UG0875

User Guide
CoaXPress IP

Microsemi
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.

50200875. 2.0 10/19

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

Updated for CoaXPress v2.0.

1.2 Revision 1.0

This was the first publication of the document. Created for CoaXPress v1.1

2 Introduction

2.1 Overview

CoaXPress is an interface to connect Devices (typically cameras) to Hosts (typically frame grabbers), it consists of one master connection and optional extension connections, which together forms a link. In the CoaXPress v2.0 protocol, each connection supports up to 12.5 Gbps data rate.

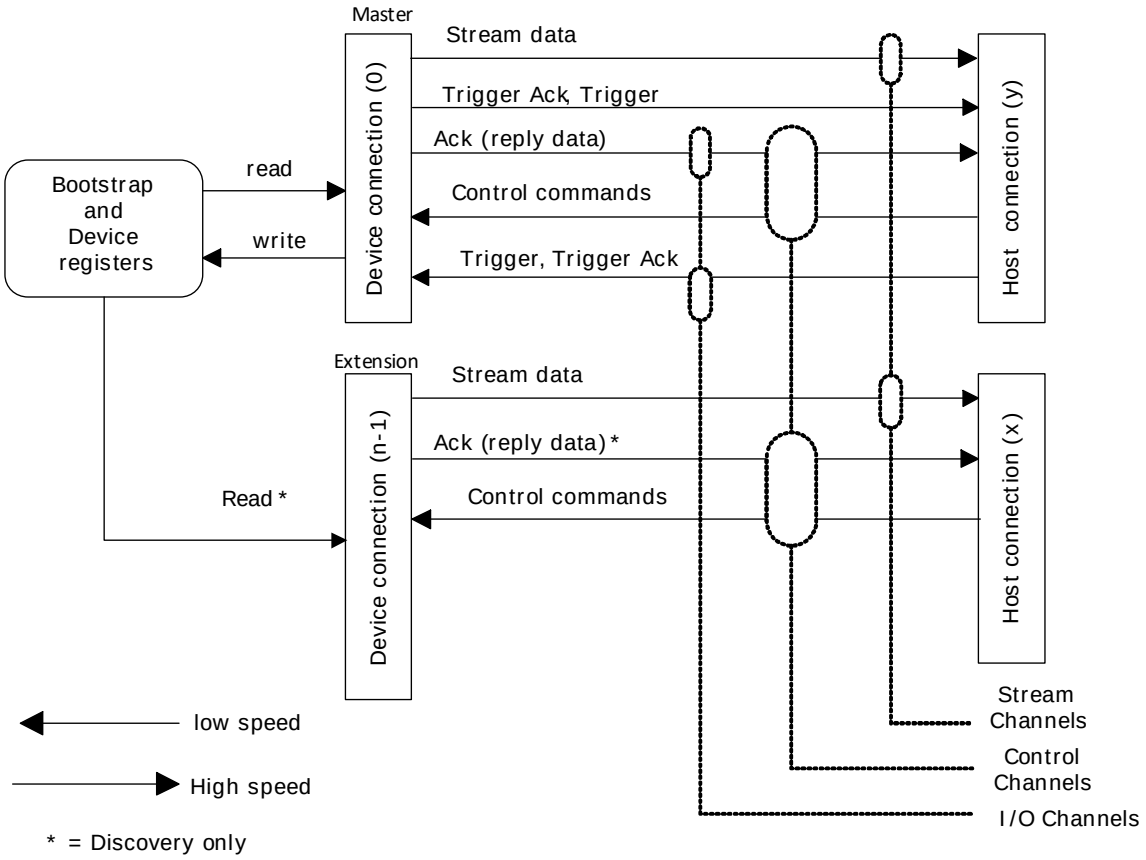
The link consists of a downconnection and upconnection. The link direction from Device to Host is downconnection, and the link direction from Host to Device is upconnection. Upconnection could be a low speed upconnection or high speed upconnection. Low speed upconnection rate is 41.6 Mbps or 20.83 Mbps, while high speed downconnection or high speed upconnection rate could be: 1.25 Gbps, 2.5 Gbps, 3.125 Gbps, 5 Gbps, 6.25 Gbps, or 12.5 Gbps.

Within one connection pair, both the upconnection and downconnection are transmitted on the same coax cable. For each connection, CoaXPress defines a set of logical channels to carry a specific data frame:

- Stream data on stream channel (for example, image data from Device to Host)
- Real time triggers on IO channel (A trigger event occurs when a triggers frame received)
- Device control on Control channel (frames from Host to read and write Device control registers)

The following figure shows the logical channel diagram when high speed up connection is not available.

Figure 1 • **Link Diagram without High Speed Upconnection**



In the preceding diagram, Stream data is transmitted on the stream channel, which is high speed downconnection from Device to Host. Trigger frame includes bidirectional Trigger and bidirectional Trigger Ack. Host transmits, Trigger and Trigger Ack on low speed master upconnection while the Device transmits, Trigger and Trigger Ack on high speed master downconnection. On Master connection, the Host transmits Control commands on low speed upconnection, which can read and write the device registers. The Device transmits registers, Control Command Acknowledgment from Device is transmitted on high speed downconnection. On extension connection, Control commands and Ack are the same as Master connection, except for Control commands are only able to read Device in Device Discovery process.

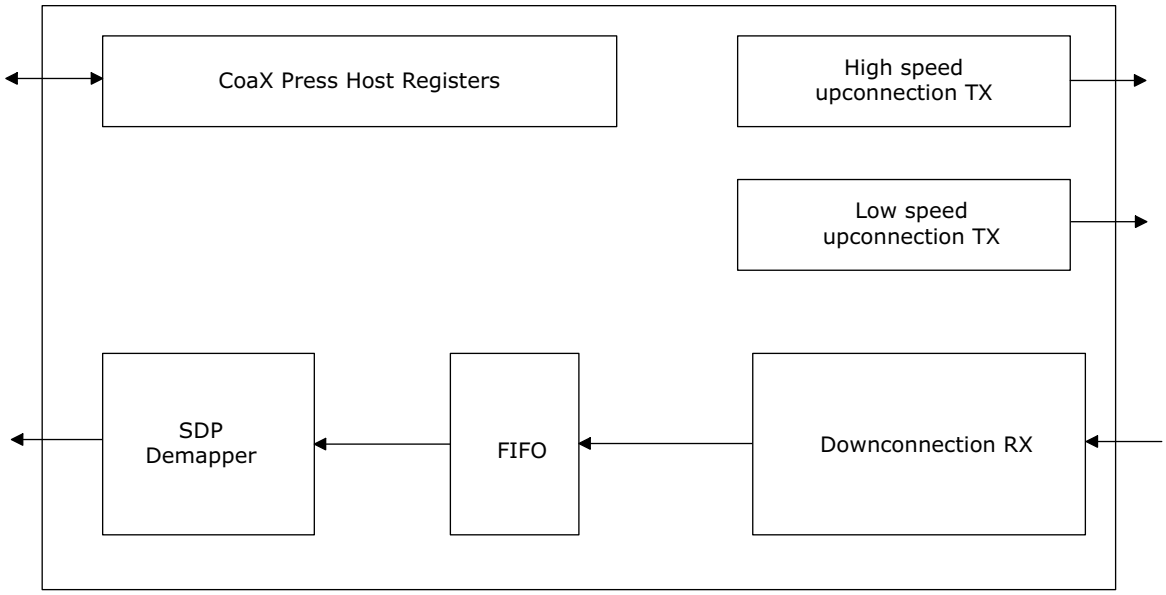
When high speed upconnection is available, the master link upstream function is moved to high speed upconnection.

CoaXPress IP implements the link layer defined in the CoaXPress v2.0 standard. CoaXPress includes CoaXPress Host IP and CoaXPress Device IP.

2.2 CoaXPress Host IP Architecture

The following figure shows the diagram of CoaXPress Host IP implementation:

Figure 2 • **CoaXPress Host IP Implementation Diagram**



As shown in the preceding diagram, CoaXPress Host IP includes downconnection RX module, SDP demapper (supports up to 2 SDP demapper), low speed upconnection TX module, and optional high speed upconnection TX module.

Downconnection RX module receives packets in downconnection channel. The stream data packet and markers are forwarded to SDP demapper, the Command packets and Event packets are buffered for the software reading.

