UG0935
User Guide
DisplayPort IP

Microsemi
a Microchip company

a Microchip company

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

50200935. 2.0 11/21

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 2.0

Removed the reference design demo information.

1.2 Revision 1.0

The first publication of this document.

2 Introduction

2.1 Overview

DisplayPort IP is targeted for the PolarFire FPGA application and includes DisplayPort Tx IP and DisplayPort Rx IP. These two IP implement part of the DisplayPort 1.4 Link Layer function.

2.2 Key Features

The key features of DisplayPort Tx and Rx IP are listed as follows:

- Support 1, 2, or 4 lanes.
- Support 8 bpc RGB/YCbCr 4:4:4 (24 bits per pixel).
- Support up to 8.1 Gbps per lane.
- Support DisplayPort 1.4 protocol.
- Only support a single video stream or SST mode, and the MST mode is not supported.
- Audio transmission is not supported.

2.3 Supported Families

- PolarFire® SoC
- PolarFire®
- RTG4™
- IGLOO®2
- SmartFusion®2

2.4 License

DisplayPort IP clear RTL is license locked and the obfuscated RTL available for free.

2.4.1 Obfuscated

Complete RTL code is provided for the core, allowing the core to be instantiated with the SmartDesign tool. Simulation, synthesis, and layout can be performed within Libero® System-on-Chip (SoC). The RTL code for the core is obfuscated.

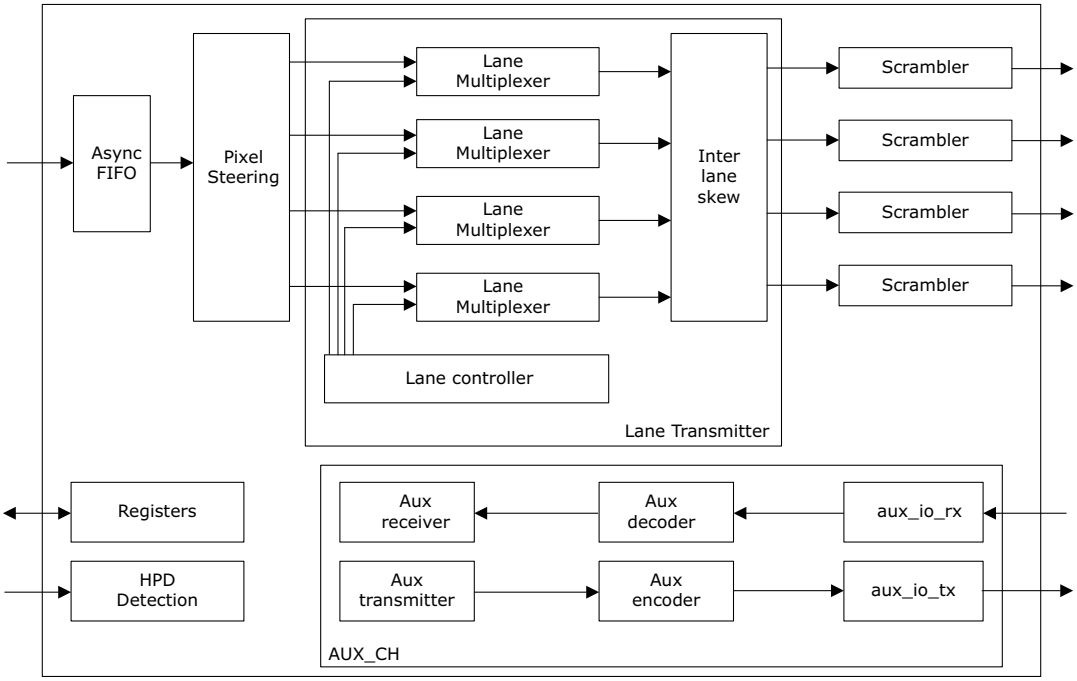
2.4.2 RTL

Complete RTL source code is provided for the core.

2.5 DisplayPort Tx IP Architecture

The following figure shows the DisplayPort Tx IP implementation.

Figure 1 • **DisplayPort Tx IP Implementation**



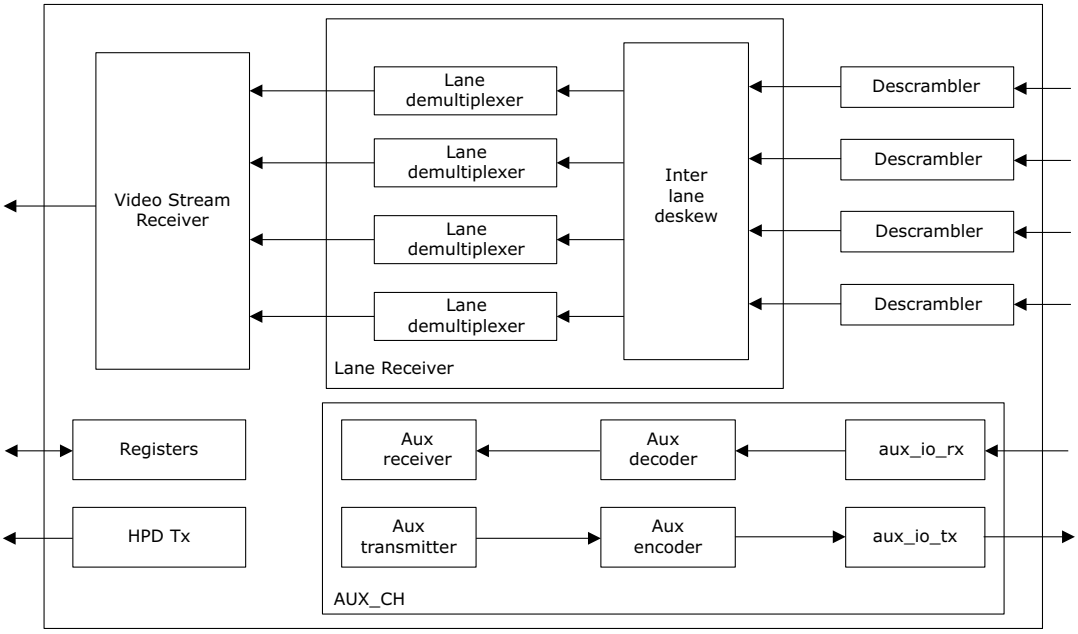
As shown in the preceding figure, DisplayPort Tx IP includes the Pixel Steering module, Lane transmitter module, Scrambler module, and AUX_CH module.

