SMPTE UHD-SDI v1.0

LogiCORE IP Product Guide

Vivado Design Suite

PG205 September 28, 2020

Table of Contents

Introduction

The serial digital interface (SDI) family of standards from the Society of Motion Picture and Television Engineers (SMPTE) is widely used in professional broadcast video equipment. The Xilinx® LogiCORE™ IP SMPTE UHD-SDI core supports SMPTE SDI data rates from SD-SDI through 12G-SDI. The core supports both transmit and receive. It is compatible with 7 series GTX transceivers and UltraScale™ and UltraScale+™ GTH transceivers. The SMPTE UHD-SDI core is similar to the SMPTE SD/HD/3G-SDI core. In addition to this IP core, Xilinx also provides UHD-SDI RX and UHD-SDI TX subsystems designed to reduce design complexity. For more information, refer to PG289 and PG290.

Features

Supports the following SMPTE standards; for details see Standards.

- SMPTE ST 259
- SMPTE RP 165
- SMPTE ST 292
- SMPTE ST 372
- SMPTE ST 424
- SMPTE ST 2081-1
- SMPTE ST 2082-1
- SMPTE ST 352

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale+™ Families (GTHE4) Zynq® UltraScale+ MPSoC (GTHE4) Kintex® UltraScale™ (GTHE3) Virtex® UltraScale (GTHE3) 7 Series (GTXE2) Zynq-7000 SoC (GTXE2) Zynq UltraScale+ RFSoC
Supported User Interfaces	N/A
Resources	Performance and Resource Utilization web page
Provided with Core	
Design Files	Verilog source code
Example Design	*Implementing SMPTE SDI Interfaces with UltraScale GTH Transceivers* (XAPP1248) [Ref 6] *Implementing SMPTE SDI Interfaces with 7 Series GTX Transceivers* (XAPP1249) [Ref 7]
Test Bench	Verilog
Constraints File	XDC
Simulation Model	Verilog source HDL
Supported S/W Driver	N/A
Tested Design Flows[2]	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Release Notes and Known Issues	Master Answer Record: 54547
All Vivado IP Change Logs	Master Vivado IP Change Log: 72775
Xilinx Support web page	

Notes:

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

Overview

The SMPTE UHD-SDI core provides hooks in the transmitter to allow the embedding of any type of ancillary data, including audio. Any ancillary data embedded in the SDI data streams is present on the data streams output by the SMPTE UHD-SDI core receiver.

For 6G-SDI and 12G-SDI, the SMPTE UHD-SDI core supports up to 16 elementary (non-multiplexed) data streams. In the SMPTE 6G-SDI and 12G-SDI mapping standards documents, the term *data streams* refers to both multiplexed and non-multiplexed data streams and care must be used when interpreting these documents to determine how many elementary data streams are used by each mapping method. At this point in the development of the 6G and 12G-SDI standards, the maximum number of elementary data streams that can be multiplexed together into a 6G-SDI data stream is 8 and into a 12G-SDI data stream is 16. As of this publishing, the 16-way interleave of data streams only occurs in dual link 12G-SDI.

Elementary data streams refers to an SDI data stream that is not multiplexed. For example, a HD-SDI signal consists of two elementary data streams, usually referred to as Y and C, that are multiplexed together onto the virtual 10-bit HD-SDI interface. Likewise, a 3G-SDI level A signal also consists of two elementary data streams, called data stream 1 and data stream 2 that are multiplexed together onto the 10-bit virtual 3G-SDI interface. A 3G-SDI level B-DL signal, however, consists of four elementary data streams, a Y and a C data stream for the HD-SDI signals that are aggregated together onto the 3G-SDI level B interface. These four elementary streams get multiplexed in a 4-way multiplex onto the 10-bit virtual 3G-SDI interface.

The SMPTE UHD-SDI core only accepts and outputs elementary, non-multiplexed, data streams on its data stream inputs and outputs. The multiplexing and de-multiplexing of data streams occurs internally to the SMPTE UHD-SDI core. Note that SD-SDI is a special case. The ST 259 standard defines an elementary data stream that actually carries both the Y and C components. This is considered to be an elementary data stream by the SMPTE UHD-SDI core.

The SMPTE UHD-SDI core does not do any video format mapping. The user application must do any necessary mapping of video onto the elementary data streams prior to feeding those streams into to the SMPTE UHD-SDI transmitter and must reconstruct the video image from the elementary streams output by the SMPTE UHD-SDI receiver. For all video formats on SD-SDI and single-link HD-SDI and for 1080p 50, 59.94, and 60Hz 4:2:2 YCbCr 10b video on 3G-SDI level A, no mapping is necessary because there is a one-to-one correspondence between the data streams of these formats and the elementary data streams into and out of the SMPTE UHD-SDI core. For dual-link HD-SDI, 3G-SDI level B-DL, multi-link 3G-SDI, 6G-SDI, and 12G-SDI, mapping of the video formats to and from elementary data streams is required and is not done in the SMPTE UHD-SDI core.

SD-SDI

The SMPTE UHD-SDI core is designed to support the 270 Mb/s bit rate (level C) of the SD-SDI standard.

The SMPTE UHD-SDI core fully supports the SMPTE RP 165 Error Detection and Handling (EDH) standard for the receive and transmit sections.

HD-SDI

Although this standard is called a 1.5 Gb/s interface, the bit rates supported by HD-SDI are actually 1.485 Gb/s and 1.485/1.001 Gb/s. The SMPTE UHD-SDI core fully supports both of these bit rates. The SMPTE UHD-SDI core also fully supports generation (TX-side) and checking (RX-side) of CRC values for each video line and the insertion (TX-side) and capture (RX-side) of line number values for each line.

3G-SDI

This standard is called a 3 Gb/s interface, but the actual bit rates are 2.97 Gb/s and 2.97/1.001 Gb/s. The SMPTE UHD-SDI core fully supports both of these bit rates. 3G-SDI supports several different mapping levels, described in the SMPTE ST 425-1 standard. These levels are called A, B-DL, and B-DS. The SMPTE UHD-SDI core supports all three of these levels. As with the HD-SDI standard, the SMPTE UHD-SDI core supports CRC generation and checking and line number insertion and capture for 3G-SDI.

6G-SDI

The actual bit rates for this standard are 5.94 Gb/s and 5.94/1.001 Gb/s. The SMPTE UHD-SDI core fully supports both of these bit rates. The SMPTE UHD-SDI core supports CRC generating and checking and line number insertion and capture for 6G-SDI. The SMPTE UHD-SDI core supports 4-way and 8-way data stream interleaving for 6G-SDI.

12G-SDI

The actual bit rates for this standard are 11.88 Gb/s and 11.88/1.001 Gb/s. The SMPTE UHD-SDI core fully supports both of these bit rates. The SMPTE UHD-SDI core supports CRC generating and checking and line number insertion and capture for 12G-SDI. The SMPTE UHD-SDI core supports 4-way, 8-way, and 16-way data stream interleaving for 12G-SDI.

Payload ID

The SMPTE UHD-SDI core implements a SMPTE ST 352 payload ID ancillary data packet insertion capability for the transmitter that works in all SDI modes (SD-SDI, HD-SDI, 3G-SDI, 6G-SDI, 12G-SDI, and dual link HD-SDI). The receive side also detects and captures the four data bytes of ST 352 payload ID packets.

Ancillary Data Support

The SMPTE UHD-SDI core allows the application to implement ancillary data packet insertion prior to transmission. While the core does not provide ancillary data packet insertion capability, except for ST 352 payload ID packets, it has the necessary data paths to allow ancillary data packet insertion to be implemented by the application. On the receive side, all embedded ancillary data is preserved by the SMPTE UHD-SDI core receiver section and is present in the SDI data streams output from the core. Applications can process the received SDI data streams to receive and/or modify the ancillary data as needed.

Complete SMPTE SDI Interface Solution

A complete SDI interface is comprised of:

- A multi-Gigabit transceiver (such as Xilinx® 7 series GTX, UltraScale™ GTH, or UltraScale+™ GTH)
- The SMPTE UHD-SDI IP core
- Control logic for the transceiver
- Third-party SDI cable driver and cable equalizer to interface the transceiver to the SDI cable
- Transceiver reference clock sources

Figure 1-1 shows the high-level block diagram of an SDI receive/transmit interface using the SMPTE UHD-SDI core. In this figure, the blocks shaded grey are external to the FPGA, everything else is contained within the FPGA. The SMPTE UHD-SDI core implements one SDI receiver and one SDI transmitter. If only a receiver or only a transmitter is needed by the application, the input ports for the unused half of the core can be tied to ground, and the output ports left unconnected. The synthesis tool optimizes the unused portion of the core out of the application.

When both the receiver and transmitter sections of the SMPTE UHD-SDI core are used, the receiver and transmitter are completely independent. They can be operating in different SDI modes and line rates.

The SMPTE UHD-SDI core can be generated in 20-bit mode in which case it only supports SD, HD, and 3G-SDI modes. Or, it can be generated in 40-bit mode in which case it supports all SDI mode from SD-SDI through 12G-SDI.

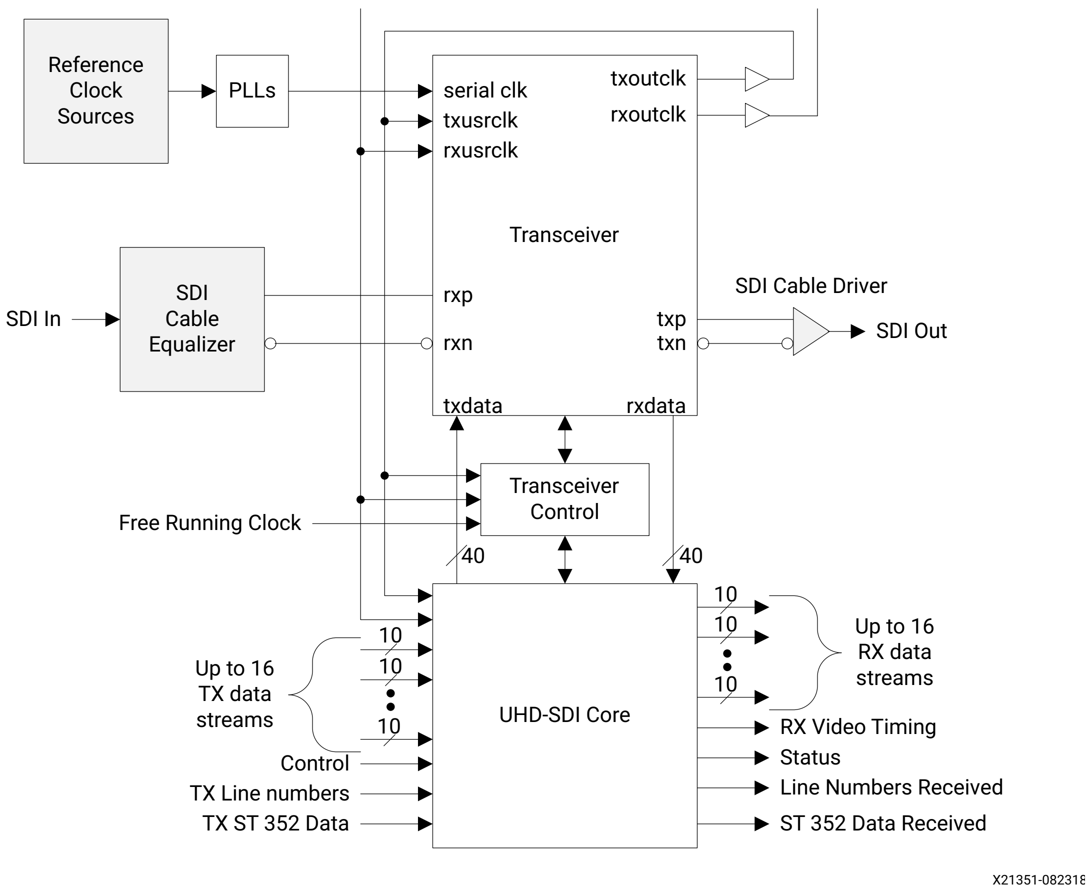


Figure 1-1: **Overview of a Complete SDI Interface**

Receiver Block Diagram

Figure 1-2 is a block diagram of the SMPTE UHD-SDI receiver. Data from the serial transceiver RX enters the SMPTE UHD-SDI receiver on the `rx_data_in` port 20-bit per clock cycle for SD, HD, and 3G modes or 40-bit per clock cycle for 6G and 12G modes. In SD mode, the 20-bit of data on `rx_data_in` go to the data recovery unit (DRU) which recovers 10 bit data words from the 11X oversampled data.

The data is descrambled by the SDI descrambler and then word aligned by the SDI framer. Immediately following the framer is the sync bit restore function. This function restores the 3FF and 000 values that are modified by the transmitter to reduce run lengths in 6G and 12G-SDI modes. These three blocks run at the full `rx_clk` speed and process 40, 20, or 10 bit of data per clock cycle depending on the SDI mode.

The output of the SDI framer goes to the stream demux which determines how many streams are interleaved together and then separates each data stream on a separate data path. There can be one, two, four, eight, or sixteen active data streams on the output of the stream demux.

Each data stream pair (and Y and C pair) go to a stream processing unit which does CRC error checking, line number capture, and ST 352 packet capture. In the diagram, the details of the stream processing unit are shown for stream processor 0 which processes data stream 1 and data stream 2. Data stream 1 also goes to the EDH processor for error detection in SD-SDI mode. Not shown in the block diagram is that data stream 2 can also be directed to stream processing unit 1 so that, in 3G-SDI level A mode, the ST 352 packet on data stream 2 can be captured. In 3G-SDI level A mode, both the Y (data stream 1) and C (data stream 2) data streams contain ST 352 packets. In addition to the 3G-SDI level A, 12G-SDI and 6G-SDI allow ST 352 packets to be sent and received on both Y and C data streams.

The stream demux also extracts the video timing and produces the `rx_eav`, `rx_sav`, and `rx_trs` timing signals. These timing signals are used by the SDI mode detection and transport detection blocks.

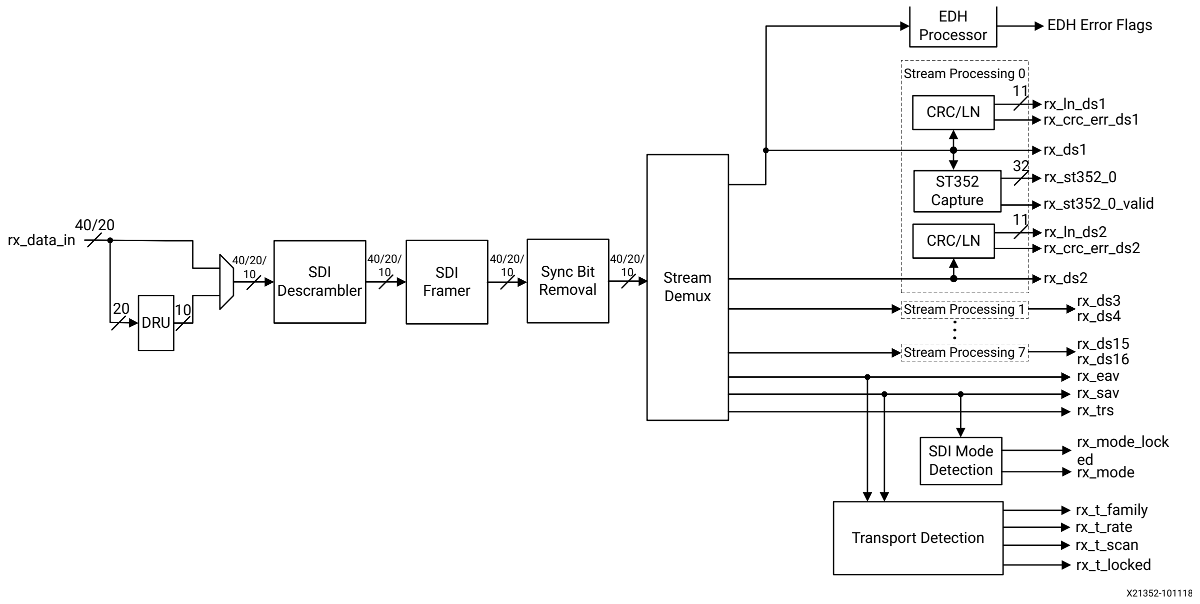


Figure 1-2: **SMPTE UHD-SDI RX Block Diagram**

Transmitter Block Diagram

Figure 1-3 shows a block diagram of the SMPTE UHD-SDI transmitter. Up to sixteen SDI data streams (`tx_ds1_in` through `tx_ds16_in`) enter the transmitter and first go through the ST 352 insertion modules where ST 352 payload ID packets are optionally inserted. The data streams output from the ST 352 insertion modules are called `tx_ds1_st352_out` through `tx_ds16_st352_out`. These streams are output so that the customer application can insert ancillary data after ST 352 packets have been inserted. The remaining portion of the transmitter can either use the streams output by the ST 352 packet insertion modules directly or the sixteen `tx_ds1_anc_in` through `tx_ds16_anc_in` data streams from the customer application ancillary data insertion function. Note that if the `tx_dsn_anc_in` data streams are used, they must be full SDI data streams, not just ancillary data.

Most frequently, ST 352 packets are only inserted into the Y data stream of each Y/C data stream pair. However, in 3G-SDI level A mode-only, ST 352 packets must be inserted into both data stream 1 and data stream 2. Muxes in the SMPTE UHD-SDI transmitter allow data stream 2 to be routed through the second ST 352 insertion module in 3G-SDI level A mode to allow that module to insert ST 352 packets into data stream 2, whereas normally data stream 3 is handled by the second ST 352 insertion module. The SMPTE UHD-SDI core has the capability to insert ST 352 packets into the C data stream and can be controlled using a parameter.

Each Y/C pair of data streams then goes through a stream processing block which optionally does line number insertion and CRC generation and insertion. Following stream processing, the data streams are interleaved by the stream MUX. This results in a multiplexed SDI data stream that is 40, 20, or 10 bit wide depending on the SDI mode. For 6G-SDI and 12G-SDI, the sync bit insertion algorithm is performed to reduce run lengths. Then the multiplexed data stream is scrambled by the SDI scrambler. In SD-SDI mode, the 10-bit scrambled data stream is processed by the SD-SDI bit replication module to replicate each bit 11 times to create a 270 Mb/s SDI serial stream with the serial transceiver TX running at 2.97 Gb/s. Finally, the data is output on the `tx_txdata` port to the serial transceiver to be serialized.

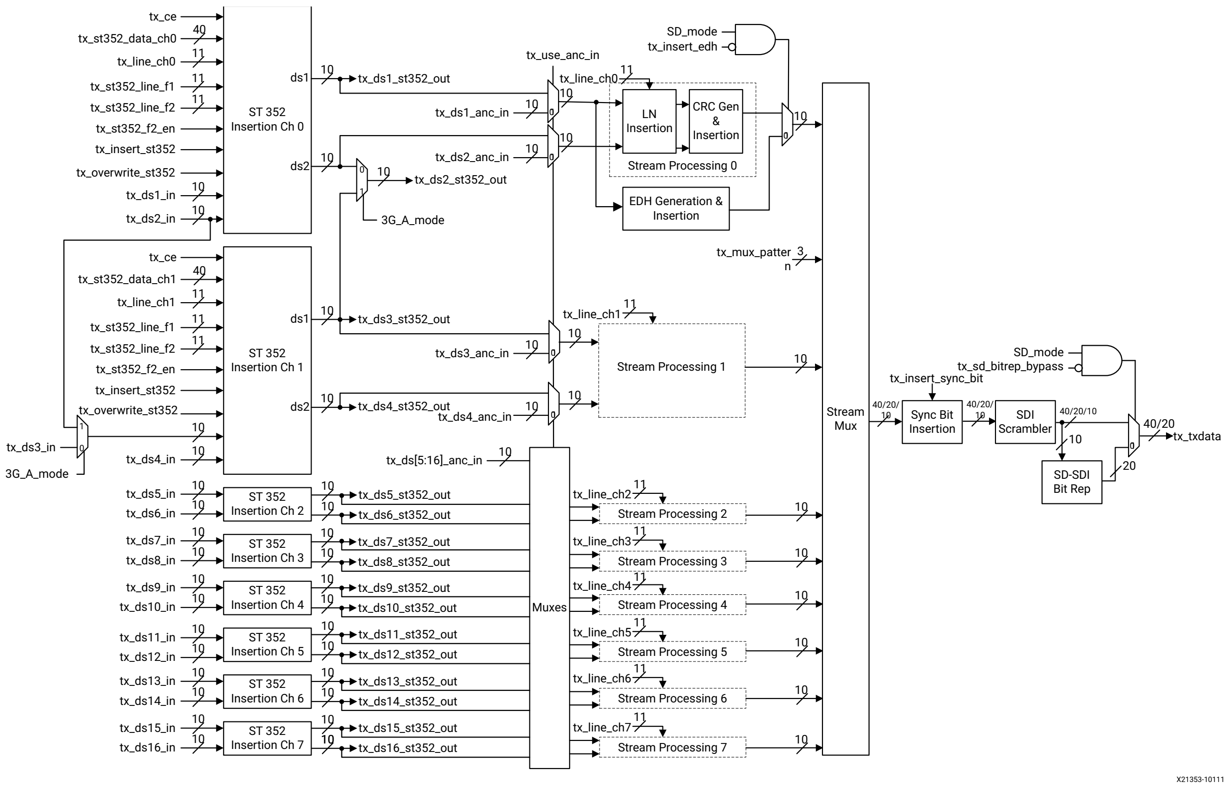


Figure 1-3: **SMPTE UHD-SDI TX Block Diagram**

1. SDI cable equalizer and cable driver are external to the FPGA.
2. The optional ANC packet insertion function is not included with the SMPTE UHD-SDI core.
3. For simplicity, the ST 352 insertion into the C data stream is not shown.