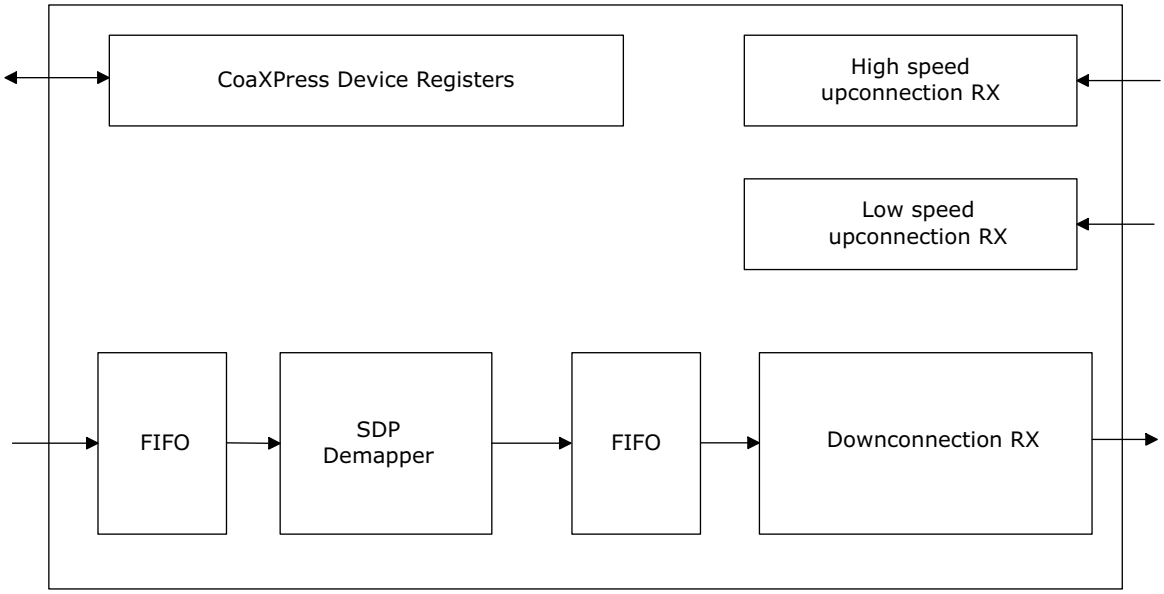
The upconnection TX modules transmit Control packet and Event packets from the software configuration, and it also transmits Trigger and Trigger Ack.

The SDP demapper gets image line packets from stream data packets payload section, it sends the image line packets to the downstream module. It also forwards the frame markers and line markers to the downstream module.

2.3 CoaXPress Device IP Architecture

The following figure shows the diagram of CoaXPress Device IP implementation:

Figure 3 • **CoaXPress Device IP Implementation Diagram**



As shown in the preceding diagram, CoaXPress Device IP includes downconnection TX module, SDP mapper (supports up to 2 SDP mapper), low speed upconnection RX module, and optional high speed upconnection RX module.

The SDP mapper generates stream data packets with image line packets as payload, it also forwards the input frame marker and line marker to downconnection TX module.

The downconnection TX module is downconnection transmitter, it transmits stream data packets, markers, Command packets, Event Packets, Heartbeat packets, Trigger, and Trigger Ack packets.

High speed upconnection RX and low speed upconnection RX module receives Trigger packets and Control Command packets from CoaXPress Host.

2.4 Key Features

CoaXPress IP features are as follows:

- Only a single lane mode.
- Supports high speed upconnection.
- Supports up to 12.5 Gbps on downconnection and high speed upconnection.
- Supports up to 2 Stream IDs (that is, up to 2 SDP mappers and 2 SDP demappers).
- Supports 41.6 Mbps or 20.83 Mbps low speed upconnection.
- Meets the requirements in CoaXPress V2.0.
- Including CoaXPress Host IP and CoaXPress Device IP.

2.5 Resource Utilization

The following table describes the resource utilization in which CoaXPress IP with 2 Stream IDs is implemented in PolarFire device:

Table 1 • **CoaXPress IP Resource Utilization**

Stream ID	LUT	DFF	uSRAM	LSRAM
CoaXPress Host IP	6447	7846	0	9
CoaXPress Device IP	5316	6143	0	9

3 Function Description

3.1 Image Packet Transmission

CoaXPress Device IP accepts image packets from the camera. Each line packet is wrapped into a stream data packet. Also, the frame marker and line marker are required. The frame marker and line marker are transmitted transparently to CoaXPress Host.

The following table describes the stream data packet:.

Table 2 • **Stream Data Packet Format**

Word	Content	Description
	4x K27.7	Start of packet indication (see section 8.4).
	4x 0x01	Stream data packet indication.
0	4x Stream ID	Unique stream ID that this packet contains data for.
1	4x Packet Tag	8 bit tag. Incremented for each packet with this stream data words N per packet.
2	4x DsizeP (15:8)	16 bit value representing the number of stream data words N per packet.
3	4x DsizeP (7:0)	
4 to N+3	Stream data	N words of stream data.
N+4	CRC	32-bit CRC calculated over the stream data 4 to (N+3).
	4x K29.7	End of packet indication.

CoaXPress Device IP maps each input image line data packet into one Stream Data Packet.

According to the CoaXPress standard protocol, the image frame marker and line marked should be transmitted properly. To support a flexible frame marker and line marker solution, user needs to generate the complete frame marker and line marker and send it to the CoaXPress Device IP. CoaXPress Device IP would not modify any marker bits or field and transmit it transparently.

There is a FIFO to buffer the input image line packet or marker, and it calculates the input packet length and drops the packet with the wrong length or error condition. When CoaXPress IP is generated, user needs to configure the image line data packet FIFPO words (that is, FIFO depth) and ensure that it buffers at least 2 image line packets.

In stream data packet transmission process, CoaXPress Device IP counts, how many packets are transmitted, the software reads this counter value.

CoaXPress Host IP receives a data stream from CoaXPress Device, it recognizes stream data packet and frame/line markers. To support a flexible frame/line marker solution, user can define the marker header and code, the host IP recognizes the markers according to this setting.

For stream data packets, the CoaXPress Host IP determines if there is an error, the error case includes CRC error, TAG error, and length error. All these error cases can be read by the software. CoaXPress Host IP also counts how many stream data packets have been received. The stream data packets overhead including header and trailer are removed, and CoaXPress Host IP outputs the stream data packet payload to downstream module.

For frame markers and line markers, CoaXPress Host IP does not modify any field or bit and output the complete marker to downstream module.

3.2 Trigger and Trigger Ack

CoaXPress IP supports 2-way directions of trigger and trigger Ack, from host to device, and from device to host.

For CoaXPress Host IP upconnection transmitter and CoaXPress Device IP downconnection transmitter, both provide input ports to accept external trigger transmission requirement signal. For Trigger on high speed channel, TriggerSource fields could be disabled by software configuration.

After receiving a Trigger frame, a Trigger Ack frame should be send immediately. The Trigger receiver displaysa signal to indicate that a Trigger is received with its Delay and TriggerSource fields. The Trigger receiver includes Host downconnection receiver, device low-speed upconnection receiver, and device high-speed upconnection receiver.

For CoaXPress Device IP, high speed downconnection transmitter is responsible for Trigger and Trigger Ack transmission. For CoaXPress Host IP, low speed upconnection transmitter is responsible for Trigger and Trigger Ack transmission when high speed upconnection is not available, else, Trigger and Trigger Ack is transmitted on high speed upconnection.

The following table shows the high-speed Trigger frame format:

Table 3 • **High Speed Trigger Format**

Word	Content	Description
0	Either 4x K28.4	Trigger packet indication - rising edge
	or 4x K28.4	Trigger packet indication - falling edge
1	4x Delay	The delay value is three minus the number of whole characters between the trigger event and the start of this trigger packet. Usage is optional, when it is not set to 0.
2	4x TriggerSource	Trigger Source from 0~15, Optional

The following table shows the low-speed Trigger frame format:

Table 4 • **Low Speed Trigger Format**

Char	Content	Description
0-2	Either K28.2 K28.4 K28.4	Trigger packet indication - rising edge
	or K28.2 K28.2 K28.2	Trigger packet indication - falling edge
3-5	3x Delay	The delay value is 239 minus the number of units of 1/24 the low speed connection bit interval (2.00 ns) between the trigger event and the start of the trigger packet. The host uses a coarser time unit by setting lower bits of this value fixed at 0 or giving it a bias. The Device uses a coarser time unit by using fewer bits of this value (discard non used lower bits of this value or use rounding) The chosen accuracy and usage of this timing correction value is left as a quality of implementation at both the Host and the Device.

The following table shows the Trigger Ack frame format:

Table 5 • **Trigger Ack Format**

Word	Content	Description
0	4x 28.6	I/O acknowledgment packet indication
1	4x Code	Acknowledgment code, repeated 4 times: 0x01 Trigger packet received OK

Trigger and Trigger Ack have the highest priority transmission order. On high speed connection, they can be inserted at any word boundary. On low speed connection, they can be inserted at any character boundary.

After the transmission of a Trigger, the transmitter is waiting for a Trigger Ack in a defined time, which is configured by user. If there is no Trigger Ack in a defined time, then the transmitter might resend a previous Trigger, or send a new trigger when a new trigger event occurs. The maximal resending trigger times is configured by user.

CoaXPress Host and Device count how many Trigger and Trigger Ack have been transmitted or received, the software can read these counters for debugging purpose.

3.3 Control Channel and Event Channel

The control channel provides register access through specific control commands, transmitted from Host to Device, and the resulting acknowledgment messages, transmitted from the Device to the Host.

