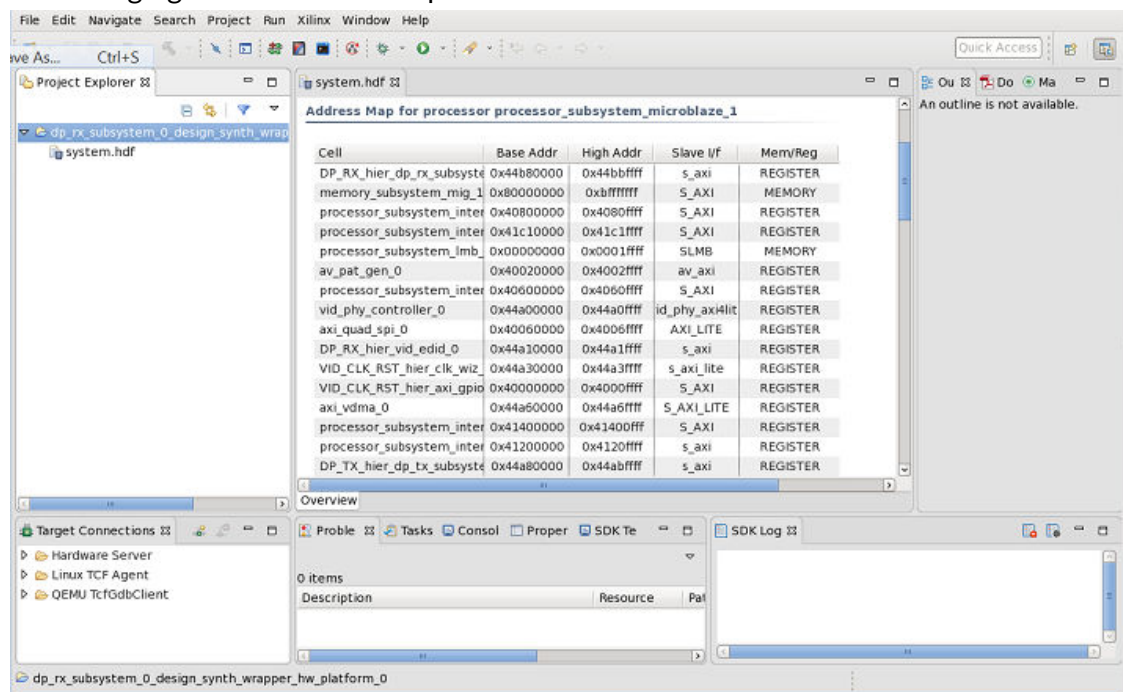
12. Ensure the **Include bitstream** is enabled and click **OK** (use the default **Export Location <Local to Project>**).



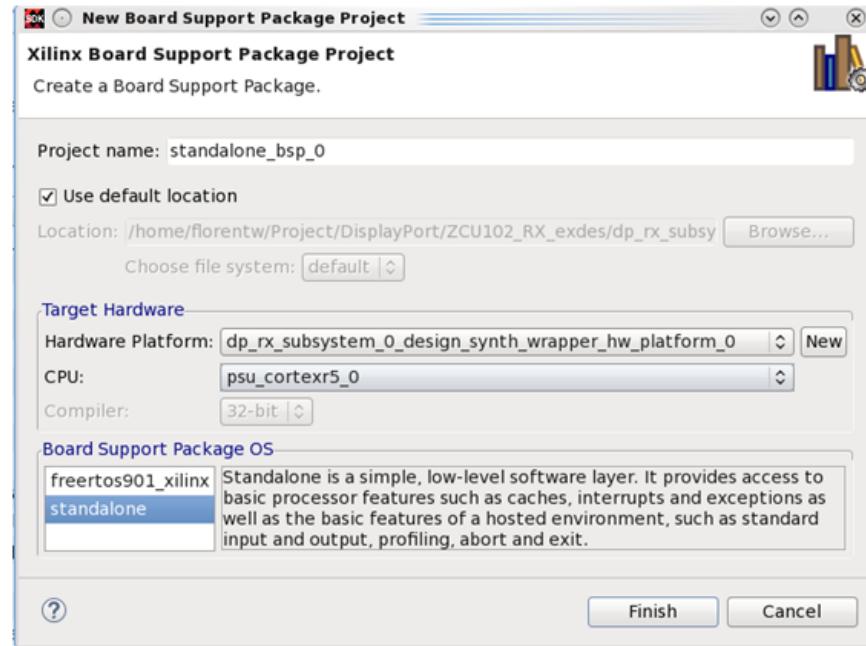
13. Click **File** → **Launch SDK**. Choose the SDK Workspace location. Keep the exported location default configuration (<Local to Project>).



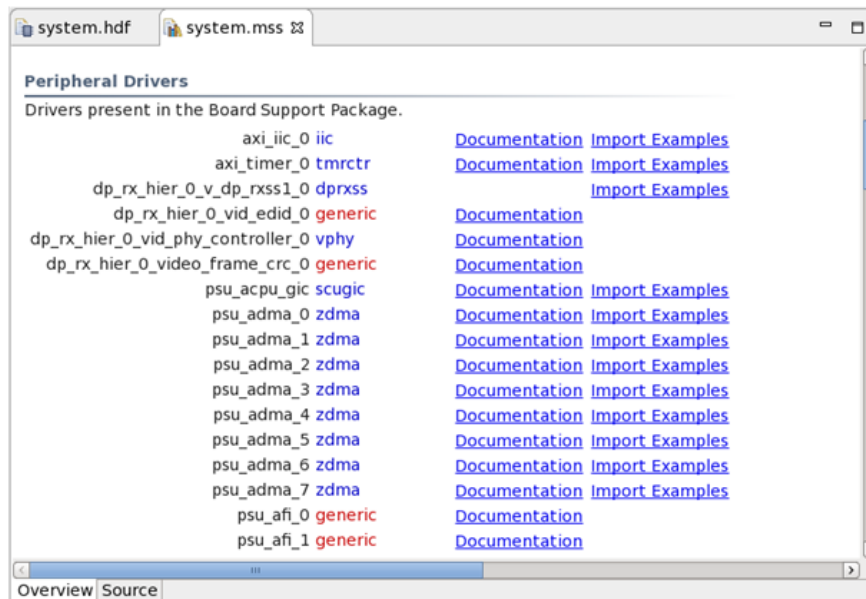
14. The following figure shows an example of the launched Vivado SDK.