Pixel Steering assigns input pixels to DisplayPort lanes. The lane transmitter module generates a video data stream with blank data or training data. Scrambler scrambles the lane transmission data. AUX_CH module transmits the AUX Request command to the DisplayPort Sink device or receives AUX Reply from the DisplayPort Sink device.

2.6 DisplayPort Rx IP Architecture

The following figure shows the DisplayPort Rx IP implementation.

Figure 2 • **DisplayPort Rx IP Implementation**



DisplayPort Rx IP includes the Descrambler module, Lane receiver module, Video Stream Receiver module, and AUX_CH module. Descrambler de-scrambles the input lane data. Lane receiver demultiplexes all kinds of data on each lane. The Video Stream Receiver gets video pixels from the lane receiver, it recovers the video stream signal. AUX_CH module receives the AUX Request command from DisplayPort Source device and transmits AUX Reply to DisplayPort Source device.

2.7 Resource Utilization

The following table lists the resource utilization of PolarFire FPGA family in DisplayPort IP (configured for 24 bits per pixel and four parallel pixels on the interface).

Table 1 • **DisplayPort IP Resource Utilization**

DisplayPort IP	LUT	DFF	uSRAM	LSRAM
DisplayPort Tx IP	13434	9958	53	3
DisplayPort Rx IP	28840	13674	32	28

3 Functional Description

3.1 HPD

DisplayPort Tx IP detects Hot Plug Detect (HPD) assertion, de-assertion, and HPD interrupt event. It reports the HPD event through an interrupt. After HPD assertion, which means a DisplayPort monitor is connected, DisplayPort Source application software should start the training procedure.

DisplayPort Rx IP outputs HPD signal according to DisplayPort Sink application software settings. After DisplayPort Rx IP is ready, DisplayPort Sink application software should set the HPD signal to 1. When it expects the DisplayPort Source device to re-read Sink device status or re-training, DisplayPort Sink application software should set an HPD to generate the HPD interrupt signal.

3.2 AUX Channel

DisplayPort Source and Sink device communicate through an AUX Channel. Source device sending request transaction to the Sink device and the Sink device sending Reply transaction to Source Device.

DisplayPort Tx and Rx IP implements the AUX transaction transmitter and receiver. For AUX transaction transmitter, DisplayPort Source or Sink application software provides all AUX transaction content bytes, DisplayPort Tx and Rx IP generate the transaction bitstream. For the AUX transaction receiver, DisplayPort Tx and Rx IP receive the transaction and extract all bytes to DisplayPort application software.

The Link Policy Maker and Stream Policy Maker should be implemented in the DisplayPort application software.

3.3 Video Stream Transmission

DisplayPort Tx and Rx IP supports 8 bpc RGB/YCbCr 4:4:4, and only supports a single video stream.

After training is done and the video stream is ready, DisplayPort Tx IP and RX IP start to transmit video stream.

DisplayPort Source application software should configure the video stream attribute MSA and enable DisplayPort Tx IP video transmission. The VSC packet is not supported. DisplayPort Tx and Rx IP are using MISC0 and MISC1 in MSA for Pixel Encoding/Colorimetry Format Indication.

After training, DisplayPort Rx IP should be enabled for video receive. DisplayPort Rx IP gets the received MSA and sets the recovery video format parameter to let DisplayPort Rx IP output the correct video stream signal.

DisplayPort Tx and Rx IP support synchronous video clock mode and asynchronous video clock mode. For synchronous video clock mode, Mvid and Nvid are fixed value in MSA, DisplayPort Source application software should set Mvid and Nvid. For asynchronous video clock mode, Nvid is fixed and configured by software. The user needs to generate Mvid and send Mvid to DisplayPort Tx IP.

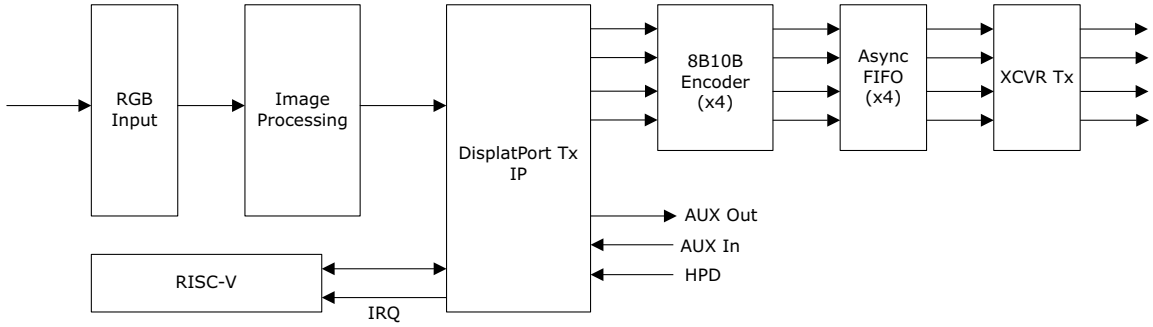
DisplayPort Rx IP does not include a video clock recovery function. The user should recovery the video clock outside DisplayPort Rx IP or use a fixed high enough frequency clock to output video stream data.

