DisplayPort v6.0

LogiCORE IP Product Guide

Vivado Design Suite

PG064 April 1, 2015

XILINX
ALL PROGRAMMABLE™

Table of Contents

Introduction

The Xilinx LogiCORE™ IP DisplayPort™ interconnect protocol is designed for transmission and reception of serial-digital video for consumer and professional displays. DisplayPort is a high-speed serial interface standard supported by PC chipsets, GPU and display controllers, HDTV and monitors from industry leaders.

This protocol replaces VGA and DVI. It is complimentary to HDMI™ outside and LVDS inside the box for higher resolution, higher frame rate and color bit depth display.

Features

- Source (TX) and Sink (RX) Controllers.
- Designed to *VESA DisplayPort Standard* v1.1a and v1.2.
- 1, 2 or 4 lanes at 1.62, 2.7 or 5.4 Gb/s.
- One, two or four pixel-wide video interface supporting up to a 4k x 2k monitor resolution.
- RGB and YCbCr color space, up to 16 bits per color.
- Auto lane rate and width negotiation.
- I2C over a 1 Mb/s AUX channel.
- Secondary channel audio support (1-8 channels).
- Supports four independent video multi-streams for Source and Sink controllers.
- Supports EDID and DPCD register space in Sink Controller.

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale™ Architecture, Zynq®-7000, 7 Series
Supported User Interfaces	Native Video, AXI4-Stream, AXI4-Lite
Resource Usage	See Resource Utilization.
Provided with Core	
Example Design	Simple RTL Source Policy Maker RTL Sink Policy Maker RTL EDID ROM, RTL I2C Controller
Test Bench	Verilog and VHDL
Constraints File	XDC Full Timing Constraints and Transceiver Physical Constraints
Simulation Model	Verilog and VHDL Wrapper
Supported S/W Driver	Standalone
Tested Design Flows[2]	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

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

Overview

This chapter contains an overview of the core as well as details about applications, licensing, and standards. The DisplayPort core is a full-featured soft IP core, incorporating all necessary logic to properly communicate on this high-speed standard. The core supports transmission of high-definition video from a standard-format main link onto up to four lanes of High-Speed Serial I/O.

Xilinx IPs has been successfully tested for hardware interoperability with many GPU sources and DisplayPort Sink devices. For additional details on the interoperability results, contact your local Xilinx sales representative.

Source Core Architecture

The Source core is partitioned into three major blocks, as shown in Figure 1-1:

- **Main Link:** Provides for the delivery of the primary video stream.
- **Secondary Link:** Integrates the delivery of audio information into the Main Link blanking period.
- **AUX Channel:** Establishes the dedicated source to sink communication channel.

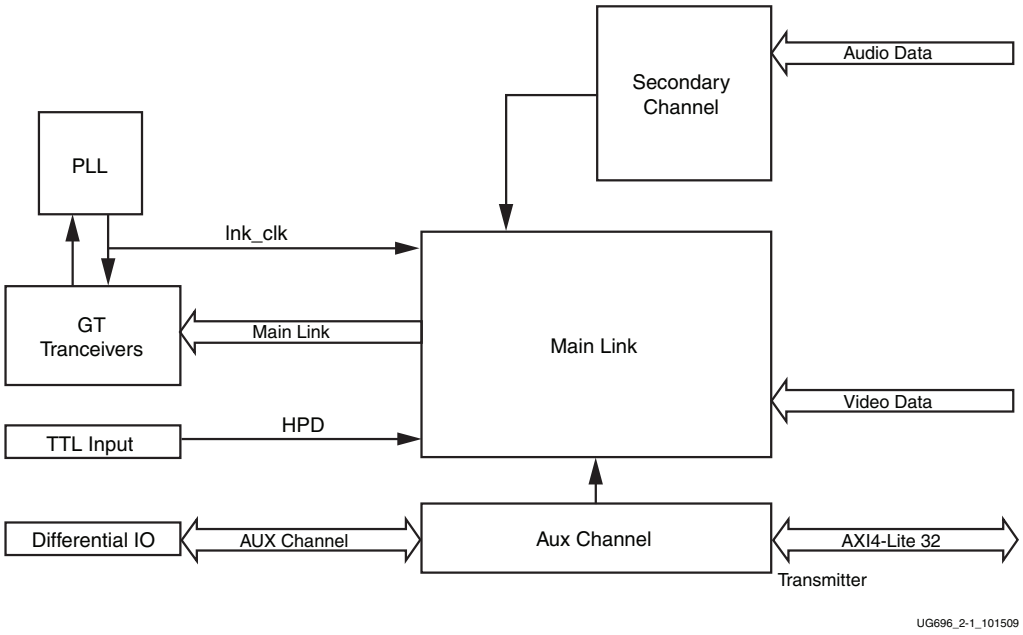

Figure 1-1: **Source Core Top Level**

Sink Core Architecture

The Sink core is partitioned into the following four major blocks

- **Main Link:** Provides for the delivery of the primary video stream.
- **Secondary Link:** 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 Display Port Configuration Data, which is used to establish the operating parameters of each core.

Figure 1-2 shows a top-level diagram of the Sink core.

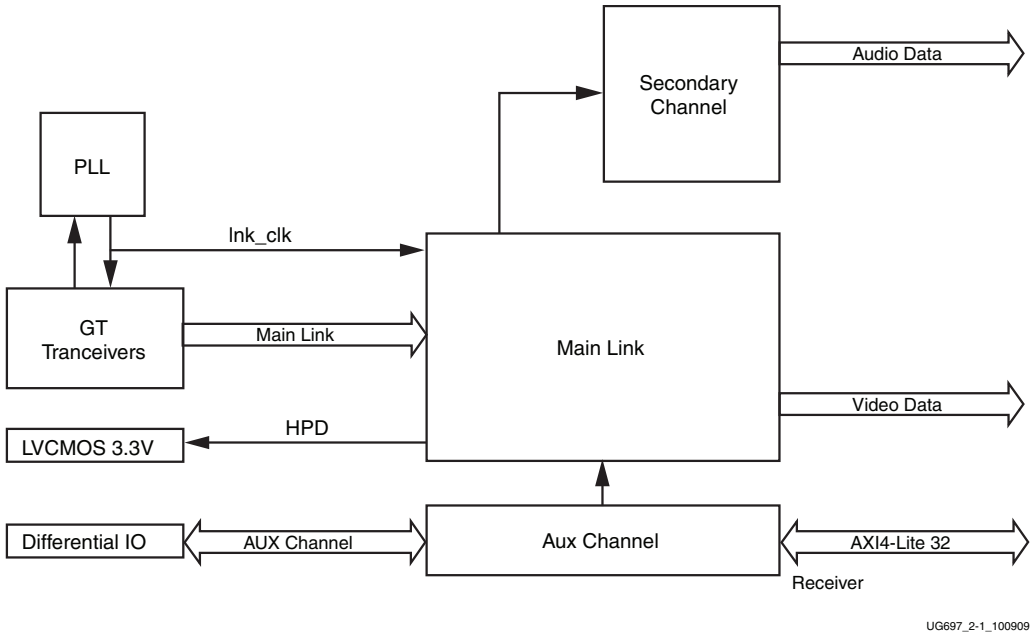

Figure 1-2: **Sink Core Top Level**

Feature Summary

Xilinx DisplayPort IP offers both Source (TX) and Sink (RX) functionality for high performance video, such as 4Kx2K resolution.

The DisplayPort IP core offers auto lane rate and width negotiation for 1, 2 or 4 lanes at 1.62, 2.7 or 5.4G based on core configuration over the AXI4-Lite interface and sink/source negotiations. The core supports vendor-specific DPCD and optional secondary audio. The DisplayPort core also provides an implementation of Multi-Stream Transport with support of up to four independent streams.

Unsupported Features

- The automated test feature is not supported.
- Bridging Function is not supported. The control registers required for bridging functionality are not included in the DisplayPort Configuration Data.
- MST audio is not supported.
- eDP optional features are not supported.
- iDP is not supported.

- GTC is not supported.

Acronym Definitions

The follow list defines acronyms frequently used in DisplayPort documentation:

- ACT: Allocation Change Trigger
- DPCD: DisplayPort Configuration Data
- eDP: Embedded Displayport
- GT: Gigabit Transceiver
- GTC: Global Time Code
- GUID: Globally Unique ID
- MST: Multi Stream Transport
- SST: Single Stream Transport
- TU: Transfer Unit
- VC Payload: Virtual Channel Payload

Licensing and Ordering Information

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

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

CAUTION! *Users attempting to use the Audio feature without a license will not see an error until implementation, at which point tools will generate an error stating that Reed Solomon Decoder license is not found.*

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

Product Specification

The Xilinx LogiCORE™ IP DisplayPort™ interconnect protocol is designed for transmission and reception of serial-digital video for consumer and professional displays. DisplayPort is a high-speed serial interface standard supported by PC chipsets, GPUs and display controllers, HDTV and monitors from industry leaders and major silicon manufacturers.

Standards

The IP described by this document is designed to be compatible with *DisplayPort Standard*, v1.1a and *DisplayPort Standard*, v1.2. For silicon status, please check the Vivado® IP catalog.

While the functional cores each include an I2C compatible interface, the design does not provide a fully compliant implementation. Specifically, the I2C interface sections do not support multiple bus masters and bus arbitration.

Performance

This section contains details about the performance of this core.

Maximum Frequencies

The core uses six clock domains. For more details about these clock domains, see Clocking in Chapter 3.

Table 2-1 shows the clock ranges.

Table 2-1: **Clock Ranges**

Clock Domain	Min	Max	Description
lnk_clk(3)	40 MHz	270 MHz(1)	Link clock
vid_clk	13.5 MHz(4)	150 MHz	Video clock
s_axi_aclk(2)	25 MHz	135 MHz	Host processor clock
aud_clk	16 MHz	100 MHz	Audio Clock (512 * Audio Sample Rate)

Table 2-1: **Clock Ranges *(Cont'd)***

Clock Domain	Min	Max	Description
s_aud_axis_aclk	16 MHz	100 MHz	= Audio Clock
m_aud_axis_aclk	16 MHz	100 MHz	= Audio Clock

Notes:

1. Valid for devices which support HBR2. HBR link rate will run at 135MHz.
2. When the Displayport IP is targeted to the UltraScale™ device, s_axi_aclk is connected to the GT wizard free-running clock input gtwiz_reset_clk_freerun_in. The clock frequency of s_axi_aclk should meet the frequency requirement, for example, s_axi_aclk <= lnk_clk (clock frequency generated by GT). For details, refer to the UltraScale FPGAs Transceiver Wizard Product Guide [Ref 18].
3. The lnk_clk and the GT reference clock, lnk_clk_p/n are different clock domains. For more details, see Clocking, page 110.
4. Valid for 1 pixel interface only. For 2, 4 pixel interfaces, the minimum video clock value can be lower than 13.5 MHz.

Resource Utilization

Table 2-2 and Table 2-3 show the DisplayPort core resource utilization for the Sink and Source cores of the Kintex®-7 family of FPGAs, respectively. These values have been generated using the Xilinx Vivado Design Suite. The values are derived from actual hardware validation systems.

Table 2-2: **Resource Utilization (TX with v1.2 Protocol, 5.4 Gb/s Link [XC7K325T FFG900-2])**

Video Interface Configuration	Slice LUTs	Slice Registers	Slice LUTs	Slice Registers
	16-bit GT Interface		32-bit GT Interface	
SST Configuration				
RGB_10bpc_2lanes_1pixel_SST	4767	3787	4808	4242
RGB_8bpc_2lanes_1pixel_SST	4618	3619	6446	4432
RGB_8bpc_2lanes_1pixel_SST_with_audio	6072	5542	7006	6427
RGB_8bpc_4lanes_4pixel_SST	4927	4809	7703	6377
MST Configuration				
RGB_10bpc_4lanes_4pixel_MSTx4streams	19606	15729	33881	22144
RGB_8bpc_4lanes_4pixel_MSTx2streams	10529	8810	17588	12418
RGB_8bpc_4lanes_4pixel_MSTx4streams	18412	14817	32229	21136
YCrCb_10bpc_4lanes_4pixel_MSTx4streams	19992	15473	34259	21888

Table 2-3: **Resource Utilization (RX with v1.2 Protocol, 5.4 Gb/s Link [XC7K325T FFG900-2])**

Video Interface Configuration	Slice LUTs	Slice Registers	Slice LUTs	Slice Registers
	16-bit GT Interface		32-bit GT Interface	
SST Configuration				
RGB_10bpc_2lanes_1pixel_SST	5439	5532	9928	6817
RGB_8bpc_2lanes_1pixel_SST	5358	5436	9496	6692
RGB_8bpc_2lanes_1pixel_SST_with_audio	6097	6454	10408	8161
RGB_8bpc_4lanes_4pixel_SST	8622	8297	17191	10859
MST Configuration				
RGB_10bpc_4lanes_4pixel_MSTx4streams	22671	19232	67817	30501
RGB_8bpc_4lanes_4pixel_MSTx2streams	14455	12880	38470	19016
RGB_8bpc_4lanes_4pixel_MSTx4streams	22467	19016	66068	30310
YCrCb_10bpc_4lanes_4pixel_MSTx4streams	22778	19264	67933	30533

Table 2-4 and Table 2-5 show the DisplayPort core resource utilization for the Sink and Source cores of the Kintex®-U family of FPGAs. These values have been generated using the Xilinx Vivado Design Suite and are derived from actual hardware validation systems.

Table 2-4: **Resource Utilization (TX with v1.2 Protocol, 5.4 Gb/s Link [xcku115-flvf1924-1L-i])**

Video Interface Configuration	Slice LUTs	Slice Registers	Slice LUTs	Slice Registers
	16-bit GT Interface		32-bit GT Interface	
SST Configuration				
RGB_10bpc_2lanes_1pixel_SST	3714	3890	5103	4715
RGB_8bpc_2lanes_1pixel_SST	3576	3795	4916	4604
RGB_8bpc_2lanes_1pixel_SST_with_audio	4834	5077	6461	6048
RGB_8bpc_4lanes_4pixel_SST	5205	5688	8177	7409
MST Configuration				
RGB_10bpc_4lanes_4pixel_MSTx4streams	20119	16580	33106	23310
RGB_8bpc_4lanes_4pixel_MSTx2streams	10576	9745	17494	13571
RGB_8bpc_4lanes_4pixel_MSTx4streams	18363	15116	32266	22750
YCrCb_10bpc_4lanes_4pixel_MSTx4streams	19778	15660	32765	22390

Table 2-5: **Resource Utilization (RX with v1.2 Protocol, 5.4 Gb/s Link [xcku115-flvf1924-1L-i])**

Video Interface Configuration	Slice LUTs	Slice Registers	Slice LUTs	Slice Registers
	16-bit GT Interface		32-bit GT Interface	
SST Configuration				
RGB_10bpc_2lanes_1pixel_SST	5463	5787	10032	7092
RGB_8bpc_2lanes_1pixel_SST	5339	5697	9539	7000
RGB_8bpc_2lanes_1pixel_SST_with_audio	6417	6854	10948	8638
RGB_8bpc_4lanes_4pixel_SST	8085	8482	16529	11108
MST Configuration				
RGB_10bpc_4lanes_4pixel_MSTx4streams	22110	19613	66921	31211
RGB_8bpc_4lanes_4pixel_MSTx2streams	14013	13179	37832	19646
RGB_8bpc_4lanes_4pixel_MSTx4streams	21804	19411	65689	31002
YCrCb_10bpc_4lanes_4pixel_MSTx4streams	22115	19645	66838	31242

Port Descriptions

This section lists the DisplayPort core ports.

Table 2-6: **Source Core I/O Signals**

Signal Name[(1)]	Direction From Core	Description
DisplayPort Processor Interface		
s_axi_aclk	Input	AXI Bus Clock.
s_axi_aresetn	Input	AXI Reset. Active-Low.
s_axi_awaddr[31:0]	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 bus.
s_axi_wstrb[3:0]	Input	Write strobes.
s_axi_wvalid	Input	Write valid.
s_axi_wready	Output	Write ready.
s_axi_bresp[1:0]	Output	Write response.
s_axi_bvalid	Output	Write response valid.

Table 2-6: **Source Core I/O Signals *(Cont'd)***

Signal Name[(1)]	Direction From Core	Description
s_axi_bready	Input	Response ready.
s_axi_araddr[31:0]	Input	Read address.
s_axi_arprot[2:0]	Input	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 response.
s_axi_rvalid	Output	Read valid.
s_axi_rready	Input	Read ready.
axi_int	Output	AXI interrupt out.
User Data Interface		
tx_vid_clk	Input	User data video clock.
tx_vid_vsync	Input	Vertical sync pulse. Active on the rising edge.
tx_vid_hsync	Input	Horizontal sync pulse. Active on the rising edge.
tx_vid_oddeven	Input	Odd/even field select. Indicates an odd (1) or even (0) field polarity.
tx_vid_enable	Input	User data video enable.
tx_vid_pixel0[47:0]	Input	Video data.
tx_vid_pixel1[47:0]	Input	Video data.
tx_vid_pixel2[47:0]	Input	Video data.
tx_vid_pixel3[47:0]	Input	Video data.
tx_vid_rst	Input	User video reset.
tx_bpc	Output	Bits per Color
tx_video_format	Output	Source Video format
tx_ppc	Output	Number of Pixels per clock
Main Link Interface		
lnk_clk_p	Input	GT reference differential clock input from p pin.
lnk_clk_n	Input	GT reference differential clock input from n pin.
lnk_clk	Output	Reference clock for the FPGA fabric.
lnk_tx_lane_p[3:0]	Output	High-speed lane serial data.
lnk_tx_lane_n[3:0]	Output	High-speed lane serial data.
AUX Channel Interface		
aux_tx_io_p	Input/Output	Positive Polarity AUX Manchester-II data

Table 2-6: **Source Core I/O Signals *(Cont'd)***

Signal Name[1]	Direction From Core	Description
aux_tx_io_n	Input/Output	Negative Polarity AUX Manchester-II data
HPD Interface		
tx_hpd	Input	Hot Plug Detect.
Audio Clock Interface		
aud_clk	Input	Audio sample clock (512 * fs). fs= sampling frequency.
aud_rst	Input	Audio Interface Reset (Active-High).
s_aud_axis_aclk	Input	Audio streaming interface clock (512 * fs)
s_aud_axis_aresetn	Input	Audio Streaming Interface Reset (Active-Low).
Shared Logic Included in Core		
lnk_clk_ibufds_out	Output	Link clock output from IBUFDS.
common_qpll_lock_out	Output	Active-High QPLL lock signal.
common_qpll_clk_out	Output	QPLL clock.
common_qpll_ref_clk_out	Output	QPLL reference clock output.
pll_lock_out	Output	Active-High PLL lock signal.
pll0_clk_out	Output	PLL clock.
pll0_ref_clk_out	Output	PLL reference clock output.
pll1_clk_out	Output	PLL clock.
pll1_ref_clk_out	Output	PLL reference clock output.
Shared Logic Included in Example Design		
lnk_clk_ibufds	Input	Link clock input from IBUFDS.
common_qpll_lock	Input	Active-High QPLL lock signal.
common_qpll_clk	Input	QPLL clock.
common_qpll_ref_clk	Input	QPLL reference clock input.
pll_lock	Input	Active High PLL lock signal.
pll0_clk	Input	PLL clock.
pll0_ref_clk	Input	PLL reference clock input.
pll1_clk	Input	PLL clock.
pll1_ref_clk	Input	PLL reference clock input.
link_bw_high_out	Output	Active-High status signal. High when link rate is 2.7 Gb/s.
link_bw_hbr2_out	Output	Active-High status signal. High when link rate is 5.4 Gb/s.
bw_changed_out	Output	Status signal to indicate the configuration change of link rate.

Table 2-6: **Source Core I/O Signals *(Cont'd)***

Signal Name[1]	Direction From Core	Description
phy_pll_reset_out	Output	Link layer driven PHY reset.

Notes:
1. Signal names beginning with `s_` or `m_` denote slave and master interfaces respectively.

Table 2-7: **Sink Core I/O Signals**

Signal Name[1]	Direction From Core	Description
DisplayPort Processor Interface		
s_axi_aclk	Input	AXI Bus Clock.
s_axi_aresetn	Input	AXI Reset. Active-Low.
s_axi_awaddr[31:0]	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 bus.
s_axi_wstrb[3:0]	Input	Write strobes.
s_axi_wvalid	Input	Write valid.
s_axi_wready	Output	Write ready.
s_axi_bresp[1:0]	Output	Write response.
s_axi_bvalid	Output	Write response valid.
s_axi_bready	Input	Response ready.
s_axi_araddr[31:0]	Input	Read address.
s_axi_arprot[2:0]	Input	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 repose.
s_axi_rvalid	Output	Read valid.
s_axi_rready	Input	Read ready.
axi_int	Output	AXI interrupt out.
User Data Interface		
rx_vid_clk	Input	User data video clock.
rx_vid_vsync	Output	Vertical sync pulse. Active on rising edge.
rx_vid_hsync	Output	Horizontal sync pulse. Active on rising edge. The vid_hsync signal only asserts to indicate when to start a new line.

Table 2-7: **Sink Core I/O Signals *(Cont'd)***

Signal Name[1]	Direction From Core	Description
rx_vid_oddeven	Output	Odd/even field select. Indicates an odd (1) or even (0) field polarity.
rx_vid_enable	Output	User data video enable.
rx_vid_pixel0[47:0]	Output	Video data.
rx_vid_pixel1[47:0]	Output	Video data.
rx_vid_pixel2[47:0]	Output	Video data.
rx_vid_pixel3[47:0]	Output	Video data.
rx_vid_rst	Input	User video reset.
rx_vid_pixel_mode	Output	Video pixel mode.
rx_vid_msa_hres	Output	Horizontal resolution of the main stream video source.
rx_vid_msa_vres	Output	Vertical resolution of the main stream video source.
Main Link Interface		
lnk_clk	Output	Reference clock for the FPGA fabric.
lnk_clk_p	Input	GT reference differential clock input from p pin.
lnk_clk_n	Input	GT reference differential clock input from n pin.
lnk_rx_lane_p[3:0]	Input	High-speed lane serial data.
lnk_rx_lane_n[3:0]	Input	High-speed lane serial data.
lnk_m_vid[23:0]	Output	M-value for clock generation.
lnk_n_vid[23:0]	Output	N-value for clock generation.
lnk_m_aud[23:0]	Output	M-value for audio clock generation.
lnk_n_aud[23:0]	Output	N-Value for audio clock generation.
AUX Channel Interface		
aux_rx_io_p	Input/Output	Positive Polarity AUX Manchester-II data.
aux_rx_io_n	Input/Output	Negative Polarity AUX Manchester-II data.
I2C Interface		
i2c_sda_in	Input	I2C serial data in.
i2c_sda_enable_n	Output	I2C data out enable. Active-Low.
i2c_scl_in	Input	I2C serial clock in.
i2c_scl_enable_n	Output	I2C serial clock output enable. Active-Low.
HPD Interface		
rx_hpd	Output	Hot Plug Detect.
Audio Clock Interface		
aud_rst	Input	Audio Interface Reset (active-High).

Table 2-7: **Sink Core I/O Signals *(Cont'd)***

Signal Name[1]	Direction From Core	Description
m_aud_axis_aclk	Input	Audio streaming interface clock (equal to 512 * fs).
m_aud_axis_aresetn	Input	Audio Streaming Interface Reset (active-Low).
Shared Logic Included in Core		
lnk_clk_ibufds_out	Output	Reference clock output from IBUFDS.
common_qpll_lock_out	Output	Active-High QPLL lock signal (GTX,GTH).
common_qpll_clk_out	Output	QPLL clock (GTX,GTH).
common_qpll_ref_clk_out	Output	QPLL reference clock output (GTX,GTH).
pll_lock_out	Output	Active High PLL lock signal (GTP).
pll0_clk_out	Output	PLL clock (GTP).
pll0_ref_clk_out	Output	PLL reference clock output (GTP).
pll1_clk_out	Output	PLL clock (GTP).
pll1_ref_clk_out	Output	PLL reference clock output (GTP).
Shared Logic Included in Example Design		
lnk_clk_ibufds	Input	Reference clock input from IBUFDS.
common_qpll_lock	Input	Active-High QPLL lock signal (GTX,GTH).
common_qpll_clk	Input	QPLL clock (GTX,GTH).
common_qpll_ref_clk	Input	QPLL reference clock input (GTX,GTH).
pll_lock	Input	Active High PLL lock signal.
pll0_clk	Input	PLL clock (GTP).
pll0_ref_clk	Input	PLL reference clock input (GTP).
pll1_clk	Input	PLL clock (GTP).
pll1_ref_clk	Input	PLL reference clock input (GTP).
link_bw_high_out	Output	Active-High status signal. High when link rate is 2.7 Gb/s.
link_bw_hbr2_out	Output	Active-High status signal. High when link rate is 5.4 Gb/s.
bw_changed_out	Output	Status signal to indicate the configuration change of link rate.
phy_pll_reset_out	Output	Link layer driven PHY reset.

Notes:
1. Signal names beginning with `s_` or `m_` denote slave and master interfaces respectively.

Audio Streaming Signals

The DisplayPort Source Audio streaming signals are listed in Table 2-8.

Table 2-8: **DisplayPort Source Audio Interface**

Name	Direction	Description
s_axis_audio_ingress_tdata [31:0]	Input	Streaming data input. • [3:0] – PR (Preamble Code) ◦ 4'b0001 -> Subframe1 / start of audio block ◦ 4'b0010 -> Subframe 1 ◦ 4'b0011 -> Subframe 2 • [27:4] – Audio Sample Word • [28] – V (Validity Bit) • [29] – U (User Bit) • [30] – C (Channel Status) • [31] – P (Parity)
s_axis_audio_ingress_tid [7:0]	Input	• [3:0] – Audio Channel ID • [7:4] – Audio Packet Stream ID
s_axis_audio_ingress_tvalid	Input	Valid indicator for audio data from master.
s_axis_audio_ingress_tready	Output	Ready indicator from DisplayPort source.

The DisplayPort Sink Audio streaming definition is listed in Table 2-9.

Table 2-9: **DisplayPort Sink Audio Interface**

Name	Direction	Description
m_axis_audio_egress_tdata [31:0]	Output	Streaming data output. • [3:0] – PR (Preamble Code) ◦ 4'b0001 -> Subframe1 / start of audio block ◦ 4'b0010 -> Subframe 1 ◦ 4'b0011 -> Subframe 2 • [27:4] – Audio Sample Word • [28] – V (Validity Bit) • [29] – U (User Bit) • [30] – C (Channel Status) • [31] – P (Parity)
m_axis_audio_egress_tid [7:0]	Output	• [3:0] – Audio Channel ID • [7:4] – Audio Packet Stream ID
m_axis_audio_egress_tvalid	Output	Valid indicator for audio data from master.
m_axis_audio_egress_tready	Input	Ready indicator from external streaming module.

MST Signals

Table 2-11 shows the MST signals for the Source core. User pixel width programming in source can be programmed independently for each stream.

Table 2-10: **MST Source Signals**

Signal Name	Direction From Core	Description
Video Stream 2		
tx_vid_clk_stream2	Input	User data video clock
tx_vid_vsync_stream2	Input	Vertical sync pulse
tx_vid_hsync_stream2	Input	Horizontal sync pulse
tx_vid_oddeven_stream2	Input	Odd/even field select
tx_vid_enable_stream2	Input	User data video enable
tx_vid_pixel0_stream2 [47:0]	Input	Video data
tx_vid_pixel1_stream2 [47:0]	Input	Video data
tx_vid_pixel2_stream2 [47:0]	Input	Video data
tx_vid_pixel3_stream2 [47:0]	Input	Video data
tx_vid_rst_stream2	Input	User Video Reset
Video Stream 3		
tx_vid_clk_stream3	Input	User data video clock
tx_vid_vsync_stream3	Input	Vertical sync pulse
tx_vid_hsync_stream3	Input	Horizontal sync pulse
tx_vid_oddeven_stream3	Input	Odd/even field select
tx_vid_enable_stream3	Input	User data video enable
tx_vid_pixel0_stream3 [47:0]	Input	Video data
tx_vid_pixel1_stream3 [47:0]	Input	Video data
tx_vid_pixel2_stream3 [47:0]	Input	Video data
tx_vid_pixel3_stream3 [47:0]	Input	Video data
tx_vid_rst_stream3	Input	User Video Reset
Video Stream 4		
tx_vid_clk_stream4	Input	User data video clock
tx_vid_vsync_stream4	Input	Vertical sync pulse
tx_vid_hsync_stream4	Input	Horizontal sync pulse
tx_vid_oddeven_stream4	Input	Odd/even field select
tx_vid_enable_stream4	Input	User data video enable
tx_vid_pixel0_stream4 [47:0]	Input	Video data
tx_vid_pixel1_stream4 [47:0]	Input	Video data
tx_vid_pixel2_stream4 [47:0]	Input	Video data
tx_vid_pixel3_stream4 [47:0]	Input	Video data
tx_vid_rst_stream4	Input	User Video Reset

Table 2-11 shows the MST signals for the Sink core. User pixel width programming in sink applies to all streams.

