ECP5 and ECP5-5G SERDES/PCS Usage Guide

November 2015 Technical Note TN1261

Introduction

The ECP5™ and ECP5-5G™ family architecture is similar to LatticeECP3™ SERDES, with enhanced granularity, low power and low cost. Up to four channels of embedded SERDES with associated Physical Coding Sublayer (PCS) logic are arranged in dual channel base.

Two channels of the LFE5UM/LFE5UM5G are built into one unit called DCU (Dual Channel Unit). Each DCU contains one reference clock input and one TxPLL. This Dual-channel based architecture provides higher clock to SERDES channel ratio than its predecessor FPGAs (ECP2M and ECP3).

Each channel of PCS contains dedicated transmit and receive logic for high-speed, full-duplex serial data transfer at data rates up to 3.2 Gbps. The PCS logic in each channel can be configured to support an array of popular data protocols including GbE, XAUI, PCI Express, CPRI, JESD2048, SD-SDI, HD-SDI and 3G-SDI. In addition, the protocol-based logic can be fully or partially bypassed in a number of configurations to allow users flexibility in designing their own high-speed data interface.

The PCS also provides bypass modes that allow a direct 8-bit or 10-bit interface from the SERDES to the FPGA logic. Each SERDES channel input can be independently AC-coupled or DC-coupled and can allow for both high-speed and low-speed operation on the same SERDES pin for applications such as Serial Digital Video.

Features

- **Up to Four Channels of High-Speed SERDES with Increased Granularity**
  - — From 270 Mbps up to 3.2 Gbps for ECP5 device family
  - — Up to 5 Gbps for ECP5-5G device family on most protocol standards. Refer to Table 1.
  - — 3.2 Gbps operation with low 85 mW power per channel
  - — Receive equalization and transmit de-emphasis with both pre-cursor and post-cursor, for small form factor backplane operation
  - — Supports PCI Express, Gigabit Ethernet (1GbE and SGMII) and XAUI, plus multiple other standards
  - — Supports user-specified generic 8b10b mode
- **Multiple Clock Rate Support**
  - — Separate reference clocks for each DCU allow easy handling of multiple protocol rates on a single device
- **Full-Function Embedded Physical Coding Sub-layer (PCS) Logic Supporting Industry Standard Protocols**
  - — Up to four channels of full-duplex data supported per device.
  - — Multiple protocol support on one chip.
  - — Supports popular 8b10b-based packet protocols.
  - — SERDES Only mode allows direct 8-bit or 10-bit interface to FPGA logic.
- **Multiple Protocol Compliant Clock Tolerance Compensation (CTC) Logic**
  - — Compensates for frequency differential between reference clock and received data rate.
  - — Allows user-defined skip pattern of two or four bytes in length.
- **Integrated Loopback Modes for System Debugging**
  - — Three loopback modes are provided for system debugging
- **Easy to Use Design Interface in Lattice Diamond™ Design Tool**
  - — System Planner/Builder provides an easy to use GUI interface to assist users to instantiate the SERDES as a link, instead of individual SERDES channels
  - — The tools allow the users to plan all resources to complete the design of a functional link, including SERDES/PCS, Lattice IPs or user design IPs, and all interface logic.

New Features Over LatticeECP3 SERDES/PCS

- LFE5UM/LFE5UM5G implements the SERDES block in Dual-Channel architecture. This architecture provides more clock resources per channel base, which allows the users to have more flexibility on utilizing the channels.
- Bit-slip feature simplifies user logic on Word Alignment. This function is important for CPRI and JESD204A/B applications.
- Supports Transmit/Receive to eDP SERDES interface, up to 2.7 Gbps.
- Supports JESD204A/B - ADC and DAC converter I/F: 312.5 Mbps to 3.125 Gbps for ECP5, to 5 Gbps for ECP5-5G.
- H-Bridge output driver reduces power consumption.

Difference Between ECP5 and ECP5-5G SERDES/PCS

ECP5-5G makes some enhancements in the SERDES to the ECP5 family. These enhancements increase the performance of the SERDES to up to 5 Gbps data rate.

In order to take full advantage of the designs that have been done with ECP5, the ECP5-5G family devices are totally pin compatible with devices in ECP-5 family.

One difference needs to be noted is the ECP5-5G devices need 1.2 V nominal on SERDES VCCA, VCCHRX and VCCHTX power supplies, compared to the 1.1 V needed for ECP-5 family. DS1044, ECP5 and ECP5-5G Family Data Sheet.

Using This Technical Note

The Lattice Diamond design tools support all modes of the PCS. Most modes are dedicated to applications for a specific industry standard data protocol. Other modes are more general purpose in nature and allow designers to define their own custom application settings. Diamond design tools allow the user to define the mode for each dual in their design. This technical note describes operation of the SERDES and PCS for all modes supported by Diamond.

This document provides a thorough description of the complete functionality of the embedded SERDES and associated PCS logic. Electrical and timing characteristics of the embedded SERDES are provided in the DS1044, ECP5 and ECP5-5G Family Data Sheet. Operation of the PCS logic is provided in the PCS section of this document. A table of all status and control registers associated with the SERDES and PCS logic which can be accessed via the SCI (SerDes Client Interface) Bus is provided in the appendices. Package pinout information is provided in the Pinout Information section of DS1044, ECP5 and ECP5-5G Family Data Sheet.

Standards Supported

The supported standards are listed in Table 1.

Table 1. Standards Supported by the SERDES/PCS

Standard	Data Rate (Mbps)	System Reference Clock (MHz)	FPGA Clock (MHz)	Number of Link Width	Encoding Style
PCI Express 1.1	2500	100	250	x1, x2, x4	8b10b
PCI Express 2.0[1]	5000	200	250	x1, x2	8b10b
Gigabit Ethernet SGMII	1250	125	125	x1	8b10b
	2500	125	250	x1	8b10b
	3125	156.25	156.25	x1	8b10b
	3125	156.25	156.25	x4	8b10b
XAUI	3125	156.25	156.25	x4	8b10b
CPRI-E.6.LV	614.4	61.44	61.44	x1	8b10b
E.12.LV	1228.8	61.44, 122.88	122.88		
E.24.LV	2457.6	122.88	122.88		
E.30.LV	3072	153.6	153.6		
E.48.LV[1]	4915.2	245.76	122.88		
SD-SDI (259M, 344M)	270, 540	54	27	x1	NRZI/Scrambled
HD-SDI (292M)	1483.5	74.175, 148.35	74.175, 148.35,	x1	NRZI/Scrambled
	1485	74.25, 148.50	74.25, 148.5		
3G-SDI (424M)	2967	148.35	148.35	x1	NRZI/Scrambled
	2970	148.5	148.5		
eDP-RBR	1620	160	160	x1, x2, x4	8b10b
HBR	2700	135	135	x1, x2, x4	8b10b
JESD204A/B[1]	312.5 - 3125/5000	31.25 - 156.25/312.5	31.25 - 156.25/312.5		
10-Bit SERDES[1]	270 - 3200/5000	27 - 160/320	27 - 160/320	x1, x2, x3, x4	N/A
8-Bit SERDES[1]	270 - 3200/5000	27 - 160/320	27 - 160/320	x1, x2, x3, x4	N/A
Generic 8b10b[1]	270 - 3200/5000	27 - 160/320	27 - 160/320	x1, x2, x3, x4	8b10b

1. Data Rates > 3.125 Gbps are only supported in ECP5-5G device family.

Architecture Overview

The SERDES/PCS block is arranged in duals containing logic for two full-duplex data channels.

Figure 1 shows the arrangement of SERDES/PCS duals on the LFE5UM-85/LFE5UM5G-85.

Figure 1. LFE5UM-85/LFE5UM5G-85 Block Diagram

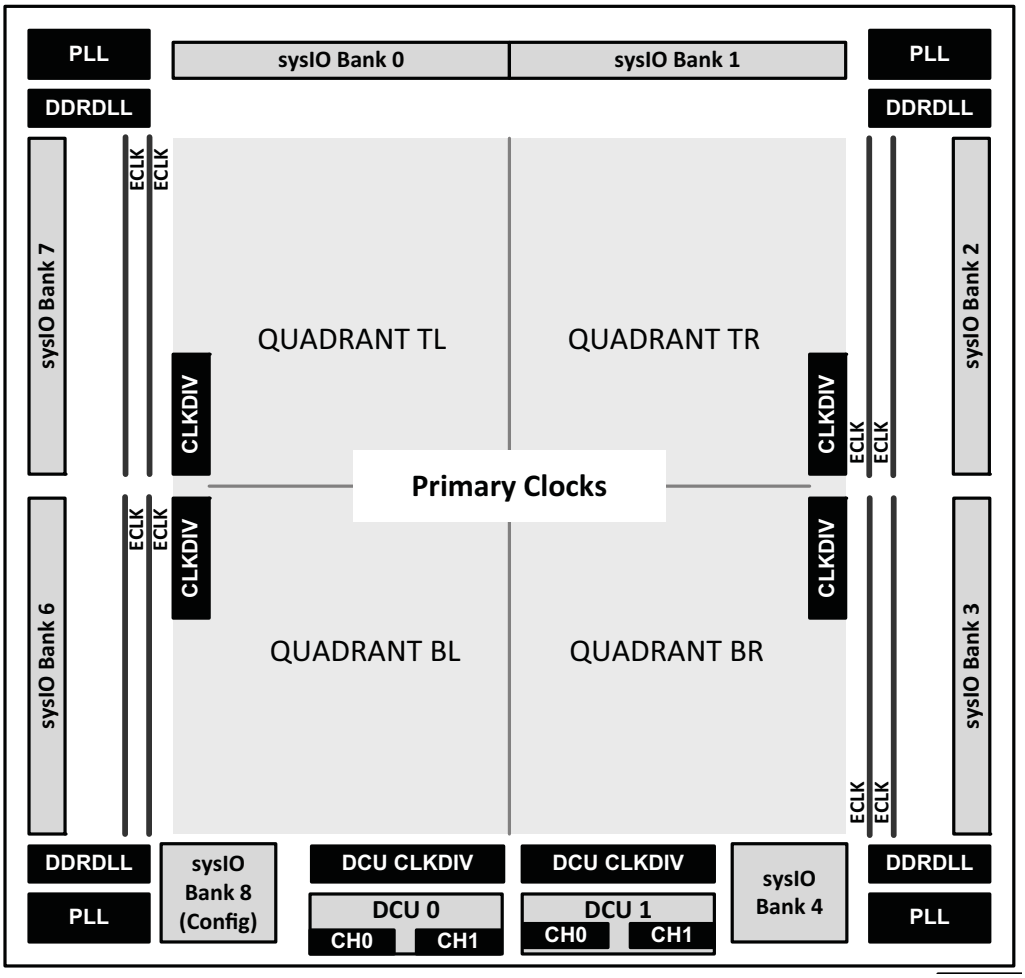


Table 2 shows the number of available SERDES/PCS channels for each LFE5UM/LFE5UM5G device.

Table 2. Number of SERDES/PCS Channels Per LFE5UM/LFE5UM5G Device

Package	LFE5UM-25/ LFE5UM5G-25	LFE5UM-45 LFE5UM5G-45	LFE5UM-85/ LFE5UM5G-85
SerDes/PCS DCUs	1	2	2
SerDes/PCS Channels	2	4	4

LFE5UM/LFE5UM5G SerDes block is defined by a dual base architecture (DCU). Each DCU includes two SerDes/PCS full-duplex Rx/Tx channels, one common AUX channel between the two SerDes channels that consists of one TX PLL and the associated EXTREF block.

Every dual can be programmed into one of several protocol-based modes. Each dual requires its own reference clock which can be sourced externally from package pins (refclkp/refclkn) or internally from the FPGA logic.

Each dual can be programmed with selected protocols that have nominal frequencies which can utilize the full and half-rate options per channel. For example, a PCI Express x1 at 2.5 Gbps and a Gigabit Ethernet channel can share same dual using the half-rate option on the Gigabit Ethernet channel. When a dual shares a PCI Express x1 channel with a non-PCI Express channel, the reference clock for the dual must be compatible with all protocols within the dual. For example, a PCI Express spread spectrum reference clock is not compatible with most Gigabit Ethernet applications.

Since each dual has its own reference clock, different duals can support different standards on the same chip. This feature makes the LFE5UM/LFE5UM5G device family ideal for bridging between different standards

PCS duals are not dedicated solely to industry standard protocols. Each dual (and each channel within a dual) can be programmed for many user-defined data manipulation modes. For example, word alignment and clock tolerance compensation can be programmed for user-defined operation.

Figure 2 is a simplified block diagram of two DCUs.

Figure 2. Simplified Block Diagram of Two DCUs in LFE5UM-45/LFE5UM5G-45 and LFE5UM-85/LFE5UM5G-85

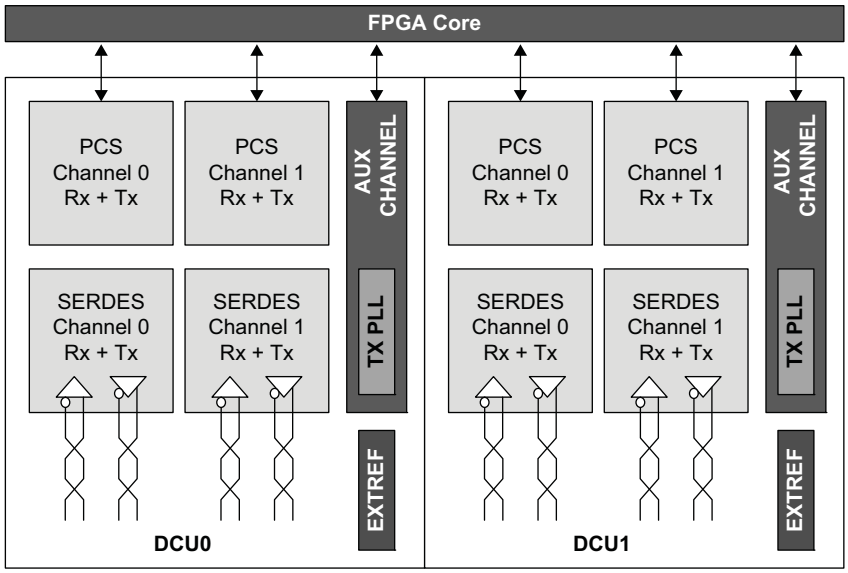
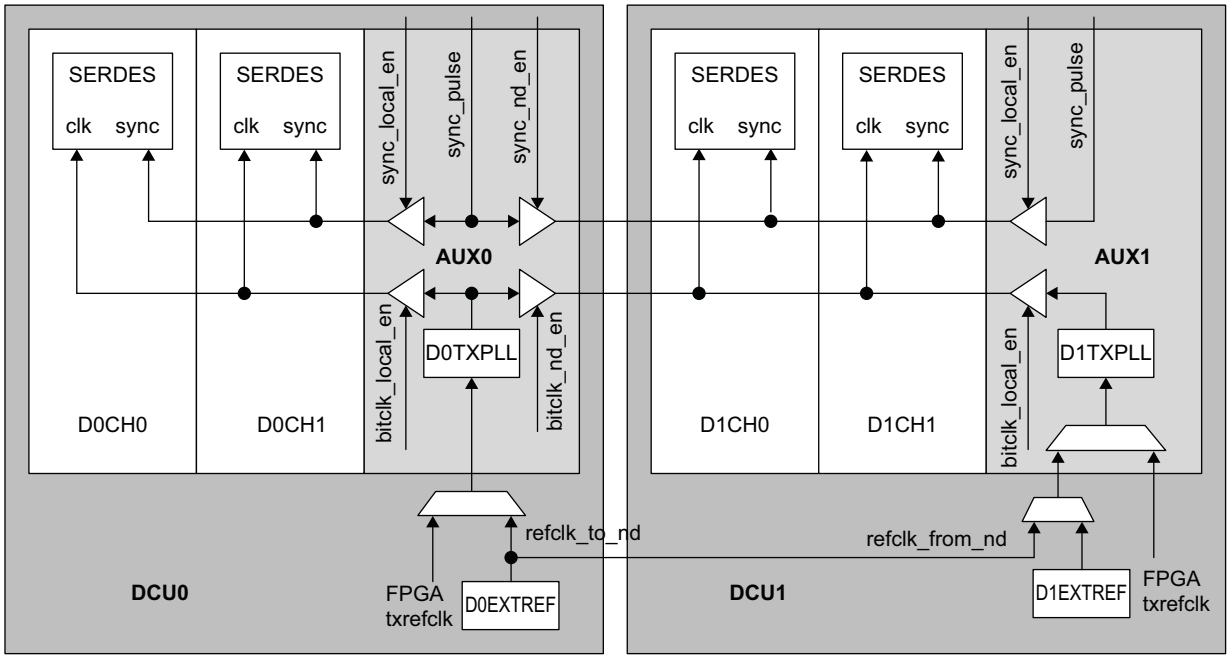


Figure 3 shows hardware arrangement of two dual and the sharing of clocking resources.

Figure 3. Two DCUs Pair with Clock Sharing



When implementing an X4 link (such as XAUI X4, or PCIe X4), one DCU Aux channel resource can be shared between two adjacent DCUs with the clock structure shown in Figure 3.

Multi-Protocol Design Consideration

The dual-channel based DCU architecture in LFE5UM/LFE5UM5G serves the purpose of providing more flexibility to the user in implementing different protocols. This is achieved by sharing resources across different DCUs when a wide interface is needed across two DCUs. Also, the LFE5UM/LFE5UM5G DCUs allow sharing the same DCU with different protocols, provided the protocols have data-rates that are multiplicity of each other.

Different combinations of protocols within a DCU are permitted subject to certain conditions. One of the most basic requirements for two protocols sharing the same DCU is that the two protocols must have data-rate that are either the same, or can be divided by the rate divider (Div1, Div2, Div11). This restriction is due to these two protocols share the same clock from the TxPLL.

A transmit clock derived from the TxPLL in DCU0 can be extended to drive the complementary neighboring DCU1 channels, providing the capability of the TxPLL of DCU0 to be used by four channels. This allows implementation of X4 link to be easily achievable.

Table 3 lists the support of mixed protocols within a DCU.

Table 3. LFE5UM/LFE5UM5G Mixed Protocol Pair

Protocol		Protocol
SGMII	&	GbE
PCIe 1.1	&	GbE
PCIe 1.1	&	SGMII
SD/HD/3G SDI	&	SD/HD/3G SDI
PCIe 1.1	&	PCIe 1.1
PCEe 2.0[1]	&	PCEe 2.0[1]
PCIe 1.1/2.0[1]	&	PCIe 1.1/2.0[1]
JESD204A/B	&	JESD204A/B

Protocol		Protocol
CPRI E6/E12	&	CPRI E6/E12
CPRI E12/E24[1]	&	CPRI E12/E24[1]
CPRI E24/E48	&	CPRI E24/E48

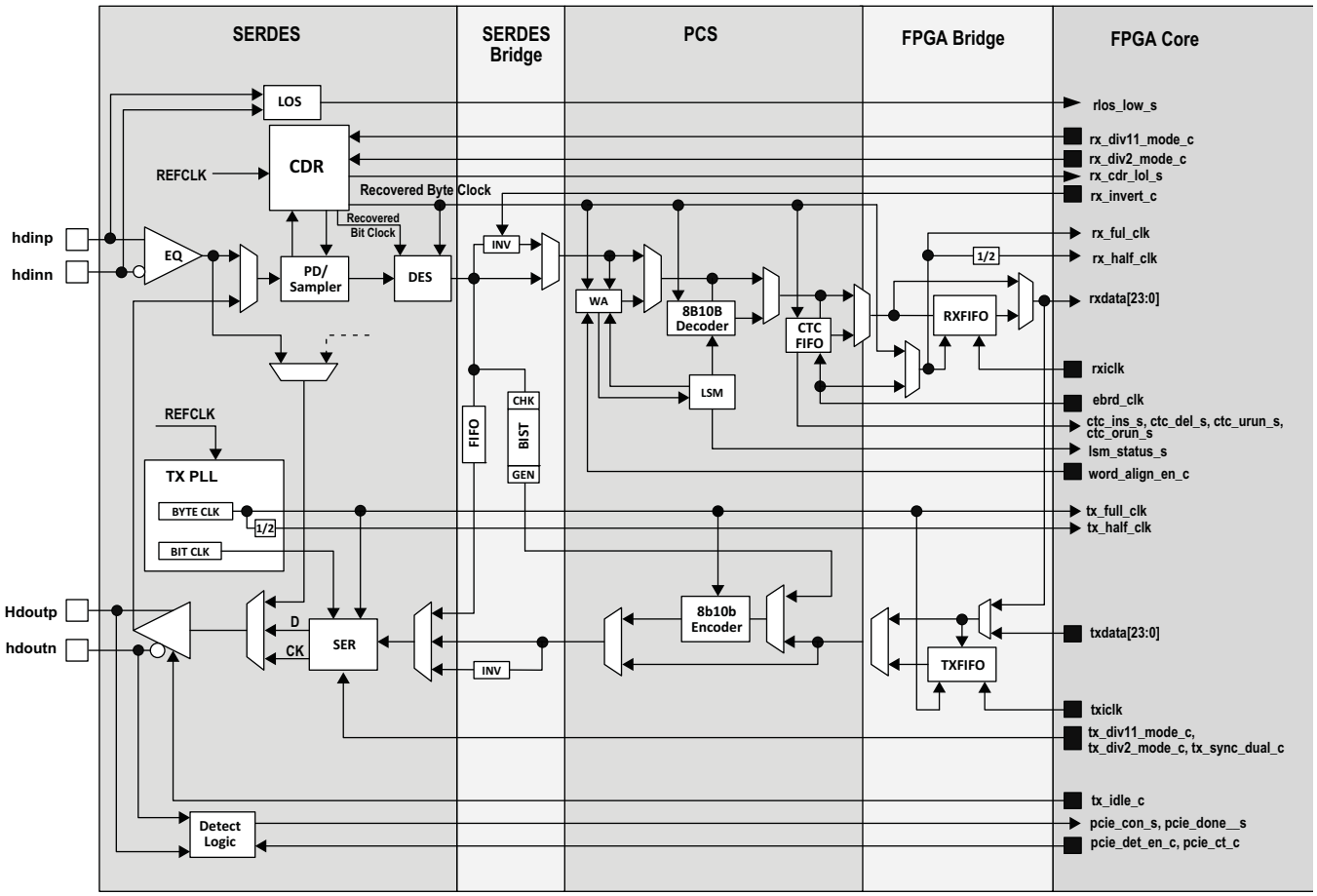
1. Supported only in ECP5-5G device family.

PCIe must be without spread spectrum to share with other protocols in the same DCU.

Detailed Channel Block Diagram

Figure 4 is a detailed block diagram representation of the major functionality in a single channel of the LFE5UM/LFE5UM5G SERDES/PCS. This diagram shows all the major blocks and the majority of the control and status signals that are visible to the user logic in the FPGA. This diagram also shows the major sub-blocks in the channel – SERDES, SERDES Bridge, PCS Core and the FPGA Bridge.

Figure 4. LFE5UM/LFE5UM5G SERDES/PCS Detailed Channel Block Diagram



Clocks and Resets

A SERDES/PCS dual supplies per-channel locked reference clocks and per-channel recovered receive clocks to the FPGA logic interface. Each dual provides clocks on both primary and secondary FPGA clock routing. The PCS/FPGA interface also has ports for the transmit clock (one per dual) and receive clocks (one per channel) supplied from the FPGA fabric.

Each dual has reset inputs to force reset of both or individual on the SERDES and PCS logic in a dual. In addition, separate resets are provided to reset the clock generation of the Rx and Tx channels. When the clock generation is reset (either in Tx or Rx, or both), the corresponding SERDES logic needs to be reset in order for the SERDES logic and PCS logic to be synchronized.

Transmit Data Bus

The signals for the transmit data path are from the FPGA to the FPGA Bridge in the PCS Block. The datapath can be geared 2:1 to the internal PCS data path, which is 8 bits wide (plus control/status signals). The highest speed of the interface for PCI Express x1 is 250 MHz in 1:1 geared mode. With 2:1 gearing (i.e. a 16-bit wide data path), the interface clock speed is 156.25 MHz (for XAUI 4x channel mode). The SERDES and PCS will support data rates up to 3.2 Gbps data that correspond to an interface speed of 160 MHz (with 2:1 gearing).

Receive Data Bus

The signals for the receive path are from the FPGA Bridge in the PCS Block to the FPGA. The data path may be geared 2:1 to the internal PCS data path which is 8 bits wide. The data bus width at the FPGA interface is 16 bits wide. When the data is geared 2:1, the lower bits (rxdata[9:0] or rxdata[7:0]) correspond to the word that has been received first and the higher bits (rxdata[19:10] or rxdata[15:8]) correspond to the word that has been received second. Table 4 describes the use of the data bus for each protocol mode.

The 2:1 gearing in LFE5UM/LFE5UM5G can be configured as user-controlled thru the FPGA logic. This allows the user to change the gearing from the core, when different rates in the protocol are selected (for example: HD-SDI and 3G-SDI)

Table 4 shows how the signals are assigned in different protocols (channel0 is shown as an example).

Table 4. Data Bus Usage by Mode

Data Bus PCS Cell Name[4]	G8B10B	CPRI	PCI Express	Gigabit Ethernet	XAUI	8-Bit SERDES	10-Bit SERDES	SDI
ff_tx_d_0_0	txdata_ch0[0]							
ff_tx_d_0_1	txdata_ch0[1]							
ff_tx_d_0_2	txdata_ch0[2]							
ff_tx_d_0_3	txdata_ch0[3]							
ff_tx_d_0_4	txdata_ch0[4]							
ff_tx_d_0_5	txdata_ch0[5]							
ff_tx_d_0_6	txdata_ch0[6]							
ff_tx_d_0_7	txdata_ch0[7]							
ff_tx_d_0_8	tx_k_ch0[0]				txc_ch0[0]	GND	txdata_ch0[8]	
ff_tx_d_0_9	tx_force_disp_ch0[0][1]			GND			txdata_ch0[9]	
ff_tx_d_0_10	tx_disp_sel_ch0[0][1]			xmit_ch0[0][2]	GND			
ff_tx_d_0_11	GND		pci_ei_en_ch0[0]	tx_disp_correct_ch0[0]	GND			
ff_tx_d_0_12	txdata_ch0[8]						txdata_ch0[10]	
ff_tx_d_0_13	txdata_ch0[9]						txdata_ch0[11]	
ff_tx_d_0_14	txdata_ch0[10]						txdata_ch0[12]	
ff_tx_d_0_15	txdata_ch0[11]						txdata_ch0[13]	
ff_tx_d_0_16	txdata_ch0[12]						txdata_ch0[14]	
ff_tx_d_0_17	txdata_ch0[13]						txdata_ch0[15]	
ff_tx_d_0_18	txdata_ch0[14]						txdata_ch0[16]	
ff_tx_d_0_19	txdata_ch0[15]						txdata_ch0[17]	

Data Bus PCS Cell Name[4]	G8B10B	CPRI	PCI Express	Gigabit Ethernet	XAUI	8-Bit SERDES	10-Bit SERDES	SDI
ff_tx_d_0_20	tx_k_ch0[1]				txc_ch0[1]	GND	txdata_ch0[18]	
ff_tx_d_0_21	tx_force_disp_ch0[1][1]			GND			txdata_ch0[19]	
ff_tx_d_0_22	tx_disp_sel_ch0[1][1]			xmit_ch0[1][2]	GND			
ff_tx_d_0_23	GND	pci_ei_en_ch0[1]	GND	GND				
ff_rx_d_0_0	rxdata_ch0[0]							
ff_rx_d_0_1	rxdata_ch0[1]							
ff_rx_d_0_2	rxdata_ch0[2]							
ff_rx_d_0_3	rxdata_ch0[3]							
ff_rx_d_0_4	rxdata_ch0[4]							
ff_rx_d_0_5	rxdata_ch0[5]							
ff_rx_d_0_6	rxdata_ch0[6]							
ff_rx_d_0_7	rxdata_ch0[7]							
ff_rx_d_0_8	rx_k_ch0[0]				rxc_ch0[0	NC	rxdata_ch0[8]	
ff_rx_d_0_9	rx_disp_err_ch0[0]	rxstatus0_ch0[0]	rx_disp_err_ch0[0]	rxdata_ch0[9]				
ff_rx_d_0_10	rx_cv_err_ch0[0][3]	rxstatus0_ch0[1]	rx_cv_err_ch0[0][3]					
ff_rx_d_0_11	NC	rxstatus0_ch0[2]	NC					
ff_rx_d_0_12	rxdata_ch0[8]	rxdata_ch0[10]						
ff_rx_d_0_13	rxdata_ch0[9]	rxdata_ch0[11]						
ff_rx_d_0_14	rxdata_ch0[10]	rxdata_ch0[12]						
ff_rx_d_0_15	rxdata_ch0[11]	rxdata_ch0[13]						
ff_rx_d_0_16	rxdata_ch0[12]	rxdata_ch0[14]						
ff_rx_d_0_17	rxdata_ch0[13]	rxdata_ch0[15]						
ff_rx_d_0_18	rxdata_ch0[14]	rxdata_ch0[16]						
ff_rx_d_0_19	rxdata_ch0[15]	rxdata_ch0[17]						
ff_rx_d_0_20	rx_k_ch0[1]	rxc_ch0[1]	NC					
ff_rx_d_0_21	rx_disp_err_ch0[1]	rxstatus1_ch0[0]	rx_disp_err_ch0[1]	rxdata_ch0[19]				
ff_rx_d_0_22	rx_cv_err_ch0[1][3]	rxstatus1_ch0[1]	rx_cv_err_ch0[1][3]					
ff_rx_d_0_23	NC	rxstatus1_ch0[2]	NC					

1. The force_disp signal will force the disparity for the associated data word on bits [7:0] to the column selected by the tx_disp_sel signal. If disp_sel is a one, the 10-bit code is taken from the 'current RD+' column (positive disparity). If the tx_disp_sel is a zero, the 10-bit code is taken from the 'current RD-' (negative disparity) column.
2. The Lattice Gigabit Ethernet PCS IP core provides an auto-negotiation state machine that generates the signal xmit. It is used to interact with the Gigabit Ethernet Idle State Machine in the hard logic.
3. When there is a code violation, the packet PCS 8b10b decoder will replace the output from the decoder with hex EE and K asserted (K=1 and d=EE is not part of the 8b10b coding space).
4. FF_TX_D_0_0: FPGA Fabric Transmit Data Bus Channel 0 Bit 0.

Control and Status Signals

Table 5 describes the control and status signals.

Table 5. Control and Status Signals and their Functions

Signal Name	Description
Transmit Control Signals	
tx_k_ch[1:0]	Per channel, active-high control character indicator.
tx_force_disp_ch[1:0]	Per channel, active-high signal which instructs the PCS to accept disparity value from disp_sel_ch[1:0] input.
tx_disp_sel_ch[1:0]	Per channel, disparity value supplied from FPGA logic. Valid when force_disp_ch[1:0] is HIGH.
tx_correct_disp_ch[1:0]	Corrects disparity identifier when asserted by adjusting the 8B10B encoder to begin in the negative disparity state.
Receive Status Signals	
rx_k_ch[1:0]	Per channel, active-high control character indicator.
rx_disp_err_ch[1:0]	Per channel, active-high signal driven by the PCS to indicate a disparity error was detected with the associated data.
rx_cv_err_ch[1:0]	Per channel, code violation signal to indicate an error was detected with the associated data.

Control Signals

Each mode has its own set of control signals which allows direct control of various PCS features from the FPGA logic. In general, each of these control inputs duplicates the effect of writing to a corresponding control register bit or bits.

{signal}_c is the control signal from the FPGA core to the FPGA bridge. All of the control signals are used asynchronously inside the SERDES/PCS.

Status Signals

Each mode has its own set of status or alarm signals that can be monitored by the FPGA logic. In general, each of these status outputs corresponds to a specific status register bit or bits. The Diamond design tools give the user the option to bring these ports out to the PCS FPGA interface.

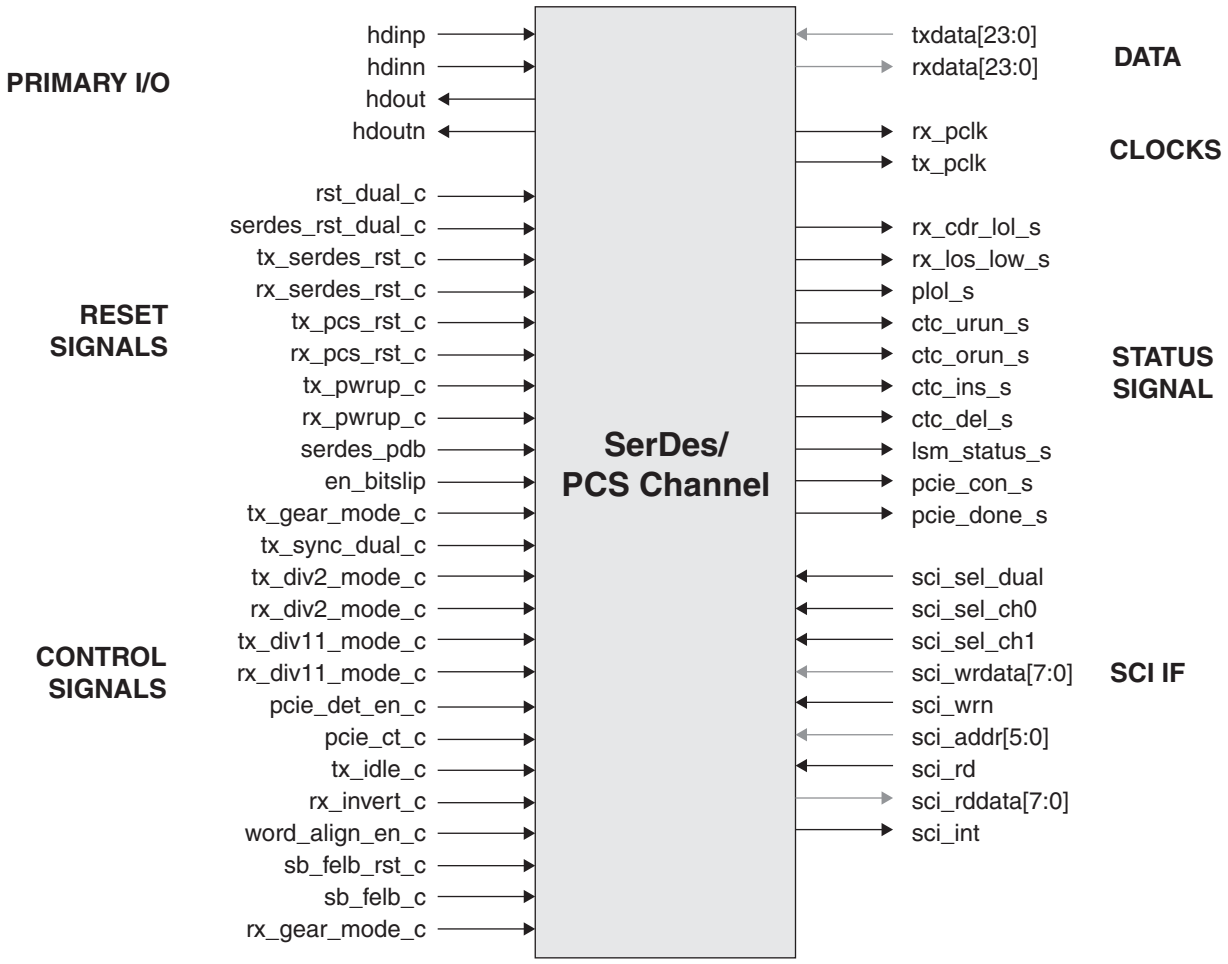
{signal}_s is the status the signal to the FPGA core from the FPGA bridge. All of the status signals are asynchronous from the SERDES/PCS. These should be synchronized to a clock domain before they are used in the FPGA design.