4 Typical Application

4.1 DisplayPort Tx IP Application

The following figure shows the typical DisplayPort Tx IP application.

Figure 3 • **Typical application for DisplayPort Tx IP**



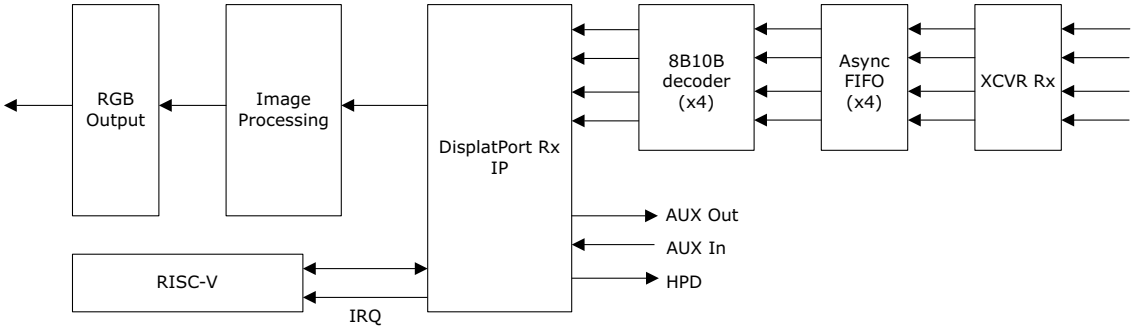
As shown in the preceding figure, the RGB input module interface gets a video stream signal. The Image Processing module processes the video stream according to system requirements. It outputs the video stream to DisplayPort Tx IP. DisplayPort Tx IP outputs four lanes data to the 8B10B encoder. After 10B encoding, lanes' data are transmitted through Transceiver lanes.

Before video stream transmission, the DisplayPort Source application software which is running on RISC-V, controls DisplayPort Tx IP to finish training work with the attached DisplayPort Sink device. To do training and Link Policy Maker, all transactions are transmitted on the AUX Channel.

4.2 DisplayPort Rx IP Application

The following figure shows the typical DisplayPort Rx IP application.

Figure 4 • **Typical application for DisplayPort Rx IP**



As shown in the preceding figure, the transceiver block receives four lanes data. There are four asynchronous FIFO to synchronize all lanes data into one clock domain. These four lanes data are decoded to 8B code in 8B10B decoder modules. DisplayPort Rx IP gets lanes' 8B data and output video stream data; it also works with RISC-V software to finish the training and Link Policy Maker. The recovered video stream data is processed in the Image Processing module and generates output on the RGB output interface.

5 DisplayPort Tx IP Interface Signal

5.1 Interface

The following table shows the input and output ports for DisplayPort Tx IP.

Table 2 • **DisplayPort Tx IP Interface signals**

Interface	Width	Direction	Description
vclk_i	1	Input	Video clock
vrst_n_i	1	Input	Low-active reset signal synchronized with vclk_i
dpclk_i	1	Input	DisplayPort IP working clock. It is DisplayPortLaneRate/40. For example, DisplayPort lane rate is 2.7 Gbps, dpclk_i is 2.7 Gbps/40 = 67.5 MHz.
dprst_n_i	1	Input	Low-active reset signal synchronized with dpclk_i
aux_clk_i	1	Input	AUX Channel clock. It is 100 MHz.
aux_rst_n_i	1	Input	Low-active reset signal synchronized with aux_clk_i
pclk_i	1	Input	APB interface clock
prst_n_i	1	Input	Low-active reset signal synchronized with pclk_i
paddr_i	16	Input	APB address
pwrite_i	1	Input	APB write signal
psel_i	1	Input	APB select signal
penable_i	1	Input	APB enable signal
pwdata_i	32	Input	APB writing data
prdata_o	32	Output	APB reading data
pready_o	1	Output	APB reading data ready signal
async_mvid_i	24	Input	Mvid for asynchronous video clock mode, this signal should be synchronized with dpclk_i. This signal would be ignored for synchronous video clock mode.
int_o	1	Output	Interrupt signal to CPU
vsync_i	1	Input	VSYNC for input video stream. It should be synchronous with vclk_i.
hsync_i	1	Input	HSYNC for input video stream. It should be synchronous with vclk_i.
pixel_val_i	1/2/4	Input	Indicates the validation of pixels on pixel_data_i port, synchronous with vclk_i
pixel_data_i	48/96/192	Input	Input video stream pixel data. It could be 1, 2, or 4 parallel pixels. It should be synchronous with vclk_i. For four parallel pixels, bit[191:144] for 1st pixel, bit[143:96] for 2nd pixel, bit[95:48] for 3rd pixel, and bit[47:0] for 4th pixel. Each pixel uses 48 bits, for RGB, bit[47:32] is R, bit[31:16] is G, bit[15:0] is B. Each color component uses the lowest BPC bits. For example, RGB with 24 bits per pixel, bit[7:0] is B, bit[23:16] is G, and bit[39:32] is R. All other bits are reserved.
hpd_i	1	Input	HPD input signal

Table 2 • **DisplayPort Tx IP Interface signals *(continued)***

Interface	Width	Direction	Description
aux_tx_en_o	1	Output	AUX Tx data enable signal
aux_tx_io_o	1	Output	AUX Tx data
aux_rx_io_i	1	Input	AUX Rx data
dp_lane_k_o	16	Output	DisplayPort output lanes' data K indication. It is synchronous with dpclk_i. Bit[15:12] for Lane0, bit[11:8] for Lane1, bit[7:4] for Lane2, and bit[3:0] for Lane3.
dp_lane_data_o	128	Output	DisplayPort output lanes' data. It is synchronous with dpclk_i. Bit[127:96] for Lane0, bit[95:64] for Lane1, bit[63:32] for Lane2, and bit[31:0] for Lane3.