The Event channel is used to provide the Device with a mechanism to asynchronously send messages and status updates to a Host. This is accomplished through specific event packets, transmitted from Device to Host, and the resulting event acknowledgment packet, transmitted from Host to Device.

The software is responsible to generate control commands/acknowledgments and Event packets/Event Acknowledgments. In each high-speed or low-speed connection transmitter, the CoaXPress Host and Device IP provides a buffer to accept packets from the software configuration, and the packets are transmitted when the configuration is completed, and transmitter is not transmitting other packets.

When received a control command/acknowledgment or Event/Event Acknowledgment, CoaXPress IP writes the received packet into a buffer and allows the software to know a received control packet is available for reading. The software must check, if there is an available packet and reads it out.

In summary, every transmitter has a buffer to accept control packets generated in the software for transmission, and every receiver has a buffer to storage received control packets and give instructions to the software to read the received packet out.

The CoaXPress Host IP and Device IP only send and receive packets for Control Channel and Event Channel, the software is responsible for the following items:

- Generate control commands or Event Acknowledgments in CoaXPress Host.
- Generate control acknowledgment or Event packets in CoaXPress Device.
- Checking if the received control packet is error free.
- Determine if the acknowledgment is timeout and resend the control command if necessary.

Refer to section “Writing and Reading Control or Event Packet” to get the details of reading and writing control or Event packets.

3.4 Connection Test

The CoaXPress Host and Device IP provides connection test to test the quality of the connection. For each connection, the transmitter includes a test frame generator, and the receiver includes a test frame receiver.

The following table shows the test frame format:

Table 6 • **Test Frame Format**

	Content				
Word	**P0**	**P1**	**P2**	**P3**	**Description**
	K27.7	K27.7	K27.7	K27.7	Start of packet indication (see section 8.4).
	0x04	0x04	0x04	0x04	Connection test indication.
0	0x00	0x01	0x02	0x03	Test data
1	0x04	0x05	0x06	0x07	
...					
63	0xFC	0xFD	0xFE	0xFF	
64	0X00	0X01	0X02	0X03	
...					
1023	0XFC	0XFD	0XFE	0XFF	
	K29.7	K29.7	K29.7	K29.7	End of packet indication.

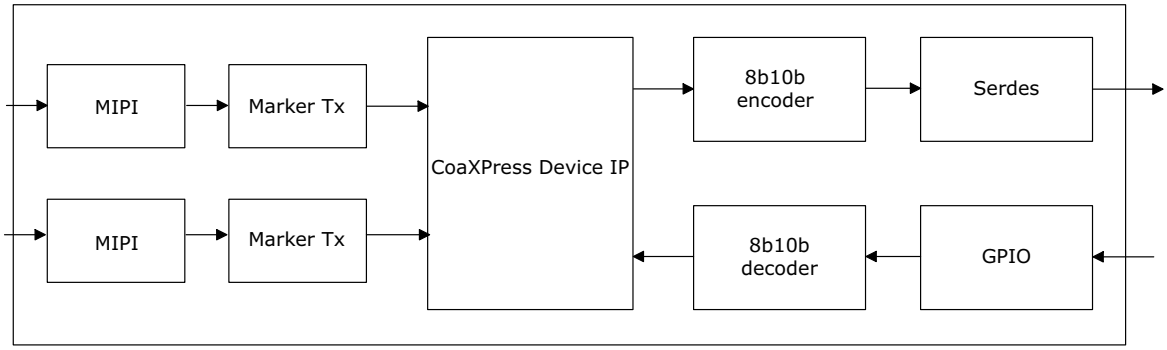
Prior to starting CoaXPress communication between Host and Device, user can start run connection test for each connection channel. There is a register control test frame transmission for each connection transmitter. Connection transmitter counts how many test frames are sent, and connection receiver counts how many test frames are received.

4 Typical Application

4.1 Device IP Application

The following diagram describes a typical CoaXPress Device IP application with 2 MIPI input interface:

Figure 4 • **Typical Application for CoaXPress Device IP**



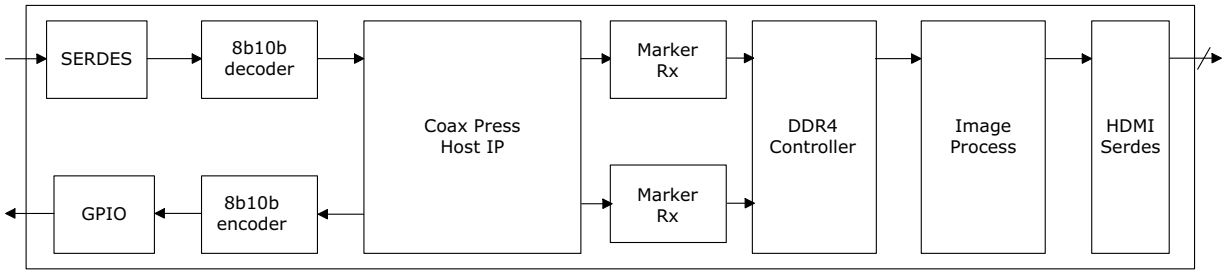
As described in the preceding diagram, the MIPI interface gets image frames and lines from an external camera, Marker Tx module generates frame marker at the starting of each frame and generates line marker at the starting of each line, it also forwards the line data to CoaXPress Device IP. CoaXPress Device IP transmit the line data packet in stream data packet with frame/line marker on downconnection. The downconnection data is encoded to 10B data and transmitted on Serdes. On upconnection, the data from 41.6 Mbps GPIO is decoded to 8B data in 8B10B decoder, CoaXPress Device IP gets 8B upconnection data and recognizes the packets on upconnection.

If quick trigger and control channel are required, the high-speed upconnection should be added. In this case, upconnection packets are transmitted on high-speed upconnection when it is established.

4.2 Host IP Application

The following diagram describes a typical CoaXPress Host IP application with 2 Stream IDs:

Figure 5 • **Typical Application for CoaXPress Host IP**



As described in the preceding diagram, this application gets data packets from CoaXPress cable, and it sends data to the HDMI interface.

Serdes gets downconnection data from the CoaXPress device and decodes it to 8B data. CoaXPress Host IP processes all the packets in downconnection data stream, the frame/line marker and image line data packets are extracted from the stream data packet, the Marker Rx module forwards image line data packets to DDR4. DDR4 reading data is sending to the HDMI interface after image signal processing.

On the Host upconnection direction, the CoaXPress Host IP generates Trigger, Trigger Ack, and Control Command packets, upconnection data stream is coded to 10B code in 8B10B encoder. The 10B data is transmitted on a GPIO port to CoaXPress Device.

If a quick trigger and control channel is required, the high-speed upconnection should be added. In this case, upconnection packets are transmitted on high-speed upconnection when it is established.

5 CoaXPress Host IP Interface Singles

5.1 Interface

CoaXPress IP includes Host IP and Device IP. The following table shows the input and output ports for CoaXPress Host IP:

Table 7 • **CoaXPress Host IP Interface Signals**

Interface	Direction	Description
tx_rst_n_i	Input	Low active reset signal with tx_clk_i synchronization
tx_clk_i	Input	Host transmitter clock
rx_rst_n_i	Input	Low active reset signal with rx_clk_i synchronization
rx_clk_i	Input	Host receiver clock
lsuc_clk_i	Input	125M clock for Low speed up connection transmitter
lsuc_rstn_i	Input	Low active reset signal with lsuc_clk_i synchronization
addr_i	Input	Reading and writing internal registers interface: address sync with tx_clk_i
cs_i	Input	Reading and writing internal registers interface: select enable, high active
wen_i	Input	Reading and writing internal registers interface: write enable, high active
ren_i	Input	Reading and writing internal registers interface: read enable, high active
wdata_i	Input	Reading and writing internal registers interface: writing data
rdata_o	Output	Reading and writing internal registers interface: reading data sync with tx_clk_i
rdval_o	Output	Indicates if data on rdata_o port is active or available.
lsuc_req_tx_r_trigger_i	Input	Require sending rising trigger in low-speed upconnection transmitter (sync with lsuc_clk_i)
lsuc_req_tx_f_trigger_i	Input	Require sending falling trigger in low-speed upconnection transmitter (sync with lsuc_clk_i)
hsuc_req_tx_r_trigger_i	Input	Require sending rising trigger in high-speed upconnection transmitter (sync with tx_clk_i)
hsuc_req_tx_f_trigger_oi	Input	Require sending falling trigger in low-speed upconnection transmitter (sync with tx_clk_i)
hsuc_trigger_src_i	Input	TriggerSource field for high speed trigger (sync with tx_clk_i)
lsuc0_tx_dval_o	Output	Indicates, if Low-Speed upconnection data is available (sync with lsuc_clk_i).
lsuc0_tx_data_o	Output	Low-Speed upconnection data (sync with lsuc_clk_i).
lsuc0_tx_k_o	Output	Low-Speed upconnection data K character indication (sync with lsuc_clk_i).
hsuc_tx_k_o	Output	High-Speed upconnection data K character indication (sync with tx_clk_i).
hsuc_tx_data_o	Output	High-Speed upconnection data (sync with tx_clk_i)
hsdc0_rx_k_i	Input	Downconnection K character indication (sync with rx_clk_i)
hsdc0_rx_data_i	Input	Downconnection data (sync with rx_clk_i)