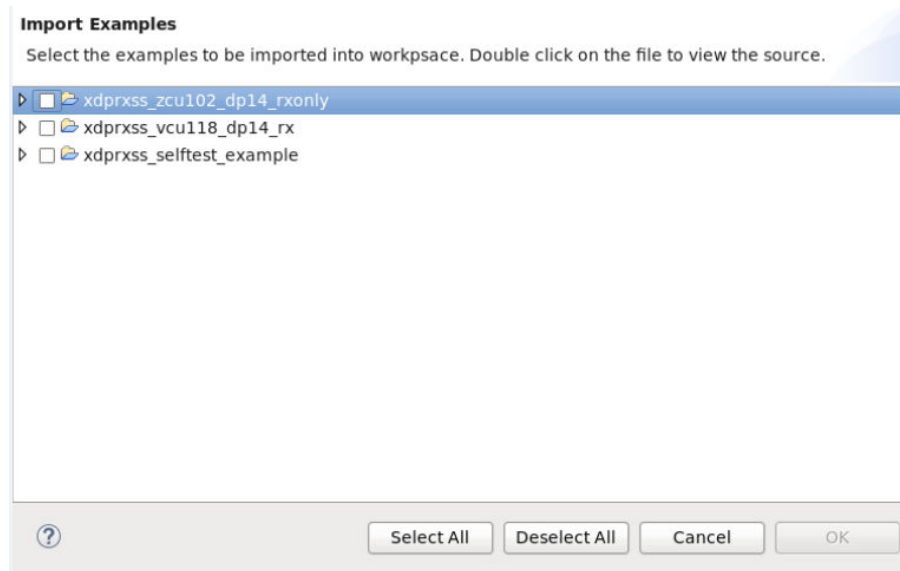


15. To create a Board Support Package (BSP), click **File** > **New** > **Board Support Package**. Enter the BSP **Project name**, click **Finish**, and then **OK**. For the ZCU102 board, ensure the target CPU is the Cortex™select *_kcu105select *_kcu105 R5_0 (psu_cortexr5_0).



16. Find the DisplayPort 1.4 RX/TX Subsystem driver in the `system.mss` file. If it is not already opened, open the file from the BSP in the **Project Explorer**. Click **Import Examples**.

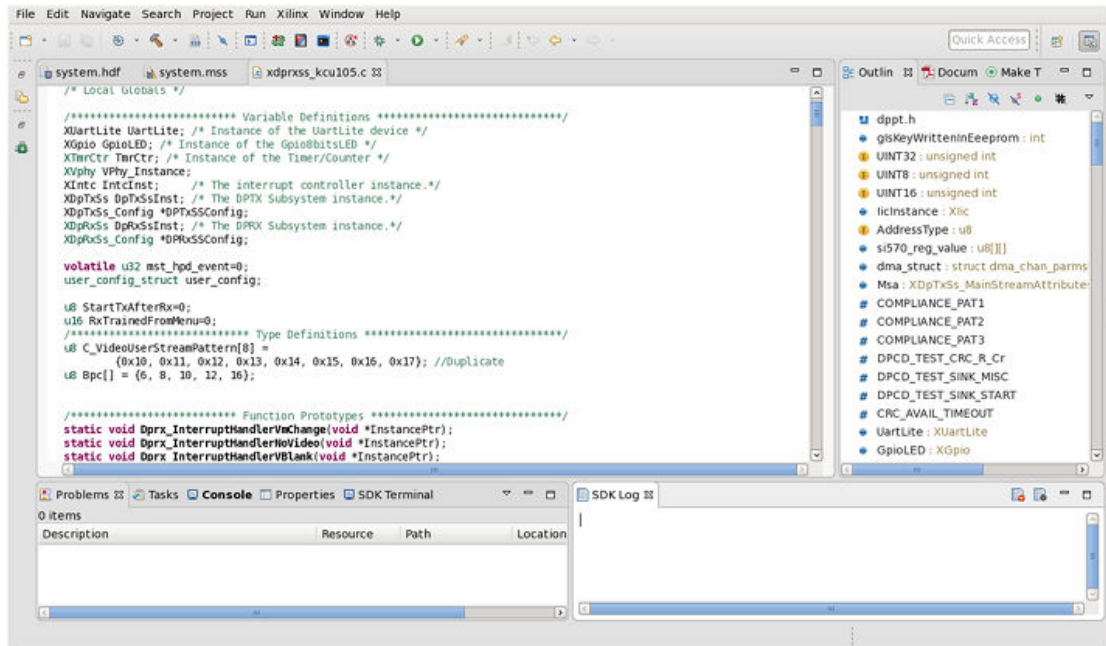




17. Select the Example Application corresponding to your hardware:

- For the Pass-through KCU105 project, select the *_kcu105_dp14 option.
- For the RX only ZCU102 project, select the *_zcu102_dp14_rxonly option.
- For the TX- only ZCU102 project, select the *_zcu102_dp14_tx option.
- For the RX-only VCU118 project, select the *_vcu118_dp14_rx option
- For the TX-only VCU118 project, select the *_vcu118_dp14_txonly option.
- For the Passthrough ZCU102 with I2S audio interface project, select the *_zcu102_dp14_pt option.
- For the Passthrough ZCU102 with HDCP project, select the *_zcu102_dp14_pt_hdcp option.

18. The following figure shows the example application successfully built and ready to use.



Hardware Setup and Run

1. Connect the Tokyo Electron Device Limited (TED) TB-FMCH-VFMC-DP module to the HPC FMC connector on the KCU105 board or to the HPC0 connector on the ZCU102 or to the FMCP HSPC connector on the VCU118 depending on your design.
2. Connect a USB cable (Type A to mini B) from the host PC to the USB UART port on the KCU105 for serial communication. For the KCU105, ZCU102, or VCU118, use Type A to micro B type of USB cable.
3. Connect a JTAG USB Platform cable or a USB Type A to Micro B cable from the host PC to the board for programming bit and elf files.
4. For pass-through or TX-only applications, connect a DP cable from the TX port of the TED TB-FMCH-VFMC-DP module to a monitor, as shown in the following figure.
5. For pass-through or RX-only applications, connect a DP cable from the RX port of the TED TB-FMCH-VFMC-DP module to a DP source (GPU), as shown in the following figure.