5.2 Configuration Parameters

The configuration parameter for CoaXPress Host IP is listed in the following table.

Table 3 • **Configuration parameters for CoaXPress Host IP**

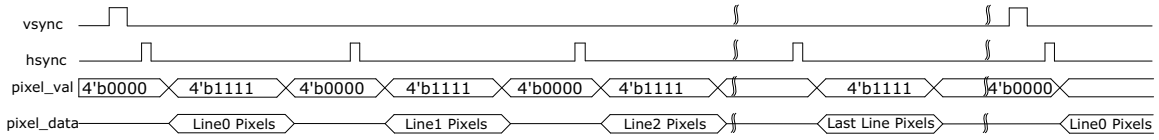
Name	Default	Description
g_IN_PIXEL_NUM	4	It defines the parallel pixel number on input video stream port.

5.3 Key Interface Description

5.3.1 Input Video Stream Interface Signal

The following figure shows the timing diagram of the input video stream signal interface.

Figure 5 • **Timing Diagram for Input Video Stream Interface Signal**



As shown in the preceding figure, hsync_i is asserted for several cycles before each line. If there are n lines in a video frame, there are n hsync_i asserted. Before the first line and the first asserted hsync_i, vsync_i should be asserted for several cycles.

For four parallel pixels input case, pixel_val_i should be 4'b1111 when input pixel data is available and 4'b0000 when input pixel data is not available. For two parallel pixels input case, pixel_val_i should be 2'b11 when input pixel data is available and 2'b00 when pixel data is not available. For the one pixel input case, pixel_val_i should be 1'b1 when input pixel data is available and 1'b0 when pixel data is not available.

6 DisplayPort Rx IP Interface Signal

6.1 Interface

The following table shows the input and output ports for DisplayPort Rx IP.

Table 4 • **DisplayPort Rx IP Interface signals**

Interface	Width	Direction	Description
vclk_i	1	Input	Video clock
vrst_n_i	1	Input	Low-active reset signal synchronized with vclk_i
dpclk_i	1	Input	DisplayPort IP working clock. It is DisplayPortLaneRate/40. For example, DisplayPort lane rate is 2.7 Gbps, dpclk_i is 2.7 Gbps/40 = 67.5 MHz
dprst_n_i	1	Input	Low-active reset signal synchronized with dpclk_i
aux_clk_i	1	Input	AUX Channel clock. It is 100 MHz.
aux_rst_n_i	1	Input	Low-active reset signal synchronized with aux_clk_i
pclk_i	1	Input	APB interface clock
prst_n_i	1	Input	Low-active reset signal synchronized with pclk_i
paddr_i	16	Input	APB address
pwrite_i	1	Input	APB write signal
psel_i	1	Input	APB select signal
penable_i	1	Input	APB enable signal
pwdata_i	32	Input	APB writing data
prdata_o	32	Output	APB reading data
pready_o	1	Output	APB reading data ready signal
int_o	1	Output	Interrupt signal to CPU
vsync_o	1	Output	VSYNC for output video stream. It is synchronous with vclk_i.
hsync_o	1	Output	HSYNC for output video stream. It is synchronous with vclk_i.
pixel_val_o	1/2/4	Output	Indicates the validation of pixels on pixel_data_o port, synchronous with vclk_i
pixel_data_o	48/96/192	Output	Output video stream pixel data. It could be 1, 2, or 4 parallel pixels. It is synchronous with vclk_i. For four parallel pixels, bit[191:144] for 1st pixel, bit[143:96] for 2nd pixel, bit[95:48] for 3rd pixel, and bit[47:0] for 4th pixel. Each pixel uses 48 bits, for RGB, bit[47:32] is R, bit[31:16] is G, and bit[15:0] is B. Each color component uses the lowest BPC bits. For example, RGB with 24 bits per pixel, bit[7:0] is B, bit[23:16] is G, and bit[39:32] is R. All other bits are reserved.
hpd_o	1	Output	HPD output signal
aux_tx_en_o	1	Output	AUX Tx data enable signal
aux_tx_io_o	1	Output	AUX Tx data
aux_rx_io_i	1	Input	AUX Rx data

Table 4 • **DisplayPort Rx IP Interface signals *(continued)***

Interface	Width	Direction	Description
dp_lane_k_i	16	Input	DisplayPort input lanes' data K indication. It is synchronous with dpclk_i. Bit[15:12] for Lane0, bit[11:8] for Lane1, bit[7:4] for Lane2, and bit[3:0] for Lane3.
dp_lane_data_i	128	Input	DisplayPort input lanes' data. It is synchronous with dpclk_i. Bit[127:96] for Lane0, bit[95:64] for Lane1, bit[63:32] for Lane2, bit[31:0] for Lane3.
mvid_val_o	1	Output	Indicates if mvid_o and nvid_o is available It is synchronous with dpclk_i.
mvid_o	24	Output	Mvid, it is synchronous with dpclk_i.
nvid_o	24	Output	Nvid, it is synchronous with dpclk_i.

6.2 Configuration Parameters

The configuration parameter for DisplayPort Rx IP is listed in the following table.