Table 2-11: **MST Sink Signals**

Signal Name	Direction From Core	Description
Video Stream 1		
rx_vid_vsync_stream1	Output	Vertical sync pulse
rx_vid_hsync_stream1	Output	Horizontal sync pulse
rx_vid_oddeven_stream1	Output	Odd/even field select
rx_vid_enable_stream1	Output	User data video enable
rx_vid_pixel0_stream1[47:0]	Output	Video data
rx_vid_pixel1_stream1[47:0]	Output	Video data
rx_vid_pixel2_stream1[47:0]	Output	Video data
rx_vid_pixel3_stream1[47:0]	Output	Video data
rx_vid_msa_hres_stream1	Output	Horizontal resolution of the main stream video source
rx_vid_msa_vres_stream1	Output	Vertical resolution of the main stream video source
Video Stream 2		
rx_vid_vsync_stream2	Output	Vertical sync pulse
rx_vid_hsync_stream2	Output	Horizontal sync pulse
rx_vid_oddeven_stream2	Output	Odd/even field select
rx_vid_enable_stream2	Output	User data video enable
rx_vid_pixel0_stream2[47:0]	Output	Video data
rx_vid_pixel1_stream2[47:0]	Output	Video data
rx_vid_pixel2_stream2[47:0]	Output	Video data
rx_vid_pixel3_stream2[47:0]	Output	Video data
rx_vid_msa_hres_stream2	Output	Horizontal resolution of the main stream video source
rx_vid_msa_vres_stream2	Output	Vertical resolution of the main stream video source
Video Stream 3		
rx_vid_vsync_stream3	Output	Vertical sync pulse
rx_vid_hsync_stream3	Output	Horizontal sync pulse
rx_vid_oddeven_stream3	Output	Odd/even field select
rx_vid_enable_stream3	Output	User data video enable
rx_vid_pixel0_stream3[47:0]	Output	Video data
rx_vid_pixel1_stream3[47:0]	Output	Video data

Table 2-11: **MST Sink Signals *(Cont'd)***

Signal Name	Direction From Core	Description
rx_vid_pixel2_stream3[47:0]	Output	Video data
rx_vid_pixel3_stream3[47:0]	Output	Video data
rx_vid_msa_hres_stream3	Output	Horizontal resolution of the main stream video source
rx_vid_msa_vres_stream3	Output	Vertical resolution of the main stream video source
Video Stream 4		
rx_vid_vsync_stream4	output	Vertical sync pulse
rx_vid_hsync_stream4	output	Horizontal sync pulse
rx_vid_oddeven_stream4	Output	Odd/even field select
rx_vid_enable_stream4	Output	User data video enable
rx_vid_pixel0_stream4[47:0]	Output	Video data
rx_vid_pixel1_stream4[47:0]	Output	Video data
rx_vid_pixel2_stream4[47:0]	Output	Video data
rx_vid_pixel3_stream4[47:0]	Output	Video data
rx_vid_msa_hres_stream4	Output	Horizontal resolution of the main stream video source
rx_vid_msa_vres_stream4	Output	Vertical resolution of the main stream video source

Register Space

Source Core

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

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

Any bits not specified in Table 2-12 are considered reserved and returns 0 upon read. The power on reset values of all the registers are 0 unless it is specified in the definition. Only address offsets are listed in Table 2-12. Base addresses are configured by the AXI Interconnect.

Table 2-12: **DisplayPort Source Core Configuration Space**

Offset	R/W	Definition
Link Configuration Field		
0x000	RW	LINK_BW_SET. Main link bandwidth setting. The register uses the same values as those supported by the DPCD register of the same name in the sink device. • [7:0] – LINK_BW_SET: Sets the value of the main link bandwidth for the sink device. ◦ 0x06 = 1.62 Gb/s ◦ 0x0A = 2.7 Gb/s ◦ 0x14 = 5.4 Gb/s
0x004	RW	LANE_COUNT_SET. Sets the number of lanes used by the source in transmitting data. • [4:0] – Set to 1, 2, or 4
0x008	RW	ENHANCED_FRAME_EN • [0] -Set to 1 by the source to enable the enhanced framing symbol sequence.
0x00C	RW	TRAINING_PATTERN_SET. Sets the link training mode. • [1:0] – Set the link training pattern according to the two bit code. ◦ 00 = Training off ◦ 01 = Training pattern 1, used for clock recovery ◦ 10 = Training pattern 2, used for channel equalization ◦ 11 = Training pattern 3, used for channel equalization for cores with DisplayPort Standard v1.2.
0x010	RW	LINK_QUAL_PATTERN_SET. Transmit the link quality pattern. • [1:0] – Enable transmission of the link quality test patterns. ◦ 00 = Link quality test pattern not transmitted ◦ 01 = D10.2 test pattern (unscrambled) transmitted ◦ 10 = Symbol Error Rate measurement pattern ◦ 11 = PRBS7 transmitted
0x014	RW	SCRAMBLING_DISABLE. Set to 1 when the transmitter has disabled the scrambler and transmits all symbols. • [0] – Disable scrambling.
0x01C	WO	SOFTWARE_RESET. Reads will return zeros. • [0] – Soft Video Reset: When set, video logic is reset (stream 1). • [1] – Soft Video Reset: When set, video logic is reset (stream 2). • [2] – Soft Video Reset: When set, video logic is reset (stream 3). • [3] – Soft Video Reset: When set, video logic is reset (stream 4). • [7] – AUX Soft Reset. When set, AUX logic is reset.
Core Enables		
0x080	RW	TRANSMITTER_ENABLE. Enable the basic operations of the transmitter. • [0] – When set to 1, stream transmission is enabled. When set to 0, all lanes of the main link output stuffing symbols.
0x084	RW	MAIN_STREAM_ENABLE. Enable the transmission of main link video information. • [0] – When set to 0, the active lanes of the DisplayPort transmitter will output only VB-ID information with the NoVideo flag set to 1. ***Note:*** Main stream enable/disable functionality is gated by the VSYNC input. The values written in the register are applied at the video frame boundary only.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x0C0	WO	FORCE_SCRAMBLER_RESET. Reads from this register always return 0x0. • [0] – 1 forces a scrambler reset.
0x0D0	RW	TX_MST_CONFIG: MST Configuration. • [0] – MST Enable: Set to 1 to enable MST functionality. • [1] – VC Payload Updated in sink: This is an WO bit. Set to 1 after reading DPCD register 0x2C0 (bit 0) is set.
Core ID		
0x0F8	RO	VERSION_REGISTER. For example, for displayport_v6_0, the VERSION REGISTER is 32'h06_00_0_0_00. • [31:24] – Core major version. • [23:16] – Core minor version. • [15:12] – Core version revision. • [11:8] – Core Patch details. • [7:0] – Internal revision.
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] ◦ 0x00: Transmit ◦ 0x01: Receive The CORE_ID values for the various protocols and cores are: • DisplayPort Standard v1.1a protocol with a Transmit core: 32'h01_01_0a_00 • DisplayPort Standard v1.2a protocol with a Transmit core: 32'h01_02_0a_00
AUX Channel Interface		
0x100	RW	AUX_COMMAND_REGISTER. Initiates AUX channel commands of the specified length. • [12] – Address only transfer enable. When this bit is set to 1, the source initiates Address only transfers (STOP is sent after the command). • [11:8] – AUX Channel Command. ◦ 0x8 = AUX Write ◦ 0x9 = AUX Read ◦ 0x0 = IC Write ◦ 0x4 = IC Write MOT ◦ 0x1 = IC Read ◦ 0x5 = IC Read MOT ◦ 0x2 = IC Write Status • [3:0] – Specifies the number of bytes to transfer with the current command. The range of the register is 0 to 15 indicating between 1 and 16 bytes of data.
0x104	WO	AUX_WRITE_FIFO. FIFO containing up to 16 bytes of write data for the current AUX channel command. • [7:0] – AUX Channel byte data.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x108	RW	AUX_ADDRESS. Specifies the address for the current AUX channel command. • [19:0] – Twenty bit address for the start of the AUX Channel burst.
0x10C	RW	AUX_CLOCK_DIVIDER. Contains the clock divider value for generating the internal 1MHz 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). • [7:0] – Clock divider value. • [15:8] – The number of AXI-Lite clocks (defined by the AXI-Lite clock name: s_axi_aclk) equivalent to the recommended width of AUX pulse. Allowable values include: 8,16,24,32,40 and 48. From DP Protocol spec, AUX Pulse Width range = 0.4 to 0.6 us. For example, for AXI Lite clock of 50MHz (=20ns), 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.
0x110	RC	TX_USER_FIFO_OVERFLOW. Indicates an overflow in the user FIFO. The event can occur if the video rate does not match the TU size programming. • [0] – FIFO_OVERFLOW_FLAG: 1 indicates that the internal FIFO has detected an overflow condition. This bit clears upon read.
0x130	RO	INTERRUPT_SIGNAL_STATE. Contains the raw signal values for those conditions which may cause an interrupt. • [3] – REPLY_TIMEOUT: 1 indicates that a reply timeout has occurred. • [2] – REPLY_STATE: 1 indicates that a reply is currently being received. • [1] – REQUEST_STATE: 1 indicates that a request is currently being sent. • [0] – HPD_STATE: Contains the raw state of the HPD pin on the DisplayPort connector.
0x134	RO	AUX_REPLY_DATA. Maps to the internal FIFO which contains up to 16 bytes of information received during the AUX channel reply. Reply data is read from the FIFO starting with byte 0. The number of bytes in the FIFO corresponds to the number of bytes requested. • [7:0] – AUX reply data
0x138	RO	AUX_REPLY_CODE. Reply code received from the most recent AUX Channel request. The AUX Reply Code corresponds to the code from the DisplayPort Standard. ***Note:*** The core will not retry any commands that were Deferred or Not Acknowledged. • [1:0] ◦ 00 = AUX ACK ◦ 01 = AUX NACK ◦ 10 = AUX DEFER • [3:2] ◦ 00 = I2C ACK ◦ 01 = I2C NACK ◦ 10 = I2C DEFER
0x13C	RW	AUX_REPLY_COUNT. Provides an internal counter of the number of AUX reply transactions received on the AUX Channel. Writing to this register clears the count. • [7:0] – Current reply count.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x140	RC	INTERRUPT_STATUS. Source core interrupt status register. A read from this register clears all values. Write operation is illegal and clears the values. • [9] – Audio packet ID mismatch interrupt, sets when incoming audio packet ID over streaming interface does not match with the info frame packet stream ID. • [5] – EXT_PKT_TXD: Extended packet is transmitted and controller is ready to accept new packet. • [4] – HPD_PULSE_DETECTED: A pulse on the HPD line was detected. The duration of the pulse can be determined by reading 0x150. • [3] – REPLY_TIMEOUT: A reply timeout has occurred. • [2] – REPLY_RECEIVED: An AUX reply transaction has been detected. • [1] – HPD_EVENT: The core has detected the presence of the HPD signal. This interrupt asserts immediately after the detection of HPD and after the loss of HPD for 2 msec. • [0] – HPD_IRQ: An IRQ framed with the proper timing on the HPD signal has been detected.
0x144	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. The respective MASK bit controls the assertion of axi_int only and does not affect events updated in the INTERRUPT_STATUS register. • [9] – Mask Audio packet ID mismatch interrupt. • [5] – EXT_PKT_TXD: Mask Extended Packet Transmitted interrupt. • [4] – HPD_PULSE_DETECTED: Mask HPD Pulse interrupt. • [3] – REPLY_TIMEOUT: Mask reply timeout interrupt. • [2] – REPLY_RECEIVED: Mask reply received interrupt. • [1] – HPD_EVENT: Mask HPD event interrupt. • [0] – HPD_IRQ: Mask HPD IRQ interrupt.
0x148	RO	REPLY_DATA_COUNT. Returns the total number of data bytes actually received during a transaction. This register does not use the length byte of the transaction header. • [4:0] – Total number of data bytes received during the reply phase of the AUX transaction.
0x14C	RO	REPLY_STATUS • [15:12] – RESERVED • [11:4] – REPLY_STATUS_STATE: Internal AUX reply state machine status bits. • [3] – REPLY_ERROR: When set to a 1, the AUX reply logic has detected an error in the reply to the most recent AUX transaction. • [2] – REQUEST_IN_PROGRESS: The AUX transaction request controller sets this bit to a '1' while actively transmitting a request on the AUX serial bus. The bit is set to 0 when the AUX transaction request controller is idle. • [1] – REPLY_IN_PROGRESS: The AUX reply detection logic sets this bit to a 1 while receiving a reply on the AUX serial bus. The bit is 0 otherwise. • [0] – REPLY_RECEIVED: This bit is set to '0' when the AUX request controller begins sending bits on the AUX serial bus. The AUX reply controller sets this bit to 1 when a complete and valid reply transaction has been received.
0x150	RO	HPD_DURATION • [15:0] – Duration of the HPD pulse in microseconds.
0x154	RO	Free running counter incrementing for every 1Mhz.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
Main Stream Attributes (Refer to the DisplayPort Standard for more details [Ref 1].)		
0x180	RW	MAIN_STREAM_HTOTAL. Specifies the total number of clocks in the horizontal framing period for the main stream video signal. • [15:0] – Horizontal line length total in clocks.
0x184	RW	MAIN_STREAM_VTOTAL. Provides the total number of lines in the main stream video frame. • [15:0] – Total number of lines per video frame.
0x188	RW	MAIN_STREAM_POLARITY. Provides the polarity values for the video sync signals. • [1] – VSYNC_POLARITY: Polarity of the vertical sync pulse. • [0] – HSYNC_POLARITY: Polarity of the horizontal sync pulse.
0x18C	RW	MAIN_STREAM_HSWIDTH. Sets the width of the horizontal sync pulse. • [14:0] – Horizontal sync width in clock cycles.
0x190	RW	MAIN_STREAM_VSWIDTH. Sets the width of the vertical sync pulse. • [14:0] – Width of the vertical sync in lines.
0x194	RW	MAIN_STREAM_HRES. Horizontal resolution of the main stream video source. • [15:0] – Number of active pixels per line of the main stream video.
0x198	RW	MAIN_STREAM_VRES. Vertical resolution of the main stream video source. • [15:0] – Number of active lines of video in the main stream video source.
0x19C	RW	MAIN_STREAM_HSTART. Number of clocks between the leading edge of the horizontal sync and the start of active data. • [15:0] – Horizontal start clock count.
0x1A0	RW	MAIN_STREAM_VSTART. Number of lines between the leading edge of the vertical sync and the first line of active data. • [15:0] – Vertical start line count.
0x1A4	RW	MAIN_STREAM_MISC0. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC0 register described in section 2.2.4 of the standard. • [0] -Synchronous Clock. • [2:1] – Component Format. • [3] – Dynamic Range. • [4] – YCbCr Colorimetry. • [7:5] – Bit depth per color/component.
0x1A8	RW	MAIN_STREAM_MISC1. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC1 register described in section 2.2.4 of the standard. • [0] – Interlaced vertical total even. • [2:1] – Stereo video attribute. • [6:3] – Reserved.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x1AC	RW	M-VID. If synchronous clocking mode is used, this register must be written with the M value as described in section 2..2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream is automatically computed by the source core and written to the main stream. These values are not written into the M-VID register for readback. • [23:0] – Unsigned M value.
0x1B0	RW	TRANSFER_UNIT_SIZE. Sets the size of a transfer unit in the framing logic On reset, transfer size is set to 64. This register must be written as described in section 2.2.1.4.1 of the standard. • [6:0] – This number should be in the range of 32 to 64 and is set to a fixed value that depends on the inbound video mode. Note that bit 0 cannot be written (the transfer unit size is always even).
0x1B4	RW	N-VID. If synchronous clocking mode is used, this register must be written with the N value as described in section 2..2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream is automatically computed by the source core and written to the main stream. These values are not written into the N-VID register for readback. • [23:0] – Unsigned N value.
0x1B8	RW	USER_PIXEL_WIDTH. Selects the width of the user data input port. Use only quad pixel mode in MST. • [2:0]: ◦ 1 - Single pixel wide interface ◦ 2 - Dual pixel wide interface ◦ 4 - Quad pixel wide interface

0x1BC	RW	USER_DATA_COUNT_PER_LANE. This register is used to translate the number of pixels per line to the native internal 16-bit datapath.
		If (HRES * bits per pixel) is divisible by 16, then
		word_per_line = ((HRES * bits per pixel)/16)
		Else
		word_per_line = (INT((HRES * bits per pixel)/16))+1
		
		For single-lane design:
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 1
		
		For 2-lane design:
		If words_per_line is divisible by 2, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 2
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,2) - 2
		
		For 4-lane design:
		If words_per_line is divisible by 4, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 4
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,4) - 4

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x1C0	RW	MAIN_STREAM_INTERLACED. Informs the DisplayPort transmitter main link that the source video is interlaced. By setting this bit to a 1, the core will set the appropriate fields in the VBID value and Main Stream Attributes. This bit must be set to a 1 for the proper transmission of interlaced sources. • [0] – Set to a 1 when transmitting interlaced images.
0x1C4	RW	MIN_BYTES_PER_TU. Programs source to use MIN number of bytes per transfer unit. The calculation should be done based on the DisplayPort Standard. MIN_BYTES_PER_TU should be greater than or equal to 4 when GT Data width is selected as 32 bit. • [6:0] – Set the value to INT((VIDEO_BW/LINK_BW)*TRANSFER_UNIT_SIZE)
0x1C8	RW	FRAC_BYTES_PER_TU. Calculating MIN bytes per TU will often not be a whole number. This register is used to hold the fractional component. • [9:0] – The fraction part of ((VIDEO_BW/LINK_BW)*TRANSFER_UNIT_SIZE) scaled by 1024 is programmed in this register.

0x1cc	RW	INIT_WAIT. This register defines the number of initial wait cycles at the start of a new line by the Framing logic. This allows enough data to be buffered in the input FIFO. The default value of INIT_WAIT is 0x20.
		If (MIN_BYTES_PER_TU <= 4)
		• [7:0] – Set INIT_WAIT to 64
		Else
		• [7:0] – Set INIT_WAIT to (TRANSFER_UNIT_SIZE - MIN_BYTES_PER_TU)

0x1D0	RW	STREAM1. Average Stream Symbol Timeslots per MTP Config: • [9:0] – TS_FRAC: Program fraction * 1000 in this field. See the *DisplayPort Standard* section 2.6.3.3 VC Payload Size Determination by a Source Payload Bandwidth Manager. • [23:16] – TS_INT: Program integer value based on the calculations.
0x1D4	RW	STREAM2. Average Stream Symbol Timeslots per MTP Config: • [9:0] – TS_FRAC: Program fraction * 1000 in this field. See the *DisplayPort Standard* section 2.6.3.3 VC Payload Size Determination by a Source Payload Bandwidth Manager. • [23:16] – TS_INT: Program integer value based on the calculations.
0x1D8	RW	STREAM3. Average Stream Symbol Timeslots per MTP Config: • [9:0] – TS_FRAC: Program fraction * 1000 in this field. See the *DisplayPort Standard* section 2.6.3.3 VC Payload Size Determination by a Source Payload Bandwidth Manager. • [23:16] – TS_INT: Program integer value based on the calculations.
0x1DC	RW	STREAM4. Average Stream Symbol Timeslots per MTP Config: • [9:0] – TS_FRAC: Program fraction * 1000 in this field. See the *DisplayPort Standard* section 2.6.3.3 VC Payload Size Determination by a Source Payload Bandwidth Manager. • [23:16] – TS_INT: Program integer value based on the calculations.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
PHY Configuration Status		
0x200	RW	PHY_CONFIG. • [1:0] – Controls the reset to the PHY section of the DisplayPort receiver core. At power up, this register has a value of 0x3, holding the receiver PHY in the reset state. Set to 0 for proper functioning of the receiver core: ◦ [0] – Set to 1 to hold the PHY in reset. Clear to release. ◦ [1] – Set to 1 to hold GTTXRESET in reset. Clear to release. • [8] – Set to 1 to hold TX_PHY_PMA Reset. Clear to release. • [9] – Set to 1 to hold TX_PHY_PCS Reset. Clear to release. • [11] – Set to configure TX_PHY_POLARITY. Default is 0. • [12] – Set to configure TX_PHY_PRBSFORCEERR. Default is 0. • [15:13] – Set to configure TX_PHY_LOOPBACK. Default is 0. • [16] – Set to enable the individual lane polarity. Set to 0 to use common polarity control through bit [11] for all lanes. • [17] – Set to configure TX_PHY_POLARITY for Lane 0. • [18] – Set to configure TX_PHY_POLARITY for Lane 1. • [19] – Set to configure TX_PHY_POLARITY for Lane 2. • [20] – Set to configure TX_PHY_POLARITY for Lane 3. • [21] – Set to 1 to enable the 8B10B coding. Default is 1. SW should not unset this bit for normal operation of TX PHY. • [27] – GT TXINHIBIT input. See appropriate transceiver user guide for programming details.
0x220	RW	PHY_VOLTAGE_DIFF_LANE_0. Controls the differential voltage swing for lane 0 of the DisplayPort link. • [3:0] – Supports up to eight levels of voltage swing for a wide variety of PHY implementations. The mapping of the four levels supported by the DisplayPort Standard to the eight levels indicated here is implementation specific.
0x224	RW	PHY_VOLTAGE_DIFF_LANE_1. Bit definition identical to that of PHY_VOLTAGE_DIFF_LANE_0.
0x228	RW	PHY_VOLTAGE_DIFF_LANE_2. Bit definition identical to that of PHY_VOLTAGE_DIFF_LANE_0.
0x22C	RW	PHY_VOLTAGE_DIFF_LANE_3. Bit definition identical to that of PHY_VOLTAGE_DIFF_LANE_0.
0x230	RW	TRANSMIT_PRBS7. Enable the pseudo random bit sequence 7 pattern transmission for link quality assessment. • [0] – 1 in this bit enables the transmission of the sequence. • [16:1] – GT PCSRSVDIN. Used in UltraScale™ devices only.
0x234	RW	PHY_CLOCK_SELECT. Instructs the PHY PLL to generate the proper clock frequency for the required link rate. • [2:0] ◦ 0x05 = 5.40 Gb/s link ◦ 0x03 = 2.70 Gb/s link ◦ 0x01 = 1.62 Gb/s link

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x238	RW	TX_PHY_POWER_DOWN [3:0]. Control PHY Power down. One bit per lane. When set to 1, moves the GT to power down mode.
0x23c	RW	PHY_PRECURSOR_LANE_0. Set the pre-cursor level for lane 0 of the DisplayPort link. • [4:0] – Controls the pre-cursor level for lane 0 of the transmitter. The mapping of the four levels supported by the *DisplayPort Standard* to the 32 levels indicated here is implementation specific. Valid for 7 series FPGAs only.
0x240	RW	PHY_PRECURSOR_LANE_1. Bit definition identical to that of PHY_PRECURSOR_LANE_0.
0x244	RW	PHY_PRECURSOR_LANE_2. Bit definition identical to that of PHY_PRECURSOR_LANE_0.
0x248	RW	PHY_PRECURSOR_LANE_3. Bit definition identical to that of PHY_PRECURSOR_LANE_0.
0x24c	RW	PHY_POSTCURSOR_LANE_0. Set the post-cursor level for lane 0 of the DisplayPort link. • [4:0] – Controls the post-cursor level for lane 0 of the transmitter. The mapping of the four levels supported by the *DisplayPort Standard* to the 32 levels indicated here is implementation specific. Valid for 7 series FPGAs only.
0x250	RW	PHY_POSTCURSOR_LANE_1. Bit definition identical to that of PHY_POSTCURSOR_LANE_0.
0x254	RW	PHY_POSTCURSOR_LANE_2. Bit definition identical to that of PHY_POSTCURSOR_LANE_0.
0x258	RW	PHY_POSTCURSOR_LANE_3. Bit definition identical to that of PHY_POSTCURSOR_LANE_0.
0x280	RO	PHY_STATUS. Provides the current status from the PHY. • [1:0] – Reset done for lanes 0 and 1. • [3:2] – Reset done for lanes 2 and 3. • [4] – PLL for lanes 0 and 1 locked. • [5] – PLL for lanes 2 and 3 locked. • [6] – FPGA fabric clock PLL locked. • [15:7] – Unused, read as 0. • [17:16] – Transmitter buffer status, lane 0. • [19:18] – Transmitter error, lane 0. • [21:20]- Transmitter buffer status, lane 1. • [23:22] – Transmitter error, lane 1. • [25:24] – Transmitter buffer status, lane 2. • [27:26] – Transmitter error, lane 2. • [29:28] – Transmitter buffer status, lane 3. • [31:30] – Transmitter error, lane 3.
0x2A0	RW	GT_DRP_COMMAND. Provides access to GT DRP ports. All channels use the same programming. • [7:0] – DRP Address • [15] – DRP Write/Read Command. ◦ 1: Write ◦ 0: Read • [31:16] – DRP Write Data (not valid for read command).

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x2A4	RO	GT_DRP_READ_DATA: Provides access to GT DRP READ Data. The data is sampled when DRP ready is asserted. • [15:0] – DRP Read Data. After issuing DRP Command register, software should wait for some time (typically 10 * AXI4-Lite Clock Period) to ensure DRP access is completed before reading the data.
0x2A8	RO	GT_DRP_CHANNEL STATUS: Provides access to GT DRP CHANNEL STATUS. • [0] – DRP Locked. Locked is asserted when IP state machine uses GT DRP. Software has to poll this bit and initiate read/write transaction only when the locked bit is set to 0.
0x4FC	RO	SINK_VID_FRAMING_ERROR_STATUS: Sink Video Framing error status. This is a debug register that is valid when GT data width is 32-Bit. • [1:0] – Stream1 error status in framing. • [9:8] – Stream2 error status in framing. • [17:16] – Stream3 error status in framing. • [25:24] – Stream4 error status in framing.
MST Interface		
0x500	RW	MAIN_STREAM_HTOTAL_STREAM2. Specifies the total number of clocks in the horizontal framing period for the main stream video signal. • [15:0] – Horizontal line length total in clocks.
0x504	RW	MAIN_STREAM_VTOTAL_STREAM2. Provides the total number of lines in the main stream video frame. • [15:0] – Total number of lines per video frame.
0x508	RW	MAIN_STREAM_POLARITY_STREAM2. Provides the polarity values for the video sync signals. • [1] – VSYNC_POLARITY: Polarity of the vertical sync pulse. • [0] – HSYNC_POLARITY: Polarity of the horizontal sync pulse.
0x50C	RW	MAIN_STREAM_HSWIDTH_STREAM2. Sets the width of the horizontal sync pulse. • [14:0] – Horizontal sync width in clock cycles.
0x510	RW	MAIN_STREAM_VSWIDTH_STREAM2. Sets the width of the vertical sync pulse. • [14:0] – Width of the vertical sync in lines.
0x514	RW	MAIN_STREAM_HRES_STREAM2. Horizontal resolution of the main stream video source. • [15:0] – Number of active pixels per line of the main stream video.
0x518	RW	MAIN_STREAM_VRES_STREAM2. Vertical resolution of the main stream video source. • [15:0] – Number of active lines of video in the main stream video source.
0x51C	RW	MAIN_STREAM_HSTART_STREAM2. Number of clocks between the leading edge of the horizontal sync and the start of active data. • [15:0] – Horizontal start clock count.
0x520	RW	MAIN_STREAM_VSTART_STREAM2. Number of lines between the leading edge of the vertical sync and the first line of active data. • [15:0] – Vertical start line count.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x524	RW	MAIN_STREAM_MISC0_STREAM2. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC0 register described in section 2.2.4 of the standard. • [0] -Synchronous Clock. • [2:1] – Component Format. • [3] – Dynamic Range. • [4] – YCbCr Colorimetry. • [7:5] – Bit depth per color/component.
0x528	RW	MAIN_STREAM_MISC1_STREAM2. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC1 register described in section 2.2.4 of the standard. • [0] – Interlaced vertical total even. • [2:1] – Stereo video attribute. • [6:3] – Reserved.
0x52C	RW	M-VID_STREAM2. If synchronous clocking mode is used, this register must be written with the M value as described in section 2..2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream as automatically computed by the source core and written to the main stream. These values are not written into the M-VID register for readback. • [23:0] – Unsigned M value.
0x530	RW	TRANSFER_UNIT_SIZE_STREAM2. Sets the size of a transfer unit in the framing logic On reset, transfer size is set to 64. • [6:0] – This number should be in the range of 32 to 64 and is set to a fixed value that depends on the inbound video mode. Note that bit 0 cannot be written (the transfer unit size is always even).
0x534	RW	N-VID_STREAM2. If synchronous clocking mode is used, this register must be written with the N value as described in section 2..2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream as automatically computed by the source core and written to the main stream. These values are not written into the N-VID register for readback. • [23:0] – Unsigned N value.
0x538	RW	USER_PIXEL_WIDTH_STREAM2. Selects the width of the user data input port. Use only quad pixel mode in MST. • [2:0]: ◦ 1 = Single pixel wide interface ◦ 2 = Dual pixel wide interface ◦ 4 = Quad pixel wide interface

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition

0x53C	RW	USER_DATA_COUNT_PER_LANE_STREAM2. This register is used to translate the number of pixels per line to the native internal datapath.
		If (HRES * bits per pixel) is divisible by 16, then
		word_per_line = ((HRES * bits per pixel)/16)
		Else
		word_per_line = (INT((HRES * bits per pixel)/16))+1
		
		For single-lane design:
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 1
		
		For 2-lane design:
		If words_per_line is divisible by 2, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 2
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,2) - 2
		
		For 4-lane design:
		If words_per_line is divisible by 4, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 4
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,4) - 4

0x540	RW	MAIN_STREAM_INTERLACED_STREAM2. Informs the DisplayPort transmitter main link that the source video is interlaced. By setting this bit to a '1', the core will set the appropriate fields in the VBID value and Main Stream Attributes. This bit must be set to a 1 for the proper transmission of interlaced sources. • [0] – Set to a 1 when transmitting interlaced images.
0x544	RW	MIN_BYTES_PER_TU_STREAM2: Programs source to use MIN number of bytes per transfer unit. The calculation should be done based on the DisplayPort Standard. • [7:0] – Set the value to INT((LINK_BW/VIDEO_BW)*TRANSFER_UNIT_SIZE)
0x548	RW	FRAC_BYTES_PER_TU_STREAM2: Calculating MIN bytes per TU will often not be a whole number. This register is used to hold the fractional component. • [9:0] – The fraction part of ((LINK_BW/VIDEO_BW)*TRANSFER_UNIT_SIZE) scaled by 1000 is programmed in this register.