Please refer to the Mode-Specific Control/Status Signals section for detailed information about these signals.

SERDES/PCS

The DCU contains two channels with both RX and TX circuits, and an auxiliary channel that contains the TX PLL. The reference clock to the TX PLL can be provided either by the primary differential reference clock pins or by the FPGA core. The dual SERDES/PCS macro performs the serialization and de-serialization function for two lanes of data. In addition, the TxPLL within the DCU provides the system clock for the FPGA logic. The dual also supports both full-data-rate and half-data-rate modes of operation on each TX and RX circuit independently. The block-level diagram is shown in Figure 5.

Figure 5. SERDES/PCS Block Signal Interface



I/O Descriptions

Table 6 lists all default and optional inputs and outputs to/from a dual.

Table 6. SERDES/PCS I/O Descriptions

Signal Name	I/O	Type	Description
Primary I/O, SERDES Dual			
hdinp	I	CH	High-speed input, positive
hdinn	I	CH	High-speed input, negative
hdoutp	O	CH	High-speed output, positive
hdoutn	O	CH	High-speed output, negative
Transmit/Receive Data Bus (See Table 3 and Table 4 for detailed data bus usage)			
rxdata[23:0]	O	CH	received data
txdata[23:0]	I	CH	transmit data
Control Signals			
tx_sync_dual_c	I	DL	Serializer Reset Rising edge Transition = Reset Level = Normal Operation

Signal Name	I/O	Type	Description
tx_div2_mode_c	I	CH	Transmit Rate Mode (Full Rate/Half Rate) Select 1 = Half Rate 0 = Full Rate
rx_div2_mode_c	I	CH	Receive Rate Mode (Full Rate/Half Rate) Select 1 = Half Rate 0 = Full Rate
tx_div11_mode_c	I	CH	Transmitter Rate Mode Select (SDI protocol only) 1: Div11 Rate 0: Full Rate
rx_div11_mode_c	I	CH	Receiver Rate Mode Select (SDI protocol only) 1: Div11 Rate 0: Full Rate
pcie_det_en_c	I	CH	FPGA logic (user logic) informs the SerDes block that it will be requesting for a PCI Express Receiver Detect operation 1 = Enable PCI Express Receiver Detect
pcie_ct_c	I	CH	1 = Request transmitter to do far-end receiver detection 0 = Normal data operation
tx_idle_c	I	CH	0 = Normal operation
rx_invert_c	I	CH	Control the inversion of received data. 1 = Invert the data 0 = Do not invert the data
tx_gear_mode_c	I	CH	Transmit Gearing Mode Select (OR'ed with tx_data_width setting. Use with tx_data_width = 0) 1 = 16/20-Bit Mode 0 = 8/10-Bit Mode
rx_gear_mode_c	I	CH	Receiver Gearing Mode Select (OR'ed with rx_data_width setting. Use with rx_data_width = 0) 1 = 16/20-Bit Mode 0 = 8/10-Bit Mode
word_align_en_c	I	CH	Control comma aligner. 1 = Enable comma aligner 0 = Lock comma aligner at current position
sb_felb_ c	I	CH	SerDes Bridge Parallel Loopback 1 = Enable loopback from RX to TX 0 = Normal data operation
sb_felb_rst_c	I	CH	SerDes Bridge Parallel Loopback FIFO Clear 1 = Reset Loopback FIFO 0 = Normal loopback operation
en_bitslip	I	CH	SerDes Word Alignment to shift byte clock by 1 UI 1 = Slip Rx byte clock by 1 UI for Word Alignment 0 = No slip
Status Signals			
rx_cdr_lol_s	O	CH	1 = Receive CDR Loss of Lock 0 = Lock maintained
rx_los_low_s	O	CH	Loss of Signal (LO THRESHOLD RANGE) detection for each channel.
ctc_urun_s	O	CH	1 = Receive clock compensator FIFO underrun error 0 = No FIFO errors.
ctc_orun_s	O	CH	1 = Receive clock compensator FIFO overrun error 0 = No FIFO errors.
ctc_ins_s	O	CH	1 = SKIP Character Added by CTC
ctc_del_s	O	CH	1 = SKIP Character Deleted by CTC

Signal Name	I/O	Type	Description
lsm_status_s	O	CH	1 = Lane is synchronized to commas 0 = Lane has not found comma
pcie_con_s	O	CH	Result of far-end receiver detection 1 = Far end receiver detected 0 = Far end receiver not detected
pcie_done_s	O	CH	1 = Far-end receiver detection complete 0 = Far-end receiver detection incomplete
rst_dual_c	I	DL	Active high, asynchronous input. Resets all SerDes channels including the auxiliary channel and PCS.
serdes_rst_dual_c	I	DL	Active high, asynchronous input to SerDes dual. Gated with software register bit fpga_reset_enable. By default fpga_reset_enable is '1'.
tx_serdes_rst_c	I	DL	Resets LOL signal in PLL
rx_serdes_rst_c	I	CH	Resets selected digital logic in the SerDes Receive channels
tx_pcs_rst_c	I	CH	Active high, asynchronous input. Resets individual Dual TX channel logic only in PCS.
rx_pcs_rst_c	I	CH	Active high, asynchronous input. Resets individual Dual RX channel logic only in PCS.
serdes_pdb	I	DL	Power down control for entire SerDes dual including AUX and channels. 0 = Power down 1 = Power up
tx_pwrup_c	I	CH	Active low Transmit Channel Power Down. 0 = Transmit Channel should be powered down
rx_pwrup_c	I	CH	Active low Receive Channel Power Down. 0 = Receive Channel should be powered down
Clocks			
rx_pclk	O	CH	Receive Channel Recovered Primary Clock. Direct connection to Primary clock network. When gearing is not enabled, this output equals the receive full rate clock. When 2:1 gearing is enabled, it is a divide-by-2 half-rate clock.
tx_pclk	O	CH	Transmit Primary Clock. Direct connection to Primary clock network. When gearing is not enabled, this output equals the transmit full rate clock. When 2:1 gearing is enabled, it is a divide-by-2 half-rate clock.
SCI Interface Signals			
sci_sel_dual	I	DL	sci dual select
sci_sel_ch0	I	CH	sci channel0 select
sci_sel_ch1	I	CH	sci channel1 select
sci_wrdata[7:0]	I	CH	sci write data
sci_rddata[7:0]	O	CH	sci read data bus
sci_wrn	I	CH	sci write strobe
sci_rd	I	CH	sci read data select
sci_addr[5:0]	I	CH	sci address bus
sci_int	O	CH	sci interrupt output

SERDES/PCS Functional Description

LFE5UM/LFE5UM5G devices have one to two duals of embedded SERDES/PCS logic. Each dual, in turn, supports two independent full-duplex data channels. A single channel can support a data link and each dual can support up to two such channels.

The embedded SERDES CDR PLLs and TX PLLs support data rates that cover a wide range of industry standard protocols.

Refer to Figure 4 for each of the items below.

- **SERDES Buffers**
  - Output Driver
  - Differential receiver
- **SERDES**
  - Linear Equalizer (LEQ)
  - CDR (Clock and Data Recovery)
  - Deserializer
  - De-emphasis
  - Serializer
  - Two Serial Loopback modes, TX-to-RX or RX-to-TX
- **SERDES Bridge (SB)**
  - Inverter: Inverts receive data, required by PCI Express
  - SERDES Bridge Parallel Loopback
- **PCS Core**
  - Word Alignment
  - 8b10b Decoder
  - 8b10b Encoder
  - Link State Machine
  - Clock Tolerance Compensation
- **FPGA Bridge (FB)**
  - Downsample FIFO
  - Upsample FIFO

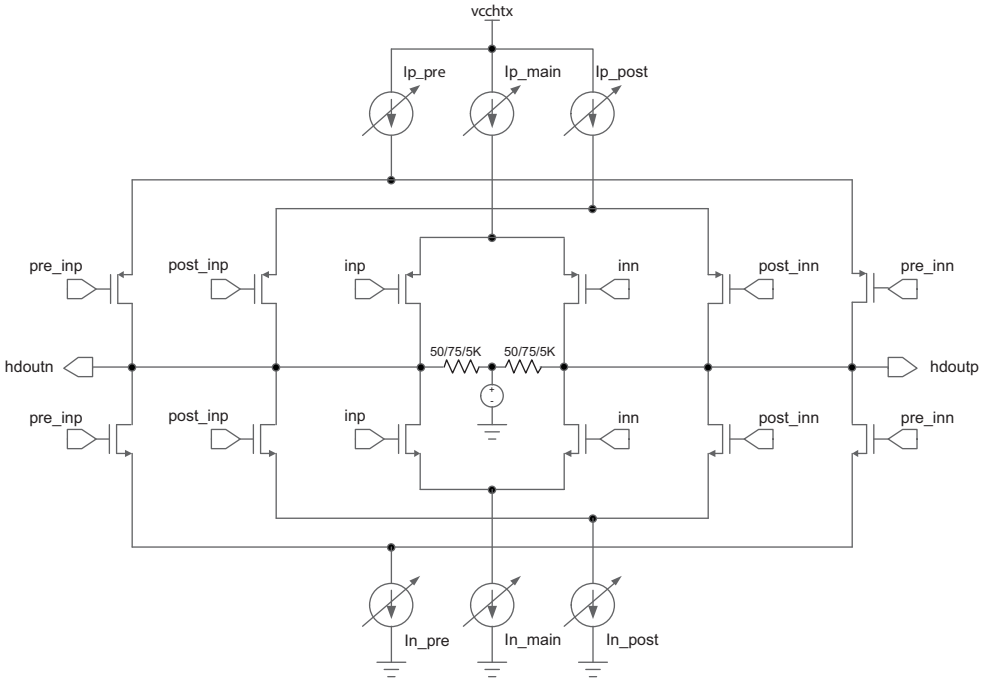
SERDES Buffers

Tx Differential Output Driver

The TX output buffer is a high speed line driver, and is used to send serialized data off-chip. It includes functions such as variable amplitude settings, variable termination resistance values, and various pre/post-cursor De-emphasis settings. TX output buffer has a H-Bridge structure as shown in Figure 6. This provides lower power consumption to reduce overall TX power. The output voltage swing is maintained by having separate power supply pad for driver only (VCCHTX). PCI Express requirements for electrical idle and receiver detection is implemented in TX driver and output pads.

Figure 6 shows the Transmitter H-Bridge Driver with the differential termination, pre- and post- cursor and associated current sources.

Figure 6. Transmitter H-Bridge Driver with Termination and De-empahsis



SERDES

Linear Equalizer

The Linear Equalizer (LEQ) is used to equalize the channel loss. Generally this is used to selectively amplify the high frequency components more which are attenuated more over a long backplane– similar to the function performed by De-emphasis on the transmitter. As the data rate of the transmission advances over multi-Gpbs, frequency-dependent attenuation can cause severe InterSymbol Interference (ISI) in the receive signal. LEQ performs the recovery of the signal from the interference. Four levels of Equalization can be selected, based on the user transmission channel conditions.

De-Emphasis

De-emphasis refers to a system process designed to increase the magnitude of some frequencies with respect to the magnitude of other frequencies. De-emphasis improves the overall signal-to-noise ratio by minimizing the adverse effects of such phenomena as attenuation differences.

LFE5UM/LFE5UM5G SERDES Transmit channel provides +/– one tap for de-emphasis (pre-cursor and post cursor). Both taps can be programmed individually, to provide 12 settings in each.

Reference Clocks

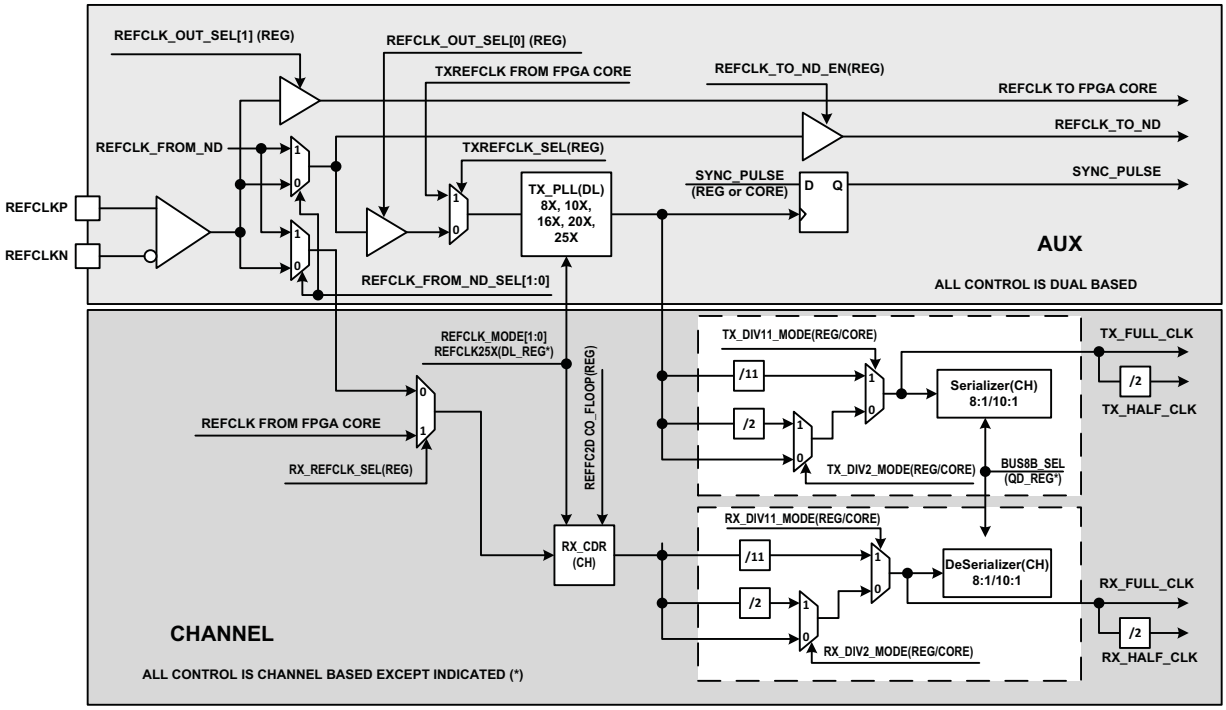
The SERDES dual contains two channels with both RX and TX circuits, and an auxiliary channel that contains the TX PLL. The reference clock to the TX PLL can be provided either by the primary differential reference clock pins, by the neighbor dual's reference clock or by the FPGA core. In addition, the PLL within the SERDES block provides an output clock that can be used as the system clock driving the FPGA fabric.

The reference clock to the RX can be provided either by the reference clock to the TX PLL or by the FPGA core. The reference clocks from the FPGA core to the TX PLL and to the RX may come from different sources.

SERDES Clock Architecture

Figure 7 shows the clocking structure inside a DCU. There is one Aux channel in each DCU. There are two Tx/Rx channels. Various control bits for the different blocks are shown. These could be dual-based control register bits or channel-based control register bits. In some cases, it can be channel control port based. Some are a combination of both register and control ports. Using both modes enables dynamic control of certain functional properties.

Figure 7. SERDES Clock Architecture



The important components of the LFE5UM/LFE5UM5G SERDES clocking architecture include:

- Per Rx and Tx diver (DIV) modes – DIV1, DIV2, DIV11. Suitable for dynamic rate switching.
- REFCLK sharing between DCUs – REFCLK input from DCU0 can be shared by DCU1. Used in x4 link application.
- Multi-channel transmit synchronization using the tx_sync_dual_c signal form the FPGA

Rate Modes

The TX in each channel can be independently programmed to run at either FULL_RATE, HALF_RATE (DIV2), or DIV11 mode.

The RX can also use a separate reference clock per channel, allowing the transmitter and receiver to run at completely different rates.

It is important to note that the PLL VCO is left untouched, supporting the highest rate for that protocol. All divided rates required by that protocol are supported by programming the divider mux selects. This enables very quick data rate change over since the PLL does not need to be reprogrammed. This is valuable in many applications.

It should be noted when Rx DIV selection is changed dynamically from the core, the Rx of the channels should be reset. The DIV is inside the CDR lock loop, and change of the value requires the loop to be re-locked again.

The TX PLL and the two CDR PLLs generally run at the same frequency, which is a multiple of the reference clock frequency. Table 7 illustrates the various clock rate modes possible. The bit clock indicated is a multiple of the reference clock frequency.

Table 7. TXPLL and RX CDRPLL Supported Modes

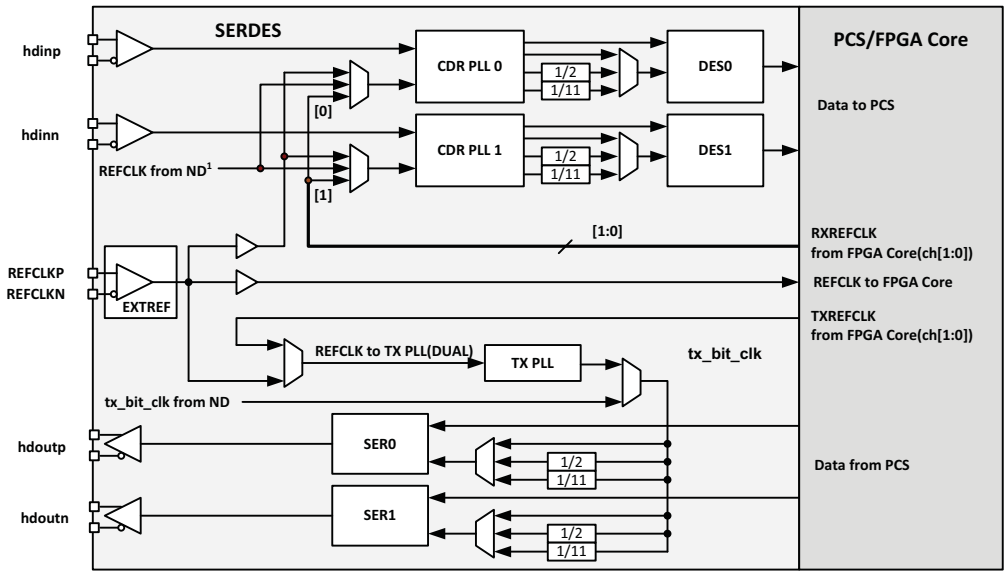
Reference Clock Mode	refclk_mode(Dual)	Bus_Width	Bit Clock(Full)	Bit Clock (div2, div11)
20x	0	10	Refclk x 20	Refclk x 10
16x	0	8	Refclk x 16	Refclk x 8
10x	1	10	Refclk x 10	Refclk x 5
8x	1	8	Refclk x 8	Refclk x 4
25x	-	8	Refclk x 25	Refclk x 12.5
25x	-	10	Refclk x 25	Refclk x 12.5
20x	0	10	Refclk x 20	Refclk x 20/11[1]
1. DIV11 mode.				

Reference Clock from FPGA Core

As shown in Figure 8, the TX reference clock can be brought from the FPGA core. In this case, the extra jitter caused by the FPGA resources to route the clock to the SERDES will be passed onto the transmit data and may violate TX jitter specifications on a case-by-case basis. Caution should be used when using an FPGA-originated SERDES TX reference clock.

The Rx reference clock can also be brought from the FPGA core. Each Rx channel can have independent RxRefClk signal. The Rx reference clock is used in the channel's CDR to acquire initial frequency lock, before switches to lock to the data. High jitter on RxRefClk when is brought in from FPGA can cause difficulty for Rx to obtain initial frequency lock. It can also cause the Rx channel RLOL (Rx Loss Of Lock) signal to go H because the Loss-of-Lock detection circuit compares the recover clock with the RxRefClk.

Figure 8. Reference Clock Block Diagram



Full Data, Div 2 and Div 11 Data Rates

Each TX Serializer and RX Deserializer can be split into full data rate and div2 rate or div11 rate depending on the protocol, allowing for different data rates in each direction and in each channel. Refer to Figure 8 for more information.

Reference Clock Sources

refclkp, refclkn

Dedicated LVDS input. This clock source is the preferred choice unless different clock sources for RX and TX are used. The input can interface to different electrical standards, such as CML, LVDS, or LVPECL.

FPGA txrefclk, rxrefclk

Reference clock from FPGA logic. This clock signal can be routed from FPGA I/O pins, or from FPGA sysCLOCK PLL. In either case, the PCLK clock tree should be used to route the clock to minimize FPGA core noise coupling. When the clock is routed from the pin, the shared PCLK pins should be used to directly connect the pin to the PCLK clock tree. The clock input on the pin can be LVDS or single-ended.

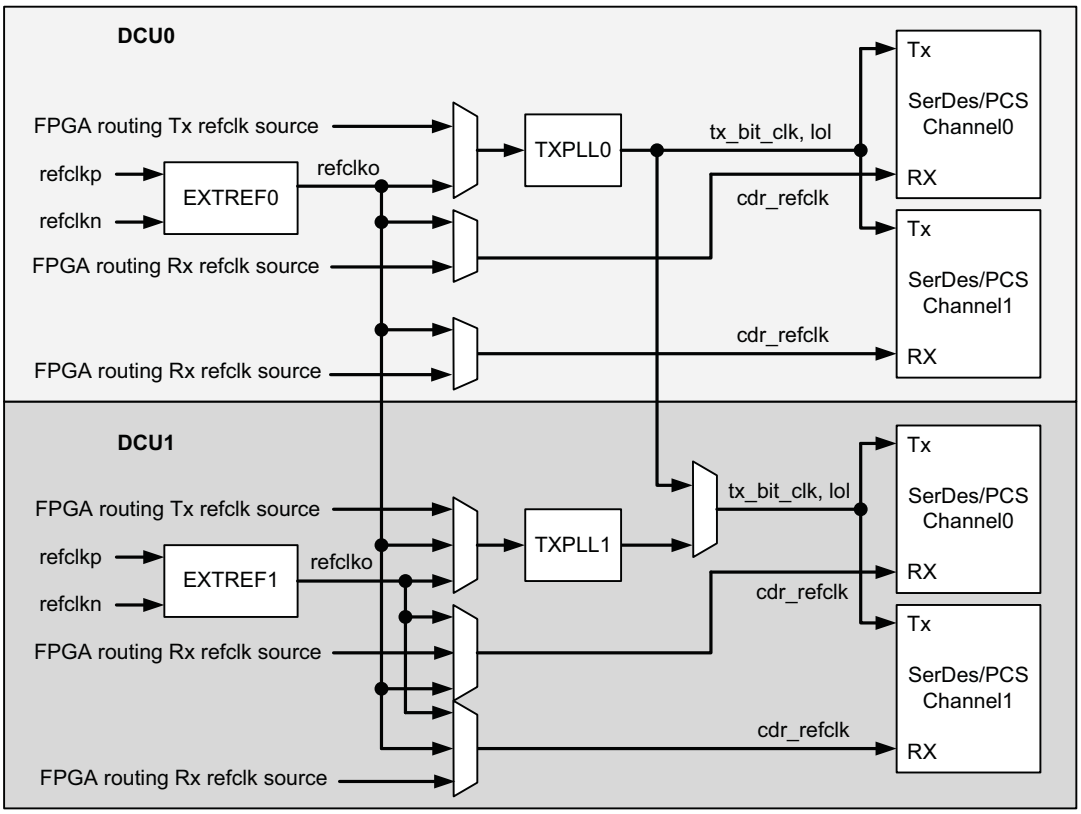
When an FPGA PLL is used as the reference clock, the reference clock to the PLL should be assigned to a dedicated PLL input pad. The FPGA PLL output jitter may not meet system specifications at higher data rates. Use of an FPGA PLL is not recommended in jitter-sensitive applications.

Clock Distribution in Complementary DCUs

Figure 9 shows the clock distribution for two DCU devices. When x4 link is used, the LFE5UM/LFE5UM5G SERDES provides sharing of the High Speed bit clocks between the two DCUs, This minimizes Transmit skew between the channels residing in different DCUs.

Also shown in the diagram is sharing of RefClk. This sharing of RefClk allow the Rx channels to use the same clock source, without having to route to two different REFCLK pins externally.

Figure 9. Clock Distribution Diagram for a Two Complementary DCU Pair



Spread Spectrum Clocking (SSC) Support

The ports on the two ends of Link must transmit data at a rate that is within 600pm of each other at all times. This is specified to allow bit-rate clock source with a +/- 300ppm tolerance. The minimum clock period cannot be violated. The preferred method is to adjust the spread technique to not allow for modulation above the nominal frequency. The data rate can be modulated from +0% to -0.5% of the nominal data rate frequency, at a modulation rate in the range not exceeding 30KHz to 33KHz. Along with the +/- 300ppm tolerance limit, both ports require the same bit rate clock when the data is modulated with an SSC.

In PCI Express applications, the root complex is responsible for spreading the reference clock. The endpoint will use the same clock to pass back the spectrum through the TX. Thus, there is no need for a separate RXREFCLK. The predominant application is an add-in card. Add-in cards are not required to use the REFCLK from the connector but must receive and transmit with the same SSC as the PCI Express connector REFCLK.

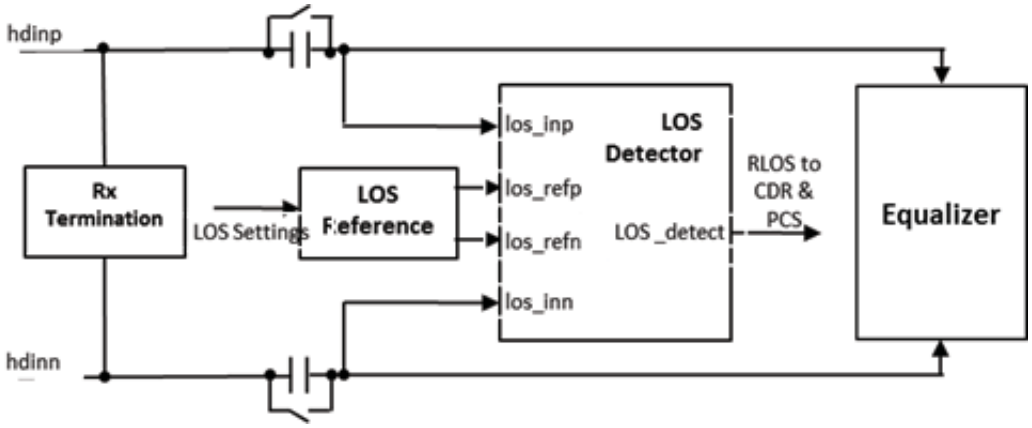
While the LFE5UM/LFE5UM5G architecture will allow the mixing of a PCI Express channel and a Gigabit Ethernet, Serial RapidIO or SGMII channel within the same dual, using a PCI Express SSC as the transmit reference clock will cause a violation of the Gigabit Ethernet, Serial RapidIO and SGMII transmit jitter specifications.

Loss of Signal

Each channel contains a Loss of Signal (LOS) detector circuit at receiver input as shown in Figure 10. LOS detector picks up the data signal from channel receiver input, and compares its amplitude with a threshold voltage. If the input signal amplitude is lower than the threshold voltage, LOS detector issues loss of signal output.

Detected LOS indicates that amplitude of input signal is too low. In that case receiver CDR may not be able to recover data from incoming signal. It is a good indication of any error in Channel link. The timing of absent signal determines meaning of handshaking between link endpoints.

Figure 10. Loss of Signal Detector connection to Receiver Inputs



LOS detection is affected by data rate, data pattern and channel degradation. LOS block shares the signal in the signal path to the Equalizer

Table 8. Response Time for Loss of Signal Detector

Description	Min.	Typ.	Max.	Units
Time to detect that signal si lost (rx_los_low 0 to 1)	-	8	10	ns
Time to detect that signal si present (rx_los_low 1 to 0)	-	8	10	ns

Loss Of Lock

Both the transmit PLL and the individual channel CDRs have digital counter-based, loss-of-lock detectors.

If the transmit PLL loses lock, the loss-of-lock for the PLL is asserted and remains asserted until the PLL reacquires lock. When the PLL loses lock, it is likely to be caused by a reference clock problem.

If a CDR loses lock, the loss-of-lock for that channel is asserted and locking to the reference clock retrains the DCO in the CDR. When the DCO re-training is achieved, loss-of-lock for that channel is de-asserted and the CDR is switched back over to lock to the incoming data. The CDR will either remain locked to the data, or will go back out of lock again in which case the re-training cycle will repeat. For detailed information on CDR loss-of-lock, please refer to the SERDES/PCS Reset section.

Table 9. Response Time for Loss-of-Lock Detector

Description	Min.	Typ.	Max.	Units
Time to detect that loop is unlocked (tx_pll_lol, rx_cdr_lol goes from 0 to 1)	-	200	500	us
Time to detect that loop is locked (tx_pll_lol, rx_cdr_lol goes from 1 to 0)	-	200	500	us

TX Lane-to-Lane Skew

Most multi-channel protocol standards require TX lane-to-lane skew is within a certain specification. A control signal in the Transmitter (that can be controlled either by a register bit or by a FPGA signal), tx_sync_dual_c, resets all the active Tx channel to start serialization with bit0.

Transmit Data

The PCS dual transmit data path consists of an 8b10b encoder and serializer per channel.

8b10b Encoder

This module implements an 8b10b encoder as described within the IEEE 802.3ae-2002 1000BASE-X specification. The encoder performs the 8-bit to 10-bit code conversion as described in the specification, along with maintaining the running disparity rules as specified. The 8b10b encoder can be bypassed on a per-channel basis by setting the attribute CHx_8B10B to “BYPASS” where x is the channel number.

Serializer

The 8b10b encoded data undergoes parallel-to-serial conversion and is transmitted off-chip via the embedded SERDES.

Receive Data

The PCS dual receive data path consists of the following sub-blocks per channel: Deserializer, Word Aligner, 8b10b Decoder, Optional Link State Machine, and Optional Receive Clock Tolerance Compensation (CTC) FIFO.

Deserializer

Data is brought on-chip to the embedded SERDES where it goes from serial to parallel.

Word Alignment (Byte Boundary Detect)

This module performs the comma code word detection and alignment operation. The comma character is used by the receive logic to perform 10-bit word alignment upon the incoming data stream. The comma description can be found in section 36.2.4.9 of the 802.3.2002 1000BASE-X specification.

A number of programmable options are supported within the word alignment module:

- Word alignment control from either embedded Link State Machine (LSM) or from FPGA control. Supported in 8-bit SERDES Only, 10-bit SERDES Only and SDI modes, in addition to 8b10b packet mode.
- Ability to set two programmable word alignment characters (typically one for positive and one for negative disparity) and a programmable per bit mask register for alignment comparison. Alignment characters and the mask register is set on a per quad basis. For many protocols, the word alignment characters can be set to “XX00000011” (jhgfiedcba bits for positive running disparity comma character matching code groups K28.1, K28.5, and K28.7) and “XX01111100” (jhgfiedcba bits for negative running disparity comma character matching code groups K28.1, K28.5, and K28.7). However, the user can define any 10-bit pattern.
- The first alignment character is defined by the 10-bit value assigned to attribute COMMA_A. This value applies to all channels in a PCS quad.
- The second alignment character is defined by the 10-bit value assigned to attribute COMMA_B. This value applies to all channels in a PCS quad. • The mask register defines which word alignment bits to compare (a ‘1’ in a bit of the mask register means check the corresponding bit in the word alignment character register). The mask registers defined by the 10-bit value assigned to attribute COMMA_M. This value applies to all channels in a PCS quad. When the attribute CHx_RXWA (word alignment) is set to “ENABLED” and CHx_ILSM (Internal Link State Machine) is set to “ENABLED”, one of the protocol-based Link State Machines will control word alignment. For more information on the operation of the protocol-based Link State Machines, see the Protocol-Specific Link State Machine section below.

8b10b Decoder

The 8b10b decoder implements an 8b10b decoder operation as described with the IEEE 802.3-2002 specification. The decoder performs the 10-bit to 8-bit code conversion along with verifying the running disparity. When a code violation is detected, the receive data, rxdata, is set to 0xEE with rx_k set to ‘1’.

External Link State Machine Option

When the attribute CHx_RXLSM (Internal Link State Machine) is set to DISABLED, and CHx_RXWA (word alignment) is set to ENABLED, the control signal word_align_en_ch[1:0]_c is used to enable the word aligner. This signal should be sourced from a FPGA external Link State Machine implemented in the FPGA fabric. When the word_align_en_ch[1:0]_c is high, the word aligner will lock alignment and stay locked. It will stop comparing incoming data to the user-defined word alignment characters and will maintain current alignment on the first successful compare to either COMMA_A or COMMA_B. If re-alignment is required, pulse the word_align_en_ch[1:0]_c low to high. The word aligner will re-lock on the next match to one of the user-defined word alignment characters. If desired, word_align_en_ch[1:0]_c can be controlled by a Link State Machine implemented externally to the PCS quad to allow a change in word alignment only under specific conditions.