Table 5 • **Configuration parameters for DisplayPort Rx IP**

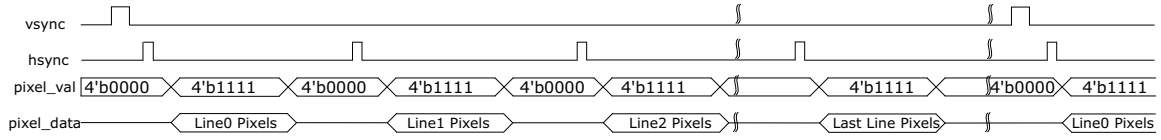
Name	Default	Description
g_MAX_OUT_PIXEL	4	Max output parallel pixels
g_LINE_BUFFER_DEPTH	2048	Output line buffer depth. It should be greater than line pixel number.

6.3 Key Interface Description

6.3.1 Output Video Stream Interface Signal

The following figure shows the timing diagram of the output video stream signal interface.

Figure 6 • **Timing Diagram for Output Video Stream Interface Signal**



As shown in the preceding figure, hsync_o is asserted for several cycles before each line. If there are n lines in a video frame, there are n hsync_o asserted. Before the first line and the first asserted hsync_o, vsync_o is asserted for several cycles. The position and width of VSYNC and HSYNC are configured by software.

7 DisplayPort Tx IP Configuration

7.1 HPD Detection

DisplayPort Tx IP detects the input HPD signal to check the status of the attached DisplayPort Sink device. The HPD event includes three types.

- HPD assertion, it means DisplayPort Sink device is connected.
- HPD de-assertion, it means DisplayPort Sink device is disconnected.
- HPD interrupt, it means the DisplayPort Sink device's status has changed.

When the HPD event is detected and interrupt is enabled, DisplayPort Tx IP could output a pulse on the int_o port and indicates the interrupt type on interrupt register 0x0188.

7.2 Transmit AUX Request Transaction

To transmit a Native Writing AUX Request transaction, DisplayPort Tx application software should do the following steps:

1. Write all the writing bytes into register 0x010C, write one byte for each writing operation.
2. Write the DPCD address into register 0x0104.
3. Write (WritingBytesNum-1) into register 0x0108.
4. Write ((WritingBytesNum << 16) | (0x00000001 << 8) | 0x00000008) into registers 0x0100.

To transmit a Native Reading AUX Request transaction, DisplayPort Tx application software should do the following steps:

1. Write the DPCD address into register 0x0104.
2. Write (ReadingBytesNum-1) into register 0x0108.
3. Write ((0x00000000<<16) | (0x00000001<<8) | 0x00000009) into registers 0x0100.

To transmit an I2C-Over-AUX Writing Request transaction, DisplayPort Tx application software should do the following steps:

1. Write all the writing bytes into register 0x010C, write one byte for each writing operation.
2. Write the DPCD address into register 0x0104.
3. Write (WritingBytesNum-1) into register 0x0108.
4. Write ((WritingBytesNum << 16) | (0x00000001 << 8) | (MOT<<2) | 0x00000000) into registers 0x0100.

To transmit an I2C-Over-AUX Reading Request transaction, DisplayPort Tx application software should do the following steps:

1. Write the DPCD reading address into register 0x0104.
2. Write (ReadingBytesNum-1) into register 0x0108.
3. Write ((0x00000000<<16) | (0x00000001<<8) | (MOT<<2) | 0x00000001) into registers into 0x0100.

7.3 Receive AUX Reply Transaction

After sending an AUX Request transaction to the DisplayPort Sink device, DisplayPort Source application software should wait for the AUX Reply transaction. When the AUX Reply is arrived and interrupt is enabled, DisplayPort Tx IP could output an interrupt signal and record this event in the interrupt register.

To read the received AUX Reply transaction from DisplayPort Tx IP, software should do the following steps:

1. Read register 0x012C to know AUX Reply transaction length AuxReplyByteNum.
2. Read register 0x0124 AuxReplyByteNum times to get all the AUX Reply Transaction bytes.
3. Software checks the Reply type by checking the first reading transaction byte bit [7:4], it could be AUX ACK, NACK, or AUX DEFER. Bit [3:0] is reserved.
4. If AuxReplyByteNum > 1, the followed bytes are reading data from the DPCD registers.

7.4 DisplayPort Lanes Training

At the first training stage, DisplayPort Tx IP should output TPS1 to get the attached DisplayPort Sink device to get LANEx_CR_DONE. The software should configure the following steps to enable TPS1 transmission:

1. Write enabled lane number into register 0x0004, it could be enabled 4 lanes, 2 lanes, or 1 lane.
2. Write 0x01 into register 0x0018 to enable TPS1.
3. Write 0x00 into register 0x0010 to disable scrambler.

At second training stage, according to the DisplayPort Sink device feature, DisplayPort Tx IP should output TPS2/TPS3/TPS4 to get the attached DisplayPort Sink device to get LANEx_EQ_DONE, LANEx_SYMBOL_LOCKED, and INTERLANE_ALIGN_DONE. The software should configure the following steps to enable TPS2/TPS3/TPS4 transmissions:

1. Write enabled lane number into register 0x0004. It could be enabled 4 lanes, 2 lanes, or 1 lane.
2. To transmit TPS2, write 0x02 into register 0x0018 to enable TPS2. For TPS3, writing 0x03. For TPS4, writing 0x04.
3. For TPS2 and TPS3, write 0x00 into register 0x0010 to disable scrambler. For TPS4, write 0x01 to enable scrambler.

In the training procedure, before sending the TPS pattern, DisplayPort Source application software might need to configure Transceiver SI settings and the Transceiver rate. The Transceiver is not part of this IP, and the Transceiver settings configuration guide is not included in this user guide.

7.5 Video Stream Transmission

After training is completed, DisplayPort Tx IP can transmit the video stream to the sink device. To enable video transmission, software should do the following configuration:

1. Enable scrambler, write 0x01 into register 0x0010.
2. Configure MSA, configure registers from address 0x00C0 to address 0x00EC.
3. Enable video transmission, write 0x01 into register 0x0000.