Feature Summary

The Xilinx SMPTE UHD-SDI IP core implements the data path for SDI receivers and transmitters. A complete SDI interface implementation consists of the SMPTE UHD-SDI core, a transceiver (internal to the FPGA or external), and control logic associated with the transceiver. The SMPTE UHD-SDI core supports SD-SDI, HD-SDI, 3G-SDI, 6G-SDI and 12G-SDI (level A, level B-DL, and level B-DS) and dual link HD-SDI standards. The SMPTE UHD-SDI core is validated with the GTX transceivers in Kintex®-7, Virtex®-7 FPGA, and Zynq®-7000 devices and GTH transceiver in UltraScale/UltraScale+ architecture.

Applications

- Professional broadcast cameras
- Professional digital video recorders
- Professional video processing equipment
- Medical imaging

Licensing

The SMPTE UHD-SDI core is provided at no additional cost with the Xilinx® Vivado® Design Suite under the terms of the Xilinx End User License. Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information about pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Chapter 2

Product Specification

Standards

The core supports the following SMPTE standards:

- SMPTE ST 259: SD-SDI at 270 Mb/s
- SMPTE RP 165: EDH for SD-SDI
- SMPTE ST 292: HD-SDI at 1.485 Gb/s and 1.485/1.001 Gb/s
- SMPTE ST 372: Dual Link HD-SDI (by instantiation of two SMPTE UHD-SDI cores)
- SMPTE ST 424: 3G-SDI with data mapped by any ST 425-x mapping at 2.97 Gb/s and 2.97/1.001 Gb/s
- SMPTE ST 2081-1: 6G-SDI with data mapped by any ST 2081-x mapping at 5.94 Gb/s and 5.94/1.001 Gb/s (including multi-link 6G-SDI)
- SMPTE ST 2082-1: 12G-SDI with data mapped by any ST 2082-x mapping at 11.88 Gb/s and 11.88/1.001 Gb/s (including multi-link 12G-SDI)
- Dual link and quad link 6G-SDI and 12G-SDI are supported by instantiating two or four SMPTE UHD-SDI cores
- SMPTE ST 352: Payload ID packets into Y-Stream and C-Stream fully supported in both the RX and TX

Performance

Maximum Frequencies

In 12G-SDI mode, the maximum frequency of the RX and TX clocks is 297 MHz. In 6G-SDI, 3G-SDI, and SD-SDI modes, the maximum frequency of the RX and TX clocks is 148.5 MHz. In HD-SDI mode, the maximum frequency of the RX and TX clocks is 74.25 MHz.

Latency

The SMPTE UHD-SDI TX latency is measured from the native SDI stream(s) input to `tx_txdata` output. The SMPTE UHD-SDI RX latency is measured from the `rx_data_in` input to Native SDI data stream(s) output. Both TX latency and RX latency clock cycles are taken into account only when the respective SDI mode clock enable is asserted.

Table 2-1 provides the latency numbers for various core configurations.

Table 2-1: **Latency for SMPTE UHD-SDI Core Configurations**

SDI Mode	TX Latency (in tx_clk)	RX Latency (in rx_clk)
SD-SDI	11	21
HD-SDI	12	21
3G-SDI Level A	12	20
3G-SDI Level B	11	10
6G-SDI (8 streams)	11	11
12G-SDI (8 streams)	11	11

Performance and Resource Utilization

For full details about performance and resource utilization, visit the Performance and Resource Utilization web page.

SMPTE UHD-SDI Core Port Descriptions

RX Ports

All ports in Table 2-2 are synchronous with `rx_clk`.

Table 2-2: **Receiver Ports**

Port Name	I/O	Description
rx_clk	I	Connect to global clock driving the rxusrclk port of the serial transceiver.
rx_rst	I	Synchronous reset input.
rx_mode_detect_rst	I	Synchronous reset that resets only the SDI mode detect search function.

Table 2-2: **Receiver Ports *(Cont'd)***

Port Name	I/O	Description
rx_data_in[19/39:0]	I	Connect to serial transceiver RXDATA port. In SD mode, the NI-DRU data output bus must be connected to the least significant 10-bit of this port (rx_data_in[9:0]). This is different than the older core where the DRU input had a separate input port. Note that in SD, HD, and 3G-SDI modes, the serial transceiver is configured for 20-bit RXDATA port width, but in 6G-SDI and 12G-SDI modes, the serial transceiver is configured for 40-bit RXDATA port width. The RXDATA port width is switched between 20-bit and 40-bit dynamically. When either 6G-SDI or 12G-SDI is supported, the full 40-bit data path between the serial transceiver and the rx_data_in port must be connected. If only 3G-SDI and slower rates are supported, connect the 20-bit RXDATA port of the serial transceiver to rx_data_in[19:0].
rx_sd_data_strobe	I	Connect to the data strobe output of the NI-DRU.
rx_frame_en	I	This input enables the SDI framer function. When this input is High, the framer automatically readjusts the output word alignment to match the alignment of each timing reference signal (TRS). TRS is a generic term referring to both EAV and SAV sequences. Normally, this input should be always be High. But, the user application may control this input to implement TRS filtering to prevent a signal misaligned TRS from causing erroneous alignment changes.

rx_bit_rate	I	This input port indicates which bit rate is being received in HD-SDI, 3G-SDI, 6G-SDI, and 12G-SDI modes. This input is only used to generate the value on the rx_t_rate output port, so if the rx_t_rate output port is not used, then it is not required that the rx_bit_rate input be driven.
		When using the Xilinx® device transceivers, the device-specific transceiver control module contains a bit rate detector that generates the signal to be connected to the rx_bit_rate input port.
		
		HD-SDI mode:
		rx_bit_rate = 0: Bit rate = 1.485 Gb/s
		rx_bit_rate = 1: Bit rate = 1.485/1.001 Gb/s
		
		3G-SDI mode:
		rx_bit_rate = 0: Bit rate = 2.97 Gb/s
		rx_bit_rate = 1: Bit rate = 2.97/1.001 Gb/s
		
		6G-SDI mode:
		rx_bit_rate = 0: Bit rate = 5.94 Gb/s
		rx_bit_rate = 1: Bit rate = 5.94/1.001 Gb/s
		
		12G-SDI mode:
		rx_bit_rate = 0: Bit rate = 11.88 Gb/s
		rx_bit_rate = 1: Bit rate = 11.88/1.001 Gb/s

Table 2-2: **Receiver Ports *(Cont'd)***

Port Name	I/O	Description
rx_mode_enable[5:0]	I	This port has unary bits to enable reception of each of the five SDI modes: Bit 0 enables HD-SDI mode Bit 1 enables SD-SDI mode Bit 2 enables 3G-SDI mode Bit 3 enables 6G-SDI mode Bit 4 enables 12G-SDI 11.88 Gb/s mode Bit 5 enables 12G-SDI 11.88/1.001 Gb/s mode When a bit is High, the corresponding SDI mode is enabled. When a bit is Low, the receiver does not attempt to detect incoming SDI signals of that mode. Disabling unused SDI modes using these bits decreases the amount of time it takes for the receiver to lock to the incoming signal when it changes modes. Previously this was a 3-bit port, now it is 5-bit to add bits for 12G and 6G.
rx_mode_detect_en	I	This port enables the SDI mode detection feature when High. When enabled, the SDI mode detector controls the receiver to search for and lock to the incoming SDI data stream. When disabled, the user application must tell the SDI receiver what SDI mode to operate in using the rx_forced_mode port.
rx_forced_mode[2:0]	I	When the rx_mode_detect_en input is Low, disabling the automatic SDI mode detection feature, the receiver operates in the SDI mode specified by the value on the rx_forced_mode_port. 000 = HD 001 = SD 010 = 3G 100 = 6G 101 = 12G 11.88 Gb/s 110 = 12G 11.88/1.001 Gb/s
rx_ready	I	When the rx_ready input is Low, the SDI mode detection function is forced into the unlocked state until rx_ready goes High. This input is used to keep the SDI mode detection function in the unlocked while the serial transceiver is being changed between line rates and reset. Once the serial transceiver has completed its reset cycle, the rx_ready can be asserted High and the SDI mode detection function determines if good data is being received or if the serial transceiver should be changed to another mode. This input is treated as an asynchronous input.
rx_mode[2:0]	O	This output port indicates the current SDI mode of the receiver: 000 = HD 001 = SD 010 = 3G 100 = 6G 101 = 12G 1000/1000 110 = 12G 1000/1001 When the receiver is not locked, the rx_mode port changes values as the receiver searches for the correct SDI mode. During this time, the rx_mode_locked output is Low. When the receiver detects the correct SDI mode, the rx_mode_locked output goes High.

Table 2-2: **Receiver Ports *(Cont'd)***

Port Name	I/O	Description
rx_mode_hd	O	High when RX is locked in HD-SDI mode
rx_mode_sd	O	High when RX is locked in SD-SDI mode
rx_mode_3g	O	High when RX is locked in 3G-SDI mode
rx_mode_6g	O	High when RX is locked in 6G-SDI mode
rx_mode_12g	O	High when RX is locked in 12G-SDI mode (either bit rate)
rx_mode_locked	O	When this output is Low, the receiver is actively searching for the SDI mode that matches the input data stream. During this time, the rx_mode_output port changes frequently. When the receiver locks to the correct SDI mode, the rx_mode_locked output goes High. When the SDI mode detect function is disabled (rx_mode_detect_en = Low), this output is always asserted High. In this case, it is not a reliable indicator of whether or not the SDI receiver is locked to the incoming SDI signal.
rx_t_locked	O	This output is High when the transport detection function in the receiver has identified the transport format of the SDI signal.
rx_t_family[3:0]	O	This output indicates which family of video signals is being used as the transport of the SDI interface. This output is only valid when rx_t_locked is High. This port does not necessarily identify the video format of the picture being transported. It only identifies the transport characteristics. See Table 2-7 for encoding of this port.
rx_t_rate[3:0]	O	This output indicates the frame rate of the transport. This is not necessarily the same as the frame rate of the actual picture. This output is only valid when rx_t_locked is High. See Table 2-8 for details on the encoding of this port.
rx_t_scan	O	This output indicates whether the transport is interlaced (Low) or progressive (High). This is not necessarily the same as the scan mode of the actual picture. This output is only valid when rx_t_locked is High.
rx_level_b_3g	O	IN 3G-SDI mode, this output is asserted High when the input signal is level B and Low when it is level A. This output is only valid when rx_mode_3g is High.
rx_ce_out[NUM_RX_CE-1:0]	O	This is the RX clock enable output. There are NUM_RX_CE copies of this clock enable on this port. These clock enables are valid in all SDI modes. In SD mode, the CEs have a nominal 5/6/5/6 cadence. In HD and 3GA modes, the CEs are always High. In 3GB mode, the CEs have a 50% duty cycle. In 6G, the duty cycle can by 100% or 50% depending on how may data streams are interleaved onto the signal. In 12G, the duty cycle can be 50% or 25% depending on how many data streams are interleaved onto the signal. This port replaces the rx_ce_sd and rx_dout_rdy_3g ports of the old core and combines their functionality by being correct for all SDI modes.

Table 2-2: **Receiver Ports *(Cont'd)***

Port Name	I/O	Description
rx_active_streams[2:0]	O	This port indicates the number of data streams that are active for the current video format being received. The number of active data streams is 2^active_streams. 000: 1 active stream 001: 2 active streams 010: 4 active streams 011: 8 active streams 100: 16 active streams
rx_ln_ds<n>[10:0]	O	Captured line number from rx_ds<n>, where <n> is 1 to 16.
rx_st352_0[31:0]	O	ST 352 payload ID packet data bytes captured from ds1
rx_st352_0_valid	O	High when rx_st352_0 is valid.
rx_st352_1[31:0]	O	The ST 352 payload ID packet data bytes captured from ds3 are output here. In 3G-SDI level A mode, the ST 352 payload ID packet data bytes from ds2 are output here.
rx_st352_1_valid	O	High when rx_st352_1 is valid.
rx_st352_2[31:0]	O	ST 352 payload ID packet data bytes captured from ds5
rx_st352_2_valid	O	High when rx_st352_2 is valid.
rx_st352_3[31:0]	O	ST 352 payload ID packet data bytes captured from ds7
rx_st352_3_valid	O	High when rx_st352_3 is valid.
rx_st352_4[31:0]	O	ST 352 payload ID packet data bytes captured from ds9
rx_st352_4_valid	O	High when rx_st352_4 is valid.
rx_st352_5[31:0]	O	ST 352 payload ID packet data bytes captured from ds11
rx_st352_5_valid	O	High when rx_st352_5 is valid.
rx_st352_6[31:0]	O	ST 352 payload ID packet data bytes captured from ds13
rx_st352_6_valid	O	High when rx_st352_6 is valid.
rx_st352_7[31:0]	O	ST 352 payload ID packet data bytes captured from ds15
rx_st352_7_valid	O	High when rx_st352_7 is valid.
rx_st352_8[31:0]	O	ST 352 payload ID packet data bytes captured from ds2 ***Note:*** This signal has the same value as rx_st352_1[31:0] for 3G-SDI level A.
rx_st352_8_valid	O	High when rx_st352_8 is valid.
rx_st352_9[31:0]	O	ST 352 payload ID packet data bytes captured from ds4
rx_st352_9_valid	O	High when rx_st352_9 is valid.
rx_st352_10[31:0]	O	ST 352 payload ID packet data bytes captured from ds6
rx_st352_10_valid	O	High when rx_st352_10 is valid.
rx_st352_11[31:0]	O	ST 352 payload ID packet data bytes captured from ds8
rx_st352_11_valid	O	High when rx_st352_11 is valid.
rx_st352_12[31:0]	O	ST 352 payload ID packet data bytes captured from ds10

Table 2-2: **Receiver Ports *(Cont'd)***

Port Name	I/O	Description
rx_st352_12_valid	O	High when rx_st352_12 is valid.
rx_st352_13[31:0]	O	ST 352 payload ID packet data bytes captured from ds12
rx_st352_13_valid	O	High when rx_st352_13 is valid.
rx_st352_14[31:0]	O	ST 352 payload ID packet data bytes captured from ds14
rx_st352_14_valid	O	High when rx_st352_14 is valid.
rx_st352_15[31:0]	O	ST 352 payload ID packet data bytes captured from ds16
rx_st352_15_valid	O	High when rx_st352_15 is valid.
rx_crc_err_ds1 to rx_crc_err_ds16	O	These 16 ports are the CRC error indicator for each data stream output. When a CRC is detected on a line, the CRC error signal of that data stream becomes asserted starting a few clock cycles after the last CRC word is output on the data stream ports following the EAV that ends the line containing the error. The CRC signal remains asserted for one line time.
rx_eav	O	This output is asserted High when the XYZ word of an EAV is present on the data stream output ports.
rx_sav	O	This output is asserted High when the XYZ word of a SAV is present on the data stream output ports.
rx_trs	O	This output is asserted High while the four consecutive words of any EAV or SAV are present on the data stream output ports, from the 3FF word through the XYZ word.
rx_ds1[9:0]	O	Data stream 1 output. In SD mode this is interleaved Y/C. In HD and 3GA modes, this is the Y channel. In 3GB mode, this is the link A Y channel. In 6G and 12G modes, this is ds1. Same as the rx_ds1a output port of previous core.
rx_ds2[9:0]	O	Data stream 2 output. Not used in SD mode. In HD and 3GA modes, this is the C channel. In 3GB mode, this is the link A C channel. In 6G and 12G modes, this is ds2. Same as the rx_ds2a port of previous core.
rx_ds3[9:0]	O	Data stream 3 output. Not used in SD, HD, and 3GA modes. In 3GB mode, this is the link B Y channel. In 6G and 12G modes this is ds3. Same as the rx_ds1b port of previous core.
rx_ds4 through rx_ds16[9:0]	O	Additional data stream output ports. The number of ports that are active depend on the number of data streams interleaved on the SDI signal. These ports are never active in SD, HD, or 3G modes.
rx_edh_errcnt_en[15:0]	I	This input controls which EDH error conditions increment the rx_edh_errcnt counter. See Table 2-4 for encoding of this port.
rx_edh_clr_errcnt	I	When High, this input clears the rx_edh_errcnt counter. This input port must be High during the same clock cycle when rx_ce_out is also High in order to clear the error counter.
rx_edh_ap	O	This output is asserted High when the active picture CRC calculated for the previous field does not match the AP CRC value in the EDH packet.

Table 2-2: **Receiver Ports *(Cont'd)***

Port Name	I/O	Description
rx_edh_ff	O	This output is asserted High when the full field CRC calculated for the previous field does not match the FF CRC value in the EDH packet.
rx_edh_anc	O	This output is asserted High when an ancillary data packet checksum error is detected.
rx_edh_ap_flags[4:0]	O	The active picture error flag bits from the most recently received EDH packet are output on this port. See Table 2-5 for encoding of this port.
rx_edh_ff_flags[4:0]	O	The full frame error flag bits from the most recently received EDH packet are output on this port. See Table 2-5 for encoding of this port.
rx_edh_anc_flags[4:0]	O	The ancillary error flag bits from the most recently received EDH packet are output on this port. See Table 2-5 for encoding of this port.
rx_edh_packet_flags[3:0]	O	This port outputs four error flags related to the most recently received EDH packet. See Table 2-6 for encoding of this port.
rx_edh_errcnt[15:0]	O	This is the SD-SDI EDH error counter. It increments once per field when any of the error conditions enabled by the rx_edh_err_en port occur during that field.

TX Ports

All ports are synchronous with `tx_clk`.

Table 2-3: **Transceiver Ports**

Port Name	I/O	Description
tx_clk	I	This clock input clocks the SDI transmitter data path. It should be driven by the same clock that drives the txusrclk port of the serial transceiver, typically the serial transceiver signal, txoutclk, after being buffered by a global clock buffer.
tx_ce	I	This is the clock enable input for the main portion of the transmitter data path. This is somewhat equivalent to the tx_din_rdy port of the old core. It must be High in SD, HD, and 3GA modes. In 3GB mode, it must have a 50% duty cycle. In 6G and 12G modes, it must have a 100% duty cycle when 4 streams are interleaved, 50% duty cycle when 8 streams are interleaved, and 25% duty cycle when all 16 data streams are interleaved.
tx_sd_ce	I	This is the clock enable for SD-SDI mode. It must have exactly a 5/6/5/6 cadence in SD-SDI mode and must be High in all other modes. This is the same as the tx_ce port of the old core.
tx_edh_ce	I	This is the clock enable for the TX EDH processor. In SD-SDI mode, it must be exactly equal to the tx_sd_ce port with its 5/6/5/6 cadence. It must be phase aligned with tx_sd_ce. In all other modes, this ce can be driven Low to reduce the power consumed by the EDH processor.

Table 2-3: **Transceiver Ports *(Cont'd)***

Port Name	I/O	Description
tx_rst	I	This is a synchronous reset input. It resets the transmitter when High. In order to fully reset the transmitter, the tx_ce, tx_sd_ce, and tx_edh_ce inputs must be High when tx_rst is asserted.
tx_mode[2:0]	I	This input port is used to select the transmitter SDI mode: 000 = HD 001 = SD 010 = 3G 100 = 6G 101 = 12G
tx_insert_crc	I	When this input is High, the transmitter generates and inserts CRC values into the data streams for each video line in all modes except SD-SDI. When this input is Low, CRC values are not inserted into the data streams. This input is ignored in SD-SDI mode.
tx_insert_ln	I	When this input is High, the transmitter inserts line numbers into all active data streams after the EAV of each video line. The line numbers must be supplied on the tx_line_ch*n* input ports of all active data stream pairs. When this input is Low, line numbers are not inserted. This input is ignored in SD-SDI mode.
tx_insert_st352	I	When this input is High, ST 352 packets are inserted into the Y channel of the data streams, otherwise the packets are not inserted. ST 352 packets are mandatory in 3G, 6G, and 12G modes and optional in HD and SD modes. This is identical to the tx_insert_vpid port on previous core.
tx_overwrite_st352	I	If this input is High, ST 352 packets already present in the data streams are overwritten. If this input is Low, existing ST 352 packets are not overwritten. This is identical to the tx_overwrite_vpid port on the previous core.
tx_insert_edh	I	When this input is High, the transmitter generates and inserts EDH packets into every field in SD-SDI mode. When this input is Low, EDH packets are not inserted. This input is ignored in all modes except SD-SDI mode.
tx_mux_pattern[2:0]	I	This specifies the data stream interleaving pattern to be used. 000 = SD, HD, and 3G level A 001 = 3G level B 010 = 8 stream interleave in 6G and 12G modes 011 = 4 stream interleave in 6G mode 100 = 16 stream interleave in 12G mode Note that this port replaces the function of the tx_level_b_3g port of the old core.
tx_insert_sync_bit	I	In 6G and 12G modes, when this port is High, the sync bit insertion function is enabled for run length mitigation. For compliance with the ST 2081 and ST 2082 standards, sync bit insertion must be enabled. However, some early implementations of 6G-SDI and 12G-SDI receivers do not support sync bit insertion and when transmitting signals to those devices, sync bit insertion can be disabled by setting this port Low.