Table 7 • **CoaXPress Host IP Interface Signals**

Interface	Direction	Description
hsdc0_rx_f_trigger_ind_o	Output	High Speed downconnection received a falling trigger (sync with rx_clk_i)
hsdc0_rx_r_trigger_ind_o	Output	High speed downconnection received a falling trigger (sync with rx_clk_i)
hsdc0_rx_trigger_src_o	Output	TriggerSource field in received rising or falling trigger in high speed downconntion receiver (sync with rx_clk_i)
hsdc0_rx_trigger_delay_o	Output	Delay field in received rising or falling trigger in high speed downconnection receiver (sync with rx_clk_i)
hsdc0_rx_heartbeat_ind_o	Output	High speed downconnection received a heartbeat packet (sync with rx_clk_i)
hsdc0_rx_heartbeat_id_o	Output	MasterHostConnection value in received heartbeat (sync with rx_clk_i)
hsdc0_rx_heartbeat_timestamp_o	Output	TimeStamp in received heartbeat (sync with rx_clk_i)
frame_marker_ind_o	Output	Indicates if output ipkt_data_o is frame marker (sync with rx_clk_i) If there are 2 SDP demapper, frame_marker_ind_o [0] is for demapper0 and frame_marker_ind_o [1] is for demapper1
line_marker_ind_o	Output	Indicates if output ipkt_data_o is line marker (sync with rx_clk_i) If there are 2 SDP demapper, line_marker_ind_o [0] is for demapper0 and line_marker_ind_o [1] is for demapper1
ipkt_data_val_o	Output	Indicates, if data on ipkt_data_o is available (sync with rx_clk_i). If there are 2 SDP demapper, ipkt_data_val_o[0] is for demapper0 and ipkt_data_val_o[1] is for demapper1
ipkt_data_sop_o	Output	Starting of image line packet or marker indication (sync with rx_clk_i) If there are 2 SDP demapper, ipkt_data_sop_o[0] is for demapper0 and ipkt_data_sop_o[1] is for demapper1.
ipkt_data_eop_o	Output	End of image line packet or marker indication (sync with rx_clk_i) If there are 2 SDP demapper, ipkt_data_eop_o[0] is for demapper0 and ipkt_data_eop_o[1] is for demapper1.
ipkt_data_o	Output	Image line packet or marker data (sync with rx_clk_i) If there are 2 SDP demapper, ipkt_data_o[31:0] are for demapper0 and ipkt_data_val_o[63:32] are for demapper1.
ipkt_data_k_o	Output	Image line packet or marker data K character indication (sync with rx_clk_i) If there are 2 SDP demapper, ipkt_data_k_o[3:0] is for demapper0 and ipkt_data_k_o[7:4] is for demapper1.

5.2 Configuration Parameters

The following table describes the configuration parameters for CoaXPress Host IP:

Table 8 • **Configuration Parameters for CoaXPress Host IP**

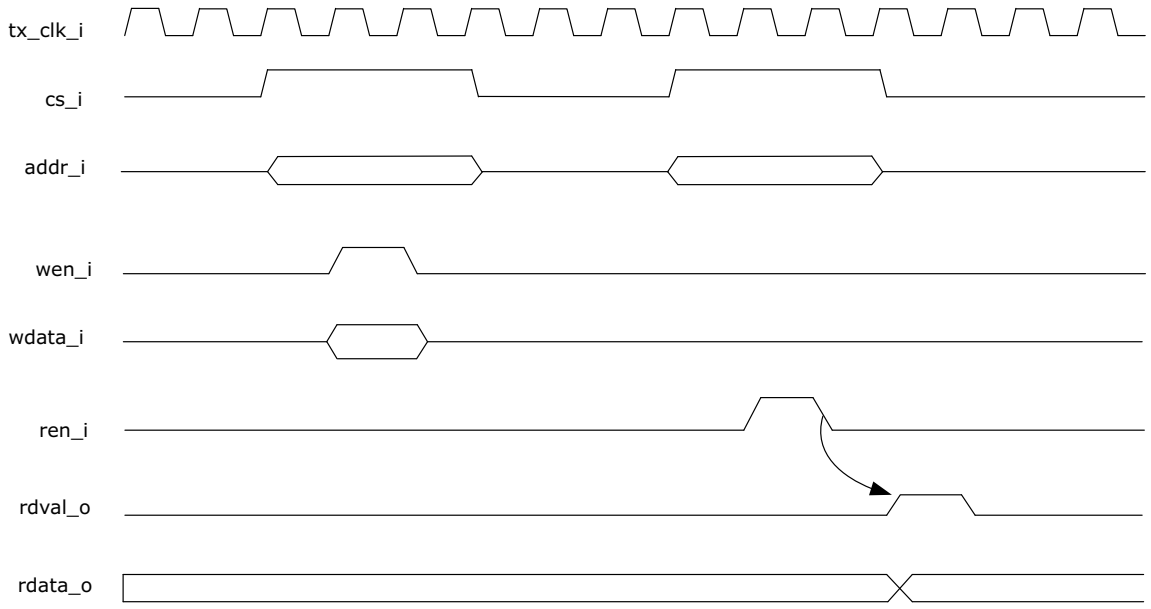
Name	Default	Description
g_SDP_PACKET_FIFO_WORDS	1024	It defines how many words (4 bytes) for FIFO in SDP demapper, it should be able to buffer at least 2 longest stream data packets.
g_CPU_PACKET_FIFO_WORDS	256	It defines how many words (4 bytes) for FIFO to buffer packets from CPU configuration, or the FIFO buffer received packets for CPU reading.
g_IMAGE_STREAM_NUM	1	SDP Demapper number (up to 2)
g_HIGH_SPPED_UPCONNECTION_EN	0	High Speed Upconnection enable or disable
g_IMAGE_STREAM_DATA_WID	32	SDP Demapper output video data bus width, it could be 32 bits or 64bit. When g_IMAGE_STREAM_NUM is 2, it must be set to 32. When g_IMAGE_STREAM_NUM is 1, it could be set to 32 or 64.

5.3 Key Interface Description

5.3.1 Reading and Writing Register Interface

The following figure shows the timing diagram of the Reading and Writing Internal Register Interface:

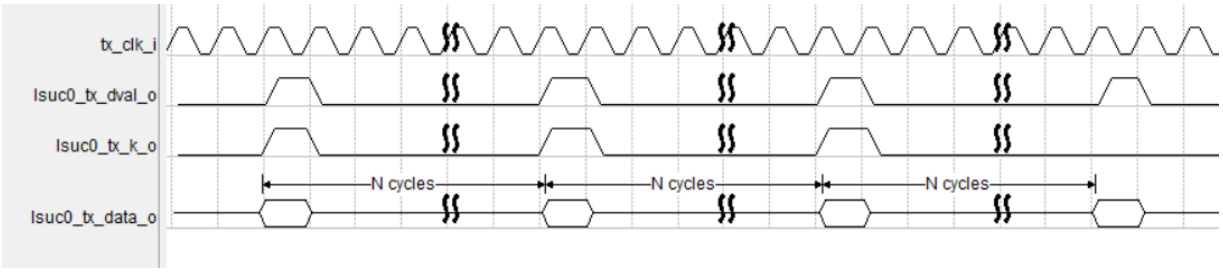
Figure 6 • **Timing diagram for Reading and Writing Host IP Registers Interface**



5.3.2 Low-Speed Upconnection Transmitter Interface

The following figure shows the timing diagram of Low Speed Upconnection Transmitter Interface:

Figure 7 • **Timing Diagram for Low Speed Upconnection Transmitter Interface**

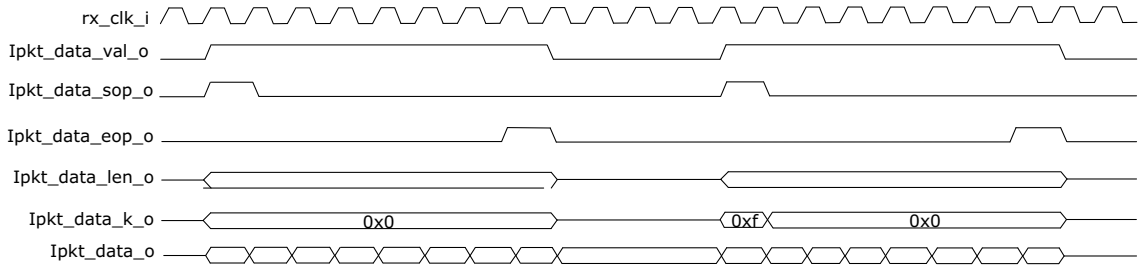


Low speed upconnection transmitter outputs 1 valid data every 480ns.