7.6 Register Definition

The following table shows the internal registers defined in DisplayPort Tx IP.

Table 6 • **DisplayPort Tx IP Registers**

Address	Bits	Name	Type	Default	Description
0x0000	[0]	video_stream_enable	RW	0x0	Enable video stream transmission
0x0008	[2:0]	lane_number	RW	0x4	DisplayPort enabled lane number: 4, 2, or 1
0x0010	[0]	scrambler_enable	RW	0x0	Enable scrambler
0x0014	[0]	Interlane_skew_enable	RW	0x1	Enable Inter lane skew
0x0018	[2:0]	training_pattern_mode	RW	0x0	Training pattern type: 0: None, 1: TPS1, 2: TPS2, 3: TPS,. and 4: TPS4.
0x001C	[0]	enhanced_BS_enable	RW	0x0	Enable Enhanced BS transmission
0x00C0	[23:0]	MSA_Mvid	RW	0x0	MSA Mvid for synchronous video clock mode
0x00C4	[23:0]	MSA_Nvid	RW	0x0	MSA Nvid
0x00C8	[15:0]	MSA_HTotal	RW	0x0	MSA HTotal
0x00CC	[15:0]	MSA_VTotal	RW	0x0	MSA_VTotal
0x00D0	[15:0]	MSA_HStart	RW	0x0	MSA HStart
0x00D4	[15:0]	MSA_VStart	RW	0x0	MSA_VStart

Table 6 • **DisplayPort Tx IP Registers *(continued)***

Address	Bits	Name	Type	Default	Description
0x00D8	[15]	MSA_HSync_Polarity	RW	0x0	MSA_HSync_Polarity
	[14:0]	MSA_HSync_Width	RW	0x0	MSA_HSync_Width
0x00DC	[15]	MSA_VSync_Polarity	RW	0x0	MSA_VSync_Polarity
	[14:0]	MSA_VSync_Width	RW	0x0	MSA_VSync_Width
0x00E0	[7:1]	MSA_MISC0_ColorIndicator	RW	0x0	MSA_MISC0_ColorimetryIndicator
	[0]	MSA_MISC0_SyncClock	RW	0x0	MSA_MISC0_SynchronousClock
0x00E4	[7:6]	MSA_MISC1_ColorIndicator	RW	0x0	MSA_MISC1_ColorimetryIndicator
	[5:3]	MSA_MISC1_Reserved	RW	0x0	MSA_MISC1_Reserved
	[2:1]	MSA_MISC1_Stero	RW	0x0	MSA_MISC1_SteroVideoAttributor
	[0]	MSA_MISC1_InterlaceEven	RW	0x0	MSA_MISC1_InterlacedVerticalTotalEven
0x00E8	[15:0]	MSA_HWidth	RW	0x0	MSA_HWidth
0x00EC	[15:0]	MSA_VHeigth	RW	0x0	MSA_VHeigth
0x0100	[23:16]	AUX_Tx_Data_Byte_Num	RW	0x0	The number of bytes to be transmitted in this AUX request transaction
	[8]	AUX_Tx_LengthField_Val	RW	0x0	1 means this AUX transaction transmission carries length field. 0 means this AUX transaction does not have length filed.
	[3:0]	AUX_Tx_Command	RW	0x0	AUX COMM[3:0] in this AUX transaction transmission
0x0104	[19:0]	AUX_Tx_DPCD_Address	RW	0x0	DPCD address in this AUX transaction transmission
0x0108	[7:0]	AUX_Tx_Length	RW	0x0	Length field in this AUX transaction transmission. It will no ignored if no Length field.
0x010C	[7:0]	AUX_Tx_Writting_Data	RW	0x0	For AUX Writing Request transaction, writing all the DPCD writing bytes into this register byte by byte.
0x0110	[18:0]	AUX_Reply_Timeout_Th	RW	0x001000	Waiting AUX Reply timeout threshold in aux_clk_i cycles
0x011C	[15:0]	AUX_TX_ Request_Num	RC	0x0	The number of AUX transactions to be transmitted
0x0120	[15:0]	AUX_RX_Reply_Num	RC	0x0	The number of AUX transactions to be received
0x0150	[15:0]	HPD_Status	RO	0x0	The status of input HPD signal
0x0154	[0]	HPD_IRQ	RC	0x0	Indicates if there is HPD interrupt
0x0180	[0]	IntMask_TotalInt	RW	0x1	Total interrupt mask. 1 means enable total interrupt and int_o
0x184	[5]	IntMask_HPD_Disconnect	RW	0x1	Interrupt mask for HPD disconnect event. 1 enable interrupt
	[4]	IntMask_HPD_Connect	RW	0x1	Interrupt mask for HPD connect event. 1 enable interrupt

Table 6 • **DisplayPort Tx IP Registers *(continued)***

Address	Bits	Name	Type	Default	Description
	[3]	IntMask_HPD_IRQ	RW	0x1	Interrupt mask for HPD IRQ. 1 enable interrupt
	[2]	IntMask_AuxReplyTimeOut	RW	0x1	Interrupt mask for AUX Reply Timeout. 1 enable interrupt
	[1]	IntMask_NewAuxReply	RW	0x1	Interrupt mask for received AUX Reply. 1 enable interrupt
	[0]	IntMask_AuxTxDone	RW	0x1	Interrupt mask for AUX Request transmission done. 1 enable interrupt
0x0188	[16]	Int_total	RC	0x0	Interrupt: total interrupt
	[5]	Int_HPD_Disconnect	RC	0x0	Interrupt: HPD disconnect event
	[4]	Int_HPD_Connect	RC	0x0	Interrupt: HPD connect event
	[3]	Int_HPD_IRQ	RC	0x0	Interrupt: HPD IRQ
	[2]	Int_AuxReplyTimeOut	RC	0x0	Interrupt: AUX Reply Timeout
	[1]	Int_NewAuxReply	RC	0x0	Interrupt: Received new AUX Reply
	[0]	Int_AuxTxDone	RC	0x0	Interrupt: Transmission of AUX Request is done

8 DisplayPort Rx IP Configuration

8.1 HPD

When the DisplayPort Sink device is ready and connected to the DisplayPort Source device, DisplayPort Sink application software should assert the HPD signal to 1 by writing 0x01 into register 0x0140.