Table 2-3: **Transceiver Ports *(Cont'd)***

Port Name	I/O	Description
tx_sd_bitrep_bypass	I	This input bypasses the 11X bit replicator used in SD-SDI mode when High. For normal operation with Xilinx serial transceiver transmitters, this input must be Low so that the bit replicator function is active.
tx_line_ch0[10:0]	I	Line number input for ds1 and ds2. Equivalent to tx_line_a port of old core. Note that for a single video format, all tx_line_dsX ports normally get the same line number. The only reasons to apply different line numbers to different tx_line_dsX ports are when aggregating video streams or to apply different line numbers to the data streams when transporting 1080p 50, 59.94, or 60 Hz video using 3G-SDI level B.
tx_line_ch1[10:0]	I	Line number input for ds3 and ds4. Equivalent to tx_line_b port of old core.
tx_line_ch2[10:0]	I	Line number input for ds5 and ds6.
tx_line_ch3[10:0]	I	Line number input for ds7 and ds8.
tx_line_ch4[10:0]	I	Line number input for ds9 and ds10.
tx_line_ch5[10:0]	I	Line number input for ds11 and ds12.
tx_line_ch6[10:0]	I	Line number input for ds13 and ds14.
tx_line_ch7[10:0]	I	Line number input for ds15 and ds16.
tx_st352_line_f1[10:0]	I	The ST 352 packets are inserted into the HANC space of the line number specified by this input port. For interlaced video, this input port specifies a line number in field 1. For progressive video, this specifies the only line in the frame where the packets are inserted. The input value must be valid during the entire HANC interval. If tx_insert_st352 is Low, this input is ignored. This is the same as the tx_vpid_line_f1 port of the old core.
tx_st352_line_f2[10:0]	I	For interlace video, ST 352 packets are inserted on the line number in field 2 indicated by this value. For progressive video, this input port must be disabled by driving the tx_st352_f2_en port Low. The input value on this port must be valid during the entire HANC interval. This port is ignored if either tx_insert_st352 or tx_st352_f2_en are Low. This is the same as the tx_vpid_line_f2 port of the old core.
tx_st352_f2_en	I	This input controls whether or not ST 352 packets are inserted on the line indicated by tx_vpid_line_f2. For interlaced video, this input must be High if ST 352 packet insertion is enabled. For progressive video, this input must be Low if ST 352 packet insertion is enabled. If ST 352 packet insertion is disabled (tx_insert_st352 = Low), this port is ignored. This port is identical to the tx_vpid_line_f2_en port of the old core.
tx_st352_data_ch0[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds1 when tx_insert_st352 is High. The data bytes are ordered like this: {byte4, byte3, byte2, byte1}. This replaces the tx_vpid_byteX ports of the old core.

Table 2-3: **Transceiver Ports *(Cont'd)***

Port Name	I/O	Description
tx_st352_data_ch1[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds3 when tx_insert_st352 is High. In 3GA mode, this port specifies the data bytes that is inserted into the ST 352 packet of ds2.
tx_st352_data_ch2[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds5 when tx_insert_st352 is High.
tx_st352_data_ch3[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds7 when tx_insert_st352 is High.
tx_st352_data_ch4[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds9 when tx_insert_st352 is High. Enabled only when 12G-SDI 16DS is selected.
tx_st352_data_ch5[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds11 when tx_insert_st352 is High. Enabled only when 12G-SDI 16DS is selected.
tx_st352_data_ch6[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds13 when tx_insert_st352 is High. Enabled only when 12G-SDI 16DS is selected.
tx_st352_data_ch7[31:0]	I	The four data bytes of the ST 352 packet to be inserted into ds15 when tx_insert_st352 is High. Enabled only when 12G-SDI 16DS is selected.
tx_insert_c_str_st352_in	I	When this input is High, ST 352 packets are inserted into the C channel of the data streams; otherwise the packets are not inserted. ST 352 packets are mandatory in 3G, 6G, and 12G modes and optional in HD and SD modes.
tx_st352_str_switch_3g_a_in	I	Setting this bit to 1'b1 uses the ST 352 payload of the tx_st352_data_ch0_c[31:0] stream instead of tx_st352_data_ch2[31:0] and the C_TX_INSERT_C_STR_ST352 parameter should be selected by the user.
tx_st352_data_ch0_c[31:0]	I	ST 352 data for C-stream of channel 0 and is inserted into DS2.
tx_st352_data_ch1_c[31:0]	I	ST 352 data for C-stream of channel 0 and is inserted into DS4
tx_st352_data_ch2_c[31:0]	I	ST 352 data for C-stream of channel 0 and is inserted into DS6
tx_st352_data_ch3_c[31:0]	I	ST 352 data for C-stream of channel 0 and is inserted into DS8
tx_st352_data_ch4_c[31:0]	I	ST 352 data for C-stream of channel 4 and is inserted into DS10. Enabled only when 12G-SDI 16DS is selected
tx_st352_data_ch5_c[31:0]	I	ST 352 data for C-stream of channel 5 and is inserted into DS12. Enabled only when 12G-SDI 16DS is selected
tx_st352_data_ch6_c[31:0]	I	ST 352 data for C-stream of channel 6 and is inserted into DS14. Enabled only when 12G-SDI 16DS is selected
tx_st352_data_ch7_c[31:0]	I	ST 352 data for C-stream of channel 7 and is inserted into DS16. Enabled only when 12G-SDI 16DS is selected
tx_ds1_in[9:0]	I	Data stream 1 input: SD=Y/C, HD=Y, 3GA=DS1(Y), 3GB=AY, 6G/12G=DS1. This is the same as the tx_video_a_y_in port of the old core.

Table 2-3: **Transceiver Ports *(Cont'd)***

Port Name	I/O	Description
tx_ds2_in[9:0]	I	Data stream 2 input: HD=C, 3GA=DS2(C), 3GB=AC, 6G/12G=DS2. This is the same as the tx_video_a_c_in port of the old core.
tx_ds3_in[9:0]	I	Data stream 3 input: 3 GB=BY, 6G/12G=DS3. This is the same as the tx_video_b_y_in port of the old core.
tx_ds4_in[9:0]	I	Data stream 4 input: 3GB=BC, 6G/12G=DS4. This is the same as the tx_video_b_c_in port of the old core.
tx_ds5_in though tx_ds16_in[9:0]	I	Data stream DS5 through DS16 inputs. These are only used when doing 8 stream or 16 stream interleaving.
tx_ds1_st352_out[9:0]	O	This is the DS1 output for ANC data insertion. This is the same as the tx_ds1a_out port of the old core.
tx_ds2_st352_out[9:0]	O	This is the DS2 output for ANC data insertion. This is the same as the tx_ds2a_out port of the old core.
tx_ds3_st352_out[9:0]	O	This is the DS3 output for ANC data insertion. This is the same as the tx_ds1b_out port of the old core.
tx_ds4_st352_out[9:0]	O	This is the DS4 output for ANC data insertion. This is the same as the tx_ds2b_out port of the old core.
tx_ds5_st352_out[9:0] through tx_ds16_st352_out[9:0]	O	These are the DS5 through DS16 output ports after the ST 352 insertion. These data streams are made available for other ANC data insertion. The number of these ports that are active depends on the number of data streams being interleaved.
tx_ds1_anc_in[9:0]	I	DS1 input after ANC packet insertion. This is the same as the tx_ds1a_in port of the old core.
tx_ds2_anc_in[9:0]	I	DS2 input after ANC packet insertion. This is the same as the tx_ds2a_in port of the old core.
tx_ds3_anc_in[9:0]	I	DS3 input after ANC packet insertion. This is the same as the tx_ds1b_in port of the old core.
tx_ds4_anc_in[9:0]	I	DS4 input after ANC packet insertion. This is the same as the tx_ds2b_in port of the old core.
tx_ds5_anc_in[9:0] through tx_ds16_anc_in[9:0]	I	DS5 through DS6 input after ANC packet insertion. The number of these ports that are active depends on the number of data streams being interleaved.
tx_use_anc_in	I	When Low, the data streams out of the ST 352 packet insertion function are routed internally to the TX output channels. When High, the TX output channels accept data streams from the tx_ds[16:1]_anc_in ports. This is the same as the tx_use_dsin port of the old core.

Table 2-3: **Transceiver Ports *(Cont'd)***

Port Name	I/O	Description
tx_txdata[19/39:0]	O	Connect to the TXDATA port of the serial transceiver. The width of this port is control by the DATA_WIDTH parameter and is either 20 or 40-bit.
tx_ce_align_err	O	This output indicates problems with the 5/6/5/6 clock cycle cadence of the tx_sd_ce input in SD-SDI mode. In SD-SDI mode, the tx_sd_ce signal must follow a regular 5/6/5/6 clock cycle cadence. If it does not, the SD-SDI serial stream is formed incorrectly. The tx_ce_align_err output goes High if the cadence is incorrect. This port is only valid in SD-SDI mode and only if tx_sd_bitrep_bypass is Low.

SD-SDI EDH Error Detection

The receiver optionally contains an EDH processor that checks the SD-SDI signal for errors. This EDH processor does not update EDH packets in the SD-SDI stream. It simply reports any errors found and also captures the error flags from each EDH packet. The receiver EDH processor can be set to either include in the core or not using the Vivado® IDE. The EDH processor has a 16-bit counter which counts the number of fields that have errors. The current error count is output on the `rx_edh_errcnt` port. The counter can be cleared by asserting `rx_edh_clr_errcnt` High. You can specify which types of errors are counted using the `rx_edh_errcnt_en` input port. This port has 16 unary bits which enable and disable 16 different error types. Any bit that is High enables the corresponding error. When this type of error is detected, the error counter increments. Any bit that is Low disables the corresponding error. Table 2-4 shows the encoding of the bits on the `rx_edh_errcnt_en` port.

Table 2-4: **Bit Encoding for Error Type**

Bit Number	Error Type
0	ANC EDH
1	ANC EDA
2	ANC IDH
3	ANC IDA
4	ANC UES
5	FF EDH
6	FF EDA
7	FF IDH
8	FF IDA
9	FF UES
10	AP EDH
11	AP EDA
12	AP IDH

Table 2-4: **Bit Encoding for Error Type *(Cont'd)***

Bit Number	Error Type
13	AP IDA
14	AP UES
15	EDH packet checksum-error

The ANC error conditions occur when there are errors in the ancillary data packets. The FF error conditions occur when there are errors in the full field. And, the AP error conditions occur when there are errors in the active portion of the picture. The EDH packet checksum error indicates a checksum error found within the EDH packet itself.

The ANC, FF, and AP error condition sets each have five individual error flags, described below. All flags are asserted High to indicate an error condition. For a complete description of the EDH, EDA, IDH, IDA, and UES error flags in the EDH packet, refer to the SMPTE RP 165 document available from SMPTE.

- EDH error: This error condition occurs when the EDH processor detects a CRC error (checksum error for ANC packets) in a field.
- EDA error: This error condition occurs when the EDA or EDH flags of the received EDH packet are asserted.
- IDH error: This error condition is currently not supported.
- IDA error: This error condition occurs when the IDA or IDH flags of the received EDH packet are asserted.
- UES error: This error condition occurs when the UES flag in the received EDH packet is asserted.

The actively computed EDH errors for the ANC, AP, and FF are also output on the `rx_edh_anc`, `rx_edh_ap`, and `rx_edh_ff` ports, respectively. Thus, the `rx_edh_anc` port is asserted whenever a checksum error is detected in an ancillary data packet. The `rx_edh_ap` port is asserted when the calculated active picture CRC does not match the AP CRC in the EDH packet. And, the `rx_edh_ff` port is asserted when the calculated full field CRC does not match the FF CRC in the EDH packet.

The EDH processor also outputs the ANC, AP, and FF flags from the EDH packet on the `rx_edh_anc_flags`, `rx_edh_ap_flags`, and `rx_edh_ff_flags` ports, respectively. These output ports are exact copies of the flags found in the last received EDH packet. This means they differ from the actively computed error conditions shown above. For example, the EDH flag (bit 0) of the `rx_edh_ap_flags` port indicates that the AP EDH flag is set in the last received EDH packet. But, the `rx_edh_ap` port indicates that the active picture CRC calculated locally by the EDH processor does not match the AP CRC value in the EDH packet. The `rx_edh_anc_flags`, `rx_edh_ap_flags`, and `rx_edh_ff_flags` ports are each 5-bit wide and are encoded as shown in Table 2-5.

Table 2-5: **Bit Encoding for ANC, AP, and FF Error Type**

Bit Number	Error Type
0	EDH
1	EDA
2	IDH
3	IDA
4	UES

The EDH processor also produces four error flags related to the format and contents of the EDH packet. These error flags are output on the `rx_edh_packet_flags` port. The encoding of this port is shown in Table 2-6.

Table 2-6: **Bit Encoding for Packet Error Type**

Bit #	Error
0	EDH packet is missing
1	Parity error in user data words of EDH packet
2	Checksum error in EDH packet
3	Format error in EDH packet – such as invalid data count

The TX EDH processor is configured, by default, to always create new EDH packets and insert them whenever the `tx_insert_edh` input of the top level of the SMPTE UHD-SDI core is High. It generates the new packets without regard to any of the flags located in existing EDH packets. The TX EDH processor does have the capability to inspect incoming EDH packets and change EDH and IDH flags to EDA and IDA flags in the EDH packet. This capability can be enabled by wiring the `receive_mode` input port of the `v_smpte_uhdsdi_edh_processor` High. By default, the `receive_mode` port is wired Low where the EDH processor is instantiated in the `v_smpte_uhdsdi_tx` module. If the `v_smpte_uhdsdi_tx` file is edited to wire the `receive_mode` port of the EDH processor High, the EDH processor then inspects any existing EDH packets and promote an EDH flag in the packet to an EDA flag and promote an IDH flag to an IDA flag.

The RX EDH processor also can be used to change EDH flags to EDA flags and IDH flags to IDA flags. If the RX EDH processor is included in the design, its `receive_mode` port is wired High. However, the SMPTE UHD-SDI RX does not use the output video stream of the EDH processor which contains the modified EDH packet. If it is necessary for the RX EDH processor to do EDH/IDH flag promotion, then the `v_smpte_uhdsdi_rx` module needs to be edited to use the data stream on the `vid_out` port of the EDH processor as the SD-SDI received video stream.

Transport Format Detection

The SDI receiver has transport format detector function. This function examines the timing of the video signals in the SDI data streams and determines which video format is being received. The operation of this function is independent of and not dependent on the ST 352 payload ID packets. This function determines the transport format, not the picture format. Usually these are the same, but not always. For example, when 1080p 60 Hz video is transported on 3G-SDI level B-DL, the video transport is actually 1080i 60 Hz – the transport is interlaced, but the picture is progressive.

The transport format detection function makes it determination of the transport format purely by looking at the video timing. It cannot distinguish between video formats that have exactly the same timing. For example, progressive segmented frame (PsF) video formats are intentionally designed to with identical timing to corresponding interlaced formats and cannot be distinguished from interlaced formats by examining the timing. The transport format detection function reports PsF video formats as interlaced formats (`rx_t_scan` is Low). The user application must examine the ST 352 payload ID packets to discover whether the actual video format is PsF or interlaced.

6G-SDI and 12G-SDI mappings defined by SMPTE always subdivide the images into multiple sub-images. Each sub-image is formatted as a regular 1080p image. The transport format detection function examines the timing of data stream 1 (ds1) only. Thus, it reports the video transport of 6G-SDI and 12G-SDI signals as 1080p signals (`rx_t_family` = 0000 and `rx_t_scan` = 1).

The `rx_t_family` output port provides a 4-bit code that indicates the matching video format family. The encoding of this output port is shown in Table 2-7. The transport detection function also determines whether or not the transport is interlaced or progressive and reports this on the `rx_t_scan` output port.

Table 2-7: **Video Format Encoding**

Code	Transport Video Format	Active Pixels
0000	SMPTE ST 274	1920 x 1080
0001	SMPTE ST 296	1280 x 720
0010	SMPTE ST 2048-2	2048 x 1080
0011	SMPTE ST 295	1920 x 1080
1000	NTSC	720 x 486
1001	PAL	720 x 576
1111	Unknown	
Others	Reserved	

The transport detector also determines the frame rate of the transport signal. This feature is dependent on the `rx_bit_rate` input in HD-SDI and 3G-SDI modes to distinguish between integer and non-integer frame rates. The `rx_t_rate` port indicates the frame rate of the transport signal as shown in Table 2-8 The encoding of the frame rate matches the encoding used in the picture rate field of SMPTE ST 352 payload ID packets. However, `rx_t_rate` shows the transport frame rate, not the picture rate. Also, the `rx_t_rate` port value is always the frame rate, even for interlaced transports.

Table 2-8: **Frame Rate Encoding**

Code	Frame Rate (Hz)
0000	None
0010	23.98
0011	24
0100	47.95
0101	25
0110	29.97
0111	30
1000	48
1001	50
1010	59.94
1011	60
Others	Reserved

Core Parameters

The core has three parameters:

- **INCLUDE_RX_EDH_PROCESSOR**: When this parameter is TRUE, the RX section includes an EDH processor for error detection in SD-SDI mode. When this parameter is FALSE, the EDH processor is not included in the RX section.

  Note: EDH is enabled by default in the transmitter.
- **DATA_WIDTH**: This parameter specifies the `rx_data_in` and `tx_txdata` port widths. The only permitted values for this parameter are 20 and 40. When only using SD, HD, and 3G modes, the DATA_WIDTH parameter should be set to 20. However, when using 6G or 12G modes, DATA_WIDTH must be set to 40. Note that this is a static parameter, it only specifies the width of these ports. If DATA_WIDTH is 40, the actual data width used changes dynamically based on the current SDI modes of the RX and TX.
- **NUM_RX_CE**: This parameter specifies the number of identical copies of the RX clock enable signal are present on the `rx_ce_out` port. The minimum value permitted is 1, in which case the `rx_ce_out` port is 1 bit wide.

Serial Transceiver PLL Usage

7 Series GTX Devices

The 7 series GTX Quad contains one QPLL that can supply a clock to any RX or TX unit in the quad. In addition, each transceiver has its own CPLL that can supply a clock to the RX and TX of that transceiver only.

There are several rules to keep in mind with any 7 series GTX transceiver application:

- To support 12G-SDI, the MGTAVCC voltage rail of the device must be 1.05 V. this voltage level also supports all other SDI bit rates.
- Only -3 speed grade 7 series device have GTX transceivers rated for 12G-SDI operation.
- Only the QPLL can be used as the clock source for any RX and TX operating at 12G-SDI and the QPLL must be operating in range 2.
- The 7 series GTX RX has not been characterized at 3G-SDI and HD-SDI rates with the QPLL operating in range 2. So, if the QPLL is operating in range 2, any GTX RX must use the CPLL as its clock source when running in 3G-SDI and HD-SDI modes. TX units may use the QPLL to transmit 3G, HD, or SD.
- At 12G-SDI line rates, the GTX CDR only has ±200 ppm tolerance. Thus, the QPLL must have a 148.5 MHz reference clock when the RX and TX units in the quad are running at the 11.88 Gb/s line rate and it must have a 148.5/1.001 MHz reference clock when the RX and TX units in the quad are running at 11.88/1.001 Gb/s line rate.
- Because there is only a single QPLL available in the quad, it is not possible to support both 11.88 Gb/s and 11.88/1.001 Gb/s simultaneously in the quad.

Note: Refer to the respective device data sheet for maximum transceiver line rate support.

Case #1: Maximum Line Rate is 6G-SDI or Lower

First, consider applications where the maximum line rate is 6G-SDI or lower. In such applications, it is common to use only the QPLL as the clock source for all receivers in the GTX quad. The QPLL operates in range 1 and is, therefore, fully characterized as a clock source for the receivers running at 3G, HD, and SD rates. And, because at 6G-SDI and lower rates, the GTX CDR has ±1250 ppm tolerance, the receivers can receive any SDI line rate, both integer frame rate and non-integer frame rate, using the single clock frequency supplied by the QPLL. The transmitters, if they are to support both integer and non-integer frame rates, must have available two different clock frequencies. The most common configuration is show in Figure 2-1 where the QPLL is given a 148.5 MHz reference clock and supplies a clock to all the RX units in the quad. Each CPLL is given a 148.5/1.001 MHz (approximately 148.35 MHz) reference clock. Each TX unit can be switched between the QPLL and the CPLL, depending on which rate they are transmitting.

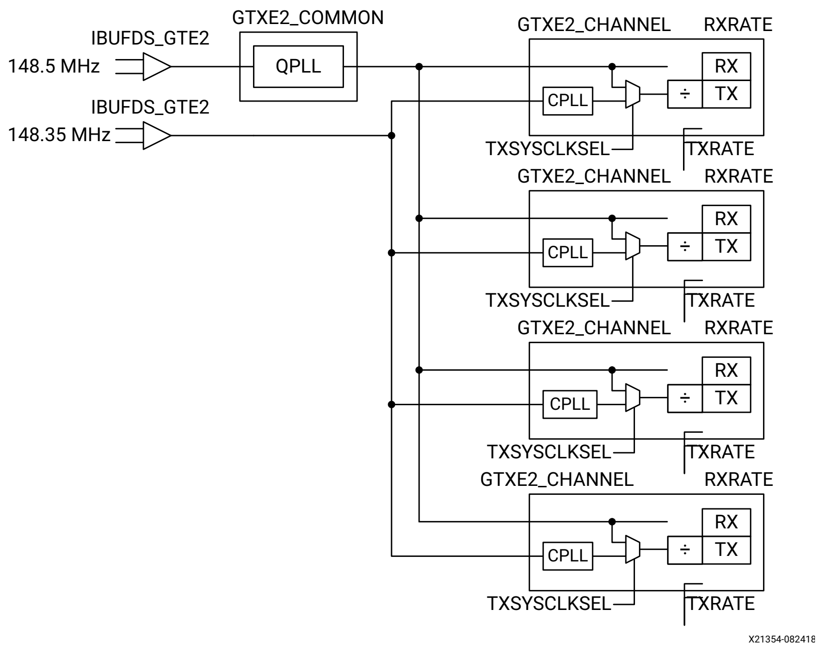


Figure 2-1: **GTX PLL Configuration #1**

Each RX and TX unit has its own PLL clock divider that can divide the serial clock from the PLL by 1, 2, 4, or 8. When the maximum line rate supported is 6G-SDI, the QPLL has a frequency of 5.94 GHz and the CPLL has a frequency of 2.97/1.001 GHz. The QPLL output is divided by 2 before being distributed to the `GTXE2_CHANNEL` transceivers, thus the two clocks available to the `GTXE2_CHANNEL` are 2.97 GHz and 2.97/1.001 GHz. At the output of the clock dividers, the clock frequency must be one half the line rate. For example, for 3G-SDI operation at 2.97 Gb/s, a clock frequency of 1.485 GHz is required after the clock divider. Table 2-9 shows the divide values that must be used in each RX and TX to run at the various SDI line rates. SD-SDI always runs at the same line rate as 3G-SDI and uses 11X oversampling techniques. When the maximum line rate supported by the application is 6G-SDI, then the divider values shown in the CPLL and QPLL (Range 1) columns are used.