5.3.3 Image packet output Interface

The following figure shows the timing diagram of image packet output interface:

Figure 8 • **Timing Diagram for Low Speed Upconnection Transmitter Interface**



This interface outputs, image line packet and markers packets including frame marker and line marker. User needs to design a downstream module to receive marker packets.

6 CoaXPress Device IP Interface Singles

6.1 Interface

The following table shows the input and output ports for CoaXPress Device IP:.

Table 9 • **CoaXPress Device IP Interface Signals**

Interface	Direction	Description
tx_rst_n_i	Input	Low active reset signal with tx_clk_i synchronization
tx_clk_i	Input	Device transmitter clock
rx_rst_n_i	Input	Low active reset signal with rx_clk_i synchronization
rx_clk_i	Input	Device receiver clock
lsuc_rst_n_i	Input	Low active reset signal with tx_clk_i synchronization
lsuc_clk_i	Input	125M low-speed upconnection receiver clock
addr_i	Input	Reading and writing internal registers interface: address sync with tx_clk_i
cs_i	Input	Reading and writing internal registers interface: select enable, high active
wen_i	Input	Reading and writing internal registers interface: write enable, high active
ren_i	Input	Reading and writing internal registers interface: read enable, high active
wdata_i	Input	Reading and writing internal registers interface: writing data sync with tx_clk_i
rdata_o	Output	Reading and writing internal registers interface: reading data sync with tx_clk_i.
rdval_o	Output	Indicates if data on rdata_o port is active or available
hsdc0_req_tx_r_trigger_i	Input	Require sending rising trigger (sync with tx_clk_i)
hsdc0_req_tx_f_trigger_i	Input	Require sending falling trigger (sync with tx_clk_i)
hsdc0_trigger_src_i	Input	TriggerSource for sending rising or falling trigger (sync with tx_clk_i).
hsdc0_req_tx_heartbeat_i	Input	Require sending heartbeat (sync with tx_clk_i)
hsdc0_heartbeat_id_i	Input	MasterHostConnectionID for sending heartbeat (sync with tx_clk_i)
hsdc0_heartbeat_time_i	Input	TimeStamp for sending heartbeat (sync with tx_clk_i)
lsuc0_rx_f_trigger_ind_o	Output	Low speed upconnection received falling trigger (sync with lsuc_clk_i)
lsuc0_rx_f_trigger_delay_o	Output	Delay field in low speed upconnection received falling trigger (sync with lsuc_clk_i)
lsuc_rx_r_trigger_ind_o	Output	Low speed upconnection received rising trigger (sync with lsuc_clk_i)
lsuc_rx_r_trigger_delay_o	Output	Delay field in low speed upconnection received rising trigger (sync with lsuc_clk_i)
hsuc_rx_f_trigger_ind_o	Output	High speed upconnection received falling trigger (sync with rx_clk_i)
hsuc_rx_f_trigger_delay_o	Output	Delay field in high speed upconnection received falling trigger (sync with rx_clk_i)
hsuc_rx_r_trigger_ind_o	Output	High speed upconnection received rising trigger (sync with rx_clk_i)
hsuc_rx_r_trigger_delay_o	Output	Delay field in high speed upconnection received rising trigger (sync with rx_clk_i)
lsuc0_rx_dval_i	Input	Indicates if data on lsuc0_rx_data_i is available (sync with lsuc_clk_i)

Table 9 • **CoaXPress Device IP Interface Signals**

Interface	Direction	Description
lsuc0_rx_k_i	Input	Low-Speed upconnection received data K character indication (sync with lsuc_clk_i)
lsuc0_rx_data_i	Input	Low-Speed upconnection received data (sync with lsuc_clk_i)
hsuc_rx_k_i	Input	High speed upconnection received data K character indication (sync with rx_clk_i)
hsuc_rx_data_i	Input	High speed upconnection received data (sync with rx_clk_i)
ipkt_val_i	Input	Indicates if data on ipkt_data_i is available or active (sync with tx_clk_i) If there are 2 SDP mappers, the first half bits (bit[1]) is for SDP mapper1, the second half bits (bit[0]) is for SDP mapper0, the same rule applies to all ipkt_xx_i input signals.
ipkt_sop_i	Input	Indicates the starting of input image line packet or marker (sync with tx_clk_i).
ipkt_eop_i	Input	Indicates the end of input image line packet or marker (sync with tx_clk_i).
ipkt_type_i	Input	Indicates if the packet is frame marker or line marker or line data (sync with tx_clk_i) 3'b001), it is line data packet 3'b010), it is line marker 3'b100), it is frame marker
ipkt_k_i	Input	Indicates the ipkt_data_i K character (sync with tx_clk_i).
ipkt_data_i	Input	Input image line data or marker data (sync with tx_clk_i)
ipkt0_fifo_alfull_o	Output	Indicates the image line packet FIFO is almost full, the previous module should pause sending data to avoid FIFO overflow (sync with tx_clk_i).
hsdc0_tx_k_o	Output	Indicates K character on hsdc0_tx_data_o port (sync with tx_clk_i)
hsdc0_tx_data_o	Output	Downconnection transmission data (sync with tx_clk_i)

6.2 Configuration Parameters

The following table describes the configuration parameter for CoaXPress Device IP:

Table 10 • **Configuration Parameters for CoaXPress Device IP**

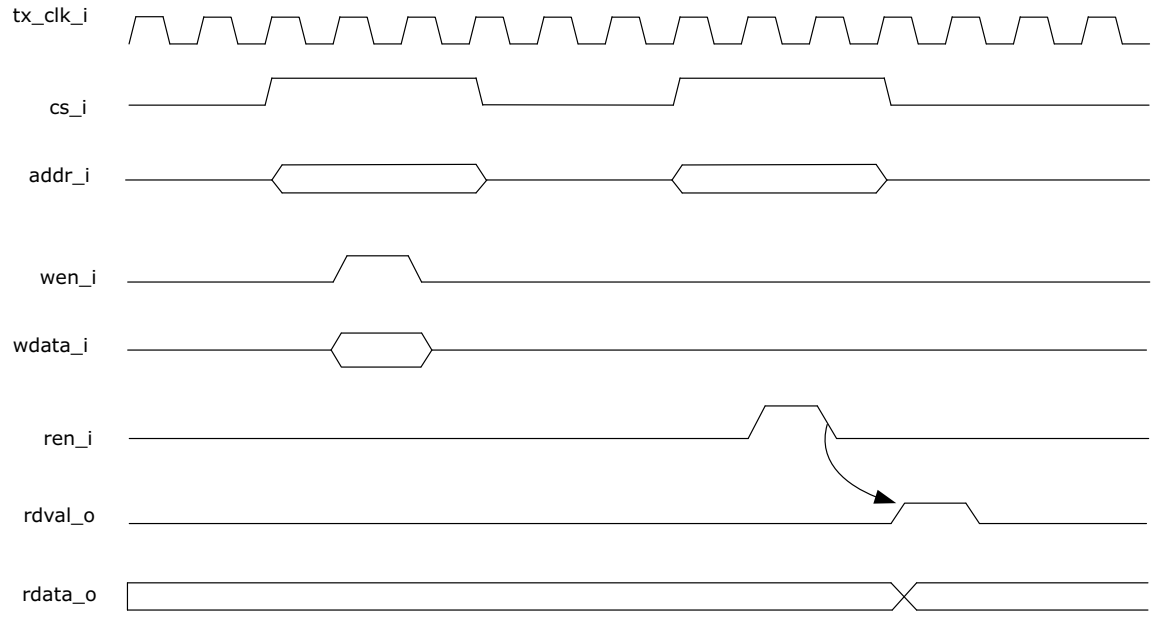
Name	Default	Description
g_SDP_PACKET_FIFO_WORDS	1024	It defines how many words (4 bytes) for FIFO in SDP demapper, it should be able to buffer at least 2 longest stream data packets.
g_CPU_PACKET_FIFO_WORDS	256	It defines how many words (4 bytes) for FIFO to buffer packets from CPU configuration or the FIFO buffer received packets for CPU reading.
g_IMAGE_STREAM_NUM	1	SDP Demapper number (up to 2)
g_IMAGE_STREAM_DATA_WID	32	SDP mapper input video data bus width, it could be 32 bits or 64bit. When g_IMAGE_STREAM_NUM is 2, it must be set to 32. When g_IMAGE_STREAM_NUM is 1, it could be set to 32 or 64.
g_HIGH_SPPED_UPCONNECTION_EN	0	High Speed Upconnection enable or disable

6.3 Key Interface Description

6.3.1 Reading and Writing Register Interface

The following figure shows the timing diagram of Reading and Writing Internal Register Interface:

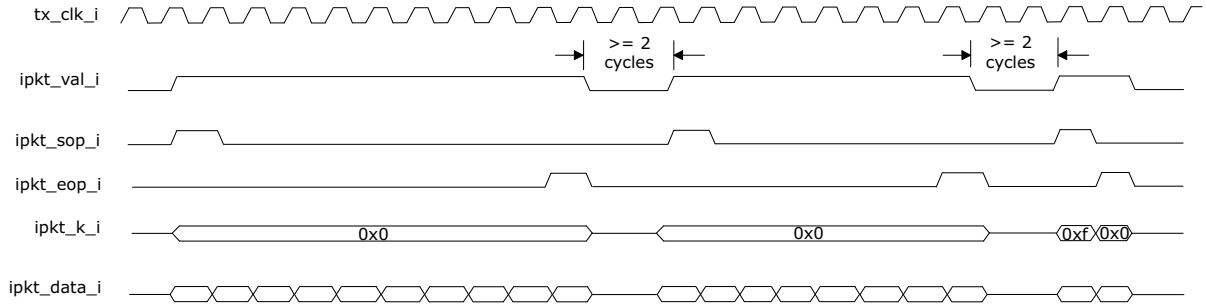
Figure 9 • **Timing Diagram for Reading and Writing Device IP Registers Interface**



6.3.2 Image packet and Marker Input Interface

The following figure shows the timing diagram of Image Packet Output Interface:

Figure 10 • **Timing Diagram for Image Packet or Marker Input Interface**



This image packer and marker input interface accepts image line packet and markers including frame marker and line marker. There should be at least two idle clock cycles between any two input packets.

