DisplayPort RX Subsystem v2.0

Product Guide

Vivado Design Suite

PG233 July 14, 2017

XILINX
ALL PROGRAMMABLE™

Table of Contents

Appendix B: Application Software Development

Appendix C: Additional Resources and Legal Notices

Introduction

DisplayPort RX Subsystem is a plug-in solution for serial digital video data reception in large Video systems of up to video resolutions of Ultra HD (UHD) at 60 fps. The subsystem provides ease of use in selecting the required mode and the rest of the customization is automated.

Features

- Support for DisplayPort Sink (RX) capabilities
- Supports single stream transport (SST) and multi-stream transport (MST)
- Dynamic support for 1.62/2.7/5.4 Gb/s line rates
- Dynamic support of 6, 8, 10, 12, or 16 bits per component (BPC).
- Dynamic support of RGB and YCbCr444/ YCbCr422/Y-Only color formats.
- Supports Audio
- Supports HDCP 1.3
- AXI IIC controller for DP159 retimer programming
- Supports native or streaming video input interface
- Supports both 16-bit and 32-bit video PHY(GT) interface

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1)(2)	UltraScale+™ Families (GTHE4) Kintex® UltraScale™ (GTHE3) Virtex® UltraScale (GTHE3) 7 series (GTXE2) 7 series (GTHE2) (Discontinued in Vivado 2017.3) Artix®-7 (GTPE2) (Discontinued in Vivado 2017.4)
Supported User Interfaces	AXI4-Stream, AXI4-Lite
Resources	Performance and Resource Utilization web page
Provided with Core	
Design Files	Hierarchical subsystem packaged with DisplayPort RX core and other IP cores
Example Design	N/A
Test Bench	N/A
Constraints File	IP cores delivered with XDC files
Simulation Model	N/A
Supported S/W Driver	Standalone
Tested Design Flows(3)	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx at the Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. For HDCP: UltraScale/UltraScale+ supports up to 5.4 Gb/s, Kintex-7/Virtex-7 (-1 speed grade supports up to 2.7 Gb/s, -2/ -3 supports up to 5.4 Gb/s), and Artix-7 is not supported.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

This chapter contains an overview of the core as well as details about features, licensing, and standards. The DisplayPort RX subsystem is a full feature, hierarchically packaged subsystem with a DisplayPort sink (RX) core ready to use in applications in large video systems.

Feature Summary

- UHD up to 60 fps supports multi-stream transport (MST) and single stream transport (SST) modes.
- Dynamic support of different BPC and color formats.
- Pixel mode support in native video interface mode.
- Support for 2 to 8 channel audio.
- Support optional HDCP 1.3 Controller.
- Support for 16-bit or 32-bit GT width.
- Support for native and streaming video input interface.

Unsupported Features

- Audio is not supported in MST mode.
- In-band stereo is not supported.
- HDCP is not supported in MST mode.
- HDCP 2.x is not supported.
- Video Streaming interface is not scalable with dynamic pixel mode selection.
- Dual-pixel splitter is not supported in native video mode.

Licensing and Ordering

License Type

This subsystem requires a license for the DisplayPort Receive core, which is provided under the terms of the Xilinx Core License Agreement. The subsystem 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 core or subsystem. Contact your local Xilinx sales representative for information about pricing and availability.

For more information about licensing for the core, see the DisplayPort product page.

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

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

License Checkers

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

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

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

Product Specification

This chapter contains a high-level overview of the core as well as performance and port details.

Overview

The DisplayPort RX Subsystem operates in the following video modes:

- Single stream transport (SST)
- Multi-stream transport (MST)

Streaming Video Interface

In the SST or the MST mode, by default the subsystem is packaged with the DisplayPort Receive core, DP Video to AXI4-Stream Bridge and the AXI IIC controller for connecting to the TI DP159. The HDCP core along with an AXI Timer core are also present as part of the DisplayPort RX Subsystem, when the HDCP feature is enabled.

In the 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 RX Subsystem is hierarchically packaged, you select the parameters and the subsystem creates the required hardware. Figure 2-1 shows the architecture of the subsystem assuming MST with four streams.

The DisplayPort RX Subsystem receives the video using the DisplayPort v1.2 protocol over 32-bit or 16-bit video PHY interface. The DisplayPort RX Subsystem works in conjunction with the Video PHY Controller configured for DP protocol.The subsystem outputs multi-pixel Video to AXI4-Stream Protocol interface.

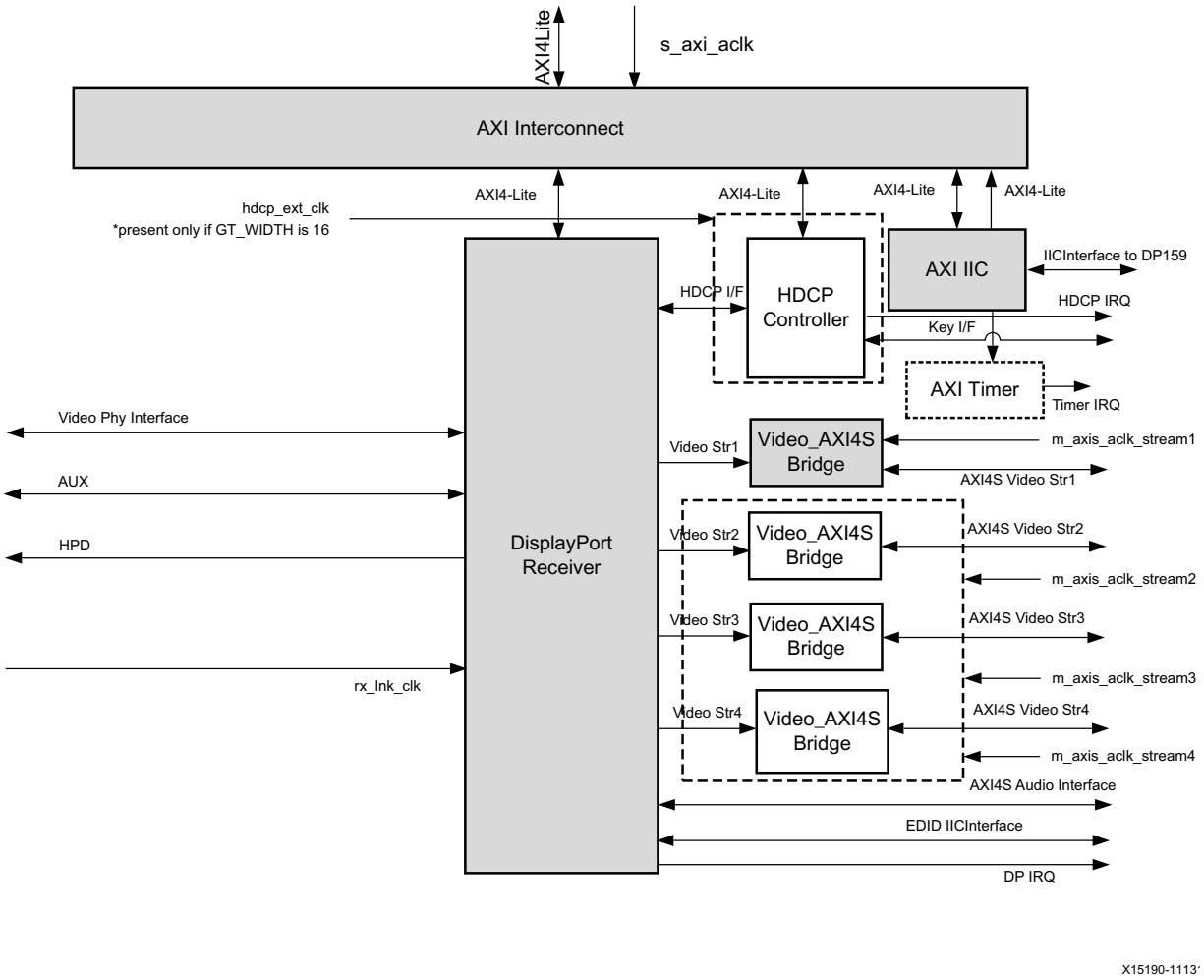

Figure 2-1: **DisplayPort RX Subsystem Block Diagram**

Native Video Interface

In the SST or MST mode, by default, the subsystem is packaged with two mandatory subcores: DisplayPort Receive core and AXI IIC controller. A HDCP core along with an AXI Timer core is also present as part of the DisplayPort RX Subsystem when the HDCP feature is enabled.

Because the DisplayPort RX Subsystem is hierarchically packaged, you select the parameters and the subsystem creates the required hardware. Figure 2-2 shows the architecture of the subsystem assuming MST with four streams.

The DisplayPort RX Subsystem receives the video using the DisplayPort v1.2 protocol over 32-bit or 16-bit video PHY interface. The DisplayPort RX Subsystem works in conjunction with the Video PHY Controller configured for DP protocol.The subsystem outputs multi-pixel Video to the AXI4-Stream Protocol interface.

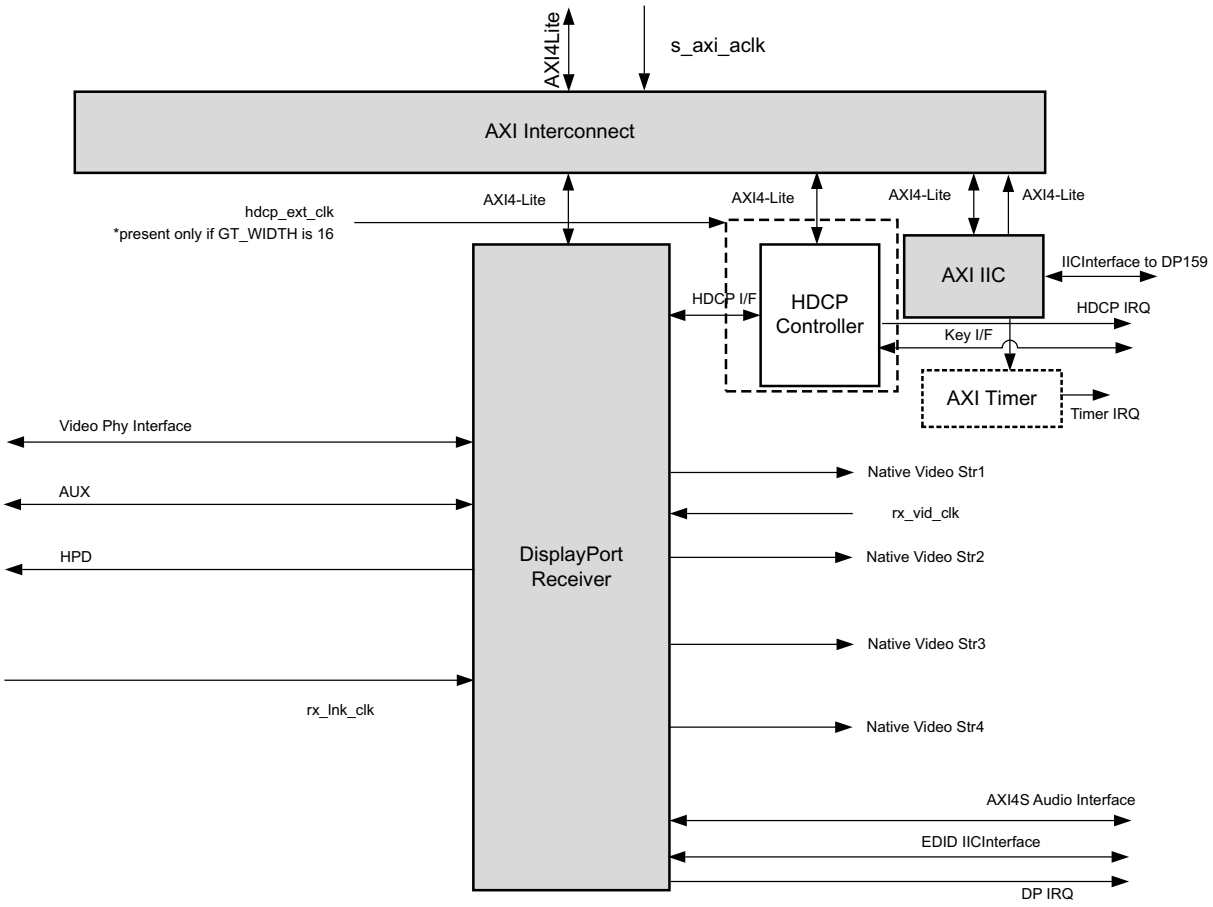

Figure 2-2: **DisplayPort RX Subsystem Block Diagram with Native Video Interface**

DisplayPort Receive (RX)

The DisplayPort Receive (RX) core contains the following four major blocks as shown in Figure 2-3:

- **Main Link**: Provides for the 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.