Figure 16: KCU105 Board Setup

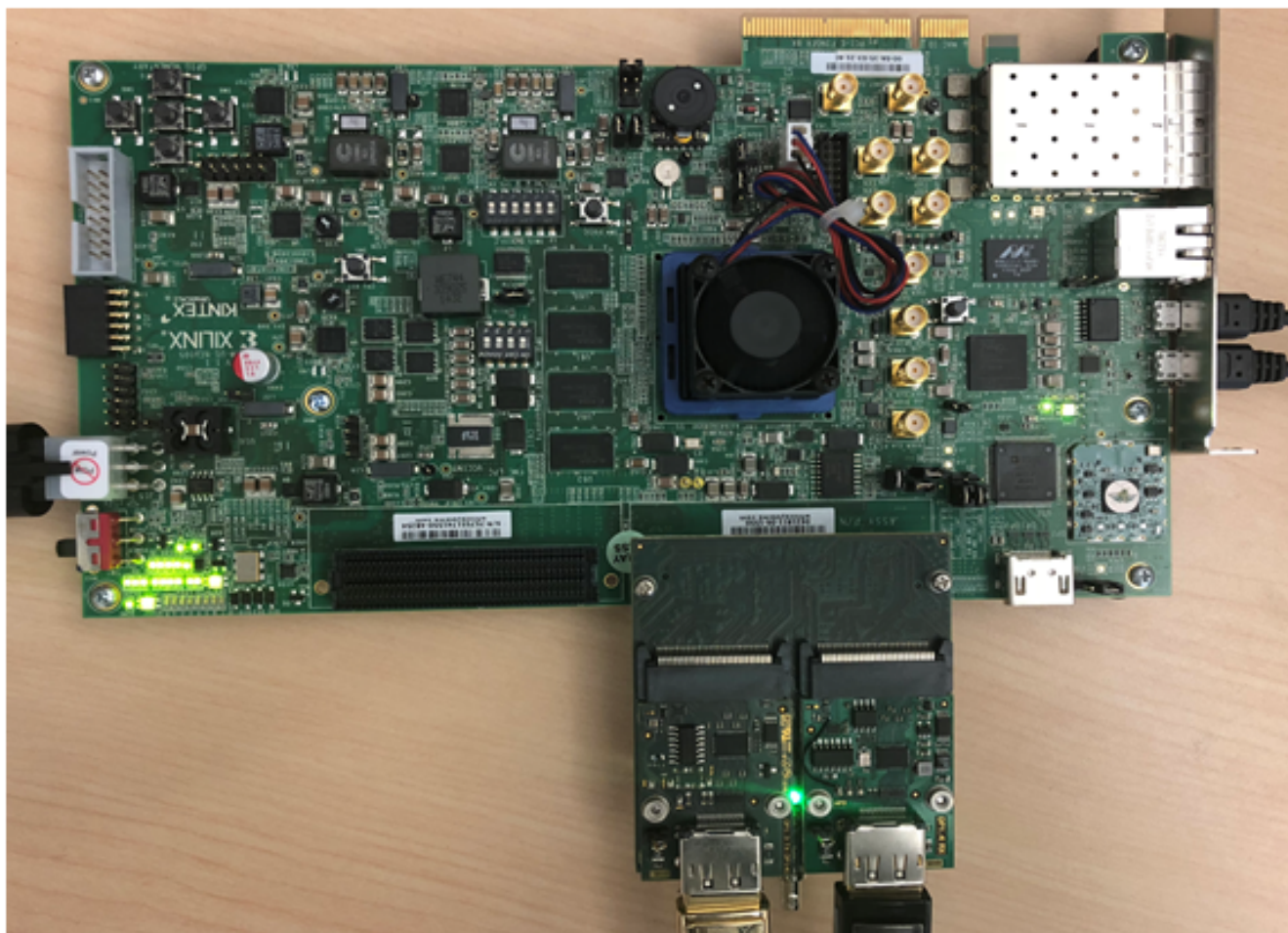
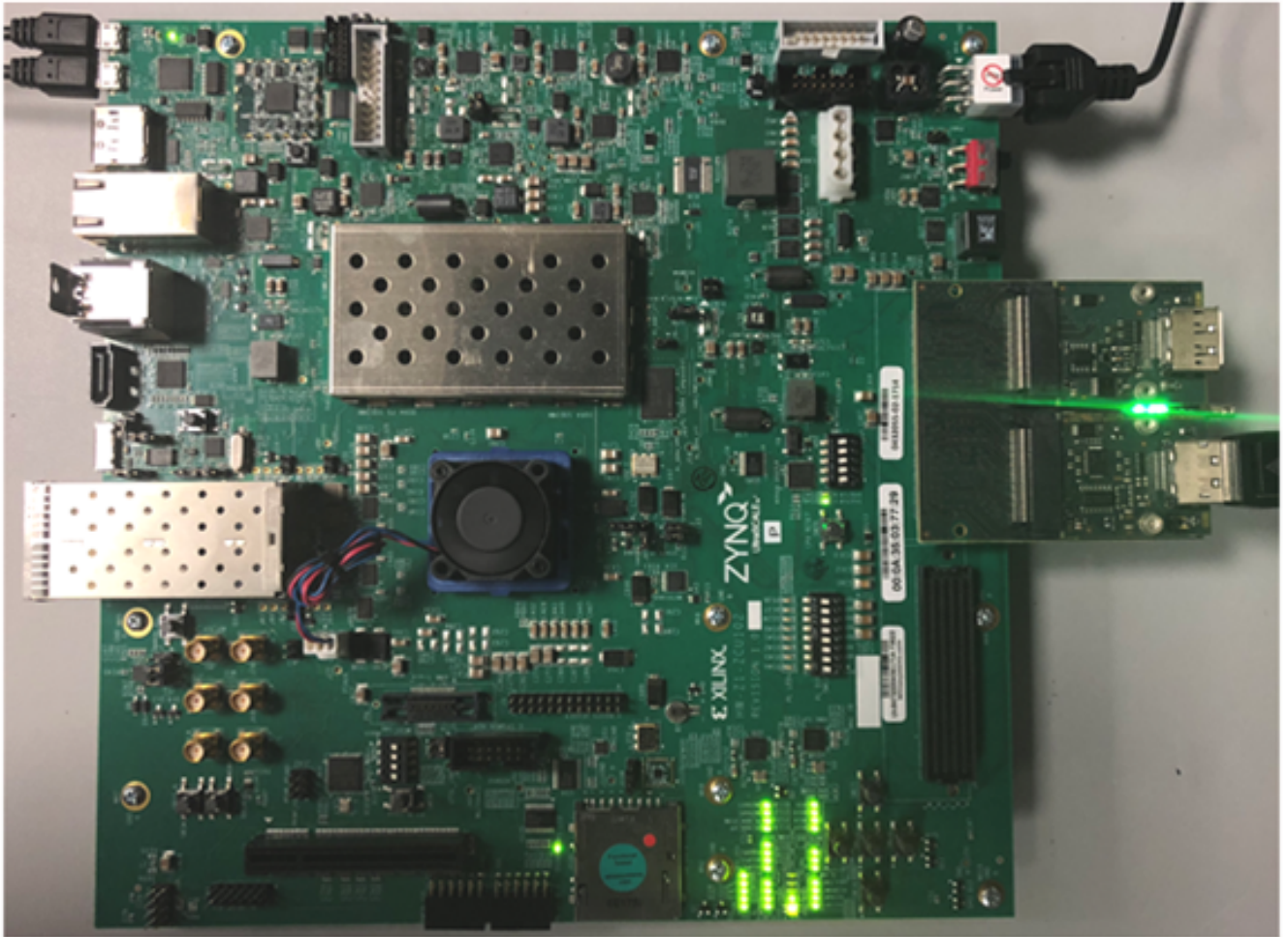
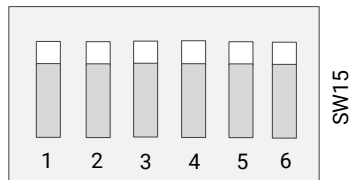