When a Link State Machine is selected and enabled for a particular channel, that channel's lsm_status_ch[1:0]_s status signal will go high upon successful link synchronization.

Idle Insert for Gigabit Ethernet Mode

This is required for Clock Compensation and Auto Negotiation (the latter is done in FPGA logic)

This module automatically inserts /I2/ symbols into the receive data stream during auto-negotiation. Whilst auto-negotiating, the link partner will continuously transmit /C1/ and /C2/ ordered sets. The clock-compensator will not delete these ordered sets as it is configured to only insert/delete /I2/ ordered sets. In order to prevent overruns and underruns in the clock-compensator, /I2/ ordered sets must be periodically inserted to provide insertion/deletion opportunities for the clock compensator.

Whilst performing auto-negotiation, this module will insert a sequence of 8 /I2/ ordered sets (2 bytes each) every 2048 clock cycles. As this module is after the 8b10b decoder, this operation will not introduce any running disparity errors. These /I2/ ordered sets will not be passed on to the FPGA receive interface as the GMII interface is driven to IDLE by the Rx state machine during auto-negotiation. Once auto-negotiation is complete, /I2/ insertion is disabled to prevent any corruption of the received data.

Note that this state machine is active only during auto negotiation. The auto negotiation state machine and the GigE receive state machines are implemented in the soft logic. This state machine depends on the signal "xmit" from the auto negotiation state machine. This signal is provided on the TX data bus. Even though this signal is relatively static (especially after auto negotiation is done) it is currently decided to put this in TX data bus. It provides a signal – "rx_even_out". This will be sent out on the receive data bus to the FPGA logic.

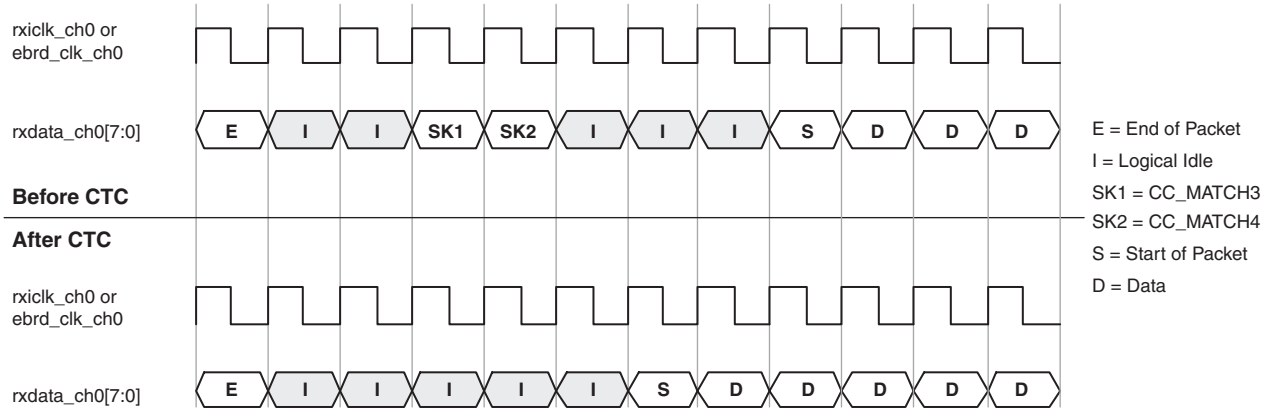
Clock Tolerance Compensation (CTC)

The Clock Tolerance Compensation (CTC) module performs clock rate adjustment between the recovered receive clocks and the locked reference clock. Clock compensation is performed by inserting or deleting bytes at predefined positions, without causing loss of packet data. A 16-byte CTC FIFO is used to transfer data between the two clock domains and will accommodate clock differences of up to the specified ppm tolerance for the LFE5UM/LFE5UM5G SERDES (see DC and Switching Characteristics section of DS1044, ECP5 and ECP5-5G Family Data Sheet).

A channel has the Clock Tolerance Compensation block enable when that channel's attribute CHx_CTC is set to "ENABLED". The CTC is bypassed when that channel's attribute CHx_CTC is set to "DISABLED". The following diagrams show 2- and 4-byte deletion.

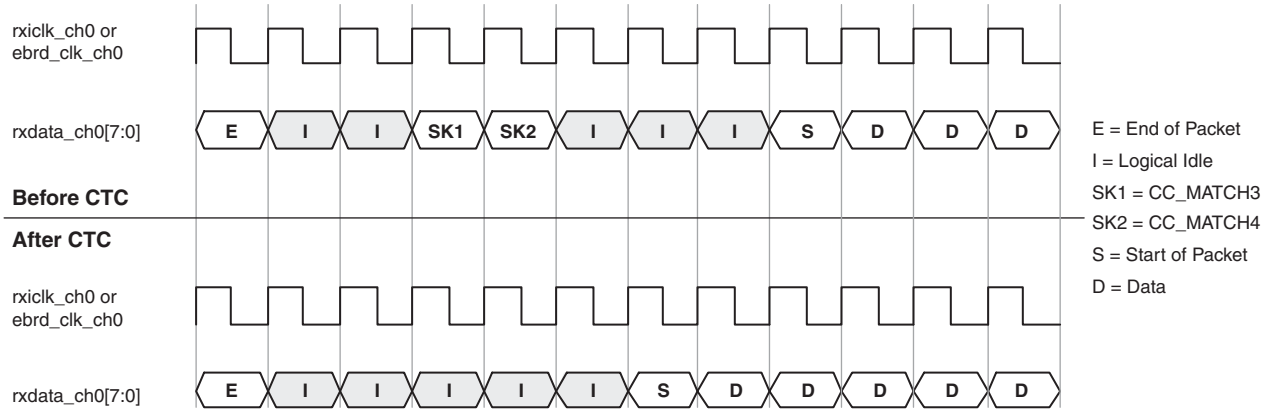
A diagram illustrating 2-byte deletion is shown in Figure 11.

Figure 11. Clock Tolerance Compensation 2-Byte Deletion Example



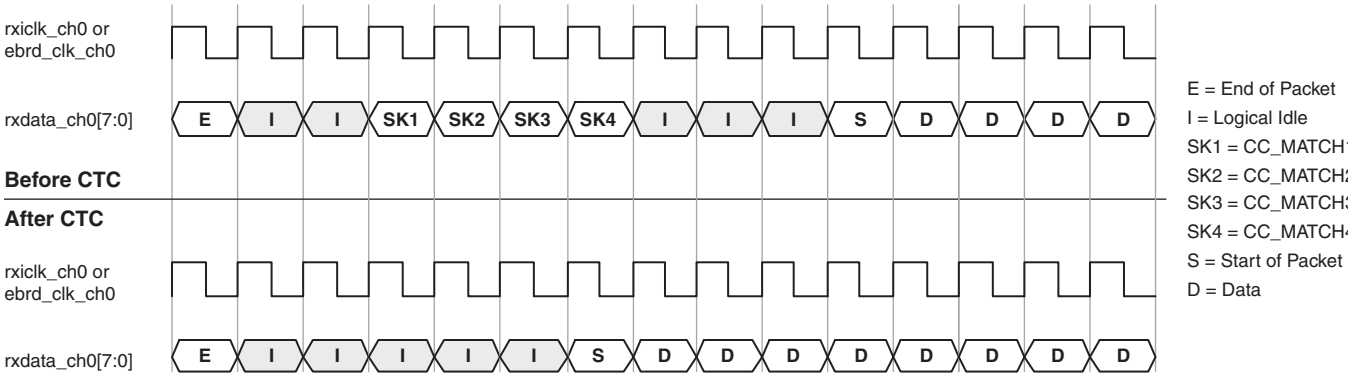
A diagram illustrating 2-byte insertion is shown in Figure 12.

Figure 12. Clock Tolerance Compensation 2-Byte Insertion Example



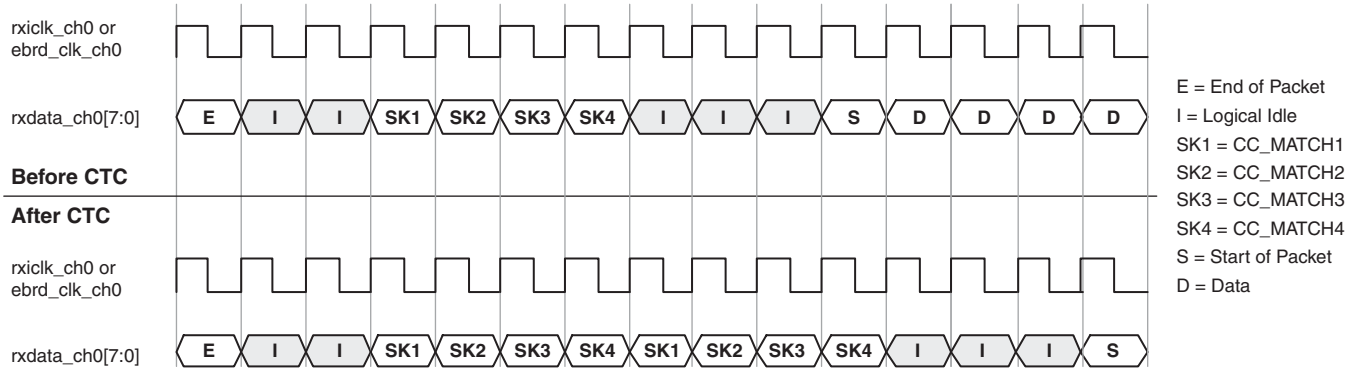
A diagram illustrating 4-byte deletion is shown in Figure 13.

Figure 13. Clock Tolerance Compensation 4-Byte Deletion Example



A diagram illustrating 4-byte insertion is shown in Figure 14.

Figure 14. Clock Tolerance Compensation 4-Byte Insertion Example



When the CTC is used, the following settings for clock compensation must be set, as appropriate, for the intended application:

- **Set the insertion/deletion pattern length using the CC_MATCH_MODE attribute.** This sets the number of skip bytes the CTC compares to before performing an insertion or deletion. Values for CC_MATCH_MODE are “2” (2-byte insertion/deletion) and “4” (4-byte insertion/deletion). The minimum interpacket gap must also be set as appropriate for the targeted application. The interpacket gap is set by assigning values to attribute CC_MIN_IPG. Allowed values for CC_MIN_IPG are “0”, “1”, “2”, and “3”. The mini-mum allowed interpacket gap after skip character deletion is performed based on these attribute settings is described in Table 10.
- **The skip byte or ordered set must be set corresponding to the CC_MATCH_MODE chosen.** For 4-byte insertion/deletion (CC_MATCH_MODE = “4”), the first byte must be assigned to attribute CC_MATCH1, the second byte must be assigned to attribute CC_MATCH2, the third byte must be assigned to attribute CC_MATCH3, and the fourth byte must be assigned to attribute CC_MATCH4. Values assigned are 10-bit binary values.

  For example:

  If a 4-byte skip ordered set is /K28.5/D21.4/D21.5/D21.5, then “CC_MATCH1” should be “0110111100”, “CC_MATCH2” = “0010010101”, “CC_MATCH3” = “0010110101” and “CC_MATCH4” = “0010110101”.

  For a 2-byte insertion/deletion (CC_MATCH_MODE = “2”), the first byte must be assigned to attribute CC_MATCH3, and the second byte must be assigned to attribute CC_MATCH4.

- **The clock compensation FIFO high water and low water marks must be set to appropriate values for the targeted protocol.** Values can range from 0 to 15 and the high water mark must be set to a higher value than the low water mark (they should not be set to equal values). The high water mark is set by assigning a value to attribute CCHMARK. Allowed values for CCHMARK are hex values ranging from “0” to “F”. The low water mark is set by assigning a value to attribute CCLMARK. Allowed values for CCLMARK are hex values ranging from “0” to “F”.
- Clock compensation FIFO overrun can be monitored on a per-channel basis on the PCS/FPGA interface port labeled cc_overrun_ch(0-3) if “Error Status Ports” is selected when generating the PCS block with the module generator.
- Clock compensation FIFO underrun can be monitored on a per-channel basis on the PCS/FPGA interface port labeled cc_underrun_ch(0-3) if “Error Status Ports” is selected when generating the PCS block with the module generator.

Calculating Minimum Interpacket Gap

Table 10 shows the relationship between the user-defined values for interpacket gap (defined by the CC_MIN_IPG attribute), and the guaranteed minimum number of bytes between packets after a skip character deletion from the PCS. The table shows the interpacket gap as a multiplier number. The minimum number of bytes between packets is equal to the number of bytes per insertion/deletion times the multiplier number shown in the table. For example, if the number of bytes per insertion/deletion is 4 (CC_MATCH_MODE is set to “4”), and the minimum interpacket gap attribute CC_MIN_IPG is set to “2”, then the minimum interpacket gap is equal to 4 (CC_MATCH_MODE = “4”) times 3 (Table 8-10 with CC_MIN_IPG = “2”) or 12 bytes. The PCS will not perform a skip character deletion until the minimum number of interpacket bytes have passed through the CTC.

Table 10. Minimum Interpacket Gap Multiplier

CC_MIN_IPG	Insertion/Deletion Multiplier Factor
0	1X
1	2X
2	3X
3	4X

Protocol Modes

Generic 8b10b Mode

The Generic 8b10b mode of the SERDES/PCS block is intended for applications requiring 8b10b encoding/decoding without the need for additional protocol-specific data manipulation. The LFE5UM/LFE5UM5G SERDES/PCS block can support Generic 8b10b applications up to 3.2 Gbps per channel in ECP5, and 5 Gbps per channel in ECP5-5G. In Generic 8b10b mode, the word aligner can be controlled from the embedded PCS Link State Machine (LSM).

When the embedded Link State Machine is selected and enabled, the lsm_status_ch[3:0]_s status signal will go high upon successful link synchronization.

To achieve link synchronization in this mode, the following conditions need to be satisfied on the 8b10b patterns received at the SERDES channel’s receiver input (hdinp_ch[0-3]/hdinn_ch[0-3]):

- Periodic 8b10b encoded comma characters need to be present in the serial data. Periodicity is required to allow the LSM to re-synchronize to commas upon loss of sync. The comma characters should correspond to the “Specific Comma” value shown below. For example, when the Specific Comma is set to K28P157, the comma value on the serial link can be the 8b10b encoded version of any of K28.1 (k=1, Data=0x3C), K28.5 (k=1, Data=0xBC), or K28.1 (k=1, Data=0xFC). Note though that K28.5 is most commonly used.
- A comma character has to be followed by a data character
- Two comma characters have to be an even number of word clock cycles apart Additional information:
- It takes roughly four good comma/data pairs for the LSM to reach link synchronization
- It takes four consecutive errors (illegal 8b10b encoded characters, code violations, disparity errors, uneven number of clock cycles between commas) to cause the LSM to unlock
- A CDR loss of lock condition will cause the LSM to unlock by virtue of the many code violation and disparity errors that will result
- When the internal reset sequence state machine is used, a CDR loss of lock (rx_cdr_lol_ch[3:0]_s) or a loss of signal (rx_los_low_ch[3:0]_s) condition will cause the RX reset sequence state machine to reset the SERDES and cause the LSM to unlock.

The two examples below illustrate the difference between a valid and an invalid even clock cycle boundary between COMMA occurrences (Note: C = comma, D = data).

Valid (even) comma boundary:

Word Clock Cycle	0	1	2	3	4	5	6	7	8	9	10	11
CHARACTER	C	D	C	D	C	D	D	D	D	D	C	D

Valid (odd) comma boundary:

Word Clock Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12
CHARACTER	C	D	C	D	C	D	D	D	D	C	D	C	D

In the invalid (odd) comma boundary case above, the comma occurrences on cycles 9 and 11 are invalid since they do not fall on an even boundary apart from the previous comma.

Alternatively, the LSM can be disabled, and the word aligner is controlled from the fabric word_align_en_ch[3:0]_c input pin.

Transmit Path

- Serializer
- 8b10b encoder

Receive Path

- Deserializer
- Word alignment to a user-defined word alignment character or characters from embedded GbE Link State Machine
- 8b10b decoding
- Clock Tolerance Compensation (optional)

Gigabit Ethernet and SGMII Modes

The Gigabit Ethernet mode of the LFE5UM/LFE5UM5G SERDES/PCS block supports full compatibility, from the Serial I/O to the GMII/SGMII interface of the IEEE 802.3-2002 1000 BASE-X Gigabit Ethernet standard.

Transmit Path

- Serializer
- 8b10b encoding

Receive Path

- Deserializer
- Word alignment based on IEEE 802.3-2002 1000 BASE-X defined alignment characters.
- 8b10b decoding
- The Gigabit Ethernet Link State Machine is compliant to Figure X (Synchronization State Machine, 1000BASE-X) of IEEE 802.3-2002 with one exception. Figure X requires that four consecutive good code groups are received in order for the LSM to transition from one SYNC_ACQUIRED_{N} (N=2,3,4) to SYNC_ACQUIRED_{N-1}. Instead, the actual LSM implementation requires five consecutive good code groups to make the transition.

- Gigabit Ethernet Carrier Detection: Section 36.2.5.1.4 of IEEE 802.3-2002 (1000BASE-X) defines the carrier_detect function. In Gigabit Ethernet mode, this feature is not included in the PCS and a carrier_detect signal is not provided to the FPGA fabric. • Clock Tolerance Compensation logic capable of accommodating clock domain differences.

Gigabit Ethernet (1000BASE-X) Idle Insert

This is required for Clock Compensation and Auto-negotiation. Auto-negotiation is done in FPGA logic. The Lattice Gigabit Ethernet PCS IP core provides the auto-negotiation discussed below.

Idle pattern insertion is required for clock compensation and auto-negotiation. Auto-negotiation is done in FPGA logic. This module automatically inserts /I2/ symbols into the receive data stream during auto-negotiation. While auto-negotiating, the link partner will continuously transmit /C1/ and /C2/ ordered sets. The clock-compensator will not delete these ordered sets as it is configured to only insert/delete /I2/ ordered sets. In order to prevent overruns and underruns in the clock-compensator, /I2/ ordered sets must be periodically inserted to provide insertion/deletion opportunities for the clock compensator.

While performing auto-negotiation, this module will insert a sequence of eight /I2/ ordered sets (2 bytes each) every 2048 clock cycles. As this module is after the 8b10b decoder, this operation will not introduce any running disparity errors. These /I2/ ordered sets will not be passed on to the FPGA receive interface as the GMII interface is driven to IDLE by the RX state machine during auto-negotiation. Once auto-negotiation is complete, /I2/ insertion is disabled to prevent any corruption of the received data.

Note that this state machine is active only during auto-negotiation. The auto-negotiation state machine and the GbE receive state machines are implemented in the soft logic. This state machine depends on the signal xmit_ch[1:0] from the auto-negotiation state machine. This signal is provided on the TX data bus. Though this signal is relatively static (especially after auto-negotiation) it is included in the TX data bus.

Table 11. GbE IDLE State Machine Control and Status Signals

Module Signal	Direction	Description
xmit_ch[1:0]	In	From FPGA logic Auto-Negotiation State Machine

Gigabit Ethernet Idle Insert and correct_disp_ch[1:0] Signals

The correct_disp_ch[1:0] signal is used on the transmit side of the PCS to ensure that an interpacket gap begins in the negative disparity state. Note that at the end of an Ethernet frame, the current disparity state of the transmitter can be either positive or negative, depending on the size and data c.ontent of the Ethernet frame.

However, from the FPGA soft-logic side of the PCS, the current disparity state of the PCS transmitter is unknown. This is where the correct_disp_ch[1:0] signal comes into play. If the correct_disp_ch[1:0] signal is asserted for one clock cycle upon entering an interpacket gap, it will force the PCS transmitter to insert an IDLE1 ordered-set into the transmit data stream if the current disparity is positive. However, if the current disparity is negative, then no change is made to the transmit data stream.

From the FPGA soft-logic side of the PCS, the interpacket gap is typically characterized by the continuous transmission of the IDLE2 ordered set which is as follows: tx_k_ch=1, txdata= 0xBC tx_k_ch=0, txdata=0x50.

Note that in the PCS channel, IDLE2s mean that current disparity is to be preserved. IDLE1s mean that the current disparity state should be flipped. Therefore, it is possible to ensure that the interpacket gap begins in a negative disparity state. If the disparity state before the interpacket gap is negative, then a continuous stream of IDLE2s are transmitted during the interpacket gap. If the disparity state before the interpacket gap is positive, then a single IDLE1 is transmitted followed by a continuous stream of IDLE2s.

In the FPGA soft-logic side of the PCS, the interpacket gap is always driven with IDLE2s into the PCS. The correct_disp_ch[3:0] signal is asserted for one clock cycle, k_cntrl=0, data=0x50 when the interpacket gap first begins. If necessary, the PCS will convert this IDLE2 into an IDLE1. For the remainder of the interpacket gap, IDLE2s should be driven into the PCS and the correct_disparity_chx signal should remain deasserted.

For example, if a continuous stream of 512 bytes of Ethernet frames and 512 bytes of /I/ are sent, it can be observed that:

- During the first interpacket gap, all negative disparity /I2/s are seen (K28.5(-) D16.2(+))
- During the next interpacket gap, the period begins with positive disparity /I1/ (K28.5 (+), D5.6 (+/- are the same)), then all remaining ordered sets are negative disparity /I2/s
- During the next interpacket gap, all negative disparity /I2/s are seen
- During the next interpacket gap, the period begins with positive disparity /I1/ (K28.5 (+), D5.6 (+/- are the same)), then all remaining ordered sets are negative disparity /I2/s

A number of programmable options are supported within the encoder module. These are:

- Ability to force negative or positive disparity on a per-word basis
- Ability to input data directly from the FIFO Bridge - external multiplexer
- Ability to replaced code words dependant upon running disparity (100BASE-X and FC)
- Software register controlled bypass mode

XAUI Mode

With the Lattice XAUI IP Core, the XAUI mode of the SERDES/PCS block supports full compatibility from Serial I/O to the XGMII interface of the IEEE 802.3-2002 XAUI standard. XAUI Mode supports 10 Gigabit Ethernet.

Transmit Path

- Serializer
- Transmit State Machine which performs translation of XGMII idles to proper ||A||, ||K||, ||R|| characters according to the IEEE 802.3ae-2002 specification
- 8b10b Encoding

Receive Path

- Deserializer
- Word alignment based on IEEE 802.3-2002 defined alignment characters.
- 8b10b Decoding
- The XAUI Link State Machine is compliant to Figure 48-7 - PCS synchronization state diagram of IEEE 802.3ae-2002 with one exception. Figure 48-7 requires that four consecutive good code groups be received in order for the LSM to transition from one SYNC_ACQUIRED_{N} (N=2,3,4) to SYNC_ACQUIRED_{N-1}. Instead, the actual LSM implementation requires five consecutive good code groups to make the transition.
- Clock Tolerance Compensation logic in PCS is disabled in XAUI mode. MCA (Multi-Channel Alignment) and CTC are done in the XAUI IP core.
- x4 multi-channel alignment should be done in FPGA core logic.

PCI Express Revision 1.1 and 2.0

The PCI Express mode of the SERDES/PCS block supports x1, x2, and x4 PCI Express applications.

Transmit Path

- Serializer
- 8b10b Encoding
- Receiver Detection
- Electrical Idle

Receive Path

- Deserializer
- Word alignment based on the Sync Code
- 8b10b Decoding
- Link Synchronization State Machine functions incorporating operations defined in the PCS Synchronization State Machine (Figure 48-7) of the IEEE 802.3ae-2002 10GBASE-X Specification.
- x2 or x4 PCI Express operation with one PCS quad set to PCI Express mode.
- Clock Tolerance Compensation logic capable of accommodating clock domain differences.
- x2 or x4 multi-channel alignment should be done in FPGA core logic.

Table 12 describes the PCI Express mode specific ports.

Table 12. PCI Express Mode Specific Ports

Signal	Direction	Class	Description
pcie_done_ch[1:0]_s	Out	Channel	1 = Far-end receiver detection complete 0 = Far-end receiver detection incomplete
pcie_con_ch[1:0]_s	Out	Channel	Result of far-end Receiver detection 1 = Far-end receiver detected 0 = Far-end receiver not detected
pcie_det_en_ch[1:0]_c	In	Channel	FPGA logic (user logic) informs the SERDES block that it will request a PCI Express Receiver Detection operation 1 = Enable PCI Express Receiver Detect 0 = Normal Operation
pcie_ct_ch[1:0]_c	In	Channel	1 = Request transmitter to do far-end receiver detection 0 = Normal Operation
rxstatus[2:0]	Out	Channel	Per channel PCI Express receive status port. RxStatus is an encoded status of the receive data path. 2 bits wide if in 16-bit bus mode.

PCI Express Termination

At the electrical level, PCI Express utilizes two uni-directional low voltage differential signaling pairs at 2.5 Gbps (for ECP5 and ECP5-5G devices), or 5 Gbps (for ECP5-5G devices only) for each lane. Transmit and receive are separate differential pairs, for a total of four data wires per lane. An input receiver with programmable equalization and output transmitters with programmable de-emphasis permits optimization of the link. The PCI Express specification requires that the differential line must be common mode terminated at the receiving end. Each link requires a termination resistor at the far (receiver) end. The nominal resistor values used are 100 Ohms. This is accomplished by using the embedded termination features of the CML inputs as shown in Figure 15. The specification requires AC coupling capacitors (CTX) on the transmit side of the link. This eliminates potential common-mode bias mismatches between transmit and receive devices. The capacitors must be added external to the Lattice CML outputs.

PCI Express L2 State

For the PCI Express L2 state, the rx_pwrup_c signal should not be de-asserted to power-down the rx channel. The rx_pcs_rst_c signal should be used to hold the channel in reset to save power.

Figure 15. PCI Express Interface Diagram

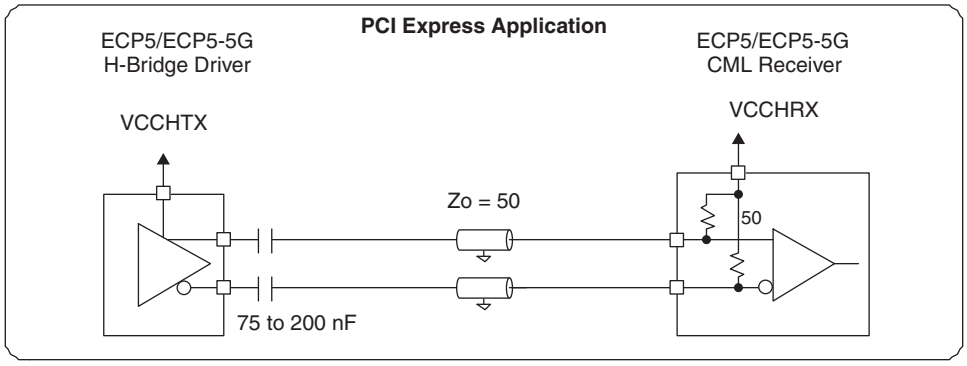


Table 13. Differential PCI Express Specifications

Symbol	Parameter	Min.	Nom.	Max.	Units	Comments	Location
ZTX-DIFF-DC	DC Differential TX Impedance	80	100	120	Ohm	TX DC Differential mode low impedance. ZTX-DIFF-DC is the small signal resistance of the transmitter measured at a DC operating point that is equivalent to that established by connecting a 100 Ohm resistor from D+ and D- while the TX is driving a static logic one or logic zero.	Internal
ZRX-DIFF-DC	DC Differential Input Impedance	80	100	120	Ohm	RX DC Differential mode impedance during all LTSSM states. When transmitting from a Fundamental Reset to Detect (the initial state of the LTSSM), there is a 5ms transition time before receiver termination values must be met on all un-configured lanes of a port	Internal
CTX	AC Coupling Capacitor	75	-	200	nF	All transmitters shall be AC coupled. The AC coupling is required either within the media or within transmitting component itself	External

PCI Express Electrical Idle Transmission

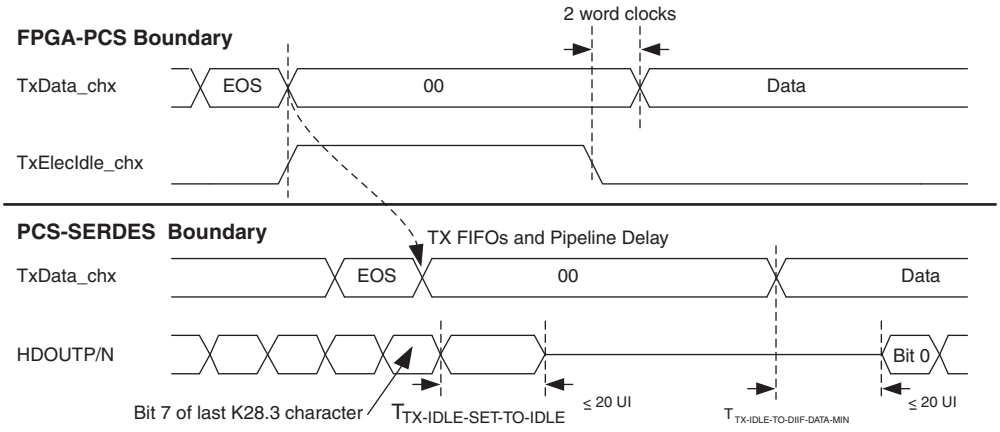
Electrical Idle is a steady state condition where the transmitter P and N voltages are held constant at the same value (i.e., the Electrical Idle Differential Peak Output Voltage, VTX-IDLE-DIFFp, is between 0 and 20mV). Electrical Idle is primarily used in power saving and inactive states.

Per the PCI Express base specification, before a transmitter enters Electrical Idle, it must always send the Electrical Idle Ordered Set (EIOS), a K28.5 (COM) followed by three K28.5 (IDL). After sending the last symbol of the Electrical Idle Ordered Set, the transmitter must be in a valid Electrical Idle state as specified by TTX-IDLE-SET-TO-IDLE is less than 20UI.

To achieve this, the Electrical Idle Enable (tx_idle_chx_c) from the FPGA core logic to the PCS is sent in along with each transmit data. This signal is pipelined similarly all the way to the PCS-SERDES boundary. For all valid data this signal is LOW. To initiate Electrical Idle, the FPGA logic pulls this signal HIGH on the clock after it transmits the last K28.5 (IDL) symbol. As the signal is pipelined to the PCS-SERDES boundary, the relationship between the transmit data and this signal is exactly the same as on the FPGA-PCS boundary.

14UI after the rising edge of the Electrical Idle enable signal at the PCS-SERDES boundary the last bit (bit7) of the last K28.3 (IDL) symbol is transmitted. 16UI (<20UI) later the transmit differential buffer achieves Electrical Idle state.

Figure 16. Transmit Electrical Idle



As long as the FPGA core logic deems that the transmitter needs to stay in Electrical Idle state it needs to clock in data (preferably all zeros) along with the Electrical Idle Enable (tx_idle_chx_c) signal active (HIGH). The transmitter is required to stay in the Electrical Idle state for a minimum of 50UI (20ns) (TTX-IDLE-MIN).

PCI Express Electrical Idle Detection

Each channel in the quad has a loss-of-signal detector. The Electrical Idle is detected once two out of the three K28.3 (IDL) symbols in the Electrical Idle Ordered Set (EOS) have been received. After the Electrical Idle Ordered Set is received, the receiver should wait for a minimum of 50ns (TTX-IDLE-MIN) before enabling its Electrical Idle Exit detector.

These signals (one per channel, four per quad) should be routed through the PCS, and should be made available to the FPGA core. The required state machine(s) to support electrical idle can then be constructed in the FPGA core.

PCI Express Receiver Detection

Figure 17 shows a Receiver Detection sequence. A Receiver Detection test can be performed on each channel of a quad independently. Before starting a Receiver Detection test, the transmitter must be put into electrical idle by setting the tx_idle_ch#_c input high. The Receiver Detection test can begin 120 ns after tx_elec_idle is set high by driving the appropriate pci_det_en_ch#_c high. This puts the corresponding SERDES Transmit buffer into receiver detect mode by setting the driver termination to high impedance and pulling both differential outputs to VCCOB through the high impedance driver termination.

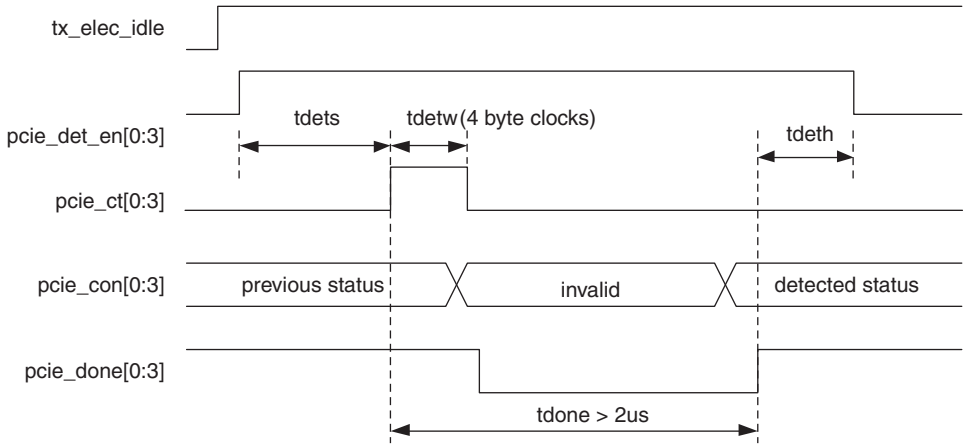
Setting the SERDES Transmit buffer into receiver detect state takes up to 120 ns. After 120 ns, the receiver detect test can be initiated by driving the channel's pcie_ct_ch#_c input high for four byte (word) clock cycles. The corresponding channel's pcie_done_ch#_s is then cleared asynchronously. After enough time for the receiver detect test to finish has elapsed (determined by the time constant on the transmit side), the pcie_done_ch#_s receiver detect status port will go high and the Receiver Detect status can be monitored at the pcie_con_ch#_s port. If at that time

the pcie_con_ch#_s port is high, then a receiver has been detected on that channel. If, however, the pcie_con_ch#_s port is low, then no receiver has been detected for that channel. Once the Receiver Detect test is complete, tx_idle_ch#_c can be deasserted.