For more details, see the *DisplayPort Product Guide* (PG064) [Ref 1].

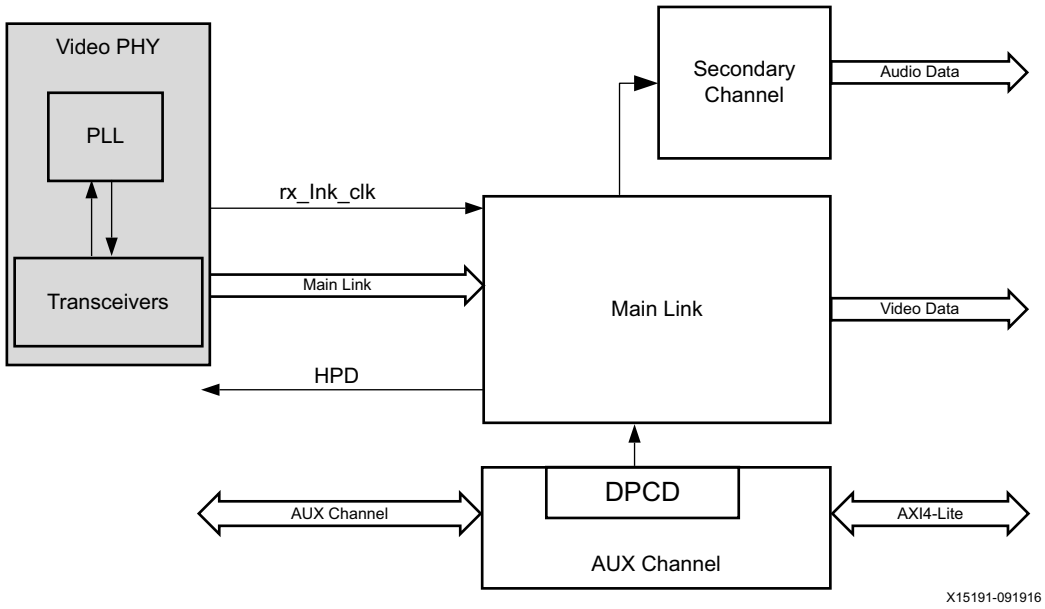

Figure 2-3: **DisplayPort Receive Core Block Diagram**

Video to AXI4-Stream Bridge

Video to AXI4 stream Bridge is used in DisplayPort RX Subsystem to convert the video output of DisplayPort receive IP to the AXI4 stream standard.

In MST mode, there are N number of bridges in the subsystem, where N = the number of AXI4-Stream outputs to the subsystem. For more details on video pixel mapping over Streaming interface, see the *AXI4-Stream Video IP and System Design Guide* (UG934) [Ref 2]. The *DisplayPort Product Guide* (PG064) [Ref 1] contains information about the DisplayPort output video format.

Pixel Mapping on Streaming Interface

By default, the pixel mode is equal to the lane count during subsystem generation.

```
Pixel_Width = MAX_BPC x 3
Interface Width = Pixel Width x LANE_COUNT
```

For example, if the system is generated using 4 lanes with `MAX_BPC` of 16, the data width will be 16x4x3 which equals to 192.

MAX_BPC	LANES	PIXEL WIDTH	INTERFACE	VIDEO BPC	Pixel 3			Pixel2			Pixel 1			Pixel 0		
					B	G	R	B	G	R	B	G	R	B	G	R
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
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
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
8	2	24	48	8							47:40	39:32	31:24	23:16	15:8	7:0

Figure 2-4: **Pixel Mapping Examples on Streaming Interface**

AXI Interconnect

The subsystem uses Xilinx AXI Interconnect IP core, as a crossbar, which contains one AXI4-Lite slave interface and two AXI4-Lite master interfaces, without HDCP enabled, in the system. With HDCP, the subsystem has four AXI4-Lite master interfaces.

Figure 2-5 shows the AXI slave structure within the DisplayPort RX Subsystem. For more details on the AXI crossbar functionality, see the *AXI Interconnect Product Guide* (PG059) [Ref 3].

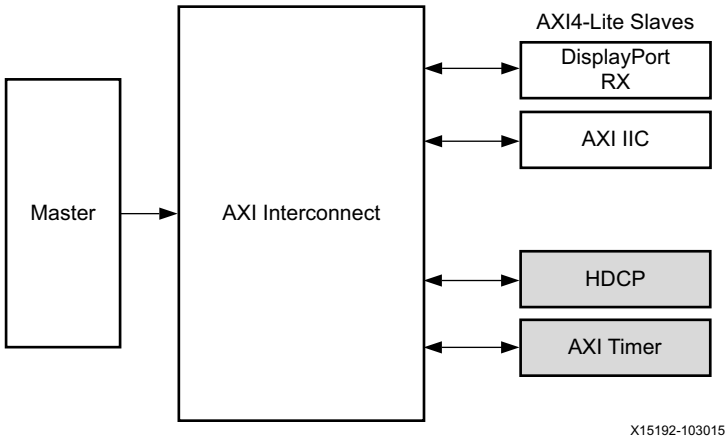

Figure 2-5: **AXI Interconnect in DisplayPort RX Subsystem**

AXI IIC

AXI IIC controller is used in DisplayPort RX Subsystem to configure the DP159 retimer through IIC interface. The AXI IIC controller in DisplayPort RX Subsystem is working at 400 KHz. DisplayPort RX Subsystem drivers handles the DP159 programming.

For more details on IIC programming for DP159, see the *DisplayPort Product Guide* (PG064) [Ref 1].

HDCP Controller

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

Figure 2-6 shows the DisplayPort RX Subsystem with HDCP controller. For more details on HDCP, see the *HDCP Product Guide* (PG224) [Ref 4].

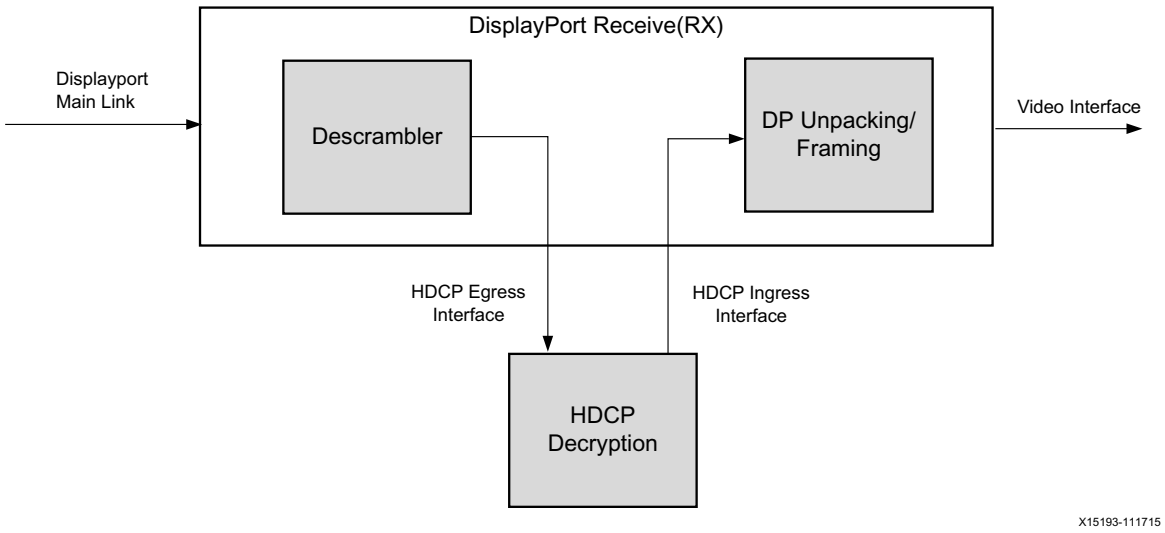

Figure 2-6: **DisplayPort RX with HDCP Controller**

AXI Timer

A 32-bit AXI Timer is used in DisplayPort RX Subsystem when the HDCP controller is enabled for decryption. The AXI Timer can be accessed through AXI4 master interface for basic timer functionality in the system.

Standards

The DisplayPort RX Subsystem is compatible with the DisplayPort v1.2 Standard, HDCP v1.3 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 Utilization

For details about Resource Utilization, visit Performance and Resource Utilization.

Port Descriptions

The DisplayPort RX Subsystem ports are described in Table 2-1.

Table 2-1: **DisplayPort RX Subsystem Ports**

Signal Name	Direction from Core	Description
AXI4-Lite Interface		
s_axi_aclk	Input	AXI Bus clock
s_axi_aresetn	Input	AXI reset. Active-Low.
s_axi_awaddr[13:0]	Input	Write address
s_axi_awprot[2:0]	Input	Protection Type
s_axi_awvalid	Input	Write address Valid
s_axi_awready	Output	Write address Ready
s_axi_wdata[31:0]	Input	Write data
s_axi_wstrb[3:0]	Input	Write Strobe
s_axi_wvalid	Input	Write data valid
s_axi_wready	Output	Write data ready
s_axi_bresp[1:0]	Output	Write response
s_axi_bvalid	Output	Write response valid
s_axi_bready	Input	Write response ready
s_axi_araddr[13:0]	Input	Read address
s_axi_arprot[2:0]	Input	Read protection type
s_axi_arvalid	Input	Read address valid
s_axi_arready	Output	Read address ready
s_axi_rdata[31:0]	Output	Read data
s_axi_rresp[1:0]	Output	Read data response
s_axi_rvalid	Output	Read data valid
s_axi_rready	Input	Read data ready
DP Video PHY Side Band Status		
s_axis_phy_rx_sb_status_tdata[15:0]	Input	Video PHY status input
s_axis_phy_rx_sb_status_tready	Output	Ready to video PHY for status
s_axis_phy_rx_sb_status_tvalid	Input	Video PHY status valid
DP Video PHY Side Band Control		
m_axis_phy_rx_sb_control_tdata[7:0]	Output	Control output to video PHY
m_axis_phy_rx_sb_control_tvalid	Output	Control output valid to video PHY
m_axis_phy_rx_sb_control_tready	Input	Control data ready input
DP Link Clock Interface		
rx_lnk_clk	Input	Link clock
DP Video PHY Main Link [Lane0 -Lane3]		
s_axis_lnk_rx_lane0_tdata[31:0]	Input	Main link data for lane0

Table 2-1: **DisplayPort RX Subsystem Ports *(Cont'd)***

Signal Name	Direction from Core	Description
s_axis_lnk_rx_lane0_tvalid	Input	Main link data valid for lane0
s_axis_lnk_rx_lane0_tready	Output	Main link data ready for lane0
s_axis_lnk_rx_lane0_tuser[11:0]	Input	Main link user data for lane0
s_axis_lnk_rx_lane1_tdata[31:0]	Input	Main link data for lane1
s_axis_lnk_rx_lane1_tvalid	Input	Main link data valid for lane1
s_axis_lnk_rx_lane1_tready	Output	Main link data ready for lane1
s_axis_lnk_rx_lane1_tuser[11:0]	Input	Main link user data for lane1
s_axis_lnk_rx_lane2_tdata[31:0]	Input	Main link data for lane2
s_axis_lnk_rx_lane2_tvalid	Input	Main link data valid for lane2
s_axis_lnk_rx_lane2_tready	Output	Main link data ready for lane2
s_axis_lnk_rx_lane2_tuser[11:0]	Input	Main link user data for lane2
s_axis_lnk_rx_lane3_tdata[31:0]	Input	Main link data for lane3
s_axis_lnk_rx_lane3_tvalid	Input	Main link data valid for lane3
s_axis_lnk_rx_lane3_tready	Output	Main link data ready for lane3
s_axis_lnk_rx_lane3_tuser[11:0]	Input	Main link user data for lane3
DP Receive Video Interface		
rx_vid_clk	Input	DisplayPort RX video clock
rx_vid_rst	Input	DisplayPort RX Video reset
DP RX SS Streaming Video Stream 1 Interface (Enabled when streaming interface is used)		
m_axis_aclk_stream1	Input	Stream1 Video clock input
m_axis_ video_stream1_tdata[191:0]	Output	Stream1 Video data
m_axis_video_stream1_tlast	Output	Stream1 Video last data, End of line pixel.
m_axis_video_stream1_tready	Input	Stream1 Video data read
m_axis_video_stream1_tuser	Output	Stream1 video user data
m_axis_video_stream1_tvalid	Output	Stream1 video data valid
DP RX SS Video Stream 2 Interface - MST		
m_axis_aclk_stream2	Input	Stream2 Video clock input
m_axis_ video_stream2_tdata[191:0]	Output	Stream2 Video data
m_axis_video_stream2_tlast	Output	Stream2 Video last data, End of line pixel.
m_axis_video_stream2_tready	Input	Stream2 Video data read
m_axis_video_stream2_tuser	Output	Stream2 video user data
m_axis_video_stream2_tvalid	Output	Stream2 video data valid