Figure 17: ZCU102 Board Setup

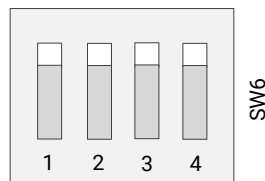


6. On the KCU105 set the mode pin to SW15:



X20381-062518

7. On the ZCU102 set the mode pin to SW6:



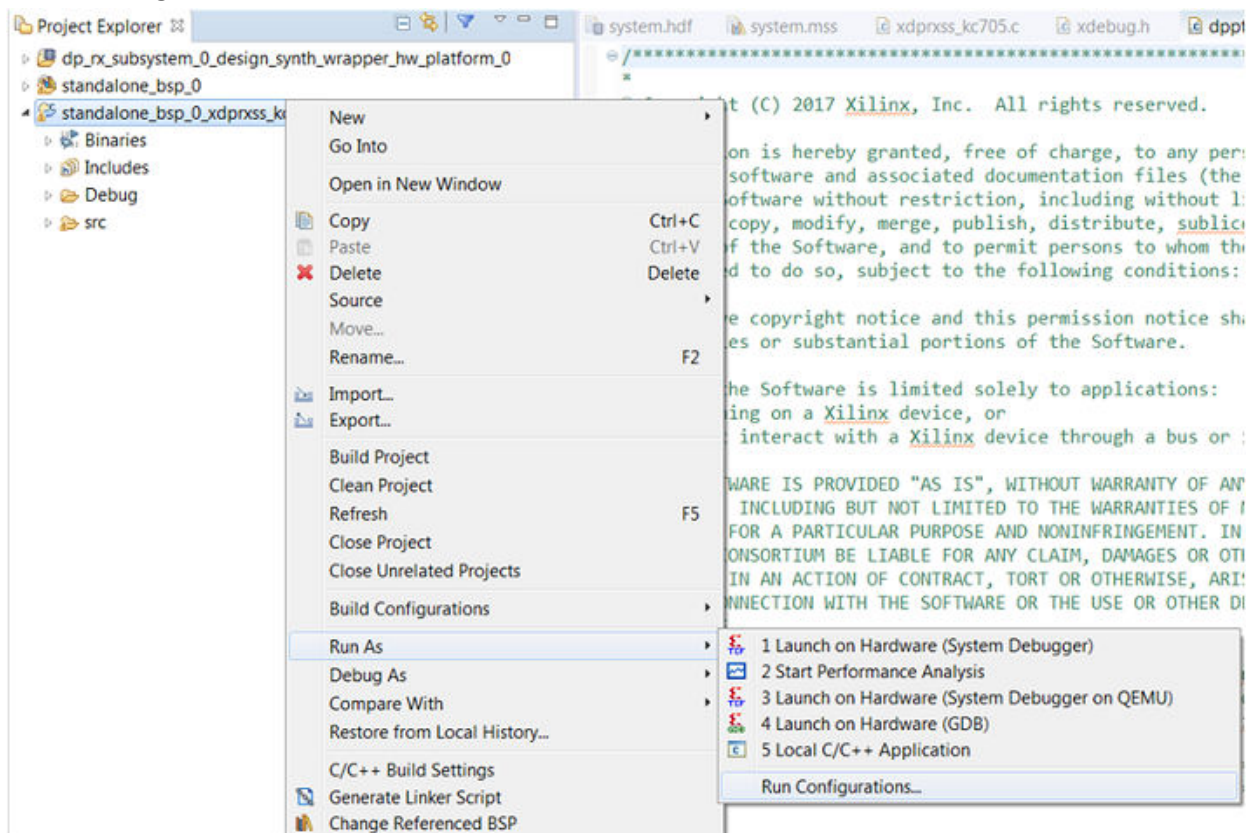
X19805-062518

8. Connect the power supply and power on the board.

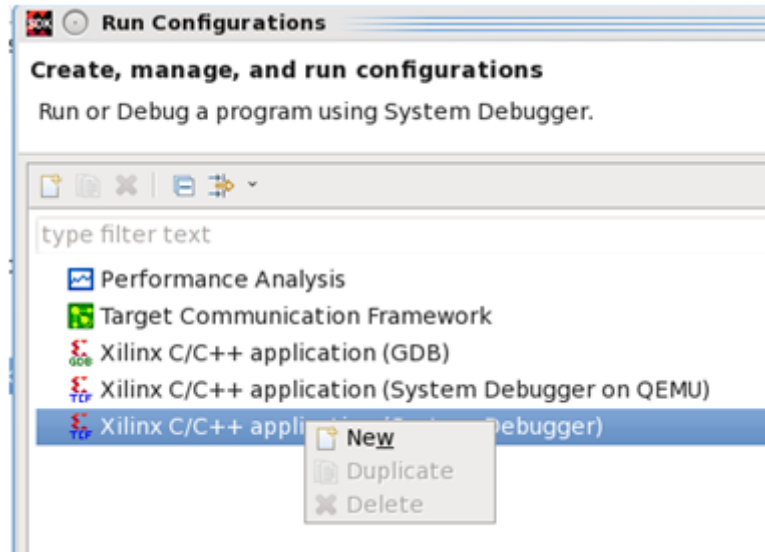
9. Start an UART terminal program such as Tera Term or Putty with the following settings:
 - a. Baud rate = 115200 for KCU105/ZCU102, 9600 for VCU118
 - b. Data bits = 8
 - c. Parity = none
 - d. Stop bits = 1
 - e. Flow Control = none

Note: With the ZCU102 board, there are four COM ports available.