Table 2-9: **Clock Dividers for SDI**

SDI Standard	CPLL	QPLL (Range 1)	QPLL (Range 2)
SD-SDI	2	2	4 (TX only)
HD-SDI	4	4	8 (TX only)
3G-SDI	2	2	4 (TX only)
6G-SDI	1	1	2
12G-SDI	N/A	N/A	1

Case #2: Support for a Single 12G-SDI Line Rate

The next case occurs when 12G-SDI and all lower rates must be supported and when only one 12G-SDI line rate, but not both, needs to be supported.

In this case, shown in Figure 2-2, the QPLL is given a single reference clock frequency, which can be either 148.5 MHz or 148.5/1.001 MHz, depending on which 12G-SDI line rate is to be supported (148.5 MHz for 11.88 Gb/s or 148.5/1.001 MHz for 11.88/1.001 Gb/s). In the example shown in the figure, the quad is only supports 11.88 Gb/s, so the reference clock frequency is 148.5 MHz. The QPLL operates in range 2 at 11.88 GHz and provides a 5.94 GHz clock to each RX and TX unit in the quad.

The CPLLs are all given a 148.5/1.001 MHz reference clock and operate at 2.97/1.001 GHz, providing a clock of that frequency to the RX and TX of the associated transceiver.

Any RX or TX in the quad running at 11.88 Gb/s must use the QPLL clock as its serial clock source and must have is PLL divider set to divide by 1. At 6G-SDI rates, the RX can use either the QPLL or the CPLL as long as the correct divider value is used (divide by 1 when using the CPLL and divide by 2 when using the QPLL). At 3G-SDI rates and lower, the RX must use the CPLL. The TX units uses the QPLL when transmitting integer frame rate SDI line rates and the CPLL when transmitting non-integer frame rate SDI lines rates. In this scenario, the only limitation is that only the 11.88 Gb/s 12G-SDI line rate is supported. It is not possible to transmit or receive at 11.88/1.001 Gb/s given the arrangement of the reference clocks.

If the QPLL is given the 148.5/1.001 MHz reference clock and the CPLL is given the 148.5 MHz reference clock, then the application can support the 11.88/1.001 Gb/s line rate, but not the 11.88 Gb/s line rate. All slower line rates can be supported.

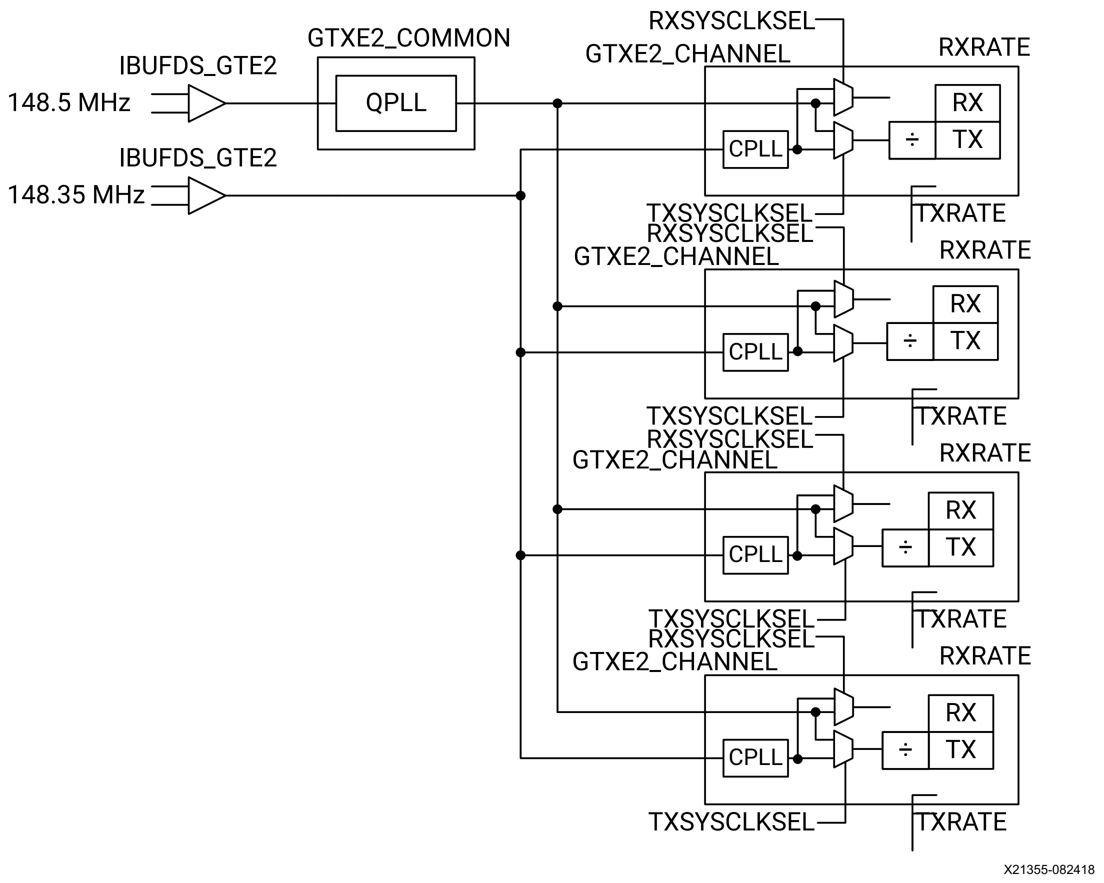


Figure 2-2: **GTX PLL Configuration #2**

Case #3: Dynamic Switching Between 12G-SDI Line Rates

If dynamic switching between the two 12G-SDI line rates is an absolute requirement, then things get much more complicated. Dynamic switching is possible, but has significant implications.

To support dynamic switching between the two 12G-SDI line rates, the reference clock to the QPLL must be dynamically switched between 148.5 MHz and 148.5/1.001 MHz. Anytime the reference clock frequency to the QPLL is dynamically switched, the QPLL must be reset. Furthermore, the only 12G-SDI rate supported by the entire quad at any point in time is dictated by which reference clock frequency is given to the QPLL. Thus, it is possible to switch the entire quad between 11.88 Gb/s and 11.88/1.001 Gb/s, but all RX and TX units in the quad running in 12G-SDI mode always runs at the that one 12G-SDI line rate dictated by the QPLL reference clock. It is not possible to have some units in the quad running at 11.88 Gb/s and others at 11.88/1.001 Gb/s.

Any RX unit that is using the QPLL as the serial clock source for 6G-SDI is upset when the reference clock to the QPLL is dynamically switched between reference clock frequencies and reset. Any TX unit in the quad that is using the QPLL as the serial clock source not only experiences an upset when the QPLL is reset, but also experiences a 1000 ppm shift in line rate as a result of the QPLL changing reference clock frequencies.

Thus, for most applications, support for dynamic switching between the two 12G-SDI line rates makes it difficult to implement multiple SDI interfaces in the same GTX quad. The application can never support RX or TX of both 12G-SDI line rates in the same GTX quad simultaneously. And, any switch between 12G-SDI line rates impacts any RX or TX unit in the quad using the QPLL.

There are several possible use cases where dynamic switching between 12G-SDI line rates may be practical. One such use case is shown in Figure 2-3. In this use case, each transceiver is RX only or TX only. The top two transceivers in the figure are RX only and the bottom two transceivers are TX only. When operating at 6G-SDI line rates and below, an RX or TX unit always uses the transceiver CPLL. The CPLL is dynamically switched between the two reference clocks as required using the `CPLLREFCLKSEL` port. Any RX or TX unit running at 12G-SDI rates, must use the QPLL as the clock source. The QPLL can be dynamically switched between the two reference clock frequencies as required. But, all units operating at 12G-SDI rates switches between the two 12G-SDI line rates simultaneously as the QPLL dynamically switches between reference clock frequencies. Any mix of RX and TX units in the quad can be supported this way; it does not have to be two RXs and two TXs.

The reason for limiting each transceiver to just RX or just TX and not both is to make it easier to use the CPLL. Because the CPLL is needed for both RX and TX, sharing the CPLL is somewhat difficult. Whenever the CPLL is dynamically switched between reference clock sources, it affects both the RX and the TX unit if they were both active and using the clock from the CPLL. If, however, the application does not care that both the RX and TX are affected by dynamically switching the CPLL between the two reference clock frequencies, then it is possible to use a transceiver to transmit and receive at the same time.

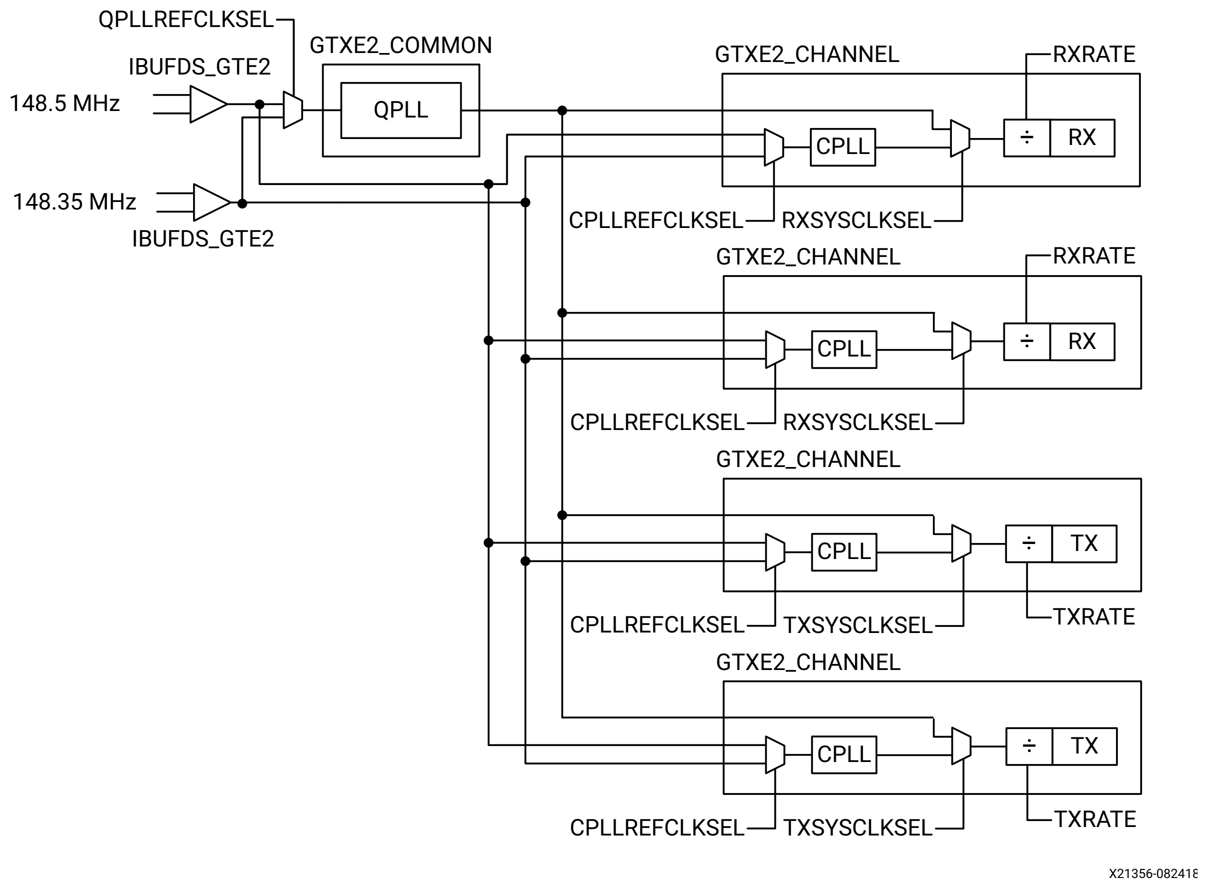


Figure 2-3: **GTX PLL Configuration #3**

UltraScale/UltraScale+ GTH

The UltraScale™/UltraScale+™ GTH Quad provides more flexibility than the 7 series GTX Quad. The UltraScale/UltraScale+ GTH Quad has two QPLLs and four CPLLs, one CPLL for each transceiver. For full flexibility, one QPLL is given a 148.5 MHz reference clock and the other QPLL is given a 148.5/1.001 MHz reference clock as shown in Figure 2-4. The clocks from both QPLLs are routed to each RX and TX unit in the quad. Each RX and TX unit can independently select the clock from either QPLL and then divide that clock by 1, 2, 4, or 8. This allows full independence of all RX units and TX units in the quad to run at any SDI line rate. Both the 11.88 Gb/s and 11.88/1.001 Gb/s line rates can be simultaneously supported in the same quad.

The receivers can use just one QPLL for all line rates except in the case of 12G-SDI, where the receivers must use the correct QPLL for the particular 12G-SDI line rate to be supported.

The TX units must dynamically switch between the two QPLLs based on which line rate they need to transmit. In such a case, the CPLLs are not needed.

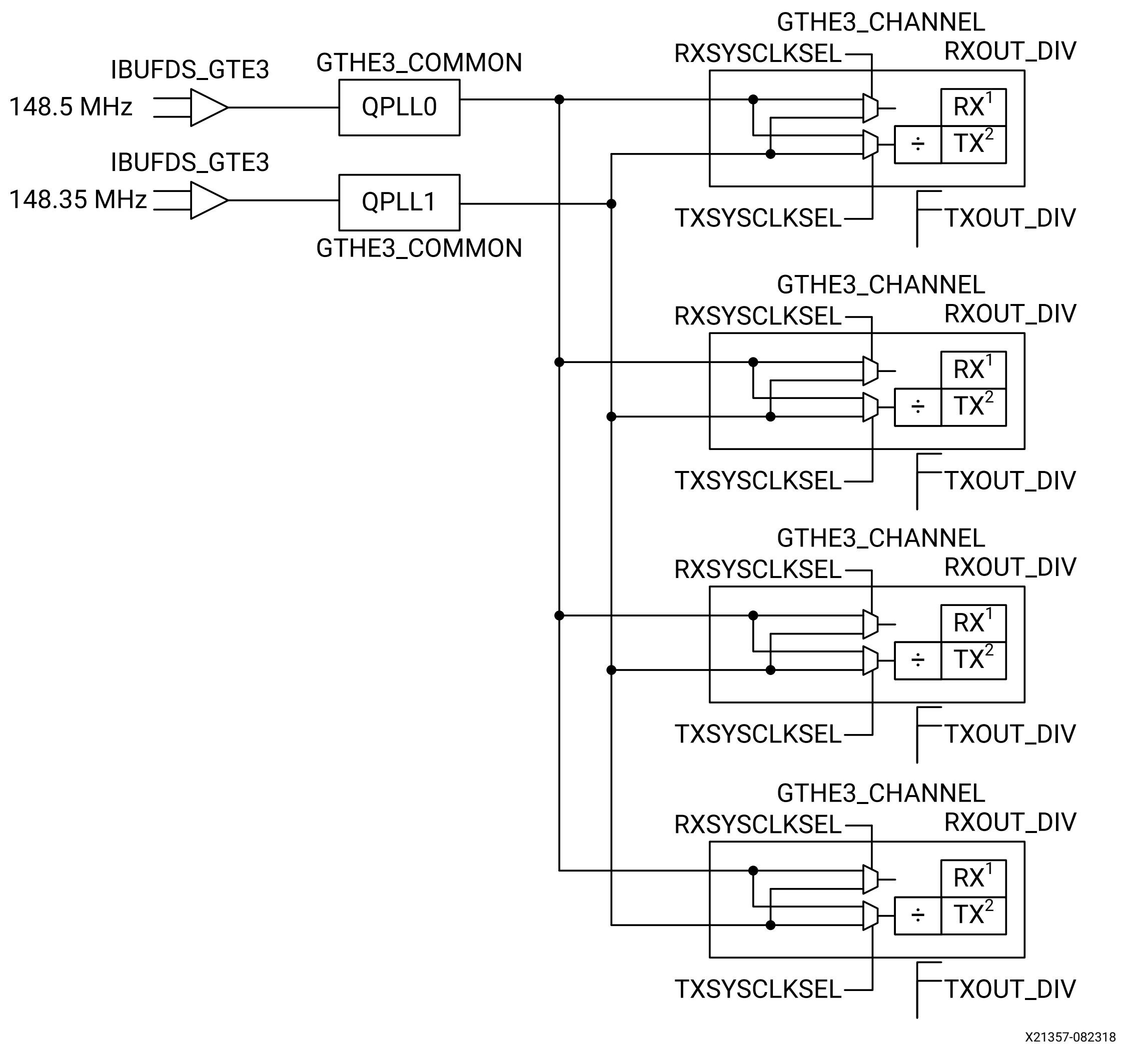


Figure 2-4: **UltraScale/UltraScale+ GTH PLL Configuration for SMPTE UHD-SDI Core**

IMPORTANT: *Xilinx recommends using a CPLL-QPLL combination with CPLL for TX and QPLL0/1 for RX, where both transmit and receive 12G-SDI integer and fractional modes using the same transceiver for Ultrascale+ devices as shown in Figure 2-5.*

For Ultrascale+ devices, when using QPLL0 and QPLL1 for the 12G-SDI integer and fraction (1/1.001) rate, switching between rates on the SDI-RX can introduce a glitch on the clock which causes CRC errors on the TX channel. CRC errors do not occur in SDI-SDI/HD-SDI/3-G SDI/6-GSDI integer/fractional modes with a QPLL0 and QPLL1 clocking combination. For more details, see Answer Record 72254 and 72449.

Therefore, it is not recommended to use this clocking configuration when both the transmit and receive 12G-SDI integer and fractional modes use the same transceiver. The integer and fractional rates for TX can be selected using a CPLL reference clock input selection with 297 MHz and 296.7 MHz respectively. This CPLL/QPLL clocking combination is not feasible with -1 speed grade devices because the CPLL does not support the 12G-SDI line rate. You need to select a Ultrascale+ GTY/GTH -2 speed grade or faster rate with >0.85V.

Refer to the *SMPTE UHDI-SDI Receiver Subsystem v2.0 Product Guide* (PG290) [Ref 10] for further explanation and an audio video loopback example design implemented using a CPLL/QPLL combination targeting a KCU116 board.

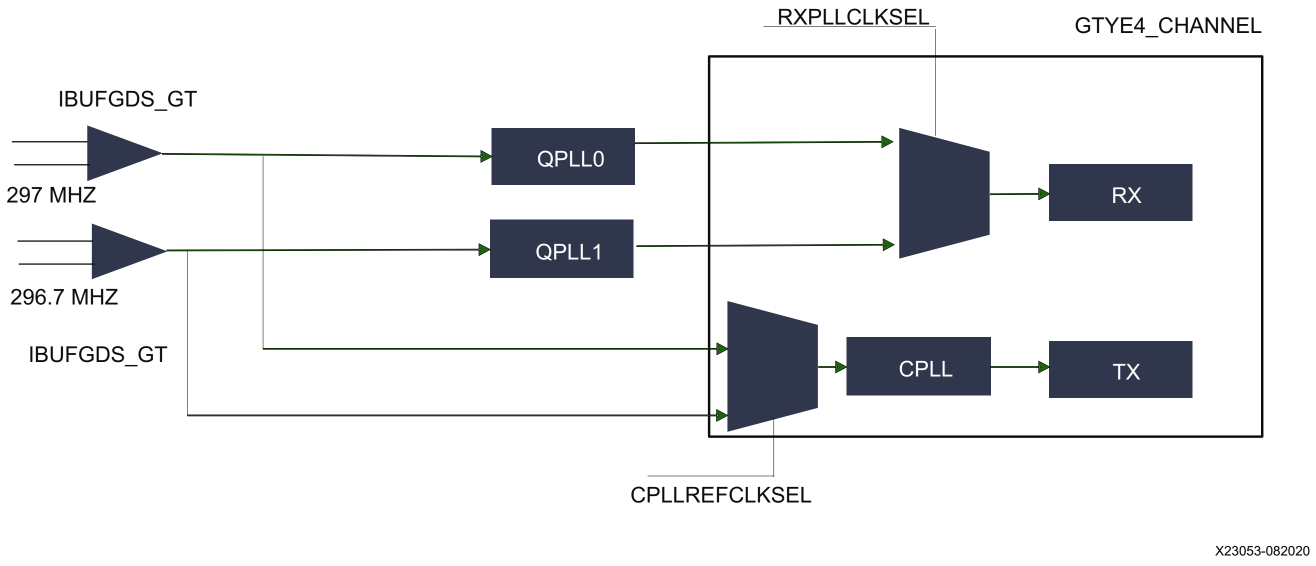


Figure 2-5: **KCU116 Audio Video Loopback Example Design GT Clocking Architecture**

Designing with the Core

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

General Design Guidelines

SMPTE UHD-SDI Receiver Operations

The SMPTE UHD-SDI receiver supports the following features:

- Error checking using the per line CRC values in all modes except SD-SDI. Separate CRC error indicators are available for each of the 16 output elementary data streams.
- Error checking using the RP 165 EDH packets in SD-SDI mode.
- Captures and outputs the line numbers embedded in the signals of all standards except SD-SDI. Separate line number outputs are available for each of the 16 elementary data streams.
- Captures and outputs the ST 352 payload ID packets for all standards. Separate ST 352 packets can be captured simultaneously from eight different elementary data streams. This is the maximum number of data streams that can have ST 352 packets because only one data stream of each data stream pair contains ST 352 packets.
- Optionally, ST 352 payload IP packets can be inserted into the C-stream
- ST 352 payload IP packets received on the C-stream are reported
- All embedded ancillary data is remains present in the data streams output by the RX and can be captured by the use application.
- Automatic SDI mode detection - the RX determines the incoming line rate and configure itself to run at that rate. (This feature can be disabled and the RX forced into any SDI mode.)
- Transport video format detection - the RX determines and reports the transport format used to carry the data streams. This feature is independent of the ST 352 packets that may or may not be embedded in the data streams.
- In 3G-SDI mode, the receiver automatically determines whether the signal is level A or level B without referring to ST 352 information.