Table 2-1: **DisplayPort RX Subsystem Ports *(Cont'd)***

Signal Name	Direction from Core	Description
DP RX SS Video Stream 3 Interface		
m_axis_aclk_stream3	Input	Stream3 Video clock input
m_axis_ video_stream3_tdata[191:0]	Output	Stream3 Video data
m_axis_video_stream3_tlast	Output	Stream3 Video last data, End of line pixel.
m_axis_video_stream3_tready	Input	Stream3 Video data read
m_axis_video_stream3_tuser	Output	Stream3 video user data
m_axis_video_stream3_tvalid	Output	Stream3 video data valid
DP RX SS Video Stream 4 Interface - MST		
m_axis_aclk_stream4	Input	Stream4 Video clock input
m_axis_ video_stream4_tdata[191:0]	Output	Stream4 Video data
m_axis_video_stream4_tlast	Output	Stream4 Video last data, End of line pixel.
m_axis_video_stream4_tready	Input	Stream4 Video data read
m_axis_video_stream4_tuser	Output	Stream4 video user data
m_axis_video_stream4_tvalid	Output	Stream4 video data valid
DP RX SS Native Video Stream 1 Interface		
rx_vid_stream1_tx_vid_enable	Output	User data video enable.
rx_vid_stream1_tx_vid_hsync	Output	Horizontal sync pulse. Active on the rising edge.
rx_vid_stream1_tx_vid_oddeven	Output	Indicates an odd (1) or even (0) field polarity.
rx_vid_stream1_tx_vid_pixel0[47:0]	Output	Video data.
rx_vid_stream1_tx_vid_pixel1[47:0]	Output	Video data.
rx_vid_stream1_tx_vid_pixel2[47:0]	Output	Video data.
rx_vid_stream1_tx_vid_pixel3[47:0]	Output	Video data.
rx_vid_stream1_tx_vid_vsync	Output	Vertical sync pulse. Active on the rising edge.
MST Stream (n = stream number 2 to 4)		
rx_vid_stream*n*_tx_vid_enable	Output	User data video enable.
rx_vid_stream*n*_tx_vid_hsync	Output	Horizontal sync pulse. Active on the rising edge.
rx_vid_stream*n*_tx_vid_oddeven	Output	Odd/even field select. Indicates an odd (1) or even (0) field polarity.
rx_vid_stream*n*_tx_vid_pixel0[47:0]	Output	Video data.
rx_vid_stream*n*_tx_vid_pixel1[47:0]	Output	Video data.
rx_vid_stream*n*_tx_vid_pixel2[47:0]	Output	Video data.

Table 2-1: **DisplayPort RX Subsystem Ports *(Cont'd)***

Signal Name	Direction from Core	Description
rx_vid_stream*n*_tx_vid_pixel3[47:0]	Output	Video data.
rx_vid_stream*n*_tx_vid_vsync,	Output	Vertical sync pulse. Active on the rising edge.
AUX IO Interface - Internal Bidirectional IOB		
aux_rx_io_p	Inout	Bi-directional AUX IO- P
aux_rx_io_n	Inout	Bi-directional AUX IO- n
AUX IO Interface - Internal Unidirectional IOB		
aux_rx_channel_in_p	Input	Unidirectional AUX channel in - P
aux_rx_channel_in_n	Input	Unidirectional AUX channel in - n
aux_rx_channel_out_p	Output	Unidirectional AUX channel out - P
aux_rx_channel_out_n	Output	Unidirectional AUX channel out- n
AUX IP Interface - External IOB		
aux_rx_data_in	Input	External AUX data input
aux_rx_data_out	Output	External AUX data output
aux_rx_data_en_out_n	Output	External AUX data enable out. Active-Low.
HPD Interface		
rx_hpd	Output	HPD from DisplayPort RX
EDID IIC Interface		
edid_iic_sci_i	Input	EDID IIC SCL input
edid_iic_sci_o	Output	EDID IIC SCL output
edid_iic_sci_t	Output	EDID IIC SCL enable. IIC SCL enable is Active-Low.
edid_iic_sda_i	Input	EDID IIC SDA input
edid_iic_sda_o	Output	EDID IIC SDA output
edid_iic_sda_t	Output	EDID IIC SDA enable. IIC SDA enable is Active-Low.
DP159 Interface		
dp159_iic_sci_i	Input	DP159 IIC SCL input
dp159_iic_sci_o	Output	DP159 IIC SCL output
dp159_iic_sci_t	Output	DP159 IIC SCL enable
dp159_iic_sda_i	Input	DP159 IIC SDA input
dp159_iic_sda_o	Output	DP159 IIC SDA output
dp159_iic_sda_t	Output	DP159 IIC SDA enable
dp159_rst	Output	DP159 IIC reset through AXI IIC controller GPIO port0

Table 2-1: **DisplayPort RX Subsystem Ports *(Cont'd)***

Signal Name	Direction from Core	Description
HDCP External Clock (Enabled when HDCP is selected with 16-bit GT interface)		
hdcp_ext_clk	Input	HDCP external clock (enabled when HDCP is selected with 16-bit GT interface)
HDCP Key Interface		
hdcp_key_aclk	Input	Key clock
hdcp_key_aresetn	Input	Key Interface reset. Active-Low
hdcp_key_tdata[63:0]	Input	AXI4-Stream Key Tdata
hdcp_key_last	Input	AXI4-Stream Key Tlast
hdcp_key_tready	Output	AXI4-Stream Key Tready
hdcp_key_tuser[7:0]	Input	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	Input	AXI4-Stream Key TValid
reg_key_sel[2:0]	Output	Selects one of the eight sets of 40 keys.
Start_key_transmit	Output	Active-High pulse starts key transmit.
Interrupts		
dprxss_dp_irq	Output	DisplayPort RX IP interrupt out
dprxss_iic_irq	Output	AXI IIC IP interrupt out
dprx_hdcp_irq	Output	HDCP IP interrupt out
dprx_timer_irq	Output	AXI Timer interrupt out

Register Space

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

- DisplayPort Receive (RX) IP
- AXI IIC
- HDCP
- AXI Timer

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 maintained 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 Table 2-2 are to be considered reserved and returns 0 upon read. Only address offsets are listed in Table 2-2. Base addresses are configured by the AXI Interconnect.

Table 2-2: **DisplayPort Sink Core Configuration Space**

Offset	R/W	Definition
Receiver Core Configuration		
0x000	RW	LINK_ENABLE. Enable the receiver • 1 - Enables the receiver core. Asserts the HPD signal when set.
0x004	RW	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 0-6. Non-zero value in this field will issue 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: 8,16,24,32,40, and 48. As per the DisplayPort protocol specifications, AUX Pulse Width range = 0.4 to 0.6 µs. • [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 × 24 = 480)) Program a value of 24 in this register.
0x008	RW	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 stream4 • [2] - End of line reset disable to the MST video stream3 • [1] - End of line reset disable to the MST video stream2 • [0] - End of line reset disable to the SST video stream/ MST video stream1

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x00C	RW	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.
0x010	RW	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). User can override this by writing a new value to this register. Use quad pixel mode in MST. • [2:0] ◦ 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	RW	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. • [31] – Mask for Cable disconnect/unplug interrupt • [30] – CRC test start interrupt • [29] – Mask MST Act sequence received interrupt • [28] – Mask interrupt generated when DPCD registers 0x1C0, 0x1C1 and 0x1C2 are written for allocation/de-allocation/partial deletion • [27] – Audio packet FIFO overflow interrupt • [18] – Training pattern 3 start interrupt • [17] – Training pattern 2 start interrupt • [16] – Training pattern 1 start interrupt • [15] – Bandwidth change interrupt • [14] – TRAINING_DONE • [13] – DOWN_REQUEST_BUFFER_READY • [12] – DOWN_REPLY_BUFFER_READ • [11] – VC Payload Deallocated • [10] – VC Payload Allocated • [9] – EXT_PKT_RXD: Set to 1 when extension packet is received • [8] – INFO_PKT_RXD: Set to 1 when info packet is received • [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 • [4] – TRAINING_LOST: Training has been lost on any one of the active lanes • [3] – VERTICAL_BLANKING: Start of the vertical blanking interval • [2] – NO_VIDEO: The no-video condition has been detected after active video received • [1] – POWER_STATE: Power state change, DPCD register value 0x00600 • [0] – VIDEO_MODE_CHANGE: Resolution change, as detected from the MSA fields.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x018	RW	MISC_CONTROL. Allows the host to instruct the receiver to pass the MSA values through unfiltered. • [2] – When set to 1, I2C DEFERs will be sent as AUX DEFERs to the source device. • [1] – When set to 1, the long I2C write data transfers are responded to using DEFER instead of Partial ACKs. • [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.
0x01C	WO	SOFTWARE_RESET_REGISTER. • [8] – Soft reset control to external HDCP FIFOs. • [7] – AUX Soft Reset. When set, AUX logic will be reset. • [0] – Soft Video Reset: When set, video logic will be reset. Reads will return zeros.
AUX Channel Status		
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	RW	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 µs, 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 µs 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 will be limited to the seven LSBs.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
0x040	RC	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. • [31] – Cable disconnect/unplug interrupt • [30] – CRC test start interrupt • [29] – MST Act sequence received interrupt • [28] – interrupt generated when DPCD registers 0x1C0, 0x1C1, and 0x1C2 are written for allocation/de-allocation/partial deletion • [27] – Audio packet FIFO overflow interrupt. • [18] – Training pattern 3 start interrupt. • [17] – Training pattern 2 start interrupt. • [16] – Training pattern 1 start interrupt. • [15] – Bandwidth change interrupt. • [14] – TRAINING_DONE: Set to 1 when training is done. • [13] – DOWN_REQUEST_BUFFER_READY: set to 1 indicating availability of Down request. • [12] – DOWN_REPLY_BUFFER_READ: Set to 1 for a read event from Down Reply Buffer by upstream source. • [11] – VC Payload Deallocated: Set to 0 when de-allocation event occurs in controller. • [10] – VC Payload Allocated: Set to 1 when allocation event occurs in controller. • [9] – EXT_PKT_RXD: Set to 1 when extension packet is received. • [8] – INFO_PKT_RXD: Set to 1 when info packet is received. • [7] – Reserved. • [6] – VIDEO: Set to 1 when a valid video frame is detected on main link. • [5] – Reserved • [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. • [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. • [2] – NO_VIDEO: the receiver has detected the no-video flags in the VBID field after active video has been received. • [1] – POWER_STATE: The transmitter has requested a change in the current power state of the receiver core. • [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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x044	RW	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. • [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