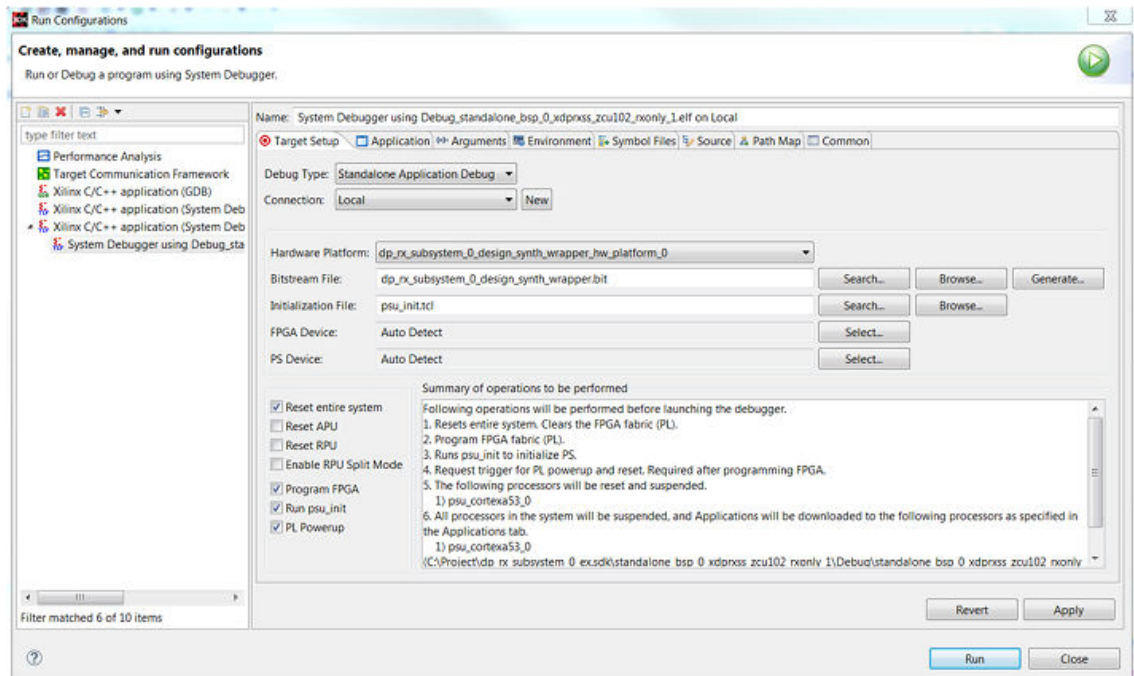
10. In the Vivado SDK, under the **Project Explorer**, right-click the application and click **Run As** → **Run Configurations**.



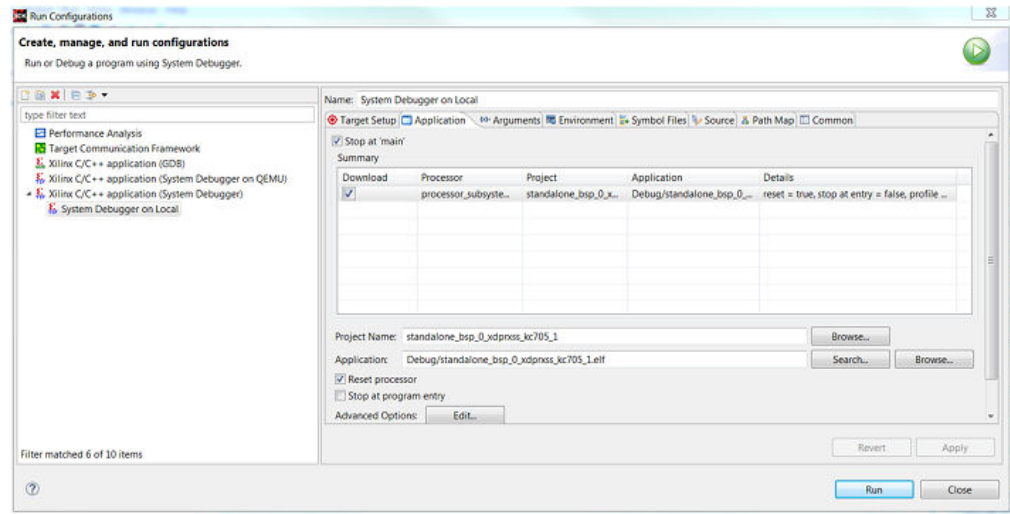
11. In the **Run Configurations** popup menu, right-click **Xilinx C/C++ application (System Debugger)** and click **New**.



12. In the **Target Setup** tab, ensure the Connection is set to **Local** and the **Reset entire system** and **Program FPGA** are enabled. If running the ZCU102, also ensure that **Run psu_init** and **PL Powerup** are enabled.