Receiver detection proceeds as follows:

1. The user drives pcie_det_en high, putting the corresponding TX driver into receiver detect mode. This sets the driver termination to high-impedance (5K ohm) and pulls both outputs of the differential driver toward common mode through the high-impedance driver termination. The TX driver takes some time to enter this state so the pcie_det_en must be driven high for at least 120ns before pcie_ ct is asserted.
2. The user drives pcie_ct high for four byte clocks.
3. SERDES drives the corresponding pcie_done low.
4. SERDES drives the internal signal (corresponding to pcie_ct) for as long as it is required (based on the time constant) to detect the receiver.
5. SERDES drives the corresponding pcie_con connection status.
6. SERDES drives the corresponding pcie_done high
7. The user can use this asserted state of pcie_done to sample the pcie_con status to determine if the receiver detection was successful.

Figure 17. PCI Express Mode Receiver Detection Sequence



PCI Express Power Down Mode

rx_serdes_rst_ch[1:0] reset signal should be used instead of rx_pwrup_ch[1:0] signal. This allows the RX termination to remain enabled at 50 Ohms so that the far-end transmitter can detect that a receiver is connected.

PCI Express Beacon Support

This section highlights how the LFE5UM/LFE5UM5G PCS can support Beacon detection and transmission. The PCI Express requirements for Beacon detection are presented with the PCS support for Beacon transmission and detection.

Beacon Detection Requirements

- Beacon is required for exit from L2 (P2) state.
- Beacon is a DC-balanced signal of periodic arbitrary data, which is required to contain some pulse widths >= 2ns (500 MHz) and < 16us (30 KHz).
- Maximum time between pulses should be < 16 us.
- DC balance must be restored within < 32 us.

- For pulse widths > 500 ns, the output beacon voltage level must be 6 db down from VTX-DIFFp-p (800 mV to 1200 mV).
- For pulse widths < 500 ns, the output beacon voltage level must be ≤ VTX-DIFFp-p and >= 3.5 db down from VTX-DIFFp-p.

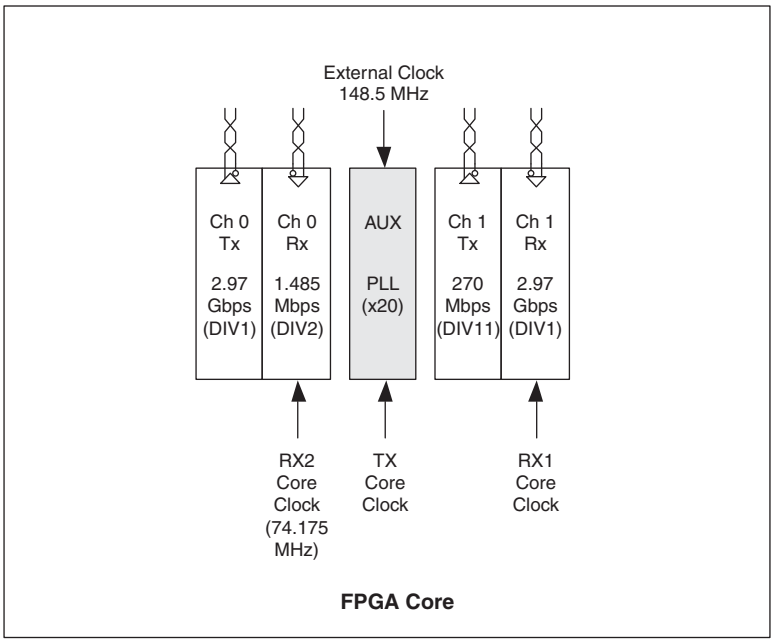
PCS Beacon Detection Support

- The signal loss threshold detection circuit senses if the specified voltage level exists at the receiver buffer.
- This is indicated by the rlos_lo_ch[1:0] signal.
- This setting can be used both for PCI Express Electrical Idle Detection and PCI Express beacon detection (when in power state P2).
- The remote transmitting device can have a beacon output voltage of 6 db down from VTX-DIFFpp (i.e., 201 mV). If this signal can be detected, then the beacon is detected.

PCS Beacon Transmission Support

Sending the K28.5 character (IDLE) (5 ones followed by 5 zeroes) provides a periodic pulse with of 2 ns occurring every 2 ns (1.0 UI = 400 ps, multiplied by 5 = 2 ns). This meets the lower requirement. The output beacon voltage level can then be VTX-DIFFp-p. This is a valid beacon transmission.

Figure 18. Example: 3G/HD/SD RX/TX At Different Rate



To support the major application requirements, a selectable DIV per RX and TX is supported. In LFE5UM/LFE5UM5G, DIV11 has been added. One potential multi-rate configuration is to provide a 148.5MHz REFCLK from the primary pins to the TX PLL. The TX PLL would be in the x20 mode. The resulting output clock would be 2.97 GHz. Then, by using the DIV2 for 1.485Gbps and DIV11 for 270Mbps, a very quick switchover can be achieved without having to re-train and lock the PLL.

Serial Digital Video and Out-Of-Band Low Speed SERDES Operation

The LFE5UM/LFE5UM5G SERDES/PCS supports any data rates that are slower than what the SERDES TX PLL and RX CDR natively support (<250Mbps: Out-Of-Band signal, OOB), by bypassing the receiver CDR and associated SERDES/PCS logic (e.g., 100Mbps Fast Ethernet, SD-SDI at 143Mbps or 177Mbps). Though these out-of-band paths primarily use low data rates, higher rates can be used for other functional reasons. See the Multi-Rate SMPTE Support section of this document for more information.

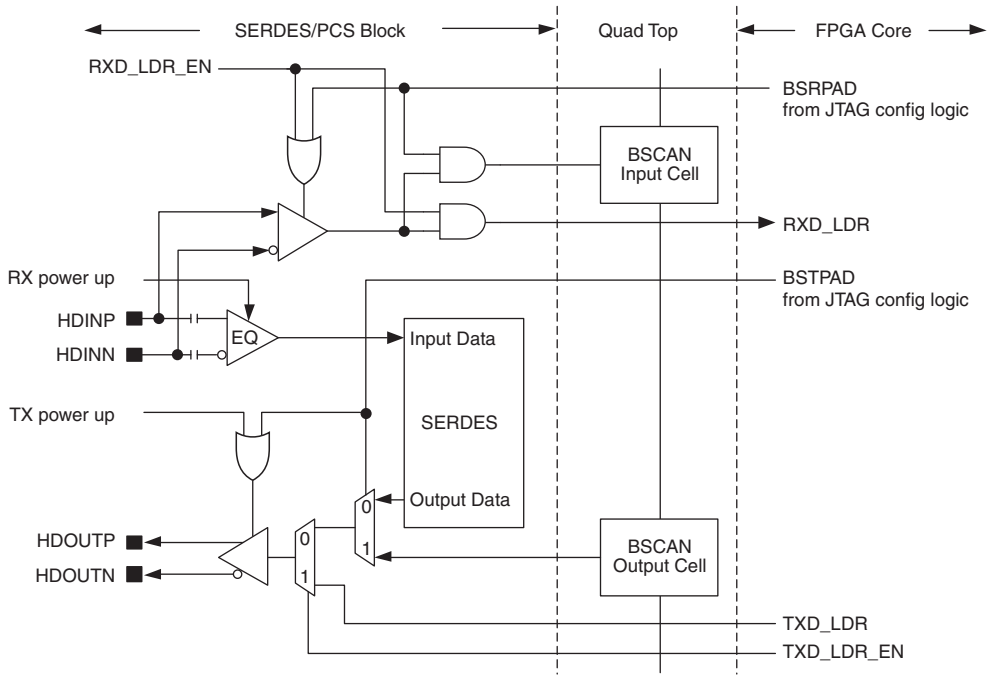
In addition, for SD-SDI, these rates sometimes must co-exist on the same differential RX pair with HD-SDI rates (i.e., SD-SDI rates may be active and then the data rate may switch over to HD-SDI rates). Since there is no way to predict which of these two rates will be in effect, it is possible to send the input data stream to two SERDES in parallel, a high-speed SERDES (already in the quad) and a lower-speed SERDES (implemented outside the quad). One possible implementation is shown in Figure 19.

There is an input per channel RXD_LDR, low data rate single-ended input from the RX buffer to the FPGA core. In the core a low-speed Clock Data Recovery (CDR) block or a Data Recovery Unit (DRU) can be built using soft logic. A channel register bit, RXD_LDR_EN, can enable this data path. If enabled by another register bit, a signal from the FPGA can also enable this in LFE5UM/LFE5UM5G.

In the transmit direction, it is also possible to use a serializer built in soft logic in the FPGA core and use the TXD_LDR pin to send data into the SERDES. It will be muxed in at a point just before the de-emphasis logic near where the regular high-speed SERDES path is muxed with the boundary scan path. This is shown conceptually in Figure 19. The low data rate path can be selected by setting a channel register bit, TX_LDR_EN.

Alternatively, on the output side, the high-speed SERDES is used to transmit either high-speed data, or lower speed data using decimation (the SERDES continues to run at high-speed, but the output data can only change every nth clock where n is the decimation factor).

Figure 19. Conceptual Low Date Rate Path for RX and TX; Example for Out-of-Band Application



Common Public Radio Interface (CPRI)

The goal of CPRI is to allow base station manufacturers and component vendors to share a common protocol and more easily adapt platforms from one user to another.

Transmit Path

- Serializer
- Transmit State Machine is set to Gigabit Ethernet Mode
- 8b10b Encoding

Receive Path

- Deserializer
- Word alignment based on IEEE 802.3-2002 1000 BASE-X defined alignment characters
- 8b10b Decoding

CPRI does not specify mechanical or electrical interface requirements. In terms of scope, CPRI has a much narrower focus than OBSAI. CPRI looks solely at the link between the RRH and the baseband module(s). In CPRI nomenclature, those modules are known as Radio Equipment (RE) and Radio Equipment Control (REC), respectively. In other words, CPRI is specifying the same interface as the OBSAI RP3 specification. CPRI primarily covers the physical and data link layer of the interface. It also specifies how to transfer the user plane data, control and management (C&M) plane data and the synchronization plane data.

CPRI has had better “traction” for two reasons - the muscle of the companies backing it and the focus on just one interface link (between the RF modules and the Baseband modules) and even at that focusing primarily on the physical and data link layers.

CPRI allows four line bit rate options; 614.4 Mbps, 1.2288 Gbps, 2.4576 Gbps, and 3.072 Gbps, and 4.9152 Gbps (ECP5-5G only); at least one of these rates needs to be supported. The higher line rate is always compared to the one that is immediately lower.

CPRI does not have a mandatory physical layer protocol, but the protocol used must meet the BER requirement of 1 x 10-12. It also specifies the clock stability and the phase noise requirements.

CPRI also recommends two electrical variants: high voltage (HV) and low voltage (LV). HV is guided by 1000Base-CX specifications in IEEE 802.3-2002 clause 39 with 100-ohm impedance. LV is guided by XAUI. LV is recommended for all rates and will be the focus for this device.

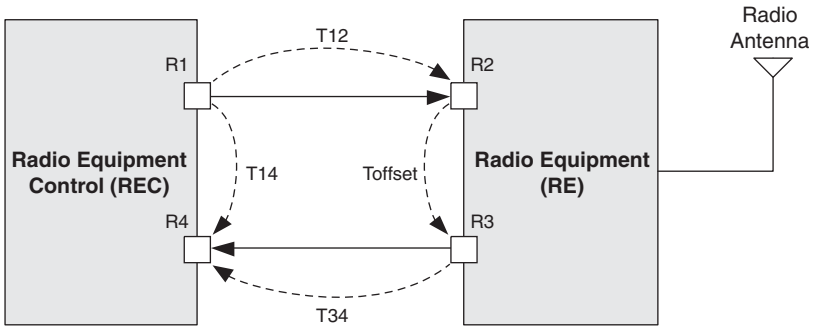
It is important to understand two link layer requirements when dealing with the CPRI specifications:

- Link delay accuracy and cable delay calibration
- Startup synchronization

Link Delay Accuracy and Cable Delay Calibration

The REs or RRHs are frequency locked to the REC or BTS. Thus, in this synchronous system it is necessary to calibrate all delays between RRHs and the BTS to meet air-interface timing requirements. The interface requires the support of the basic mechanisms to enable calibrating the cable delay on links and the round trip delay on single- and multi-hop connections. Specifically, the reference points for delay calibration and the timing relationship between input and output signals at RE (Radio Equipment) are defined. All definitions and requirements are described for a link between REC Master Port and RE Slave Port in the single-hop scenario as shown in Figure 20.

Figure 20. Link Between REC Mater Port and RE Slave Port (Single Hop Scenario)



Reference points R1-4 correspond to the output point (R1) and the input point (R4) of REC, and the input point (R2) and the output point (R3) of an RE terminating a particular logical connection. The antenna is shown as Ra for reference.

- T12 is the delay of the downlink signal from the output point of REC (R1) to the input point of RE (R2), essentially the downlink cable delay.
- T34 is the delay of the uplink signal from the output point of RE (R3) to the input point of the REC (R4), essentially the uplink cable delay.
- Toffset is the frame offset between the input signal at R2 and the output signal at R3.
- T14 is the frame timing difference between the output signal at R1 and the input signal at R4 (i.e., the round trip delay - RTT).

Delay measurement is accomplished using frame timing. CPRI has a 10ms frame based on the UMTS radio frame number or Node B Frame Number, also known as BFN. Each UMTS Radio Frame has 150 hyperframes (i.e., each HyperFrame is 66.67us) with the corresponding hyperframe number (HFN = 0<=Z<=149). Each hyperframe has 256 (0<=W<=255) basic frames (i.e. each basic frame is 260.42ns = Tchip or Tc).

An RE determines the frame timing of its output signal (uplink) to be the fixed offset (Toffset) relative to the frame timing of its input signal (downlink). Toffset is an arbitrary value, which is greater than or equal to 0 and less than 256 Tc (it cannot slip beyond a hyperframe). Different REs may use different values for Toffset. REC knows the value of Toffset of each RE in advance (pre-defined value or RE informs REC by higher layer message).

To determine T14, the downlink BFN and HFN from REC to RE is given back in uplink from the RE to the REC. In the case of an uplink-signaled error condition, the REC treats the the uplinks BFN and HFN as invalid. So, T14 = T12 + Toffset + T34.

As stated earlier the system is synchronous. Further, assuming that hyperframes are of fixed length and the RRH-BTS interconnect (cable length) is equal in both directions (i.e.,T12 = T34, both optical fibers are in one bundle), the interconnect delay devolves down to (T14 - Toffset)/2. The method for determining T14 has been discussed earlier. So the major component that affects delay calibration is Toffset. Thus, the interconnect delay is the difference in hyperframe arrival and departure times measured at each side of the link.

Delay calibration requirements are driven by 3GPP and UTRAN requirements specifically requirements R-21 in the CPRI specification (CPRI v3.0 page 20), which states that the accuracy of the round trip delay measurement of the cable delay of one link is +/- Tc/16. Additionally, requirement R-20 states that the round trip time absolute accuracy of the interface, excluding the round trip group delay on the transmission medium (excluding the cable length), shall meet a similar requirement (+/- Tc/16 for T14). Taking into account the previous discussion, the absolute link delay accuracy in the downlink between REC master port and RE slave port excluding the cable length is half of the above requirement (+/- Tc/32 or approximately 8ns (8.138ns)). Thus, both T14 and Toffset need absolute accuracy less than +/- 8ns.

Next it is important to determine how many bits of uncertainty can be acceptable for the different rates. Essentially, the various CPRI and OBSAI bit rates can be multiplied by 8.138ns to determine the number of bits worth of indeterminism/variance is acceptable. The impact of this will become clear subsequently when the SERDES serial/parallel data path is discussed.

Most SERDES have a certain level of uncertainty that is introduced in the serializing and de-serializing process. Thus, a SERDES with 16-bit bus architecture may have twice the delay uncertainty as a SERDES with a 8-bit architecture because the number of bits per word is doubled.

TX and RX latency respectively in Table 17 is listed. The table also lists the variability between the latency. This variability directly contributes to the absolute delay accuracy required from earlier discussion. The variability comes from three sources: TX FPGA Bridge FIFO, RX FPGA Bridge FIFO and RX Clock Tolerance Compensation FIFO. Since the CPRI system is a synchronous system, the RX CTC FIFO is bypassed and the RX recovered clock is used.

The remaining contributors to the latency variability are the FPGA Bridge FIFO. This FIFO can be bypassed if the interface to the FPGA is 8-bit bus mode. In 16-bit interface mode, the FPGA Bridge FIFO cannot be bypassed because the 2:1 gearing is done via the FIFO.

SDI (SMPTE) Mode

The SDI mode of the LFE5UM/LFE5UM5G SERDES/PCS block supports all three SDI modes, SD-SDI, HD-SDI and 3G-SDI.

Transmit Path

- Serializer

Receive Path

- Deserializer
- Optional word alignment to user-defined alignment pattern.

The following data rates are the most popular in the broadcast video industry.

- SD-SDI (SMPTE259M): 270 Mbps
- HD-SDI (SMPTE292M): .... 1.485 Gbps; Fractional Rate: 1.485 Gbps / 1.001 = 1.4835 Gbps
- 3G-SDI (SMPTE424M): .... 2.97 Gbps; Fractional Rate: 2.97 Gbps / 1.001 = 2.967 Gbps

SDI connections do not require to be bi-directional, that means the Rx and Tx channels can be running at different data rates in SD / HD / 3G. The LFE5UM/LFE5UM5G product family, mixing of these rates in one Rx / Tx channel is possible because each of the Rx and Tx channel has its own rate divider. The maximum rate generated by the TxPLL or CDR clock recovery circuit at 2.97Gbps can support 3G-SDI in the Rx / Tx channels independently with its own /1 rate divider; supports HD-SDI with independent /2 rate divider; and supports SD-SDI with independent /11 rate divider.

Most designers have indicated that they would like to see support on dynamically switching between these 3 rates (SD, HD, and 3G). The reason is that in a broad-cast studio, or a satellite head-end or cable head-end, they do not necessarily have prior knowledge of what the RX data rate will be.

The time to switch between different rates should be kept as minimal as possible. On the Tx side, the switching can be very quick, by simply changing the selection of the /1, /2, or /11 rate divider. The TxPLL stays locked to 2.97 MHz, and is not affected.

On the Rx side, each channel has its own CDR, Clock Data Recovery, circuit. The switching between different data rates by changing the rate divider causes a little more time for the CDR to be re-locked.

Depending on the geography where the equipment is deployed, either the full HD/3G-SDI rates (Europe/Asia) are used while transmitting video or the fractional rates (North America - NTSC). This allows us to develop two potential solution example cases for multi-rate SMPTE support with high quad utilization. Please note that simultaneous support of 3G/HD Full TX Rate(s) and Fractional TX Rate(s) is not possible in the same SERDES quad. In general, based on the above, geographically partitioned usage is an acceptable limitation.

Also note the following reference clock requirements:

- Tri-rate SDI (SD / HD / 3G): reference clock frequency should be set to 148.5 MHz, with 20X setting to generate full data rate at 2.97 Gbps
- Dual-rate (HD / 3G): reference clock frequency should be 148.5 MHz for full SDI rate, and 148.35 MHz for fractional SDI rate
- SD-SDI Only: reference clock frequency should be 54 MHz. This is because the Fmin for reference clock is 50 MHz. SD-SDI should be using 10X setting to generate full data rate at 540 Mbps, and use /2 rate-divider for 270 Mbps SD-SDI

The LFE5UM/LFE5UM5G SDI Serdes supports all SMPTE compliance tests, except 3G-SDI Level-A pathological compliance test pattern.

Embedded Display Port (eDP)

The eDP mode of the LFE5UM/LFE5UM5G SERDES/PCS block supports two different eDP modes, Reduced Bit Rate (RBR) and High Bit Rate (HBR).

Transmit Path

- Serializer

Receive Path

- Deserializer
- Optional word alignment to user-defined alignment pattern.

The following data rates are used in the eDP interface.

- RBR: 1.62 Gbps
- HBR: 2.70 Gbps

eDP is an emerging standard that is customized from the Display Port standard. The LFE5UM/LFE5UM5G families provide the transmit and receive buffers for the eDP interface. The FPGA logic allows the user to fully customize his requirements, including the Auxiliary channel.

The following reference clock is required for these eDP modes:

- RBR: 162 MHz
- HBR: 135 MHz

Example of Dual-based Channel Arrangements

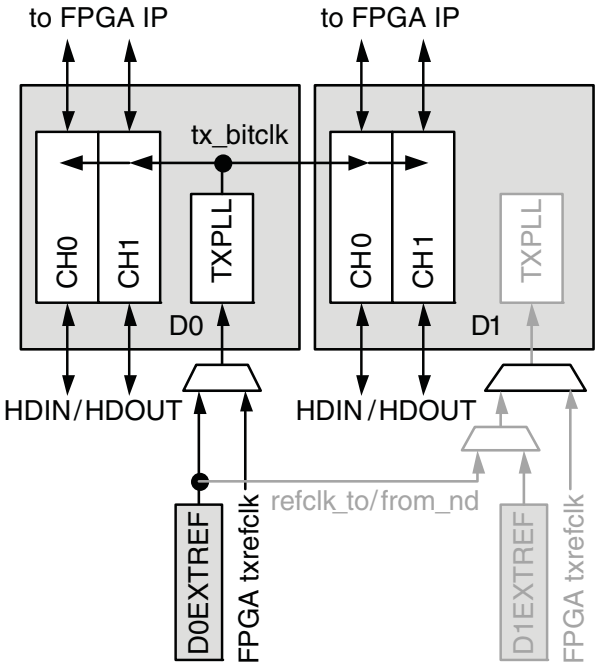
This section shows 5 examples of how instances created in users' view, and how they are placed and connected physically. The user will see how the improved granularity works in dual base architecture.

Tx Case 1:

All channels in Dual0 and Dual1 use D0 REFCLK(D0EXTREFB or FPGA txrefclk) and D0TXPLL

In this example, all 4channels use same tx_bitclk from D0TXPLL.

Figure 21. Tx Case 1

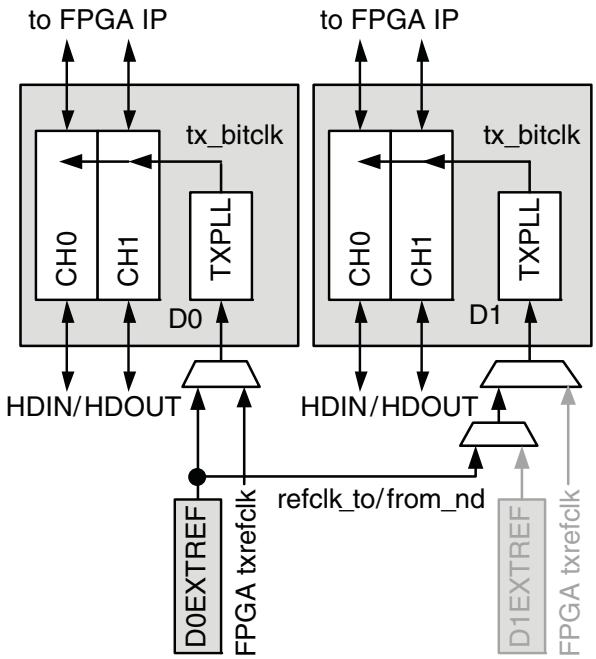


Tx Case 2:

Dual0 channels use D0 Refclk(D0EXTREFB or FPGA txrefclk) and D0TXPLL.

Dual1 channels use D0 Refclk(D0EXTREFB or FPGA txrefclk) and D1TXPLL.

Figure 22. Tx Case 2



In this example, both D0TXPLL and D1TXPLL use D0 Refclk. The tx_bitclks may have different frequency depending on the txpll multiplier. txpll multiplier is not a user attribute but is automatically calculated with data rate and refclk frequency.

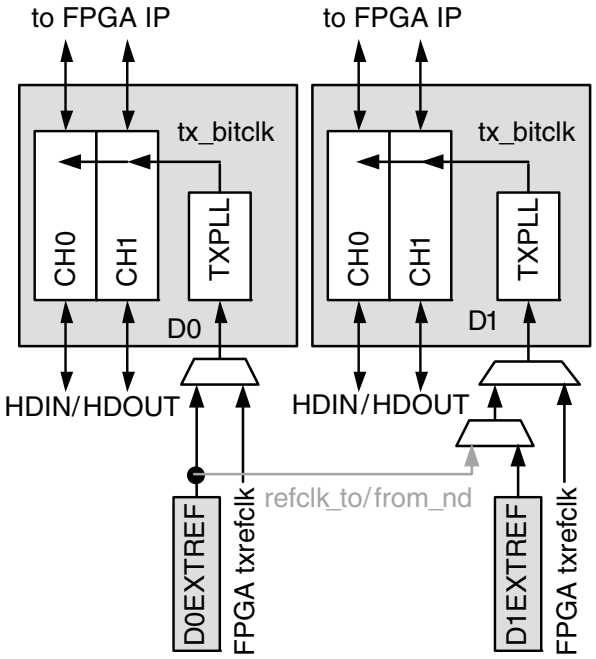
Tx Case 3:

Dual0 channels use D0 Refclk(D0EXTREFB or FPGA txrefclk) and D0TXPLL

Dual1 channels use D1 Refclk(D1EXTREFB or FPGA txrefclk) and D1TXPLL

In this example, no clocks are crossing Dual boundary.

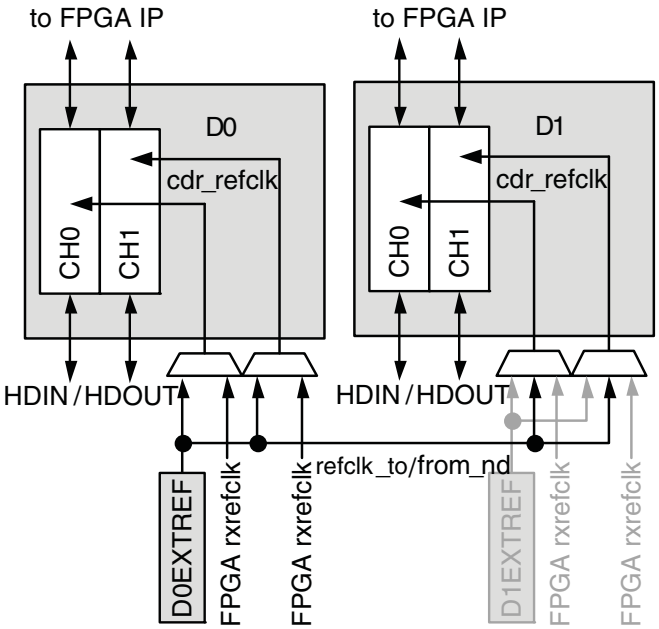
Figure 23. Tx Case 3



Rx Case1:

All channels in Dual0 and Dual1 use D0 REFCLK(D0EXTREFB or FPGA rxrefclk[cdr_refclk])

Figure 24. Rx Case 1

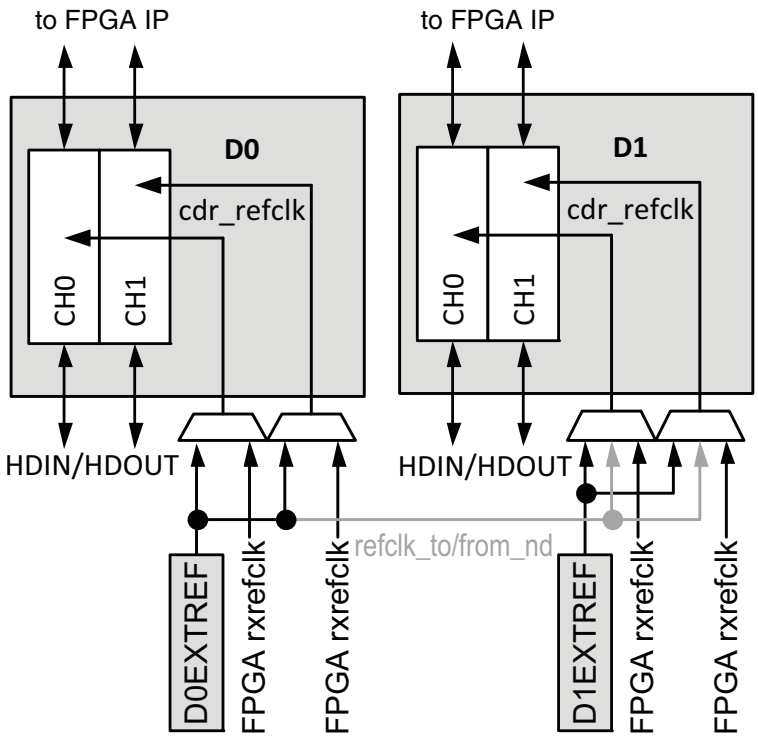


Rx Case 2:

Dual0 channels use Dual0 Refclk(D0EXTREFB or FPGA rxrefclk).

Dual1 channels use Dual1 Refclk(D1EXTREFB or FPGA rxrefclk).

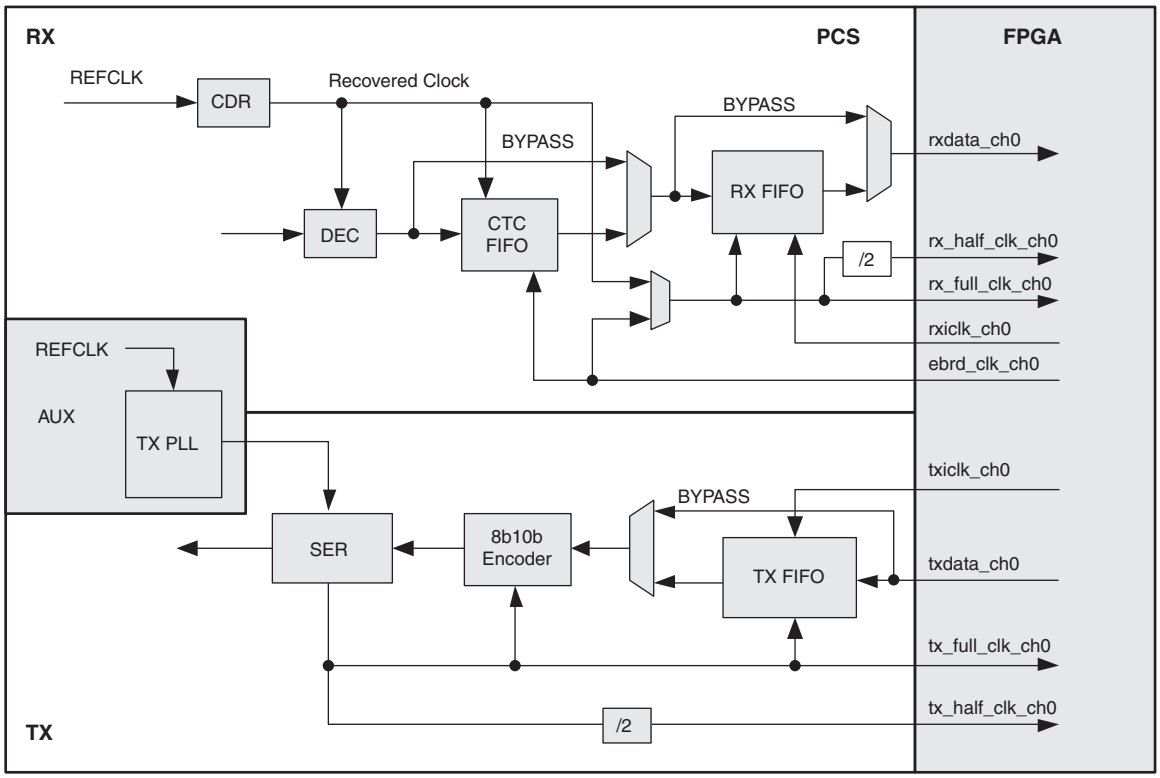
Figure 25. Rx Case 2



FPGA Interface Clocks

Figure 26 shows a conceptual diagram of the later stage of the PCS core and the FPGA Bridge and the major clocks that cross the boundary between PCS and the FPGA.

Figure 26. Conceptual PCS/FPGA Clock Interface Diagram



In the above diagram and in the subsequent clock diagrams in this section, please note that suffix “i” indicates the index [1:0] i.e., one for each channel. It is a requirement that if any of the selectors to the clock muxes are changed by writes to register bits, the software agent will reset the logic clocked by that muxed clock.

The PCS outputs 8 clocks. There are two transmit clocks (per channel) and two receive clocks (per channel). The two transmit clocks provide full rate and half rate clocks and are all derived from the TX PLL. There are also two clocks (full and half) per receive channel. These clocks are muxed into two clocks output per channel (rx_pclk, tx_pclk). These clocks are routed to PCLK to minimize skew and jitter.

Illustration on how the clocks are used can be seen in the Interface cases shown in Table 13.

Table 14. Seven User Interface Cases between SerDes/PCS Dual and FPGA Core

Interface	Datapath Width	RX CTC FIFO	RX Down Sample FIFO	TX Up Sample FIFO	Notes
Case I_a	8/10 bit	Yes	Yes	Yes	(2)
Case I_b	8/10 bit	Bypass	Yes	Yes	(2)
Case I_c	8/10 bit	Yes	Bypass	Bypass	(2)
Case I_d	8/10 bit	Bypass	Bypass	Bypass	(2)
Case II_a	16/20 bit	Yes	Yes	Yes	(1), (2)
Case II_b	16/20 bit	Bypass	Yes	Yes	(1), (2)

1. When using a 16/20-bit datapath width, the TX phase-shift (upsample) FIFO and the RX phase-shift FIFO (downsample) are always used. They cannot be bypassed. It is not required that both RX and TX have the same FPGA interface datapath width simultaneously. There is independent control available. For the sake of brevity, they have been represented together in the same use case.
2. The TX phase-shift (upsample) FIFO and the RX phase-shift FIFO (downsample) don't need to be bypassed together. They are independently controllable. Again, for the sake of brevity, they have been represented here in the same case.

2:1 Gearing

For guaranteed performance of the FPGA global clock tree, it is recommended to use a 16/20-bit wide interface for SERDES line rates greater than 2.5 Gbps. In this interface, the FPGA interface clocks are running at half the byte clock frequency.

Even though the 16/20-bit wide interface running at half the byte clock frequency can be used with all SERDES line rates, the 8/10-bit wide interface is preferred when the SERDES line rate is low enough to allow it (2.5 Gbps and below) because this results in the most efficient implementation of IP in the FPGA core.

The decision matrix for the six interface cases is explained in Table 15.