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x048	RC	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 Case register of stream 1. Reserved bits return 0. See offset 0x040 for more description on each interrupt. • [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
0x050	RW	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. • [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. • [7:0] – HSYNC_PULSE_WIDTH: Specifies the number of clock cycles the horizontal sync pulse is asserted. The vid_hsync signal will be high for the specified number of clock cycles.
0x058	RW	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. • [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.
0x060	RW	[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 1.2.
0x064	RW	• [31] – Set to override Training Pattern 1 (TP1) score. • [14:0] – Training pattern1 (TP1) score to override.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x068	RW	• [31] – Set to override the Training Pattern2/3 scores. • [12:0] – Training pattern2/3 (TP2/TP3) score to override.
0x06C	RO	Provides the contents of DPCD registers 0x1C0, 0x1C1, and 0x1C2 • [21:16] – Time slot count • [13:8] – Starting Time Slot • [5:0] – VC Payload ID
0x074	RW	[5] – Enable the 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	RW	CRC for Red color
0x07C	RW	CRC for Green color
0x080	RW	CRC for Blue color
DPCD Fields		
0x084	RW	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	RW	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	RW	REMOTE_COMMAND. General byte for passing remote information to the transmitter. • [7:0] – Remote data byte.
0x090	RW	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	RW	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	RW	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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x09C	RW	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 DPCD address space. • [4:0] – Link rate override value for DisplayPort Standard v1.2a designs • [3:0] – Link rate override value for DisplayPort Standard v1.1a designs ◦ 0x6 - 1.62 Gb/s ◦ 0xA - 2.7 Gb/s ◦ 0x14 - 5.4 Gb/s
0x0A0	RW	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 DPCD address space. • [7] – ENHANCED_FRAME_CAP: Capability override • [6] – TPS3_SUPPORTED: Capability override for DisplayPort Standard v1.2a protocol designs only. Reserved for v1.1a protocol. • [4:0] – Lane count override value (1, 2, or 4 lanes).
0x0A4	RW	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 v1.2a protocol). • [7:6] – SYMBOL ERROR COUNT SEL Override • [5] – SCRAMBLING_DISABLE Override • [4] – RECOVERED_CLOCK_OUT_EN Override • [3:2] – LINK_QUAL_PATTERN_SET Override for DisplayPort Standard v1.1a only. • [1:0] – TRAINING_PATTERN_SELECT Override
0x0A8	RW	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	RW	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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x0B0	RW	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	RW	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 *	RW	Override DPCD Control Register. Setting this register to 0x1 enables AXI/APB write access to DPCD capability structure.
0x0BC	RW	Override DPCD DOWNSPREAD control field. Register 0x0B8 must be set to 1 to override DPCD values. • [0] – MAX_DOWNSPREAD Override
0x0C0	RW	Override DPCD LINK_QUAL_LANE0_SET field for DPCD1.2 version only. Register 0x0B8 must be set to 1 to override DPCD values. • [2:0] – LINK_QUAL_LANE0_SET override
0x0C4	RW	Override DPCD LINK_QUAL_LANE1_SET field for DPCD1.2 version only. Register 0x0B8 must be set to 1 to override DPCD values. • [2:0] – LINK_QUAL_LANE1_SET override
0x0C8	RW	Override DPCD LINK_QUAL_LANE2_SET field for DPCD1.2 version only. Register 0x0B8 must be set to 1 to override DPCD values. • [2:0] – LINK_QUAL_LANE2_SET override
0x0CC	RW	Override DPCD LINK_QUAL_LANE3_SET field for DPCD1.2 version only. Register 0x0B8 must be set to 1 to override DPCD values. • [2:0] – LINK_QUAL_LANE3_SET override
0x0D0	RW	• [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 configuration programming stage only.
0x0D4	RW	[6:0] – Sink device count: Recommended to be programmed during initialization of the Sink device. In SST mode, the value should be 1.
0x0E0	RW	GUID word 0. Allows you to set up GUID if required from host interface. Valid for DPCD1.2 version only. • [31:0] – Lower 4 bytes of GUID DPCD field

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x0E4	RW	GUID word 1. Allows you to set up GUID if required from host interface. Valid for DPCD1.2 version only. • [31:0] – Bytes 4 to 7 of GUID DPCD field
0x0E8	RW	GUID word 2. Allows you to set up GUID if required from host interface. Valid for DPCD1.2 version only. • [31:0] – Bytes 8 to 11 of GUID DPCD field
0x0EC	RW	GUID word 3. Allows you to set up GUID if required from host interface. Valid for DPCD1.2 version only. • [31:0] – Bytes 12 to 15 of GUID DPCD field
0x0F0	RW	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 DPCD1.2 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		
0x0FC	RO	CORE_ID. Returns the unique identification code of the core and the current revision level. • [31:24] – DisplayPort protocol major version • [23:16] – DisplayPort protocol minor version • [15:8] – DisplayPort protocol revision • [7:0] – Core mode of operation ◦ 0x00: Transmit ◦ 0x01: Receive Depending on the protocol and core used, the CORE_ID values are as follows: • DisplayPort Standard v1.1a, Receive core: 32'h01_01_0a_01 • DisplayPort Standard v1.2a, Receive core: 32'h01_02_0a_01

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x110	RO	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. • [11] – Video Timing FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the Video Timing FIFO is overflowed for stream4. • [10] – Video Timing FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the Video Timing FIFO is overflowed for stream3. • [9] – Video Timing FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the Video Timing FIFO is overflowed for stream2. • [8] – Video Timing FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the Video Timing FIFO is overflowed. • [7] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the Video unpack FIFO is overflowed for stream4. • [6] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the Video unpack FIFO is overflowed for stream3. • [5] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the Video unpack FIFO is overflowed for stream2. • [4] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the Video unpack FIFO is overflowed. • [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. • [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. • [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. • [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.
0x114	RO	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. • [3] – State of the vertical sync pulse for Stream 4. • [2] – State of the vertical sync pulse for Stream 3. • [1] – State of the vertical sync pulse for Stream 2. • [0] – State of the vertical sync pulse for Stream 1.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
PHY Configuration and Status		
0x208	RO	PHY_STATUS. Provides status for the receiver core PHY. • [31:30] – RX buffer status, lane 3. • [29:28] – RX buffer status, lane 2. • [27:26] – RX buffer status, lane 1. • [25:24] – RX buffer status, lane 0. • [23] – RXBYTEISALIGNED status of PHY, lane 3. • [22] – RXBYTEISALIGNED status of PHY, lane 2. • [21] – RXBYTEISALIGNED status of PHY, lane 1. • [20] – RXBYTEISALIGNED status of PHY, lane 0. • [15:14] – RX voltage low, lanes 2 and 3. • [13:12] – RX voltage low, lanes 0 and 1. • [11:10] – PRBS error, lanes 2 and 3. • [9:8] – PRBS error, lanes 0 and 1. • [7] – Receiver Clock locked. • [6] – FPGA fabric clock PLL locked. • [5] – PLL for lanes 2 and 3 locked (Tile 1). • [4] – PLL for lanes 0 and 1 locked (Tile 0). • [3:2] – Reset done for lanes 2 and 3 (Tile 1). • [1:0] – Reset done for lanes 0 and 1 (Tile 0).

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x214	RW	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 will force training to fail until this value is met. ***Note:*** It is not recommended to change this register value. • [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). ◦ 00: Default (pre-emphasis adjust request will get 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 • [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). ◦ 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 • [1:0] – The minimum voltage swing setting matches the values defined in the DisplayPort Standard for the TRAINING_LANEX_SET register.
0x21C	RW	CDR_CONTROL_CONFIG. • [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.
0x220	RW	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. • [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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
DisplayPort Audio		
12'h300	RW	RX_AUDIO_CONTROL. This register enables audio stream packets in main link. • [0] – Audio Enable
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: [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 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
12'h330 to 12'h350	RO	RX_AUDIO_EXT_DATA • [31:0] – Word formatted as per extension packet described in protocol standard. Packet length is fixed to 32 bytes in Sink controller. User 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. • 1st word - [31:24] = HB3 [23:16] = HB2 [15:8] = HB1 [7:0] = HB0 • 2nd word - DB3,DB2,DB1,DB0 . . • 9th word -DB31,DB30,DB29,DB28 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.
DPCD Configuration Space *(Refer to the DisplayPort 1.1a standard for detailed descriptions of these registers)*		
0x400	RO	DPCD_LINK_BW_SET. Link bandwidth setting. • [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.
0x404	RO	DPCD_LANE_COUNT_SET. Number of lanes enabled by the transmitter. • [4:0] – Contains the number of lanes that are currently enabled by the attached transmitter. Valid values fall in the range of 1-4.
0x408	RO	DPCD_ENHANCED_FRAME_EN. Indicates that the transmitter has enabled the enhanced framing symbol mode. • [0] – Set to 1 when enhanced framing mode is enabled.
0x40C	RO	DPCD_TRAINING_PATTERN_SET. Current value of the training pattern registers. • [1:0] – TRAINING_PATTERN_SET: Set the link training pattern according to the two bit code: ◦ 00 = Training not in progress ◦ 01 = Training pattern 1 ◦ 10 = Training pattern 2 ◦ 11 = RESERVED

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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: ◦ 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. • [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: ◦ 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 ◦ 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 ◦ 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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
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	DPCD_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 ◦ 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. • [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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x440	RO	DPCD_LANE23_STATUS. Value of the lane 2 and lane 3 training status registers. • [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.
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.
MSA Values		
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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x510	RO	MSA_HTOTAL. Tells the receiver core how many video clock cycles will 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] – The 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, and a '0' indicates an active-Low vertical sync.
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] – The 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: ◦ 00 = RGB ◦ 01 = YCbCr 4:2:2 ◦ 10 = YCbCr 4:4:4 ◦ 11 = Reserved • [0] – CLOCK_MODE: ◦ 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: These bits are always set to 0. • [2:1] – STEREO_VIDEO: Used only when stereo video sources are being transmitted. See the *DisplayPort Standard v1.1a* section 2.24 for more information. • [0] – INTERLACED_EVEN: 1 indicates that the number of lines per frame is an even number.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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 (p44) in the *DisplayPort Standard* for more information. The default value is 0x19.
MST MSA Values		
0x540	RO	MSA_HRES_STREAM2. The horizontal resolution detected in the Main Stream Attributes. • [15:0] – Represents the number of pixels in a line of video.
0x544	RO	MSA_HSPOL_STREAM2. Horizontal sync polarity. • [0] – Indicates the polarity of the horizontal sync as requested by the transmitter.
0x548	RO	MSA_HSWIDTH_STREAM2. Specifies the width of the horizontal sync pulse. • [14:0] – Specifies the width of the horizontal sync in terms of the recovered video clock.
0x54C	RO	MSA_HSTART_STREAM2. 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.
0x550	RO	MSA_HTOTAL_STREAM2. Tells the receiver core how many video clock cycles will occur between leading edges of the horizontal sync pulse. • [15:0] – Total number of video clocks in a line of data.
0x554	RO	MSA_VHEIGHT_STREAM2. Total number of active video lines in a frame of video. • [15:0] – The vertical resolution of the received video.
0x558	RO	MSA_VSPOL_STREAM2. Specifies the vertical sync polarity requested by the transmitter. • [0] – A value of 1 in this register indicates an active-High vertical sync, and a 0 indicates an active-Low vertical sync.
0x55C	RO	MSA_VSWIDTH_STREAM2. 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.
0x560	RO	MSA_VSTART_STREAM2. 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.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x564	RO	MSA_VTOTAL_STREAM2. Total number of lines between sequential leading edges of the vertical sync pulse. • [15:0] – The total number of lines per video frame is contained in this value.
0x568	RO	MSA_MISC0_STREAM2. 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: ◦ 00 = RGB ◦ 01 = YCbCr 4:2:2 ◦ 10 = YCbCr 4:4:4 ◦ 11 = Reserved • [0] – CLOCK_MODE: ◦ 0 = Synchronous clock mode ◦ 1 = Asynchronous clock mode
0x56C	RO	MSA_MISC1_STREAM2. 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: These bits are always set to 0. • [2:1] – STEREO_VIDEO: Used only when stereo video sources are being transmitted. See the *DisplayPort Standard v1.1a* section 2.24 for more information. • [0] – INTERLACED_EVEN: 1 indicates that the number of lines per frame is an even number.
0x570	RO	MSA_MVID_STREAM2. 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.
0x574	RO	MSA_NVID_STREAM2. 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.
0x578	RO	MSA_VBID_STREAM2. The most recently received VB-ID value is contained in this register. • [7:0] – VBID: See Table 2-3 (p44) in the *DisplayPort Standard* for more information. The default value is 0x19.
0x580	RO	MSA_HRES_STREAM3. The horizontal resolution detected in the Main Stream Attributes. • [15:0] – Represents the number of pixels in a line of video.
0x584	RO	MSA_HSPOL_STREAM3. Horizontal sync polarity. • [0] – Indicates the polarity of the horizontal sync as requested by the transmitter.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x588	RO	MSA_HSWIDTH_STREAM3. Specifies the width of the horizontal sync pulse. • [14:0] – Specifies the width of the horizontal sync in terms of the recovered video clock.
0x59C	RO	MSA_HSTART_STREAM3. 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.
0x590	RO	MSA_HTOTAL_STREAM3. Tells the receiver core how many video clock cycles will occur between leading edges of the horizontal sync pulse. • [15:0] – Total number of video clocks in a line of data.
0x594	RO	MSA_VHEIGHT_STREAM3. Total number of active video lines in a frame of video. • [15:0] – The vertical resolution of the received video.
0x598	RO	MSA_VSPOL_STREAM3. Specifies the vertical sync polarity requested by the transmitter. • [0] – A value of 1 in this register indicates an active-High vertical sync, and a '0' indicates an active-Low vertical sync.
0x59C	RO	MSA_VSWIDTH_STREAM3. 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.
0x5A0	RO	MSA_VSTART_STREAM3. 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.
0x5A4	RO	MSA_VTOTAL_STREAM3. Total number of lines between sequential leading edges of the vertical sync pulse. • [15:0] – The total number of lines per video frame is contained in this value.
0x5A8	RO	MSA_MISC0_STREAM3. 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: ◦ 00 = RGB ◦ 01 = YCbCr 4:2:2 ◦ 10 = YCbCr 4:4:4 ◦ 11 = Reserved • [0] – CLOCK_MODE: ◦ 0 = Synchronous clock mode ◦ 1 = Asynchronous clock mode