- In 6G and 12G-SDI modes, the receiver automatically determines the number of elementary data streams in the multiplexed SDI stream (4, 8, or 16) without referring to ST 352 information.
- The receiver outputs EAV, SAV, and TRS timing signals.
- In 6G-SDI and 12G-SDI modes, the receiver restores the 3FF and 000 values that are modified by the transmitter to meet the sync bit insertion (run length mitigation) feature of these standards.

SMPTE UHD-SDI Transceiver Operations

The transmitter section has the following features:

- Optional generation and insertion of CRC values for error detection on each line for all modes except SD-SDI.
- Optional generation and insertion of RP 165 EDH packets for error detection in SD-SDI mode.
- Optional formatting and insertion of line numbers into all data streams immediate after the EAV in all modes except SD-SDI.
- Optional generation and insertion of ST 352 payload ID packets on up to 8 different elementary data streams.
- Supports 1, 2, 4, 8, or 16 data stream multiplexing.
- Automatic sync bit insertion (run length mitigation) for 6G-SDI and 12G-SDI modes. While this sync bit insertion is mandatory in the 6G and 12G-SDI standards, it can be disabled in the SMPTE UHD-SDI transmitter via a control port to support fielded equipment designed before the standards were released that do not support the sync bit insertion feature.
- Hooks are provided to allow the user application to insert ancillary data into the elementary data streams after the SMPTE UHD-SDI transmitter has inserted ST 352 packets, but before any other processing of the data streams occurs.

Clocking

Serial Transceiver RX Reference Clocks

For SD, HD, 3G, and 6G-SDI modes, only a single reference clock frequency is required to receive any supported SDI bit rate. The CDR in the serial transceiver is configured for ±1250 ppm tolerance allowing both bit rates of each SDI standard (such as the HD-SDI bit rates of 1.485 Gb/s and 1.485/1.001 Gb/s, which differ by exactly 1000 ppm) to be received using a single reference clock frequency. Typically, this reference clock frequency is 148.5 MHz.

For 12G-SDI, the serial transceiver CDR does not support ±1250 ppm tolerance. The CDR tolerance is reduced to ±200 ppm at 12G-SDI line rates. This requires different reference clock frequencies to be used to receive 11.88 Gb/s and 11.88/1.001 Gb/s. Typically, the two reference clock frequencies used are 148.5 MHz (to receive 11.88 Gb/s) and 148.5/1.001 MHz (to receive 11.88/1.001 Gb/s).

At 12G-SDI line rates, only the QPLL can be used as the RX clock source. The CPLL cannot be used for 12G-SDI. In the 7 series GTX transceiver quad there is only a single QPLL. Therefore, to switch any receiver in the quad between the 11.88 Gb/s and 11.88/1.001 Gb/s line rates, the reference clock frequency to the QPLL must be dynamically switched between 148.5 MHz and 148.5/1.001 MHz. The implication of this is that it is not possible to have one RX in the GTX quad receiving 11.88 Gb/s while another RX in the same quad is receiving 11.88/1.001 Gb/s.

The UltraScale™/UltraScale+™ GTH transceiver quad provides more flexibility for 12G-SDI because it has two QPLLs. By supplying one QPLL with a 148.5 MHz reference clock and the other QPLL with a 148.5/1.001 MHz reference clock and, on a per RX basis, selecting which QPLL is used to clock the RX, it is possible to have some receivers in the UltraScale/UltraScale+ GTH quad receiving 11.88 Gb/s while others are receiving 11.88/1.001 Gb/s.

Serial Transceiver TX Reference Clocks

For the transmitter, without using PICXO, different reference clock frequencies are required to transmit the 1000/1000 line rates and the 1000/1001 rates. If both 1000/1000 and 1000/1001 families of rates are to be supported simultaneously, then reference clock frequencies of both 148.5MHz and 148.5/1.001MHz must be available to the PLLs in the quad.

At 12G-SDI line rates, the transmitters can only be clocked by the QPLL. So with 7 series GTX transceivers, as with the RX, it is not possible to have some transmitters running at 11.88 Gb/s and others running at 11.88/1.001 Gb/s in the same quad. In UltraScale/UltraScale+ GTH this restriction does not apply because there are two QPLLs per quad.

Note: Xilinx recommends using a CPLL/QPLL combination with CPLL for TX and QPLL0/1 for RX where both transmit and receive at 12G-SDI integer and fractional modes are using the same transceiver for Ultrascale+ devices as shown in Figure 2-5.

Data Path Clocks

The serial transceiver transceivers output clocks from both the receiver and the transmitter called `RXOUTCLK` and `TXOUTCLK`. `RXOUTCLK` is a word-rate recovered clock generated by the CDR in the serial transceiver. `TXOUTCLK` is a word-rate clock generated by the serial transceiver TX.

Typically, RXOUTCLK and TXOUTCLK are routed through global clock buffers (BUFG). After being buffered by a BUFG the RXOUTCLK drives the RXUSRCLK and RXUSCRCLK2 ports of the serial transceiver and the rx_clk port of the SMPTE UHD-SDI core. Likewise, the global version of TXOUTCLK drives the TXUSRCLK and TXUSRCLK2 ports of the serial transceiver and the tx_clk port of the SMPTE UHD-SDI core.

SD-SDI Mode

The 270 Mb/s SD-SDI line rate is too slow to be directly supported by the serial transceiver transmitters and receivers. To receive SD-SDI, the line rate of the serial transceiver RX is set to 2.97 Gb/s and it over-samples the incoming SD-SDI signal by a factor of 11. A data recovery unit, called the NI-DRU, recovers the actual data from the oversampled data. The NI-DRU generates a data strobe which runs at a nominal rate of 27 MHz and is asserted with a nominal cadence of 5/6/5/6 RXOUTCLK clock cycles. This data strobe is, therefore, asserted, on average, once every 5.5 RXOUTCLK cycles. The RXOUTCLK frequency is 148.5 MHz and 148.5 MHz / 5.5 = 27 MHz. The NI-DRU data strobe is output on the rx_ce_out port when the receiver is in SD-SDI mode. When rx_ce_out is High, the data on the output data stream is valid.

The NI-DRU in the SMPTE UHD-SDI receiver works identically to the NI-DRU used with the older SMPTE UHD-SDI core. For more details about the operation of the NI-DRU and the operation of the SMPTE UHD-SDI RX in SD-SDI mode, see the section entitled SD-SDI Considerations: Receiving SD-SDI in *Implementing SMPTE SDI Interfaces with Kintex-7 GTX Transceivers* (XAPP592) [Ref 8].

It is important to note that the 5/6/5/6 cadence of rx_ce_out can sometimes vary. For example, occasionally, it may be 5/5/5/6 or 5/6/6/6 or other patterns. This is occurs because of jitter and because the RX reference clock is not synchronous with the incoming SD-SDI signal. This is described in more detail in *Implementing SMPTE SDI Interfaces with Kintex-7 GTX Transceivers* (XAPP592) [Ref 8].

Figure 3-1 is a timing diagram for receiver when running in SD-SDI mode. It shows the most important timing and data signals. In this example drawing, rx_ce_out clock enable output is shown with its typical 5/6/5/6 cycle cadence of the 148.5 MHz rx_usrclk. The received data stream is output on the rx_ds1 port and be captured by downstream modules only when rx_ce_out is High. The figure shows the occurrence of the EAV timing signal. The rx_trs timing output is asserted during all four words of the EAV sequence and the rx_eav output is asserted only during the final word of the EAV sequence.

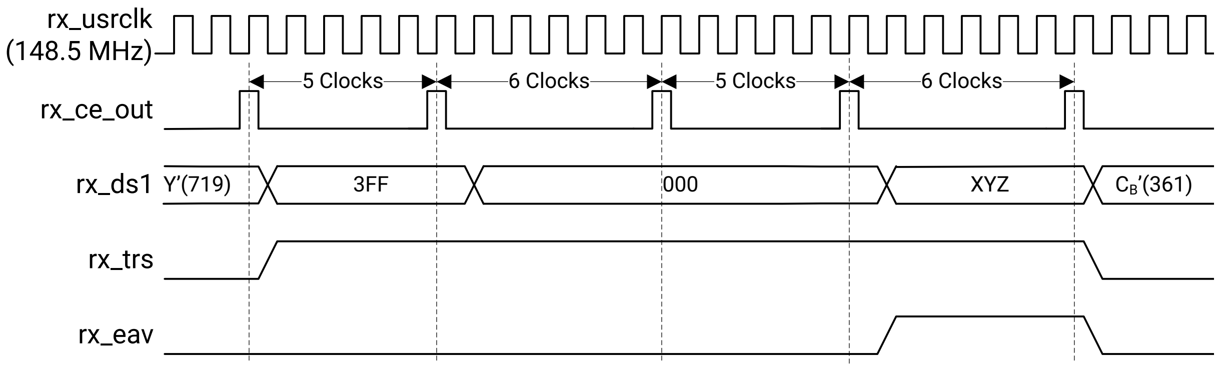


Figure 3-1: **SD-SDI RX Timing Diagram**

For the TX in SD-SDI mode, the application must provide a clock enable signal to the TX on its `tx_sd_ce` and `tx_edh_ce` inputs that always has exactly a 5/6/5/6 cycle `TXOUTCLK` cadence. Whenever the `tx_sd_ce` is High, the SMPTE UHD-SDI TX accepts a 10-bit word on the single active data stream port. The serial transceiver TX runs at 2.97 Gb/s line rate and each encoded SD-SDI bit is replicated and sent by the serial transceiver TX 11 times, producing a 270 Mb/s signal.

Figure 3-2 show the timing of the essential signals for the TX in SD-SDI mode. The `tx_clk` frequency is 148.5 MHz. The `tx_sd_ce` and `tx_edh_ce` clock enable inputs must be asserted with a 5/6/5/6 cycle cadence of `tx_clk`, resulting in a 27 MHz data rate on the `tx_ds1_in` data stream input. The `tx_ce` clock enable input is always High in this mode. In SD-SDI mode, the `tx_line_ch0` input is only used if ST 352 packets are being inserted. The ST 352 packet insertion module requires accurate line numbers on the `tx_line_ch0` port and the four user data words to be inserted into the ST 352 packet on the `tx_st352_data_ch0` port. The data on both of these ports must be valid starting at the last word of the EAV sequence and remaining valid throughout the entire HANC period.

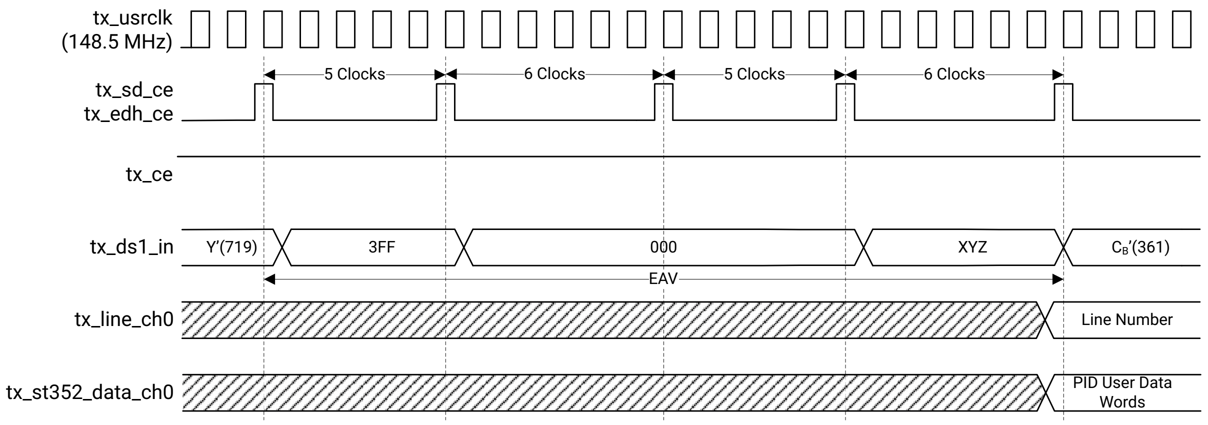


Figure 3-2: **SD-SDI TX Timing Diagram**

HD-SDI Mode

When a serial transceiver RX or TX is running in HD-SDI mode, the RXOUTCLK or TXOUTCLK runs at 74.25 MHz (or 74.25/1.001 MHz). In HD-SDI mode, there are only two 10-bit elementary data streams and the data streams run at the full clock RXOUTCLK or TXOUTCLK frequency (the rx_ce_out or tx_ce clock enable signals are always asserted).

Figure 3-3 is a timing diagram for the receiver in HD-SDI mode. The rx_clk runs at 74.25 MHz (or 74.25/1.001 MHz). Not shown is the rx_ce_out signal which is always High in HD-SDI mode. The Y data stream is output on rx_ds1 and the C data stream on rx_ds2. In Figure 3-3, the EAV sequence is received. The rx_trs output is asserted High during all four words of the EAV sequence and the rx_eav signal is asserted during the fourth word (XYZ word) of the EAV sequence. Captured line numbers are output on rx_ln_ds1, changing to the new value immediately after the second LN word is output and remaining unchanged until the same point of the next line. If a CRC error is detected on one of the data streams, the corresponding CRC error output becomes asserted two clock cycles after the second CRC word is output on the data streams and remains asserted until the same point on the next line.

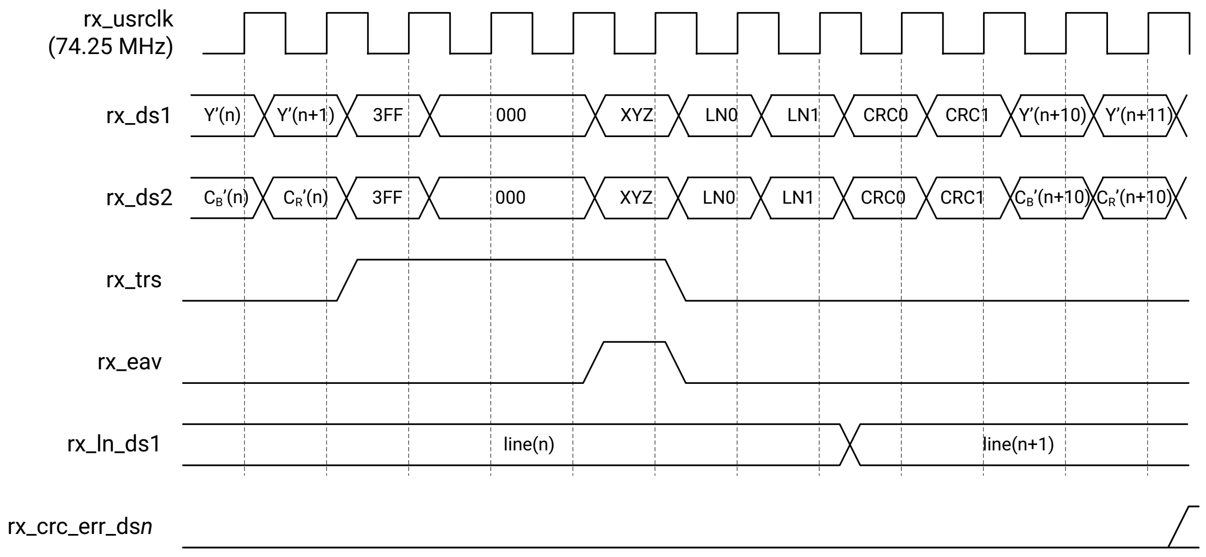


Figure 3-3: **HD-SDI RX Timing Diagram**

Figure 3-4 is a timing diagram for the transmitter in HD-SDI mode. The tx_clk runs at 74.25 MHz (or 74.25/1.001 MHz). The tx_ce input must be asserted High all of the time. The Y data stream is input on the tx_ds1_in port and the C data stream on the tx_ds2_in port. If the data streams do not already have line numbers embedded after the EAV, then valid line numbers must be provided on the tx_line_ch0 input port. Line numbers must also be provided if ST 352 packets are being inserted, even if line numbers are already embedded in the data streams. The line numbers must be stable on the tx_line_ch0 input port by the same clock cycle in which the XYZ word of the EAV enters on the data stream inputs. The line number must remain stable through the duration of the HANC period (until the SAV occurs). If ST 352 packets are to be inserted, then the four user data bytes of the ST 352 packet must be valid on the `tx_sd352_data_ch0` port by the same clock cycle in which the XYZ word of the EAV enters on the data stream inputs and must remain stable through the duration of the HANC period.

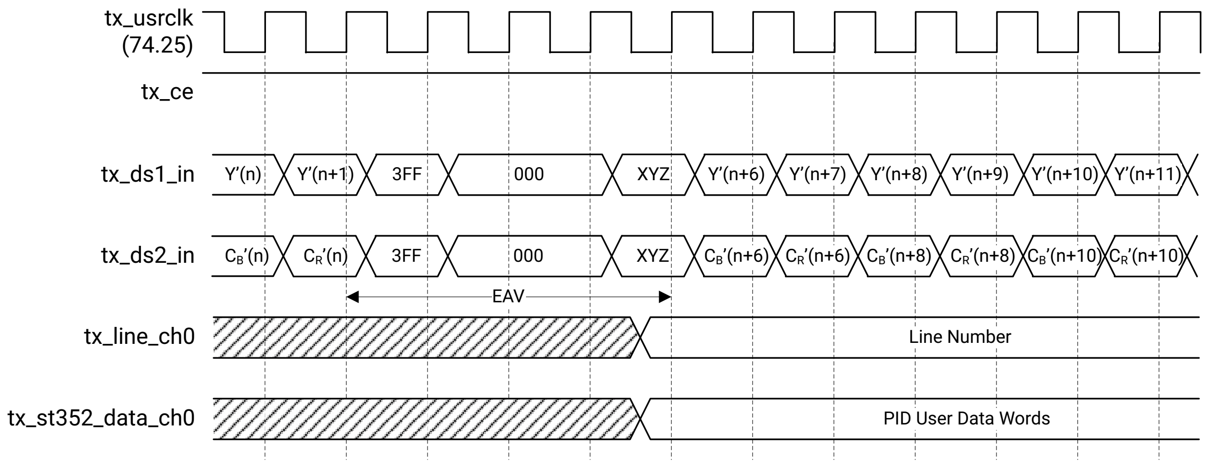


Figure 3-4: **HD-SDI TX Timing Diagram**

3G-SDI Mode

In 3G-SDI mode, the RXOUTCLK/TXOUTCLK frequency is 148.5 MHz (or 148.5/1.001 MHz). In 3G-SDI level A mode, there are two 10-bit elementary data streams and these data streams run at the full RXOUTCLK/TXOUTCLK frequency (the `rx_ce_out` or `tx_ce` clock enable signals are always asserted). However, in level B mode, there are four 10-bit elementary data streams active. In level B mode, the clock enables (`rx_ce_out` and `tx_ce`) are asserted every other clock cycle (50% duty cycle).

Figure 3-5 is the timing diagram for the receiver in 3G-SDI level A mode. In this case, a 1080p 50 or 60 Hz image is being received. The `rx_clk` is running at 148.5 MHz (or 148.5/1.001 MHz). The `rx_ce_out` is always High. The Y data stream is output on `rx_ds1` and the C data stream on `rx_ds2`. The timing of the other signals shown is identical to that described for HD-SDI receiver. Two line number outputs, `rx_ln_ds1` and `rx_ln_ds2`, and two CRC error outputs, `rx_crc_err_ds1` and `rx_crc_err_ds2`, are active.