DisplayPort Sink application software should monitor the status of the sink device. If the sink device needs a source device to read the DPCD registers, sink device software should send an HPD interrupt by writing 0x01 into register 0x0144, then write 0x00 into 0x0144.

8.2 Receive AUX Request Transaction

When DisplayPort Rx IP received an AUX Request transaction and interrupt is enabled, the software should receive the NewAuxReply event interrupt. The software should do the following steps to read the received AUX Request transaction from DisplayPort IP:

1. Read register 0x012C to know the length (RequestBytesNum) of the received AUX transaction.
2. Read register 0x0124 RequestBytesNum times to get all the bytes of the received AUX transaction.
3. AUX Request transaction COMM[3:0] is the first reading byte bit [7:4].
4. DPCD address is ((FirstByte[3:0]<<16) | (SecondByte[7:0]<<8) | (ThirdByte[7:0])).
5. AUX Request Length field is FourthByte[7:0].
6. For DPCD writing Request transaction, all the bytes after the length field are writing data.

8.3 Transmit AUX Reply Transaction

After received an AUX Request transaction, the software should configure DisplayPort Rx IP to transmit an AUX Reply transaction as soon as possible. The software is responsible to determine all the Reply transaction bytes, which includs the Reply type.

To transmit an AUX Reply, software should do the following steps:

1. If AUX Reply transaction including DPCD reading data, write all the read data into register 0x010C byte by byte. If no DPCD reading data to be transmitted, skip this step.
2. Determine how many DPCD reading bytes (AuxReadBytesNum). If no DPCD reading bytes, AuxReadBytesNum is 0.
3. Determine the AUX Reply type (ReplyComm).
4. Write ((AuxReadBytesNum<<16) | ReplyComm) into register 0x0100.

8.4 DisplayPort Lanes Training

At the first training stage, the DisplayPort Source device transmits TPS1 to make the attached DisplayPort Sink device to get LANEx_CR_DONE.

At the second training stage, the DisplayPort Source device transmits TPS2/TPS3/TPS4 to get the attached DisplayPort Sink device to get LANEx_EQ_DONE, LANEx_SYMBOL_LOCKED, and INTERLANE_ALIGN_DONE.

This LANEx_CR_DONE means FPGA Transceiver CDR is locked. Since, FPGA Transceiver is not part of DisplayPort Rx IP, LANEx_CR_DONE.

LANEx_SYMBOL_LOCKED means 8B10B decoder decodes 8B bytes correctly. Since the 8B10B decoder is not part of DisplayPort Rx IP, LANEx_SYMBOL_LOCKED.

Before the training procedure, DisplayPort Sink application software should let the source device know that DisplayPort Rx IP supports TPS3 and TPS4.

When the source device is sending TPS3/TPS4 (Source device writes DPCD_0x0102 to indicates TPS3/TPS4 transmission), software should do the following steps to check if training is done:

1. Write enabled lanes number into register 0x0000.
2. Write 0x00 into register 0x0014 to disable descrambler for TPS3. Write 0x01 to enable descrambler for TPS4.
3. Waiting until Source device reading DPCD_0x0202 and DPCD_0x0203 DPCD registers.
4. Read register 0x0038 to know if DisplayPort Rx IP lanes received TPS3. Set LANEx_EQ_DONE to 1 when received TPS3.
5. Read register 0x0018 to know if all lanes are aligned. Set INTERLANE _ALIGN_DONE to 1 if all lanes are aligned.

In the training procedure, the software might need to configure Transceiver SI settings and Transceiver lane rate.

8.5 Video Stream Receiver

After training is completed, DisplayPort Rx IP should enable the video stream receiver. To enable the video receiver, the software should do the following configuration:

1. Write 0x01 into register 0x0014 to enable descrambler.
2. Write 0x01 into register 0x0010 to enable video stream receiver.
3. Reading MSA from register 0x0048 to register 0x006C until getting meaningfully MSA values.
4. Write FrameLinesNumber into register 0x00C0. Write LinePixelsNumber into register 0x00D8. For example, we know it is 1920x1080 video stream from MSA, then write 1080 into register 0x00C0 and write 1920 into register 0x00D8.
5. Read register 0x01D4 to check if the recovered video stream frame has expected HWidth and expected VHeight.
6. Read register 0x01F0 to clear and discard the reading value since this register records the status from the last reading.
7. Waiting for about 1 second or several seconds, Read register 0x01F0 again. Checking bit [5] to check if the recovered video stream HWidth is locked. 1 means unlocked and 0 means locked. Checking bit [21] to check if recovered the video stream VHeight is locked. 1 means unlocked and 0 means locked.

8.6 Register Definition

The following table shows the internal registers defined in DisplayPort Rx IP.