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x5AC	RO	MSA_MISC1_STREAM3. 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: These bits are always set to 0. • [2:1] – STEREO_VIDEO: Used only when stereo video sources are being transmitted. See the *DisplayPort Standard v1.1a* section 2.24 for more information. • [0] – INTERLACED_EVEN: 1 indicates that the number of lines per frame is an even number.
0x5B0	RO	MSA_MVID_STREAM3. 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.
0x5B4	RO	MSA_NVID_STREAM3. 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.
0x5B8	RO	MSA_VBID_STREAM3. The most recently received VB-ID value is contained in this register. • [7:0] – VBID: See Table 2-3 (p44) in the *DisplayPort Standard* for more information. The default value is 0x19.
0x5C0	RO	MSA_HRES_STREAM4. The horizontal resolution detected in the Main Stream Attributes. • [15:0] – Represents the number of pixels in a line of video.
0x5C4	RO	MSA_HSPOL_STREAM4. Horizontal sync polarity. • [0] – Indicates the polarity of the horizontal sync as requested by the transmitter.
0x5C8	RO	MSA_HSWIDTH_STREAM4. Specifies the width of the horizontal sync pulse. • [14:0] – Specifies the width of the horizontal sync in terms of the recovered video clock.
0x5CC	RO	MSA_HSTART_STREAM4. 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.
0x5D0	RO	MSA_HTOTAL_STREAM4. Tells the receiver core how many video clock cycles will occur between leading edges of the horizontal sync pulse. • [15:0] – Total number of video clocks in a line of data.
0x5D4	RO	MSA_VHEIGHT_STREAM4. Total number of active video lines in a frame of video. • [15:0] – The vertical resolution of the received video.
0x5D8	RO	MSA_VSPOL_STREAM4. Specifies the vertical sync polarity requested by the transmitter. • [0] – A value of 1 in this register indicates an active-High vertical sync, and a '0' indicates an active-Low vertical sync.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x5DC	RO	MSA_VSWIDTH_STREAM4. 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.
0x5E0	RO	MSA_VSTART_STREAM4. 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.
0x5E4	RO	MSA_VTOTAL_STREAM4. Total number of lines between sequential leading edges of the vertical sync pulse. • [15:0] – The total number of lines per video frame is contained in this value.
0x5E8	RO	MSA_MISC0_STREAM4. 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: ◦ 00 = RGB ◦ 01 = YCbCr 4:2:2 ◦ 10 = YCbCr 4:4:4 ◦ 11 = Reserved • [0] – CLOCK_MODE: ◦ 0 = Synchronous clock mode ◦ 1 = Asynchronous clock mode
0x5EC	RO	MSA_MISC1_STREAM4. 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: These bits are always set to 0. • [2:1] – STEREO_VIDEO: Used only when stereo video sources are being transmitted. See the *DisplayPort Standard v1.1a* section 2.24 for more information. • [0] – INTERLACED_EVEN: 1 indicates that the number of lines per frame is an even number.
0x5F0	RO	MSA_MVID_STREAM4. 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.
0x5F4	RO	MSA_NVID_STREAM4. 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.
0x5F8	RO	MSA_VBID_STREAM4. The most recently received VB-ID value is contained in this register. • [7:0] – VBID: See Table 2-3 (p44) in the *DisplayPort Standard* for more information. The default value is 0x19.

Table 2-2: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0xA00 to 0xAFF	RO	DOWN_REQUEST_BUFFER. Down Request Buffer address space. User has to read side band message request from the address 0xA00 – 0xA30. The rest of the address space is reserved.
0xB00 to 0xBFF	WO	DOWN_REPLY_BUFFER. Down Reply Buffer address space. User has to write side band message reply in the address starting from 0xB00 for every new reply. Reply Buffer can handle up to 32 bytes. The rest of the address space is reserved.
0xC00 to 0xCFF	RO	UPSTREAM_REQUEST_BUFFER. Reserved for future.
0xD00 to 0xDFF	WO	UPSTREAM_REPLY_BUFFER. Reserved for future.
0x800 to 0x8FF	RW	PAYLOAD_TABLE. This address space maps to the VC Payload table that is maintained in the core. Write access is provided when VCPayload table is in software control.
Vendor Specific DPCD		
0xE00 to 0xEFC	RW	SOURCE_DEVICE_SPECIFIC_FIELD. User access to Source specific field of DPCD address space. AXI accesses are all word-based (32 bits). • 0xE00 to 0xE02: Read Only (IEEE OUI Value Programmed by Source) • 0xE03 to 0xEFF: Write/Read
0xF00 to 0xFFC	RW	SINK_DEVICE_SPECIFIC_FIELD. User access to Sink specific field of DPCD address space. AXI accesses are all word-based (32 bits). • 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 Product Guide* (PG090) [Ref 5].

HDCP Registers

For details about the HDCP registers, see the *HDCP Product Guide* (PG224) [Ref 4].

AXI Timer Registers

For details about the AXI Timer registers, see the *AXI Timer Product Guide* (PG079) [Ref 6].

Designing with the Core

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

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 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 will 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 will detect 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 the user data interface.

DP159 Retimer

Xilinx expects the use of the TI DP159 Retimer along with the DisplayPort RX solution. DP159 as a retimer provides better SI features. As a retimer, the DP159 removes the random and ISI jitter from video source. The DP159 configuration is controlled through an I2C interface. The DisplayPort RX design needs an external I2C controller to configure the DP159 Retimer. For more details, refer the *SNx5DP159_Product_Preview* [Ref 7].

Note: IIC controller needs to work at 400 KHz or higher speed.

For details on reference clock requirements and its connectivity, see Clocking.

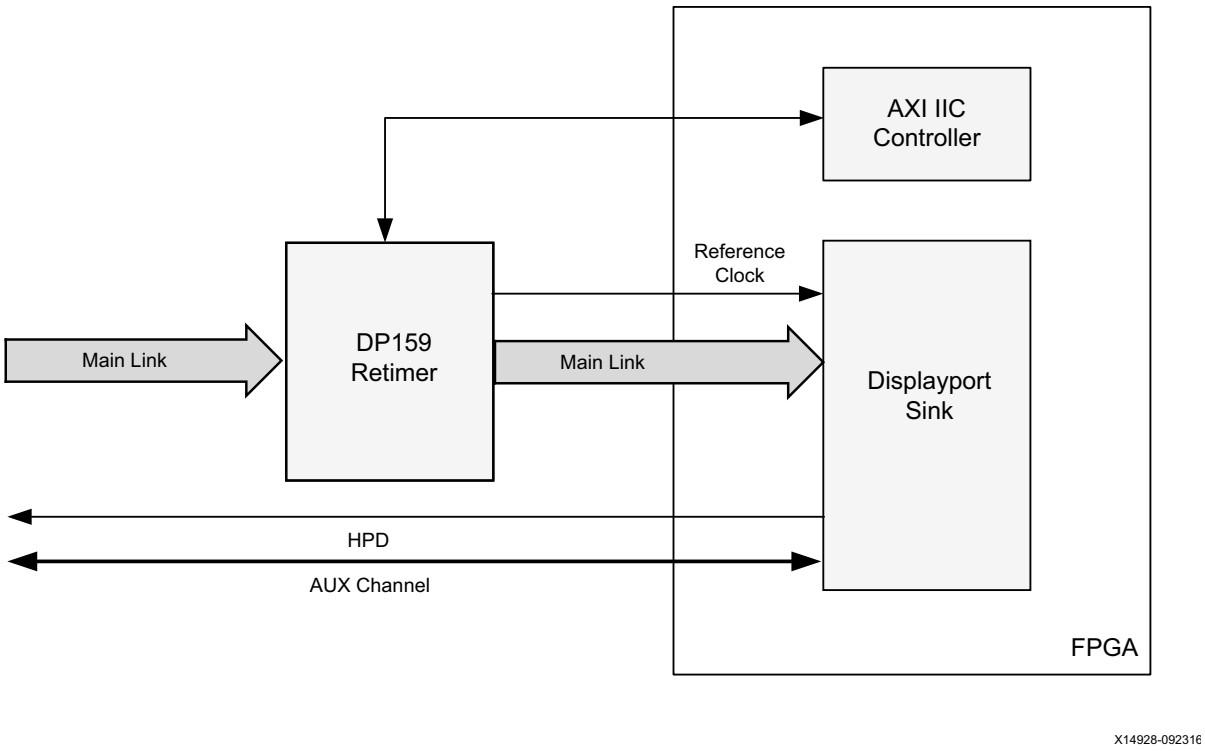

Figure 3-1: **DP159 Retimer**

DP159 Retimer Register Programming

The DP159 does not monitor the AUX transactions between DisplayPort source and DisplayPort sink during link training, as a typical DisplayPort retimer would. Because of this Display port Software driver handles the required DP159 configuration.

For more details on the programming sequence or the DP159 register details, see *DP159 as DisplayPort Retimer* [Ref 8].

Note: Users do not have to control the DP159 retimer as the DisplayPort Software Driver handles the programming.

The sequences of the DP159 retimer, handled in the DisplayPort driver, are summarized in the following sections:

- DP159 Initialization
- TP1 Interrupt Handler (TP1_Handler)
- TP23 Interrupt Handler (TP23_Handler)
- DP159 Re-Initialization

DP159 Initialization

The following registers must be programmed immediately after power-up to configure the DP159 PLL, TX and RX blocks.

Table 3-1: **DP159 Initialization**

IIC Address	Write/Read	Data	Description
0xFF	Write	0x00	Select Page 0
0x09	Write	0x36	Enable X-mode
0x0A	Write	0x7B	Disable HPD_SNK pass-through to HPD_SRC.
0x0D	Write	0x80	Enable Clock on AUX. Select 1/20 mode.
0x0C	Write	0x6D	Set TX Swing to Maximum
0x10	Write	0x00	Turn off pattern verifier
		0x11	Len = PRBS23, Sel = PRBS mode to turn off char-alignment
0x16	Write	0xF1	Disable char-alignment on all lanes
0xFF	Write	0x01	Select Page 1
Configure PLL			
0x00	Write	0x02	Enable Band Gap
0x04	Write	0x80	PLL_FBDIV[7:0]
0x05	Write	0x00	PLL_FBDIV[10:8]
0x08	Write	0x00	PLL_PREDIV[7:0]
0x0D	Write	0x02	Selects Lane0 for clock
0x0E	Write	0x03	CDR_CONFIG [4:0]. FIXED, LN0
0x01	Write	0x01	CP_EN is PLL mode
0x02	Write	0x3F	CP_Current is high
0x0B	Write	0x33	Loop filter to 8K
0xA1	Write	0x02	Override PLL settings
0xA4	Write	0x02	Override PLL settings
Configure TX Block			
0x10	Write	0xF0	Disable for all four TX lanes

Table 3-1: **DP159 Initialization *(Cont'd)***

IIC Address	Write/Read	Data	Description
0x11	Write	0x30	TX_RATE is Full Rate, TX_TERM = 75 to 150, TX_INVPAIR = None
0x14	Write	0x00	HDMI_TWPST1 is 0dB de-emphasis
0x12	Write	0x03	SLEW_CTRL is Normal, SWING is 600 mV.
0x13	Write	0xFF	FIR_UPD. Load TX settings
0x13	Write	0x00	
Configure RX Block			
0x30	Write	0XE0	Disable Receivers except lane0
0x32	Write	0x00	PD_RXINT
0x31	Write	0x00	RX_Rate is full
0x4D	Write	0x08	EQFTC = 0 and EQLEV = 8
0x4C	Write	0x01	Enable Fixed EQ
0x34	Write	0x01	Enable Offset Correction
0x32	Write	0xF0	Load RX settings
0x32	Write	0X00	
0x33	Write	0xF0	Load EQ settings.
0xFF	Write	0x00	Select Page0
0x0A	Write	0X3B	Enable HPD_SNK pass thru to HPD_SRC. Retimer
0xFF	Write	0x01	Select Page1

TP1 Interrupt Handler (TP1_Handler)

The GPU must inform the sink of the link bandwidth (LINK_BW_SET) and the number of lanes (LANE_COUNT_SET) before beginning the link training. At this stage, the software must program the PLL enable and number of RX lanes of DP159. Once the PLL lock has been achieved, the software must immediately transition the PLL mode of operation from PLL_MODE to PD_MODE. It is also important to enable the TX lanes, in this stage, so that the DisplayPort sink can start performing the clock recovery.