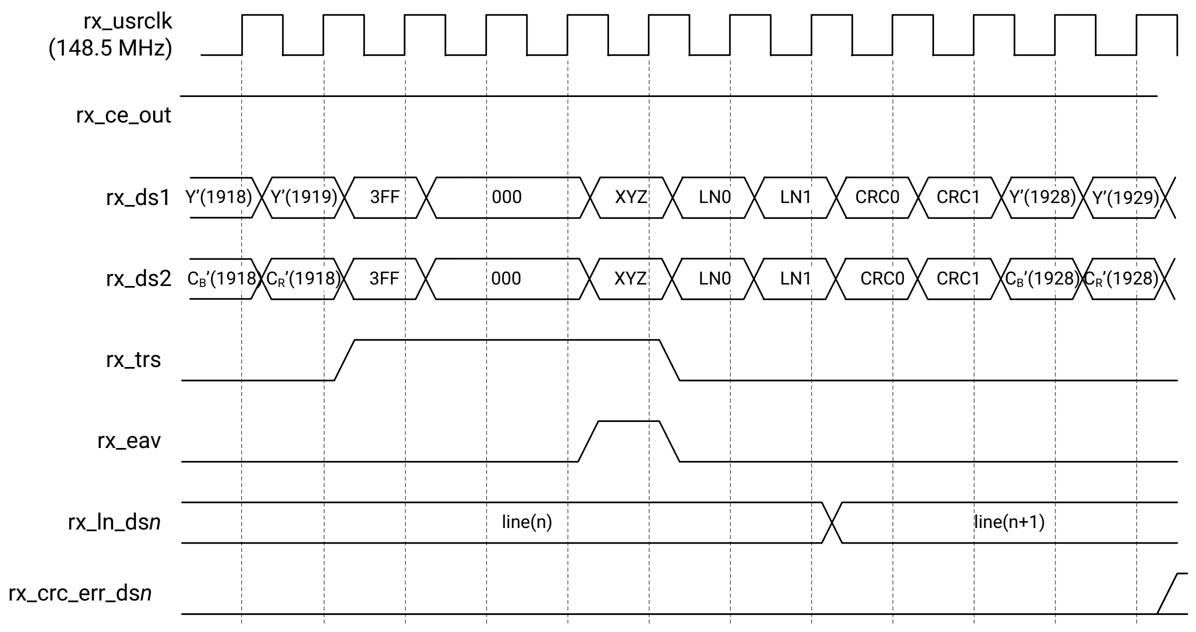


Figure 3-5: **3G-SDI Level A RX Timing Diagram**

Figure 3-6 is a timing diagram for the receiver in 3G-SDI level B mode. The `rx_clk` runs at 148.5 MHz or 148.5/1.001 MHz. The `rx_ce_out` signal is asserted every other clock cycle. Four elementary streams are received, a Y and a C data stream for Link A on `rx_ds1` and `rx_ds2` and a Y and a C data stream for Link B on `rx_ds3` and `rx_ds4`. The `rx_trs` signal is asserted as all four words of the EAV are output on the data stream ports. The `rx_eav` output is asserted when the XYZ word of the EAV is output. Four line number output ports, `rx_ln_ds1` through `rx_ln_ds4`, are active. Line numbers change immediately after the second LN word is output on the data stream ports.

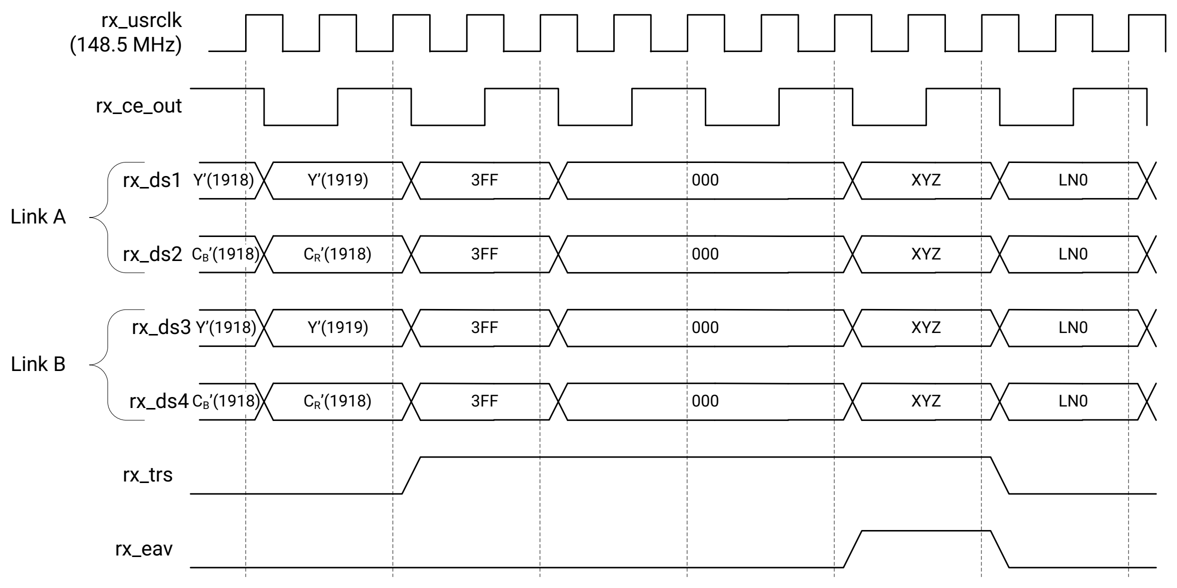


Figure 3-6: **3G-SDI Level B RX Timing Diagram**

Figure 3-7 is a timing diagram for the transmitter in 3G-SDI level A mode. The `tx_clk` runs at 148.5 MHz (or 148.5/1.001 MHz). The `tx_ce` input must be asserted High all of the time. In this example, a 1080p 50 or 60 Hz image is being transmitted. The Y data stream enters on `tx_ds1_in` and the C data stream on `tx_ds2_in`. If the data streams do not already have line numbers embedded after the EAV, then valid line numbers must be provided on the `tx_line_ch0` input port. Line numbers must also be provided if ST 352 packets are being inserted, even if line numbers are already embedded in the data streams. The line numbers must be stable on the `tx_line_ch0` input port by the same clock cycle in which the XYZ word of the EAV enters on the data stream inputs. The line number must remain stable through the duration of the HANC period (until the SAV occurs).

If ST 352 packets are to be inserted, then the four user data bytes of the ST 352 packets for data stream 1 must be valid on the `tx_sd352_data_ch0` port by the same clock cycle in which the XYZ word of the EAV enters on the data stream inputs and must remain stable through the duration of the HANC period. The four user data bytes for the ST 352 packets inserted into data stream 2 must be supplied on `tx_st352_data_ch1`.

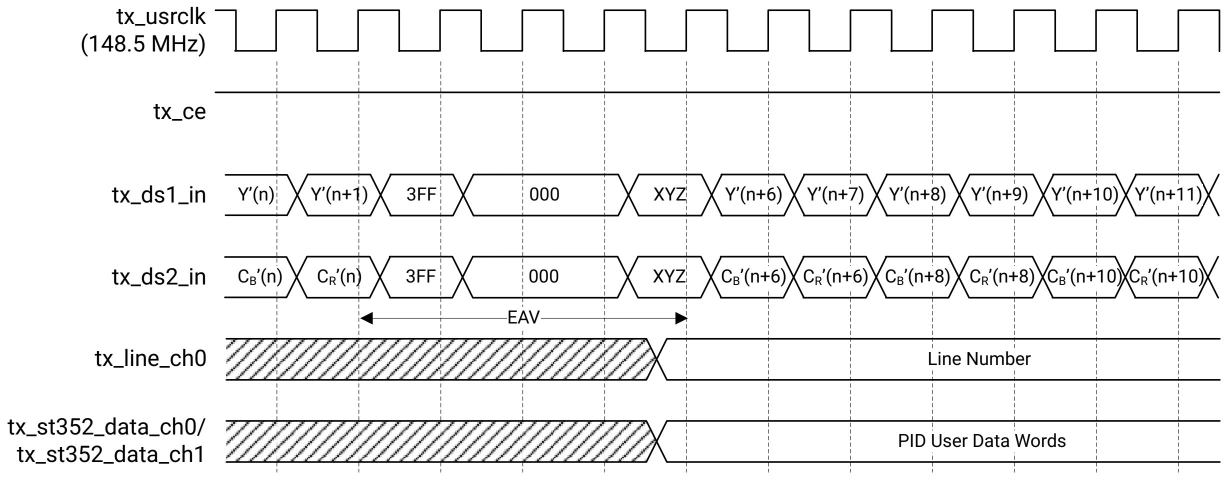


Figure 3-7: **3G-SDI Level A TX Timing Diagram**

Figure 3-8 is a timing diagram for the transmitter in 3G-SDI level B mode. The `tx_clk` runs at 148.5 MHz or 148.5/1.001 MHz. The `tx_ce` input must be asserted every other clock cycle. Four data stream inputs, `tx_ds1_in` through `tx_ds4_in`, are active. Data is captured on these data stream input ports on the rising edge of `tx_clk` when `tx_ce` is High. Line numbers supplied on

- `tx_line_ch0` are inserted into data streams 1 and 2
- `tx_line_ch1` are inserted into data streams 3 and 4.

Line numbers must also be supplied on `tx_line_ch0` and `tx_line_ch1` in order for ST 352 packet insertion to function. If ST 352 packet insertion is enabled (through a port), the user data words for the ST 352 packets inserted into data stream 1 must be supplied on `tx_st352_data_ch0` and the user data words for the ST 352 packets inserted into data stream 3 must be supplied on `tx_st352_data_ch1`. If **Insert ST352 in C-Stream** is enabled, the user data words for the ST 352 packets inserted into

- data stream 2 must be supplied on `tx_st352_data_ch0_c`
- data stream 4 must be supplied on `tx_st352_data_ch1_c`.

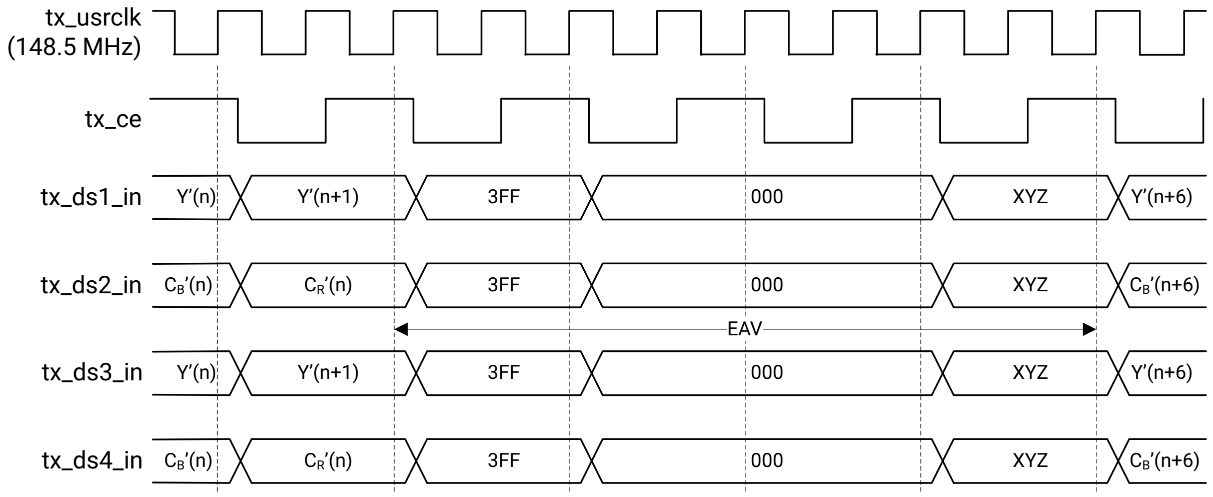


Figure 3-8: **3G-SDI Level B TX Timing Diagram**

6G-SDI Mode

In 6G-SDI mode, the RXOUTCLK/TXOUTCLK frequency is 148.5 MHz (or 148.5/1.001 MHz). Depending on the 6G-SDI mapping mode used, there may be either four or eight active data streams. When four data streams are active, these data streams run at the full RXOUTCLK/TXOUTCLK frequency (`rx_ce_out` and `tx_ce` signals are always High). When eight data streams are active, these data streams run at half the RXOUTCLK/TXOUTCLK frequency and the `rx_ce_out` and `tx_ce` signals are asserted every other clock cycle (50% duty cycle).

Figure 3-9 is a timing diagram of the receiver running in 6G-SDI mode receiving a signal containing four data streams. The `rx_clk` frequency is 148.5 MHz or 148.5/1.001 MHz. The `rx_ce_out` output is always High. The four received data streams are output on `rx_ds1` through `rx_ds4`. The diagram shows the EAV being received and the `rx_trs` and `rx_eav` behave the same as with other SDI standards. Line numbers are captured from all four data streams and output on `rx_ln_ds1` through `rx_ln_ds4`. CRC errors are detected individually for all four data streams and indicated on the `rx_crc_err_ds1` through `rx_crc_err_ds4` outputs.

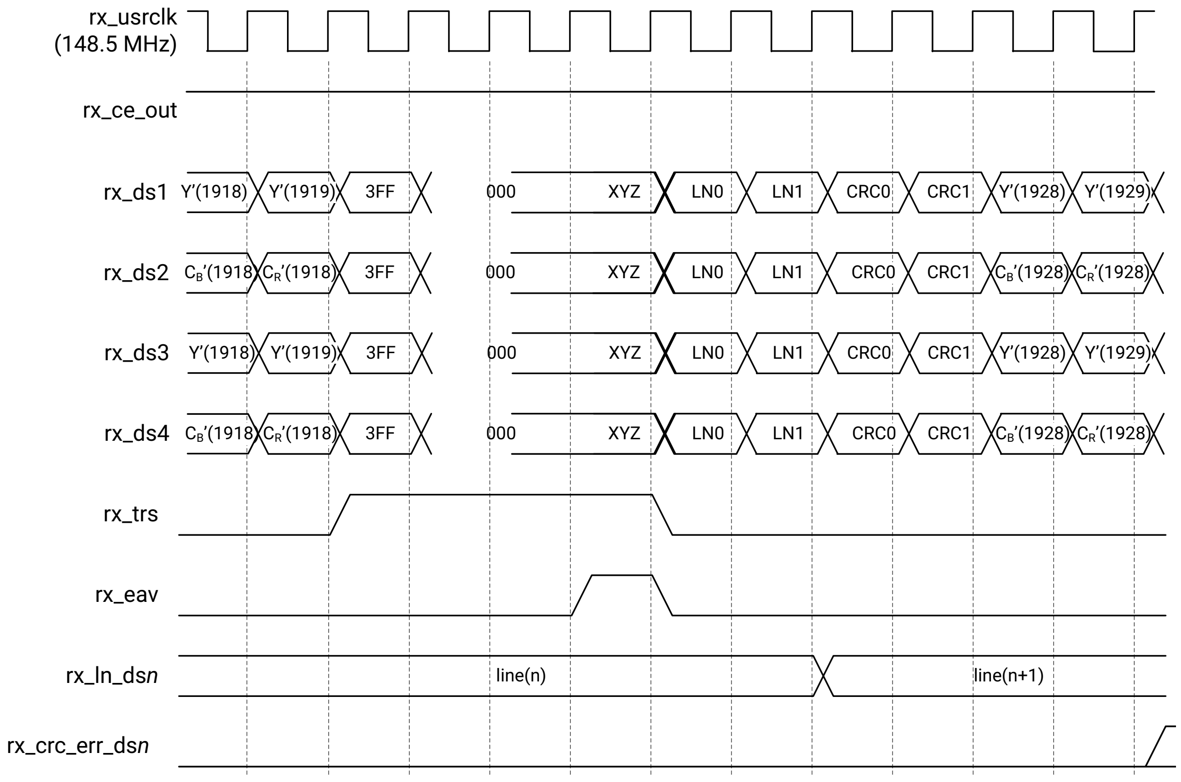


Figure 3-9: **6G-SDI RX 4-way Interleave Timing Diagram**

Figure 3-10 is a timing diagram for the receiver in 6G-SDI mode when receiving a signal containing eight data streams. The `rx_clk` frequency is 148.5 MHz or 148.5/1.001 MHz. The `rx_ce_out` signal is asserted every other clock cycle. The eight received data streams are output on `rx_ds1` through `rx_ds8`. The diagram shows the EAV being received and the `rx_trs` and `rx_eav` behave the same as with other SDI standards. Line numbers are captured from all eight data streams and output on `rx_ln_ds1` through `rx_ln_ds8`. CRC errors are detected individually for all eight data streams and indicated on the `rx_crc_err_ds1` through `rx_crc_err_ds8` outputs.

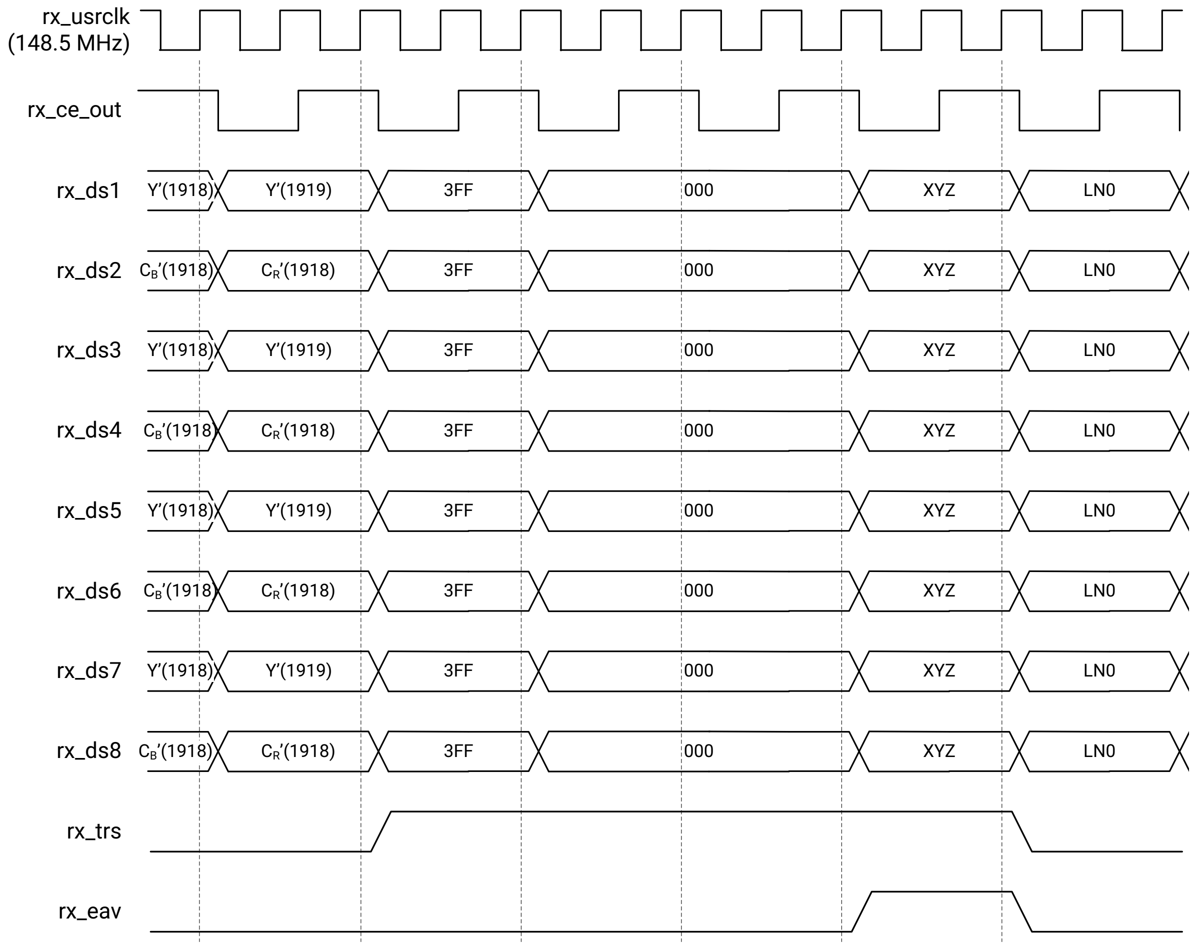

rx_usrclk (148.5 MHz)
rx_ce_out
rx_ds1 Y'(1918) Y'(1919) 3FF 000 XYZ LN0
rx_ds2 CB'(1918) CR'(1918) 3FF 000 XYZ LN0
rx_ds3 Y'(1918) Y'(1919) 3FF 000 XYZ LN0
rx_ds4 CB'(1918) CR'(1918) 3FF 000 XYZ LN0
rx_ds5 Y'(1918) Y'(1919) 3FF 000 XYZ LN0
rx_ds6 CB'(1918) CR'(1918) 3FF 000 XYZ LN0
rx_ds7 Y'(1918) Y'(1919) 3FF 000 XYZ LN0
rx_ds8 CB'(1918) CR'(1918) 3FF 000 XYZ LN0
rx_trs
rx_eav

Figure 3-10: **6G-SDI RX 8-way Interleave Timing Diagram**

Figure 3-11 is a timing diagram for the transmitter in 6G-SDI mode when transmitting a signal containing four data streams. The `tx_clk` frequency is 148.5 MHz or 148.5/1.001 MHz. The `tx_ce` input must be High all of the time. The four data streams being transmitted must be on `tx_ds1_in` through `tx_ds4_in`. Line numbers supplied on

- `tx_line_ch0` are inserted into data streams 1 and 2
- `tx_line_ch1` are inserted into data streams 3 and 4.

Line numbers must also be supplied on `tx_line_ch0` and `tx_line_ch1` in order for ST 352 packet insertion to function. If ST 352 packet insertion is enabled, for Y-stream (by default), the user data words for the ST 352 packets inserted into

- data stream 1 must be supplied on `tx_st352_data_ch0`
- data stream 3 must be supplied on `tx_st352_data_ch1`.

You must supply the `tx_st352_data_ch0_c` and `tx_st352_data_ch1_c` signals for ST 352 insertion into the C-stream.