0x54C	RW	INIT_WAIT_STREAM2: This register defines the number of initial wait cycles at the start of a new line by the Framing logic. This allows enough data to be buffered in the input FIFO.
		
		If (MIN_BYTES_PER_TU <= 4)
		• [7:0] – Set INIT_WAIT to 64
		Else
		• [7:0] – Set INIT_WAIT to (TRANSFER_UNIT_SIZE - MIN_BYTES_PER_TU)

0x550	RW	MAIN_STREAM_HTOTAL_STREAM3. Specifies the total number of clocks in the horizontal framing period for the main stream video signal. • [15:0] – Horizontal line length total in clocks.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x554	RW	MAIN_STREAM_VTOTAL_STREAM3. Provides the total number of lines in the main stream video frame. • [15:0] – Total number of lines per video frame.
0x558	RW	MAIN_STREAM_POLARITY_STREAM3. Provides the polarity values for the video sync signals. • [1] – VSYNC_POLARITY: Polarity of the vertical sync pulse. • [0] – HSYNC_POLARITY: Polarity of the horizontal sync pulse.
0x55C	RW	MAIN_STREAM_HSWIDTH_STREAM3. Sets the width of the horizontal sync pulse. • [14:0] – Horizontal sync width in clock cycles.
0x560	RW	MAIN_STREAM_VSWIDTH_STREAM3. Sets the width of the vertical sync pulse. • [14:0] – Width of the vertical sync in lines.
0x564	RW	MAIN_STREAM_HRES_STREAM3. Horizontal resolution of the main stream video source. • [15:0] – Number of active pixels per line of the main stream video.
0x568	RW	MAIN_STREAM_VRES_STREAM3. Vertical resolution of the main stream video source. • [15:0] – Number of active lines of video in the main stream video source.
0x56C	RW	MAIN_STREAM_HSTART_STREAM3. Number of clocks between the leading edge of the horizontal sync and the start of active data. • [15:0] – Horizontal start clock count.
0x570	RW	MAIN_STREAM_VSTART_STREAM3. Number of lines between the leading edge of the vertical sync and the first line of active data. • [15:0] – Vertical start line count.
0x574	RW	MAIN_STREAM_MISC0_STREAM3. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC0 register described in section 2.2.4 of the standard. • [0] -Synchronous Clock. • [2:1] – Component Format. • [3] – Dynamic Range. • [4] – YCbCr Colorimetry. • [7:5] – Bit depth per color/component.
0x578	RW	MAIN_STREAM_MISC1_STREAM3. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC1 register described in section 2.2.4 of the standard. • [0] – Interlaced vertical total even. • [2:1] – Stereo video attribute. • [6:3] – Reserved.
0x57C	RW	M-VID_STREAM3. If synchronous clocking mode is used, this register must be written with the M value as described in section 2.2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream as automatically computed by the source core and written to the main stream. These values are not written into the M-VID register for readback. • [23:0] – Unsigned M value

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x580	RW	TRANSFER_UNIT_SIZE_STREAM3. Sets the size of a transfer unit in the framing logic On reset, transfer size is set to 64. • [6:0] – This number should be in the range of 32 to 64 and is set to a fixed value that depends on the inbound video mode. Note that bit 0 cannot be written (the transfer unit size is always even).
0x584	RW	N-VID_STREAM3. If synchronous clocking mode is used, this register must be written with the N value as described in section 2.2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream as automatically computed by the source core and written to the main stream. These values are not written into the N-VID register for readback. • [23:0] – Unsigned N value
0x588	RW	USER_PIXEL_WIDTH_STREAM3. Selects the width of the user data input port. Use only quad pixel mode in MST. • [2:0]: ◦ 1 = Single pixel wide interface ◦ 2 = Dual pixel wide interface ◦ 4 = Quad pixel wide interface

0x58C	RW	USER_DATA_COUNT_PER_LANE_STREAM3. This register is used to translate the number of pixels per line to the native internal 16-bit datapath.
		If (HRES * bits per pixel) is divisible by 16, then
		word_per_line = ((HRES * bits per pixel)/16)
		Else
		word_per_line = (INT((HRES * bits per pixel)/16))+1
		
		For single-lane design:
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 1
		
		For 2-lane design:
		If words_per_line is divisible by 2, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 2
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,2) - 2
		
		For 4-lane design:
		If words_per_line is divisible by 4, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 4
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,4) - 4

0x590	RW	MAIN_STREAM_INTERLACED_STREAM3. Informs the DisplayPort transmitter main link that the source video is interlaced. By setting this bit to a 1, the core will set the appropriate fields in the VBID value and Main Stream Attributes. This bit must be set to a 1 for the proper transmission of interlaced sources. • [0] – Set to a 1 when transmitting interlaced images.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x594	RW	MIN_BYTES_PER_TU_STREAM3: Programs source to use MIN number of bytes per transfer unit. The calculation should be done based on the DisplayPort Standard. • [7:0] – Set the value to INT((LINK_BW/VIDEO_BW)*TRANSFER_UNIT_SIZE)
0x598	RW	FRAC_BYTES_PER_TU_STREAM3: Calculating MIN bytes per TU will often not be a whole number. This register is used to hold the fractional component. • [9:0] – The fraction part of ((LINK_BW/VIDEO_BW)*TRANSFER_UNIT_SIZE) scaled by 1000 is programmed in this register.

0x59C	RW	INIT_WAIT_STREAM3: This register defines the number of initial wait cycles at the start of a new line by the Framing logic. This allows enough data to be buffered in the input FIFO.
		
		If (MIN_BYTES_PER_TU <= 4)
		• [7:0] – Set INIT_WAIT to 64
		Else
		• [7:0] – Set INIT_WAIT to (TRANSFER_UNIT_SIZE - MIN_BYTES_PER_TU)

0x5A0	RW	MAIN_STREAM_HTOTAL_STREAM4. Specifies the total number of clocks in the horizontal framing period for the main stream video signal. • [15:0] – Horizontal line length total in clocks.
0x5A4	RW	MAIN_STREAM_VTOTAL_STREAM4. Provides the total number of lines in the main stream video frame. • [15:0] – Total number of lines per video frame.
0x5A8	RW	MAIN_STREAM_POLARITY_STREAM4. Provides the polarity values for the video sync signals. • [1] – VSYNC_POLARITY: Polarity of the vertical sync pulse. • [0] – HSYNC_POLARITY: Polarity of the horizontal sync pulse.
0x5AC	RW	MAIN_STREAM_HSWIDTH_STREAM4. Sets the width of the horizontal sync pulse. • [14:0] – Horizontal sync width in clock cycles.
0x5B0	RW	MAIN_STREAM_VSWIDTH_STREAM4. Sets the width of the vertical sync pulse. • [14:0] – Width of the vertical sync in lines.
0x5B4	RW	MAIN_STREAM_HRES_STREAM4. Horizontal resolution of the main stream video source. • [15:0] – Number of active pixels per line of the main stream video.
0x5B8	RW	MAIN_STREAM_VRES_STREAM4. Vertical resolution of the main stream video source. • [15:0] – Number of active lines of video in the main stream video source.
0x5BC	RW	MAIN_STREAM_HSTART_STREAM4. Number of clocks between the leading edge of the horizontal sync and the start of active data. • [15:0] – Horizontal start clock count.
0x5C0	RW	MAIN_STREAM_VSTART_STREAM4. Number of lines between the leading edge of the vertical sync and the first line of active data. • [15:0] – Vertical start line count.

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x5C4	RW	MAIN_STREAM_MISC0_STREAM4. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC0 register described in section 2.2.4 of the standard. • [0] -Synchronous Clock. • [2:1] – Component Format. • [3] – Dynamic Range. • [4] – YCbCr Colorimetry. • [7:5] – Bit depth per color/component.
0x5C8	RW	MAIN_STREAM_MISC1_STREAM4. Miscellaneous stream attributes. • [7:0] – Implements the attribute information contained in the DisplayPort MISC1 register described in section 2.2.4 of the standard. • [0] – Interlaced vertical total even. • [2:1] – Stereo video attribute. • [6:3] – Reserved.
0x5CC	RW	M-VID_STREAM4. If synchronous clocking mode is used, this register must be written with the M value as described in section 2.2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream as automatically computed by the source core and written to the main stream. These values are not written into the M-VID register for readback. • [23:0] – Unsigned M value.
0x5D0	RW	TRANSFER_UNIT_SIZE_STREAM4. Sets the size of a transfer unit in the framing logic On reset, transfer size is set to 64. • [6:0] – This number should be in the range of 32 to 64 and is set to a fixed value that depends on the inbound video mode. Note that bit 0 cannot be written (the transfer unit size is always even).
0x5D4	RW	N-VID_STREAM4. If synchronous clocking mode is used, this register must be written with the N value as described in section 2.2.5.2 of the standard. When in asynchronous clocking mode, the M value for the video stream as automatically computed by the source core and written to the main stream. These values are not written into the N-VID register for readback. • [23:0] – Unsigned N value.
0x5D8	RW	USER_PIXEL_WIDTH_STREAM4. Selects the width of the user data input port. Use only quad pixel mode in MST. • [2:0]: ◦ 1 = Single pixel wide interface ◦ 2 = Dual pixel wide interface ◦ 4 = Quad pixel wide interface

Table 2-12: **DisplayPort Source Core Configuration Space *(Cont'd)***

Offset	R/W	Definition

0x5DC	RW	USER_DATA_COUNT_PER_LANE_STREAM4. This register is used to translate the number of pixels per line to the native internal 16-bit datapath.
		If (HRES * bits per pixel) is divisible by 16, then
		word_per_line = ((HRES * bits per pixel)/16)
		Else
		word_per_line = (INT((HRES * bits per pixel)/16))+1
		
		For single-lane design:
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 1
		
		For 2-lane design:
		If words_per_line is divisible by 2, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 2
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,2) - 2
		
		For 4-lane design:
		If words_per_line is divisible by 4, then
		Set USER_DATA_COUNT_PER_LANE = words_per_line - 4
		Else
		Set USER_DATA_COUNT_PER_LANE = words_per_line + MOD(words_per_line,4) - 4

0x5E0	RW	MAIN_STREAM_INTERLACED_STREAM4. Informs the DisplayPort transmitter main link that the source video is interlaced. By setting this bit to a 1, the core will set the appropriate fields in the VBID value and Main Stream Attributes. This bit must be set to a 1 for the proper transmission of interlaced sources. • [0] – Set to a 1 when transmitting interlaced images.
0x5E4	RW	MIN_BYTES_PER_TU_STREAM4. Programs source to use MIN number of bytes per transfer unit. The calculation should be done based on the DisplayPort Standard. • [7:0] – Set the value to INT((LINK_BW/VIDEO_BW)*TRANSFER_UNIT_SIZE)
0x5E8	RW	FRAC_BYTES_PER_TU_STREAM4. Calculating MIN bytes per TU will often not be a whole number. This register is used to hold the fractional component. • [9:0] – The fraction part of ((LINK_BW/VIDEO_BW)*TRANSFER_UNIT_SIZE) scaled by 1000 is programmed in this register.

0x5EC	RW	INIT_WAIT_STREAM4. This register defines the number of initial wait cycles at the start of a new line by the Framing logic. This allows enough data to be buffered in the input FIFO.
		If (MIN_BYTES_PER_TU <= 4):
		• [7:0] – Set INIT_WAIT to 64
		Else
		• [7:0] – Set INIT_WAIT to (TRANSFER_UNIT_SIZE - MIN_BYTES_PER_TU)

0x800 - 0x8FF	WO	PAYLOAD_TABLE. This address space maps to the VC payload table that is maintained in the core. • [7:0] – Payload data

DisplayPort Audio

The DisplayPort Audio registers are listed in Table 2-13.

Table 2-13: **DisplayPort Audio Registers**

Offset	R/W	Definition
0x300	R/W	TX_AUDIO_CONTROL. Enables audio stream packets in main link and provides buffer control. • [0]: Audio Enable
0x304	R/W	TX_AUDIO_CHANNELS. Used to input active channel count. Transmitter collects audio samples based on this information. • [2:0] Channel Count
0x308	Write Only	TX_AUDIO_INFO_DATA. [31:0] Word formatted as per CEA 861-C Info Frame. Total of eight words should be written in following order: • 1st word – ◦ [7:0] = HB0 ◦ [15:8] = HB1 ◦ [23:16] = HB2 ◦ [31:24] = HB3 • 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. No protection is provided for wrong operations by software.
0x328	R/W	TX_AUDIO_MAUD. M value of audio stream as computed by transmitter. • [23:0] = Unsigned value computed when audio clock and link clock are synchronous.

Table 2-13: **DisplayPort Audio Registers *(Cont'd)***

Offset	R/W	Definition
0x32C	R/W	TX_AUDIO_NAUD. N value of audio stream as computed by transmitter. • [23:0] = Unsigned value computed when audio clock and link clock are synchronous.
0x330 - 0x350	WO	TX_AUDIO_EXT_DATA. [31:0] = Word formatted as per Extension packet described in protocol standard. Extended packet is fixed to 32 Bytes length. The controller has buffer space for only one extended packet. Extension packet address space can be used to send the audio Copy management packet/ISRC packet/VSC packets. TX is capable of sending any of these packets. A total of nine words should be written in following order: • 1st word - ◦ [7:0] = HB0 ◦ [15:8] = HB1 ◦ [23:16] = HB2 ◦ [31:24] = HB3 • 2nd word - DB3,DB2,DB1,DB0 ... • 9th word -DB31,DB30,DB29,DB28 See the DisplayPort Standard for HB* definition. No protection is provided for wrong operations by software. This is a key-hole memory. So, nine writes to this address space is required.

Sink Core

The DisplayPort Configuration Data is implemented as a set of distributed registers which may 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 may change while being read from the configuration space, two scenarios may 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 may 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 2-14 are to be considered reserved and will return '0' upon read. Only address offsets are listed in Table 2-14. Base addresses are configured by the AXI Interconnect.

Table 2-14: **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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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). • [7:0] – Clock divider value. • [15:8] – The number of AXI-Lite clocks (defined by the AXI-Lite clock name: s_axi_aclk) equivalent to the recommended width of AUX pulse. Allowable values include: 8,16,24,32,40 and 48. From DP Protocol spec, AUX Pulse Width range = 0.4 to 0.6 us. For example, for AXI Lite clock of 50MHz (=20ns), 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.
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 only 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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

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

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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 us 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.
0x03C	RO	REQUEST_LENGTH. The length of the most recent AUX request is written to this register. The length of the AUX request is the value of this register plus one. • [3:0] – Contains the length of the AUX request. Transaction lengths from 1 to 16 bytes are supported. For address only transactions, the value of this register will be 0.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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. • [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 2-14: **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
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

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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 may 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.
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	• [14:0] – Training pattern1 (TP1) score to override. • [31] – Set to override Training Pattern 1 (TP1) score.
0x068	RW	• [12:0] – Training pattern2/3 (TP2/TP3) score to override. • [31] – Set to override the Training Pattern2/3 scores.
0x06C	RO	Provides the contents of DPCD registers 0x1C0, 0x1C1 and 0x1C2 • [5:0] – VC Payload ID • [13:8] – Starting Time Slot • [21:16] – Time slot count
0x074	RW	[5] – Enable the CRC support. The CRC has to be calculated outside the DP 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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x090	RW	DEVICE_SERVICE_IRQ. Indicates DPCD DEVICE_SERVICE_IRQ_VECTOR state. • [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. • [1] – Reflects SINK_SPECIFIC_IRQ state of DPCD 0x201 register. This bit is RO. • [4] – Set to 1 to indicate a new DOWN Reply Buffer Message is ready.
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.
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. • [4:0] – Link rate override value for DisplayPort Standard v1.2 designs • [3:0] – Link rate override value for DisplayPort Standard v1.1a designs ◦ 0x6 - 1.62 G ◦ 0xA - 2.7 G ◦ 0x14 - 5.4 G
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. • [4:0] – Lane count override value (1, 2 or 4 lanes • [6] – TPS3_SUPPORTED: Capability override for DisplayPort Standard v1.2 protocol designs only. Reserved for v1.1a protocol. • [7] – ENHANCED_FRAME_CAP: Capability override
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. • [1:0] – TRAINING_PATTERN_SELECT Override • [3:2] – LINK_QUAL_PATTERN_SET Override for DisplayPort Standard v1.1a only. • [4] – RECOVERED_CLOCK_OUT_EN Override • [5] – SCRAMBLING_DISABLE Override • [7:6] – SYMBOL ERROR COUNT SEL Override • [15:8] – TRAINING_AUX_RD_INTERVAL (the values are based on DisplayPort Standard v1.2 protocol).

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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. • [1:0] – VOLTAGE SWING SET override • [2] – MAX_SWING_REACHED override • [4:3] – PRE-EMPHASIS_SET override • [5] – MAX_PRE-EMPHASIS_REACHED override • [7:6] – Reserved
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.
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.
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.
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

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x0D0	RW	• [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. • [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. • [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. • [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. • [5-7] – Unused • [8] – Clears the VCPayload Table contents. This bit is auto cleared.
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 setup GUID if required from host interface. Valid for DPCD1.2 version only. • [31:0] – Lower 4 bytes of GUID DPCD field
0x0E4	RW	GUID word 1. Allows you to setup 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 setup 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 setup 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		
0x0F8	RO	VERSION Register. For example, for displayport_v6_0, the VERSION REGISTER will be 32'h06_00_0_0_00. • [31:24] – Core major version • [23:16] – Core minor version • [15:12] – Core version revision • [11:8] – Core Patch details • [7:0] – Internal revision

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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
0x110	RO	USER_FIFO_OVERFLOW. This status bit indicates an overflow of the user data FIFO of pixel data. This event may occur if the input pixel clock is not fast enough to support the current DisplayPort link width and link speed. • [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. • [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. • [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. • [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. • [4] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the Video unpack FIFO is overflowed. • [5] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the Video unpack FIFO is overflowed for stream2. • [6] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the Video unpack FIFO is overflowed for stream3. • [7] – Video Unpack FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the Video unpack FIFO is overflowed for stream4. • [8] – Video Timing FIFO_OVERFLOW_FLAG (Stream 1): 1 indicates that the Video Timing FIFO is overflowed. • [9] – Video Timing FIFO_OVERFLOW_FLAG (Stream 2): 1 indicates that the Video Timing FIFO is overflowed for stream2. • [10] – Video Timing FIFO_OVERFLOW_FLAG (Stream 3): 1 indicates that the Video Timing FIFO is overflowed for stream3. • [11] – Video Timing FIFO_OVERFLOW_FLAG (Stream 4): 1 indicates that the Video Timing FIFO is overflowed for stream4.
0x114	RO	USER_VSYNC_STATE. Provides a mechanism for the host processor to monitor the state of the video data path. This bit is set when vsync is asserted. • [0] – State of the vertical sync pulse for Stream 1. • [1] – State of the vertical sync pulse for Stream 2. • [2] – State of the vertical sync pulse for Stream 3. • [3] – State of the vertical sync pulse for Stream 4.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
PHY Configuration and Status		
0x200	RW	PHY_CONFIG. • [1:0] – When set a value of 0x3, the receiver PHY will be held in reset. This value must be set to a value of 0 before the receiver core will function properly. Controls the reset to the PHY section of the DisplayPort receiver core. At power up, this register has a value of 0x3: ◦ [0]: GT PLL Reset ◦ [1]: GT Rx Reset • [8] – Set to 1 for RX_PHY_PMA Reset. Clear to release. • [9] – Set to 1 for RX_PHY_PCS Reset. Clear to release. • [10] – Set to 1 for RX_PHY_BUF Reset. Clear to release. • [11] – Set to 1 for RX_PHY_DFE_LPM Reset. Clear to release. • [12] – Set to configure RX_PHY_POLARITY. Default is 0. • [15:13] – Set to configure RX_PHY_LOOPBACK. Default is 0. • [16] – Set to 1 for RX_PHY_EYESCANRESET. Clear to release. • [17] – Set to configure RX_PHY_EYESCANTRIGGER. • [18] – Set to 1 for RX_PHY_PRBSCNTRESET. Clear to release. • [19] – Set to configure RX_PHY_RXLPMHFHOLD. Default is 0. • [20] – Set to configure RX_PHY_RXLPMLFHOLD. Default is 0. • [21] – Set to configure RX_PHY_RXLPMHFOVERDEN. Default is 0. • [22] – Set to configure RX_PHY_CDRHOLD. Default is 0. • [23] – Set to 1 for issuing reset at every training iteration. Recommended value is 1. Advanced users can modify. • [24] – Set to 1 for issuing reset at every link rate change. Recommended value is 1. Advanced users can modify. • [25] – Set to 1 for issuing reset at start of training pattern 1. Recommended value is 1. Advanced users can modify. • [26] – Set to enable the individual lane polarity. Set to 0 to use common polarity control through bit [12] for all lanes. • [27] – Set to configure RX_PHY_POLARITY for Lane 0. • [28] – Set to configure RX_PHY_POLARITY for Lane 1. • [29] – Set to configure RX_PHY_POLARITY for Lane 2. • [30] – Set to configure RX_PHY_POLARITY for Lane 3. See the appropriate transceiver user guide for programming details.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x208	RO	PHY_STATUS. Provides status for the receiver core PHY. • [1:0] – Reset done for lanes 0 and 1 (Tile 0). • [3:2] – Reset done for lanes 2 and 3 (Tile 1). • [4] – PLL for lanes 0 and 1 locked (Tile 0). • [5] – PLL for lanes 2 and 3 locked (Tile 1). • [6] – FPGA fabric clock PLL locked. • [7] – Receiver Clock locked. • [9:8] – PRBS error, lanes 0 and 1. • [11:10] – PRBS error, lanes 2 and 3. • [13:12] – RX voltage low, lanes 0 and 1. • [15:14] – RX voltage low, lanes 2 and 3. • [20] – Symbol lock, lane 0. • [21] – Symbol lock, lane 1. • [22] – Symbol lock, lane 2. • [23] – Symbol lock, lane 3. • [25:24] – RX buffer status, lane 0. • [27:26] – RX buffer status, lane 1. • [29:28] – RX buffer status, lane 2. • [31:30] – RX buffer status, lane 3.
0x210	RW	• [3:0] – RX_PHY_POWER_DOWN. These bits allow the receiver core to conditionally power down specific lanes of the PHY if supported for a particular technology implementation. These bits should be written only after the training process has been completed and the link is stable. ◦ [3] – LANE_3_POWER_DOWN: Set o a 1 to power down the PHY for lane 3. ◦ [2] – LANE_2_POWER_DOWN: Set to a 1 to power down the PHY for lane 2. ◦ [1] – LANE_1_POWER_DOWN: Set to a 1 to power down the PHY for lane 1. ◦ [0] – LANE_0_POWER_DOWN: Set to a 1 to power down the PHY for lane 0. • [19:4] – GT PCSRSVDIN value. Used only in UltraScale™ devices.

Table 2-14: **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. • [1:0] – The minimum voltage swing setting matches the values defined in the DisplayPort Standard for the TRAINING_LANEX_SET register. • [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 • [6:4] – VSWING_SWEEP_CNT (only for Advanced users). • [9:8] – SET_VSWING (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 • [13:12] – SET_PREEMP (only for Advanced users). • [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
0x21C	RW	CDR_CONTROL_CONFIG. • [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'h11364. • [31]: Use DFE Control. Applicable only for 5.4G in GTX/GTH. Default value is 1'b1.
0x2A0	RW	GT_DRP_COMMAND. Provides access to GT DRP ports. All channels use same programming. • [7:0] – DRP Address • [15] – DRP Write/Read Command ◦ 1: Write ◦ 0: Read • [31:16] – DRP Write Data (not valid for read command)
0x2A4	RO	GT_DRP_READ_DATA: Provides access to GT DRP READ Data. The data is sampled when DRP ready is asserted. • [15:0] – DRP Read Data. After issuing DRP Command register, software should wait for some time (typically 10 * AXI4-Lite Clock Period) to ensure DRP access is completed before reading the data.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x2A8	RO	GT_DRP_CHANNEL_STATUS: Provides access to GT DRP CHANNEL STATUS. • [0] – DRP Locked. Locked is asserted when IP state machine uses GT DRP. Software polls this bit and initiate read/write transaction only when the locked bit is set to 0.
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: [7:0] = HB0 [15:8] = HB1 [23:16] = HB2 [31:24] = HB3 • 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 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
12'h330- 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 - [7:0] = HB0 [15:8] = HB1 [23:16] = HB2 [31:24] = HB3 • 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
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

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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. • [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 • [2] – MAX_SWING_REACHED: Set to '1' when the maximum driven current 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 • [5] – MAX_PRE-EMPHASIS_REACHED: Set to 1 when the maximum pre-emphasis setting is reached.
0x424	RO	DPCD_TRAINING_LANE_1_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. The fields of this register are identical to DPCD_TRAINING_LANE_0_SET.
0x428	RO	DPCD_TRAINING_LANE_2_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. The fields of this register are identical to DPCD_TRAINING_LANE_0_SET.
0x42C	RO	DPCD_TRAINING_LANE_3_SET. Used by the transmitter during link training to configure the receiver PHY for lane 0. The fields of this register are identical to DPCD_TRAINING_LANE_0_SET.
0x430	RO	DPCD_DOWNSPREAD_CONTROL. The transmitter uses this bit to inform the receiver core that downspreading has been enabled. • [0] – SPREAD_AMP: Set to 1 for 0.5% spreading or 0 for none.
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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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 • [5] – LANE_1_CHANNEL_EQ_DONE • [4] – LANE_1_CLOCK_RECOVERY_DONE • [2] – LANE_0_SYMBOL_LOCKED • [1] – LANE_0_CHANNEL_EQ_DONE • [0] – LANE_0_CLOCK_RECOVERY_DONE
0x440	RO	DPCD_LANE23_STATUS. Value of the lane 2 and lane 3 training status registers. • [6] – LANE_3_SYMBOL_LOCKED • [5] – LANE_3_CHANNEL_EQ_DONE • [4] – LANE_3_CLOCK_RECOVERY_DONE • [2] – LANE_2_SYMBOL_LOCKED • [1] – LANE_2_CHANNEL_EQ_DONE • [0] – LANE_2_CLOCK_RECOVERY_DONE
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 may 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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
0x508	RO	MSA_HSWIDTH. Specifies the width of the horizontal sync pulse. • [14:0] – Specifies the width of the horizontal sync in terms of the recovered video clock.
0x50C	RO	MSA_HSTART. This main stream attribute is the number of clock cycles between the leading edge of the horizontal sync and the first cycle of active data. • [15:0] – Number of blanking cycles before active data.
0x510	RO	MSA_HTOTAL. Tells the receiver core how many video clock cycles 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

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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
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.