Table 3-2: **TP1 Interrupt Handler**

Address	Read/Write	Data	Description
Bandwidth and Number of Lanes			
0x00	Write	0x02	Enable Bandgap, DISABLE PLL, clear A_LOCK_OVR
0x01	Write	0x01	CP_EN = PLL (reference) mode
0x0B	Write	0x33	Set PLL control
0x02	Write	0x3F	Set CP_CURRENT

Table 3-2: **TP1 Interrupt Handler *(Cont'd)***

Address	Read/Write	Data	Description
0x30	Write	0xE1 0xC3 0x0F	Set RX Lane count Lane count 1 Lane count2 Lanecount4
0x00	Write	0x03	Enable Bandgap, Enable PLL, clear A_LOCK_OVR
0x4C	Write	0x01	Enable fixed EQ
0x4D	Write	0x08 0x18 0x28	Set EQFTC and EQLEV (fixed EQ) HBR2 HBR RBR
0x10	Write	0xE1 0xC3 0x0F	Enable TX lanes Lane count 1 Lane count2 Lanecount4
0x00	Write	0x23	Enable PLL and Bandgap, set A_LOCK_OVR
Determine the PLL lock of DP159. If achieved, change PLL mode based on the lane rate. Continue programming, after the DP159 PLL lock.			
0x02	Write	0x5F 0x27 0x1F	CP_CURRENT HBR2 HBR RBR
0x0B	Write	0x30	PLL loop filter 1K
0x01	Write	0x02	CP_EN is PD mode
0xFF	Write	0x00	Select Page0
0x16	Write	0x11 0x31 0xF1 0xF1	Set DP_TST_EN per #lanes, latch FIFO errors Lane Count1 Lane count 2 Lane count4 Set DP_TST_EN on all lanes to disable char-alignment
0x10	Write	0x00	Disable PV
0xFF	Write	0x01	Select Page1

TP23 Interrupt Handler (TP23_Handler)

Upon completing the clock recovery phase of link training, the source transitions to the channel equalization phase. Once in the channel equalization phase, software should enable adaptive equalization in DP159.

Table 3-3: **TP23 Interrupt Handler**

Address	Read/Write	Data	Description
0x4C	Write	0x03	Enable Adaptive Equalization
0xFF	Write	0x00	Select Page0
0x15	Write	0x18	Clear BERT counters and TST_INTQ latches
0x18	Read		BERT counters [7:0] read and error counters increment
0x19	Read		BERT counters[11:8] read and error counters increment
0xFF	Write	0x01	Select Page1

DP159 Re-Initialization

The PLL, RX, and TX settings of the DP159 must be re initialized while the DisplayPort Source is no longer detected. For example, on receiving the cable unplug interrupt event.

Table 3-4: **DP159 Re-Initialization**

Address	Write/Read	Data	Description
0x00	Write	0x02	Disable PLL, clear A_LOCK_OVR
0x34	Write	0x01	Enable Offset Correction
0x02	Write	0x3F	Set CP_CURRENT is high BW
0x01	Write	0x01	CP_EN is PLL mode
0x0B	Write	0x33	PLL Loop filter 8K
0x4D	Write	0x08	EQFTC = 0 and EQLEV = 8
0x4C	Write	0x01	Set to Fixed EQ
0x33	Write	0xF0	Load Equalization settings
0x10	Write	0xF0	Disable all TX lanes
0x30	Write	0xE0	Enable RX Lane 0 only

DisplayPort RX Programming Sequence with DP159

The sequence for programming DisplayPort RX with DP159 is summarized in the following steps:

1. Reset the DP159.
2. Initialize interrupt handler in the system.
3. Initialize IIC controller. The IIC controller is used to program the DP159.
4. Remove reset over the DP159.
5. Initialize the DP159. For details, see the DP159 Initialization.

6. Initialize DisplayPort RX. The initialization sequence is handled by the DisplayPort RX driver.
    a. Disable the main link.
    b. Apply GT (CPLL&PHY) reset.
    c. Set the AUX clock divider (number of AXI clocks for the 1 MHz AUX clock generation).
    d. Set the RX Voltage Swing and pre-emphasis training setting (0x1445 in offset 0x214). Training requests Vswing start at level1 and fixed pre-emphasis at level1.
    e. Configure the tDLOCK period to 10 µs (Offset 0x21C).
    f. Remove the DisplayPort RX out of GT reset.
    g. Wait for PHY ready (CPLL lock and rest done status).
    h. Enable the DisplayPort receiver link.
    i. Enable the DTG.
    j. Apply and remove the soft reset.
7. DisplayPort RX receives the training pattern 1 (TP1) as the source initiates the training sequence after reading the RX capabilities. DisplayPort RX generates TP1 start interrupt.
8. TP1 interrupt handler. For details, see the TP1 Interrupt Handler (TP1_Handler).
    a. After the completion of DP159 TP1 programming, power down the unused lanes of DisplayPort based on the lane count.
    b. Apply GT (CPLL&PHY) reset.
    c. Follow the DisplayPort reset sequencing and remove the reset.
9. DisplayPort RX receives training pattern 2 (TP2)/ training pattern 3 (TP3), as the source initiates the training sequence based on RX DPCD capabilities once the clock recovery is complete. DisplayPort RX generates TP2/TP3 start interrupt.
10. TP23 interrupt handler. For details, see the TP23 Interrupt Handler (TP23_Handler).
11. Monitor the training done or the training lost interrupts for the training status.
12. In case of a cable unplug, re-initialize the DP159 by following the re-initialize sequence details provided in DP159 Re-Initialization.

A graphical representation of the DisplayPort RX programming sequence with DP159 is shown in Figure 3-2.

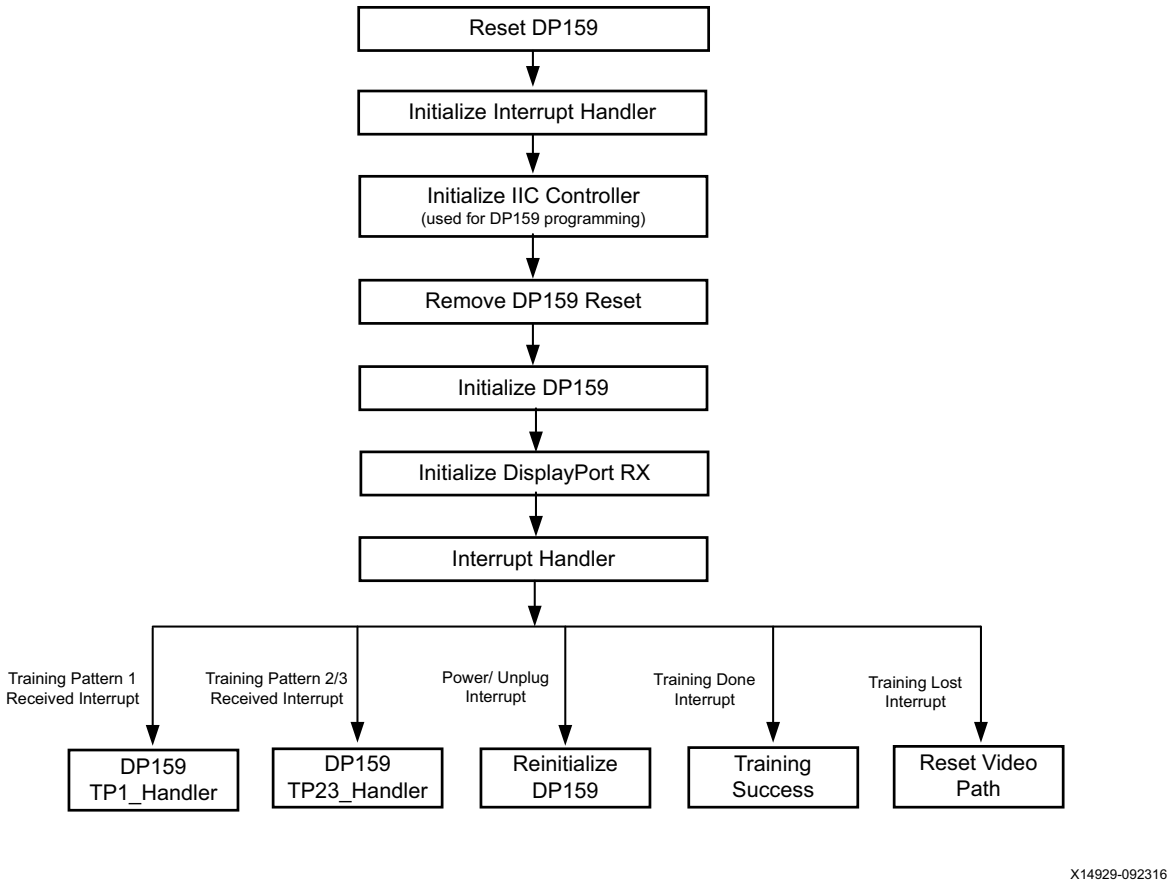

Figure 3-2: **DisplayPort RX Programming Sequence with DP159**

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 section 3.5.1.3 of the *DisplayPort Standard v1.2a* [Ref 9].

The Main Link for the Sink core drives a stream of video data toward the user. 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.

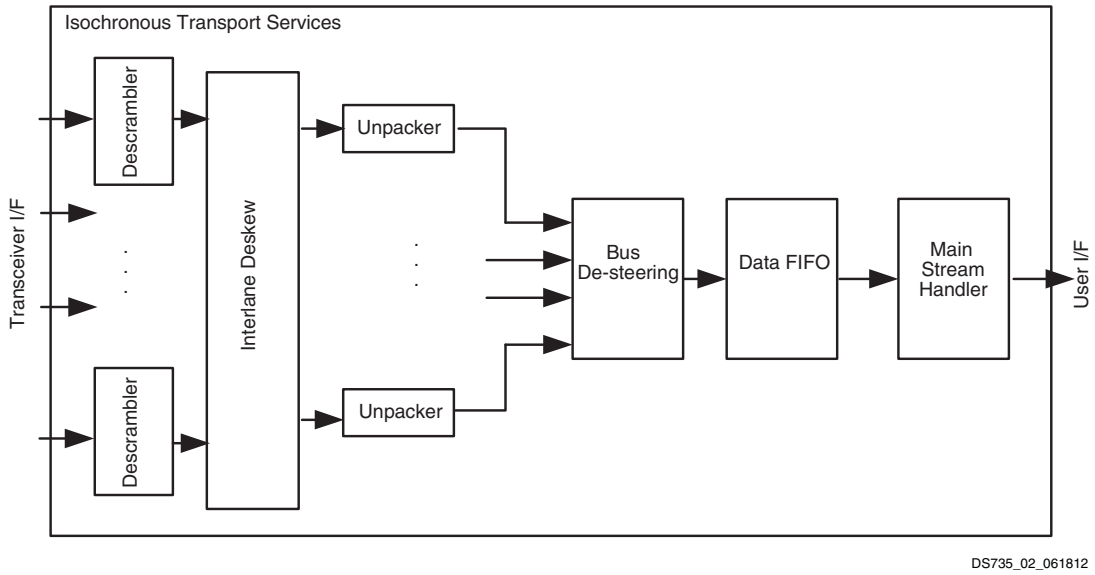

Figure 3-3: **Sink Main Link Datapath**

Figure 3-4 shows the flow diagram for link training.