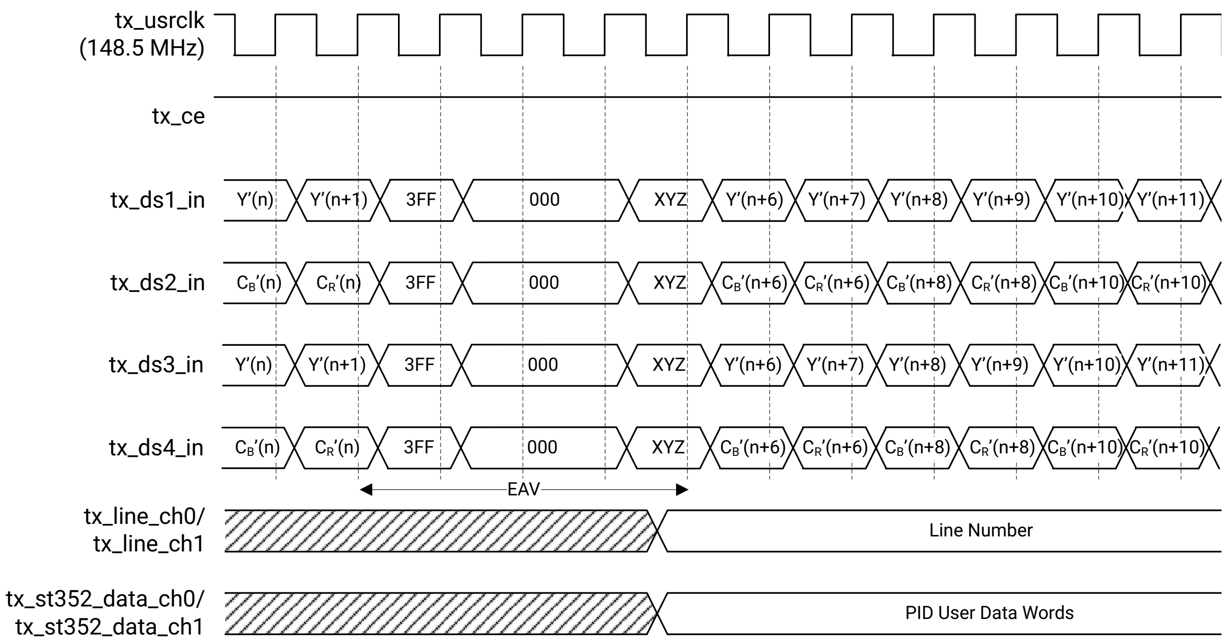

tx_usrclk (148.5 MHz)
tx_ce
tx_ds1_in
Y'(n) Y'(n+1) 3FF 000 XYZ Y'(n+6) Y'(n+7) Y'(n+8) Y'(n+9) Y'(n+10) Y'(n+11)
tx_ds2_in
C_B'(n) C_R'(n) 3FF 000 XYZ C_B'(n+6) C_R'(n+6) C_B'(n+8) C_R'(n+8) C_B'(n+10) C_R'(n+10)
tx_ds3_in
Y'(n) Y'(n+1) 3FF 000 XYZ Y'(n+6) Y'(n+7) Y'(n+8) Y'(n+9) Y'(n+10) Y'(n+11)
tx_ds4_in
C_B'(n) C_R'(n) 3FF 000 XYZ C_B'(n+6) C_R'(n+6) C_B'(n+8) C_R'(n+8) C_B'(n+10) C_R'(n+10)
EAV
tx_line_ch0/ tx_line_ch1
Line Number
tx_st352_data_ch0/ tx_st352_data_ch1
PID User Data Words

Figure 3-11: **6G-SDI TX 4-way Interleave Timing Diagram**

Figure 3-12 is a timing diagram for the transmitter in 6G-SDI mode when transmitting a signal containing eight data streams. The `tx_clk` frequency is 148.5 MHz or 148.5/1.001 MHz. The `tx_ce` input must be asserted every other clock cycle. The eight data streams being transmitted must be on `tx_ds1_in` through `tx_ds8_in`. Line numbers supplied on

- `tx_line_ch0` are inserted into data streams 1 and 2
- `tx_line_ch1` are inserted into data streams 3 and 4
- `tx_line_ch2` are inserted into data streams 5 and 6
- `tx_line_ch4` are inserted into data streams 7 and 8.

Line numbers must also be supplied on `tx_line_ch0` through `tx_line_ch4` in order for ST 352 packet insertion to function. If ST 352 packet insertion is enabled (through a port), the user data words for the ST 352 packets inserted into

- data stream 1 must be supplied on `tx_st352_data_ch0`
- data stream 3 must be supplied on `tx_st352_data_ch1`
- data stream 5 must be supplied on `tx_st352_data_ch2`
- data stream 7 must be supplied on `tx_st352_data_ch3`.

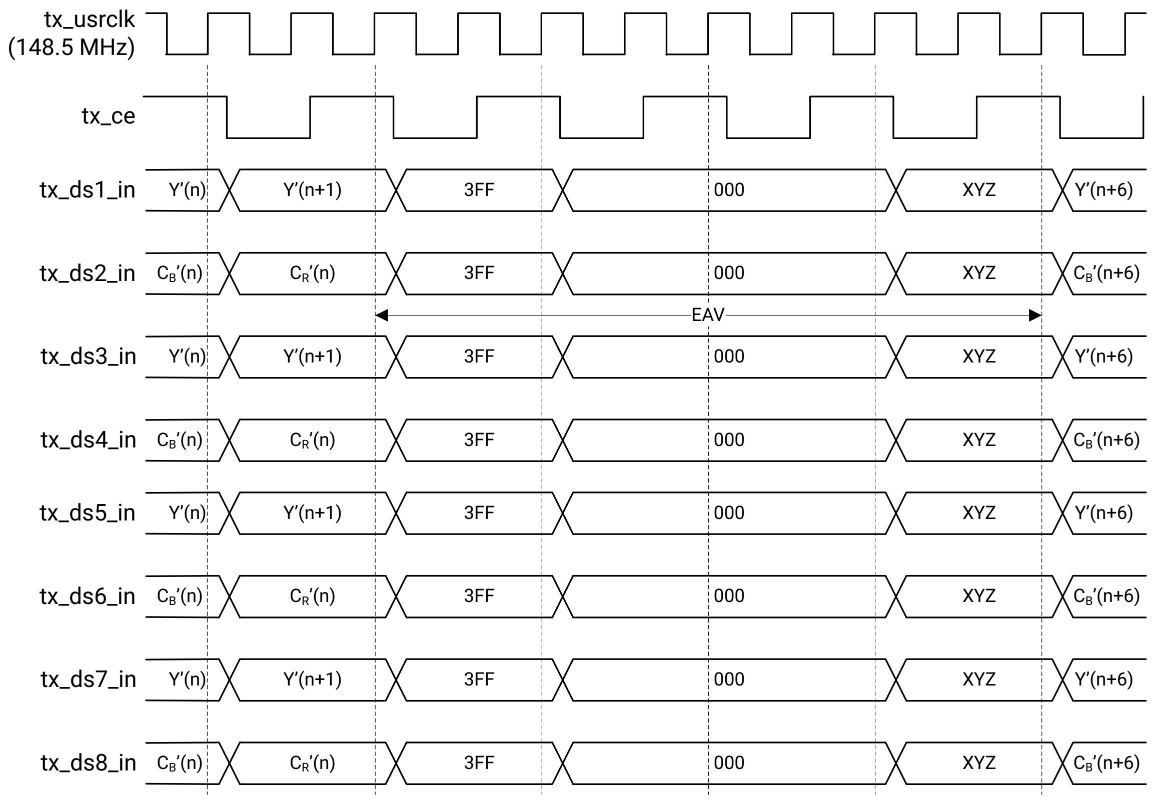


Figure 3-12: **6G-SDI TX 8-way Interleave Timing Diagram**

12G-SDI Mode

In 12G-SDI mode, the RXOUTCLK/TXOUTCLK frequency is 297 MHz (or 297/1.001 MHz). Depending on the 12G-SDI mapping mode used, there may be either eight or sixteen active elementary data streams. When eight elementary data streams are active, the data streams run at half the RXOUTCLK/TXOUTCLK frequency and the `rx_ce_out` and `tx_ce` have 50% duty cycles. When sixteen elementary data streams are active, the data streams run at one quarter the RXOUTCLK/TXOUTCLK frequency and the `rx_ce_out` and `tx_ce` have 25% duty cycles.

The timing diagram for 8-way interleave on a 12G-SDI RX interface is identical to Figure 3-12, the 8-way interleave 6G-SDI RX interface with the exception that `rx_clk` run at 297 MHz for 12G-SDI.

Figure 3-13 is a timing diagram for the receiver in 12G-SDI mode when receiving a signal containing sixteen data streams. The `rx_clk` frequency is 297 MHz or 297/1.001 MHz. The `rx_ce_out` signal is asserted one clock cycle out of every four (25% duty cycle). The sixteen received data streams are output on `rx_ds1` through `rx_ds16`. Not all sixteen data streams are shown in the diagram, but they are all active. Figure 3-13 shows the EAV being received and the `rx_trs` and `rx_eav` behave the same as with other SDI standards. Line numbers are captured from all sixteen data streams and output on `rx_ln_ds1` through `rx_ln_ds16`. CRC errors are detected individually for all sixteen data streams and indicated on the `rx_crc_err_ds1` through `rx_crc_err_ds16` outputs. The line number and CRC error outputs are not shown on the timing diagram, but their timing is the same as with other SDI modes relative to the position of the EAV and LN words in the data streams.

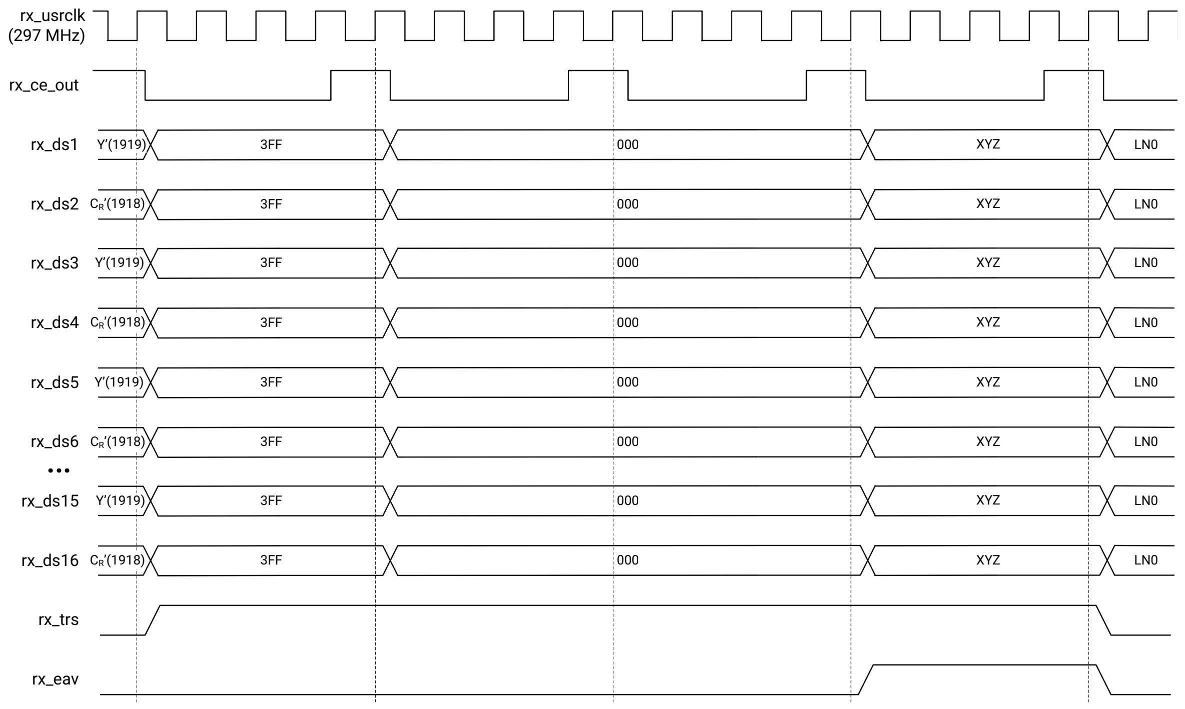

rx_usrclk (297 MHz)
rx_ce_out
rx_ds1 Y'(1919) 3FF 000 XYZ LN0
rx_ds2 C_R'(1918) 3FF 000 XYZ LN0
rx_ds3 Y'(1919) 3FF 000 XYZ LN0
rx_ds4 C_R'(1918) 3FF 000 XYZ LN0
rx_ds5 Y'(1919) 3FF 000 XYZ LN0
rx_ds6 C_R'(1918) 3FF 000 XYZ LN0
...
rx_ds15 Y'(1919) 3FF 000 XYZ LN0
rx_ds16 C_R'(1918) 3FF 000 XYZ LN0
rx_trs
rx_eav

Figure 3-13: **12G-SDI RX 16-way Interleave Timing Diagram**

The timing diagram for 12G-SDI TX with eight interleaved data streams is identical to Figure 3-13, the 8-way interleaved 6G-SDI transmitter timing diagram, except that the `tx_clk` frequency is 297 MHz.

Figure 3-14 is a timing diagram for the transmitter in 12G-SDI mode when transmitting a signal containing sixteen data streams. The `tx_clk` frequency is 297 MHz or 297/1.001 MHz. The `tx_ce` input must be asserted every other clock cycle. The sixteen data streams being transmitted must be on `tx_ds1_in` through `tx_ds16_in`. When either line number insertion or ST 352 packet insertion is enabled (through the `tx_insert_ln` port), line numbers must be supplied on the eight line number input ports `tx_line_ch0` through `tx_line_ch7`. There is one line number input port for each pair of data streams. The line number on `tx_line_ch0` are associated with data streams `tx_ds1_in` and `tx_ds2_in`, `tx_line_ch1` with `tx_ds3_in` and `tx_ds4_in`, and so on. As with the other SDI modes, the line number values on the line number inputs must be stable by the rising edge of the `tx_clk` on which the XYZ words of the EAVs enter the data stream input ports and must remain stable until the beginning of the SAV, in other words, for the duration of the HANC interval on each line. If ST 352 packet insertion is enabled, the line numbers must be supplied to the transmitter as just described and the user data words for the ST 352 packets must also be supplied on the `tx_st352_data_ch0` through `tx_st352_data_ch7` input ports. There is one ST 352 user data input port for each pair of data streams. ST 352 packets are only inserted into the odd numbered data stream of each pair (the Y data stream). The ST 352 data supplied on `tx_st352_data_ch0` is associated with data stream 1 (`tx_ds1_in`), `tx_st352_data_ch1` is associated with data stream 3 (`tx_ds3_in`), and so on. The data on these ST 352 data input ports must meet the same timing requirements as the line numbers. You can insert ST3 52 into the C data stream, shown in the waveform. You must supply the ST 352 payload on `tx_st352_data_ch0_c` to `tx_st352_data_ch7_c`, depending on the available data streams.

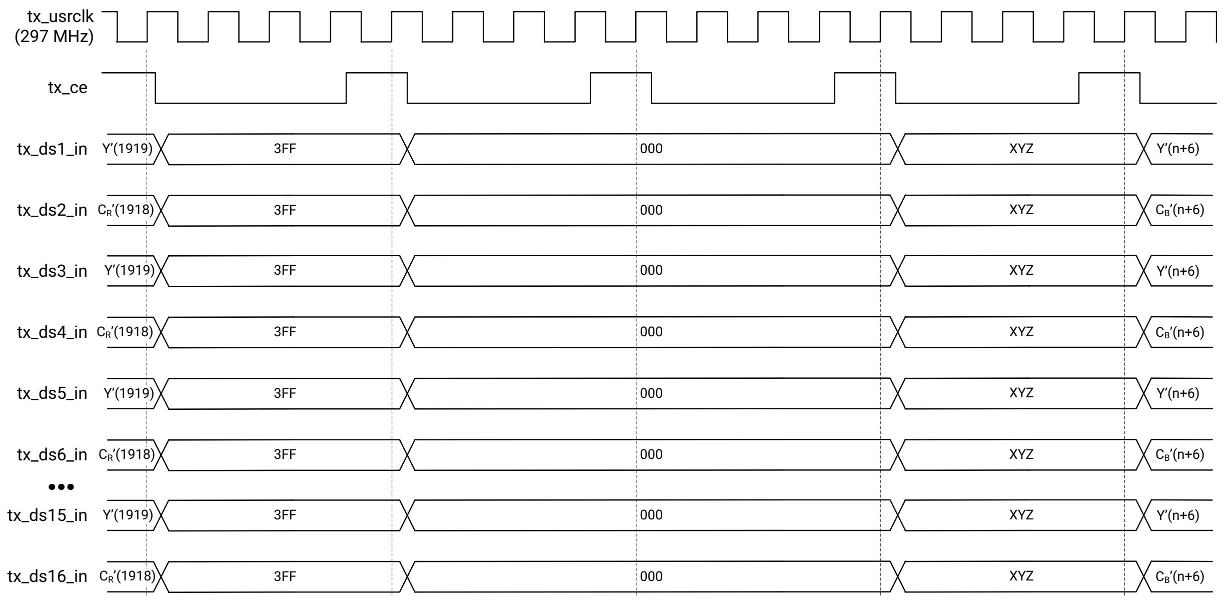


Figure 3-14: **12G-SDI TX 16-way Interleave Timing Diagram**

Resets

Resetting the SMPTE UHD-SDI core and the associated serial transceiver is device-specific. Refer to *Implementing SMPTE SDI Interfaces with UltraScale GTH Transceivers* [Ref 6] and *Implementing SMPTE SDI Interfaces with 7 Series GTX Transceivers* [Ref 7] for detailed information about resets.

Dual Link Example Use Cases

Dual link and quad link 6G-SDI and 12G-SDI are supported by instantiating two or four SMPTE UHD-SDI cores. In other words, there are two ways to achieve 12G dual link with the core depending on whether there are 8 or 16 incoming data streams.

Use Case 1 (12G-8DS)

For each core instance, drive `tx_mux_pattern` with 3'b010. The `tx_ce` port must be driven with a 50% duty cycle. Using this setting, the data is output in the following format:

{ds2, ds6, ds4, ds8} when `ce` is Low

{ds1, ds5, ds3, ds7} when `ce` is High

Use Case 2 (12G-16DS)

For each core instance, drive `tx_mux_pattern` with 3'b100. The `tx_ce` port must be driven with a 25% duty cycle. Using this setting, data is output in the following format:

{ds4, ds12, ds8, ds16} after the rising edge of the clock in which `ce` is asserted

{ds2, ds10, ds6, ds14} for second clock cycle

{ds3, ds11, ds7, ds15} for third clock cycle

{ds1, ds9, ds5, ds13} for last clock cycle

However, for dual-link HD-SDI, 3G-SDI level B-DL, multi-link 3G-SDI, 6G-SDI, and 12G-SDI, mapping of the video formats to and from elementary data streams is to be done independently outside of the SMPTE UHD-SDI core.

tx_ce Port Requirements based on tx_mux_pattern Value

The following requirements must be met:

- tx_mux_pattern 3'b000 requires ce to be asserted all of the time. ds1 passes straight through to output bits [19:10] and ds2 passes through to output bits [9:0].
- tx_mux_pattern 3'b001 requires ce to be asserted with a 50% duty cycle. When ce is Low, {ds2, ds4} is output. When ce is High, {ds1, ds3} is output.
- tx_mux_pattern 3'b010 requires ce to be asserted with a 50% duty cycle. When ce is Low, {ds2, ds6, ds4, ds8} is output. When ce is High, {ds1, ds5, ds3, ds7} is output.
- tx_mux_pattern 3'b011 requires ce to be asserted all of the time. The output is {ds1, ds3, ds2, ds4}.
- tx_mux_pattern 3'b100 requires ce to be asserted 25% of the time. After the rising edge of the clock in which ce is asserted, the output is {ds4, ds12, ds8, ds16}. The next clock cycle the output is {ds2, ds10, ds6, ds14}. The next clock cycle the output is {ds3, ds11, ds7, ds15}. And in the fourth and final clock cycle, the output is {ds1, ds9, ds5, ds13}.

Design Flow Steps

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

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 1]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 3]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4]

Customizing and Generating the Core

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

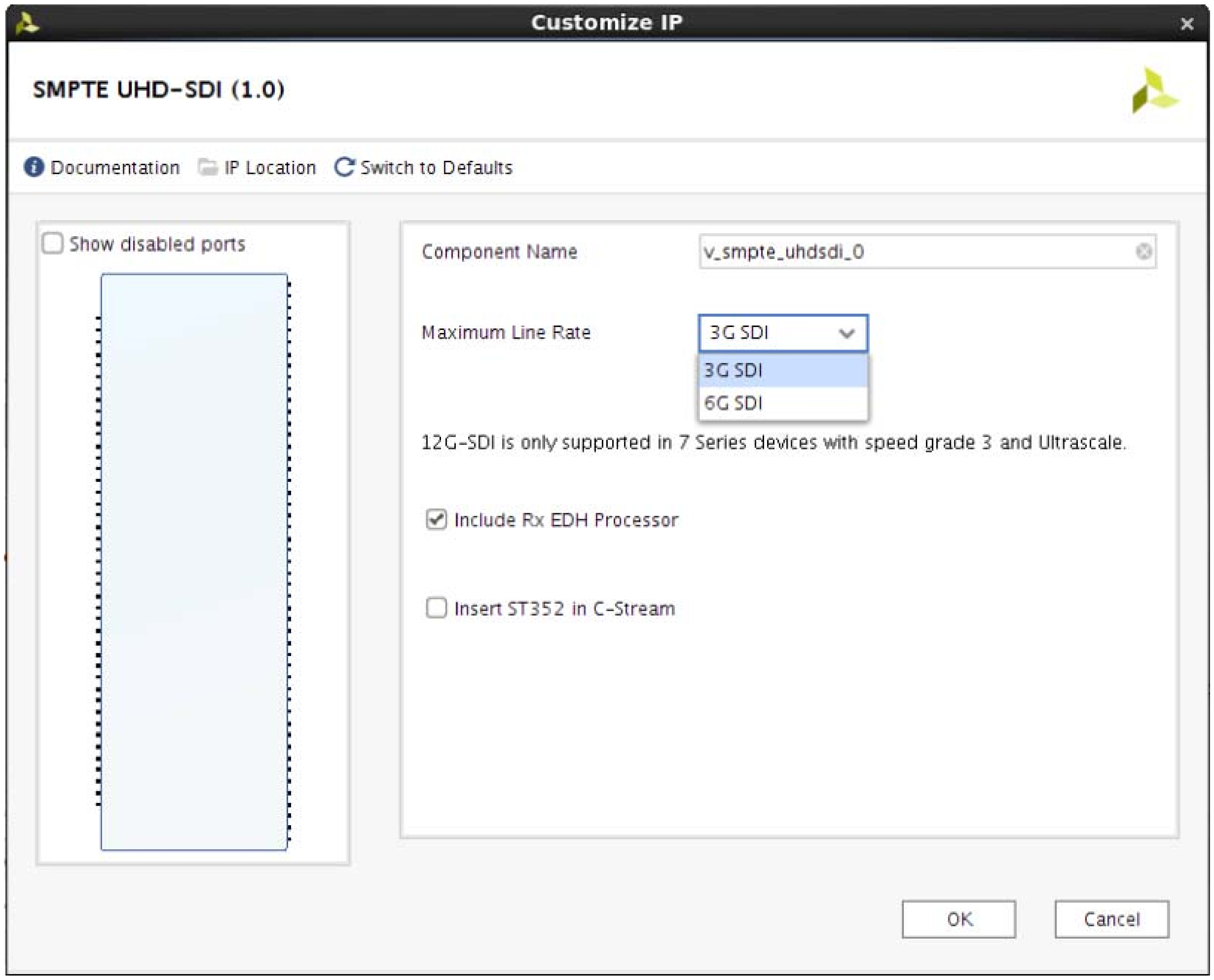
Vivado Integrated Design Environment

You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

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

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 3].