User needs to design an upstream module to generate a complete marker packet and send markers to CoaXPress Device IP. The port ipkt_k_i should only be no-zero for marker packets, it should be constantly zero for image line packets.

7 CoaXPress Host IP Configuration Guide

7.1 Connection Test

To test high speed upconnection, software writes register 0x0059 bit [0] to 1, then read register 0x0061 to know how many test frames have been transmitted. To stop high speed upconnection test function, software writes the register 0x0059 bit [0] to 0.

To test low speed upconnection, software writes register 0x004A bit [0] to 1, then read register 0x0044 to know how many test frames have been transmitted. To stop low speed upconnection test function, software writes the register the address is 0x004A bit [0] to 0

CoaXPress Host IP downconnection receiver always counts how many Test frames have been received, the software should read register 0x0030 to know it, and read register 0x0031 to know how many error test frames have been received.

7.2 Receive Stream Data Packet

To receive stream data packet correctly, user need to configure CoaXPress Host IP items as follows:

- The frame marker header word and code word (the first and second word of frame marker).
- The length of frame marker.
- The line marker header word and code word (the first and second word of line marker).
- The length of line marker.
- The stream ID for each SDP demapper (Stream Data Packet demapper).

By default, the frame marker and line marker configured in CoaXPress IP are the Rectangular Image Stream marker defined in CoaXPress V2.0 Section 10.4.6. If different marker is used in the application, user should configure the marker as follows:

- Set register 0x0038 to frame marker first word 8B code.
- Set register 0x0039 to frame marker second word.
- Set register 0x003C to frame marker length in unit of 64bit. For example, if the frame marker is 25 words or 26 words, set register 0x003C to 0x0D. If the frame marker is 23 or 24 words, set register 0x003C to 0x0C.
- Set register 0x003A to line marker first word 8B code.
- Set register 0x003B to line marker second word.
- Set register 0x003D to line marker length in unit of 64bit. For example, if the line marker is 3 words or 4 words, set register 0x003D to 0x02. If the line marker is 2 words, set register 0x003D to 0x01.

7.3 Enable UpConnection Transmitter

To use high speed upconnection, user need to enable it when generates the CoaXPress IP.

By default, both low speed and high speed upconnection transmitter can transmit test frame, control messages and Event Ack messages through software configuration. However, for Trigger and Trigger Ack transmission, user need to enable the low-speed or high-speed transmitter first. To enable low-speed transmitter, user need to write register 0x004F bit [0] to 1. To enable high-speed transmitter, user need to write register 0x006A bit [0] to 1.

The low speed upconnection transmitter supports 20.83Mbps or 41.6Mbps. By default, after power up, it supports 41.6Mbps, to switch to 20.83Mbps, user need to write register 0x004F bit [16] to 1.

7.4 Software Sending Packets

In high speed upconnection transmitter and low speed upconnection transmitter, there is a FIFO to buffer packets from the software. When the whole packet writing is completed, upconnection transmitter starts to transmit the packets in buffer.

Software can write any packet content into the buffer, but it needs to observe the following rules:

1. The first word must be 0xfbfbfbfb which is frame header word defined in CoaXPress.
2. The last word must be 0xfdfdfdfd which is frame trailer word defined in CoaXPress.
3. The minimal writing packet length is 5 words.

For high speed upconnection transmitter, software should write 64 bits packet data into buffer for each writing operation. The writing steps for a single writing operation are as follows:

1. Writes the writing data bit [63:32] into register 0x0052.
2. Writes the writing data bit [31:0] into register 0x0051.
3. To write all 64bit data info buffer, set register 0x0050 bit [2] to 1.
4. To only write bit [63:32] into buffer, set register 0x0050 bit [2] to 0.

When set register 0x0050 bit [1] to [1], otherwise, set it to 0.

When writing the last word, set register 0x0050 bit [0] to 1 otherwise, set it to 0.

when all above settings are done, set register 0x0050 bit [8] to 1, the rising edge of this value writes writing data into buffer.

set register 0x0050 bit [8] to 0.

For example, the complete packet is: 0xfbfbfbfb, 0x03030303, 0x00010203, 0x04050607, 0x08090a0b, 0x0c0d0e0f, 0xfdfdfdfd, software should execute the following writing procedure the write this packet into high speed upconnection transmitter buffer:

1. Set register 0x0052 to 0xfbfbfbfb, set register 0x0051 to 0x03030303, set register 0x0050 to 0x00000006. Then set register 0x0050 to 0x00000106. The first 2 words are writing done.
2. Set register 0x0052 to 0x00010203, set register 0x0051 to 0x04050607, set register 0x0050 to 0x00000004. Then Set register 0x0050 to 0x00000104. The 3rd and 4th words are writing done.
3. Set register 0x0052 to 0x08090a0b, set register 0x0051 to 0x0c0d0e0f, set register 0x0050 to 0x00000004. Then Set register 0x0050 to 0x00000104. The 5th and 6th words are writing done.
4. Set register 0x0052 to 0xfdfdfdfd, set register 0x0050 to 0x00000001. Then Set register 0x0050 to 0x00000101. The 7th word is writing done.

For low speed upconnection transmitter, software should write 32 bits packet data into buffer for each writing operation. The writing steps for a single writing operation are as follows:

1. Writes the writing data bit [31:0] into register 0x0048.
2. When writing the first word, set register 0x0047 bit [1] to 1, otherwise, set it to 0.
3. When writing the last word, set register 0x0047 bit [0] to 1, otherwise, set it to 0.
4. When all above settings are done, set register 0x0049 bit [0] to 1, the rising edge of this value writes writing data into buffer.
5. Set register 0x0049 bit [0.] to 0

For example, the complete packet is: 0xfbfbfbfb, 0x03030303, 0x00010203, 0x04050607, 0x08090a0b, 0xfdfdfdfd, software should execute the following writing procedure the write this packet into high speed upconnection transmitter buffer:

1. Set register 0x0048 to 0xfbfbfbfb, set register 0x0047 to 0x00000002. Set register 0x0049 to 0x00000001, then set register 0x0049 to 0x00000000. The first word is writing done.
2. Set register 0x0048 to 0x03030303, set register 0x0047 to 0x00000000. Set register 0x0049 to 0x00000001, then set register 0x0049 to 0x00000000. The 2nd word is writing done.
3. Set register 0x0048 to 0x00010203, set register 0x0047 to 0x00000000. Set register 0x0049 to 0x00000001, then set register 0x0049 to 0x00000000. The 3rd word is writing done.
4. Set register 0x0048 to 0x04050607, set register 0x0047 to 0x00000000. Set register 0x0049 to 0x00000001, then set register 0x0049 to 0x00000000. The 4th word is writing done.
5. Set register 0x0048 to 0x08090a0b, set register 0x0047 to 0x00000000. Set register 0x0049 to 0x00000001, then set register 0x0049 to 0x00000000. The 5th word is writing done.
6. Set register 0x0048 to 0xfdfdfdfd, set register 0x0047 to 0x00000001. Set register 0x0049 to 0x00000001, then set register 0x0049 to 0x00000000. The 6th word is writing done.

7.5 Software Reading Received Packets

In CoaXPress Host downconnection receiver, there is a FIFO to buffer received Command Acknowledgment packets or Event packets from CoaXPress Device. When received a complete packet, the software reads the received packet.

To execute one reading operation in software, the following steps should be observed:

1. Read register 0x0011 bit [0], if the reading value is 1, it means there are data to be read, software should execute the following reading operation. Otherwise, no data to be read, and software should stop reading and waiting for some time to check it again.
2. Set register 0x0012 bit [0] to 1, then set register 0x0012 bit [0] to 0, the rising edge of this value execute 1 reading operation.
3. Read register 0x0015 to get reading data bit [63:32], then read register 0x0014 to get reading data bit [31:0].
4. Read register 0x0013 bit [0], if the reading value is 1, it means the reading data indicates how many words in the reading packet. Otherwise, the reading data is packet data.
5. Read register 0x0013 bit [1], if the reading value is 1, it means all reading data bit [63:0] is available. Otherwise, only bit [63:32] is available, bit [31:0] should be ignored.
6. Read register 0x0013 bit [2], if the reading value is 1, it means the reading data is the last word of the packet.
7. Read register 0x0013 bit [3], if the reading value is 1, it means the reading data is the first word of the packet.