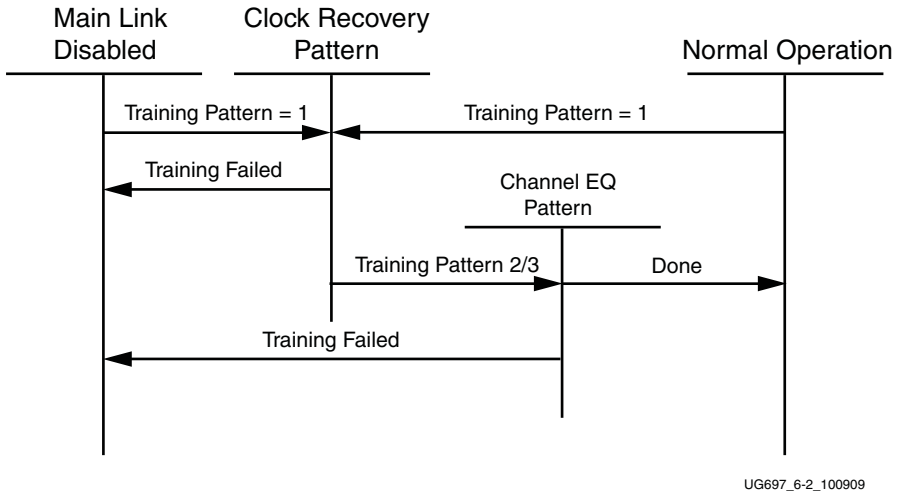

Figure 3-4: **Link Training States**

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 receiver 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 Receiver core places this information on dedicated signals and provides an update flag to signal a change in these values. Figure 3-5 shows how to use the M and N values from the core to generate a clock. See section 2.2.3 of the *DisplayPort Standard v1.2a* [Ref 9] for more details.

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. See section 2.2.3 of the DisplayPort Standard v1.2a [Ref 9] for more details.*

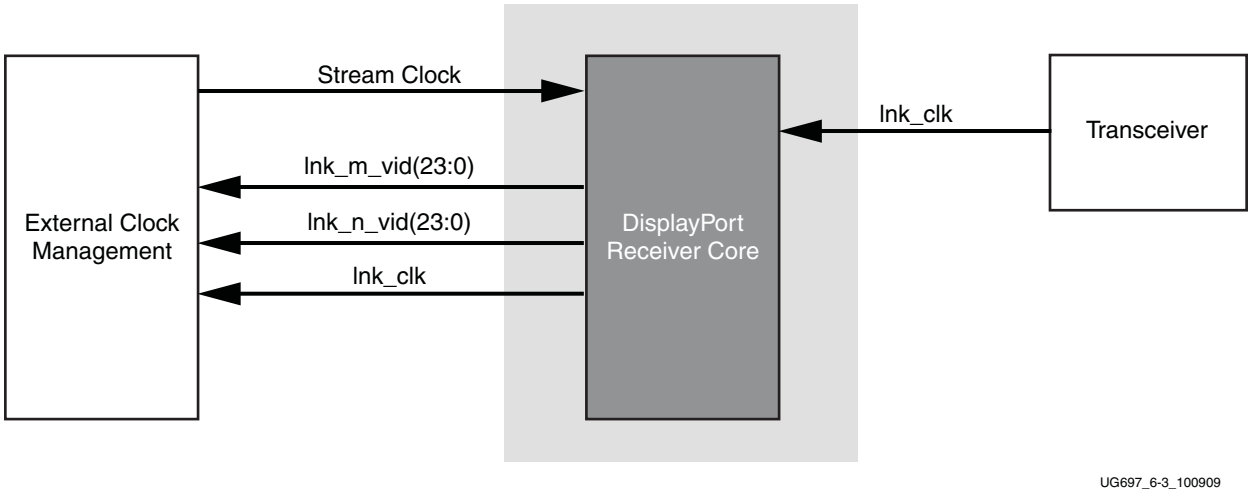

Figure 3-5: **Receiver Clock Generation**

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. The user must either read the new MSA values from register space or use the dedicated ports provided on the Main Link in order 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 will occur. At that point, video data and MSA values should not be relied on.

Once 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

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 Figure 3-6, 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.

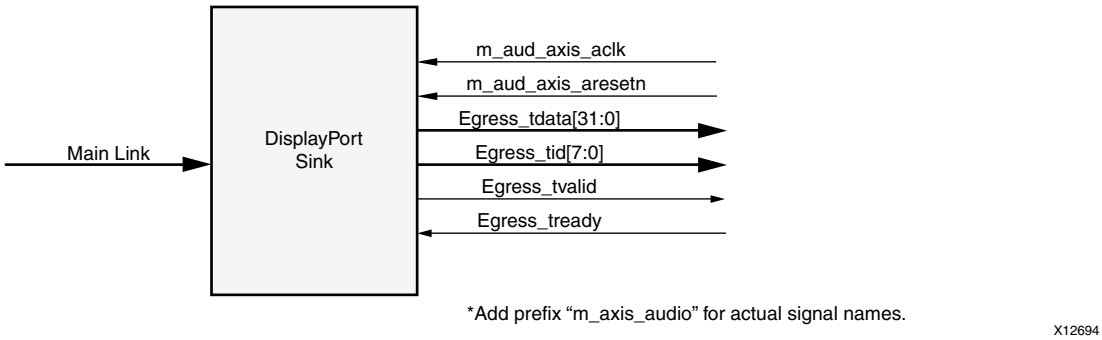

Figure 3-6: **Audio Data Interface of DisplayPort Sink System**

Multi Channel Audio

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

Audio Management

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

Programming DisplayPort Sink

1. Disable Audio by writing 0x00 to RX_AUDIO_CONTROL register. The disable bit also flushes the buffers in DisplayPort Sink. When there is a change in video/audio parameters, Xilinx recommends following this step.
2. Enable Audio by writing 0x01 to RX_AUDIO_CONTROL register.
3. For reading 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 & 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 µs) or wait until MUTE bit is removed.
4. Enable Audio in DisplayPort Receiver.

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 (Recommended)

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 × fs (Audio Sample Rate) clock frequency based on MAUD and NAUD values. External clock source is preferred for better precision.

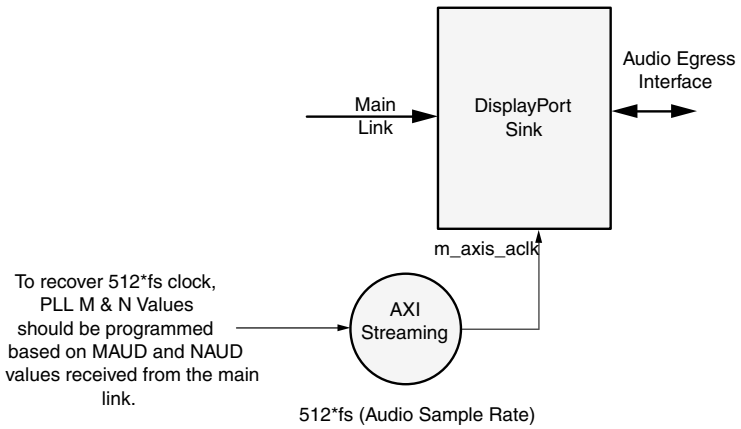

Figure 3-7: **Sink: Audio Clocking**

Sampling Frequencies

The DisplayPort RX Subsystem with a GT data width of 16-bit mode supports up to 8-channels of audio with maximum supported sampling frequency of 192 KHz for all link rates.

The DisplayPort RX Subsystem with a GT data width of 32-bit has limitations in the maximum supported sampling frequency rate depending on link rates. This limitation is due to the `lnk_clk` frequency reduction in 32-bit GT data width mode. Table 3-5 shows the maximum sampling rates with support up to eight channels.

Table 3-5: **Maximum Sampling Frequencies**

Link Rate (Gb/s)	Sampling Frequency (KHz)
5.4	192.0
2.7	176.4
1.62	96.0

Programming the Core in MST Mode

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

Enabling MST

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

Perform the following steps to program MST AUX Messaging:

1. Wait for DOWN_REQUEST_BUFFER_READY status in interrupt, and read from DOWN_REQUEST_BUFFER. Continue to collect side-band messages as per *DisplayPort Standard v1.2a Section 2.1.11.9*. 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, Sink HW updates the VC Payload Table by monitoring the AUX transactions. Alternatively, if you are an advanced user, you can use a software control to the VC Payload

Table (Setting the bit 1 in 0x0D0 enables it), with which 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. Once the software finishes writing to the VC payload table, software has to set bit 4 in 0x0D0.

Interrupt

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 DisplayPort Standard v1.2a specification. The LINK_ADDRESS reply helps the source to identify the topology of the sink.*
For example, if the sink core is configured for 4 MST streams, it receives the multi-stream input from the DisplayPort TX and outputs four individual streams in native video format. In this case, the LINK_ADDRESS_REPLY can be modeled to contain 1 input and 4 output logical ports, with their DisplayPort device plug status set as 1 and Peer Device Type set as 3.

Reduced Blanking

DisplayPort IP supports CVT standard RB and RB2 reduced blanking resolutions. As per the CVT specifications RB/RB2 resolution has HBLANK ≤ 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 receiver). 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.

DisplayPort transmitter knows the resolution ahead of time hence reset disable can be done during initialization. In DisplayPort receiver when 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.

Clocking

This section describes the link clock (`rx_lnk_clk`), video clock (`rx_vid_clk`) and video bridge AXI4-Stream master interface clock. In the MST mode, single `rx_vid_clk` connects to all the stream video interfaces. For information on other clocks, see the *DisplayPort Product Guide* (PG064) [Ref 1].

`rx_vid_clk` should be 150 MHz or higher. `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 RX Subsystem generated by the Video PHY (GT). The frequency of rx_lnk_clk is <line_rate>/40 MHz for the 32-bit video PHY(GT) data interface.

In 16-bit GT interface hdcp_ext_clk has to be provided by the user from external MMCM. The frequency requirement of hdcp_ext_clk is rx_lnk_clk/2.

Table 3-6 shows the clock ranges.

Table 3-6: **Clock Ranges**

Clock Domain	Min (MHz)	Max (MHz)	Description
rx_lnk_clk	40	270	Link clock
rx_vid_clk	150	200	Video clock
s_axi_aclk	25	135	Host processor clock

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. For MST with four streams, there are four video resets.
- dp159_rst: Active-High soft reset to the DP159 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 Controller Product Guide* (PG090) [Ref 5].

Address Map Example

Table 3-7 shows an example based on a subsystem base address of 0x44C0_0000 (14 bits). There are no registers in Video to AXI4-Stream bridge. ’

Table 3-7: **Address Map Example**

Name	SST	MST
DisplayPort RX	0x44C0_0000	0x44C0_0000
AXI IIC Controller	0x44C1_0000	0x44C1_0000
HDCP Controller	0x44C2_0000	0x44C2_0000
AXI Timer	0x44C3_0000	0x44C3_0000

Programming Sequence

For PHY related programming, see the *Video PHY Controller Product Guide* (PG230) [Ref 10].

For programming sequence of SST/MST modes and audio, see the *DisplayPort Product Guide* (PG064) [Ref 1].

For HDCP related programming sequence, see the *HDCP Controller Product Guide* (PG224) [Ref 4].

Design Flow Steps

This chapter describes customizing and generating the subsystem. More detailed information about the standard Vivado® design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 11]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 12]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 13]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 14]

Customizing and Generating the Subsystem

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

If you are customizing and generating the subsystem in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 11] 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 subsystem by specifying values for the various parameters associated with the subsystem IP cores using the following steps:

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

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 12] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 13].

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

Customizing the IP

The configuration screen is shown in Figure 4-1.