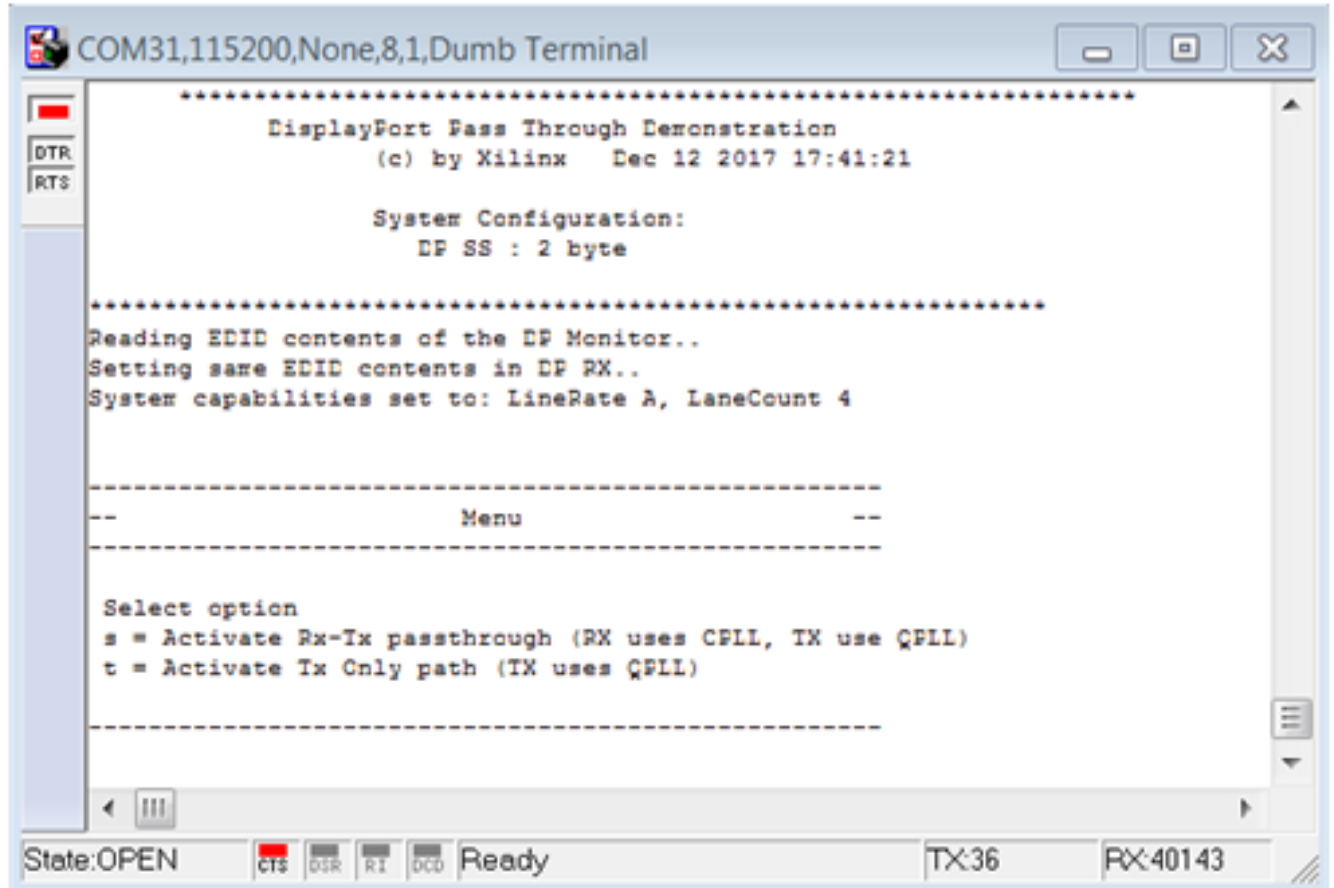
13. In the **Application** tab, ensure the application download is enabled and click **Run**.



Display User Console

In a Pass-Through application (KCU105), as soon as the application is executed, it checks if a Monitor is connected or not. If a monitor is already connected, then it starts up the following options as shown in the following figure to choose from (KCU105).

Figure 18: DisplayPort User Console



Selecting either `r` or `s` puts the system in Pass-Through mode, where the video received by the RX is forwarded to the TX. This configures the `vid_phy_controller` and sets up the DisplayPort for RX. If a DisplayPort Source (for example, GPU) is already connected to DisplayPort RX Subsystem, then it starts the training. Otherwise, the training starts when the cable is plugged in. As soon as the training is completed, the application starts the DisplayPort TX Subsystem. The video should be seen on the monitor after the TX is up. The previous figure shows the UART transcript. The transcript might differ based on the training done by the GPU.

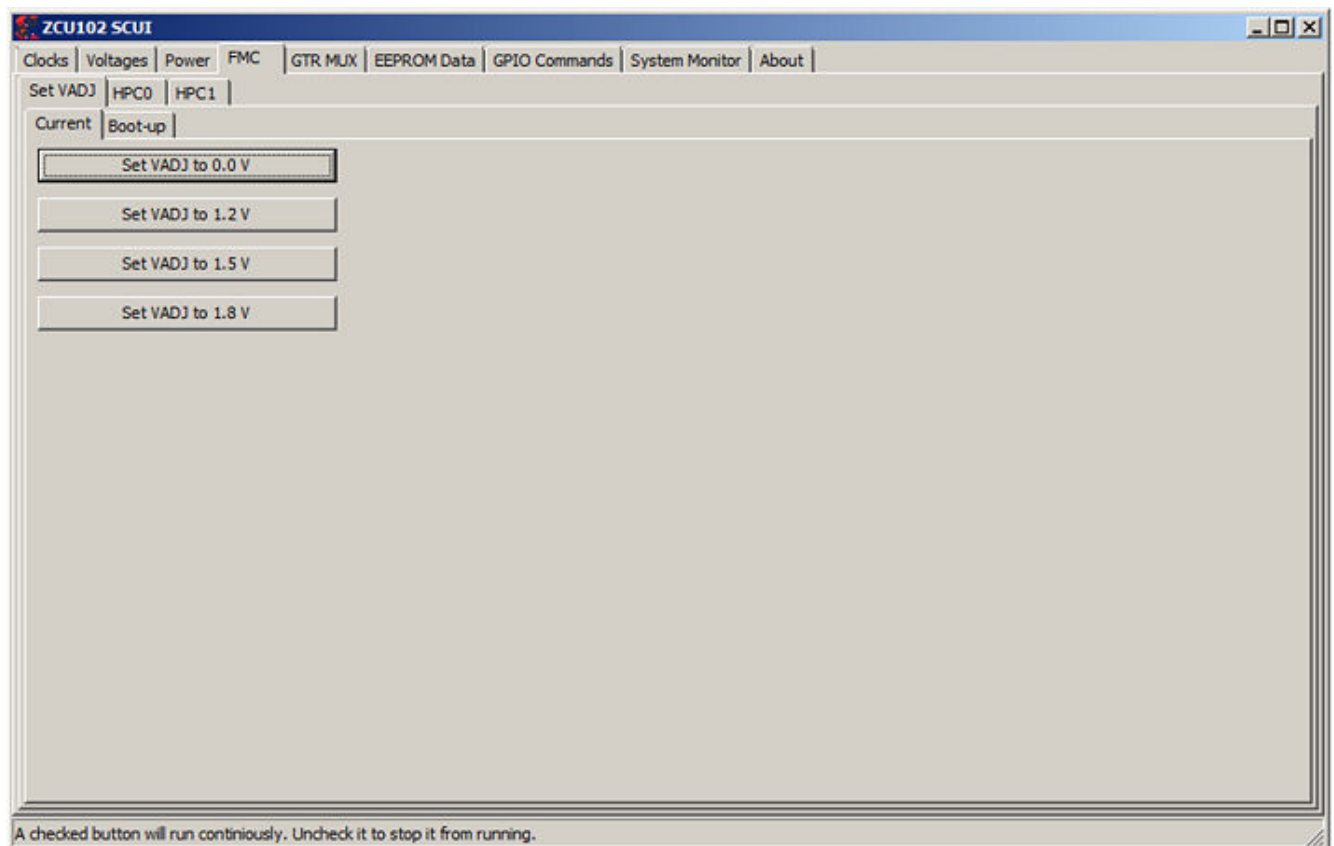
Setting the FMC Voltage to 1.8V

To run the example design on the ZCU102 board, ensure that only one ZCU102 board is connected to the host PC. This tool does not work with multiple ZCU102 connected to the host PC. There is no UART selection in this tool. Also, the FMC voltage is set to 1.8V. If you forget to set the FMC voltage, the following symptoms might occur:

- Random AUX failures
- Training failures

To set the FMC VADJ voltage:

1. Connect the ZCU102 board from the host PC to the USB UART port and power up the board.
2. Open the ZCU102 SCUI tool and select the **FMC** tab.
3. On the **Set VADJ** tab, select the **Set VADJ to 1.8V**.