Software should execute step 1 periodically to check if there is any received packet in buffer, and read the packet out by the following steps there until they is no data to be read.

7.6 Register Definition

The following table shows the internal registers defined in CoaXPress Host IP:

Table 11 • **CoaXPress Host IP Registers**

Addr	Bits	Name	Type	Default	Description
0x0011	[0]	hsdc_sw_rx_pkt_val	RO	0x0	Indicates if there is received packet data to be software reading
0x0012	[0]	hsdc_sw_rx_pkt_ren	RW	0x0	The rising edge of this bit reads 64 bits data
0x0013	[3]	hsdc_sw_rx_pkt_sop	RO	0x0	Indicates if the reading data is the packet header
	[2]	hsdc_sw_rx_pkt_eop	RO	0x0	Indicates if the reading data is the packet end word
	[1]	hsdc_sw_rx_pkt_mod	RO	0x0	1 means all 64bit reading data is available, 0 means only reading data bit [63:32] is available.
	[0]	hsdc_sw_rx_pkt_len	RO	0x0	1 means the reading data is packet length (in unit of words or 4 bytes), 0 means the reading data is packet data.
0x0014	[31:0]	hsdc_sw_rx_data0	RO	0x0	Reading data bit [63:32] from high-speed downconnection packet buffer
0x0015	[31:0]	hsdc_sw_rx_data1	RO	0x00000000	Reading data bit [31:0] from high-speed downconnection packet buffer.
0x0020	[0]	hsdc_trigger_src_en	RW	0x1	Enable TriggerSource field in high speed Trigger packet.
0x0030	[31:0]	hsdc_rx_test_num	RC	0x00000000	High speed downconnection receiver received test frame number.
0x0031	[31:0]	hsdc_rx_err_test_num	RC	0x00000000	High speed downconnection receiver received error test frame number.
0x0038	[31:0]	frame_marker_header	RW	0x7c7c7c7c	Frame marker first word 8B code
0x0039	[31:0]	frame_marker_code	RW	0x01010101	Frame marker second word
0x003A	[31:0]	line_marker_header	RW	0x7c7c7c7c	Enable sending test frame on low speed upconnection
0x003B	[31:0]	line_marker_code	RW	0x02020202	Line marker second word
0x003C	[7:0]	frame_marker_len	RW	0x0D	Frame marker length in unit of 2 words (64 bits)
0x003D	[7:0]	line_marker_len	RW	0x01	Line marker length in unit of 2 words (64bits)
0x0040	[15:8]	lsuc_ack_timeout_th	RW	0x1	Defines how many low speed char time is waiting Ack timeout time
	[3:0]	lsuc_send_trigger_times	RW	0x3	Defines the maximal sending times for sending the same trigger event
0x0044	[31:0]	lsuc_tx_test_num	RC	0x00000000	How many test frame has been send in low-speed upconnection transmitter

Table 11 • **CoaXPress Host IP Registers**

Addr	Bits	Name	Type	Default	Description
0x0045	[15:0]	lsuc_tx_sw_pkt_num	RC	0x0000	How many software packets has been send in low-speed upconnection transmitter
0x0047	[1]	lsuc_tx_sw_pkt_sop	RW	0x0	Low speed transmitter. software packet writing SOP
	[0]	lsuc_tx_sw_pkt_eop	RW	0x0	Low speed transmitter software packet writing EOP
0x0048	[31:0]	lsuc_tx_sw_pkt_data	RW	0x00000000	Low speed transmitter software packet writing data
0x0049	[0]	lsuc_tx_sw_pkt_wen	RW	0x0	Low speed transmitter software packet writing enable
0x004A	[0]	lsuc_tx_test_frame_en	RW	0x0	Enable sending test frame in low speed transmitter
0x004F	[0]	lsuc_enable	RW	0x1	Enable low speed transmitter to send Trigger and Trigger Ack
	[16]	lsuc_speed_mode	RW	0x0	0 means 41.6Mbps, 1 means 20.83Mbps
0x0050	[8]	hsuc_tx_sw_pkt_wen	RW	0x0	High speed transmitter, 1 means software packet data writing enable. signal
	[2]	hsuc_tx_sw_pkt_mod	RW	0x0	High speed transmitter software packet data mode
	[1]	hsuc_tx_sw_pkt_sop	RW	0x0	High speed transmitter software packet data SOP indication
	[0]	hsuc_tx_sw_pkt_eop	RW	0x0	High speed transmitter software packet data EOP indication
0x0051	[31:0]	hsuc_tx_sw_pkt_data0	RW	0x00000000	High speed transmitter software packet writing data bit [31:0]
0x0052	[31:0]	hsuc_tx_sw_pkt_data1	RW	0x00000000	High speed transmitter software packet writing data bit [63:32]
0x0055	[15:0]	hsuc_ack_timeout_time	RW	0x1000	Define how many clock cycles is waiting Ack timeout time
0x0056	[3:0]	hsuc_send_trigger_times	RW	0x3	Defines the maximal sending times for sending the same trigger event
0x0057	[16]	hsuc_trigger_src_en	RW	0x1	Enable high-speed transmitter TriggerSource field when sending Trigger
0x0059	[0]	hsuc_test_frame_en	RW	0X0	Enable sending test frame in high speed transmitter
0x0061	[31:0]	hsuc_tx_test_frame_num	RC	0X00000000	How many test frames have been send in high speed transmitter
0x0063	[15:0]	hsuc_tx_sw_pkt_num	RC	0x0000	How many software packets have been send
0x006A	[0]	hsuc_enable	RW	0x0	Enable high speed transmitter to send Trigger and Trigger Ack

Table 11 • **CoaXPress Host IP Registers**

Addr	Bits	Name	Type	Default	Description
0x0081	[7:0]	sdp_demapper0_sid	RW	0x0	SDP demapper0 stream ID
0x0082	[6]	sdp_demapper0_sdp_err	RC	0x0	Found SDP packet error in demapper0
	[5]	sdp_demapper0_tag_err	RC	0x0	Found SDP Tag error in demapper0
	[4]	sdp_demapper0_crc_err	RC	0x0	Found SDP CRC error in demapper0
0x0085	[7:0]	sdp_demapper1_sid	RW	0x1	SDP demapper1 stream ID
0x0086	[6]	sdp_demapper1_sdp_err	RC	0x0	Found SDP packet error in demapper1
	[5]	sdp_demapper1_tag_err	RC	0x0	Found SDP Tag error in demapper1
	[4]	sdp_demapper1_crc_err	RC	0x0	Found SDP CRC error in demapper1

8 CoaXPress Device IP Configuration Guide

8.1 Connection Test

To test downconnection, the software must set register 0x0019 bit [0] to 1, then read register 0x0021 to know how many test frames have been transmitted. To stop downconnection test function, the software must set register0x0019 bit [0] to 0.

CoaXPress Device IP upconnection receiver always counts how many Test frames have been received. For high speed upconnection, the software must read register 0x0060 to know it and read register 0x0061 to know how many error test frames have been received. For low speed upconnection, the software must read register 0x0080 to know it, and read register 0x0081 to know how many error test frames have been received.

8.2 Transmit Stream Data Packet

To transmit stream data packet correctly, user need to configure CoaXpress Host IP items as follows:

- The length of line marker.
- The stream ID for each SDP mapper (Stream Data Packet demapper).

By default, the frame marker and line marker configured in CoaXPress IP are the Rectangular Image Stream marker defined in CoaXPress V2.0 Section 10.4.6. If different marker is used in the application, user should configure the marker as follows:

- Set register 0x0000 bit [23:16] to line marker length in unit of 64bit.
- Set register 0x0002 bit [23:16] to line marker length in unit of 64bit. For example, if the line marker is 3 words, 4 words, set register 0x003D to 0x02. If the line marker is 2 words, set register 0x0002 to 0x01.

8.3 Software Sending Packets

In high speed downconnection transmitter, there is a FIFO to buffer packets from the software. When the whole packet writing is completed, downconnection transmitter starts to transmit the packets in buffer.

Software can write any packet content into the buffer, but it needs to observe the following rules:

1. The first word must be 0xfbfbfbfb which is frame header word defined in CoaXPress.
2. The last word must be 0xfdfdfdfd which is frame trailer word defined in CoaXPress.
3. The minimal writing packet length is 5 words.