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
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

Table 2-14: **DisplayPort Sink Core Configuration Space *(Cont'd)***

Offset	R/W	Definition
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.
0xA00 - 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 - 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 - 0xCFF	RO	UPSTREAM_REQUEST_BUFFER. Reserved for future.
0xD00 - 0xDFF	WO	UPSTREAM_REPLY_BUFFER. Reserved for future.
0x800 - 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		
0xE00-0xEFC	RW	SOURCE_DEVICE_SPECIFIC_FIELD. User access to Source specific field of DPCD address space. AXI accesses are all word-based (32 bits). • 0xE00 - 0xE02 : Read Only (IEEE OUI Value Programmed by Source) • 0xE03 - 0xEFF : Write/Read
0xF00-0xFFC	RW	SINK_DEVICE_SPECIFIC_FIELD. User access to Sink specific field of DPCD address space. AXI accesses are all word-based (32 bits). • 0xF00 - 0xF02 : Read Only (IEEE OUI Value from GUI) • 0xF03 - 0xFFF : Write/Read

Designing with the Core

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

Source Overview

The Source core moves a video stream from a standardized main link through a complete DisplayPort Link Layer, and onto High-Speed Serial I/O for transport to a Sink device.

Main Link Setup and Management

This section is intended to elaborate on and act as a companion to the link training procedure, described in section 3.5.1.3 of the *VESA DisplayPort Standard* v1.2 [Ref 2].

For your convenience, the DisplayPort Source core comes with an example controller design. The first is a simple RTL-based state machine that might be used to quickly demonstrate the proper startup procedure. This is provided because simulating the full Policy Maker example design requires many hours of simulation to complete. The RTL-based state machine should only be used for simulation and for establishing a quick link with the Xilinx Sink core. This controller is not expected to interoperate with other standard products.

If you require more capability and tuning, the reference Link Policy Maker is available as full C source code in the *DisplayPort Transmit Reference Design Application Note* (XAPP1178) [Ref 8]. The Policy Maker sets up and maintains the link with varying levels of interaction done by you. If you decide to use the provided software, this section may be treated as reference.

Regardless of whether the provided Policy Maker is used, Xilinx advises all users of the source core to use a MicroBlaze™ processor or similar embedded processor to properly initialize and maintain the link. The tasks encompassed in the Link and Stream Policy Makers are likely too complicated to be efficiently managed by a hardware-based state machine.

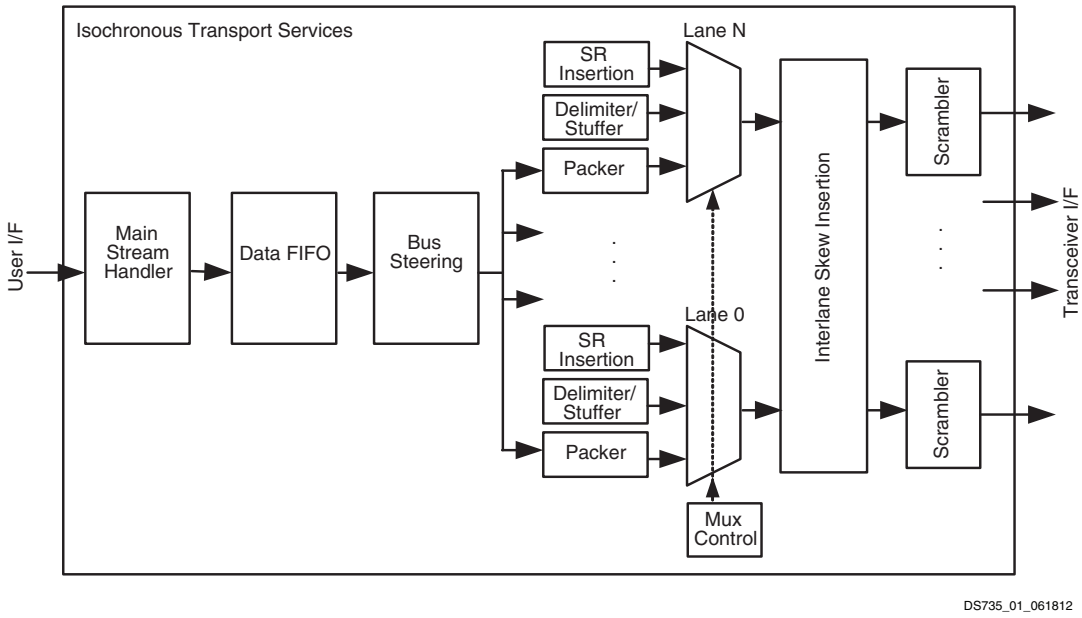

Figure 3-1: **Source Main Link Datapath**

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 data path is blocked and the link training controller issues the specified pattern. Care must be taken to place the Sink device in the proper link training mode before the source state machine enters a training state. Otherwise, unpredictable results may occur.

Figure 3-2 shows the flow diagram for link training. For details, refer to the *VESA DisplayPort Standard v1.2*[Ref 2].

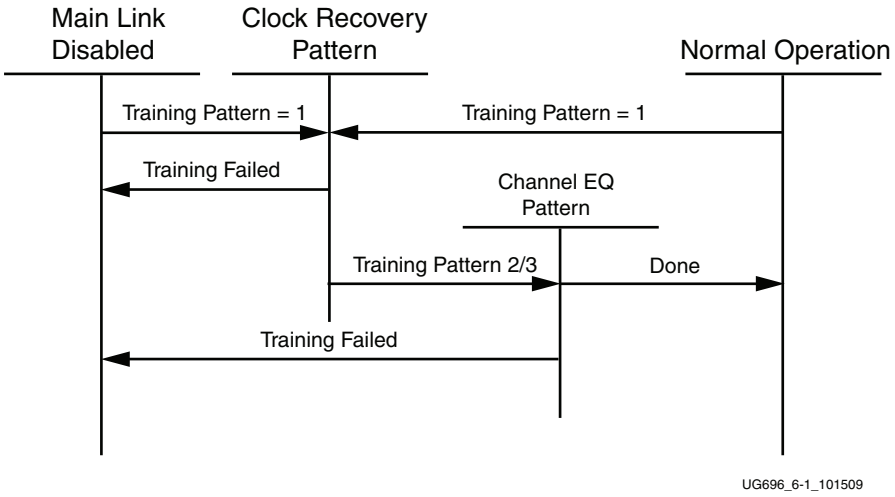

Figure 3-2: **Link Training States**

Source Core Setup and Initialization

The following text contains the procedural tasks required to achieve link communication. For description of the DPCD, see VESA DisplayPort Standard v1.2.

IMPORTANT: *During initialization ensure that TX PHY 8b10b coding enable bit is not cleared in offset 0x200.*

Source Core Setup

1. Place the PHY into reset.
   - PHY_RESET = 0x01
2. Disable the transmitter.
   - TRANSMITTER_ENABLE = 0x00
3. Set the clock divider.
   - AUX_CLOCK_DIVIDER = (see register description for proper value)
4. Set DisplayPort clock speed.
   - PHY_CLOCK_SELECT = desired link speed
5. Bring the PHY out of reset.
   - PHY_RESET = 0x00
6. Wait for the PHY to be ready.
   - (PHY_STATUS & 0x3F) == 0x3F
7. Enable the transmitter.
   - TRANSMITTER_ENABLE = 0x01
8. (Optional) Turn on the interrupt mask for HPD.
   - INTERRUPT_MASK = 0x00

Note: At this point, the source core is initialized and ready to use. The link policy maker should be monitoring the status of HPD and taking appropriate action for connect / disconnect events or HPD interrupt pulses.

Upon HPD Assertion

1. Read the DPCD capabilities fields out of the sink device (0x00000 - 0x0000B) via the AUX channel.
2. Determine values for lane count, link speed, enhanced framing mode, downspread control and main link channel code based on each link partners' capability and needs.
3. Write the configuration parameters to the link configuration field (0x00100 - 0x00101) of the DPCD via the AUX channel.

Note: Some sink devices' DPCD capability fields are unreliable. Many source devices start with the maximum transmitter capabilities and scale back as necessary to find a configuration the sink device can handle. This could be an advisable strategy instead of relying on DPCD values.

4. Equivalently, write the appropriate values to the Source core's local configuration space.
    a. LANE_COUNT_SET
    b. LINK_BW_SET
    c. ENHANCED_FRAME_EN
    d. PHY_CLOCK_SELECT

Training Pattern 1 Procedure (Clock Recovery)

1. Turn off scrambling and set training pattern 1 in the source via direct register writes.
    - SCRAMBLING_DISABLE = 0x01
    - TRAINING_PATTERN_SET = 0x01
2. Turn off scrambling and set training pattern 1 in the sink DPCD (0x00102 - 0x00106) via the AUX channel.
3. Wait for the aux read interval configured in TRAINING_AUX_RD_INTERVAL DPCD Register (0x0000E) before reading status registers for all active lanes (0x00202 - 0x00203) via the AUX channel.
4. If clock recovery failed, check for voltage swing or preemphasis level increase requests (0x00206 -0x00207) and react accordingly.
    - Run this loop up to five times. If after five iterations this has not succeeded, reduce link speed if at high speed and try again. If already at low speed, training fails.

Training Pattern 2 Procedure (Symbol Recovery, Interlane Alignment)

1. Turn off scrambling and set training pattern 2 in the source via direct register writes.
    - SCRAMBLING_DISABLE = 0x01
    - TRAINING_PATTERN_SET = 0x02
2. Turn off scrambling and set training pattern 2 in the sink DPCD (0x00102 - 0x00106) via the AUX channel.
3. Wait for aux read interval configured in TRAINING_AUX_RD_INTERVAL DPCD Register (0x0000E) then read status registers for all active lanes (0x00202 - 0x00203) via the AUX channel.
4. Check the channel equalization, symbol lock, and interlane alignment status bits for all active lanes (0x00204) via the AUX channel.
5. If any of these bits are not set, check for voltage swing or preemphasis level increase requests (0x00206 -0x00207) and react accordingly.

6. Run this loop up to five times. If after five iterations this has not succeeded, reduce link speed if at high speed and Return to the instructions for Training Pattern 1. If already at low speed, training fails.
7. Signal the end of training by enabling scrambling and setting training pattern to 0x00 in the sink device (0x00102) via the AUX channel.
8. On the source side, re-enable scrambling and turn off training.
   - TRAINING_PATTERN_SET = 0x00
   - SCRAMBLING_DISABLE = 0x00

At this point, training has completed.

Note: Training pattern 3 replaces training pattern 2 for 5.4 G link rate devices. See the DisplayPort Standard v1.2 for details.

Enabling Main Link Video

Main link video should not be enabled until a proper video source has been provided to the source core. Typically the source device will want to read the EDID from the attached sink device to determine its capabilities, most importantly its preferred resolution and other resolutions that it supports should the preferred mode not be available. Once a resolution has been determined, set the Main Stream Attributes in the source core (0x180 - 0x1B0). Enable the main stream (0x084) only when a reliable video source is available.

IMPORTANT: *When the main link video is enabled, the scrambler/de-scrambler must be reset every 512th BS Symbol as described in section 2.2.1.1 of the DisplayPort standard. For simulation purposes, you should force a scrambler reset by writing a '1' to 0x0c0 before the main link is enabled to reduce the amount of time after startup needed to align the scramber/de-scrambler.*

Accessing the Link Partner

The DisplayPort core is configured through the AXI4-Lite host interface. The host processor interface uses the DisplayPort AUX Channel to read the register space of the attached sink device and determines the capabilities of the link. Accessing DPCD and EDID information from the Sink is done by writing and reading from register space 0x100 through 0x144. (For information on the DPCD register space, refer to the *VESA DisplayPort Standard v1.2*.)

Before any AUX channel operation may be completed, you must first set the proper clock divide value in 0x10C. This must be done only one time after a reset. The value held in this register should be equal to the frequency of `s_axi_aclk`. So, if `s_axi_aclk` runs at 135 MHz, the value of this register should be 135 ('h87). This register is required to apply a proper divide function for the AUX channel sample clock, which must operate at 1 MHz.

The act of writing to the AUX_COMMAND initiates the AUX event. Once an AUX request transaction is started, the host should not write to any of the control registers until the REPLY_RECEIVED bit is set to '1,' indicating that the sink has returned a response.

AUX Write Transaction

An AUX write transaction is initiated by setting up the AUX_ADDRESS, and writing the data to the AUX_WRITE_FIFO followed by a write to the AUX_COMMAND register with the code 0x08. Writing the command register begins the AUX channel transaction. The host should wait until either a reply received event or reply timeout event is detected. These events are detected by reading INTERRUPT_STATUS registers (either in ISR or polling mode).

When the reply is detected, the host should read the AUX_REPLY_CODE register and look for the code 0x00 indicating that the AUX channel has successfully acknowledged the transaction.

Figure 3-3 shows a flow of an AUX write transaction.

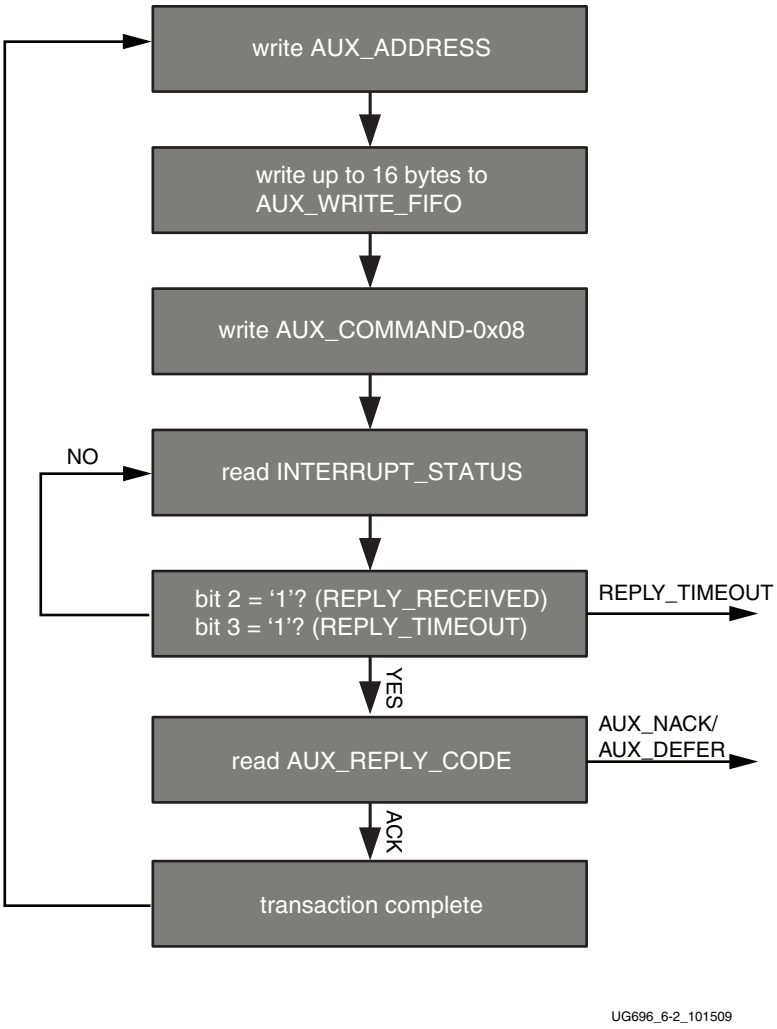

Figure 3-3: **AUX Write Transaction**

AUX Read Transaction

The AUX read transaction is prepared by writing the transaction address to the AUX_ADDRESS register. Once set, the command and the number of bytes to read are written to the AUX_COMMAND register. After initiating the transfer, the host should wait for an interrupt or poll the INTERRUPT_STATUS register to determine when a reply is received.

When the REPLY_RECEIVED signal is detected, the host may then read the requested data bytes from the AUX_REPLY_DATA register. This register provides a single address interface to a byte FIFO which is 16 elements deep. Reading from this register automatically advances the internal read pointers for the next access.

Figure 3-4 shows a flow of an AUX read transaction.

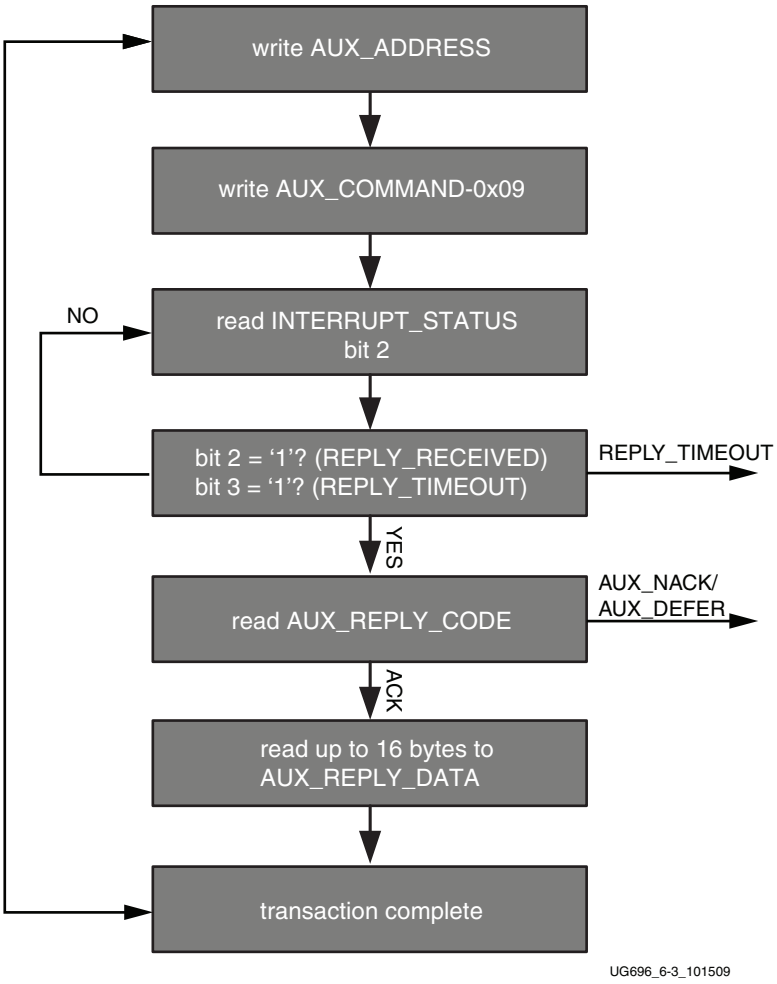

Figure 3-4: **AUX Read Transaction**

Commanded I2C Transactions

The core supports a special AUX channel command intended to make I2C over AUX transactions faster and easier to perform. In this case, the host will bypass the external I2C master/slave interface and initiate the command by directly writing to the register set.

The sequence for performing these transactions is exactly the same as a native AUX channel transaction with a change to the command written to the AUX_COMMAND register. The supported I2C commands are summarized in Table 3-1.

Table 3-1: **I2C over AUX Commands**

AUX_COMMAND[11:8]	Command
0x0	IIC Write
0x4	IIC Write MOT
0x1	IIC Read
0x5	IIC Read MOT
0x6	IIC Write Status with MOT
0x2	IIC Write Status

By using a combination of these commands, the host may emulate an I2C transaction.

Figure 3-5 shows the flow of commanded I2C transactions.

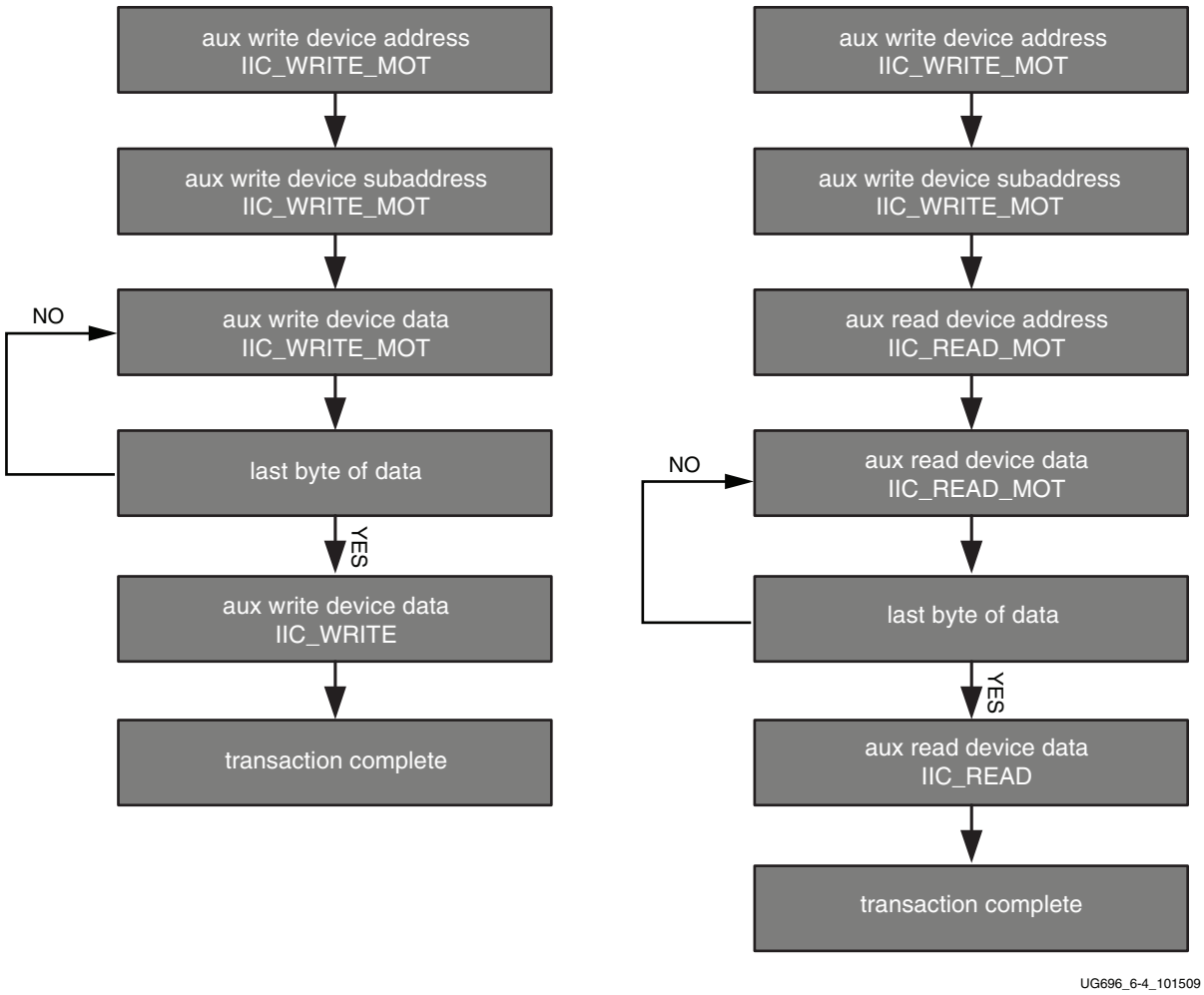

Figure 3-5: **Commanded I2C Device Transactions, Write (Left) and Read (Right)**

Since I2C transactions may be significantly slower than AUX channel transactions, the host should be prepared to receive multiple AUX_DEFER reply codes during the execution of the above state machines.

The AUX-I2C commands are as follows:

- MOT Definition:
  - Middle Of Transaction bit in the command field.
  - This controls the stop condition on the I2C slave.
  - For a transaction with MOT set to 1, the I2C bus is not STOPPED, but left to remain the previous state.
  - For a transaction with MOT set to 0, the I2C bus is forced to IDLE at the end of the current command or in special Abort cases.

- Partial ACK:
  - For I2C write transactions, the Sink core can respond with a partial ACK (ACK response followed by the number of bytes written to I2C slave).

Special AUX commands include:

- Write Address Only and Read Address Only: These commands do not have any length field transmitted over the AUX channel. The intent of these commands are to:
  - Send address and RD/WR information to I2C slave. No Data is transferred.
  - End previously active transaction, either normally or through an abort.

  The Address Only Write and Read commands are generated from the source by using bit [12] of the command register with command as I2C WRITE/READ.

- Write Status: This command does not have any length information. The intent of the command is to identify the number of bytes of data that have been written to an I2C slave when a Partial ACK or Defer response is received by the source on a AUX-I2C write.

  The Write status command is generated from the source by using bit [12] of the command register with command as I2C WRITE STATUS.

- IIC Timeout: The sink controller monitors the IIC bus after a transaction starts and looks for an IIC stop occurrence within 1 second. If an IIC stop is not received, it is considered as an IIC timeout and the sink controller issues a stop condition to release the bus. This timeout avoids a lock-up scenario.

Generation of AUX transactions are described in Table 3-2.

Table 3-2: **Generation of AUX Transactions**

Transaction	AUX Transaction	I2C Transaction	Usage	Sequence
Write Address only with MOT = 1	START -> CMD -> ADDRESS -> STOP	START -> DEVICE_ADDR -> WR -> ACK/NACK	Setup I2C slave for Write to address defined	1. Write AUX Address register(0x108) with device address. 2. Issue command to transmit transaction by writing into AUX command register (0x100). Bit [12] must be set to 1.
Read Address only with MOT = 1	START -> CMD -> ADDRESS -> STOP	START -> DEVICE_ADDR -> RD -> ACK/NACK	Setup I2C slave for Read to address defined.	1. Write AUX Address register with device address. 2. Issue command to transmit transaction by writing into AUX command register. Bit [12] must be set to 1.
Write / Read Address only with MOT = 0	START -> ADDRESS -> STOP	STOP	To stop the I2C slave, used as Abort or normal stop.	1. Write AUX Address register (0x108) with device address. 2. Issue command to transmit transaction by writing into AUX command register (0x100). Bit [12] must be set to 1.
Write with MOT = 1	START -> CMD -> ADDRESS -> LENGTH -> D0 to DN -> STOP	I2C bus is IDLE or New device address START -> START/RS -> DEVICE_ADDR -> WR -> ACK/NACK -> DATA0 -> ACK/NACK to DATAN -> ACK/NACK I2C bus is in Write state and the same device address DATA0 -> ACK/NACK to DATAN -> ACK/NACK	Setup I2C slave write data.	1. Write AUX Address register(0x108) with device address. 2. Write the data to be transmitted into AUX write FIFO register (0x104). 3. Issue write command and data length to transmit transaction by writing into AUX command register (0x100). Bits [3:0] represent length field.

Table 3-2: **Generation of AUX Transactions *(Cont'd)***

Transaction	AUX Transaction	I2C Transaction	Usage	Sequence
Write with MOT = 0	START -> CMD -> ADDRESS -> LENGTH -> D0 to DN -> STOP	I2C bus is IDLE or Different I2C device address START -> START/RS -> DEVICE_ADDR -> WR -> ACK/NACK -> DATA0 -> ACK/NACK to DATAN -> ACK/NACK -> STOP I2C bus is in Write state and the same I2C device address DATA0 -> ACK/NACK to DATAN -> ACK/NACK -> STOP	Setup I2C slave write data and stop the I2C bus after the current transaction.	1. Write AUX Address register (0x108) with device address. 2. Write the data to be transmitted into AUX write FIFO register (0x104). 3. Issue write command and data length to transmit transaction by writing into AUX command register (0x100). Bits [3:0] represent length field.
Read with MOT = 1	START -> CMD -> ADDRESS -> LENGTH -> STOP	I2C bus is IDLE or Different I2C device address START -> START/RS -> DEVICE_ADDR -> RD -> ACK/NACK -> DATA0 -> ACK/NACK to DATAN -> ACK/NACK I2C bus is in Write state and the same I2C device address DATA0 -> ACK/NACK to DATAN -> ACK/NACK	Setup I2C slave read data.	1. Write AUX Address register (0x108) with device address. 2. Issue read command and data length to transmit transaction by writing into AUX command register (0x100). Bits [3:0] represent the length field.

Table 3-2: **Generation of AUX Transactions *(Cont'd)***

Transaction	AUX Transaction	I2C Transaction	Usage	Sequence
Read with MOT = 0	START -> CMD -> ADDRESS -> LENGTH -> D0 to DN -> STOP	I2C bus is IDLE or Different I2C device address START -> START/RS -> DEVICE_ADDR -> RD -> ACK/NACK -> DATA0 -> ACK/NACK to DATAN -> ACK/NACK -> STOP I2C bus is in Write state and the same I2C device address DATA0 -> ACK/NACK to DATAN -> ACK/NACK -> STOP	Setup I2C slave read data and stop the I2C bus after the current transaction.	1. Write AUX Address register (0x108) with device address. 2. Issue read command and data length to transmit transaction by writing into AUX command register (0x100). Bits [3:0] represent the length field.
Write Status with MOT = 1	START -> CMD -> ADDRESS -> STOP	No transaction	Status of previous write command that was deferred or partially ACKED.	1. Write AUX Address register (0x108) with device address. 2. Issue status update command to transmit transaction by writing into AUX command register (0x100). Bit [12] must be set to 1.
Write Status with MOT = 0	START -> CMD -> ADDRESS -> STOP	Force a STOP and the end of write burst	Status of previous write command that was deferred or partially ACKED. MOT = 0 will ensure the bus returns to IDLE at the end of the burst.	1. Write AUX Address register (0x108) with device address. 2. Issue status update command to transmit transaction by writing into AUX command register (0x100). Bit [12] must be set to 1.