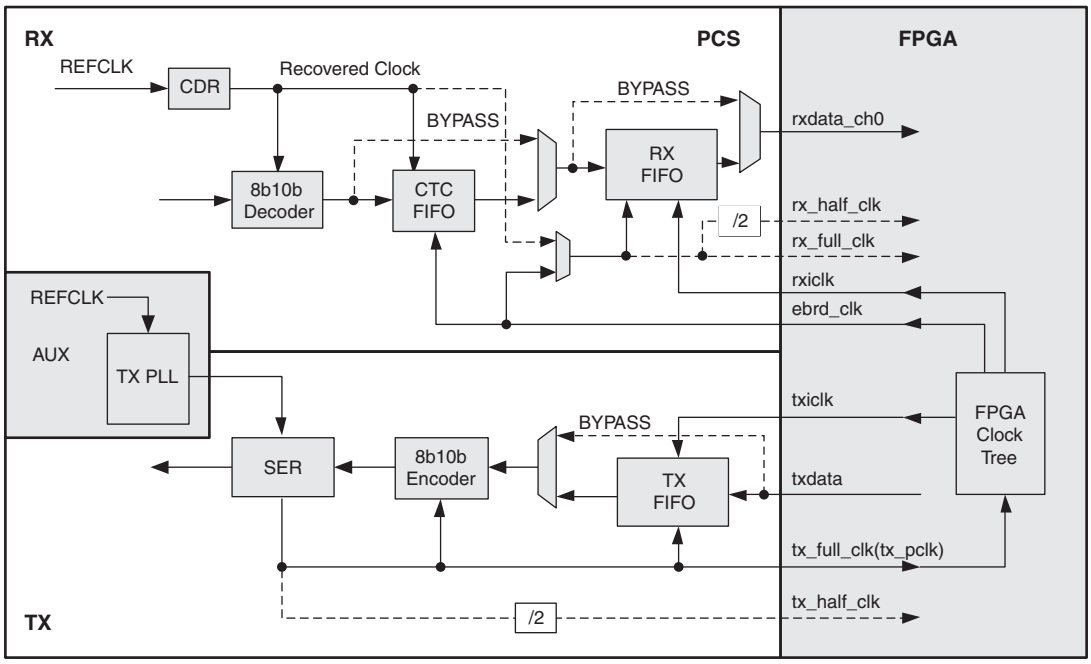
Table 15. Decision Matrix for Six Interface Cases

SERDES Line Rate	Dapath Width	Multi-Channel Alignment Required?	CTC Required?	RX FIFO Required?	Interface Case
2.5 Gbps and below	8/10 Bit (1:1 Gearing)	No, Single-Channel Link	Yes	Yes	Case I_a[1]
				No	Case I_c[1]
			No	Yes	Case I_b[2]
				No	Case I_d[2]
		Yes, Multi-Channel Link	Must Bypass, CTC Not Used	Yes	Case I_b[3]
				No	Case I_d[3]
Above 2.5 Gbps[7]	16/20 Bit (2:1 Gearing)	No, Single-Channel Link	Yes	Yes	Case II_a[4]
			No	Yes	Case II_b[5]
		Yes, Multi-Channel Link	Must Bypass, CTC Not Used	Yes	Case II_b[6]

1. This case is intended for single-channel links at line rates of 2.5 Gbps and lower (8/10-bit wide interface) that require clock tolerance compensation in the quad. CTC is required when both ends of the link have separate reference clock sources that are within +/- 300ppm of each other. Case I_a is used if the IP in the core requires the RX phase-shift FIFO. Case I_b is used if the IP does not require this FIFO.
2. This case is intended for single-channel links at line rates of 2.5 Gbps and lower (8/10-bit wide interface) that do NOT require clock tolerance compensation in the quad. CTC is not required when both ends of the link are connected to the same reference clock source. This is often the case for chip-to-chip links on the same circuit board. There is exactly 0ppm difference between the reference clocks and so CTC is not required and can be bypassed. CTC is also not required in the quad when this function is performed by the IP in the core.
3. This case is intended for multi-channel links at line rates of 2.5 Gbps and lower (8/10-bit wide interface). Multi-channel alignment MUST be done in the FPGA design. Since multi-channel alignment must be done prior to CTC, the CTC FIFO in the quad MUST be bypassed when multi-channel alignment is required and so both multi-channel alignment and CTC (if required) are done by the FPGA design.
4. This case is intended for single-channel links at line rates of 3.2 Gbps and lower that require a 2:1 gearbox between the quad and the FPGA core (16/20 bit wide interface). Clock tolerance compensation is included in the quad. CTC is required when both ends of the link have separate reference clock sources that are within +/- 300ppm of each other.
5. This case is intended for single-channel links at line rates of 3.2 Gbps and lower that require a 2:1 gearbox between the quad and the FPGA core (16/20-bit wide interface). Clock tolerance compensation is NOT included in the quad. CTC is not required when both ends of the link are connected to the same reference clock source. This is often the case for chip-to-chip links on the same circuit board. There is exactly 0ppm difference between the reference clocks and so CTC is not required and can be bypassed. CTC is also not required in the quad when this function is performed by the FPGA design.
6. This case is intended for multi-channel links at line rates of 3.2 Gbps and lower that require a 2:1 gearbox between the quad and the FPGA core (16/20-bit wide interface). Multi-channel alignment MUST be done by the FPGA design. Since multi-channel alignment must be done prior to CTC, the CTC FIFO in the quad MUST be bypassed when multi-channel alignment is required and so both multi-channel alignment and CTC (if required) are done by the FPGA design.
7. Higher than 3.2 Gbps, the 16/20 Bit needs to further gearing with additional 2:1 Gearing in the core, to increase the datapath width, and reduce the clock frequency in the core.

Case I_a: 8/10bit, CTC FIFO NOT Bypassed

Figure 27. 8/10-Bit, CTC FIFO and RX/TX FIFOs NOT Bypassed



1. The TX FIFO acts as a Phase Shift FIFO only in this case.
2. The RX FIFO acts as a Phase Shift FIFO only in this case.
3. The quad-level full rate clock from the TX PLL (tx_full_clk) has direct access to the FPGA center clock mux. This is a relatively higher performance path. A Global Clock Tree out of the Center Clock Mux is used to clock the user's interface logic in the FPGA. Some leaf nodes of the clock tree are connected to the FPGA Transmit Input clock (txiclk), the CTC FIFO Read Clock per Channel (ebrd_clk) through the FPGA Receive Input clock (rxiclk). This case is possibly the most common single-channel use case.

Example of Clock and Data Signals Interface in FPGA Logic (Case I_a)

Below is a portion of the SERDES/PCS module instantiation in the top module which describes how clock and data ports are mapped in Verilog.

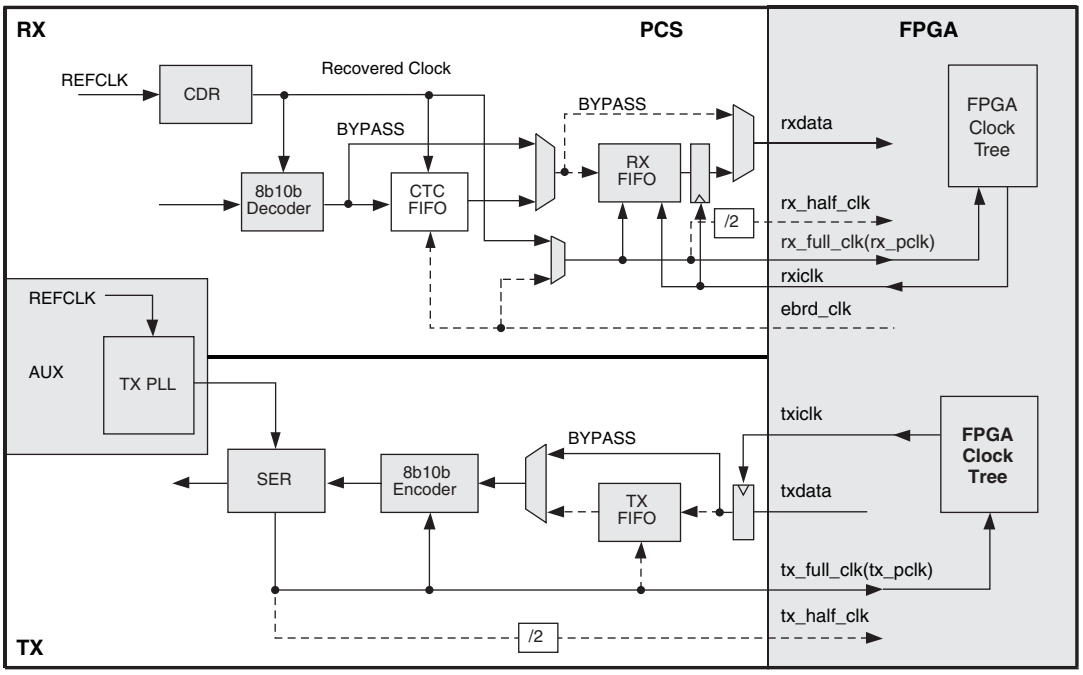
```
.txiclk_ch0(txclk),
.rxiclk_ch0(txclk),
.rx_full_clk_ch0(),
.tx_full_clk_ch0(txclk),
.tx_half_clk_ch0(),
.txdata_ch0(txdata_2_pcs),
.rxdata_ch0(rxdata_from_pcs),
.tx_k_ch0(txkcntl_2_pcs),
.rx_k_ch0(rxkcntl_from_pcs),
```

ebrd_clk_ch0 is routed automatically by the software, depending on the case.

Note that tx_full_clk_ch0 uses wire name 'txclk' and feeds both txi_clk_ch0 and rxi_clk_ch0, as shown in Figure 27.

Case I_b: 8 /10bit, CTC FIFO Bypassed

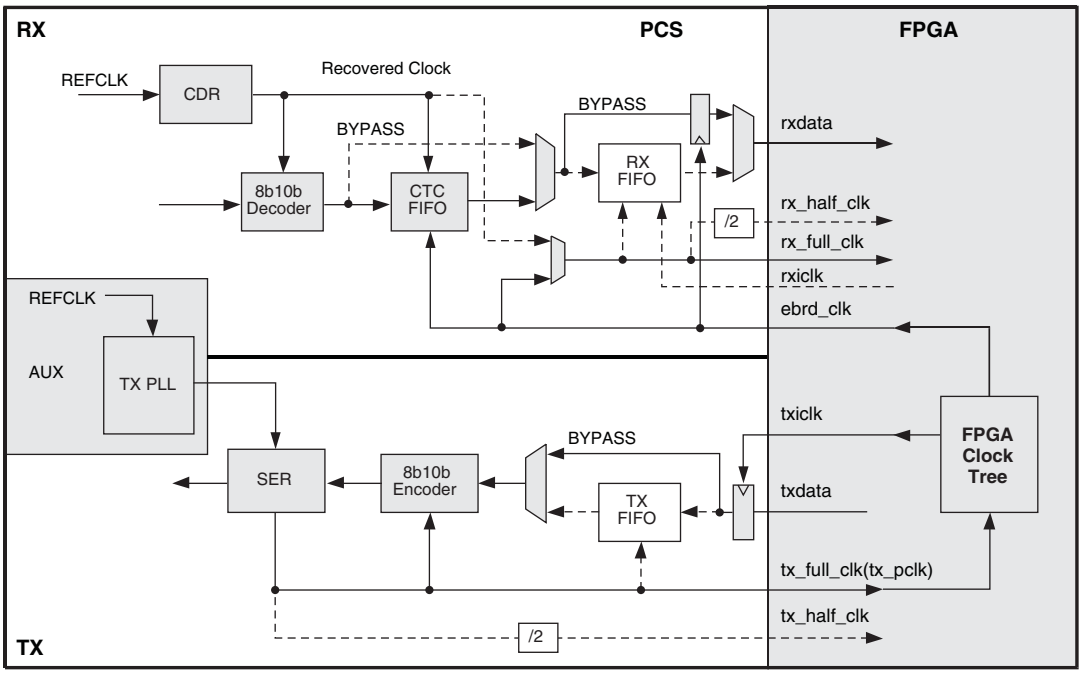
Figure 28. 8/10-Bit, CTC FIFO Bypassed



1. The TX FIFO acts as a Phase Shift FIFO only in this case.
2. The RX FIFO acts as a Phase Shift FIFO only in this case.
3. The TX FPGA Channel input clock is clocked similarly as in the previous case using a clock tree driven by a direct connection of the full rate transmit FPGA output clock to the FPGA center clock mux. Once the CTC FIFO is bypassed, the recovered clock needs to control the write port of the RX FIFO. The recovered clock of each channel may need to drive a separate local or global clock tree (i.e., up to four local or global clock trees per quad). The clock tree will then drive the FPGA receive clock input to control the read port of the RX FIFO. The reason for bypassing the CTC FIFO in this case is most likely for doing multi-channel alignment in the FPGA core. It implies that CTC using an elastic buffer will be done in the FPGA core. The CTC FIFOs can be written by either the recovered clocks or by a master recovered clock. The read of the CTC FIFO will be done using the TX clock via the TX clock tree.

Case I_c: 8/10bit, FPGA Bridge FIFOs Bypassed

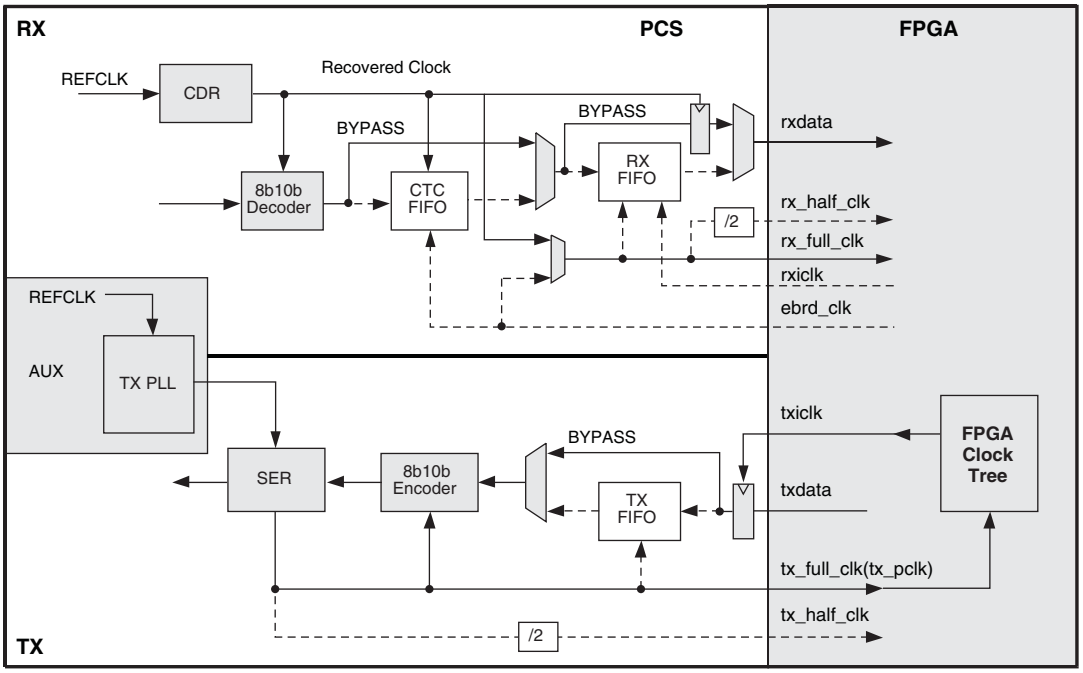
Figure 29. 8/10-Bit, RX/TX FIFO Bypassed



1. The TX channel clocking is similar to the previous two cases. On the RX channel, the FPGA input clock is now ebrd_clki. The FPGA TX clock tree drives this clock. In this case, ebrd_clki is automatically routed by the software.

Case I_d: 8/10bit, CTC FIFO and FPGA Bridge FIFOs Bypassed

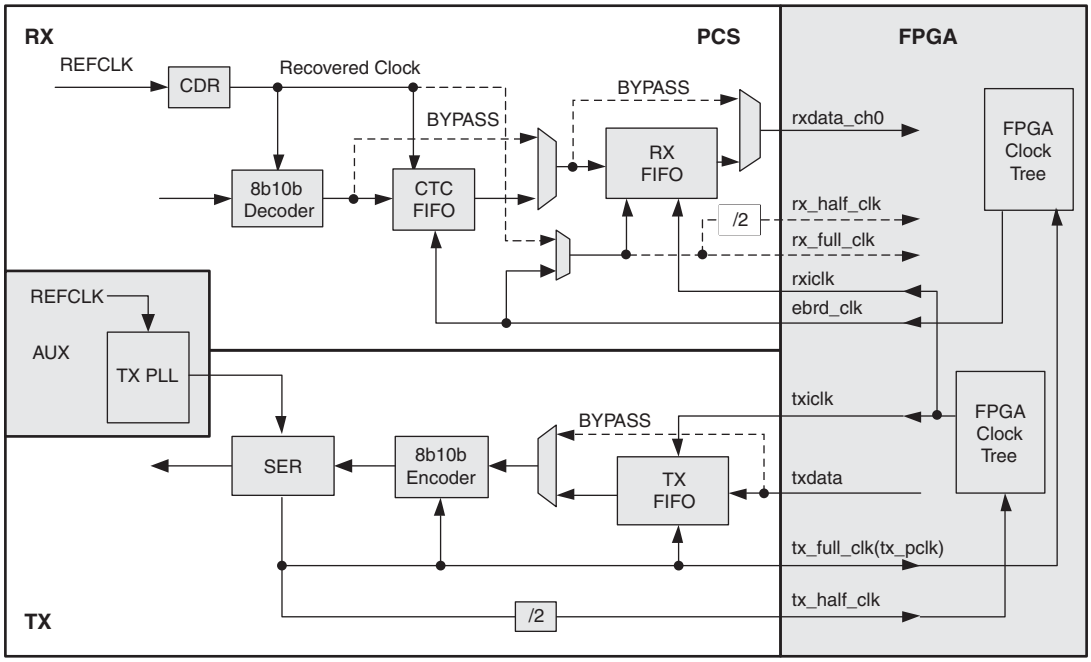
Figure 30. 8/10-Bit, CTC FIFO and RX/TX FIFOs Bypassed



1. FPGA clock trees can be interchangeably thought of as clock domains in this case. The TX channel clock¬ing is similar to the previous three cases. On the RX channel, the recovered channel RX clock is sent out to the FPGA. This case is useful for supporting video applications.

Case II_a: 16/20bit, CTC FIFO NOT Bypassed

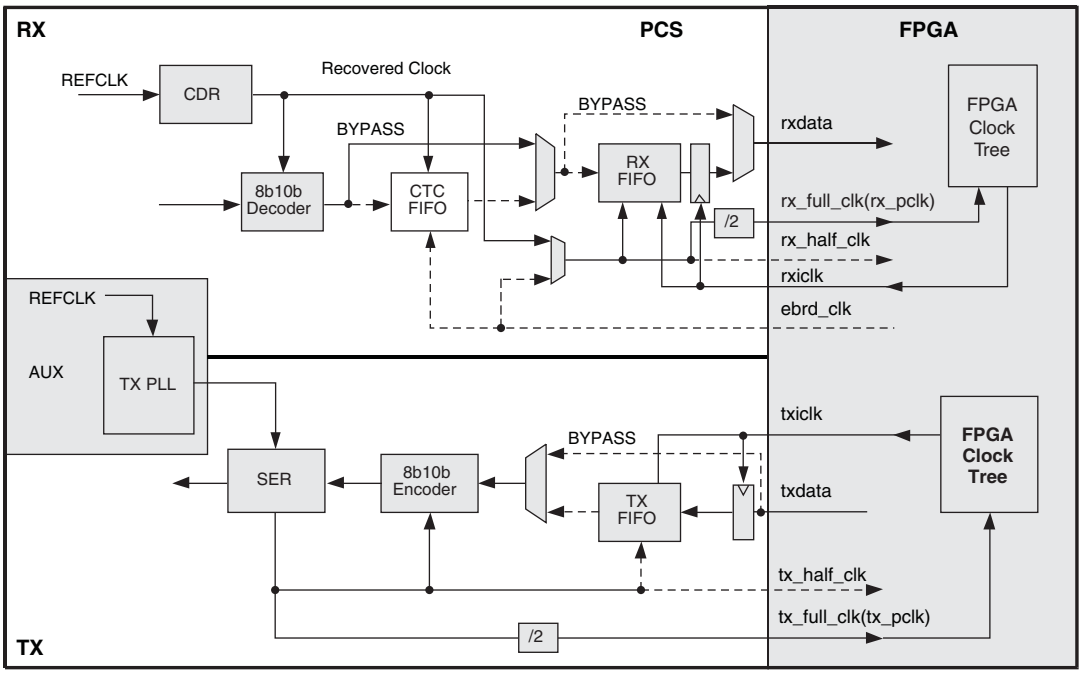
Figure 31. 16/20-Bit, CTC FIFO and RX/TX FIFOs NOT Bypassed



1. The TX FIFO acts both as a Phase Shift FIFO and Upsample FIFO in this case.
2. The RX FIFO acts both as a Phase Shift FIFO and Downsample FIFO in this case.
3. This is a very common single channel use case when the FPGA is unable to keep up with full byte frequency. Two clock trees are required. These clock trees are driven by direct access of transmit full-rate clock and transmit half-rate clock to the FPGA clock center mux. The full-rate clock tree drives the CTC FIFO read port and the RX FIFO write port. The half-rate clock tree drives the RX FIFO and the FPGA logic.

Case II_b: 16/20bit, CTC FIFO Bypassed

Figure 32. 16/20-Bit, CTC FIFO Bypassed



1. The TX FIFO acts both as a Phase Shift FIFO and Upsample FIFO in this case
2. The RX FIFO is acting both as a Phase Shift FIFO and Downsample FIFO in this case.
3. This is a very common multi-channel alignment use case when the FPGA is unable to keep up with full byte frequency. The receive clock trees (up to four) can be local or global. They are running a half-rate clock. The transmit clock tree is driven by direct access of the transmit half-rate clock to the FPGA clock center mux.

SERDES/PCS Block Latency

Table 16 provides transmit and receive latencies, respectively, in the SerDes as well as different stages inside the PCS. The latency is in number of parallel word clocks.

Table 16. Transmit/Receive SerDes/PCS Latency

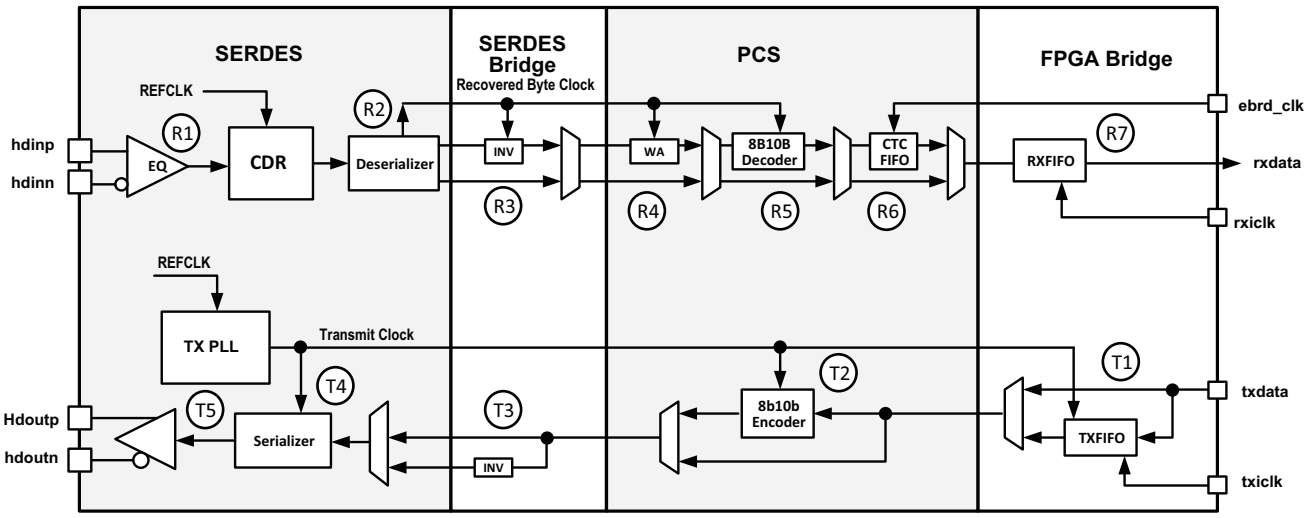
Item	Transmit Data Latencies1	Description	Min	Typ	Max	Exact	Byp	Unit4
T1	FPGA Bridge	Gearing Disabled(with different clocks)	1	3	5	N/A	1	byte_clk
		Gearing Disabled(with same clocks)	N/A	N/A	N/A	3	1	byte_clk
		Gearing Enabled2	1	3	5	N/A	N/A	word_clk
T2	8b10b Encoder		N/A	N/A	N/A	2	1	byte_clk
T3	SerDes Bridge		N/A	N/A	N/A	2	1	byte_clk
T4	Serializer	x8 mode (BUS8BIT_SEL = 0)	N/A	N/A	N/A	15 + Δ1	N/A	UI + ps
		x10 mode (BUS8BIT_SEL = 1)	N/A	N/A	N/A	18 + Δ1	N/A	UI + ps
T5	Driver	De-emphasis ON	N/A	N/A	N/A	1 + Δ2	N/A	UI + ps
		Pre-emphasis OFF	N/A	N/A	N/A	0 + Δ3	N/A	UI + ps
	Receive Data Latencies[2]	**Description**	**Min**	**Typ**	**Max**	**Exact**	**Byp**	**Units 10**
R1	Input Buffer	Equalization ON	N/A	N/A	N/A	Δ1	N/A	UI + ps
		Equalization OFF	N/A	N/A	N/A	Δ2	N/A	UI + ps
R2	De-Serializer	x8 mode (BUS8BIT_SEL = 0)	N/A	N/A	N/A	10 + Δ3	N/A	UI + ps
		x10 mode (BUS8BIT_SEL = 1)	N/A	N/A	N/A	12 + Δ3	N/A	UI +ps
R3	SerDes Bridge		N/A	N/A	N/A	2	1	byte_clk
R4	Word Aligner3		N/A	N/A	N/A	4	0	byte_clk
R5	8B10B Decoder		N/A	N/A	N/A	1	1	byte_clk
R6	CTC		7	15	23	N/A	1	byte_clk
R7	FPGA Bridge	Gearing Disabled (with different clocks)	1	3	5	N/A	1	byte_clk
		Gearing Disabled (with same clocks)	N/A	N/A	N/A	3	1	byte_clk
		Gearing Enabled	1	3	5	N/A	N/A	word_clk

1. Δ1 = -100 ps, Δ2 = +88 ps, Δ3 = +112 ps.
2. Δ1 = +118 ps, Δ2 = +132 ps, Δ3 = +721 ps.
3. Table 16 shows word aligner latency depending on word alignment offset. The exact offset can be found in channel status register.

Table 17. Workd aligner Latency vs. Offset

wa_offset	Latency (Word Clocks)
0	4.0
1	3.9
2	3.8
3	3.7
4	3.6
5	3.5
6	3.4
7	3.3
8	3.2
9	3.1

Figure 33. Transmitter and Receiver Latency Block Diagram



SERDES Client Interface

Introduction

LFE5UM/LFE5UM5G provides SCI interface to the core thru general routing.

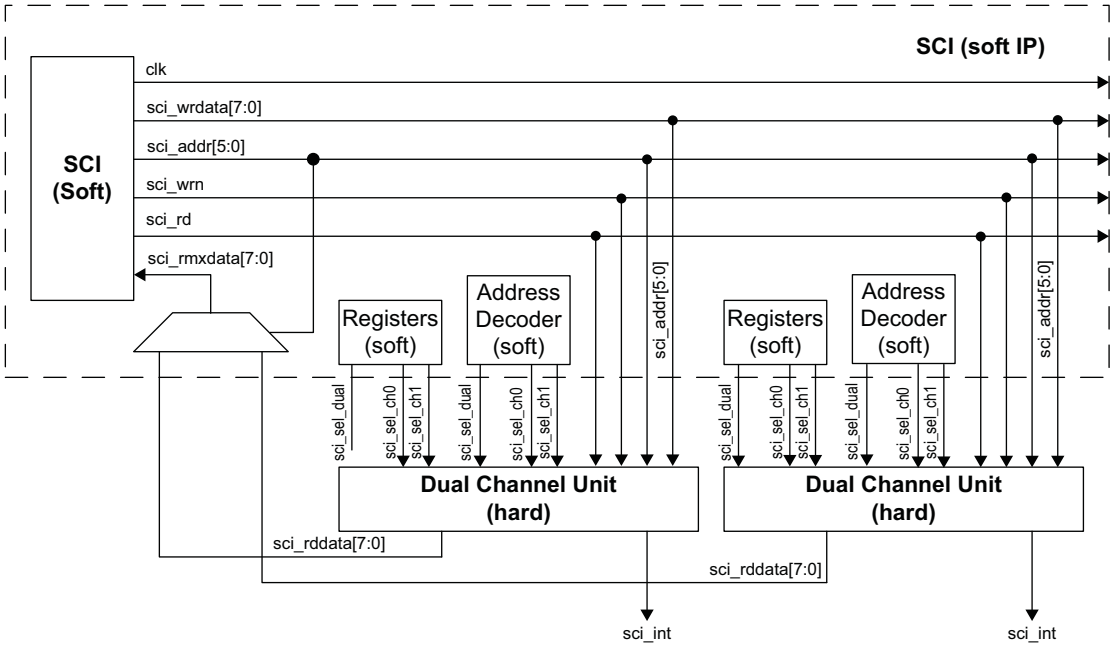
The settings in most of the configuration bits are shadowed into registers that can be read/write by the FPGA core. LFE5UM/LFE5UM5G provides the interface signals that can be controlled by the core. User logic can change these settings via his logic. Care should be taken when the settings are changed, and is recommended and, most of the time, required to perform a reset to the SERDES and PCS, in order to obtain correct operation after the changes.

Table 18 describes the addressing map of control registers and Figure 34 shows the block diagram of SCI Interface.

Table 18. SCI address map for up to four SerDes/PCS duals

Address Bits	Description
SCIADDR[5:0]	**Register address bits:** 000000 = Select register 0 000001 = Select register 1 … 111110 = Select register 62 111111 = Select register 63
SCIADDR[8:6]	**Channel address bits** 000 = Select channel 0 001 = Select channel 1 010 = Reserved 011 = Reserved 100 = Select aux channel 101 = Reserved 110 = Reserved 111 = Reserved
SCIADDR[10:9]	**Dual address bits** 00 = Select Dual0 01 = Select Dual1 10 = Reserved 11 = Reserved

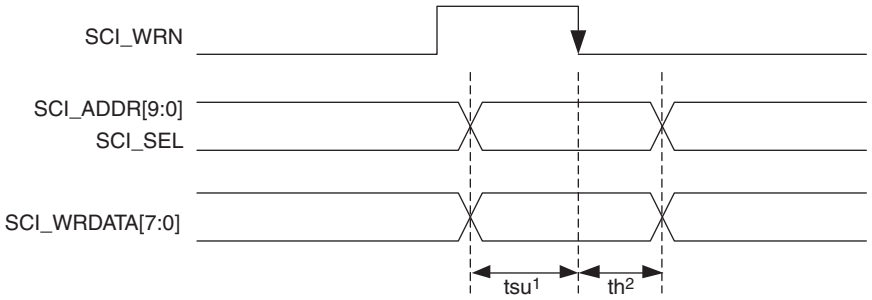
Figure 34. LFE5UM/LFE5UM5G SCI(SerDes Client Interface) Block Diagram



The SCI is an 8-bit asynchronous control interface to access the SERDES/PCS channels and TX PLL inside the DCU. The scisel lines are used to select a resource and then the sciwstn or sci_rd ports are used to latch in the command to write or read 8-bit data from sci_wrdata or to sci_rddata respectively. In addition there is a sci_int port to allow an interrupt to be sent from any of the DCU resources to the FPGA. Refer to Table 18 for ports description.

Read and write operations through this interface are asynchronous. In the WRITE cycle the write data and write address must be set up and held in relation to the falling edge of the SCI_WR. In the READ cycle the timing has to be in relation with the SCI_RD pulse. Figure 35 and Figure 36 show the WRITE and READ cycles, respectively.

Figure 35. SCI WRITE Cycle, Critical Timing



1. tsu is the setup time for address and write data prior to the falling edge of the write strobe.
2. th is the hold time for address and write data after the falling edge of the write strobe.

Note: To avoid accidental writing to control registers, registers should be used at the SCI input ports to drive them low at power-up reset.

Figure 36. SCI READ Cycle, Critical Timing

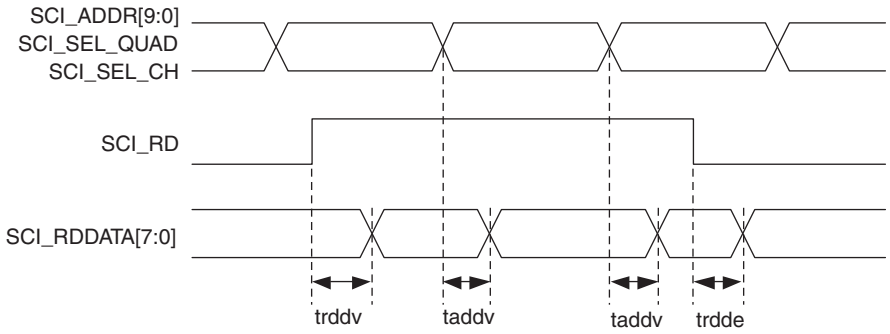


Table 19. Timing Parameter

Parameter	Typical Value	Units
t_{SU}, trddv, taddv	1.1	ns
t_H, trdde	0.9	ns

The SCI interface is as simple as memory read/write. Here is an example of the pseudo code:

Write:

- Cycle 1: Set sci_addr[5:0], sciw_data[7:0], sci_sel = 1'b1
- Cycle 2: Set sci_wrn from 0 ≥ 1
- Cycle 3: Set sci_wrn from 1 ≥ 0, sci_sel = 1'b0

Read:

- Cycle 1: Set sci_addr[5:0], sci_sel = 1'b1
- Cycle 2: Set sci_rd from 0 ≥ 1
- Cycle 3: Obtain reading data from sci_rddata[7:0]
- Cycle 4: Set sci_rd from 1 ≥ 0

Interrupts and Status

The status bit may be read via the SCI, which is a byte wide and thus reads the status of eight interrupt status signals at a time. The SCI_INT signal goes high to indicate that an interrupt event has occurred. The user is then required to read the DIF status register that indicates whether the interrupt came from the quad or one of the channels. This register is not cleared on read. It is cleared when all interrupt sources from the quad or channel are cleared. Once the aggregated source of the interrupt is determined, the user can read the registers in the associated quad or channel to determine the source of the interrupt. Table 19 and Table 20 list all the sources of interrupt.