Tested Equipment

The following table lists the tested equipment used with the example design.

Table 36: Source Equipment

Sink Type	Brand Name	Model Name	Driver Version	Platform
GPU	NVIDIA	GTX 1080	21.21.13.7290	Windows 10
GPU	NVIDIA	GTX1060	-	Windows 10
GPU	AMD	RX 460	21.19.384.37	Windows 10
GPU	AMD	R460	-	-
GPU	AMD	7600	-	-
Laptop	Dell	PrecisionM4700	-	-
Laptop	Dell	E7440	-	-
Laptop	Apple	MBP	2014	macOS
Tester	Unigraf	DPT-200	-	-
Tester	Unigraf	DPT-323	-	-
Tester	Unigraf	DPT-400	-	-

Upgrading

There is no direct upgrade path due to the new retimer. Xilinx[®] recommends starting with a new design.

Frequently Asked Questions

Q. Can both RX and TX be used on the same GT quad for DisplayPort?

A. Yes. The Video PHY Controller supports the capability of performing both RX and TX on the GT quads. However, they cannot be different protocols.

Q. Does the Video PHY Controller support different protocols for RX and TX?

A. No. The Video PHY Controller must use the same protocol if both RX and TX is being used.

Q. I am having link training issues. What are some things that can be done to improve link training?

A. Perform the following:

1. Verify that all relevant ARs are taken into account.
2. Increase the AUX_DEFER value in register offset `0x004`.

Q. Does the Xilinx[®] subsystem support my resolution and frame rate?

A. DisplayPort should operate at any resolution and frame rate as long as the DisplayPort link is not oversubscribed. Use the following equation to determine if the custom resolution can be supported:

$$(H_{\text{Total}} \times V_{\text{Total}} \times \text{bits_per_component} \times \text{frame rate}) < (0.8 \times \text{link_lane} \times \text{num_lanes})$$

Q. Can I get more information on EDID?

A. The EDID block is outside of the DisplayPort controller and is connected using the I2C interface. It is your responsibility to add the proper EDID content. If you select a different clock source for the EDID block, ensure that the I2C bus has the proper false path constraints.

Driver Documentation

The driver documentation can be found at the [Xilinx GitHub](#) page.

Debugging

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

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

- Vivado Synthesis
- Vivado Implementation
- `write_bitstream` (Tcl command)

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

Finding Help on Xilinx.com

To help in the design and debug process when using the subsystem, the [Xilinx Support web page](#) contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the subsystem. This guide, along with documentation related to all products that aid in the design process, can be found on the [Xilinx Support web page](#) or by using the Xilinx[®] Documentation Navigator. Download the Xilinx Documentation Navigator from the [Downloads page](#). For more information about this tool and the features available, open the online help after installation.

Answer Records

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

Answer Records for this subsystem can be located by using the Search Support box on the main [Xilinx support web page](#). To maximize your search results, use 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 DisplayPort 1.4 RX Subsystem

AR [70294](#)

Technical Support

Xilinx provides technical support in the Xilinx Support web page 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 contact Xilinx Technical Support, navigate to the [Xilinx Support web page](#).

Debug Tools

There are many tools available to address RX 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](#)).

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing.

Receive – Training

This section contains debugging steps if the clock recovery or channel equalization is not happening at sink.

1. Try with a different source such as the DisplayPort Analyzer.
2. Change the cable and check again.
3. Put an AUX Analyzer in the Receive path and check if the various training stages match with those mentioned in DisplayPort Overview.
4. Probe the `lnk_clk` output and check the SI of the Clock is within the Phase Noise mask of the respective GT.
5. Check the RX Initialization Status register (`0x0028`) and PLL Lock Status (`0x0018`) register of the Video PHY Controller for Reset done and PLL lock for the active lanes.
6. Check the `0x43C` and `0x440` registers for Symbol_Locked, Channel Equalization, and Clock Recovery Done.

Related Information

[DisplayPort Overview](#)

Receive – Retimer Related Issues

This section contains debugging steps for issues related to Retimer. Proper operation of Retimer is essential for the training to complete successfully.

- IIC checks:
 - Check if the IIC speed is 100 KHz or higher speed (1 MHz).
 - Check if the IIC writes are happening properly to the Retimer IC.
 - Check if the IIC writes are interrupt or polling based. If it is interrupt based, it is like calling an interrupt within another interrupt routine. Ensure this function is correct, or better to go with the polling mode, as Retimer IIC writes are supposed to happen at Retimer training events.
- Until the training is done ensure only the TP1 and TP23 interrupts are enabled.
- Ensure that you have no other software code in between TP1 interrupt to training done duration.
- Check whether the TP1 and TP23 handlers are called correctly when the TP1 and TP23 interrupts are detected.
- Probe the `lnk_clk` output and check the SI of the Clock is within the Phase Noise mask of the respective GT.
- Avoid using PRINTF to monitor the Retimer configuration as the configuration must be completed as quickly as possible to meet the DisplayPort Standard requirements.

Receive – Issues After Training

This section contains debugging steps if the monitor is not displaying video even after a successful training or if the monitor display is noisy.

1. If the video timing counters are reporting 0 lines, toggle the DTG enable and software-video reset and check again.
2. Check the symbol and disparity error counters `0x448` and `0x44C` through AXI reads. If the errors are accumulating, the alignment bit might go off eventually. Perform `dprx_init` once and toggle HPD so the source can train the sink again.
3. Training lost can occur:
 - When there is change in link configuration and RX is in previously trained state
 - Either symbol lock/channel equalization/clock recovery failure
 - Lane inactivity

Receive – Audio

If the audio is not played at the Sink device or the audio is noisy, check if the programming steps mentioned in Audio Management have been followed correctly.

Related Information

[Audio Management](#)

Receive – Sink MST

This section contains debugging steps for issues with the Sink device.