Handling I2C Read Defers/Timeout:

- The Sink core could issue a DEFER response for a burst read to I2C. The following are the actions that can be taken by the Source core.
  - Issue the same command (previously issued read, with same device address and length) and wait for response. The Sink core on completion of the read from I2C (after multiple defers) should respond with read data.
  - Abort the current read using:
    - Read to a different I2C slave
    - Write command
    - Address-only Read or write with MOT = 0.

Handling I2C Write Partial ACK:

- The sink could issue a partial ACK response for a burst Write to I2C. The following are the actions that can be taken by the Source core:
  - Use the Write status command to poll the transfers happening to the I2C. On successful completion, the sink should issue an NACK response to these requests while intermediate ones will get partial ACK.
  - Issue the same command (previously issued with the same device address, length and data) and wait for response. On completion of the write to I2C (after multiple partial ACK), the Sink core should respond with an ACK.
  - Abort the current Write using:
    - Write to a different I2C slave
    - Read command
    - Address-only Read or Write with MOT = 0.

Handling I2C Write Defer/Timeout:

- The Sink core could issue a Defer response for a burst write to I2C. The following are the actions that can be taken by the Source core:
  - Use the Write status command to poll the transfers happening to the I2C. On successful completion, the Sink core should issue an ACK response to these request while intermediate ones will get a partial ACK.
  - Issue the same command (previously issued with the same device address, length and data) and wait for response. The Sink core on completion of the write to I2C (after multiple Defers) should respond with an ACK.
  - Abort the current Write using:
    - Write to a different I2C slave

- Read command
- Address only Read or Write with MOT = 0.

Transmitter Audio/Video Clock Generation

The transmitter clocking architecture supports both the asynchronous and synchronous clocking modes included in the *DisplayPort Standard v1.2*. The clocking mode is selected by way of the Stream Clock Mode register (MAIN_STREAM_MISC0 bit[0]). When set to '1', the link and stream clock are synchronous, in which case the MVid and NVid values are a constant. In synchronous clock mode, the source core uses the MVid and NVid register values programmed by the host processor via the AXI4-Lite interface.

When the Stream Clock Mode register is set to '0', asynchronous clock mode is enabled and the relationship between MVid and NVid is not fixed. In this mode, the source core will transmit a fixed value for NVid and the MVid value provided as a part of the clocking interface.

Figure 3-6 shows a block diagram of the transmitter clock generation process.

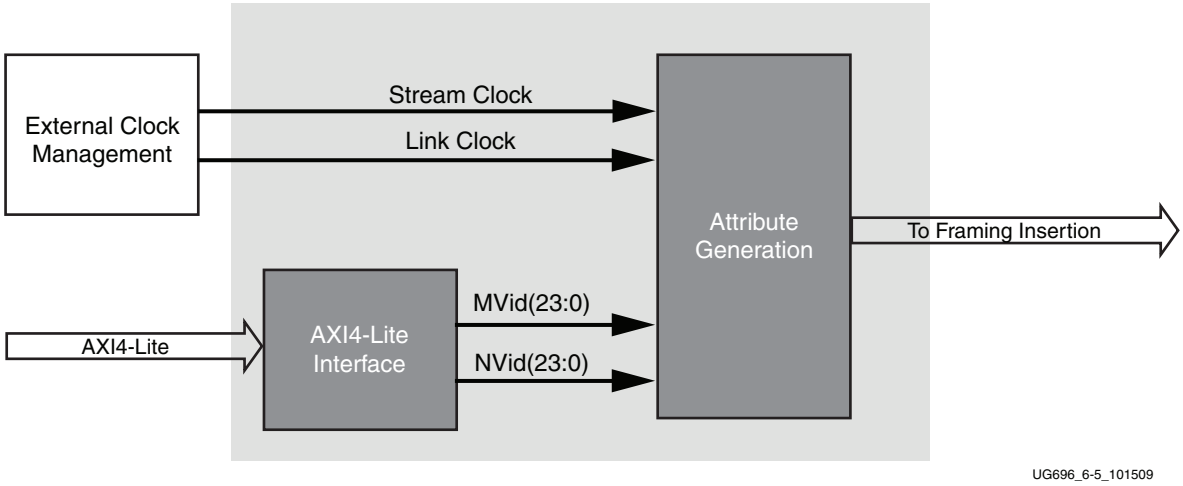

Figure 3-6: **Transmitter Audio/Video Clock Generation**

Hot Plug Detection

The Source device must debounce the incoming HPD signal by sampling the value at an interval greater than 250 microseconds. For a pulse width between 500 microseconds and 1 millisecond, the Sink device has requested an interrupt. The interrupt is passed to the host processor through the AXI4-Lite interface.

If HPD signal remains Low for greater than 2 milliseconds, the sink device has been disconnected and the link should be shut down. This condition is also passed through the AXI4-Lite interface as an interrupt. The host processor must properly determine the cause of the interrupt by reading the appropriate DPCD registers and take the appropriate action.

For details, refer to the *VESA DisplayPort Standard v1.2*[Ref 2].

HPD Event Handling

HPD signaling has three use cases:

- Connection event defined as HPD_EVENT is detected, and the state of the HPD is 1.
- Disconnection event defined as HPD_EVENT is detected, and the state of the HPD is 0.
- HPD IRQ event as captured in the INTERRUPT_STATUS register bit "0".

Figure 3-7 shows the source core state and basic actions to be taken based on HPD events.

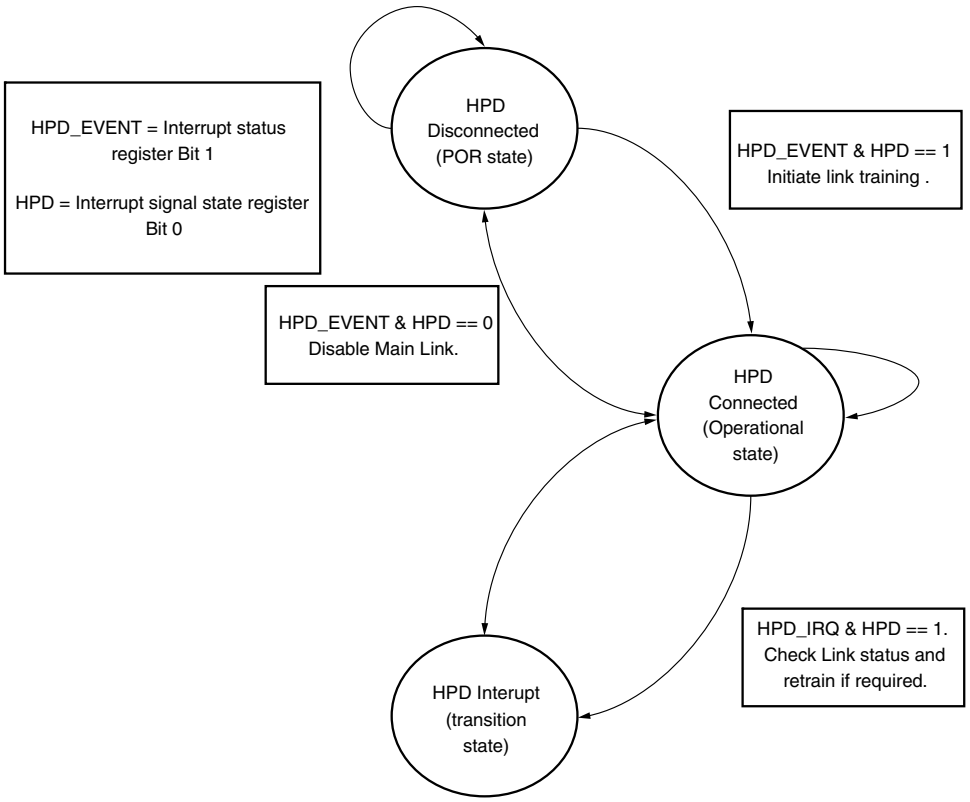

Figure 3-7: **HPD Event Handling in Source Core**

Secondary Channel Operation

The current version of the DisplayPort IP supports 8-channel Audio. Secondary Channel features from the DisplayPort Standard v1.1a are supported.

The DisplayPort Audio IP core is offered as modules to provide flexibility and freedom to modify the system as needed. As shown in Figure 3-8, the Audio interface to the DisplayPort core is defined using an AXI4-Stream interface to improve system design and IP integration.

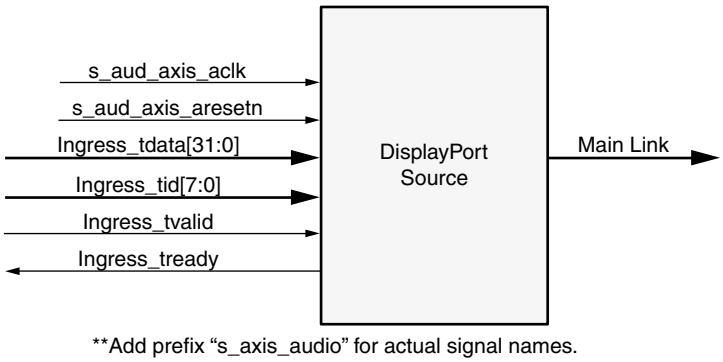

Figure 3-8: **Audio Data Interface of DisplayPort Source System**

32-bit AXI TDATA is formatted according as follows:

Control Bits + 24-bit Audio Sample + Preamble

The ingress channel buffer in the DisplayPort core will accept data from the streaming interface based on buffer availability and audio control programming. A valid transfer takes place when `tready` and `tvalid` are asserted as described in the AXI4-Stream protocol. The ingress channel buffer acts as a holding buffer.

The DisplayPort Source has a fixed secondary packet length [Header = 4 Bytes + 4 Parity Bytes, Payload = 32 Sample Bytes + 8 Parity Bytes]. In a 1-2 channel transmission, the Source accumulates eight audio samples in the internal channel buffer, and then sends the packet to main link.

Programming DisplayPort Source

1. Disable Audio by writing 0x00 to TX_AUDIO_CONTROL register. The disable bit will also flush the buffers in DisplayPort Source and set MUTE bit in VB-ID. When there is a change in video/audio parameters, it is recommended to follow this step.
2. Write Audio Info Frame (Based on your requirement. This may be optional for some systems.). Audio Info Frame consists of 8 writes. The order of write transactions are important and follow the steps mentioned in the Table 2-13.
3. Write Channel Count to TX_AUDIO_CHANNELS register (the value is actual count -1).
4. If the system is using synchronous clocking then write MAUD and NAUD values TX_AUDIO_MAUD and TX_AUDIO_NAUD registers.
5. Enable Audio by writing 0x01 to TX_AUDIO_CONTROL register.

Re-Programming Source Audio

1. Disable Audio in DisplayPort TX core.
2. Wait until Video/Audio clock is recovered and stable.
3. Enable Audio in DisplayPort TX core.
4. Wait for some time (in µs).

Info Packet Management

The core provides an option to program a single Info packet. The packet will be transmitted to Sink once per every video frame or 8192 cycles.

To change an Info packet during transmission, follow these steps:

1. Disable Audio (Since new info packet means new audio configuration). The disable audio will also flush internal audio buffers.
2. Follow steps mentioned in Programming DisplayPort Source.

Extension Packet Management

A single packet buffer is provided for the extension packet. If the extension packet is available in the buffer, the packet is transmitted as soon as there is availability in the secondary channel. The packet length is FIXED to eight words (32 bytes).

Use the following steps to write an extended packet in the DisplayPort Source controller:

1. Write nine words (as required) into TX_AUDIO_EXT_DATA buffer.
2. Wait for EXT_PKT_TXD interrupt.
3. Write new packet (follow step 1).

Audio Clocking (Recommendation)

The system should have a clock generator (preferably programmable) to generate 512 X fs (Audio Sample Rate) clock frequency. The same clock (aud_clk) is used by DisplayPort Source device to calculate MAUD and NAUD when running in asynchronous clocking mode.

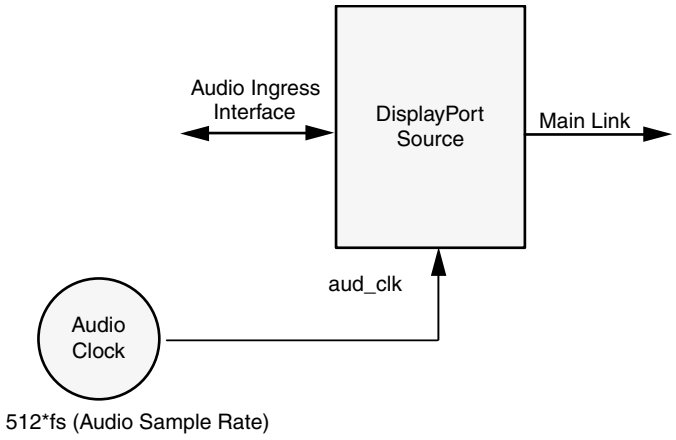

Figure 3-9: **Source: Audio Clocking**

Programming the Core in MST Mode

The section details the steps to program the core in MST mode.

Enabling MST

The following steps are recommended to enable MST functionality:

1. Bring up the main link by following training procedure.
2. Send side band messages using the AUX channel to discover the link (how many downstream nodes are connected and their capabilities).
3. Enable MST by writing '1' to bit 0 of the MST Config register.
4. Discover MST downstream devices as recommended in section 1.2.1 in the *DisplayPort Standard*. The software for MST discovery may cary based on system performance.
5. Allocate timeslots based on configuration and the Sink Payload Bandwidth Number (PBN). Typical sideband messages used before VC Payload allocation are Link Address Request, Clear Payload Table, and Enumerate Path Resources.
    a. Program VC Payload Buffer 12'h0x800 onwards as per allocation requirement.
    b. Program the Sink core with the same allocation timeslots using AUX channel as described in section 2.6.4 in the *DisplayPort Standard*.
    c. Wait until Sink accepts allocation programming (check DPCD reads to monitor status).
    d. After Sink sets VC Payload Allocated (DPCD Address=0x02C0), set VC Payload Allocated bit in MST Config register (12'h0x0D0). This enables the source controller to send an ACT trigger.

6. Wait until ACT Handled bit is set in DPCD Address (0x02C0).
7. Program Video attributes for required streams. Program user pixel width to 4 for all the streams.
8. Program Rate Governing registers 0x1D0, 0x1D4, 0x1D8 & 0x1DC based on the stream requirement.
   - Program TRANSFER UNIT Size = # of timeslots allocated for that stream. (VC payload size source)
   - Program FRAC_BYTES_PER_TU = TS_FRAC
   - Program MIN_BYTES_PER_TU = TS_INT
   - Program INIT_WAIT = 0

   Note: Repeat step 7 for each steam.
9. Enable MST by writing 1 to bit 0 of MST Config register.

After these steps are done, the source controller will start sending MST traffic as per VC Payload programming in the main link.

Payload Bandwidth Management

The following steps manage payload bandwidth in the source controller.

1. Calculate Target_Average_StreamSymbolTimeSlotsPerMTP based on the *DisplayPort Standard v.1.2* or later. To do this, program VC payload size with calculated Target_Average_StreamSymbolTimeSlotsPerMTP and align it with nearest even boundary.

   For example if the value is 13, program VC payload size for this particular stream to 14.
2. In MST mode when GT data width is 4 byte the VC Payload should be multiple of 4.
3. The VC payload calculation for UHD (1920X2200) stream, RGB color sampling, 8 Bits Per Color at 5.4 Gb/s, 4 lanes is given below.

   VC Payload Band width = LINK_RATE*Lane_count*10

   = 5.4*4*10

   = 2160

   Average Stream symbol Time slot per MTP = (Pixel_rate* Bits_per_pixel/8/VC Payload_Band width) *64

   = (297 MHz * 24 /8/2160)*64

   = 26.4

VC Payload Size = 2/4 symbol aligned of (Average Stream symbol Time slot per MTP)

= 28

4. Program VC Payload table as defined in DPCD standard.
5. Program VC Payload table in source controller as defined in registers 12'h0x800 – 12'h0x8FC.

Sink 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, as described in Figure 3-10.

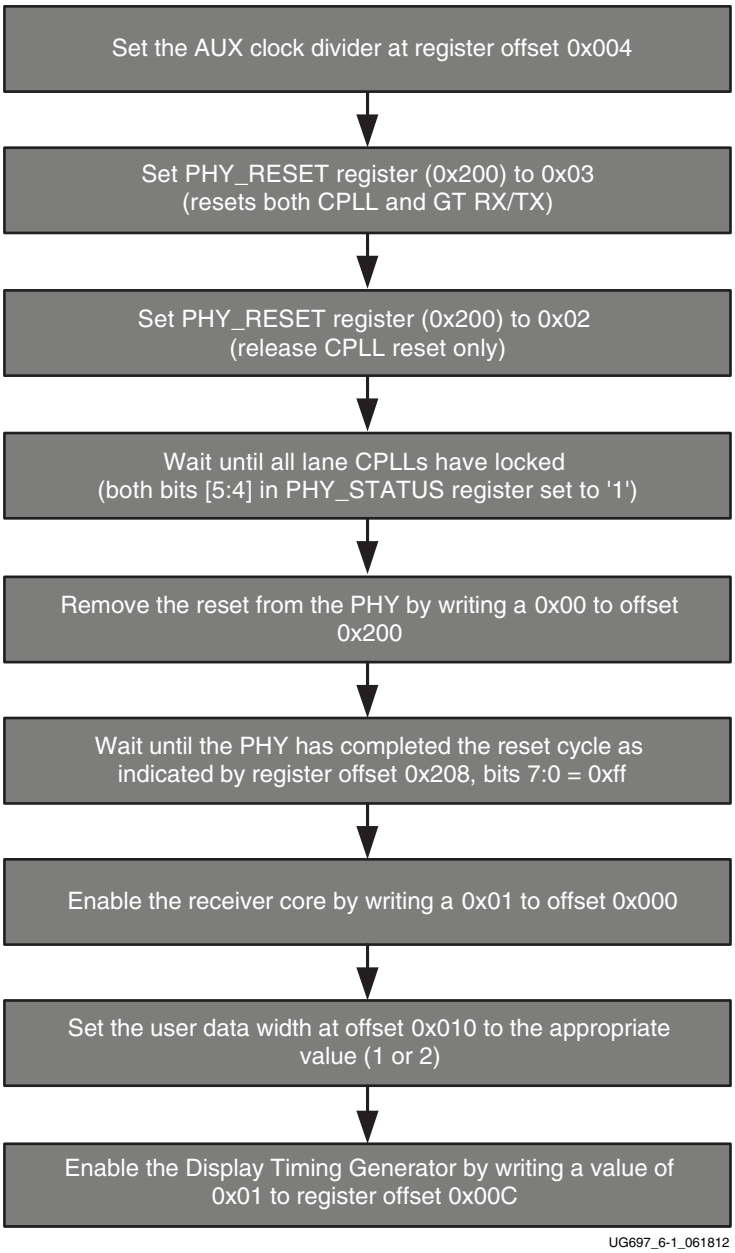

Figure 3-10: **Receiver Core Initialization**

Note: In UltraScale™ devices, the initialization sequence applies to CPLL reset only, GT TX/RX reset need not be applied. After releasing CPLL reset, wait for the reset done status and follow the instructions as provided.

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 may 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.

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 data path 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 may occur.

The link training process is specified in section 3.5.1.3 of the *DisplayPort Standard v1.2*.

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's 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.

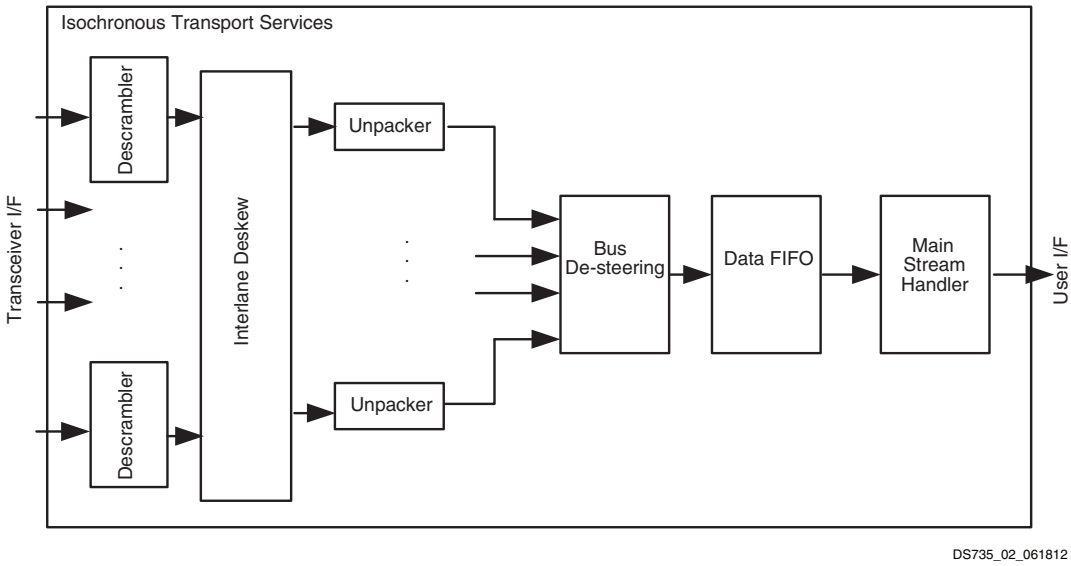

Figure 3-11: **Sink Main Link Datapath**

Figure 3-12 shows the flow diagram for link training.

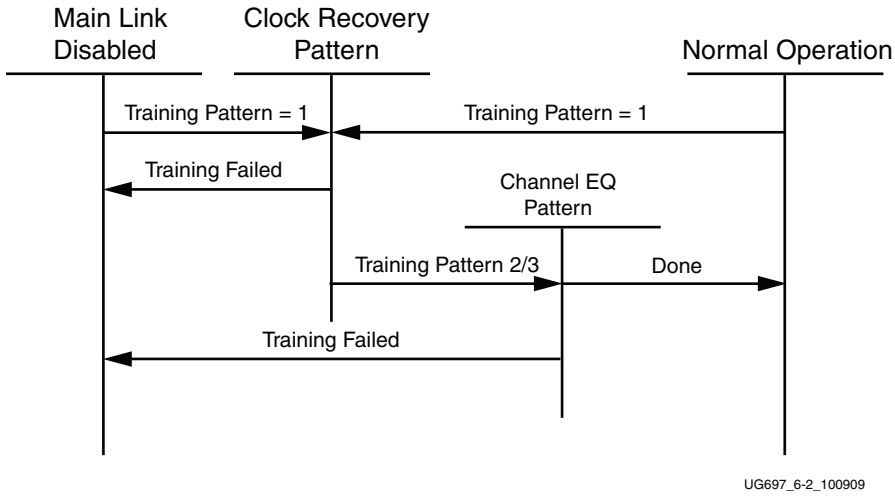

Figure 3-12: **Link Training States**

Receiver Clock Generation

The receiver core requires the generation of a video stream clock for transmitting the recovered image data over the user data interface. Data fields within the Main Stream Attributes (M and N values) provide the information by which an accurate stream clock may be reconstructed. The receiver core places this information on dedicated signals and provides an update flag to signal a change in these values. Alternatively, the user may use a fast clock to pull data from the User Data Interface and push it into a frame buffer.

Figure 3-13 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 v1.2* for more details.

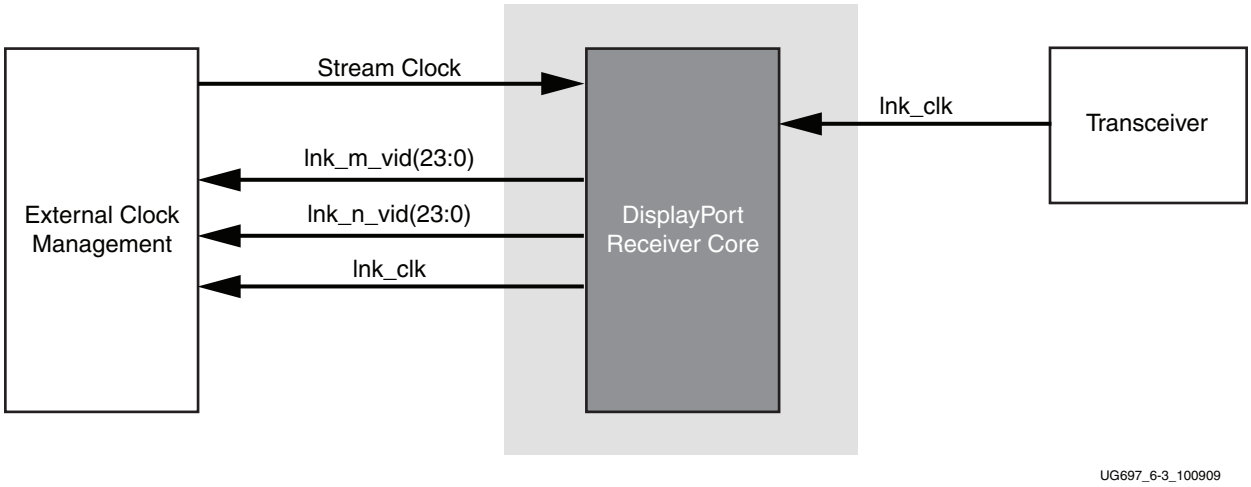

Figure 3-13: **Receiver Clock Generation**

Common Event Detection

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

Transition from Video to No Video

In the course of operation, the source core may 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 pattern can be detected by the VERTICAL_BLANKING interrupt. Upon the reception of a VERTICAL_BLANKING interrupt, if disabled, you may then reenable 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 will properly reset itself and apply 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 the PLL is locked and reset is complete, 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 3-14, 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.

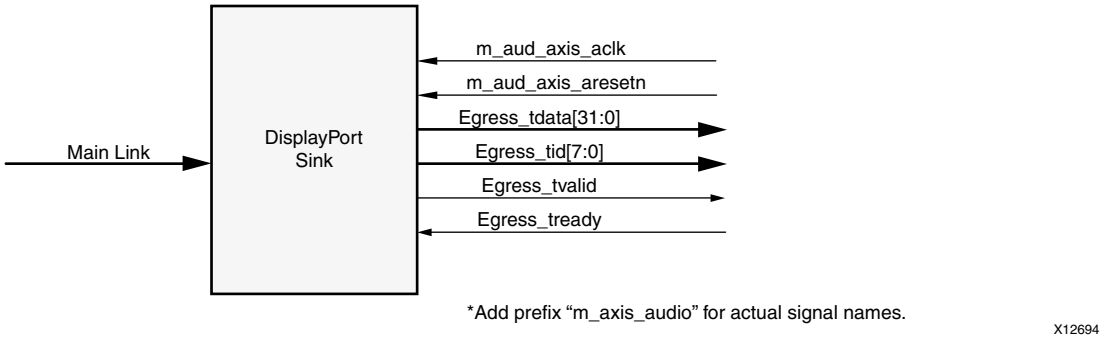

Figure 3-14: **Audio Data Interface of DisplayPort Sink System**

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, it is recommended to follow 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's 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 will receive 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 X fs (Audio Sample Rate) clock frequency based on MAUD and NAUD values.

The AXI4-Stream clock does not need to be related to the Audio clock. See Clocking for more details.

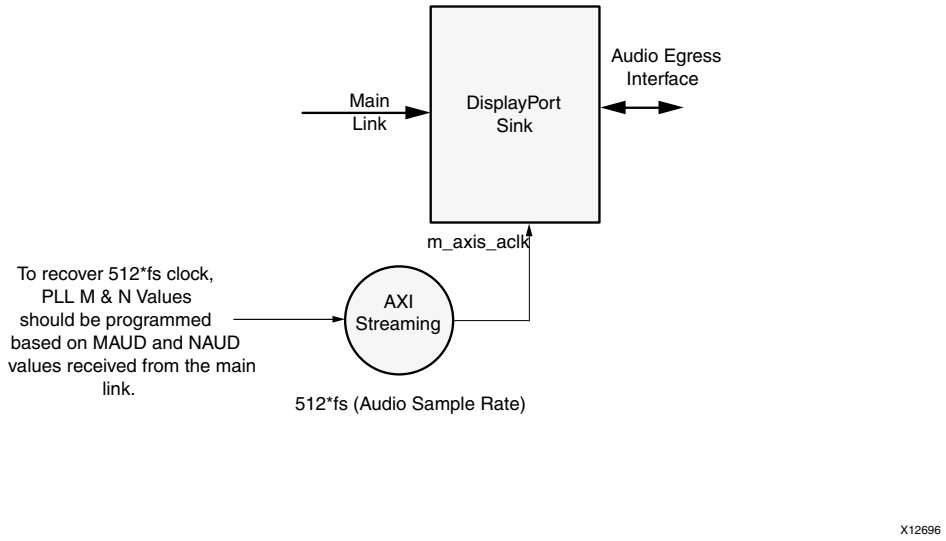

Figure 3-15: **Sink: Audio Clocking**

Sampling Frequencies

The Displayport RX with GT Datawidth 16-bit mode supports upto 8-channels of audio with maximum supported sampling frequency of 192Khz for all link rates. The Displayport RX with GT Datawidth 32-bit mode has limitation in maximum supported sampling frequency for 2.7Gb/s and 1.62Gb/s link rates.