Table 20. Dual Interrupt Sources

Dual SCI_INT Source	Description	Register Name
int_dl_out	Dual Interrupt. If there is an interrupt event anywehere in the dual, this register bit will be active. This register bit is cleared when all interrupt events have been cleared	PCS Dual Status Register DL_20
int_ch_out[1:0]	Channel Interrupt. If there is an interrupt event anywhere in the respective channel, this register bit will be active. These register bits are cleared when all interrupt sources in the respective channel have been cleared.	PCS Dual Status Register DL_20
ls_sync_statusn[1:0]_int ls_sync_status[1:0]_int	Link Status Low (out of sync) channel interrupt & Link Status High (in sync) channel interrupt	PCS Dual Status Register DL_22_INT
~PLOL, PLOL	Interrupt generated on ~PLOL and PLOL - PLL Loss of Lock	PCS Dual Status Register DL_25

Table 21. Channel Interrupt Sources

Channel SCI_INT Source	Description	Register Name
fb_tx_fifo_error_int fb_rx_fifo_error_int cc_overrun_int cc_underrun_int	FPGA Bridge TX FIFO Error Interrupt FPGA Bridge RX FIFO Error Interrupt CTC FIF Overrun and Underrun Interrupts	PCS Channel General Interrupt Status Register CH_INT_23
pci_det_done_int rlos_lo_int ~rlos_lo_int rlol_int ~rlol_int	Interrupt generated for pci_det_done Interrupt generated for rlos_lo Interrupt generated for ~rlos_lo Interrupt generated for rlol Interrupt generated for ~rlol	PCS Dual Status Register DL_INT_20

Dynamic Configuration of the SERDES/PCS Dual

The SERDES/PCS dual can be controlled by registers that are accessed through the optional SERDES Client Interface.

When controlled by the configuration memory cells, it is a requirement that the SERDES/PCS duals must reach a functional state after configuration is complete, without further intervention from the user. This means that any special reset sequences that are required to initialize the SERDES/PCS dual must be handled automatically by the hardware. In other words, use of the SCI is optional. The SERDES/PCS dual does NOT assume that the soft IP is present in the FPGA core.

SERDES Debug Capabilities

PCS Loopback Modes

The LFE5UM/LFE5UM5G family provides three loopback modes controlled by control signals at the PCS/FPGA interface for convenient testing of the external SERDES/board interface and the internal PCS/FPGA logic interface.

RX-to-TX Serial Loopback Mode

This mode loops serial receive data back onto the transmit buffer without passing through the CDR or de-serializer. Select the RX-to-TX Serial Loopback option in the Clarity Designer (System Builder/Planner) GUI and SW will set necessary control bits.

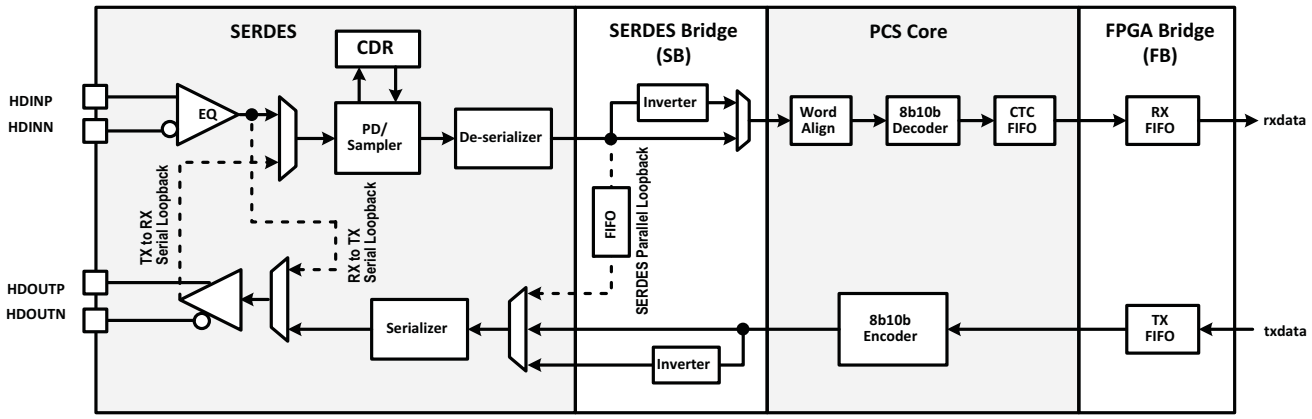
TX-to-RX Serial Loopback Mode

This mode loops back serial transmit data back onto the receiver CDR block. Select the TX-to-RX Serial Loopback option in the Clarity Designer (System Builder/Planner) GUI and SW will set necessary control bits.

SERDES Parallel Loopback Mode

Loops parallel receive data back to the transmit data path without passing through the PCS logic. When disabled in the Clarity Designer (System Builder/Planner) GUI, the parallel loopback mode can be dynamically controlled from the FPGA core control signals sb_felb_c and sb_felb_rst_c. If the dynamic feature of this loopback mode is not used, the two control signals should be tied to ground. When the loopback mode is enabled in the Clarity Designer (System Builder/Planner) GUI, the control register bit for the loopback mode is set. The control signals from the FPGA core, sb_felb_c and sb_felb_rst_c, are not available in the PCS module.

Figure 37. LFE5UM/LFE5UM5G SERDES Three Loopback Modes



ECO Editor

The Diamond ECO Editor supports all of the attributes of the primitives. The ECO Editor includes all settings that the user sees in the GUI tab, including pre-defined setting based on protocols.

ECO Editor is a graphical representation of all the user configurable settings, and allows the user to change the settings in that environment. Care must be taken on making these changes, because the changes will not be checked by the design SW tools for correctness.

ECO Editor changes only settings inside the SERDES. Changes in the SERDES that would affect the external connection requires the user to re-run the SW flow. Changes made in ECO Editor that do not change external connection will just require the user to re-generate the bitstream with the same placement and routing done in the core.

Please refer to ECO Editor User Guide in Diamond for more information.

Other Design Considerations

Simulation of the SERDES/PCS

All SERDES/PCS simulation models are located in the installation directory, under ….\cae_library\simulation\black-box directory

16/20-Bit Word Alignment

The SERDES/PCS recognizes byte boundary by aligning to COMMA characters, but the PCS receiver cannot recognize the 16-bit word boundary. When Word Aligner is enabled, the PCS can only do BYTE alignment. The 16-bit word alignment should be done in the FPGA fabric and is fairly straight forward. The simulation model works in the same way. It can be enhanced if users implement an alignment scheme as described below.

For example, if transmit data at the FPGA interface are:

```
YZABCDEFGHIJKLM... (each letter is a byte, 8-bit or 10-bit)
```

Then the incoming data in PCS after 8b10b decoder and before rx_gearbox are:

```
YZABCDEFGHIJKLM...
```

After rx_gearbox, they can become:

1. ZY}{BA}{DC}{FE}{HG}{JI}{LK}....

or

2. AZ}{CB}{ED}{GF}{IH}{KJ}{ML}...

Clearly, sequence 2 is not aligned. It has one byte offset, but 16/20-bit alignment is needed. Let's say the special character 'A' should be always placed in the lower byte.

character 'A' should be always placed in the lower byte.

Flopping one 20-bit data combines with the current 16/20-bit data to form 32/40-bit data as shown below:

1. `{DCBA}{HGFE}{LKJI}...`

```
        ^
        | **Found the A in lower 10-bit, set the offset to '0`, send out aligned
        data 'BA`
```

Next clock cycle:

```
{FEDC}{JIHG}{NMLK}...
        ^
        | **send out aligned data 'DC'
```

etc.

After the 16/20-bit alignment, the output data are:

```
{ZY}{BA}{DC}{FE}{HG}{JI}{LK}...
```

2. `{CBAZ}{GFED}{KJIH}....`

```
          ^
          | **Found the A in upper 10-bit, set the offset to '10', send out aligned
        data 'BA'
```

Next clock cycle:

```
{EDCB}{IHGF}{MLKJ}...
  ^
  | **send out aligned data 'DC'
```

etc.

After the 20-bit alignment, the output data are:

```
{ZY}{BA}{DC}{FE}{HG}{JI}{LK}...
```

Note: The LSB of a 8/10-bit byte or a 16/20-bit word is always transmitted first and received first.

Unused Dual/Channel and Power Supply

On unused dual and channels, VCCA should be powered up. VCCHRX, VCCHTX, HDINP/N, HDOUTP/N and REF-CLKP/N should be left floating. Unused channel outputs are tristated, with approximately 10 KOhm internal resistor connecting between the differential output pair. During configuration, HDOUTP/N are pulled high to VCCHTX.

Even when a channel is used as Rx Only mode or Tx Only mode, both the VCCHTX and VCCHRX of the channel must be powered up. Unused SERDES is configured in power down mode by default.

SERDES/PCS Reset

Reset and Power-Down Control

The SERDES dual has reset and power-down controls for the entire macro and also for each transmitter and receiver as shown in Figure 38. The reset signals are active high and the power-down is achieved by driving the pwrup signals low. The operation of the various reset and power-down controls are described in the following sections.

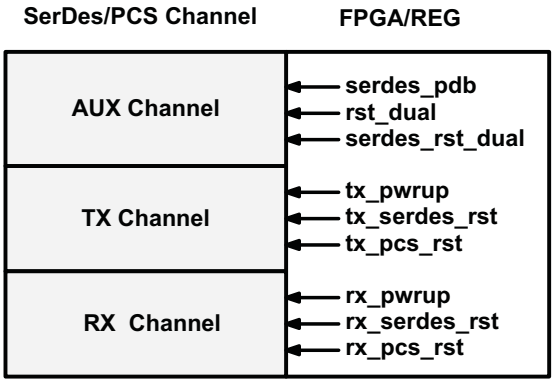
Note: When the device is powering up and the chip level power-on-reset is active, the SERDES control bits (in the PCS) will be cleared (or will take on their default value). This will put the SERDES quad into the power-down state.

After configuration is complete, the whole device, including the SerDes/PCS duals, reaches a functional state without further intervention from the user. The reset sequence to initialize the dual is handled by the hardware.

Reset and Power-Down Control

The SerDes dual has reset and power-down controls for the whole macro as well as individual reset and power-down controls for each transmitter and receiver as shown in Figure 38. The reset signals are active high and the power-down signals are active low. The operation of the various reset and power-down controls are described in following sections.

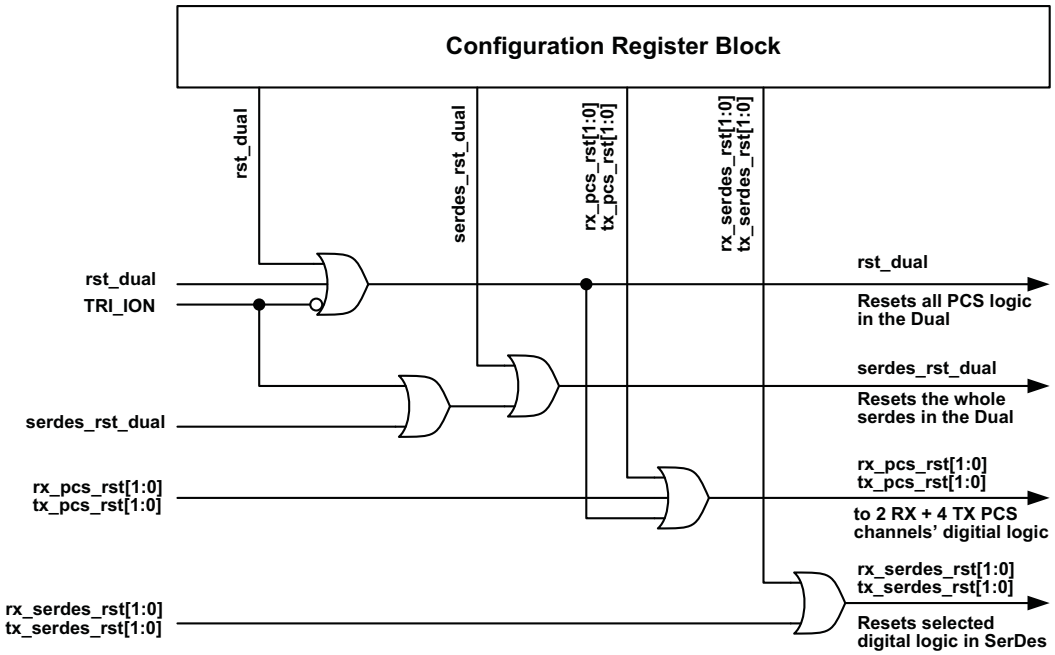
Figure 38. Reset and Power-Down Control



Reset Generation

Reset generation and distribution are handled by the clocks and the resets block. The internal reset signal for all blocks within the PCS is active low with an asynchronous assert and a synchronous de-assert. The reset logic is shown in Figure 39 and the corresponding table is in Table 22.

Figure 39. Global Reset Diagram



Note in Figure 39, each of the reset signal controlled by FPGA logic has a corresponding register bit. These register bits can be accessed thru the SCI interface, and user can write "1" to the register to activate the reset, and write "0" to release it.

Table 22 describes the part of SERDES/PCS that are affected by the different resets.

Table 22. Reset Table

Reset Signal	PCS TX	PCS RX	SERDES TX	SERDES RX	TX PLL	PLOL	RX CDR	CONTROL REGs
tx_pcs_rst[1:0] (fpga)	x							
tx_pcs_rst[1:0] (register)	x							
rx_pcs_rst[1:0] (fpga)		x						
rx_pcs_rst[1:0] (register)		x						
rst_dual (fpga)	x	x						
rst_dual (register)	x	x						
serdes_rst_dual (fpga)			x	x	x	x	x	
serdes_rst_dual (register)			x	x	x	x	x	
rx_serdes_rst[1:0] (fpga)				x			x	
rx_serdes_rst [1:0] (register)				x			x	
tx_serdes_rst (fpga)			x		x	x		
tx_serdes_rst (register)			x		x	x		
(Configuration)	x	x	x	x	x	x	x	x

Table 23 describes the power down signals.

Table 23. Power-Down Control Description

Signal		Description
FPGA	Register	
serdes_pd		Active-low asynchronous input to the SERDES quad, acts on all channels including the auxiliary channel. When driven low, it powers down the whole macro including the transmit PLL. All clocks are stopped and the macro power dissipation is minimized. After release, both the TX and RX reset sequences should be followed.
tx_pwrup_ch[0:3]_c	tpwrup[0:3]	Active-high transmit channel power-up – Powers up the serializer and output driver. After release, the TX reset sequence should be followed.
rx_pwrup_ch[0:3]_c	rpwrup[0:3]	Active-high receive channel power-up – Powers up CDR, input buffer (equalizer and amplifier) and loss-of-signal detector. After release, the RX reset sequence should be followed.

Table 24. Power-Down/Power-Up Timing Specification

Parameter	Description	Min.	Typ.	Max.	Units
t_{PWRDN}	Power-down time after serdes_pd	20			ns
t_{PWRUP}	Power-up time after serdes_pd	20			ns

Reset Sequence

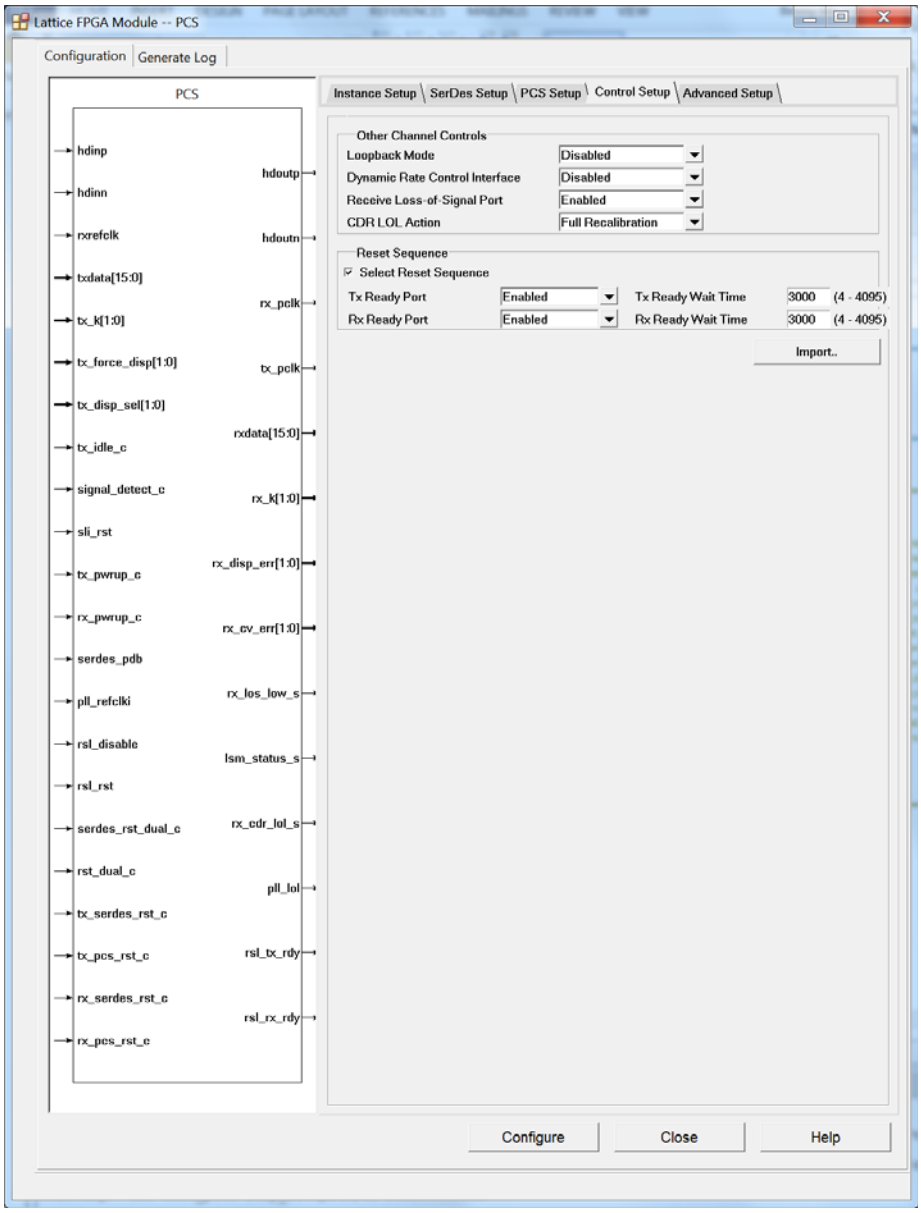
There are many different reasons uses may want to assert a reset, but there are times when he wants to only reset one channel, or just the Rx or Tx alone. The different resets provide a lot of flexibility for the users. This is especially important considering if the link shows only Rx issue, he does not have to reset the Tx channel, in order to maintain connection to the far end, or visa versa. Also, it is important that when a Dual is connected to two different links, clearing one Tx channel should keep the other channel running. Since two Tx channels in the Dual is sharing the same TxPLL, and if TxPLL is not the source of problem, then the users do not need to reset everything associated with the channel he wants to re-initialize.

However, users have to maintain proper sequencing between PCS and SERDES and sometimes PLL/CDR so that the three blocks are properly synchronized. The proper sequence should involve:

1. PLL needs to be locked first. This goes with TxPLL and CDR PLL. In the case of both are being reset, CDR PLL should wait for TxPLL lock, before it release the reset on it to start to clock on the CDR.
2. Once the PLLs are locked, the SERDES reset should be released. This properly generates the byte clock that synchronizes to the bit clock inside the SERDES
3. Last to be released is the PCS reset, where all the user logic.

Clarity Designer in the Diamond Design Software provides an option for the user to select a Reset Sequence Logic module, implemented in the FPGA logic, to synchronize the resets above, so that all three blocks: PCS, SERDES and PLL/CDR, operate correctly. This option can be found in the Control Setup tab of the PCS module in Clarity Designer, as shown in Figure 40.

Figure 40. Reset Sequence in Clarity Design, PCS Module

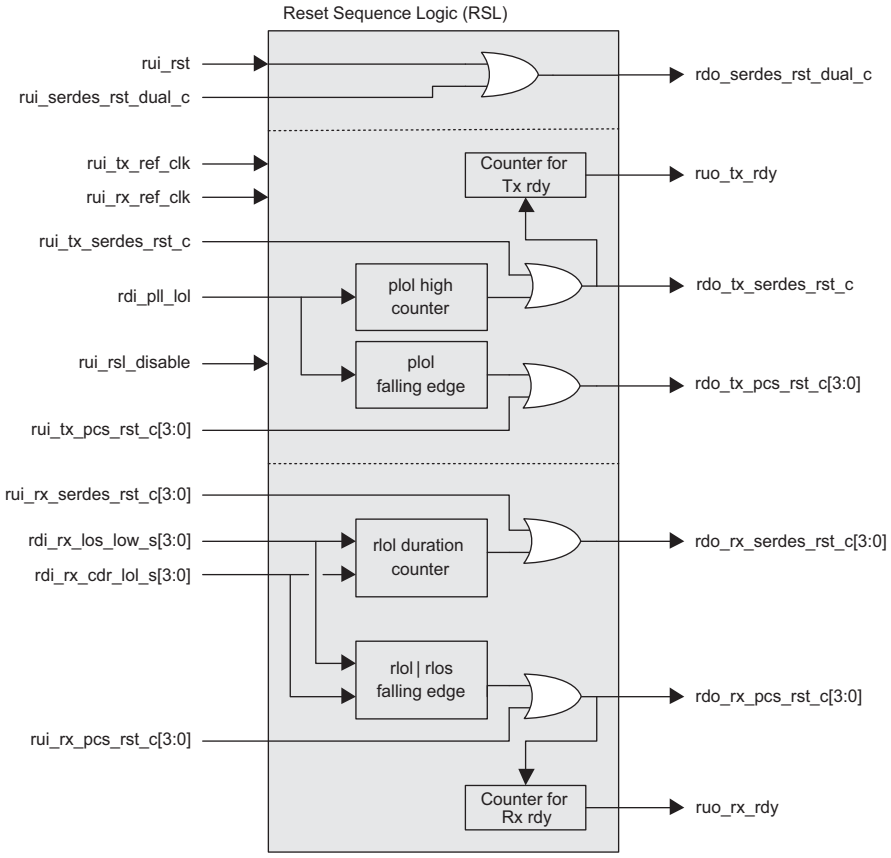


The Reset Sequence Logic module is selected by default. Two additional ports are added to the PCS module: rsl_disable and rsl_rst. Rsl_rst resets all the logic in the module, and rsl_disable is used to de-assert all outputs in the Reset Sequence Logic module.

Also by default, both rsl_tx_rdy and rsl_rx_rdy ports are not enabled. These ports can be enabled to allow the user logic to monitor the state of the PCS channels. The option of wait time is used to delay these signals to be active, to filter out any event that can cause these signals to toggle while the TxPLL and Rx CDR get locked.

The Reset Sequence Logic block diagram is shown in Figure 41.

Figure 41. Reset Sequence Logic (RSL) Block Diagram



This block can be seen in RTL code generated by the Clarity Design in Diamond Design Software.

Clarity Design treats each user interface as one instance, and only one RSL is created per instance. When an instance is multi-channel, the RSL needs to control channels within one DCU, or channels across multiple DCUs. When 2 single-channel instances are created with RSL in each, and they are placed in same DCU (example: 2 PCIe X1 instances in one DCU), these 2 RSLs need to be merged to control one DCU. These are described in following sections:

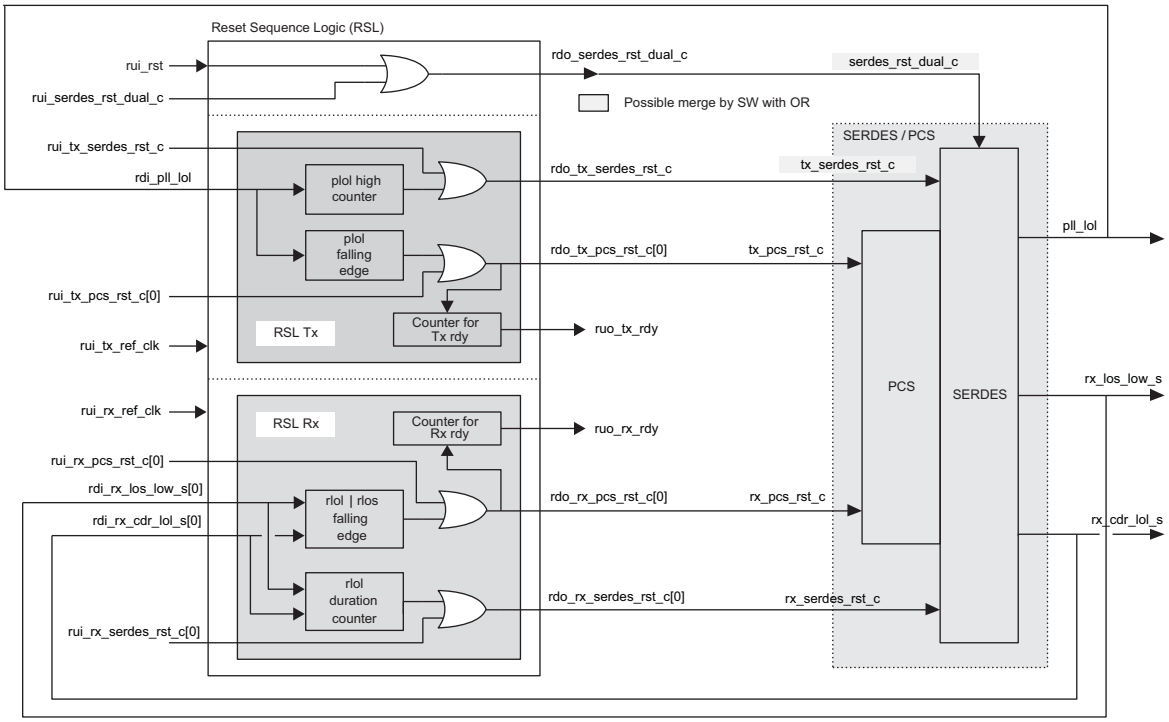
Connectivity for Single-Channel RSL

The connectivity of one channel RSL to one DCU channel is shown in 9-42. The port name prefixes for RSL block uses the following convention:

- rdi_ : RSL inputs ports that are driven by DCU outputs (including those merged by SW OR gate)
- rui_ : RSL inputs coming from the user
- rdo_ : RSL outputs driving DCU inputs
- ruo_ : RSL outputs going out to user logic

Except for the above prefix, the RSL port names mostly match the corresponding DCU port names as assigned by the ASB Generator wrapper. The ports highlighted in green are common for both channels of the DCU and are to be merged by SW when two channels are packed into a DCU. Note that the RSL is positioned as an add-on to the existing PCS wrapper and hence the currently used PCS wrapper signal names are used instead of DCU primitive names.

Figure 42. RSL Connectivity to Single-Channel in One DCU



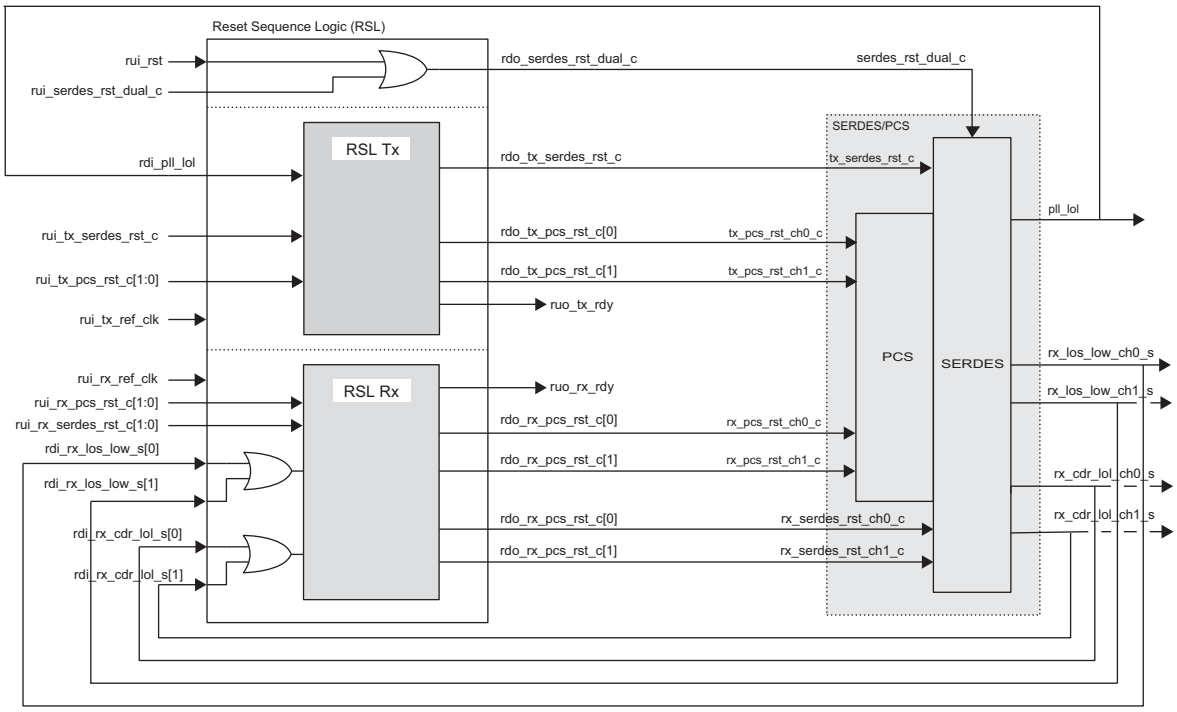
The example in Figure 42 shows Channel 0 is used in the DCU, and Channel 1 is not used. If channel 1 is also used for another instance, the RSL in Channel 1 needs to be merged with Channel 0, as described in later section.

Connectivity for Multiple-Channel RSL

The logic used to generate the reset commands is the same as used in Single-Channel RSL. As shown in Figure 43. The status outputs (LOL, LOS signals) from multiple channels are OR'ed together before applying to the RSL. The user override reset command for each channel is OR'ed with the reset command from the generation logic to drive the reset inputs of each channel.

The content and connectivity for 4-channel RSL is similar to the 2-channel case shown in Figure 9-43, except that there are four user override reset commands, one for each channel. There are four sets of reset signals to the four channels of the DCUs (2 DCU primitives are used for creating a 4-channel wrapper). There is an important difference to be noted in the way 4-channel PCS is generated. The 2-channel PCS wrapper brings out command and status ports consistent to the 2-channel DCU pin-outs, but the 4-channel wrapper does it differently. For 4-channel PCS wrapper, there is only one pll_lol output brought out even though there are 2 different outputs from each of the DCU primitives. Similarly there is only one serdes_rst_dual_c input and one tx_serdes_rst_c input into the wrapper, even though there are two of those inputs available for each of the DCUs in the wrapper. In short the 4-channel PCS wrapper has common Tx status and controls for both the DCU instances in it. These ports are connected to the status/control ports of DCU0 instance. The RSL content and connectivity are designed to match with this behavior of the Diamond PCS wrapper.

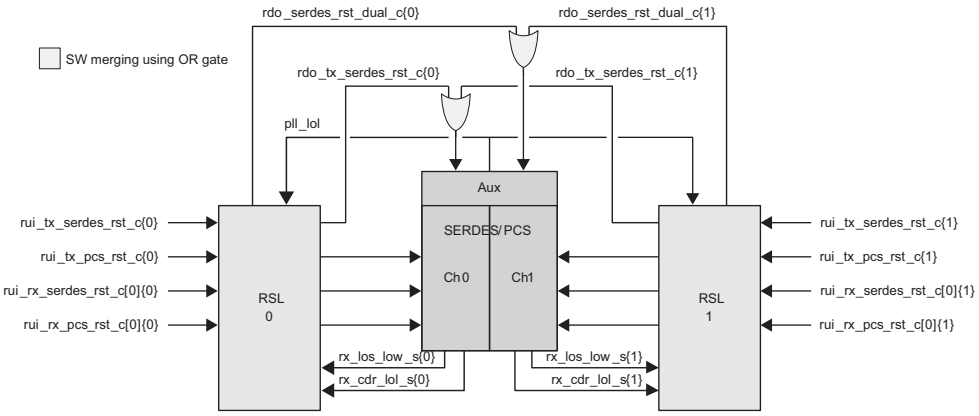
Figure 43. Connectivity of 2-Channel RSL in 2-Channel DCU



Connectivity of Multiple Instance in a DCU

When two Single-Channel instances are created in Clarity Designer, and they are placed into the same DCU, the SW needs to merge the common output signals using an OR gate between the two RSLs, as shown in Figure 44.

Figure 44. Merging Two Single-Channel PCS Modules with RSL Included



In the figure, indices inside curly braces are used to identify signals out of different Single-Channels PCS modules. The SW merge rules for connecting individual channel based signals to common dual signals are as follows.

- The status output from SERDES, pll_lol (ffs_plol) is connected to each of the channel's input port rdi_pll_lol.
- The macro reset command output from each of the channel's RSL (rdo_serdes_rst_dual_c) is ORed by the SW and connected to the serdes_rst_dual_c (ffc_macro_rst) input of the SERDES/PCS block.
- The SERDES transmit reset command output from each of the channel's RSL (rdo_tx_serdes_rst_c) is ORed by the SW and connected to the tx_serdes_rst_c (trst) input of the SERDES/PCS block.

- SW will merge the common dual signals (D_FFC_DUAL_RST, D_FFC_MACRO_RST, D_FFC_TRST, D_SCAN_RESET) based on the port name in DCUA primitive. All the drivers to each of the above named ports are ORed together and connected to that port.

Note that when two Single-Channel instances are connected to same DCU, any event that would cause one instance to reset the DCU would also reset the other instance. This is because the two instances placed within one DCU share the TxPLL in the DCU, and resetting the TxPLL by one instance would reset both instances at the same time.