Table 7 • **DisplayPort Rx IP Registers**

Address	Bits	Name	Type	Default	Description
0x0000	[2:0]	Enabled_Lanes_Number	RW	0x4	Enabled lanes number 4 lanes, 2 lanes, or 1 lane
0x0004	[2:0]	Out_Parallel_Pixel_Number	RW	0x4	The number of parallel pixels at video stream output interface
0x0010	[0]	Video_Stream_Enable	RW	0x0	Enable video stream receiver
0x0014	[0]	Descramble_Enable	RW	0x0	Enable descrambler
0x0018	[0]	InterLane_Alignment_Status	RO	0x0	Indicates if lanes are aligned
0x001C	[1]	Alignment_Error	RC	0x0	Indicates if there is error in alignment procedure
	[0]	New_Alignement	RC	0x0	Indicates if there was a new alignment event. When lanes are not aligned, a new alignment is expected. When lanes are aligned and there was a new alignment, it means lanes are out of alignment and aligned again.

Table 7 • **DisplayPort Rx IP Registers *(continued)***

Address	Bits	Name	Type	Default	Description
0x0038	[14:12]	Lane3_RX_TPS_Mode	RO	0x0	Lane3 received TPSx mode. 2 means TPS2, 3 means TPS3, and 4 means TPS4.
	[10:8]	Lane2_RX_TPS_Mode	RO	0x0	Lane2 received TPSx mode
	[6:4]	Lane1_RX_TPS_Mode	RO	0x0	Lane1 received TPSx mode
	[2:0]	Lane0_RX_TPS_Mode	RO	0x0	Lane0 received TPSx mode
0x0044	[7:0]	Rx_VBID	RO	0x00	Received VBID
0x0048	[15:0]	MSA_HTotal	RO	0x0	Received MSA_HTotal
0x004C	[15:0]	MSA_VTotal	RO	0x0	Received MSA_VTotal
0x0050	[15:0]	MSA_HStart	RO	0x0	Received MSA_HStart
0x0054	[15:0]	MSA_VStart	RO	0x0	Received MSA_VStart
0x0058	[15]	MSA_VSync_Polarity	RO	0x0	Received MSA_VSYNC_Polarity
	[14:0]	MSA_VSync_Width	RO	0x0	Received MSA_VSYC_Width
0x005C	[15]	MSA_HSync_Polarity	RO	0x0	Received MSA_HSYNC_Polarity
	[14:0]	MSA_HSync_Width	RO	0x0	Received MSA_HSYNC_Width
0x0060	[15:0]	MSA_HWidth	RO	0x0	Received MSA_HWidth
0x0064	[15:0]	MSA_VHeight	RO	0x0	Received MSA_VHeight
0x0068	[7:0]	MSA_MISC0	RO	0x0	Received MSA_MISC0
0x006C	[7:0]	MSA_MISC1	RO	0x0	Received MSA_MISC1
0x00C0	[15:0]	Video_Frame_Line_Number	RW	0x438	The number of lines in a received video frame
0x00C4	[15:0]	Video_VSYNC_Width	RW	0x0004	Defines the output video VSYNC width in vclk_i cycles
0x00C8	[15:0]	Video_HSYNC_Width	RW	0x0004	Defines the output video HSYNC width in vclk_i cycles
0x00CC	[15:0]	VSYNC_To_HSYNC_Width	RW	0x0008	Defines the distance between VSYNC and HSYNC in vclk_i cycles
0x00D0	[15:0]	HSYNC_To_Pixel_Width	RW	0x0008	Defines the distance between HSYNC and first line pixel in cycles
0x00D8	[15:0]	Video_line_pixels	RW	0x0780	The number of pixels in a received video line
0x0100	[23:16]	AUX_Tx_Data_Byte_Num	RW	0x00	The number of DPCD reading data bytes in the AUX Reply
	[3:0]	AUX_Tx_Command	RW	0x0	The Comm[3:0] in AUX Reply (Reply Type)
0x010C	[7:0]	AUX_Tx_Writing_Data	RW	0x00	Write all DPCD reading data bytes for the AUX Reply
0x011C	[15:0]	Tx_AUX_Reply_Num	RC	0x0	The number of AUX Reply transactions to be transmitted
0x0120	[15:0]	Rx_AUX_Request_Num	RC	0x0	The number of AUX Request transactions to be received
0x0124	[7:0]	AUX_Rx_Read_Data	RO	0x00	Read all bytes of received AUX Request transaction

Table 7 • **DisplayPort Rx IP Registers *(continued)***

Address	Bits	Name	Type	Default	Description
0x012C	[7:0]	AUX_Rx_Request_Length	RO	0x00	The number of bytes in the received AUX Request transaction
0x0140	[0]	HPD_Status	RW	0x0	Set HPD output value
0x0144	[0]	Send_HPD_IRQ	RW	0x0	Write to 1 to send a HPD interrupt
0x0148	[19:0]	HPD_IRQ_Width	RW	0x249F0	Defines the HPD IRQ low-active pulse width in aux_clk_i cycles
0x0180	[0]	IntMask_Total_Interrupt	RW	0x1	Interrupt Mask: total interrupt
0x0184	[1]	IntMask_NewAuxRequest	RW	0x1	Interrupt Mask: Received new AUX Request
	[0]	IntMask_TxAuxDone	RW	0x1	Interrupt Mask: Transmit AUX Reply done
0x01A0	[15]	Int_TotalInt	RC	0x0	Interrupt: total interrupt
	[1]	Int_NewAuxRequest	RC	0x0	Interrupt: Received new AUX Request
	[0]	Int_TxAuxDone	RC	0x0	Interrupt: Transmit AUX Reply done
0x01D4	[31:16]	Video_Output_LineNum	RO	0x0	The number of lines in an output video frame
	[15:0]	Video_Output_PixelNum	RO	0x0	The number of pixels in an output video line
0x01F0	[21]	Video_LineNum_Unlock	RC	0x0	1 means output video frame lines number is not locked
	[5]	Video_PixelNum_Unlock	RC	0x0	1 means output video pixels number is not locked