Link rate 5.4 Gb/s rate supports up to 8-channels with maximum sampling frequency of 192Khz, where as 2.7Gb/s link rate supports up to 8 channels with maximum audio sampling frequency of 176.4Khz and 1.62Gb/s link rate supports up to 8 channels with maximum audio sampling frequency of 96 Khz. This limitation is due to the lnk_clk frequency reduction in 32-bit GT Data width mode.

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 bringup 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 v1.2 Section 2.1.11.9. After a complete side-band 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.

Source Core Interfaces

This section describes the Source core interfaces.

User Data Interface

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

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

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

Figure 3-16 shows the typical signalling of a full frame of data.

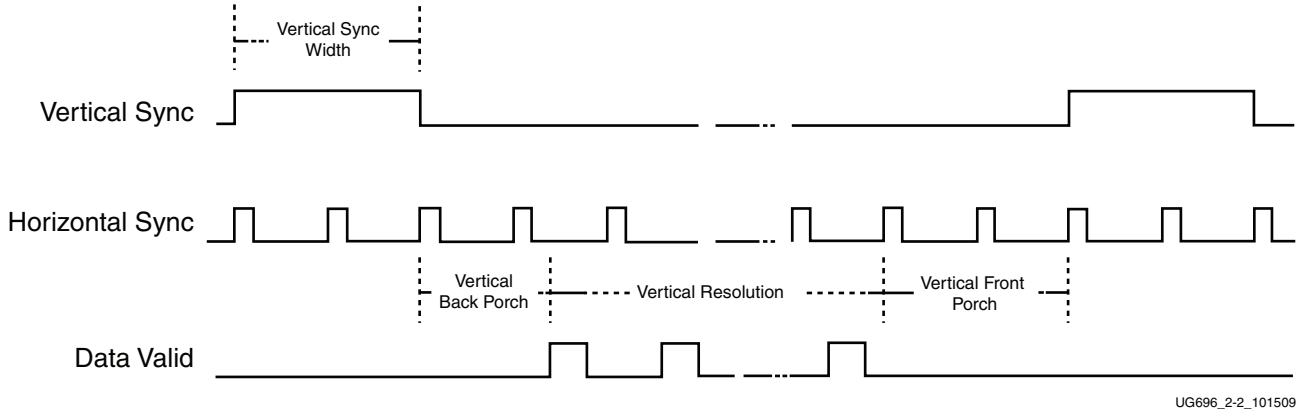

Figure 3-16: **User Interface Vertical Timing**

Similarly, the horizontal timing information is defined by a front porch, back porch, and pulse width. The porch values are defined as the number of clocks between the horizontal sync pulse and the start or end of active data. Pixel data is only accepted into the image data interface when the data valid flag is active-High, as shown in Figure 3-17.

Note that the data valid signal must remain asserted for the duration of a scan line. Dropping the valid signal might result in improper operation.

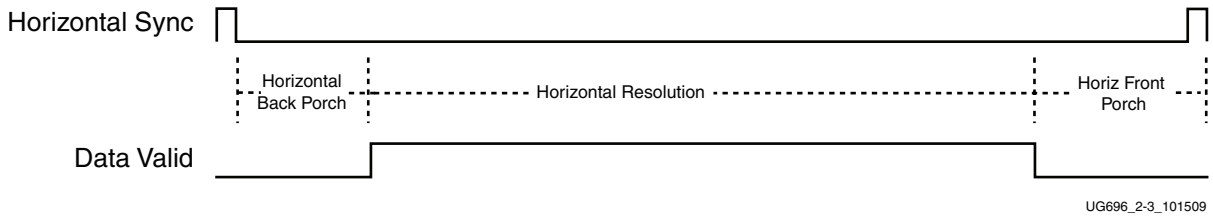

Figure 3-17: **User Interface Horizontal Timing**

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

active frame size is shown in Figure 3-18.

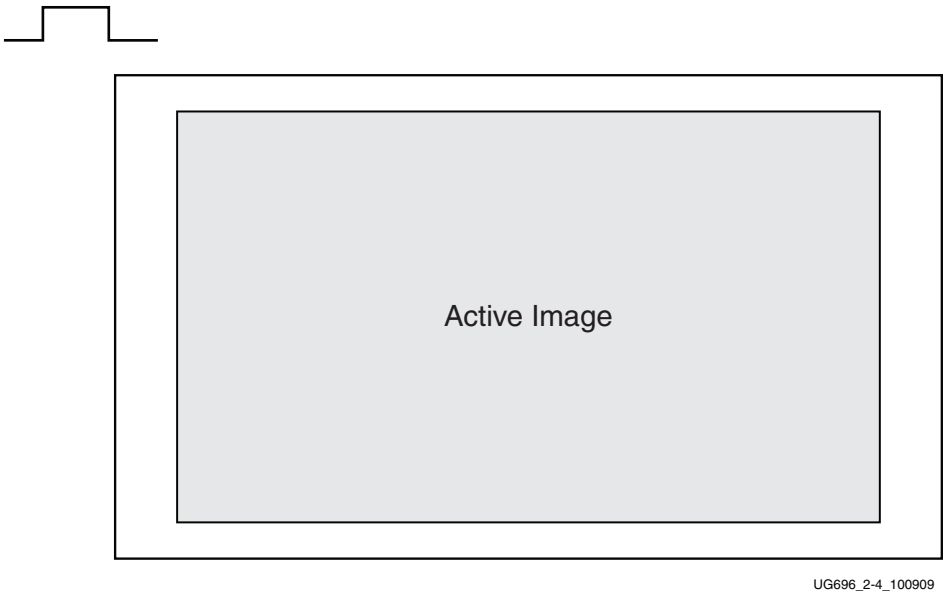

Figure 3-18: **Active Image Data**

The User Data Interface can accept one, two, or four pixels per clock cycle. The vid_pixel width is always 48 bits, regardless of if all bits are used. For pixel mappings that do not require all 48 bits, the convention used for this core is to occupy the MSB bits first and leave the lower bits either untied or driven to zero. Table 3-3 provides the proper mapping for all supported data formats.

Table 3-3: **Pixel Mapping for the User Data Interface**

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

Table 3-3: **Pixel Mapping for the User Data Interface *(Cont'd)***

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

Notes:

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

Selecting the Pixel Interface

The Pixel clock is supported up to 150 MHz, and it is very difficult to meet timing above this frequency. However, you have the option of selecting a single, dual or quad pixel video interface. See Clocking for more details.

To determine the necessary pixel interface to support a specific resolution, it is important to know the active resolution and blanking information.

Note: In a quad pixel interface, if the resolution is not divisible by 4, you should add zeroes at the end of frame, over the video interface pixel data.

For example:

To support an active resolution of 2560x1600@60, there are two possible blanking formats: Normal Blanking and Reduced Blanking, as defied by the VESA standard.

2560x1600@60 + Blanking = 3504x1658@60

Requires a Pixel clock of 348.58 MHz

2560x1600@60 + Reduced Blanking = 2720x1646@60

Requires a Pixel clock of 268.63 MHz

Assuming a pixel clock of 150MHz and a dual Pixel interface:

2560x1600@60 + Blanking = 3504x1658@60 = 348.58 MHz

348.58 MHz / 2 = 172.28 MHz

2560x1600@60 + Reduced Blanking = 2720x1646@60 = 268.63 MHz

268.63 MHz / 2 = 134.31 MHz

With a dual Pixel interface, the DisplayPort IP can support 2560x1600 only if there is a Reduced Blanking input. If full Blanking support is needed, then a 4 Pixel interface should be used.

Figure 3-19, Figure 3-20, and Figure 3-21 show timing diagrams for the three Pixel interface options.

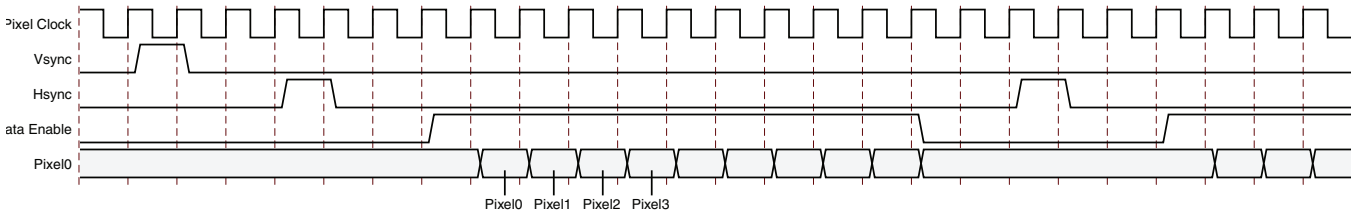

Figure 3-19: **Single Pixel Timing**

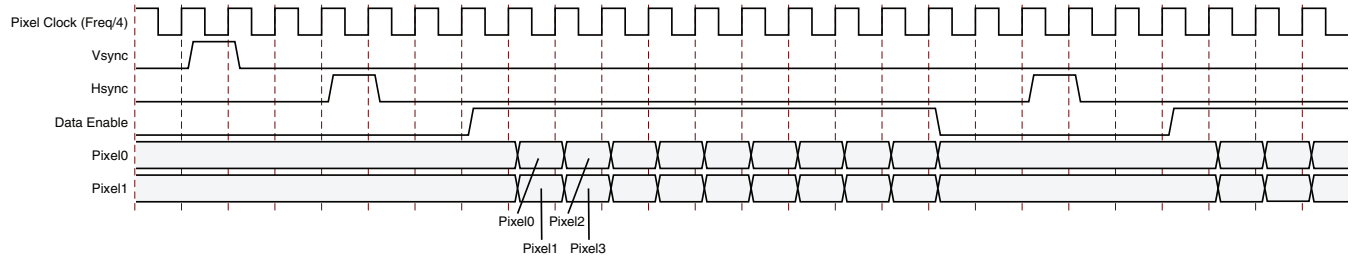

Figure 3-20: **Dual Pixel Timing**

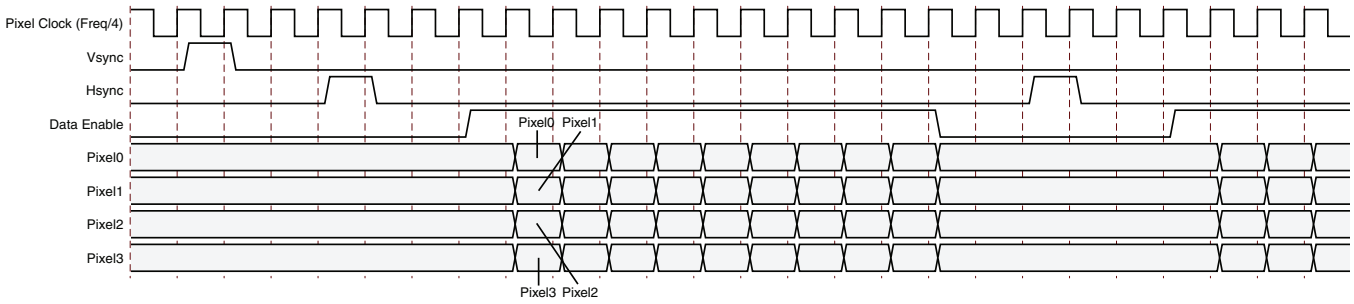

Figure 3-21: **Quad Pixel Timing**

Host Processor Interface

The host processor bus uses an AMBA AXI4-Lite interface, which was selected because of its simplicity. The processor bus allows for single reads and writes to configuration space. See Source Core in Chapter 2 for full address mapping.

Additionally, the host processor interface is the gateway for initiating and maintaining the main link. This is done through Link and Device services, which include EDID and DPCD reads. Main link initiation concludes with a Link Training sequence, which is also started through this interface. Refer to Link Training as well as the *VESA DisplayPort Standard* v1.1 [Ref 2] for more information about the initiation sequence.

The core comes with an example design policy maker in C source code. For users who do not have specific needs to control or tune the core, this is an ideal resource.

AXI4-Lite Read and Write Cycles

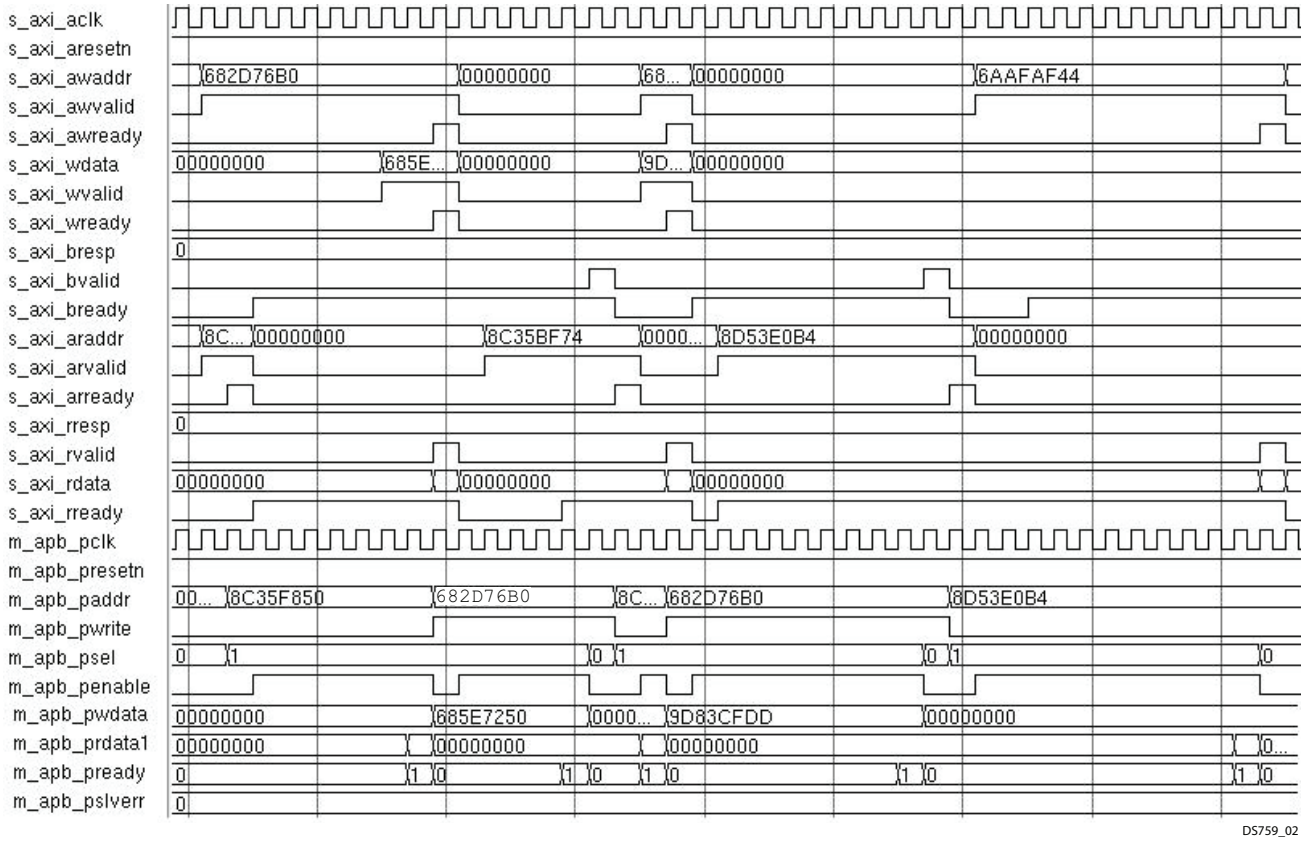

Figure 3-22: **AXI4-Lite Read and Write Cycles**

The AXI4-Lite write transfer begins with the address, write signal, and write data set to their proper values on the first rising edge of the clock. The first clock cycle of the transfer is called the SETUP cycle. On the second rising edge of the clock, the enable signal is asserted and the ENABLE cycle is entered. The address, data, and control signals all remain valid through both cycles of the transfer. The transfer completes on the following rising edge of the clock, as shown in Figure 3-22.

The AXI4-Lite read transfer begins with the SETUP cycle on the first rising edge of the clock with the address and control signals at their proper values. As with the write transfer, the enable signal is asserted on the next rising edge marking the beginning of the ENABLE cycle. The slave peripheral must provide data during this cycle. The read data is sampled on the next rising edge of the clock at the end of the ENABLE cycle. This transfer is shown in Figure 3-22.

Transceiver Interface

The transceivers have been pulled out of the core and are provided as instances in the top-level wrapper. You may choose up to four high-speed lanes. Despite the number of

lanes that have been chosen, the negotiation process is handled by a policy maker, which may elect for fewer number of in-use lanes. Additionally, the core supports 5.4 Gb/s, 2.7 Gb/s and 1.62 Gb/s operation. The negotiation process also determines the actual line rate.

You must provide the appropriate reference clock on the lnk_clk_p/n ports. These ports must be physically located on the appropriate MGTREFCLK pins. Additionally, you must physically locate the `lnk_tx_lane` ports to the appropriate pins. To find the appropriate placement locations, refer to the transceiver user guide for the FPGA family used (References).

For 7 series FPGAs, a common reference clock of 135 MHz (harmonic of 27 MHz) is needed for 1.62, 2.7 and 5.4 Gb/s link rates. See Clocking for more details. UltraScale™ transceivers require reference clock of 270 MHz to work with 1.62 link rate. All other link rates can work with common reference clock of 135 MHz.

The transceivers have been tuned for optimal communication. The constraints related to transceiver tuning have been placed directly in the RTL instance. Users may want to review these values and make sure they are fully aware of their functions.

AUX Channel Interface/HPD Interface

The AUX channel is used for link and device communication between source and sink devices. The AUX channel uses Manchester-II Coding and requires a 1 MHz (or a multiple of 1 MHz) clock source. The AXI4-Lite clock is used to run the internal operations of the AUX Channel logic. As a result, using the bus interface clock in this way restricts the AXI4-Lite clock frequency to an integer multiple of 1 MHz.

See Constraining the Core in Chapter 4 for more details on the IO standard of AUX Channel Interface/HPD Interface.

Audio Interface

Audio samples are transferred to the Displayport Audio engine through the AXI4-Stream interface.

Debug Interface

The Debug Interface include Link GT, Link Control and AUX debugging signals. The Link-related signals are taken from the TX PHY file, `<component_name>_tx_phy.v`.

Link GT Signals (lnk_debug_gt)*

Table 3-8 lists the Link GT signals.

Table 3-4: **LINK GT Signals**

Position	Signal Name	Description
31:0	lnk_tx_lane*_data	32- Bit Link data, In 16-Bit GT data width mode lower 16 bits are active
35:32	lnk_tx_lane*_k_char	4- Bit kchar value. , In 16-Bit GT data width mode lower 2 bits are active
39:36	lnk_tx_lane*_override_disparity	4-Bit link disparity value, , In 16-Bit GT data width mode lower 2 bits are active
44:40	i_tx_postcursor_lane_*	Post cursor level of the link
49:45	i_tx_precursor_lane_*	Pre cursor level of the link
52:50	i_tx_voltage_swing_lane_*	Voltage swing level
54:53	i_tx_buffer_status_lane_*	GT buffer status
55	Reserved	NA
56	i_pll_lock_detect_tile_*	PLL lock status signal
68:57	Reserved	NA

LINK Control Signals (lnk_debug_control)

Table 3-9 lists the LINK Control signals.

Table 3-5: **LINK Control Signals**

Position	Signal Name	Description
0	i_phy_reset	CPLL reset through APB or internal auto reset after DRP line rate change
1	i_tx_phy_reset	CPLL reset through APB
2	i_tx_phy_reset_2	GT TX reset through APB
4:3	i_reset_done_tile0	Reset done status for line1, line0
6:5	i_reset_done_tile1	Reset done status for line3, line2
7	link_bw_high	Indicates the 2.7 Gbps line rate selection
8	link_bw_hbr2	Indicates the 5.4 Gbps line rate selection
9	link_bw_rbr	Indicates the 1.62 Gbps line rate selection
14:10	i_drp_state	DRP state
15	i_drp_enable	Enable to the DRP
16	i_drp_write	Write enable to the DRP
32:17	i_drp_read_data	Read data from DRP
48:33	i_drp_write_data	Write data to DRP
49	i_drp_ready00	DRP ready from line0
50	i_drp_ready01	DRP ready from line1
51	i_drp_ready10	DRP ready from line2

Table 3-5: **LINK Control Signals *(Cont'd)***

Position	Signal Name	Description
52	i_drp_ready11	DRP ready from line3
60:53	i_drp_addr	DRP address
63:61	i_tx_enable_prbs7	GT TX PRBS SEL input
67:64	i_tx_power_down	GT TX power down control
71:68	i_tx_pma_reset_done_out	GT TX PMA reset done status out
72	pll1_lock_in (GTP) / gt0_qpllclk_lock (GTX/GTH)	PLL1 lock input (GTP) QPLL lock (GTX/GTH)
95:73	Reserved	NA

AUX Debug Signals

Table 3-10 lists the AUX Debug signals.

Table 3-6: **AUX Debug Signals**

Position	Signal Name
0	aux_data_in
1	aux_data_out
2	aux_data_enable_n
3	hot_plug_detect

Sink Core Interfaces

This section details the Sink core interfaces.

User Data Interface

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

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

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

The user has the option to use the resolved M and N values from the stream to generate a clock, or to use a sufficiently-fast clock and pipe the data into a line buffer. Xilinx recommends using a fast clock and ignoring the M and N values unless you can be certain of the source of these values. Unlike the Source Core, when using a fast clock, the data valid signal may toggle within a scan line. Figure 3-23 shows the typical signalling of a full frame of data.

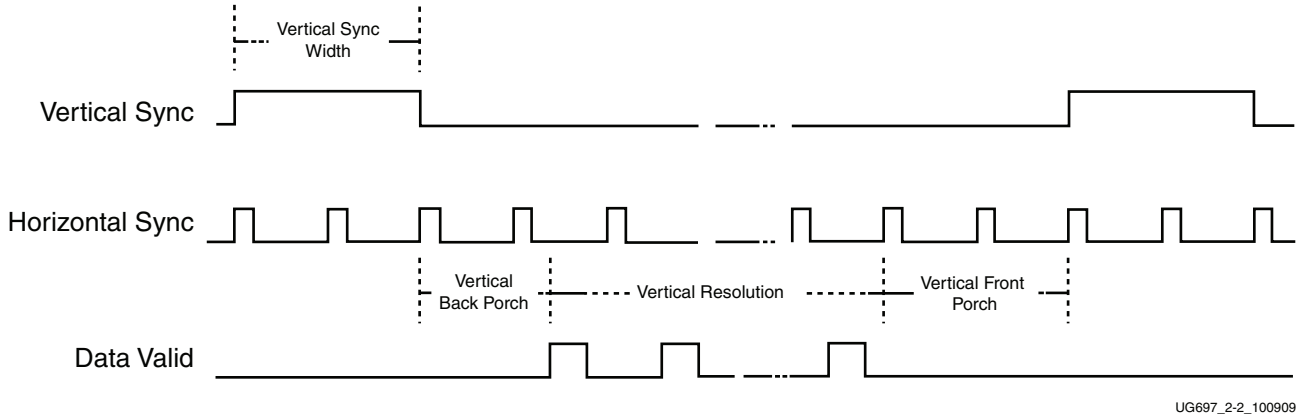

Figure 3-23: **User Interface Vertical Timing**

The horizontal timing information is defined by a front porch, back porch, and pulse width. The porch values are defined as the number of clocks between the horizontal sync pulse and the start or end of active data. Pixel data is only accepted into the image data interface when the data valid flag is active-High. Figure 3-24 is an enlarged version of Figure 3-23, giving more detail on a single scan line. The horizontal sync pulse should be used as a line advance signal. Use the rising edge of this signal to increment the line count. Note that Data Valid may toggle if using a fast clock.

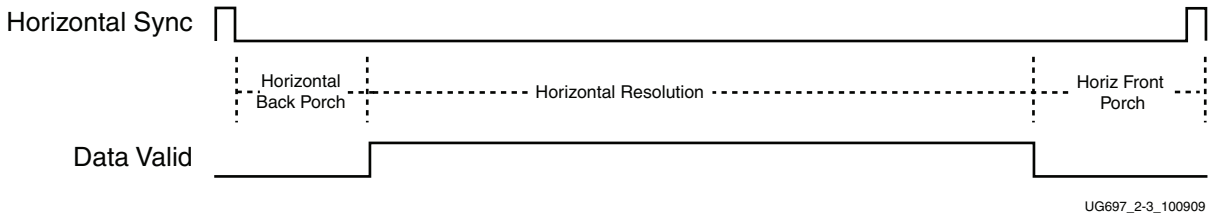

Figure 3-24: **User Interface Horizontal Timing**

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

active frame size is shown in Figure 3-25.

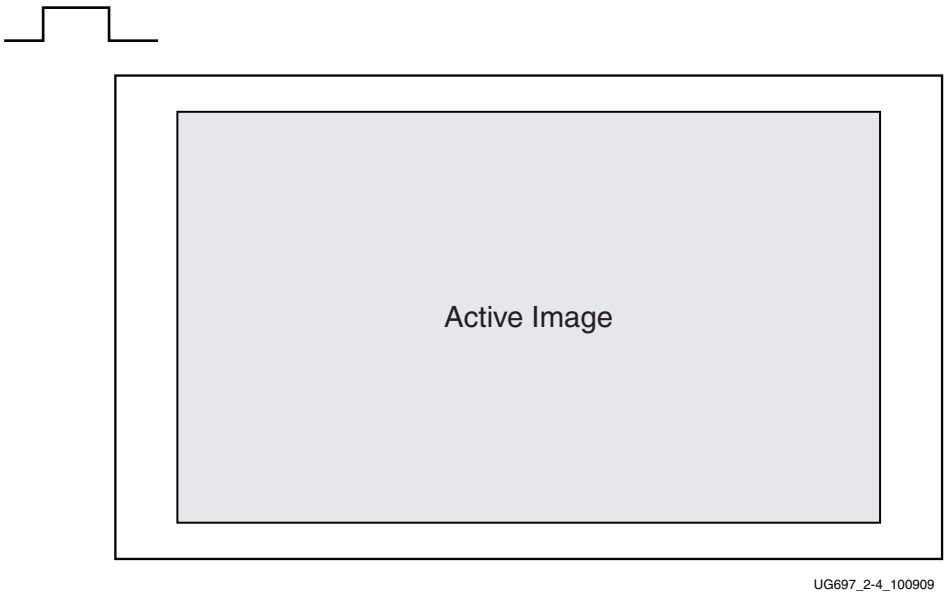

Figure 3-25: **Active Image Data**

The User Data Interface can accept one, two, or four pixels per clock cycle. The second pixel is active only when USER_PIXEL_WIDTH is set and the negotiated number of lanes is greater than one. See Selecting the Pixel Interface for more information on correctly setting the USER_PIXEL_WIDTH value.

The vid_pixel width is always 48 bits, regardless of if all bits are used. For pixel mappings that do not require all 48 bits, the convention used for this core is to occupy the MSB bits first and leave the lower bits either untied or driven to zero. Table 3-7 provides the proper mapping for all supported data formats.

Table 3-7: **Pixel Mapping for the User Data Interface**

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

Table 3-7: **Pixel Mapping for the User Data Interface *(Cont'd)***

Format	BPC/BPP	R	G	B	Cr	Y	Cb	Cr/Cb	Y
YCrCb422	10/20							[47:38]	[31:22]
YCrCb422	12/24							[47:36]	[31:20]
YCrCb422	16/32							[47:32]	[31:16]
YONLY	8/8								[47:40]
YONLY	10/10								[47:38]
YONLY	12/12								[47:36]
YONLY	16/16								[47:32]

Notes:

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

The design allows use of a faster pixel clock. The interface timing in this case will be as shown Figure 3-26.

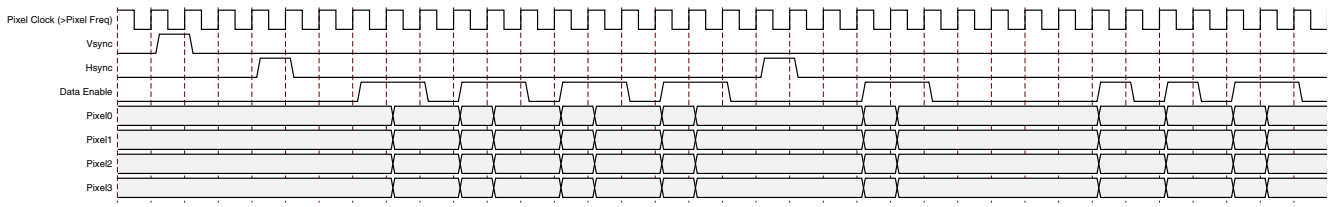

Figure 3-26: **RX Pixel Timing**

Host Processor Interface

The host processor bus uses an AXI4-Lite interface, which was selected because of its simplicity. The processor bus allows for single reads and writes to the configuration space. See Chapter 2, Register Space for address mapping.