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



Component Name: v_smpte_uhdsdi_0

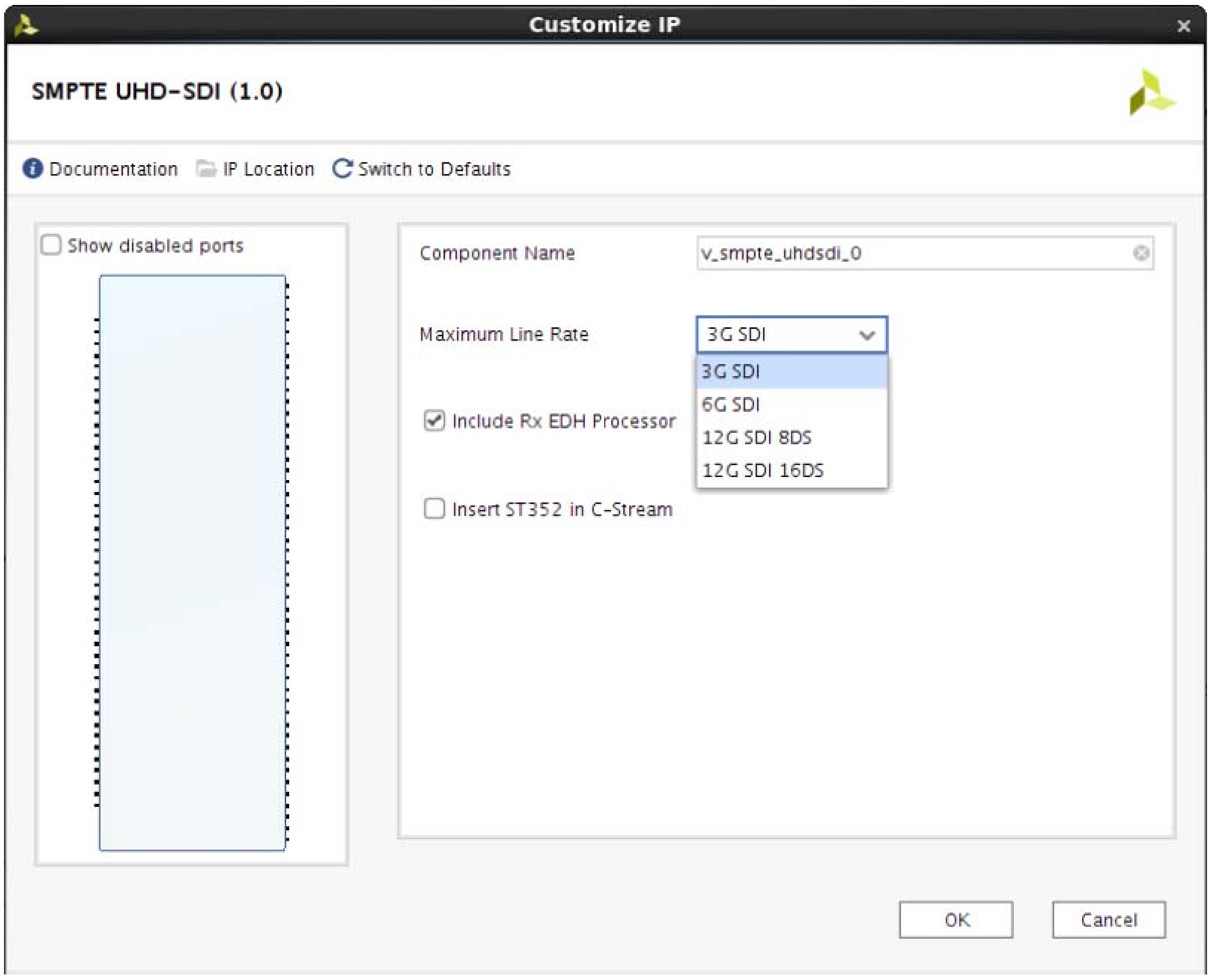
Maximum Line Rate: 3G SDI (options: 3G SDI, 6G SDI)

12G-SDI is only supported in 7 Series devices with speed grade 3 and Ultrascale.

[✓] Include Rx EDH Processor

[ ] Insert ST352 in C-Stream

Figure 4-1: **SMPTE UHD-SDI Vivado IDE (7 Series Speed Grade 1 and 2)**


Customize IP

SMPTE UHD-SDI (1.0)

Documentation   IP Location   Switch to Defaults

Show disabled ports

Component Name: v_smpte_uhdsdi_0

Maximum Line Rate: 3G SDI

- 3G SDI
- 6G SDI
- 12G SDI 8DS
- 12G SDI 16DS

[✓] Include Rx EDH Processor

[ ] Insert ST352 in C-Stream

OK   Cancel

Figure 4-2: **SMPTE UHD-SDI Vivado IDE (7 Series Speed Grade 3 and UltraScale/UltraScale+)**

The Vivado IDE displays a representation of the IP symbol on the left side and the parameter assignments on the right, as follows:

- **Component Name**: The base name of output files generated for the module. Names must begin with a letter and must be composed of characters a to z, 0 to 9 and "_". The name v_smpte_uhdsdi_v1_0 cannot be used as a component name.
- **Maximum Line Rate**: Specifies the maximum rate supported for that device family.

  For 7 series speed grade 3 and UltraScale™/UltraScale+™ devices, the following line rate are supported:

  - 3G SDI
  - 6G SDI
  - 12G SDI 8 Data Stream
  - 12G SDI 16 Data Stream

For 7 series speed grade grade 1 and 2 devices, the following line rate are supported:

- 3G SDI
- 6G SDI

- **Include Rx EDH Processor**: If this parameter is ENABLED, then the EDH processor for the receiver section of the SMPTE UHD-SDI core is included. If this parameter is DISABLED, then the EDH processor for the receiver section is not included.
- **Insert ST352 in C-stream**: If this parameter is enabled, St352 payload packets are inserted into the C channel of the data streams.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Constraining the Core

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

Required Constraints

The period of the `rx_clk` and `tx_clk` must be constrained depending on the maximum line rate to be supported.

The EDH processors in the design also need multi cycle clock path constraints which are automatically provided when the core is generated.

Device, Package, and Speed Grade Selections

For 7 series devices, -3 speed grade parts are required to support 12G-SDI. Not all packages support 12G-SDI line rates. The MGTAVCC voltage rail must be set to 1.05 V if 12G-SDI operation is required. This voltage level also support s the other SDI lines rates. If the maximum line rate is 6G-SDI or slower, then -1 speed grade devices are sufficient and the MGTAVCC voltage rail can be set to the normal value of 1.00 V.

UltraScale/UltraScale+ GTH transceivers support operation at all SDI rates up to and including 12G-SDI in -1 speed grade devices.

Clock Frequencies

Applications that support 12G-SDI operation must constrain the frequency of the SMPTE UHD-SDI core `rx_clk` and `tx_clk` to 297 MHz. The source of the `rx_clk` is usually the serial transceiver signal, `RXOUTCLK`. The source of the `tx_clk` is usually the serial transceiver signal, TXOUTCLK. The exact constraints to be used on these clocks depends on the hierarchical structure of the design, but they are similar to the constraints shown below with an application specific path to the TXOUTCLK and RXOUTCLK pins of the serial transceiver.

```
create_clock -period 3.333 -name tx0_outclk -waveform {0.000 1.667} [get_pins SDI/
GTX/gtxe2_i/TXOUTCLK]
create_clock -period 3.333 -name rx0_outclk -waveform {0.000 1.667} [get_pins SDI/
GTX/gtxe2_i/RXOUTCLK]
```

When the maximum line rate is 6G-SDI or slower, the maximum clock frequency of rx_clk and tx_clk is 148.5 MHz and the constraints below are appropriate:

```
create_clock -period 6.667 -name tx0_outclk -waveform {0.000 3.333} [get_pins SDI/
GTX/gtxe2_i/TXOUTCLK]
create_clock -period 6.667 -name rx0_outclk -waveform {0.000 3.333} [get_pins SDI/
GTX/gtxe2_i/RXOUTCLK]
```

Simulation

This section contains information about simulating IP in the Vivado Design Suite. For comprehensive information about Vivado simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4].

Synthesis and Implementation

This section contains information about synthesis and implementation in the Vivado Design Suite. For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Test Bench

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

As shown in Figure 5-1, the demonstration test bench is a simple System Verilog module which generates all the supported SMPTE test patterns (SD-SDI, 3G-SDI, 6G-SDI, and 12G-SDI). The Test Pattern Generator (TPG) drives the SDI data to the core transmitter, which is loop backed to the core receiver. The output of the core receiver is driven to the SDI slave which performs the protocol and data check. The shaded blocks in the following figure are the System Verilog modules and the data recovery unit (DRU) block is Verilog.

IMPORTANT: *This simulation test bench is used to run simulation on the UHD-SDI data core. It does not include serial transceiver integration.*

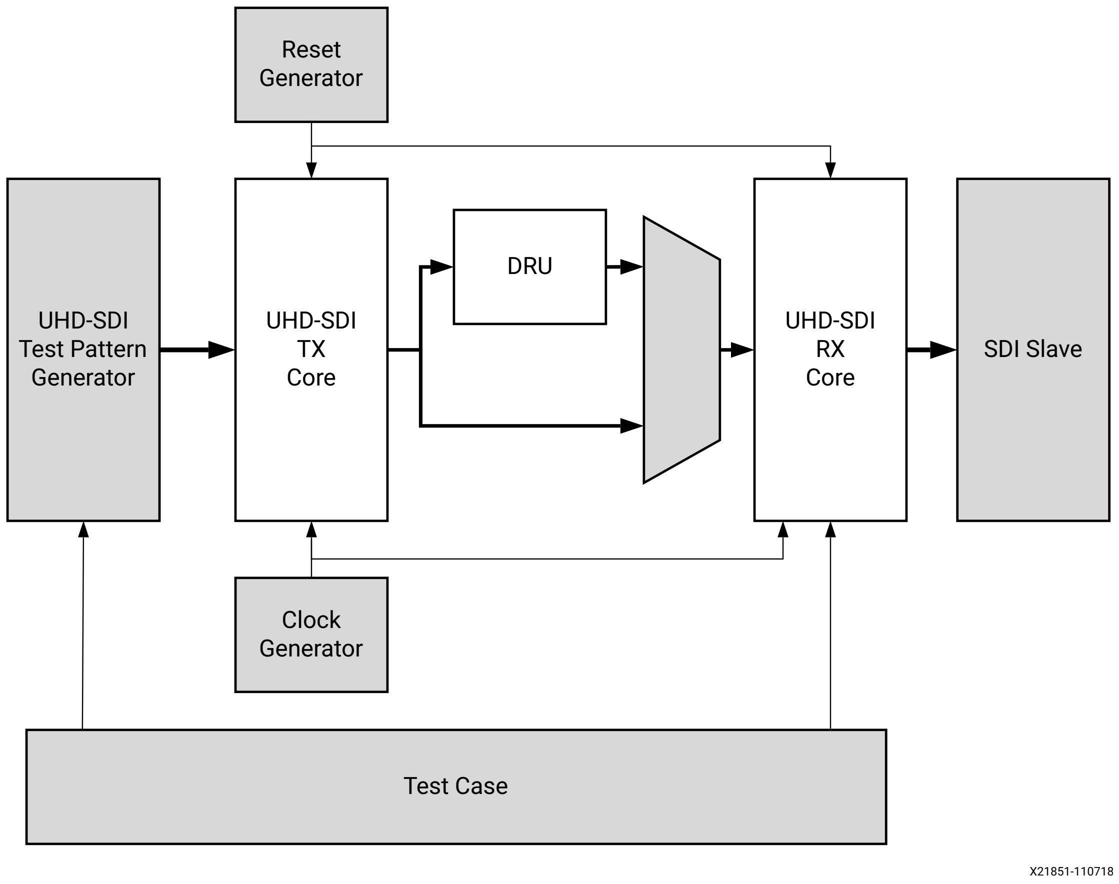


Figure 5-1: **SMPTE UHD-SDI Test Bench**

Appendix A

Verification, Compliance, and Interoperability

This appendix provides details about how this IP core was tested for compliance with the protocol to which it was designed.

Hardware Testing

The SMPTE UHD-SDI core has been tested with standard off-the-shelf SDI test equipment and with a variety of preliminary SMPTE UHD-SDI devices. It is compliant with the SMPTE SDI standards.

Upgrading

Upgrading in the Vivado Design Suite

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

Debugging

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

Finding Help on Xilinx.com

To help in the design and debug process when using the SMPTE UHD-SDI, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the SMPTE UHD-SDI. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

The Solution Center specific to the SMPTE UHD-SDI core is listed below.

- Xilinx Video Solution Center

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product.

Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as

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

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

Master Answer Record for the SMPTE UHD-SDI

AR: 54547

Technical Support

Xilinx provides technical support at the Xilinx Support web page for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support, navigate to the Xilinx Support web page.

Debug Tools

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

Vivado Design Suite Debug Feature

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

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

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

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 5].

Reference Boards

Various Xilinx development boards support the SMPTE UHD-SDI. These boards can be used to prototype designs and establish that the core can communicate with the system.

- KCU105 (see *Implementing SMPTE SDI Interfaces with UltraScale GTH Transceivers (XAPP1248)* [Ref 6])
- KC705 (see *Implementing SMPTE SDI Interfaces with 7 Series GTX Transceivers (XAPP1249)* [Ref 7])

Simulation Debug

Simulation of an SDI interface is not typically done on only the SMPTE UHD-SDI core itself, but usually also includes a simulation module of the transceiver plus device-specific logic to control that transceiver and interface it to the SMPTE SD/HD/3G-SDI core. The required transceiver interface and control logic is found in device-specific application notes.

Simulation of the transmit portion of the SMPTE UHD-SDI core requires a source of video providing one or more digital video standards that are supported by the core. The example designs in the device-specific application notes provide video test pattern generators that can provide digital video to the SDI transmitter.

Simulation of the receive portion of the SMPTE UHD-SDI core requires a SDI signal source. The easiest way to provide this the simulation model is with a complete simulation of the SDI transmitter including a video source, the TX portion of the SMPTE UHD-SDI core, and the transceiver simulation model.

IMPORTANT: Note that the loopback modes of the transceiver can be used to loop the transceiver TX section back into the RX section. This is often convenient for simulation as well as hardware debug.

To determine if the RX section is correctly receiving the SDI signal, look for pulses on the `rx_eav` and `rx_sav` signals at the correct places on each line, absence of pulses on the `rx_crc_err_dsN` signals, assertion of the `rx_mode_locked` signal. Verify that the `rx_mode` output port is indicating the correct SDI mode when `rx_mode_locked` is asserted. Verify that the video streams output by the receiver match the video streams input to the transmitter, with some amount of latency. Assertion of `rx_t_locked` takes much longer than assertion of `rx_mode_locked` because the transport format detector may take up to two frames of video before it can determine the video format, so the format detector outputs (`rx_t_locked`, `rx_t_family`, `rx_t_rate`, and `rx_t_scan`) are not correct unless the simulation is allowed to run for two full frames of video.

Hardware Debug

Experience has shown that most of the issues that occur when implementing SDI interfaces with Xilinx transceivers are related to clocking and with incorrect use of the transceivers. The SMPTE UHD-SDI core is very robust and when the transceiver is properly connected to the SMPTE UHD-SDI core and the clocks are connected correctly, the SMPTE UHD-SDI core almost always works as designed. The first thing to check when a SMPTE UHD-SDI receiver or transmitter is not working is that the associated clock from transceiver is running at the correct speed. For HD-SDI the receive and transmit clocks that are connected, usually through a BUFG, to the `rx_clk` and `tx_clk` ports of the SMPTE UHD-SDI core should have a frequency of either 74.25 MHz or 74.1758 MHz, depending on which HD-SDI bit rate is being received or transmitted. For 3G-SDI and 6G-SDI, these clocks should have a frequency of 148.5 MHz or 148.35 MHz, again depending on which bit rate is being received or transmitted. For 12G-SDI, these clocks should have a frequency of 297 MHz or 296.7 MHz. For SD-SDI, the `rx_clk` frequency can be either 148.5 MHz or 148.35 MHz, depending on the frequency of the reference clock used by the RX section of the transceiver. Also for SD-SDI TX, the `tx_clk` must always be 148.5 MHz, never 148.35 MHz.

If the clocks are halted or running at incorrect frequencies, then there is probably a problem with the transceiver or the reference clocks to the transceiver. Perhaps the PMA PLLs did not get properly reset after the reference clocks became stable. Using Vivado Analyzer to manually assert various resets on the transceiver can help to determine if there is a problem with a reset signal not getting asserted or not being asserted at the right time. Also check that the resets to the transceiver are not being asserted all of the time and preventing the transceiver from operating properly.

Another issue that occurs commonly is that the clock driving `tx_clk` port of the SMPTE UHD-SDI core and the TXUSRCLK and TXUSRCLK2 ports of the transceiver is not frequency locked to the reference clock used by the TX portion of the transceiver. This problem is avoided if the TXOUTCLK of the transceiver is the clock driving `tx_clk`, TXUSRCLK, and TXUSRCLK2. But, if another clock is used to drive these clock inputs, then this clock must be frequency locked to TXOUTCLK and the reference clock driving the TX section of the transceiver. If these clocks are not frequency locked, then the TXBUFFER in the transceiver quickly under or overflows, causing corruption of the serial bit stream generated by the transceiver. Xilinx transceivers have a status port that can be monitored to determine if the TXBUFFER is under or overflowing. On the 7-series transceiver, the TXBUFFER under/overflow signal is bit 1 of the TXBUFSTATUS port. If this signal ever becomes asserted High, then the TXBUFFER has under or overflowed.

The loopback mechanism built into Xilinx transceivers is a very useful debugging tool. This mechanism allows you to loop the TX portion of the transceiver back to the RX portion, internally to the transceiver. Thus, if you have a working SDI transmitter, you can use that to test that the SDI receiver is working. Or if you have a working SDI receiver, you can use that to test that the SDI transmitter is working correctly.

Another useful technique used to determine if the data stream going into the TXIN port of the transceiver from the TX section of the SMPTE UHD-SDI core is a valid SDI data stream involves connecting the data stream that goes from the transceiver TXDATA port to the `rx_data_in` port of the RX section of the SMPTE UHD-SDI core and then monitor the status and video output ports of the RX section of the core using the Vivado Analyzer. If the RX section of the core locks to the data stream and produces correct video and status signals on its output, then the data stream going into the TX section of the transceiver is correct. Note that to verify SD-SDI with this technique, the appropriate device-specific SDI wrapper should be used instead of just the SMPTE UHD-SDI core or the extra SDI receiver used to monitor the TXDATA data stream.

General Checks

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

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the `locked` port.

Appendix D

Additional Resources and Legal Notices

Xilinx Resources

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

Documentation Navigator and Design Hubs

Xilinx Documentation Navigator provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open the Xilinx Documentation Navigator (DocNav):

- From the Vivado® IDE, select **Help > Documentation and Tutorials**.
- On Windows, select **Start > All Programs > Xilinx Design Tools > DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In the Xilinx Documentation Navigator, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

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

References

These documents provide supplemental material useful with this product guide:

1. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
2. *Vivado Design Suite User Guide: Designing with IP* (UG896)
3. *Vivado Design Suite User Guide: Getting Started* (UG910)
4. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
5. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
6. *Implementing SMPTE SDI Interfaces with UltraScale GTH Transceivers* (XAPP1248)
7. *Implementing SMPTE SDI Interfaces with 7 Series GTX Transceivers* (XAPP1249)
8. *Implementing SMPTE SDI Interfaces with Kintex-7 GTX Transceivers* (XAPP592)
9. *SMPTE UHD-SDI Transmitter Subsystem Product Guide* (PG289)
10. *SMPTE UHD-SDI Receiver Subsystem Product Guide (*PG290*)*

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
09/28/2020	1.0	Added details for CPLL/QPLL 12G-SDI line rates in Chapter 2, UltraScale/ UltraScale+ GTH.
12/05/2018	1.0	Added support for C-stream ST 352 payload insertion and extraction in Table 2-2 and Table 2-3.
06/06/2018	1.0	Added a section for latency information in Chapter 2, Performance section.
10/04/2017	1.0	Added support for UltraScale+ GTH transceivers.
04/05/2017	1.0	Added Dual Link Example Use Cases section to Chapter 3, Designing with the Core.
04/01/2015	1.0	Initial Xilinx release.

Please Read: Important Legal Notices