Clarity Designer (System Builder/Planner) Overview

The Clarity Designer (System Builder/Planner) is an embedded tool in the Diamond Software. With the Clarity Designer tool, the user can specify the full function of the link he wants to implement. This includes SERDES channels, the PCS, the reference clock, and the IP and/or wrapper.

Diamond Overview

For LFE5UM/LFE5UM5G device family following primitive names are used.

DCU -> DCUA
EXTREF -> EXTREFB

SerDes/PCS HW is composed of three functional blocks, as shown in Figure 5. Because the functional block TxPLL has limited sharing, it implemented as part of the SerDes/PCS tab in the GUI.

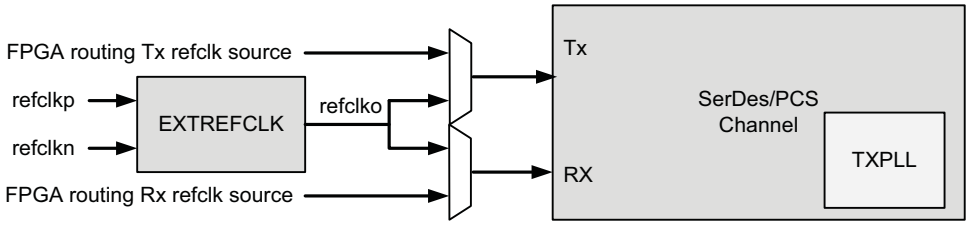
D0CH0: DCU0 Channel0

D0EXTREF: DCU0 EXTREF

Top Level Block Diagram

This section describes the usage of three blocks, SerDes/PCS channel, TXPLL and EXTREF. A top level diagram in Figure 45 shows how the three blocks are integrated.

Figure 45. Three Components Connectivity (for 1-lane link)



SerDes/PCS Channel

The SerDes physical channel and its corresponding PCS block are combined into single SerDes/PCS channel block, and multiple channels can be instantiated together as one instance, serving as multi-lane link. When a soft IP is included as part of the instance in the Clarity Designer (System Builder/Planner), Diamond software will route the connection signals from SerDes/PCS to/from the soft IP as defined in the IP.

Refer to Figure 5 for the SerDes/PCS channel primitive and Table 5 signal description.

TXPLL

This block is embedded into SerDes/PCS GUI tab but is captured as a separate primitive because the output of the PLL can be shared and used in SerDes/PCS channels outside of its own DCU.

EXTREF

This is the input buffer block for the reference clock input. PLLs connects to this buffer element when external reference clock source is used.

Single-ended reference clock function is supported, with proper settings of termination resistor and DC bias on the pins. Refer to Lattice technical note, Electrical Recommendations for Lattice SERDES(TN1114) for example circuit.

Connectivity: Captured by SW to provide the implicit connections that user defines. The routing of this is transparent to the users. All users have to do make the connections in the Clarity Designer (System Builder/Planner).

Clock routing: This is a dual based routing block that is captured in the SW. This block connects all the clocking signals in the SerDes/PCS.

SERDES/PCS Generation in Clarity Designer (System Builder/Planner)

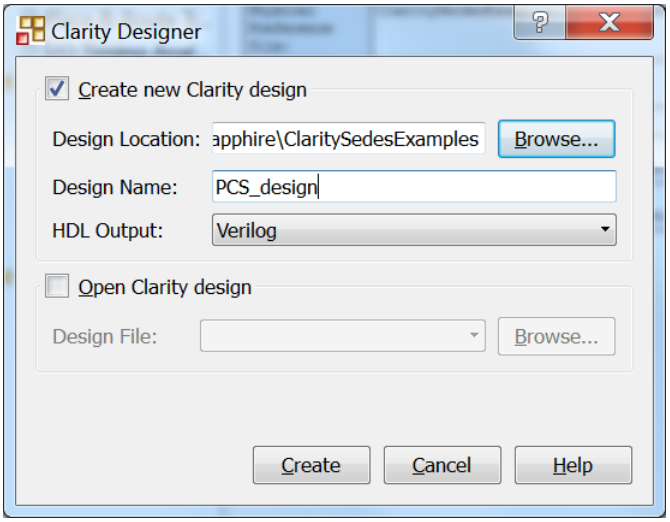
Clarity Designer (Clarity) is a new tool that targets to provide a more integrated module/function generator environment to the users.

The Clarity allows the user to design the functional block from the top level, pulling in all the necessary modules from the top, then goes into each module to customize what he needs. It is a top-down approach that allows the user looks at his design as a functional block that he can simulate and place in the device. An example of this is an PCIe controller, where the user uses the PCIe IP, the PIPE interface, and the Serdes/PCS blocks.

Once the blocks are identified, the user can push into the GUI of the block, which has similar interface of customizing SERDES on ECP3.

Figure 46 shows the view of Clarity when it is opened in Diamond.

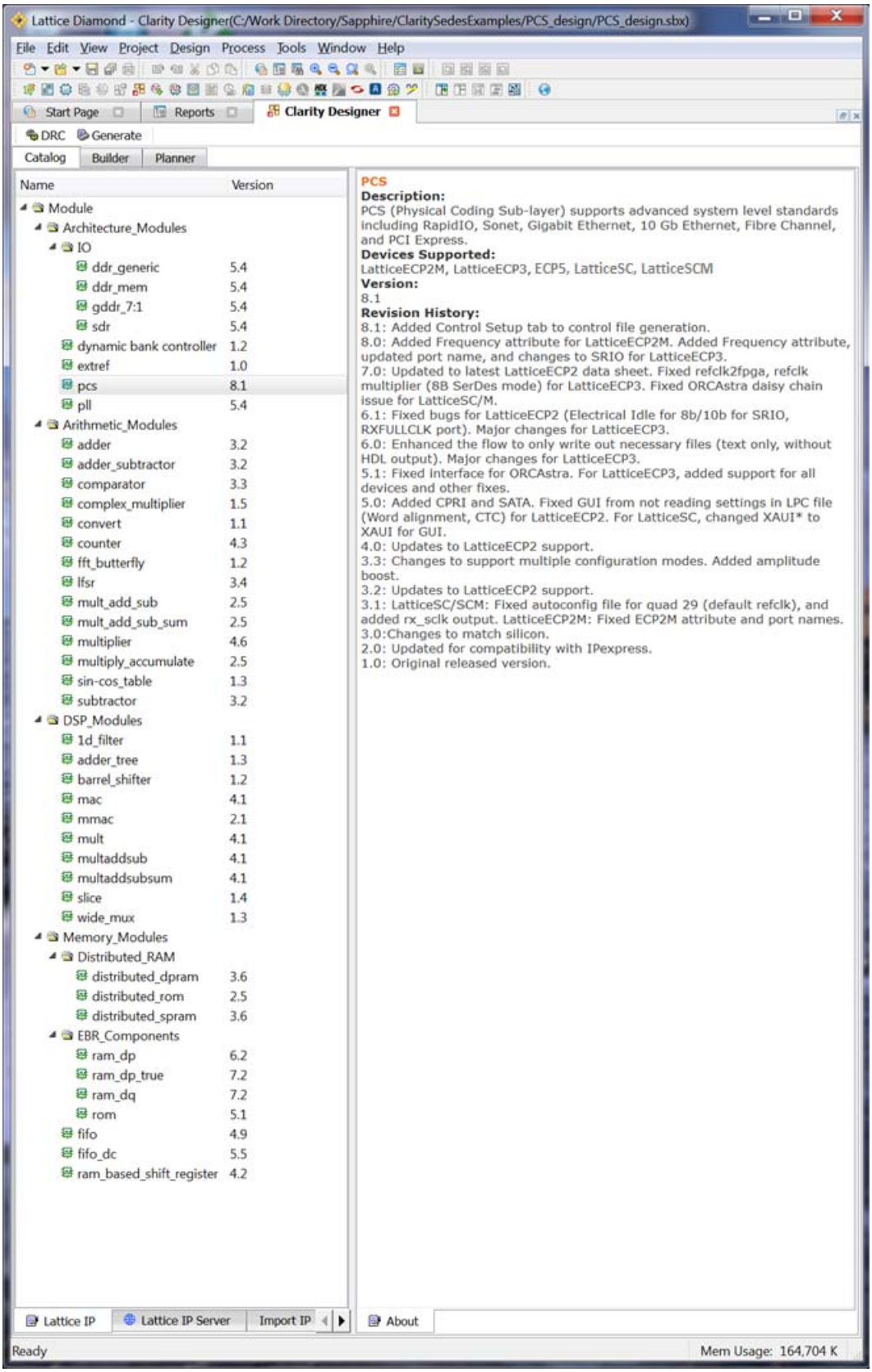
Figure 46. Clarity Designer Tool



The Clarity opens the project in the project directory specifies in the path.

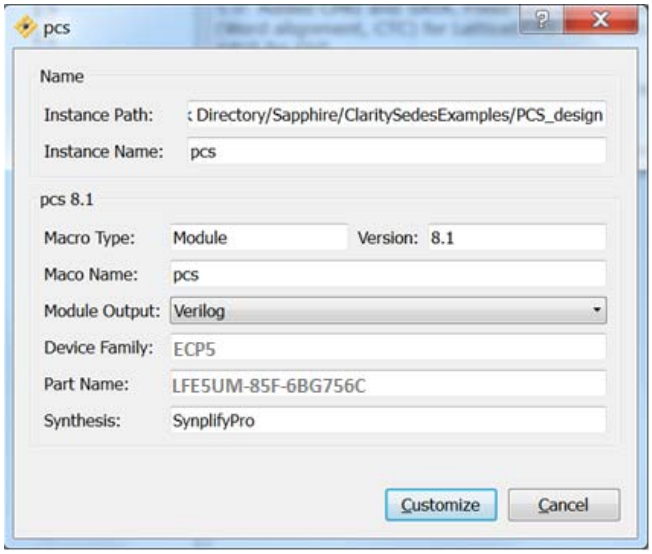
Once the project is created, a list of modules is shown in the catalog. This is shown in Figure 47.

Figure 47. Clarity Designer Catalog



When PCS is selected, a menu opens up to specify the details of the module to be created.

Figure 48. PCS Module in Clarity Designer



After all the information is filled in, the Custom button brings up the GUI for all the settings of that instance. The first tab is shown in Figure 49.

Figure 49. Instance Setup Tab

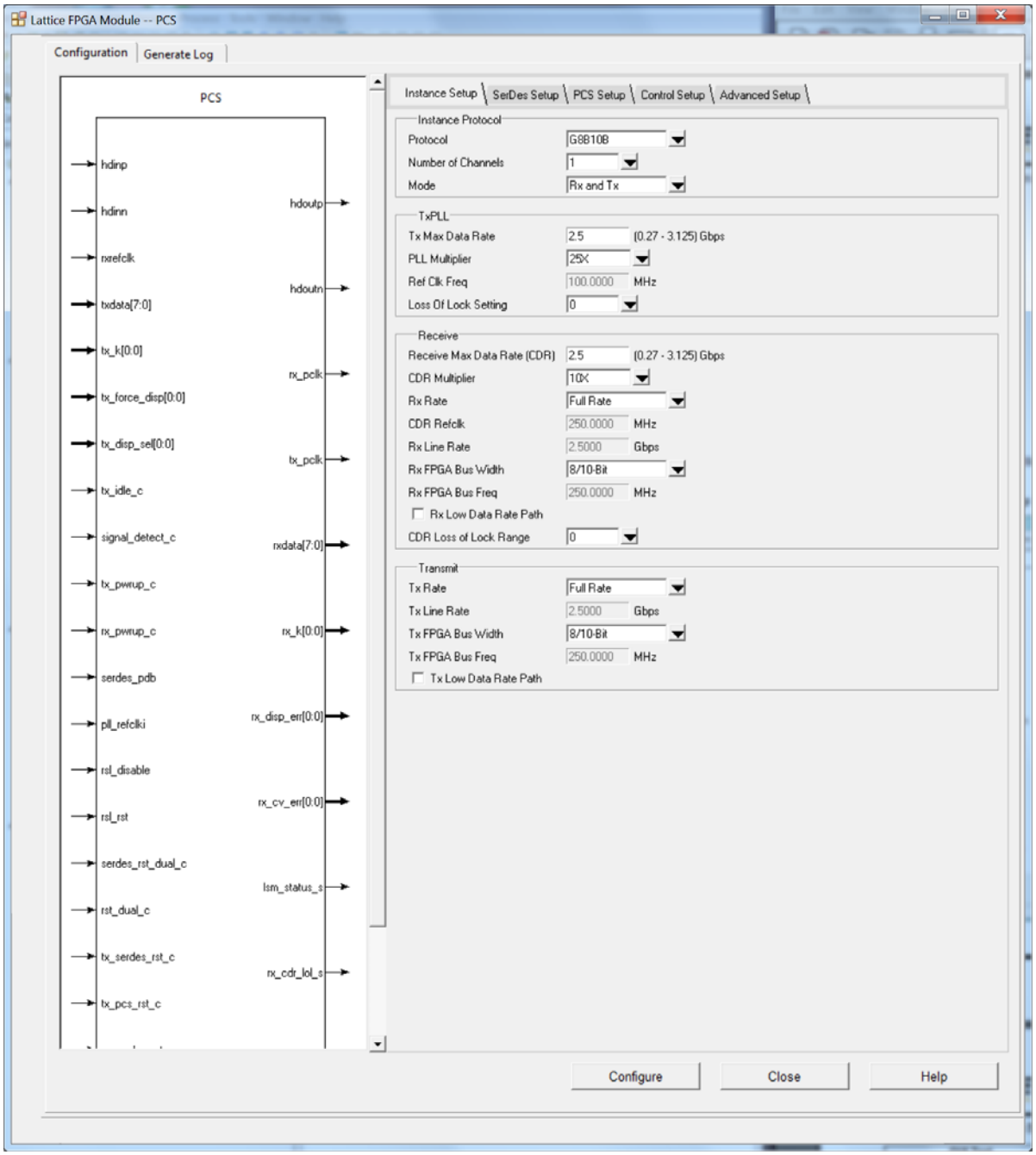


Table 25. Instance Setup Tab Description

GUI Text	Attribute Names	Values	Default Value
Number of Channels	NUM_CHS	1, 2, 4	1
Protocol	PROTOCOL	See Protocol Table below	G8B10B
Mode	CH_MODE	Rx and Tx, Rx Only, Tx, Only	Rx and Tx
Rx Datapath	RXLDR	On/Off	Off
Tx Datapath	TXLDR	On/Off	Off
Tx Max Data Rate	TX_MAX_RATE	0.27 - 3.125/5.0[1]	2.5
PLL Multiplier	TXPLLMULT	8X, 10X, 16X, 20X, 25X	25X
Ref Clk Freq	REFCLK_RATE	27 - 312.5	100
CDR Loss of Lock Range	CDRLOLRANGE	0-3	0
TxPLL Loss Of Lock Setting	TXPLLLOLTHRESHOLD	0-3	0
Tx Rate Divider	TX _RATE_DIV	Full Rate, Div2 Rate, Div11 Rate	Full Rate
Tx Line Rate	TX_LINE_RATE	135 Mbps - 3.125/5.0[1] Gbps	2.5 Gbps
Receive Max Data Rate (CDR)	CDR_MAX_RATE	0.27 - 3.125/5.0[1]	2.5
CDR Refclk	CDR_REF_RATE	27 - 312.5	Calculated
CDR Multiplier	CDR_MULT	8X, 10X, 16X, 20X, 25X	10X
Rx Line Rate	RX_LINE_RATE	135 Mbps - 3.125/5.0[1] Gbps	2.5 Gbps
Tx FPGA Bus Width	TX_DATA_WIDTH	8/10-Bit, 16/20-Bit	8/10- Bit
Tx FPGA Bus Freq	TX_FICLK_RATE	Calculated	Calculated
Rx Rate	RX_RATE_DIV	Full Rate, Div2 Rate, Div11 Rate	Full Rate
Rx FPGA Bus Width	RX_DATA_WIDTH	8/10-Bit, 16/20-Bit	8/10-Bit
Rx FPGA Bus Freq	RX_FICLK_RATE	Calculated	Calculated

1. For ECP5-5G family devices only.

Protocols selectable by the users:

- G8B10B
- PCIe
- GbE
- SGMII
- XAUI
- SDI
- CPRI
- JESD204
- 10BSER
- 8BSER
- eDP

Figure 50. SERDES Setup Tab

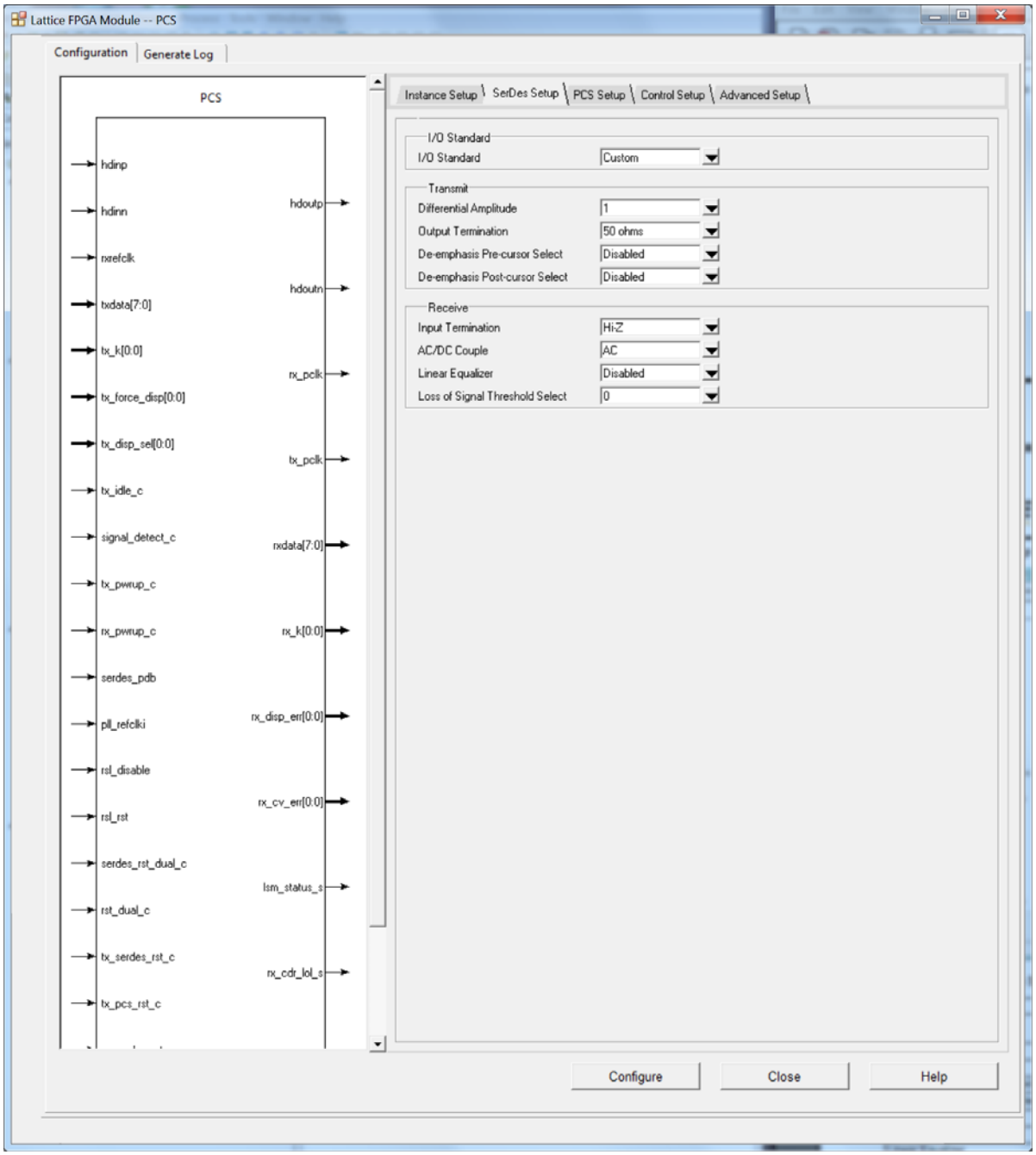


Table 26. SERDES Setup Tab Description

GUI Text	Attribute Names	Values	Default Value
I/O Standard	IO_TYPE	Protocol dependent, Custom	Based on Protocol
Differential Amplitude	TXAMPLITUDE	100 - 1300 mV, in 20 mV increments	1000
Output Termination	TXDIFFTERM	50, 75, 5K Ohms	5k Ohms
De-emphasis Pre-cursor Select	TXDEPRE	Disabled, 0-11	Disabled
De-emphasis Post-cursor Select	TXDEPOST	Disabled, 0-11	Disabled
Input Termination	RXDIFFTERM	50, 60, 75 Ohms, HiZ	Hi-Z
AC/DC Couple	RXCOUPLING	AC, DC	AC
Linear Equalizer	LEQ	Disabled, 0-3	Disabled
Loss of Signal Threshold Select	RXLOSTHRESHOLD	0-7	0

Figure 51. PCS Setup Tab

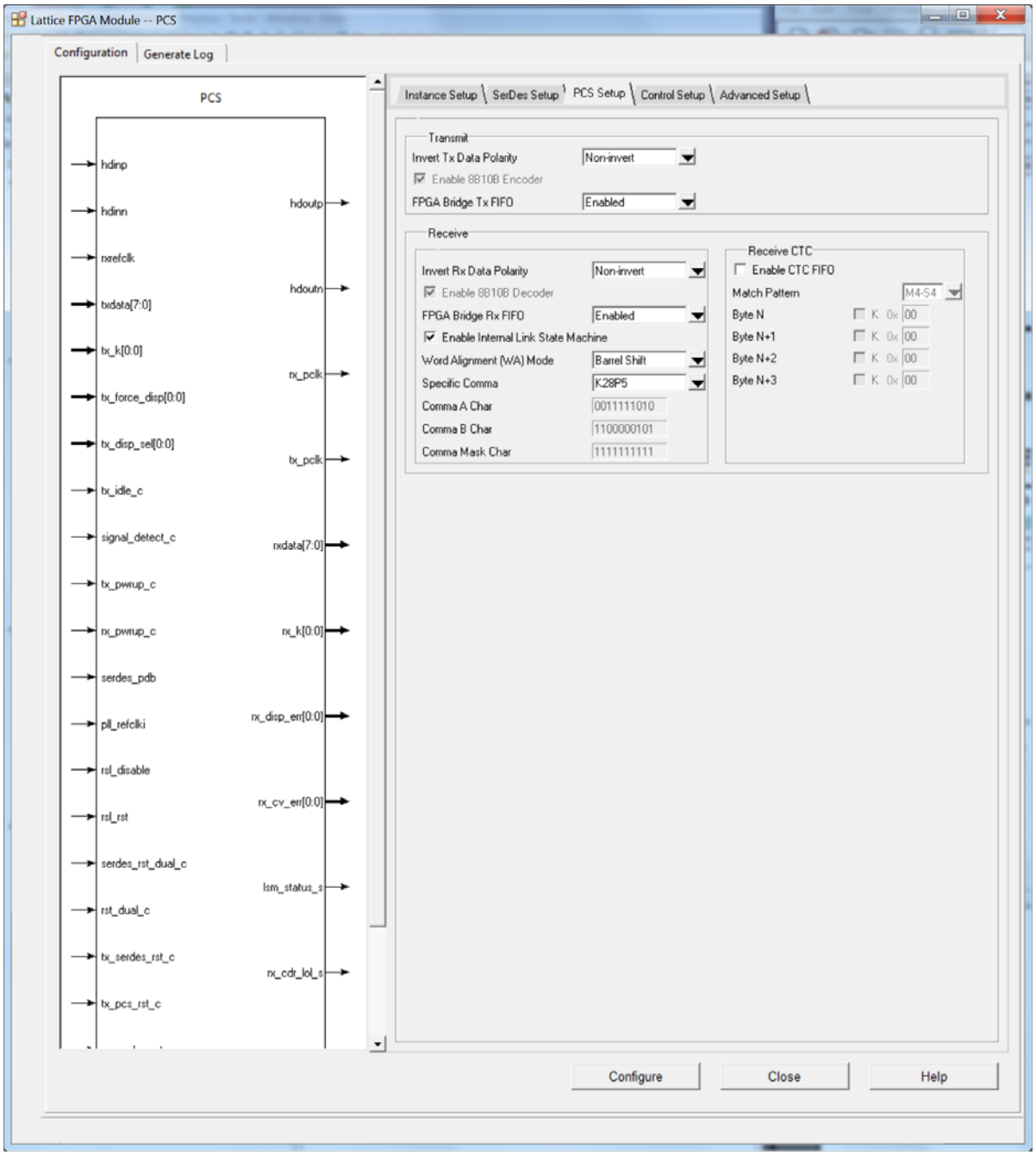


Table 27. PCS Setup Tab Description

GUI Text	Attribute Names	Values	Default Value
Invert Tx Data Polarity	TXINVPOL	Non-invert, Invert	Non-invert
Enable 8B10B Encoding	TX8B10B	Enabled, Disabled	Enabled
FPGA Bridge Tx FIFO	TXFIFO_ENABLE	Enabled, Disabled	Enabled
Invert Rx Data Polarity	RXINVPOL	Non-invert, Invert, User Port	Non-invert
Enable 8B10B Decoder	RX8B10B	Enabled, Disabled	Enabled
FPGA Bridge Rx FIFO	RXFIFO_ENABLE	Enabled, Disabled	Enabled
Enable Internal Link State Machine	RXLSM	Enabled, Disabled	Disabled
Word Alignment (WA) Mode	RXWA	Disabled, Barrel Shift, Bit Slip	Disabled
Specific Comma	RXSC	User Defined, K28P5, K28P157	User Defined
Comma A Character	RXCOMMAA	10-bit Binary	1100000101
Comma B Character	RXCOMMAB	10-bit Binary	0011111010
Comma Mask	RXCOMMAM	10-bit Binary	1111111111
Enable CTC FIFO	RXCTC	Enabled, Disabled	Disabled
Match Pattern	RXCTCMATCHPATTERN	M1-S1, M2-S2, M4-S4, M4-S1	M1-S1
Byte N (Kchar, Hex)	RXCTCBYTEN	K, 8-bit Hex	0 00H
Byte N+1 (Kchar, Hex)	RXCTCBYTEN1	K, 8-bit Hex	0 00H
Byte N+2 (Kchar, Hex)	RXCTCBYTEN2	K, 8-bit Hex	0 00H
Byte N+3 (Kchar, Hex)	RXCTCBYTEN3	K, 8-bit Hex	0 00H
Rx Multi-Channel Alignment Enable	RXMCAENABLE	Disabled, Enabled	Disabled
Alignment Character A	ACHARA	K, 8-bit Hex	0 00H
Alignment Character B	ACHARB	K, 8-bit Hex	0 00H
Alignment Character Mask	ACHARM	K, 8-bit Hex	0 00H

Figure 52. Control Setup Tab

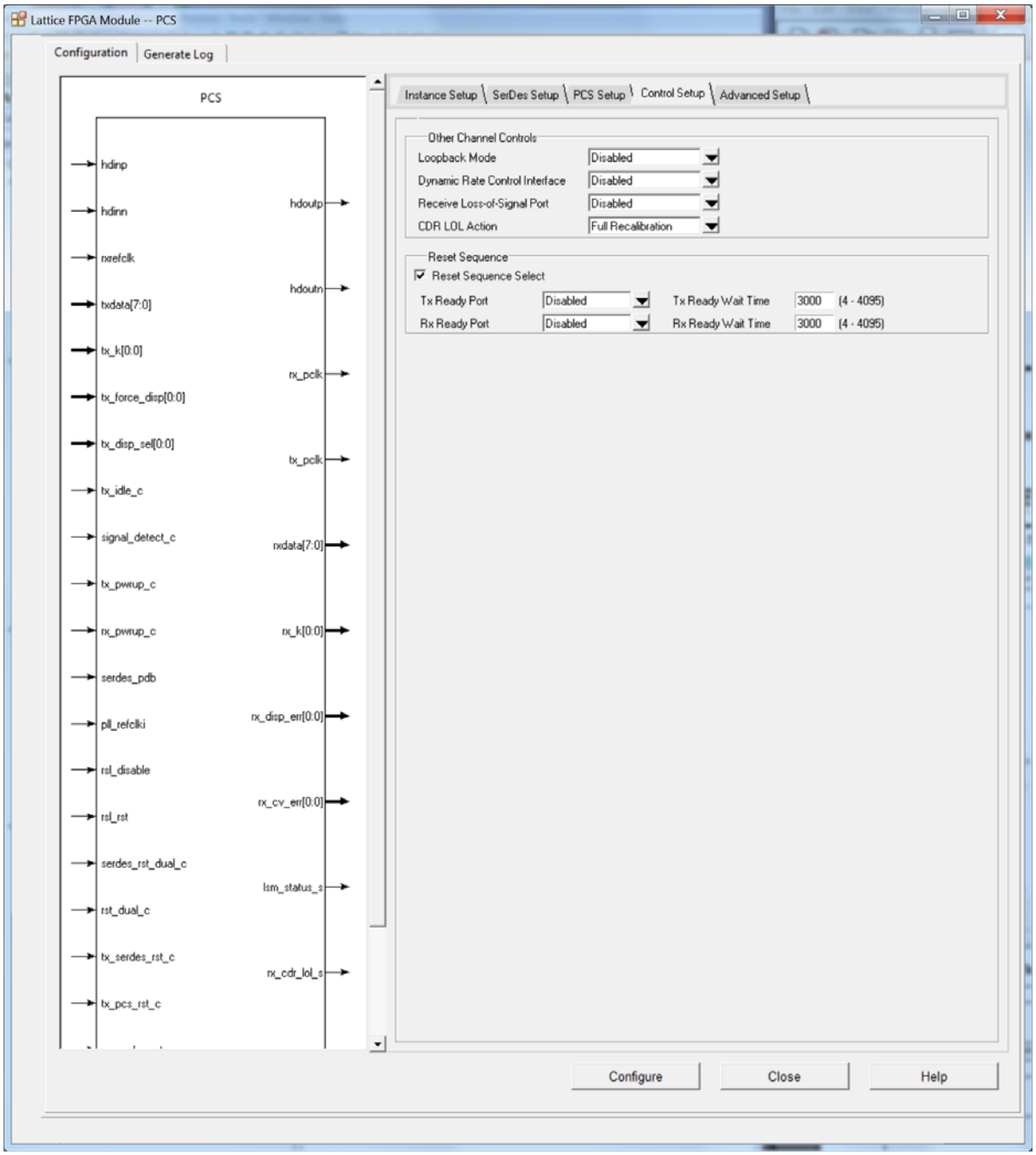


Table 28. Control Setup Tab Description

GUI Text	Attribute Names	Values	Default Value
Loopback Mode	LOOPBACK	Disabled, Rx EQ to Tx, Rx Paralle to Tx, Tx Output to Rx	Disabled
Dynamic Rate Control Interface	RCSRC	Disabled, Eanbled	Disabled
Receive Loss-of-Signal Port	LOSPORT	Disabled, Enabled	Disabled
CDR LOL Action	CDRLOLACTION	“Nothing", "Full Recalibratoin", "Simple Recalbration"	Full Recalibration
Reset Sequence Select	RSTSEQSEL	None, "Tx Before Rx", "Tx/Rx Independent"	Tx Before Rx

Figure 53. Advanced Setup Tab

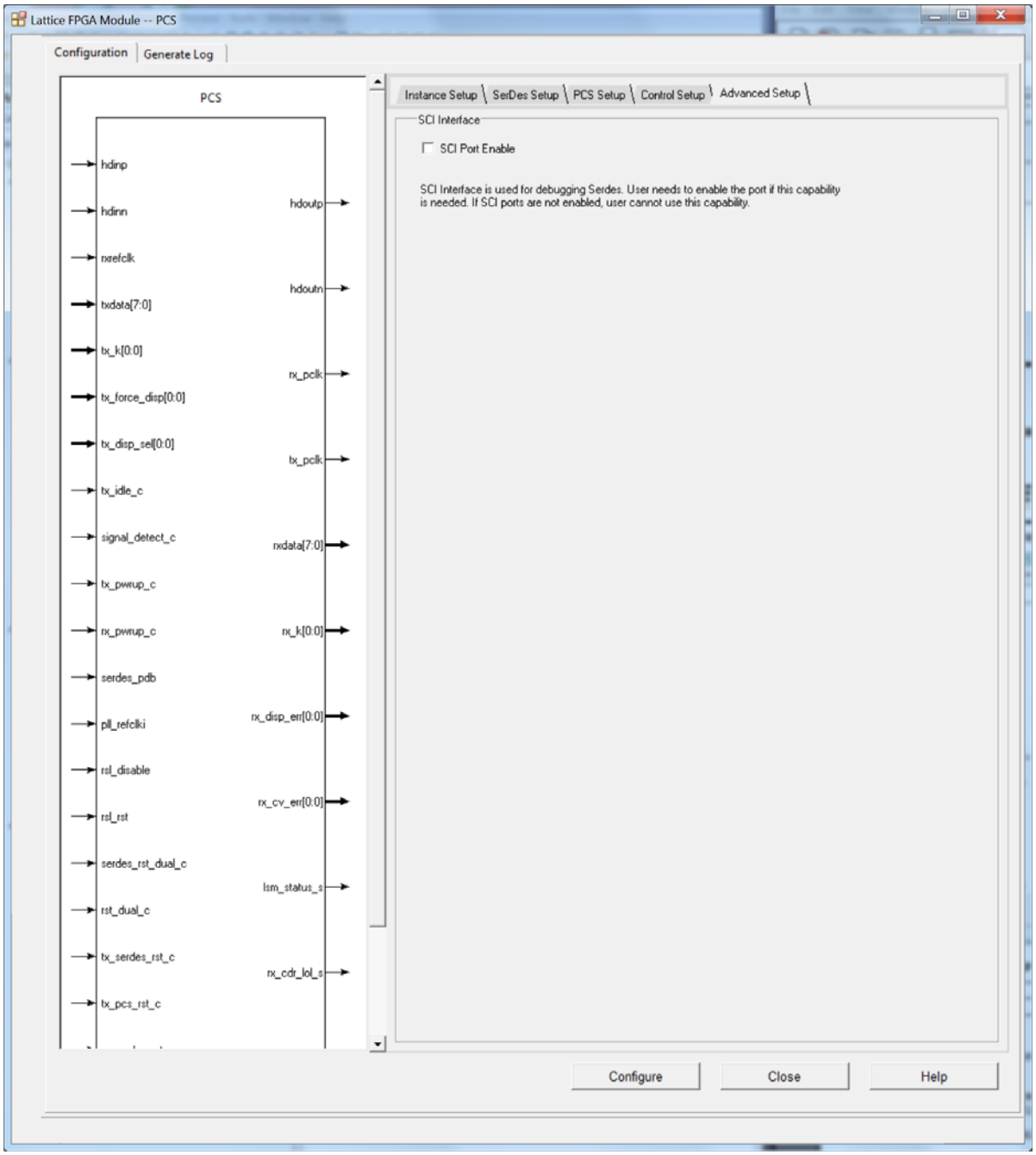


Table 29. Advanced Setup Tab Description

GUI Text	Attribute Names	Values	Default Value
SCI Port Enable	SCI	Enabled, Disabled	Disabled

EXTREF Block

The EXTREF is the reference clock input buffer primitive for the dedicated external clock inputs to the SerDes TxPLL.