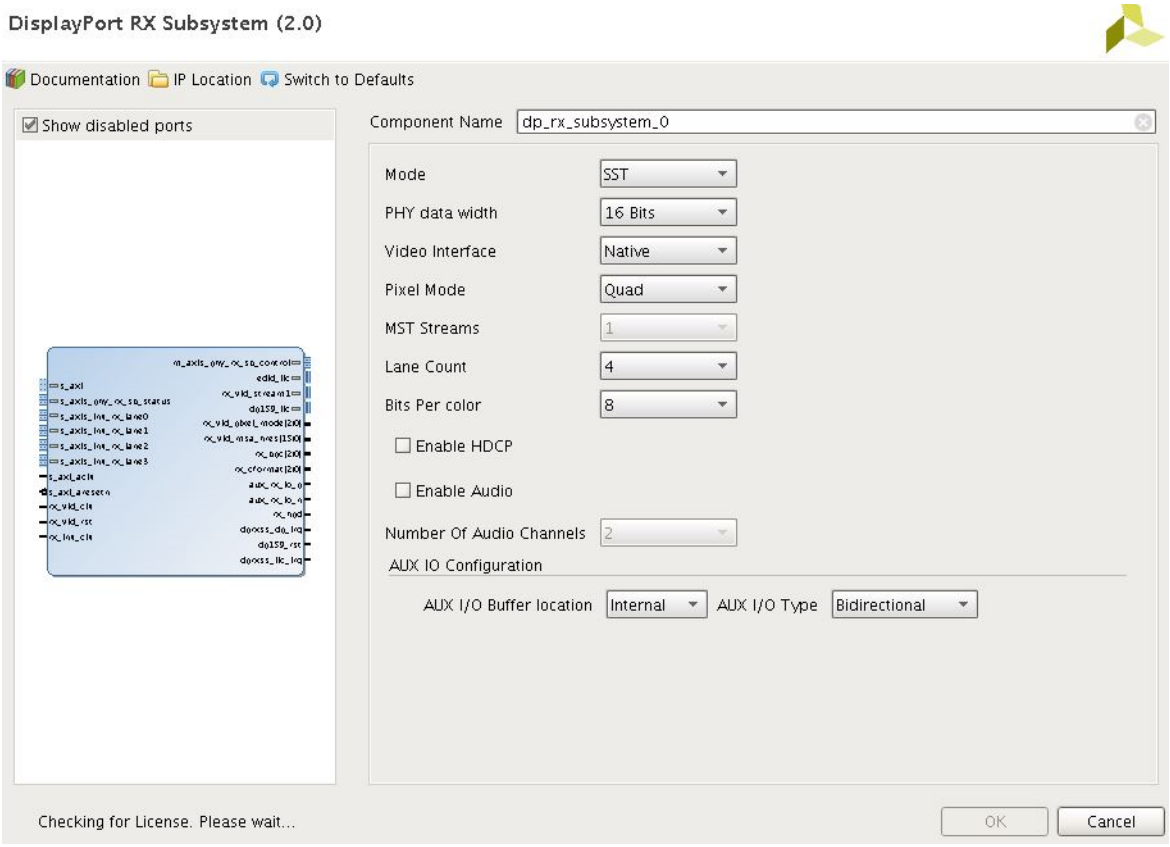

Figure 4-1: **Configuration Screen**

- **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 dp_rx_subsystem_0.
- **Mode**: Select the desired resolution for the DisplayPort IP. The default value is SST.
- **PHY Data Width**: Select 16-bit or 32-bit GT data width.
- **Video Interface**: Select streaming or native input video interface.
- **Pixel Mode**: Enabled when native interface is selected. Select single, dual or quad pixel mode.
- **MST Streams**: Select the number of streams in MST mode.
- **Lane Count**: Select the number of lanes.
- **Bits Per Color**: Select the desired bit per component (BPC).

- **Enable HDCP**: Enables HDCP.
- **Enable Audio**: Enables audio support.
- **Number of Audio Channels**: Select the number of audio channels.
- **AUX I/O Buffer location**: Select buffer location for AUX channel
- **AUX I/O Type**: Selection of Bi-Directional or Uni-directional buffer type.

User Parameters

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

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

Vivado IDE Parameter/Value	User Parameter/Value	Default Value
Mode	MODE	SST
PHY Data Width	PHY_DATA_WIDTH	16
Video Interface	VIDEO_INTERFACE	AXI4-Stream
Pixel Mode	PIXEL_MODE	Quad
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
AUX IO Buffer Location	AUX_IO_LOC	Internal
AUX IO Type	AUX_IO_TYPE	Bidirectional

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 12].

Constraining the Core

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

Required Constraints

There are no required constraints for this core. Being a subsystem, all sub-cores generate their own constraints and the same is applied in the subsystem.

Device, Package, and Speed Grade Selections

See IP Facts for details about supported devices.

Clock Frequencies

See Clocking in Chapter 3 for more details about clock frequencies.

Clock Management

There are no specific clock management constraints.

Clock Placement

There are no specific clock placement constraints.

Banking

There are no specific banking constraints.

Transceiver Placement

Transceiver is external to DisplayPort RX Subsystem hence there are no specific transceiver placement constraints.

I/O Standard and Placement

For details on the specific I/O constraints, see the *DisplayPort Product Guide* (PG064) [Ref 1].

Simulation

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

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 12].

Debugging

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

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

Finding Help on Xilinx.com

To help in the design and debug process when using the DisplayPort 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 DisplayPort 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 core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as

- Product name

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

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

Master Answer Record for the DisplayPort Subsystem

AR: 65447

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 DisplayPort Subsystem design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

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

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

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

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 16].

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. For details on debug steps for common hardware issues, see the *DisplayPort Product Guide* (PG064) [Ref 1].

Receive – Training

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

- Try with a different source such as the DisplayPort Analyzer.
- Change the cable and check again.
- Put an AUX Analyzer in the receive path and check if the various training stages match with the one's mentioned in DisplayPort Overview in Chapter 3.
- Probe the `lnk_clk` output and check the SI of the Clock is within the Phase Noise mask of the respective GT Transceiver.
- 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.
- Check the 0x43C and 0x440 registers for Symbol_Locked, Channel Equalization and Clock Recovery Done.

Receive – DP159 Related Issues

This section contains debugging steps for issues related to DP159. Proper operation of DP159 is essential for the training to complete successfully.

- IIC checks:
  - Check if the IIC speed is 400 KHz or higher speed (1 MHz).
  - Check if the IIC writes are happening properly to the DP159 IC
  - Check if the IIC writes are interrupt or polling based. If it is interrupt based, it would be like calling an interrupt within another interrupt routine. Make sure this function correct, or better to go with polling mode, as DP159-IIC writes are supposed to happen at DP training events
- Until the training is done make sure 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.

- The IP assumes that the DP159 forwarded clock is connected to MGTREFCLK1. For UltraScale the fixed clock is connected to MGTREFCLK0. Ensure that your hardware is wired accordingly.
- 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 DP159 configuration as the configuration must be completed as quickly as possible in order 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.

- If the video timing counters are reporting 0 lines, toggle the DTG enable and software-video reset and check again.
- 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.
- 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 in Chapter 3 have been followed correctly.

Receive – Sink MST

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

- Check if the GPU connected is recognizing the streams properly. Read the MSA of all the streams and verify against the GPU data.
- Read the VC Payload table through 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 could 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. Only in 5.4 x 4, four streams of 1080p will be possible. With HBR, only two 1080p streams will be possible. The link rate downshift could be because of training failure—make sure that a DisplayPort v1.2a cable is used.
- Make sure a DisplayPort v1.2a cable is used with DP159 in between.

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 200 MHz.

Application Software Development

The software is capable of detecting an MST/SST RX connected to the subsystem based on if a MST or SST software flow is executed. Figure B-1 shows the DisplayPort RX Subsystem application software flow for the SST mode.

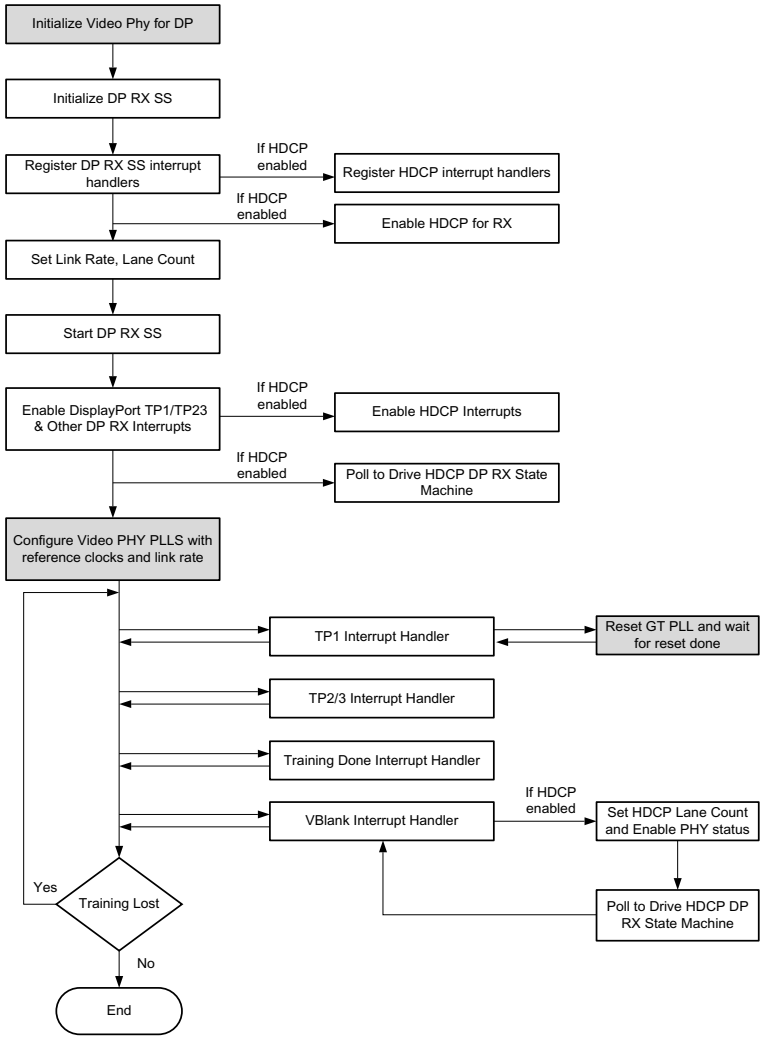

Figure B-1: **DisplayPort RX Subsystem Software flow for SST mode**

Note: Video PHY is external to the DisplayPort RX Subsystem and must be configured for the subsystem to work as expected. For more details on Video PHY configuration, see the *Video PHY Product Guide* (PG230) [Ref 10].

Appendix C

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 provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open the Xilinx Documentation Navigator (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 the Xilinx Documentation Navigator, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

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

References

These documents provide supplemental material useful with this product guide:

1. *DisplayPort Product Guide* (PG064)
2. *AXI4-Stream Video IP and System Design Guide* (UG934)
3. *AXI Interconnect Product Guide* (PG059)
4. *HDCP Controller Product Guide* (PG224)
5. *AXI IIC Bus Interface Product Guide* (PG090)
6. *AXI Timer Product Guide* (PG079)
7. *SNx5DP159_Product_Preview*
8. *DP159 as a DisplayPort Retimer* (SLLA358)
9. VESA *DisplayPort Standard v1.2a*, December 22, 2009
10. *Video PHY Controller Product Guide* (PG230)
11. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
12. *Vivado Design Suite User Guide: Designing with IP* (UG896)
13. *Vivado Design Suite User Guide: Getting Started* (UG910)
14. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
15. *ISE to Vivado Design Suite Migration Guide* (UG911)
16. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
17. *Vivado Design Suite User Guide: Implementation* (UG904)
18. *AXI Reference Guide* (UG1037)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/14/2017	2.0	Updated 7 series (GTHE2) and Artix-7 support in IP Facts.
06/07/2017	2.0	Vivado Design Suite release for DisplayPort RX v2.0.
04/05/2017	2.0	• Updated Supported Device Family in IP Facts table. • Added In-band stereo in Unsupported Features section. • Added 0x01C, Bit[8] and 0x21C, Bit[30] description to DisplayPort Sink Core Configuration Space table in Product Specification chapter. • Added DisplayPort Registers Sink Core in Product Specification chapter. • Added DisplayPort Overview in Designing with the Core chapter. • Updated rx_vid_clk clock min/max range in Clock Ranges table in Designing with the Core chapter. • Added Reduced Blanking in Designing with the Core chapter. • Updated Hardware Debug section in Debug Appendix.
12/20/2016	2.0	Added HDCP note for Supported Device Family in IP Facts table.
11/30/2016	2.0	Added Important note in Standards section.
10/05/2016	2.0	Updated HDCP features.
04/06/2016	2.0	Added support for 16 bit GT interface and native with pixel mode.
11/18/2015	1.0	Initial Xilinx release.

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.