Use the Sink core Host Processor Interface to enable and set up the core. This interface may also be used to check the status of training.

AXI4-Lite Read and Write Cycles

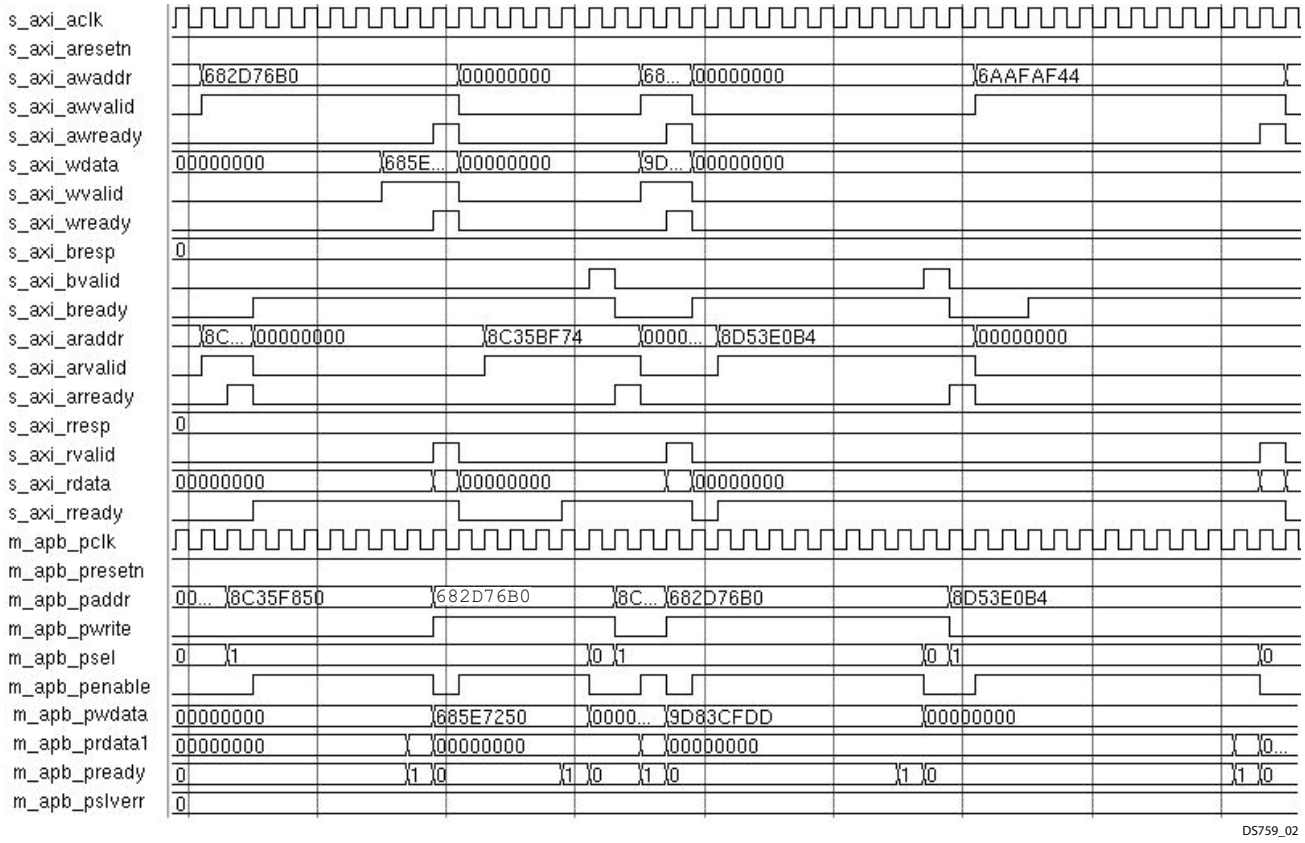

Figure 3-27: **AXI4-Lite Read and Write Cycles**

The AXI4-Lite write transfer begins with the address, write signal, and write data set to their proper values on the first rising edge of the clock. The first clock cycle of the transfer is called the SETUP cycle. On the second rising edge of the clock, the enable signal is asserted and the ENABLE cycle is entered. The address, data, and control signals all remain valid through both cycles of the transfer. The transfer completes on the following rising edge of the clock, as shown in Figure 3-27.

The AXI4-Lite read transfer begins with the SETUP cycle on the first rising edge of the clock with the address and control signals at their proper values. As with the write transfer, the enable signal is asserted on the next rising edge marking the beginning of the ENABLE cycle. The slave peripheral must provide data during this cycle. The read data is sampled on the next rising edge of the clock at the end of the ENABLE cycle. This transfer is shown in Figure 3-27.

Transceiver Interface

The transceivers have been pulled out of the core and are provided as instances in the top-level wrapper. The user may choose up to four high-speed lanes. Despite the number of

lanes that have been chosen, the core automatically handles the negotiation process, which may result in a fewer number of in-use lanes. The negotiation process also determines the actual line rate.

The user must provide the appropriate reference clock on the `lnk_clk_p/n` ports. These ports must be physically located on the appropriate MGTREFCLK pins. Additionally, you must physically locate the `lnk_tx_lane` ports to the appropriate pins. To find the appropriate placement locations, refer to the transceiver user guide for the FPGA family used (References).

For 7 series FPGAs, a common reference clock of 135 MHz is needed for 1.62, 2.7 and 5.4 Gb/s link rates. See Clocking for more details.

The transceivers have been tuned for optimal communication. The constraints related to transceiver tuning have been placed directly in the RTL instance. Users may want to review these values and make sure they are fully aware of their functions.

AUX Channel Interface/HPD Interface

AUX Channel Services are provided through a dedicated differential pair in the PHY layer. The data operates at a frequency of 1 Mb/s with all data Manchester-II encoded. The functional independence of the AUX Channel allows for a design which is independent of the main link with the exception of the DisplayPort Configuration Data (DPCD). All DPCD registers are considered to be asynchronous to the link clock. Where necessary, synchronization stages will be used to properly sample the data in the main link design.

The AXI4-Lite clock is used to run the internal operations of the AUX Channel logic. In addition, the AXI4-Lite clock is used to derive the data rate of the Manchester-II encoded transmit and reply data. Using the bus interface clock in this way restricts the AXI4-Lite clock frequency to an integer multiple of 1 MHz. This restriction is required in order to generate the Manchester-II codes at the frequency of 1 Mb/s.

See Constraining the Core in Chapter 4 for the IO standards of these ports.

DisplayPort Configuration Data

The DisplayPort Configuration Data is implemented as a set of registers which may be read or written from the AXI4-Lite interface. While these registers are not technically part of the AUX Channel interface, they are integrated here for access via the AXI4-Lite bus interface. These registers are considered to be synchronous to the AXI4-Lite domain and asynchronous to all others.

For parameters that may change while being read from the configuration space, two scenarios may exist. In the case of single bits, the data may be read without concern as either the new value or the old value will be read as valid data. In the case of multiple bit fields, a lock bit may be maintained to prevent the status values from being updated while

the read is occurring. For multi-bit configuration data, a toggle bit will be used indicating that the local values in the functional core should be updated.

I2C Interface

Note: This is a pass-through interface. The expectation is for the controller to be built outside of the core. See the example design included with the core for details (Chapter 5, Detailed Example Design).

The Source core enables the I2C protocol over the AUX channel. For direct access via I2C and as an alternative to the host processor bus, use this dedicated interface. Figure 3-28 shows an example I2C Transaction.

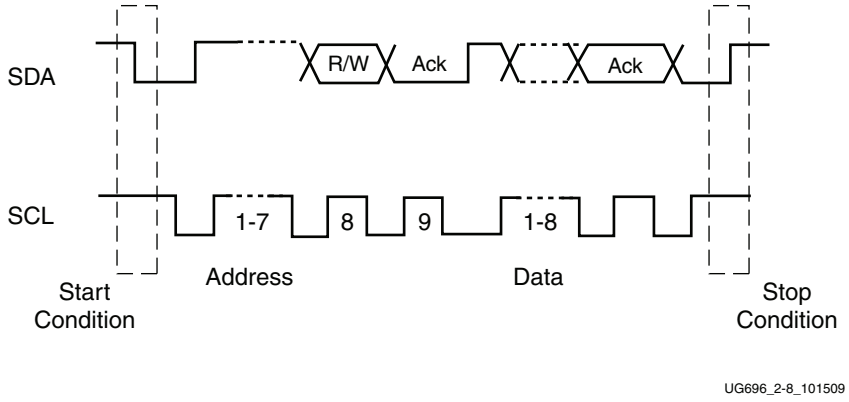

Figure 3-28: **I2C Transaction**

Audio Interface

Audio is received from the DisplayPort link and transferred over the AXI4-Stream interface. Any audio controller, capable of processing streaming audio data, can play the audio.

Debug Interface

The Debug Interface include Link GT, Link Control and AUX debugging signals. The Link related signals are taken from the RX PHY file `<component_name>_rx_phy.v`.

Link GT Signals (lnk_debug_gt)*

Table 3-8 lists the Link GT signals.

Table 3-8: **LINK GT Signals**

Position	Signal Name	Description
31:0	lnk_rx_lane*_data	32- Bit Link data, In 16-Bit GT data width mode lower 16 bits are active
35:32	lnk_rx_lane*_k_char	4- Bit kchar value. , In 16-Bit GT data width mode lower 2 bits are active

Table 3-8: **LINK GT Signals *(Cont'd)***

Position	Signal Name	Description
39:36	lnk_rx_lane*_disparity_error	4-Bit link disparity error input,, In 16-Bit GT data width mode lower 2 bits are active
43:40	lnk_rx_lane*_symbol_error	4-Bit link symbol error input,, In 16-Bit GT data width mode lower 2 bits are active
44	I_comma_align_enable*	Enable comma align when TP1 is not enabled
45	i_rx_low_voltage_lane*	GT RX ELECIDLE Out
46	I_lane0_symbol_locked	GT BYTE Is aligned out status
49:47	i_rx_buffer_status_lane_*	GT RX buffer status
50	i_rx_pma_reset_done_out*	RX PMA reset done out
51	i_eyescandataerror_out*	GT RX eye scan data error out
52	I_comma_det_out*	GT RX comma detect out
67_53	I_dmonitor_out*	GT dmonitor out
68	I_prbs_error_lane*	Gt RX PRBS Error out

LINK Control Signals (lnk_debug_control)

Table 3-9 lists the LINK Control signals.

Table 3-9: **LINK Control Signals**

Position	Signal Name	Description
0	i_phy_reset	CPLL reset through APB or internal auto reset after DRP line rate change
1	i_rx_phy_reset	CPLL reset through APB
2	i_rx_phy_reset_2	GT RX reset through APB
3	i_rx_phy_reset_3	Internal GT RX auto reset generated when DP Equalization mode changed from LPM to DFE for 5.4Gbps line rate
4	i_rx_phy_reset_auto	Auto reset after DRP line rate change
5	i_rx_phy_pcs_reset	GT RX PCS reset
7:6	i_training_pattern_q	Training pattern
11:8	i_rx_power_down	Gt RX power down selection
13:12	i_reset_done_tile0	Reset done status for line1, line0
15:14	i_reset_done_tile1	Reset done status for line3, line2
16	i_prbs_enable_set	Enable status for link quality patterns or custom quality patterns
19:17	i_prbs_test_enable	GT RX PRBS Select signal
20	i_tp1_start_detect_sync	TP1 start detect
21	i_tp2_start_detect_sync	TP2 start detect

Table 3-9: **LINK Control Signals *(Cont'd)***

Position	Signal Name	Description
22	i_tp3_start_detect_sync	TP3 start detect
23	bw_changed	Bandwidth change status
24	link_bw_high	Indicates the 2.7 Gbps line rate selection
25	link_bw_hbr2	Indicates the 5.4 Gbps line rate selection
26	link_bw_rbr	Indicates the 1.62 Gbps line rate selection
31:27	i_drp_state	DRP state
32	i_drp_enable	Enable to the DRP
33	i_drp_write	Write enable to the DRP
34	i_drp_ready00	DRP ready from line0
35	i_drp_ready01	DRP ready from line1
36	i_drp_ready10	DRP ready from line2
37	i_drp_ready11	DRP ready from line3
53:38	i_drp_read_data	Read data from DRP
69:54	i_drp_write_data	Write data to DRP
77:70	I_drp_addr	DRP address
81:78	lane_count	Lane count
82	dpcd_mst_en	MST enable status from DPCD 0x111 bit[0]
83	dpcd_up_req_en	MST up request enable status from DPCD 0x111 bit[1]
84	dpcd_upstream_is_src	Upstream is source indication. Bit[2] of DPCD 0x111
85	dpcd_enhanced_frame_en	Enhanced frame mode selection
87:86	link_qual_pattern	Link quality pattern selection
89:88	dpcd_power_state	DPCD power state
90	dpcd_downspread_control	DPCD down speed control
91	I_rx_user_ready	GT RX userready
92	phy_use_dfe	USE DFE selection to the GT
93	i_pll_lock_detect_tile_0*	PLL lock status for line0 and line1
94	i_pll_lock_detect_tile_1*	PLL lock status for line2 and line3
95	pll1_lock_in (GTP) / gt0_qpllclk_lock (GTX/GTH)	PLL1 lock input (GTP) QPLL lock (GTX/GTH)

AUX Debug Signals

Table 3-10 lists the AUX Debug signals.

Table 3-10: **AUX Debug Signals**

Position	Signal Name
0	aux_data_in
1	aux_data_out
2	aux_data_enable_n
3	hot_plug_detect

Clocking

This section contains detailing about clocking.

Transceiver Reference Clock

The reference clock for the transceivers is lnk_clk_p/n.

- For 7 series FPGAs, a common reference clock of 135 MHz (harmonic of 27 MHz) is needed for 1.62, 2.7 and 5.4 Gb/s link rates. For more details, see the Transceiver Interface, page 98 for the Source core or Transceiver Interface, page 105 for the Sink core.
- For UltraScale™ FPGAs, reference clock of 135 MHz is needed for 2.7 and 5.4 Gb/s link rates. To support 1.62 Gb/s link rate, a common reference clock of 270 MHz is needed.

The core uses six clock domains:

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

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

  Note: lnk_clk = link_rate/20, when GT-Data width is 16-bit. lnk_clk = link_rate/40, when GT-Data width is 32-bit.

- **vid_clk.** This is the primary user interface clock. It has been tested to run as fast as 150 MHz, which accommodates to a screen resolution of 2560x1600 when using two-wide pixels and larger when using the four-wide pixels. See Selecting the Pixel Interface in Chapter 3 for more information on how to select the appropriate pixel interface. Based on the *DisplayPort Standard*, the video clock can be derived from the link clock using `mvid` and `nvid`.

- **s_axi_aclk.** This is the processor domain. It has been tested to run as fast as 135 MHz. The AUX clock domain is derived from this domain, but requires no additional constraints. In UltraScale FPGA s_axi_aclk clock is connected to a free-running clock input. gtwiz_reset_clk_freerun_in is required by the reset controller helper block to reset the transceiver primitives. For details on the clock frequency recommendation, refer *UltraScale FPGAs Transceivers Wizard LogiCORE IP Product Guide*[Ref 18]. A new GUI parameter is added for AXI_Frequency, when the Displayport IP is targeted to Ultrascale FPGA. The requirement is s_axi_aclk <= lnk_clk.
- **aud_clk**. This is the audio interface clock. The frequency will be equal to 512 x audio sample rate. See Audio Clocking (Recommendation) for more details.
- **s_aud_axis_aclk**. This clock is used by the source Audio streaming interface. This clock should be = 512 x audio sample rate.
- **m_aud_axis_aclk**. This clock is used by the sink audio streaming interface. This clock should be = 512 x audio sample rate.

Resets

Resets for the Source and Sink cores of the DisplayPort solution are as follows:

- Source Core Resets
  - **s_axi_aresetn**. AXI Reset. Active-Low. Synchronous to `s_axi_aclk`.
  - **tx_vid_rst**. User video reset. Synchronous to `vid_clk`.
  - **aud_rst**. Audio Interface Reset. Active-High. Synchronous to `aud_clk`.
  - **s_aud_axis_aresetn**. Audio Streaming Interface Reset. Active-Low. Synchronous to `s_aud_axis_aclk`.
- Sink Core Resets
  - **s_axi_aresetn**. AXI Reset. Active-Low. Synchronous to `s_axi_aclk`.
  - **rx_vid_rst**. User video reset. Synchronous to `vid_clk`.
  - **aud_rst**. Audio Interface Reset. Active-High.
  - **m_aud_axis_aresetn**. Audio Streaming Interface Reset. Active-Low. Synchronous to `m_aud_axis_aclk`.

Shared Logic

Shared Logic provides a flexible architecture that works both as a standalone core and as a part of a larger design with one or more core instances. This minimizes the amount of HDL modifications required while retaining the flexibility to address more uses of the core.

The DisplayPort core v4.0 and prior has a fixed RTL hierarchy. A fixed RTL hierarchy limits logic sharing such as clock management logic, quad PLLs for GTs and reset state machines across multiple instances of the same core.

The shared logic RTL hierarchy is called <component_name>_support. Figure 3-29 and Figure 3-30 show two hierarchies where the shared logic block is contained in the core or in the example design, respectively. In these figures, <component_name> is the name of the generated core. The difference between the two hierarchies is the boundary of the core. It is controlled using the Shared Logic Tab in the Vivado IDE.

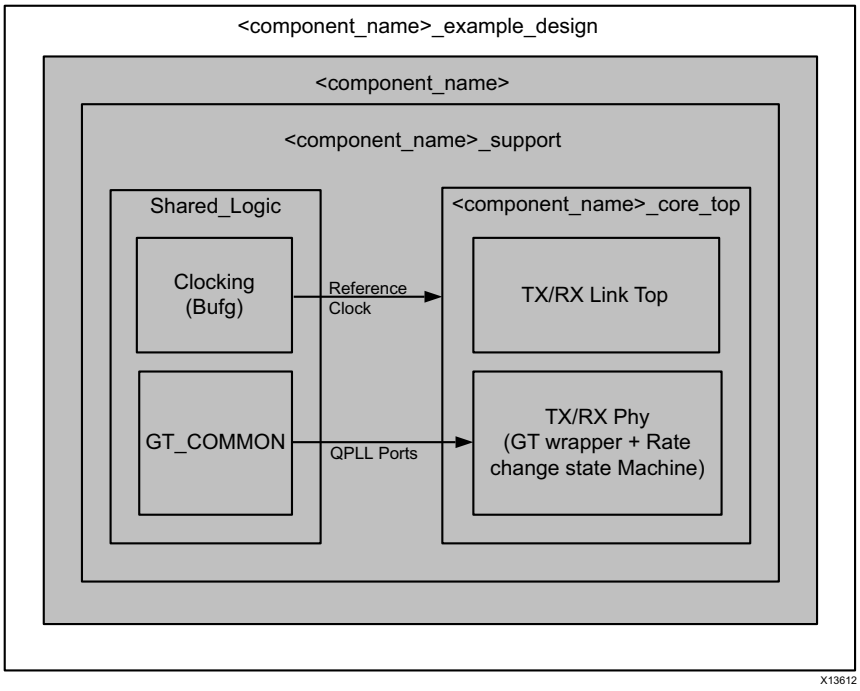

Figure 3-29: **Shared Logic Included in Core**

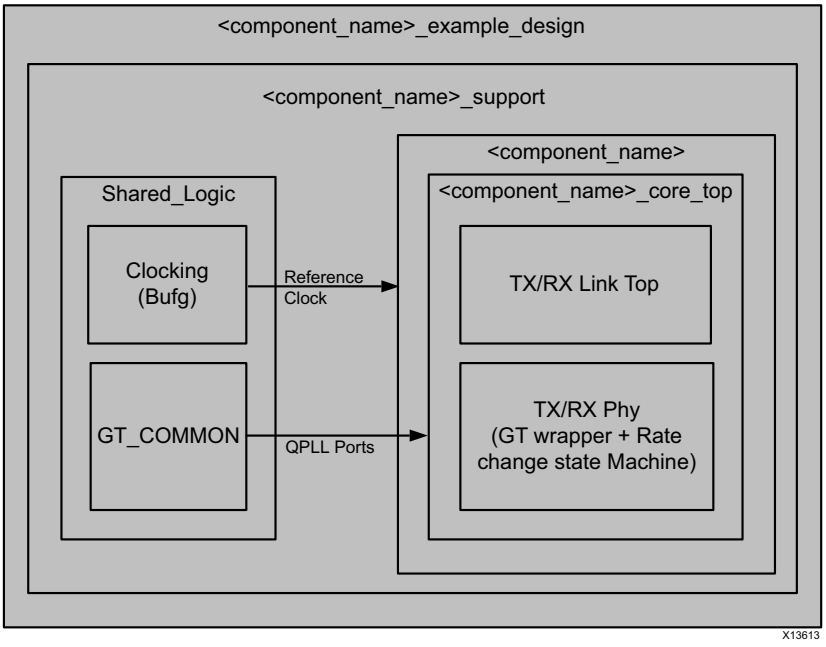

Figure 3-30: **Shared Logic Included in Example Design**

32-bit GT Interface Design Considerations

Select a GT Interface width if 32 bits to ease timing when 5.4 Gb/s designs are targeted on a -1 speed grade device. This allows you to operate on a lower `lnk_clk`.

Note: The Displayport IP supports UltraScale GTH only. The clocking scheme for the design is same for 16-bit and 32-bit interfaces.

The clocking scheme for the design is the same for 16 bit and 32 bit interfaces for GTX and GTH devices. For GTP devices, an MMCM is added in the TX PHY module (for Source) and RX PHY module (for Sink) to generate the appropriate `lnk_clk` for the core and TX/RXUSRCLKs for the GTs.

IMPORTANT: *Xilinx recommends that the MMCM settings remain unchanged. The MMCM DRP module is used to dynamically change the divider settings of the MMCM when a bandwidth change occurs. See XAPP888 for details about the MMCM DRP module [Ref 17].*

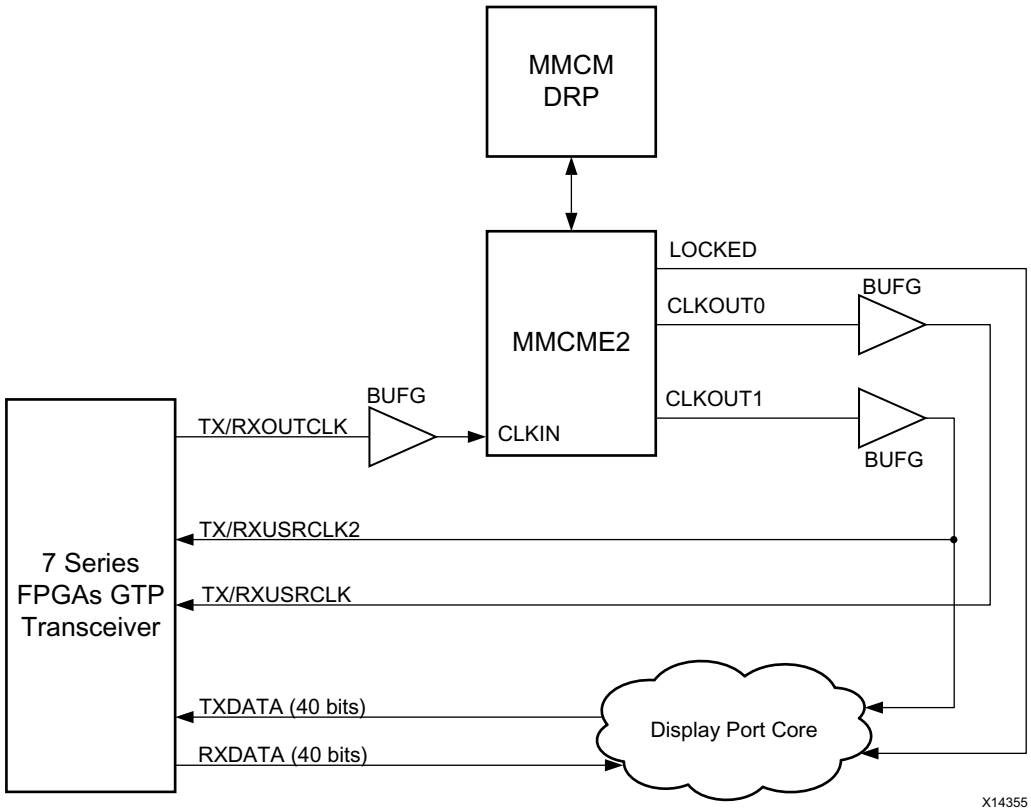

Figure 3-31: **32-Bit GT Interface**

The data bus width and associated data/control signal width is modified to suit 32-bit GT interfaces. For the 16-bit mode of operation, the lower 16 bits of the bus are used.

The DisplayPort TX core has specific constraints in SST/MST mode when used with a GT interface with a data width of 32 bits. To meet the constraints, you need to balance the number of lanes with the link rate. Other than the following constraints, there are no differences in programming the DP TX/RX for a GT interface data width of 16 bits.

- MIN_BYTES_PER_TU (Offset 0x1C4) should be greater than or equal to 4.
- In MST mode, the VC Payload size should be multiple of 4.

Design Flow Steps

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

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 3]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 12]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 15]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 10]

Customizing and Generating the Core

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

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 3] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values 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.

The Source (TX) and Sink (RX) core are generated independently through the Xilinx Vivado software using the Vivado Integrated Design Environment (IDE).

This section describes the options used to generate and customize the cores. The Source and Sink cores are generated independently, and you may choose to generate only one or both cores. You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

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

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 12] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 15].

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

Configuration Tab

Figure 4-1 shows the DisplayPort Vivado IDE main configuration screen. Descriptions of the Vivado IDE options on this screen are provided in the following text.

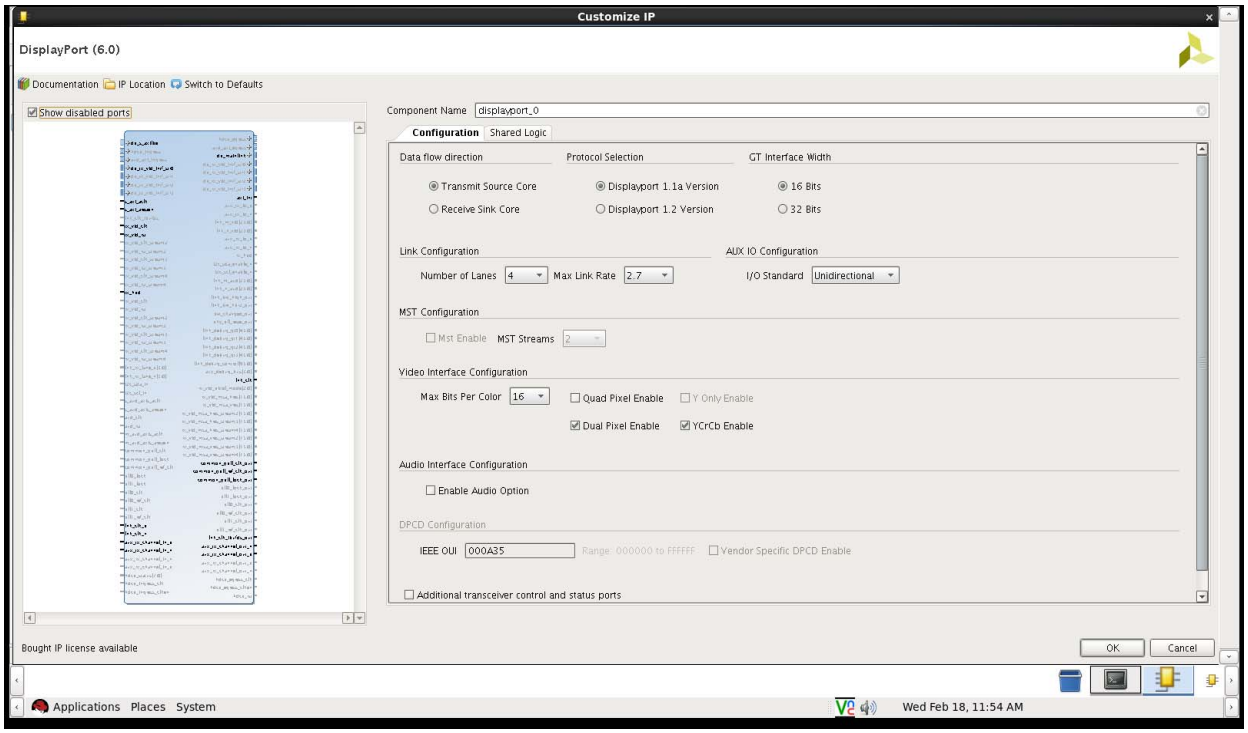

Figure 4-1: **DisplayPort IP Configuration Tab**

- **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.
- **Data Flow Direction**: Select either the Sink (RX) or Source (TX) core with the Data Flow Direction radio button. If both directions are desired, you must generate both a TX and RX core separately and combine these with the supplied wrapper files.
- **Protocol Selection**: Select the protocol version for which the core is to be generated.
  - **DisplayPort 1.1** : DisplayPort Standard 1.1 features.
  - **DisplayPort 1.2** : DisplayPort Standard 1.2 features.