- Check if the connected GPU is recognizing the streams correctly. Read the MSA of all the streams and verify against the GPU data.
- Read the VC Payload table through an AXI write and check if the allocated stream IDs are sequential in the slots. The 0th slot is not used and should not contain any of the allocated stream IDs.
- Check the symbol and disparity error counters through AXI reads. If there are a lot of errors, there might be video defects.
- Check with AUX analyzer to see if all the sideband messages are decoded properly.
- Check the link rate and lane count at which it is trained. Four streams of 1080p are possible only in 5.4 x 4. With HBR, only two 1080p streams are possible. The link rate downshift might be because of a training failure—ensure that a DisplayPort v1.4 cable is used.
- Make sure a DisplayPort v1.4 cable is used with MCDP6000 in between source and sink.

Receive – FIFO Overflow

How do I resolve the USER_FIFO_OVERFLOW interrupts (0x110) when I am using the DisplayPort in Receiver mode?

This is caused when the incoming DisplayPort data stream on the `lnk_clk` domain is not fast enough compared to the outgoing data stream on the `rx_vid_clk` domain.

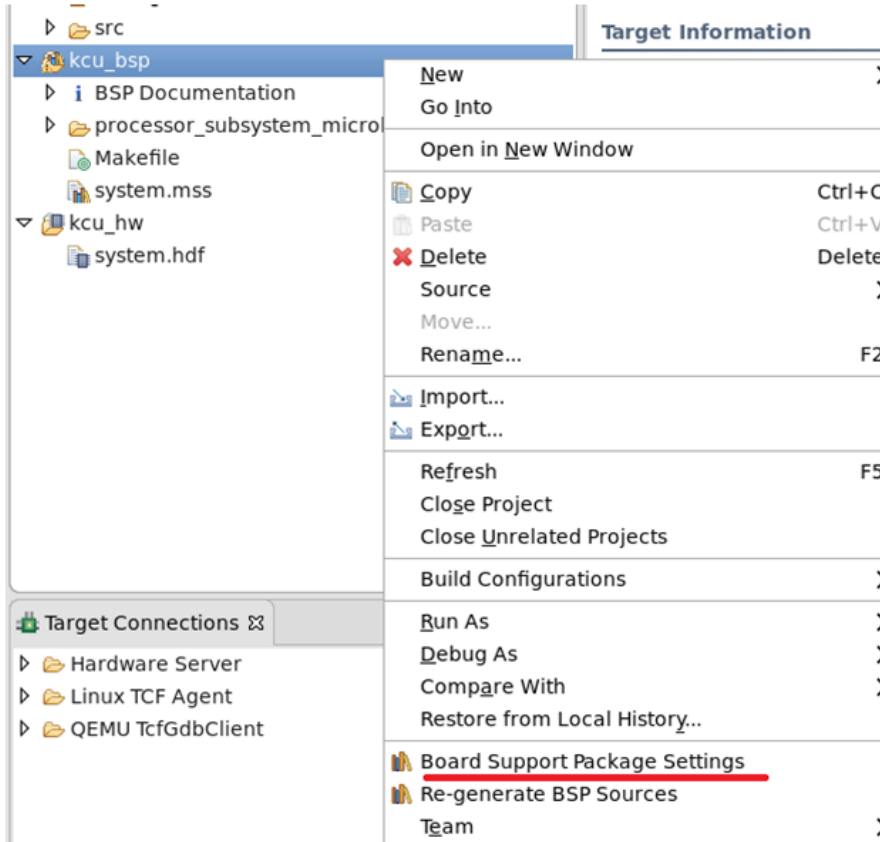
There are two ways to resolve this:

1. If possible, increase HBLANK from the source.
2. Increase the `rx_vid_clk` frequency to the maximum tested of 270 MHz.

Software Debug

This section shows how to navigate to the DisplayPort debug driver information.

1. Click **bsp** in the SDK and right-click to select **Board Support Package Settings** (see figure).



2. The **Board Support Package Settings** window pops up. Navigate to the **processor_subsystem_microblaze_*** (see figure below).
3. Select the **extra_compiler_flags** in the Name column. Add the **-DDEBUG** flag in the Value column. You can view all of the debug information from the DisplayPort driver as well as DisplayPort subsystem driver debug information.

Board Support Package Settings

Control various settings of your Board Support Package.

- Overview
 - standalone
 - drivers**
 - processor_subsystem_microblaze_1**

Name	Value	Default
archiver	mb-ar	mb-ar
compiler	mb-gcc	mb-gcc
compiler_flags	-O2 -c	-O2 -c
extra_compiler_flags	all -Wextra -DDEBUG	-ffunction-sections
xmdstub_peripheral	none	none

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. Video PHY Controller LogiCORE IP Product Guide ([PG230](#))
2. UltraFast Design Methodology Guide for the Vivado Design Suite ([UG949](#))
3. VESA DisplayPort Standard v1.4 ([VESA website](#))
4. Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator ([UG994](#))
5. AXI4-Stream Video IP and System Design Guide ([UG934](#))
6. AXI Interconnect LogiCORE IP Product Guide ([PG059](#))
7. AXI IIC Bus Interface LogiCORE IP Product Guide ([PG090](#))
8. Vivado Design Suite User Guide: Designing with IP ([UG896](#))
9. Vivado Design Suite User Guide: Getting Started ([UG910](#))
10. Vivado Design Suite User Guide: Logic Simulation ([UG900](#))
11. ISE to Vivado Design Suite Migration Guide ([UG911](#))
12. Vivado Design Suite User Guide: Programming and Debugging ([UG908](#))
13. Vivado Design Suite User Guide: Implementation ([UG904](#))
14. Vivado Design Suite: AXI Reference Guide ([UG1037](#))
15. AXI Timer LogiCORE IP Product Guide ([PG079](#))
16. ZCU102 System Controller GUI Tutorial ([XTP433](#)) (registration required)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
12/05/2018 v2.0	
HDCP Key Interface	HDCP Ports added
DisplayPort MST Stream	MST Ports added
Programming the Core in MST Mode	MST Programming added
Pixel Mapping Examples on AXI4-Stream Interface (UG934-Compliant)	UG934-compliant pixel mapping
04/04/2018 v1.0	
Initial Xilinx release.	N/A

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby **DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE**; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at <https://www.xilinx.com/legal.htm#tos>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at <https://www.xilinx.com/legal.htm#tos>.

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

© Copyright 2018 Xilinx, Inc. Xilinx, the Xilinx logo, Alveo, Artix, ISE, Kintex, Spartan, Versal, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. AMBA, AMBA Designer, Arm, ARM1176JZ-S, CoreSight, Cortex, PrimeCell, Mali, and MPCore are trademarks of Arm Limited in the EU and other countries. The DisplayPort Icon is a trademark of the Video Electronics Standards Association, registered in the U.S. and other countries. All other trademarks are the property of their respective owners.