For high speed downconnection transmitter, software should write 64 bits packet data into buffer for each writing operation. The writing steps for a single writing operation are as follows:

1. Writes the writing data bit [63:32] into register 0x0012.
2. Writes the writing data bit [31:0] into register 0x0011.
3. To write all 64bit data info buffer, set register 0x0010 bit [2] to 1.
4. To only write bit [63:32] into buffer, set register 0x0010 bit [2] to 0.
5. When writing the first word, set register 0x0010 bit [1] to 1, otherwise, set it to 0.
6. When writing the last word, set register 0x0010 bit [0] to 1, otherwise, set it to 0.
7. When all above settings are done, set register 0x0010 bit [8] to 1, the rising edge of this value writes writing data into buffer.
8. Set register 0x0010 bit [8] to 0

For example, the complete packet is: 0xfbfbfbfb, 0x03030303, 0x00010203, 0x04050607, 0x08090a0b, 0x0c0d0e0f, 0xfdfdfdfd, software should execute the following writing procedure the write this packet into high speed upconnection transmitter buffer:

1. Set register 0x0012 to 0xfbfbfbfb, set register 0x0011 to 0x03030303, set register 0x0010 to 0x00000006, then set register 0x0010 to 0x00000106. The first 2 words writing are done.

   Set register 0x0012 to 0x00010203, set register 0x0011 to 0x04050607, set register 0x0010 to 0x00000004.and then set register 0x0010 to the0x00000104. The 3rd and 4th words are writing done.

   Set register 0x0012 to 0x08090a0b, set register 0x0011 to 0x0c0d0e0f, set register 0x0010 to 0x00000004, and then set register 0x0010 to 0x00000104. The 5th and 6th words are writing done.

2. Set register 0x0012 to 0xfdfdfdfd, set register 0x0010 to 0x00000001, then set register 0x0010 to 0x00000101. The 7th word is writing done.

8.4 Software Reading Received Packets

In CoaXPress Device high speed and low speed upconnection receiver, there is a FIFO to buffer received Command packets or Event Ack packets from the CoaXPress Device. When received a complete packet, the software can read the received packet.

To execute one reading operation for high speed upconnection buffer in software, the following steps should be observed:

1. Read register 0x0041 bit [0], if the reading value is 1, it means they are data to be read, software should execute the following reading operation. Otherwise, no data to be read, and software should stop reading and waiting for some time to check it again.
2. Set register 0x0042 bit [0] to 1, then set register 0x0042 bit [0] to 0, the rising edge of this value execute 1 reading operation.
3. Read register 0x0045 to get reading data bit [63:32], then read register 0x0044 to get reading data bit [31:0].
4. Read register 0x0043 bit [0], if the reading value is 1, it means the reading data indicates how many words in the reading packet. Otherwise, the reading data is packet data bit [0] is 1, the reading data is Word N+2.
5. Read register 0x0043 bit [1], if the reading value is 1, it means all reading data bit [63:0] is available. Otherwise, only bit [63:32] is available, bit [31:0] should be ignored.
6. Read register 0x0043 bit [2], if the reading value is 1, it means the reading data is the last word of the packet.
7. Read register 0x0043 bit [3], if the reading value is 1, it means the reading data is the first word of the packet.

Software should execute step 1 periodically to check if there is any received packet in buffer, then read the packet out by repeating the following steps until there is no data to be read.

To execute one reading operation for low speed upconnection, buffer in software, the following steps should be observed:

1. Read register 0x008A bit [4], if the reading value is 1, it means there are data to be read, software should execute the following reading operation. Otherwise, no data to be read, and software should stop reading and waiting for some time to check it again.

   Set register 0x0089 bit [0] to 1, then Set register 0x0089 bit [0] to 0, the rising edge of this value execute 1 reading operation.

2. Read register 0x008C reading data bit [31:0].
3. Read register 0x008B, bit [2] if the reading value is 1, it means the reading data indicates how many words in the reading packet. Otherwise, the reading data is packet data.
4. Read register 0x008B bit [0], if the reading value is 1, it means the reading data is the last word of the packet.
5. Read register 0x008B bit [1], if the reading value is 1, it means the reading data is the first word of the packet.

Software should execute step 1 periodically to check if there is any received packet in buffer, then read the packet out by repeating the following steps until there is no data to be read.

8.5 Register Definition

The following table shows the internal registers defined in CoaXPress Device IP:

Table 12 • **CoaXPress Device IP Registers**

Addr	Bits	Name	Type	Default	Description
0x0000	[23:16]	sdp0_line_marker_len	RW	0x01	SDP mapper0 image line marker length in unit of 64bit
	[7:0]	sdp0_sid	RW	0x00	SDP mapper0 stream ID
0x0002	[23:16]	sdp1_line_marker_len	RW	0x01	SDP mapper1 line marker length in unit of 64bit
	[7:0]	sdp1_sid	RW	0x00	SDP mapper1 stream ID
0x0010	[8]	hsdc_sw_tx_pkt_wen	RW	0x0	The positive edge of this value writes hsdc_sw_tx_pkt_data into buffer
	[2]	hsdc_sw_tx_pkt_mod	RW	0x0	Indicates hsdc_sw_tx_pkt_data is packet all bits available or only bit [63:32] are available
	[1]	hsdc_sw_tx_pkt_sop	RW	0x0	Indicates hsdc_sw_tx_pkt_data is packet SOP
	[0]	hsdc_sw_tx_pkt_eop	RW	0x0	Indicates hsdc_sw_tx_pkt_data is packet EOP
00x11	[31:0]	hsdc_sw_tx_pkt_data0	RW	0x00000000	Software packet writing data [31:0]
00x12	[31:0]	hsdc_sw_tx_pkt_data1	RW	0x00000000	Software packet writing data [63:32]
00x15	[15:0]	hsdc_ack_timeout_time	RW	0x1000	Define how many clock cycles is waiting Ack timeout time
00x16	[3:0]	hsdc_send_trigger_times	RW	0x3	Defines the maximal sending times for sending the same trigger event
00x17	[16]	hsdc_trigger_src_en	RW	0x1	Enable high speed Trigger TriggerSource field
00x19	[0]	hsdc_test_en	RW	0x0	Enable high-speed transmitter sending test frame
00x21	[31:0]	hsdc_tx_test_frame_num	RW	0x00000000	How many test frames have been transmitted
00x23	[15:0]	hsdc_tx_sw_pkt_num	RC	0x0000	How many software packets have been transmitted
00x41	[0]	hsuc_sw_rx_pkt_val	RO	0x0	Indicates if there is received packet data to be software reading in high speed upconnection receiver
00x42	[0]	hsuc_sw_rx_pkt_ren	RW	0x0	The rising edge of this bit reads 64 bits data from high speed upconnection receiver buffer

Table 12 • **CoaXPress Device IP Registers**

Addr	Bits	Name	Type	Default	Description
00x43	[3]	hsuc_sw_rx_pkt_sop	RO	0x0	Indicates if hsuc_sw_rx_data is the packet header
	[2]	hsuc_sw_rx_pkt_eop	RO	0x0	Indicates if hsuc_sw_rx_data is the packet end word
	[1]	hsuc_sw_rx_pkt_mod	RO	0x0	1 means all 64bit reading data hsuc_sw_rx_data is available, 0 means only bit [63:32] is available.
	[0]	hsuc_sw_rx_pkt_len	RO	0x0	1 means the hsuc_sw_rx_data is packet length (in unit of 32bit words), 0 means the reading data is packet data
00x44	[31:0]	hsuc_sw_rx_data0	RO	0x00000000	Reading data bit [63:32] from high-speed upconnection receiver packet buffer
00x45	[31:0]	hsuc_sw_rx_data1	RO	0x00000000	Reading data bit [31:0] from high-speed upconnection receiver packet buffer
00x50	[0]	hsuc_trigger_src_en	RW	0x1	Enable TriggerSource field in high speed upconnection receiver
00x60	[31:0]	hsuc_rx_test_num	RC	0x00000000	High speed upconnection receiver received test frame number.
00x61	[31:0]	hsuc_rx_err_test_num	RC	0x00000000	High speed upconnection receiver received error test frame number.
0x0080	[31:0]	lsuc_rx_test_num	RC	0x00000000	How many test frames have been received in low-speed upconnection receiver.
0x0081	[31:0]	lsuc_rx_err_test_num	RC	0x00000000	How many error test frames have been received in low-speed upconnection receiver.
0x0089	[0[	lsuc_rx_sw_pkt_ren	RW	0x0	The posedge edge of this value read 1, 32bit data from low speed upconnection receiver software packet buffer.
0x008A	[4]	lsuc_rx_sw_pkt_val	RO	0x0	Indicates if there is any received software packet data to be read in low speed upconnection receiver.
0x008B	[2]	lsuc_sw_rx_pkt_len	RO	0x0	1 means the lsuc_sw_rx_data is packet length (in unit of 32bit words), 0 means the reading data is packet data.
	[1]	lsuc_sw_rx_pkt_sop	RO	0x0	Indicates if lsuc_sw_rx_pkt_data is packet SOP
	[0]	lsuc_sw_rx_pkt_eop	RO	0x0	Indicates if lsuc_sw_rx_pkt_data is packet EOP
0x008C	[31:0]	lsuc_rx_sw_pkt_data	RO	0x00000000	Reading data from low speed receiver software packet buffer