- **GT Interface Width:** Selects the GT interface data width per lane. The 32-bit data width configuration improves the timing for lower speed grade devices.
- **Link Configuration**:
  - **Number of Lanes**: Choose 1, 2, or 4 maximum lanes. Choose fewer lanes for a more optimized design. More lanes allow for higher overall bandwidth and higher resolutions
  - **Max Link Rate**: Select the maximum link rate to be supported by the design.
- **AXI Frequency**: Set the frequency of AXI clock connected in MHz. This is required to be set for UltraScale™ devices only.
- **AUX IO Configuration**: Select this option to choose bidirectional or unidirectional IOs.
- **MST Configuration**:
  - **MST Enable**: Select to enable the multi-streaming (defaults to two streams) support.
  - **MST Streams**: Choose 2, 3, or 4 streams based on the desired application.
- **Video Interface Configuration**:
  - **Max Bits Per Color**: Choose the maximum bits per color that the core supports. The default is 16 bits per color.
  - **Quad Pixel Enable**: Select this check box to enable the four pixel-wide video interface. The quad pixel interface option is available for four-lane designs.
  - **Dual Pixel Enable**: Select this check box to enable the two pixel-wide video interface. The dual pixel interface option is available for two- and four-lane designs.
  - **Y Only Enable**: Select this check box to enable Y-Only color space logic. This option is available on cores using *DisplayPort Standard v1.2* only.
  - **YCRCB Enable**: Select this check box to enable YCRCB-4:2:2 color space.
- **Audio Interface Configuration**:
  - **Enable Audio Option**: Select this check box to enable generation of the core with eight-channel audio support.
  - **Audio Channels**: Select the number of audio channels.
- **DPCD Configuration**:
  - **IEEE OUI**: This Receiver Sink core option allows you to preset the OUI register value before synthesis generation. The value defaults to Xilinx's OUI.
  - **Vendor Specific DPCD Enable**: The Receive Sink core check box to enable use of a vendor-specific DPCD area. Select this check box to enable this logic.
- **Additional Transceiver Control and Status Ports**: Select this option to enable top-level debug ports.

Shared Logic Tab

Figure 4-2 shows the DisplayPort Vivado IDE shared logic tab. Descriptions of the Vivado IDE options on this screen are provided in the following text.

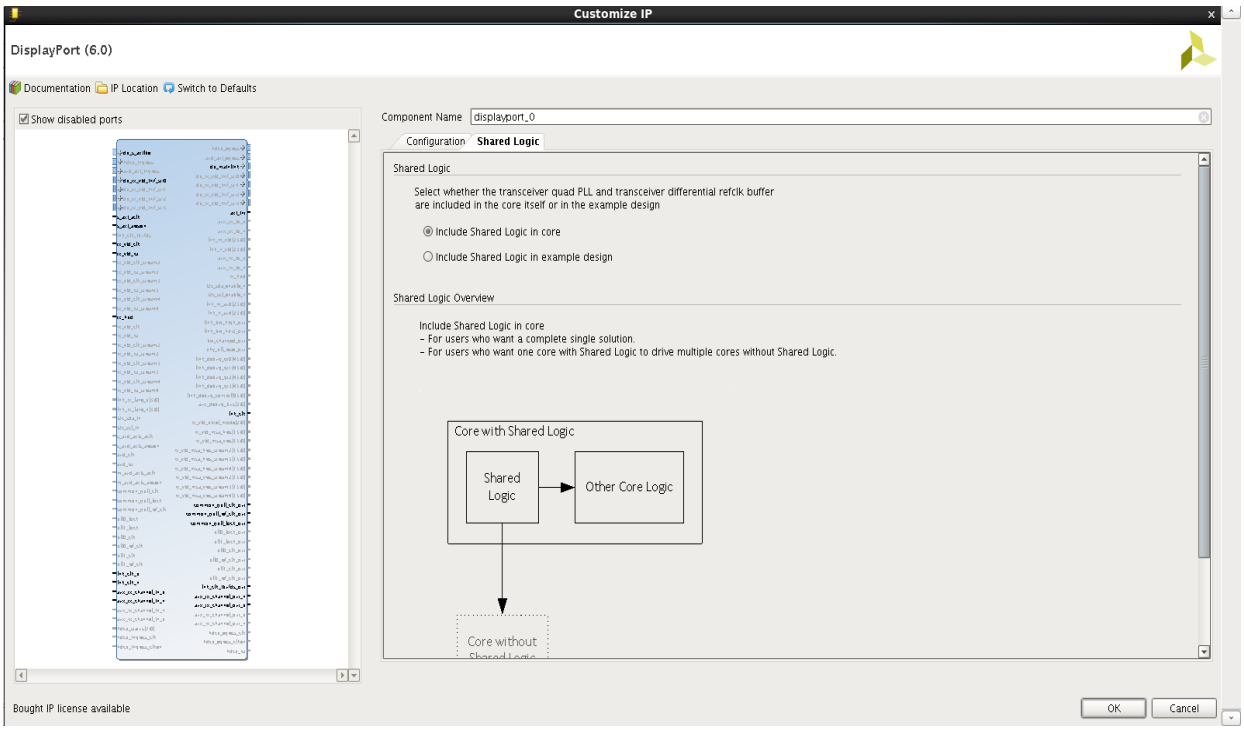

Figure 4-2: **DisplayPort IP Shared Logic Tab**

Select whether shared logic (including clocking, PLL and reset logic) is included in the core itself or in the example design (see Shared Logic in Chapter 3).

Parameterization

This section contains details about parameterization of the Source and Sink cores.

Source Core Parameterization

You may specify a number of options through the Vivado tool, which will determine the presence of certain functions. It is advisable to disable any feature that is not needed in order to reduce resource utilization. Table 4-1 describes the parameterizable options.

Table 4-1: **Parameterizable Options**

Parameter	Default Value	Description
LANE_SUPPORT	4	{1, 2, 4} Indicates the maximum number of lanes to be supported for transmission. Note that unused lane support hardware will be removed from the design.
LINK_RATE	2.7	{1.62, 2.7, 5.4} Indicates the maximum link rate in Gb/s supported by the design.
SECONDARY_SUPPORT	0	Enables secondary channel logic to send Audio packets.
AUDIO_CHANNELS	2	Current version of IP supports 2-channel audio. The value is hard coded.
MST_ENABLE	0	Multi-stream support is enabled when protocol selection is DisplayPort 1.2.
NUMBER_OF_MST_STREAMS	2	Indicates the total number of streams supported. Enabled when MST_ENABLE is TRUE.
PROTOCOL_SELECTION	0	Protocol selection: • 0: DisplayPort 1.1a • 1: DisplayPort 1.2
MAX_BITS_PER_COLOR	16	{8, 10, 12, 16} Sets maximum bits-per-color support and optimizes IP accordingly.
QUAD_PIXEL_ENABLE	0	Enables support of quad-pixel video interface.
DUAL_PIXEL_ENABLE	1	Enables support of dual-pixel video interface.
YCRCB_ENABLE	1	Enables YCrCb 4:2:2 colorimetry support.
YONLY_ENABLE	0	Enables Y-Only colorimetry support

Sink Core Parameterization

The user may specify a number of options through the Vivado tool, which will determine the presence of certain functions. Table 4-2 shows the parametrization options.

TIP: *Note that it is advisable to disable any feature that is not needed in order to reduce resource utilization.*

Table 4-2: **Parameterizable Options**

Parameter	Default Value	Description
LANE_SUPPORT	4	{1, 2, 4} Indicates the maximum number of lanes to be supported for transmission. Note that unused lane support hardware will be removed from the design.
LINK RATE	2.7	{1.62, 2.7, 5.4 } Indicates the maximum link rate in Gb/s supported by the design.

Table 4-2: **Parameterizable Options *(Cont'd)***

Parameter	Default Value	Description
SECONDARY_SUPPORT	0	Enables secondary channel logic to send Audio packets.
AUDIO_CHANNELS	2	Current version of IP supports 2-channel audio. The value is hard-coded.
PROTOCOL_SELECTION	0	Protocol selection: • 0: DisplayPort Standard v1.1a • 1: DisplayPort Standard v1.2
MST_ENABLE	0	Multi-stream support is enabled when protocol selection is DisplayPort 1.2.
NUMBER_OF_MST_STREAMS	2	Indicates the total number of streams supported. Enabled when MST_ENABLE is TRUE.
MAX_BITS_PER_COLOR	8	Sets maximum bits per color support and optimizes IP accordingly.
QUAD_PIXEL_ENABLE	0	Enables support of quad-pixel video interface.
DUAL_PIXEL_ENABLE	1	Enables support of dual-pixel video interface.
YCRCB_ENABLE	1	Enables YCrCb 4:2:2 colorimetry support.
YONLY_ENABLE	0	Enabled Y-Only colorimetry support.
IEEE_OUI	24'h000A35	{24-bit value} Indicates the user's OUI value
VENDOR_SPECIFIC	0	Enables DPCD space of vendor-specific fields in the Sink core.

User Parameters

Table 4-3 shows the relationship between the GUI fields in the Vivado IDE and the User Parameters (which can be viewed in the Tcl console).

Table 4-3: **GUI Parameter to User Parameter Relationship**

GUI Parameter/Value[(1)]	User Parameter/Value[(1)]	Default Value
Component Name	Component_Name	displayport_v4_2
Data flow direction	Data_flow_direction	Transmit_Source_Core
Transmit Source Core	Transmit_Source_Core	
Receive Sink Core	Receive_Sink_Core	
Protocol Selection	Protocol_Selection	DP_1_1_A
Displayport 1.1a Version	DP_1_1_A	
Displayport 1.2 Version	DP_1_2	
Link Configuration		
Number of Lanes: 1, 2, 4	Number_of_Lanes: 1, 2, 4	4
Max Link Rate: 1.62, 2.7, 5.4	Link_Rate: 1.62, 2.7, 5.4	2.7
AUX IO Configuration		

Table 4-3: **GUI Parameter to User Parameter Relationship *(Cont'd)***

GUI Parameter/Value[1]	User Parameter/Value[1]	Default Value
I/O Standard	aux_io_type	1
Bidirectional: 0	Bidirectional: 0	
Unidirectional: 1	Unidirectional: 1	
MST Configuration		
Mst Enable: FALSE, TRUE	MST_Enable: FALSE, TRUE	FALSE
MST Streams: 2, 3, 4	Number_of_MST_Streams: 2, 3, 4	2
Video Interface Configuration		
Max Bits Per Color: 8, 10, 12, 16	Max_Bits_Per_Color: 8, 10, 12, 16	16
Quad Pixel Enable: FALSE, TRUE	Quad_Pixel_Enable: FALSE, TRUE	FALSE
Dual Pixel Enable: FALSE, TRUE	Dual_Pixel_Enable: FALSE, TRUE	FALSE
YCrCb Enable: FALSE, TRUE	YCRCB_Enable: FALSE, TRUE	FALSE
Y Only Enable: FALSE, TRUE	YOnly_Enable: FALSE, TRUE	FALSE
Audio Interface Configuration		
Enable Audio Option: FALSE, TRUE	Enable_of_Audio_Channels: FALSE, TRUE	FALSE
Audio Channels: 1-8	Number_of_Audio_Channels: 1-8	2
DPCD Configuration		
IEEE OUI: Range - 0x00000 to 0xFFFFFF	IEEE_OUI: Range - 0x00000 to 0xFFFFFF	000A35
Vendor Specific DPCD Enable: FALSE, TRUE	Vendor_Specific: FALSE, TRUE	FALSE
Additional transceiver control and status ports: FALSE, TRUE	TransceiverControl: FALSE, TRUE	FALSE
Shared Logic	SupportLevel	1
Include Shared Logic in core	1	
Include Shared Logic in example design	0	

Notes:

1. Parameter values are listed in the table where the GUI parameter value differs from the user parameter value. Such values are shown in this table as indented below the associated parameter.

Output Generation

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

Constraining the Core

This section defines the constraint requirements of the DisplayPort core. An example user constraints file (XDC) is provided along with the example design, which implements the constraints defined in this chapter.

When a Kintex®-7 is selected as the target device, the XDC will be generated for an XC7K325T-FFG900-2 device as an example. The example designs and XDCs can be retargeted for other devices. Information is provided in this chapter to indicate which constraints to modify when targeting devices other than those shown in the example designs.

Board Layout

For board layout concerns, refer to the *VESA DisplayPort Standard*[Ref 2]. For layout of the high-speed I/O lanes, refer to the appropriate section of the relative transceiver user guide. See References in Appendix D. Special consideration must be made for the AUX channel signals. See I/O Standard and Placement.

Required Constraints

To operate the core at the highest performance rating, the following constraints must be present. Prorate these numbers if slower performance is desired.

```
create_clock -name lnk_clk_p -period 7.407 [get_ports lnk_clk_p]# 135MHz ( Till 2.7G)
create_clock -name axi_aclk  -period 20    [get_pins -hier *s_axi_aclk]# 50MHz
create_clock -name vid_clk -period 6.667 [get_pins -hier *tx_vid_clk]# 150MHz
```

IP-level constraints are generated along with core. System-level constraints should be defined as needed. For reference, see the example design XDC file.

Device, Package, and Speed Grade Selections

Supported devices are listed in IP Facts, page 4. For 5.4 G rate, a device speed grade of -1 is supported when GT Data width is selected as 32 bits. See 32-bit GT Interface Design Considerations in Chapter 3 for more details.

Clock Frequencies

See Maximum Frequencies in Chapter 2 for more details about clock frequencies.

Clock Management

See Clocking in Chapter 3 and Shared Logic in Chapter 3 for details about clock management.

Transceiver Placement

Placement of the GT is board specific. For designs that target certain parts and families, the GT placement is set in the constraints file.

I/O Standard and Placement

This section contains details about I/O constraints.

AUX Channel

The *VESA DisplayPort Standard* [Ref 1] describes the AUX channel as a bidirectional LVDS signal. For 7 series designs, the core uses IOBUFDS (bi-directional buffer) as the default with the LVDS standard. You should design the board as recommended by the VESA DP Protocol Standard. For reference, see the example design XDC file.

For Kintex®-7 and Artix®-7 devices supporting HR IO banks, use the following constraints:

For Source:

```
set_property IOSTANDARD LVDS_25   [get_ports aux_tx_io_p]
set_property IOSTANDARD LVDS_25   [get_ports aux_tx_io_n]
```

For Sink:

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

For Virtex®-7 devices supporting HP IO banks, use the following constraints:

For Source:

```
set_property IOSTANDARD LVDS_25   [get_ports aux_tx_io_p]
set_property IOSTANDARD LVDS_25   [get_ports aux_tx_io_n]
```

For Sink:

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

HPD

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

For Kintex-7 and Artix-7 devices supporting HR IO banks, use the following constraints:

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

For Virtex-7 devices supporting HP IO banks, use the following constraints:

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

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

High-Speed I/O

The four high-speed lanes operate in the LVDS (LVDS25) IO standard. Board design and connectivity should follow DP standard recommendations.

Simulation

For details, see the Vivado User Guide: Logic Simulation (UG900) [Ref 10].

IMPORTANT: *For cores targeting 7 series or Zynq-7000 devices, UNIFAST libraries are not supported. Xilinx IP is tested and qualified with UNISIM libraries only.*

Synthesis and Implementation

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

Detailed Example Design

This chapter provides detailed information about the example design, including a description of files and the directory structure generated by the Xilinx Vivado® tool, the purpose and contents of the provided scripts, the contents of the example HDL wrappers, and the operation of the demonstration test bench.

Detailed information about available example designs for the DisplayPort core can also be found in the *DisplayPort Transmit Reference Design Application Note* (XAPP1178) [Ref 8].

Top-Level Example Design

The following files describe the top-level example design for the DisplayPort cores.

```
<project_dir>/<displayport_component_name>/example_design/<component_name>_exdes.v
```

The top-level example design adds flip-flops to the user data interface. This allows the entire design to be synthesized and implemented in a target device to provide post place-and-route gate-level simulation.

Policy Maker

The following files describe the Policy Maker design for the DisplayPort cores:

Sink Core

```
<project_dir>/<displayport_component_name>/example_design/
<displayport_component_name>_dport_rx_fsm_cntrl.v
```

Source Core

```
<project_dir>/<displayport_component_name>/example_design/
<displayport_component_name>_dport_tx_fsm_cntrl.v
```

Each policy maker design contains a state machine, which connects to the processor interface. An instruction set has been stored in RAM, which can be modified as you see fit. The basic instruction set provided demonstrates the rudimentary procedure for setting up the cores.

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

EDID ROM

These fully functional Sink-only files demonstrate how to connect an EDID to the core.

```
<project_dir>/<displayport_component_name>/example_design/
<displayport_component_name>_iic_edid_rom.vhd

<project_dir>/<displayport_component_name>/example_design/
<displayport_component_name>_iic_rom.vhd
```

Additionally, this EDID can be used in hardware. Adjust the register values as needed.

Test Bench

This chapter contains information about the test bench provided in the Vivado® Design Suite.

The demonstration test bench is a simple Verilog program to exercise the example design and the cores. The following files describe the demonstration test bench.

Source Core

The Source test bench file is located at:

```
<project_dir>/<displayport_component_name>/example_design/simulation/
<displayport_component_name>_tb.v
```

The source demonstration test bench performs the following tasks:

- Generates input clock signals
- Applies a reset to the example design
- Asserts HPD to the Source core
- Responds to AUX channel requests
- Drives video data on the user data interface

Sink Core

The Sink test bench file is located at:

```
<project_dir>/<displayport_component_name>/example_design/simulation/
<displayport_component_name>_tb.v
```

The sink demonstration test bench performs the following tasks:

- Generates input clock signals
- Applies a reset to the example design
- Sets the lane count of the Sink core to 4 through the AUX channel

- Sets the bandwidth of the Sink core to 2.7 Gb/s through the AUX channel
- Alerts the Sink core that training is beginning
- Sends training patterns 1 and 2 across the high-speed lanes
- Sets the power state value through the AUX channel

Verification, Compliance, and Interoperability

The DisplayPort cores have been verified with functional simulation and extensive hardware testing for v1.1a and some hardware testing for v1.2. Interoperability tests for DisplayPort Standard v1.2 features are in progress.

Simulation

A parameterizable transaction-based test bench was used to test the core. Broad protocol and implementation-specific coverage were used to fully verify the cores. The tests included the following:

- Full I2C operation over the AUX channel
- Bandwidth and performance tests
- Main link stress tests
- Processor interface register read and write accesses
- Scramble/Descramble quality checks
- Video and Audio data integrity checks

Hardware Testing

The DisplayPort cores have been validated using a Kintex®-7 FPGA Evaluation Kit (KC705). Preliminary testing for MST TX and RX interoperability has been completed.

Migrating and Upgrading

This appendix contains information about migrating a design from ISE® to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating to the Vivado Design Suite

The BASEADDR, HIGHADDR, SPDIF_BASEADDR and SPDIF_HIGHADDR parameters are not available when using the core with the Vivado Design Suite. For information about migrating to the Vivado Design Suite, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 16].

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations that take place when you upgrade to a more current version of this IP core in the Vivado Design Suite.

Parameter Changes

Table B-1 lists the parameter changes for this version of the core.

Table B-1: **Parameter Changes**

Parameter	Default Value	Comments
GT Data Width	16	Parameter added
aux_io_type	Unidirectional	Parameter added
Support Level	Include Shared Logic in core	Parameter added
Transceiver Control	FALSE	Parameter added

Port Changes

The ports are enabled based on your configuration of the IP. As a result, the upgrade log for the DisplayPort core shows warnings of detecting external port differences when upgrading IP from old versions. Table B-2 shows the ports that might be affected.

Table B-2: **Port Changes**

Interface / Signal name	Comments
TX User Data Interface	Ports are enabled only in Transmit Source core configuration. Enabling the tx_vid_pixel 0,1,2,3 ports is done based on the setting of the dual/quad pixel parameter.
TX Main Link Interface	Ports are enabled only in Transmit Source core configuration. The width of lnk_tx_lane_p/n is based on the link configuration.
Streaming audio Interface	Enabled only when audio is enabled.
Audio Clock Interface	Enabled only when audio is enabled.
RX User Data Interface	Ports are enabled only in Receive Sink Core configuration. Enabling the rx_vid_pixel0,1,2,3 ports is done based on the setting of the dual/quad pixel parameter.
RX Main Link Interface	Ports are enabled only in Receive Sink Core configuration. Also, width of lnk_rx_lane_p/n are based on the link configuration
Streaming audio Interface	Enabled only when audio is enabled.
AUX Channel Interface	aux_tx_io_p/n is enabled only in Transmit Source core, and aux_rx_io_p/n is enabled only in Receive Sink core.
I2C Interface	Enabled only in Receive Sink core configuration.
tx_hpd	Enabled only in Transmit Source core configuration.
rx_hpd	Enabled only in Receive Sink core configuration.
Ports when Shared Logic Enabled	Additional ports have been added for shared logic. See Port Descriptions in Chapter 2 for more details

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 Licensing and Ordering Information in Chapter 1 for more details.*

Finding Help on Xilinx.com

To help in the design and debug process when using the DisplayPort, the Xilinx Support web page (www.xilinx.com/support) 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. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page (www.xilinx.com/support) or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Design Tools tab on the Downloads page (www.xilinx.com/download). 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 are listed below, and 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 Core

AR 54522

Contacting Technical Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support:

1. Navigate to www.xilinx.com/support.
2. Open a WebCase by selecting the WebCase link located under Additional Resources.

When opening a WebCase, include:

- Target FPGA including package and speed grade.
- All applicable Xilinx Design Tools and simulator software versions.
- Additional files based on the specific issue might also be required. See the relevant sections in this debug guide for guidelines about which file(s) to include with the WebCase.

Note: Access to WebCase is not available in all cases. Log in to the WebCase tool to see your specific support options.

Debug Tools

There are many tools available to address DisplayPort design issues. It is important to know which tools are useful for debugging various situations.

Vivado Lab Edition

Vivado® Lab Edition inserts logic analyzer and virtual I/O cores directly into your design. Vivado Lab Edition also allows you to set trigger conditions to capture application and

integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx devices.

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

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

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. Xilinx recommends having an external auxiliary channel analyzer to understand the transactions between the Source and Sink cores.

General Checks

Ensure that all the timing constraints for the core were properly incorporated from the example design and that all constraints were met during implementation.

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the `locked` port.
- If your outputs go to 0, check your licensing.
- Check that the reset polarities are properly connected.

Source Controller

This section contains debugging steps for issues with the Source controller.

- If the monitor is not displaying video:
  - Check if Source controller is able to train the Monitor (Sink). Read Lane Status from Monitor (Sink). Make sure the Source core setup and initialization steps provided in Source Overview in Chapter 3 are followed correctly. Ensure that PHY 8b10b enable bit is set in offset 0x200 for normal video.
  - Check if USER_DATA_COUNT_PER_LANE, MIN_BYTES_PER_TU and FRAC_BYTES_PER_TU are calculated based on the available bandwidth.

    - Check if the video input to the Source controller is following proper video timing and has the proper video pixel frequency. Custom logic or a VTC (Video Timing Controller) Detector can be used to check the video timing.

- If the AUX reads always or randomly fail:
    - Check if the AUX connectivity at the IO level is correct.
    - Check the noise levels of AUX bus using the debug ports or with an external probe.
    - Check if the HPD is asserted to confirm Sink device is connected. Plug and unplug the device to see if the HPD activity is changing.
    - Check if AUX_CLOCK_DIVIDER is properly programmed to 1 Mb/s rate at the AUX bus.
- If audio is not played at Sink device:
    - Check if the audio data is transferred at the Source AXI4-Stream interface.
    - Follow steps mentioned in Audio Management in Chapter 3.

Sink Controller

This sections contains debugging steps for issues with the Sink controller.

If the monitor is not displaying video or if there is a training loss:

- Check if the Vivado Logic Analyzer is trained. Read Lane Status from Monitor (Sink) using the AXI Interface.
    - In the Vivado Lab Edition, add all lane_data, lane_kchar and training pattern signals from PHY module.
        - Trigger use of training pattern and capture snapshots during training pattern1/ pattern 2 and random snapshots by using Blanking Start Symbols as the trigger.
    - Check if GT PLL Lock is achieved. If not, check clock connectivity.
    - Check if GT RESET_DONE signals are asserted. If not, check PLL lock and reset logic.
    - All active lanes or some active lanes have disparity and symbol errors.
        - Add a counter to symbol error flags and monitor the error count.
        - If the issue persists, check the signal integrity of the board and review with Xilinx GT experts.

If the monitor display is noisy and has many errors:

- Check if the video pipeline at the Sink controller output is using single/dual/quad pixel based on programming.
- Check the MSA & MISC0 registers to ensure the proper video format and BPC is used in the system.

- Check the link errors in SYM_ERR_CNT01 and SYM_ERR_CNT23 registers.
- Ensure a certified DP1.2 cable is used.

If the Source is unable to read/write using the AUX channel:

- Check if the AUX connectivity at the IO level is correct.
- Check the noise levels of the AUX bus.
- Check if AUX_CLOCK_DIVIDER is properly programmed to 1 Mb/s rate at AUX bus.

If audio is not played at the Sink device or the audio is noisy:

- Check if the audio data is received at the Sink AXI4-Stream interface.
- Check if the info packet is received. It is not mandatory to receive the info packet, but it would confirm that the secondary channel is active (if source transmits the info packet).
- For noisy audio, check if there are so many errors that the error decoder in the Sink core is not be able to correct them.
- Follow steps mentioned in Audio Management in Chapter 3.

Appendix D

Additional Resources and Legal Notices

Xilinx Resources

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

References

These documents provide supplemental material useful with this user guide:

1. VESA *DisplayPort Standard v1.1a*, January 11, 2008
2. VESA *DisplayPort Standard v1.2*, December 22, 2009
3. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
4. *Implementing a DisplayPort Source Policy Maker Using a MicroBlaze Embedded Processor* (XAPP493)
5. High-bandwidth Digital Content Protection System v1.3 Amendment for DisplayPort, v1.0
6. *AMBA AXI Protocol*, v2.0
7. *7 Series FPGAs GTX Transceivers User Guide* (UG476)
8. *DisplayPort Transmit Reference Design Application Note* (XAPP1178)
9. *Xilinx Vivado AXI Reference Guide* (UG1037)
10. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
11. *Vivado Design Suite User Guide: Implementation* (UG904)
12. *Vivado Design Suite User Guide: Designing with IP* (UG896)
13. *Vivado Design Suite Migration Methodology Guide* (UG911)
14. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
15. *Vivado Design Suite User Guide: Getting Started* (UG910)

16. *ISE to Vivado Design Suite Migration Methodology Guide* (UG911)
17. *MMCM and PLL Dynamic Reconfiguration* (XAPP888)
18. UltraScale FPGAs Transceiver Wizard Product Guide (PG182)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/01/2015	6.0	• Added ultrascale GTH simulation support. • Added external Audio streaming interface ports. Internal SPDIF IP support removed. • Added 3-8 channel audio support in SST. • Added 8b10b encoding control in TX PHY Configuration register.
11/19/2014	5.0	• Updated CORE_ID values for DisplayPort Standard v1.2a protocol in Table 2-10. • Updated stream numbering in Table 2-10. • Updated receiver core initialization shown in Figure 3-10.
10/01/2014	5.0	• Added 32-Bit GT Data Width support. • Added MST Support in RX. • Added programming sequence of the MST RX.
06/04/2014	4.2	• Added GUI Parameter to User Parameter mapping table. • Added polarity information for the User Data Interface.
12/18/2013	4.2	• Added details about the Pixel clock options. • Updated details about MST TX programming. • Added details about debugging the transceiver.
10/02/2013	4.1	Revision number advanced to 4.1 to align with core version number. Added Shared Logic in Chapter 3, Debug Interface and Debug Interface.
03/20/2013	2.0	Updated core version to v4.0. Added support for Multi-Stream Transport (MST).
07/25/2012	1.0	Initial Xilinx release as a Product Guide. Replaces DS802, *LogiCORE IP DisplayPort Data Sheet* and UG767, *LogiCORE IP DisplayPort User Guide*. Added Control PHY Power Down register. Added Artix®-7 device support.

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 http://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 http://www.xilinx.com/legal.htm#tos.