The EXTREF primitive allows the users to select how the reference clock input buffer is configured. There is one EXTREF block in each DCU.

Figure 54. EXTREF Block Diagram

The EXTREF module takes the differential external reference clock pins (refclkp/n) and sends out a single-ended clock signal to the SerDes TxPLL. The EXTREF can only connect to SerDes TxPLL in the same DCU, or to SerDes TxPLL in the complementary sharing DCU.

EXTREF I/O Port Description

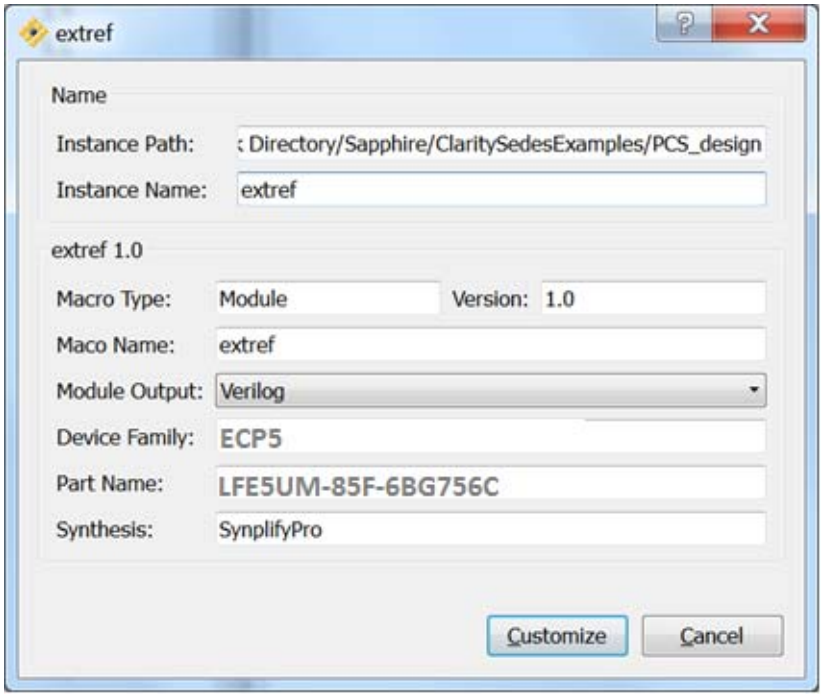
refclkp – External reference clock positive input port

refclkn – External reference clock negative input port

refclko - Single-ended clock signal to be connected to PCS PLL inputs

When the REFCLK module is selected, the EXTREF instance table will be brought up

Figure 55. EXTREF Module In Clarity Designer



The GUI tab and description of the External RERCLK primitive is show below:

Figure 56. EXTREF Tab

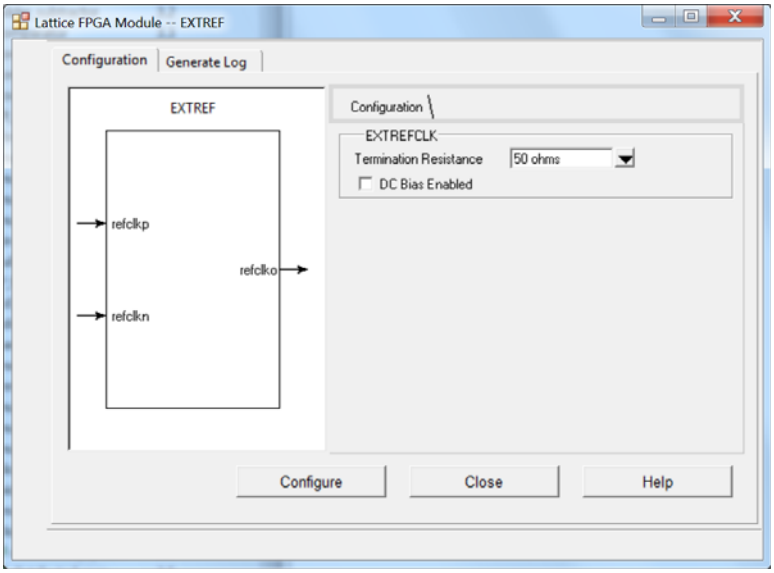


Table 30. EXTREF Tab Description

GUI Text	Attribute Names	Values	Default Value
Termination Resistance	EXTREFTERMRES	50 Ohms, Hi-Z	50 Ohms
DC Bias Enabled	EXTREFDCBIAS	Disabled, Enabled	Disabled

References

- TN1033, High-Speed PCB Design Considerations
- TN1114, Electrical Recommendations for Lattice SERDES
- DS1044, ECP5 and ECP5-5G Family Data Sheet
- HB1009, LatticeECP3 Family Handbook
- DS1021, LatticeECP3 Family Data Sheet
- TN1176, LatticeECP3 SERDES/PCS Usage Guide
- Clarity Designer Manual
- ECO Editor Manual

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Date	Version	Change Summary
November 2015	1.1	Added support for ECP5-5G.
		Changed document title to ECP5 and ECP5-5G SERDES/PCS Usage Guide.
		Updated Introduction section. Revised information on ECP5 architecture and PCS logic support.
		Updated Features section. Revised details of Up to Four Channels of High-Speed SERDES with Increased Granularity feature.
		Updated New Features Over LatticeECP3 SERDES/PCS section. Added Transmit/Receive SERDES to eDP SERDES interface support.
		Added Difference Between ECP5 and ECP5-5G SERDES/PCS section.
		Updated Standards Supported section. Revised Table 1, Standards Supported by the SERDES/PCS. — Added PCI Express 2.0 standard. — Added E.48.LV under CPRI-E.6.LV — Changed SD-SDI (259M, 344M) and JESD204A/B values. — Added eDP-RBR and HBR. — Removed Serial RapidIO (SR/LR) standard. — Added footnote.
		Updated Multi-Protocol Design Consideration section. Revised Table 3, LFE5UM/LFE5UM5G Mixed Protocol Pair. — Changed SRIO to SGMII. — Removed SRIO 1.25G/2.5G. — Added protocols,
		Updated Receive Data Bus section. Revised Table 4, Data Bus Usage by Mode. Removed SRIO column.
		Updated Tx Differential Output Driver section. Removed 1.2 V indicated as VCCHTX.
		Updated Generic 8b10b Mode section. Revised Generic 8b10b applications supported by LFE5UM SERDES/PCS block.
		Updated section heading to PCI Express Revision 1.1 and 2.0.
		Updated PCI Express Termination section. — Revised information on differential signaling pairs. — Revised information on PCI Express L2 State. — Updated Figure 15, PCI Express Interface Diagram.
		Removed Serial RapidIO (SRIO) Mode section.
		Updated Common Public Radio Interface (CPRI) section. — Revised four line bit rate options allowed by CPRI. Added descriptive contents. — Revised information on link layer requirements. Removed OBSAI specifications.
		Updated SDI (SMPTE) Mode section. Revised HD-SDI (SMPTE292M) and 3G-SDI (SMPTE424M) information. Added descriptive contents.
		Added Embedded Display Port (eDP) section.
		Updated 2:1 Gearing section. Revised Table 15, Decision Matrix for Six Interface Cases. — Changed SERDES Line Rate to Above 2.5 Gbps. — Added footnote 7.
		Updated Reset Sequence section. Added information on Reset Sequence Logic (RSL).
		Updated Clarity Designer (System Builder/Planner) Overview section. Modified Differential Amplitude value in Table 23, SERDES Setup Tab Description.

Date	Version	Change Summary
		Updated SERDES/PCS Generation in Clarity Designer (System Builder/Planner) section. — Removed SRIO and added eDP in the list of protocols selected by users. — Updated Table 25, Instance Setup Tab Description. Updated values for TX Max Data Rate, PLL Multiplier, Tx Line Rate, Receive Max Data Rate (CDR), Rx Line Rate. Added footnote. — Updated Table 26, SERDES Setup Tab Description. Updated default value for Differential Amplitude.
		Updated Technical Support Assistance section.
March 2014	1.0	Initial release.

Appendix A. Configuration Registers

There are specific dual-level registers and channel-level registers. In each category, there are SerDes specific registers and PCS specific registers. Within these sub-categories there are:

- Control registers,
- Status registers,
- Status registers with clear-on-read
- Interrupt Control registers,
- Interrupt Status registers,
- Interrupt Source registers (clear-on-read)

The following abbreviations are used to indicate what type of register access for each is supported:

- R/W = Read/Write
- RO = Read Only

Dual Registers Overview

Table 31. Dual Interface Registers Map

BA = Base Address (Hex)

BA	Register name	D7	D6	D5	D4	D3	D2	D1	D0
PER DUAL PCS CONTROL REGISTERS (10)									
00	DL_00	reg_sync_toggle	force_int	char_mode	xge_mode	Spare	Spare	Spare	Spare
01	DL_01	Internal use only							
02	DL_02	high_mark[3]	high_mark[2]	high_mark[1]	high_mark[0]	low_mark[3]	low_mark[2]	low_mark[1]	low_mark[0]
03	DL_03	Reserved	Reserved	Reserved	Reserved	Reserved	pfifo_clr_sel	Internal use only	Internal use only
04	DL_04	Internal use only							
05	DL_05	Internal use only							
06	DL_06	Internal use only							
07	DL_07	Internal use only							
08	DL_08	Internal use only							
09	DL_09	Internal use only	Internal use only	ls_sync_status_1_int_ctl	ls_sync_status_0_int_ctl	Reserved	Reserved	ls_sync_statusn_1_int_ctl	ls_sync_statusn_0_int_ctl
PER DUAL SERDES CONTROL REGISTERS (6)									
0A	DL_0A	Internal use only	Reserved	tx_refck_sel	refck_dcbias_en	refck_rterm	Reserved	refck_out_sel[1]	Internal use only
0B	DL_0B	refck25x	bus8bit_sel	Reserved	refck_from_nd_sel[1]	refck_from_nd_sel[0]	refck_to_nd_en	refck_mode[1]	refck_mode[0]
0C	DL_0C	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	cdr_lol_set[1]	cdr_lol_set[0]
0D	DL_0D	rg_set[1]	rg_set[0]	rg_en	pll_lol_set[1]	pll_lol_set[0]	Internal use only	Internal use only	Internal use only
0E	DL_0E	Internal use only							
0F	DL_0F	plol_int_ctl	~plol_int_ctl	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
PER DUAL CLOCK RESET REGISTERS (2)									
10	DL_10	Reserved	Reserved	txpll_pwdnb	refck_pwdnb	macropdb	macro_rst	dual_rst	trst
11	DL_11	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
12	DL_12	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
13	DL_13	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
PER DUAL SERDES CONFIGURATION REGISTERS									
15	DL_15	Internal use only							
16	DL_16	Internal use only							
17	DL_17	Internal use only							
18	DL_18	Internal use only							
19	DL_19	Internal use only							
1A	DL_1A	Internal use only							
1B	DL_1B	Internal use only							
1C	DL_1C	Internal use only							
1D	DL_1D	Internal use only							
PER DUAL PCS STATUS REGISTERS (5)									
20	DL_20	Reserved	Reserved		int_dua_out	Reserved	Reserved	int_cha_out[1]	int_cha_out[0]

21	DL_21	Reserved	Reserved	ls_sync_status_1	ls_sync_status_0	Reserved	Reserved	ls_sync_statusn_1	ls_sync_statusn_0
22	DL_22	Reserved	Reserved	ls_sync_status_1_int	ls_sync_status_0_int	Reserved	Reserved	ls_sync_statusn_1_int	ls_sync_statusn_0_int
23	DL_23	Internal use only							
24	DL_24	Internal use only							
PER DUAL SERDES STATUS REGISTERS (4)									
25	DL_25	plol	~plol	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
26	DL_26	plol_int	~plol_int	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
27	DL_27	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
28	DL_28	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Per Dual PCS Control Register Details

Table 32. PCS Control Register 1 (DL_00)

Bit	Name	Description	Type	Default
3:0	Reserved		R/W	0
4	xge_mode	1 = Selects 10Gb Ethernet (selects a different 10G XAUI LSM and some slight differences in 8B10B encoding behavior relative to GE mode). 0 = Depends on Channel Mode Selection	R/W	
5	char_mode	1 = Enable SerDes characterization mode 0 = Disable SerDes characterization mode	R/W	0
6	force_int	1 = Force to generate interrupt signal 0 = Normal operation	R/W	0
7	reg_sync_toggle	Transition = Reset the four Tx Serializers to minimize Tx Lane-to-Lane Skew Level = Normal operation of Tx Serializers	R/W	0

Table 33. PCS Control Register 3 (DL_02)

Bit	Name	Description	Type	Default
3:0	low_mark [3:0]	Clock compensation FIFO low water mark. Mean is 4'b1000	R/W	4'b0101
7:4	high_mark [3:0]	Clock compensation FIFO high water mark. Mean is 4'b1000	R/W	4'b1011

Table 34. PCS Control Register 4 (DL_03)

Bit	Name	Description	Type	Default
0		Internal use only		
1		Internal use only		
2	pfifo_clr_sel	1 = pfifo_clr signal or channel register bit clears the FIFO 0 = pfifo_error internal signal self clears the FIFO	R/W	0
7:3	Reserved		R/W	0

Table 35. PCS Interrupt Control Register 10 (DL_09)

Bit	Name	Description	Type	Default
0	ls_sync_statusn_0_int_ctl	1 = Enable interrupt for ls_sync_status_0 when it goes low (out of sync) 0 = Disable interrupt for ls_sync_status_0 when it goes low (out of sync)	R/W	0
1	ls_sync_statusn_1_int_ctl	1 = Enable interrupt for ls_sync_status_1 when it goes low (out of sync) 0 = Disable interrupt for ls_sync_status_1 when it goes low (out of sync)	R/W	0
3:2	Reserved			
4	ls_sync_status_0_int_ctl	1 = Enable interrupt for ls_sync_status_0 (in sync) 0 = Disable interrupt for ls_sync_status_0 (in sync)	R/W	0
5	ls_sync_status_1_int_ctl	1 = Enable interrupt for ls_sync_status_1 (in sync) 0 = Disable interrupt for ls_sync_status_1 (in sync)	R/W	0
7:6	Internal use only			

Per Dual SerDes Control Register Details

Table 36. SERDES Control Register 1 (DL_0A)

Bit	Name	Description	Type	Default
1:0	REFCK_OUT_SEL [1:0]	Refck outputs enable/disable control[0]: 0 = rx_refck_local output disable 1 = rx_refck_local output enable	R/W	2'b00
		[1]: 0 = refck2core output disable 1 = refck2core output enable		
2	Reserved		R/W	0
3	REFCK_RTERM	Termination at reference clock input buffer 1 = 100 ohm 0 = High Impedance (default)	R/W	
4	REFCK_DCBIAS_EN	1 = Reference clock internal dc bias enabled	R/W	
		0		
5	TX_REFCK_SEL	TxPLL reference clock select		
		1 = ck_core_tx 0 = rx_refck_local	R/W	0
6	Reserved		R/W	0
7	Internal use only			

Table 37. SERDES Control Register 2 (DL_0B)

Bit	Name	Description	Type	Default

1:0	REFCK_MODE[1:0]	If REFCK25X=0, then
		00 = Internal high speed bit clock is 20x
		01 = Internal high speed bit clock is 10x
		10 = Internal high speed bit clock is 16x
		11 = Internal high speed bit clock is 8x
		If REFCK25X=1, then
		xx = Internal high speed bit clock is 25x
			R/W	2'b00

2	REFCK_TO_ND_EN	1 = Enable REFCLK to Neighboring Dual	R/W	0
3	REFCK_FROM_ND_SEL[0]	1 = Select TX REFCLK from Neighboring Dual	R/W	0
4	REFCK_FROM_ND_SEL[1]	1 = Select RX REFCLK from Neighboring Dual	R/W	0
5	Reserved			
6	BUS8BIT_SEL	1 = Select 8-bit bus width 0 = Select 10-bit bus width (default)	R/W	0
7	REFCK25X	1 = Internal high speed bit clock is 25x (for REFCLK = 100 MHz only) 0 = see REFCK_MODE	R/W	0

Table 38. SERDES Control Register 3 - (DL_0C)

Bit	Name	Description	R/W	Default
1:0	CDR_LOL_SET [1:0]	CDR loss of lock setting: Lock / Unlock: 00 = +/- 1000ppm x2 / = +/- 1500ppm x2 01 = +/- 2000ppm x2 / = +/- 2500ppm x2 10 = +/- 4000ppm / = +/- 7000ppm 11 = +/- 300ppm / = +/- 450ppm	R/W	2'b00
7:2	Reserved			

Table 39. SERDES Control Register 4 (DL_0D)

Bit	Name	Description		R/W	Default
2:0	TX_VCO_CK_DIV [2:0]	VCO output frequency select: 00x = Divided by 1 / 01x = Divided by 2 100 = Divided by 4 / 101 = Divided by 8 110 = Divided by 16 / 111 = Divided by 32		R/W	3'b000
4:3	PLL_LOL_SET [1:0]	TxPLL loss of lock setting: Lock / Unlock: 00 = +/- 300 ppm x2 / = +/- 600 ppm x2 01 = +/- 300 ppm / = +/- 2000 ppm 10 = +/- 1500 ppm / = +/- 2200 ppm 11 = +/- 4000 ppm / = +/- 6000 ppm		R/W	2'b00
5	Internal use only				
7:6	Internal use only				

Table 40. SERDES Interrupt Control Register 6 (DL_0F)

Bit	Name	Description	R/W	Default
5:0	Reserved			
6	~PLOL_INT_CTL	1 = Interrupt enabled for obtaining lock on PLOL. 0 = Interrupt not enabled for obtaining lock PLOL.	R/W	0
7	PLOL_INT_CTL	1 = Interrupt enabled for loss of lock on PLOL. 0 = Interrupt not enabled for loss of lock PLOL.	R/W	0

Per Dual Reset and Clock Control Register Details

Table 41. Reset and Clock Control Register 1 (DL_10)

Bit	Name	Description	R/W	Default
0	trst	1 = Reset TxPLL Loss of Lock	R/W	0
1	dual_rst	1 = Assert dual reset	R/W	0
2	macro_rst	1 = Assert macro reset	R/W	0
3	macropdb	0 = Assert power down	R/W	1
4	refck_pwdnb	Reference clock power down control		
		0 = Power down 1 = Power up	R/W	0
5	txpll_pwdnb	TXPLL power down control		
		0 = Power down 1 = Power up	R/W	0
6	Reserved			
7	Reserved			

Table 42. Reset and Clock Control Register 2 (DL_11)

Bit	Name	Description	R/W	Default
7:0	Reserved			

Table 43. Serdes Control Register 6 (DL_12)

Bit	Name	Description	R/W	Default
7:0	Reserved			

Table 44. Serdes Control Register 7 (DL_13)

Bit	Name	Description	R/W	Default
7:0	Reserved			

Per Dual PCS Status Register Details

Table 45. PCS Status Register 1 (DL_20)

Bit	Name	Description	R/W	Int
1:0	int_cha_out [1:0]	Per Channel Interrupt status	RO	N
3:2	Reserved			
4	int_dua_out	Per Dual Interrupt status	RO	N
5	Spare	Delayed global resetn from tri_ion	RO	N
7:6	Reserved			

Table 46. PCS Status Register 2 (DL_21)

Bit	Name	Description	R/W	Int
0	ls_sync_statusn_0	1 = Alarm generated on sync_status_0 when it goes low (out of sync) 0 = Alarm not generated on sync_status_0 when it goes low (out of sync)	RO	Y
1	ls_sync_statusn_1	1 = Alarm generated on sync_status_1 when it goes low (out of sync) 0 = Alarm not generated on sync_status_1 when it goes low (out of sync)	RO	Y
3:2	Reserved			
4	ls_sync_status_0	1 = Alarm generated on sync_status_0. 0 = Alarm not generated on sync_status_0.	RO	Y
5	ls_sync_status_1	1 = Alarm generated on sync_status_1. 0 = Alarm not generated on sync_status_1.	RO	Y
7:6	Reserved			

Table 47. PCS Packet Interrupt Status Register 3 (DL_22)

Bit	Name	Description	R/W	Int
0	ls_sync_statusn_0_int	1 = Interrupt generated on sync_status_0 when it goes low (out of sync) 0 = Interrupt not generated on sync_status_0 when it goes low (out of sync)	RO CR	Y
1	ls_sync_statusn_1_int	1 = Interrupt generated on sync_status_1 when it goes low (out of sync) 0 = Interrupt not generated on sync_status_1 when it goes low (out of sync)	RO CR	Y
3:2	Reserved			
4	ls_sync_status_0_int	1 = Interrupt generated on sync_status_3 (in sync) 0 = Interrupt not generated on sync_status_3 (in sync)	RO CR	Y
5	ls_sync_status_1_int	1 = Interrupt generated on sync_status_2 (in sync) 0 = Interrupt not generated on sync_status_2 (in sync)	RO CR	Y
7:6	Reserved			

Per Dual SerDes Status Register Details

Table 48. SERDES Status Register 1 (DL_25)

Bit	Name	Description	R/W	Int
5:0	Reserved			
6	PLOL_STSB	1 = PLL lock obtained	RO	Y
7	PLOL_STS	1 = PLL Loss of lock	RO	Y

Table 49. SERDES Interrupt Status Register 2 (DL_26)

Bit	Name	Description	R/W	Int
5:0	Reserved			
6	~PLOL_INT	1 = Interrupt generated on ~PLOL. 0 = Interrupt not generated ~PLOL.	RO CR	Y
7	PLOL_INT	1 = Interrupt generated on PLOL. 0 = Interrupt not generated PLOL.	RO CR	Y

Per Channel Register Overview

Table 50. Channel Interface Registers Map

BA = Base Address (Hex)

BA	Register name	D7	D6	D5	D4	D3	D2	D1	D0
PER CHANNEL GERNERAL REGISTERS (16)									
00	CH_00	Spare	Spare	Spare	wa_mode	rio_mode	pcie_mode	Spare	uc_mode
01	CH_01	enable_cg_align	prbs_enable	Internal use only	ge_an_enable	Spare	Internal use only	invert_tx	invert_rx
02	CH_02	pfifo_clr	pcie_ei_en	pcs_det_time_sel[1]	pcs_det_time_sel[0]	rx_gear_mode	tx_gear_mode	Reserved	Reserved
03	CH_03	Spare	sb_bypass	Internal use only	sb_bist_sel	Internal use only	sel_bist_txd4enc	tx_gear_bypass	fb_loopback
04	CH_04	lsm_disable	signal_detect	rx_gear_bypass	ctc_bypass	dec_bypass	wa_bypass	rx_sb_bypass	sb_loopback
05	CH_05	min_ipg_cnt[1]	min_ipg_cnt[0]	match_4_enable	match_2_enable	Spare	Spare	Spare	Spare
06	CH_06	cc_match_1[7]	cc_match_1[6]	cc_match_1[5]	cc_match_1[4]	cc_match_1[3]	cc_match_1[2]	cc_match_1[1]	cc_match_1[0]
07	CH_07	cc_match_2[7]	cc_match_2[6]	cc_match_2[5]	cc_match_2[4]	cc_match_2[3]	cc_match_2[2]	cc_match_2[1]	cc_match_2[0]
08	CH_08	cc_match_3[7]	cc_match_3[6]	cc_match_3[5]	cc_match_3[4]	cc_match_3[3]	cc_match_3[2]	cc_match_3[1]	cc_match_3[0]
09	CH_09	cc_match_4[7]	cc_match_4[6]	cc_match_4[5]	cc_match_4[4]	cc_match_4[3]	cc_match_4[2]	cc_match_4[1]	cc_match_4[0]
0A	CH_0A	cc_match_4[9]	cc_match_4[8]	cc_match_3[9]	cc_match_3[8]	cc_match_2[9]	cc_match_2[8]	cc_match_1[9]	cc_match_1[8]
0B	CH_0B	udf_comma_mask[7]	udf_comma_mask[6]	udf_comma_mask[5]	udf_comma_mask[4]	udf_comma_mask[3]	udf_comma_mask[2]	udf_comma_mask[1]	udf_comma_mask[0]
0C	CH_0C	udf_comma_a[7]	udf_comma_a[6]	udf_comma_a[5]	udf_comma_a[4]	udf_comma_a[3]	udf_comma_a[2]	udf_comma_a[1]	udf_comma_a[0]
0D	CH_0D	udf_comma_b[7]	udf_comma_b[6]	udf_comma_b[5]	udf_comma_b[4]	udf_comma_b[3]	udf_comma_b[2]	udf_comma_b[1]	udf_comma_b[0]
0E	CH_0E	udf_comma_a[9]	udf_comma_a[8]	udf_comma_b[9]	udf_comma_b[8]	udf_comma_mask[9]	udf_comma_mask[8]		
0F	CH_0F	Reserved	Reserved	Reserved	Reserved	cc_underrun_int_ctl	cc_overrun_int_ctl	fb_rx_fifo_error_int_ctl	fb_tx_fifo_error_int_ctl
PER CHANNEL SERDES CONTROL REGISTERS (15)									
10	CH_10	tx_post_sign	tx_pre_sign	tdrv_post_en	tdrv_pre_en	ldr_core2tx_sel	tx_div11_sel	rate_mode_tx	tpwdnb
11	CH_11	Spare	tx_cm_sel[1]	tx_cm_sel[0]	rterm_tx[4]	rterm_tx[3]	rterm_tx[2]	rterm_tx[1]	rterm_tx[0]
12	CH_12	tdrv_slice3_sel[1]	tdrv_slice3_sel[0]	tdrv_slice2_sel[1]	tdrv_slice2_sel[0]	tdrv_slice1_sel[1]	tdrv_slice1_sel[0]	tdrv_slice0_sel[1]	tdrv_slice0_sel[0]
13	CH_13	tdrv_slice4_cur[1]	tdrv_slice4_cur[0]	tdrv_slice3_cur[1]	tdrv_slice3_cur[0]	tdrv_slice5_sel[1]	tdrv_slice5_sel[0]	tdrv_slice4_sel[1]	tdrv_slice4_sel[0]
14	CH_14	tdrv_slice2_cur[1]	tdrv_slice2_cur[0]	tdrv_slice1_cur[2]	tdrv_slice1_cur[1]	tdrv_slice1_cur[0]	tdrv_slice0_cur[2]	tdrv_slice0_cur[1]	tdrv_slice0_cur[0]
15	CH_15	tdrv_slice5_cur[1]	tdrv_slice5_cur[0]	tdrv_dat_sel[1]	tdrv_dat_sel[0]	lb_ctl[3]	lb_ctl[2]	lb_ctl[1]	lb_ctl[0]
16	CH_16	rxterm_cm[1]	rxterm_cm[0]	rcv_dcc_en	rx_refck_sel	ldr_rx2core_sel	rx_div11_sel	rate_mode_rx	rpwdnb
17	CH_17	rxin_cm[1]	rxin_cm[0]]	Reserved	rterm_rx[4]	rterm_rx[3]	rterm_rx[2]	rterm_rx[1]	rterm_rx[0]
18	CH_18	fc2dco_dloop	fc2dco_floop	Reserved	Spare	Reserved	Reserved	Reserved	Reserved
19	CH_19	Spare	req_lvl_set[1]	req_lvl_set[0]	rx_rate_sel[3]	rx_rate_sel[2]	rx_rate_sel[1]	rx_rate_sel[0]	req_en
1A	CH_1A	band_calib_mode	en_recalib	dco_calib_rst	dco_facq_rst	pden_sel	rx_dco_ck_div[2]	rx_dco_ck_div[1]	rx_dco_ck_div[0]
1B	CH_1B	rlos_sel	rx_los_en	rx_los_hyst_en	rx_los_ceq[1]	rx_los_ceq[0]	rx_los_lvl[2]	rx_los_lvl[1]	rx_los_lvl[0]
1C	CH_1C	Spare	Spare	Spare	Spare	Spare	Spare	Spare	Spare
1D	CH_1D	Spare	Spare	Spare	Spare	Spare	Spare	Spare	Spare
1E	CH_1E		pci_det_done_int_ctl	rlos_int_ctl	~rlos_int_ctl	Reserved	Reserved	rlol_int_ctl	~rlol_int_ctl
1F	CH_1F	rrst	lane_rx_rst	lane_tx_rst	ff_tx_f_clk_dis	ff_tx_h_clk_en	ff_rx_f_clk_dis	ff_rx_h_clk_en	sel_sd_rx_clk
20	CH_20	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
PER CHANNEL PCS STATUS REGISTERS (6)									
30	CH_30				pfifo_error	cc_underrun	cc_overrun	fb_rx_fifo_error	fb_tx_fifo_error
31	CH_31	prbs_error_cnt[7]	prbs_error_cnt[6]	prbs_error_count[5]	prbs_error_cnt[4]	prbs_error_cnt[3]	prbs_error_cnt[2]	prbs_error_cnt[1]	prbs_error_cnt[0]
32	CH_32					wa_offset[3]	wa_offset[2]	wa_offset[1]	wa_offset[0]
33	CH_33	Reserved	Reserved	Reserved	Reserved	cc_underrun_int	cc_overrun_int	fb_rx_fifo_error_int	fb_tx_fifo_error_int
34	CH_34	Reserved	ffs_ls_sync_status	fb_rxrst_o	fb_txrst_o	Reserved	Reserved	cc_re_o	cc_we_o
35	CH_35	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
PER CHANNEL SERDES STATUS REGISTERS (7)									
36	CH_36		pci_det_done	rlos	~rlos			rlol	~rlol
37	CH_37	dco_calib_err	dco_calib_done	dco_facq_err	dco_facq_done				pci_connect
38	CH_38	Reserved							
39	CH_39	Reserved							
3a	CH_3A		pci_det_done_int	rlos_int	~rlos_int			rlol_int	~rlol_int
3b	CH_3B	Reserved							
3c	CH_3C	Reserved							

Per Channel PCS Register Details

Table 51. PCS Control Register 1 (CH_00)

Bit	Name	Description	R/W	Default
0	uc_mode	1 = Selects User Configured mode 0 = Selects other mode (PCIe, RapidIO, 10GbE, 1GbE)	R/W	0
1	Reserved			
2	pcie_mode	1 = PCI-Express mode of operation 0 = Selects other mode (RapidIO, 10GbE, 1GbE)	R/W	0
3	rio_mode	1 = Selects Rapid-IO mode 0 = Selects other mode (10GbE, 1GbE)	R/W	0
4	wa_mode	1 = bitslip word alignment mode 0 = barrel shift word alignment mode	R/W	0
7:5	Reserved		R/W	0

Table 52. PCS Control Register 2 (CH_01)

Bit	Name	Description	R/W	Default
0	invert_rx	1 = Invert received data 0 = Do not invert received data.	R/W	0
1	invert_tx	1 = Invert transmitted data 0 = Do not invert transmitted data.	R/W	0
2	Internal use only			
3	Reserved			
4	ge_an_enable	1 = Enable GigE Auto Negotiation 0 = Disable GigE Auto Negotiation	R/W	0
5	Internal use only			
6	Internal use only			
7	enable_cg_align	Only valid when operating in uc_mode 1 = Enable continuous comma alignment 0 = Disable continuous comma alignment	R/W	0

Table 53. PCS Control Register 3 (CH_02)

Bit	Name	Description	R/W	Default
0	tx_ch	1 = Transmit PCS inputs are sourced from test characterization ports. The test characterization mode should be enabled.	R/W	0
1:0	Reserved			
2	tx_gear_mode	1 = Enable 2:1 gearing for transmit path on selected channels 0 = Disable 2:1 gearing for transmit path on selected channels (no gearing)	R/W	0
3	rx_gear_mode	1 = Enable 2:1 gearing for receive path on selected channels 0 = Disable 2:1 gearing for receive path on selected channels (no gearing)	R/W	0
5:4	pcs_det_time_sel[1:0]	PCS connection detection time 11 = 16 us, 10 = 4 us, 01 = 2 us 00 = 8 us	R/W	0
6	pcie_ei_en	1 = PCI Express Electrical Idle 0 = Normal operation	R/W	0
7	pfifo_clr	1 = Clears PFIFO if dual register bit pfifo_clr_sel is set to 1. This signal is Ored with interface signal pfifo_clr. 0 = Normal operation	R/W	0

Table 54. PCS Control Register 4 (CH_03)

Bit	Name	Description	R/W	Default
0	fb_loopback	1 = Enable loopback in the PCS just before FPGA bridge from RX to TX. 0 = Normal data operation.	R/W	0
1	tx_gear_bypass	1 = Bypass PCS Tx gear box 0 = Normal operation	R/W	0
2	Internal use only			
3	enc_bypass	1 = Bypass 8b10b encoder 0 = Normal operation	R/W	0
4	Internal use only			
5	sb_pfifo_lp	1 = Enable Parallel Loopback from Rx to Tx via parallel fifo 0 = Normal data operation	R/W	0
6	sb_bypass	1 = Invert TX data after SerDes Bridge 0 = Do not invert TX data after SerDes Bridge (Note: Loopback data is inverted)	R/W	0
7	Reserved			

Table 55. PCS Control Register 5 (CH_04)

Bit	Name	Description	R/W	Default
0	sb_loopback	1 = Enable loopback in the PCS from Tx to Rx in SerDes bridge. 0 = Normal data operation.	R/W	0
1	rx_sb_bypass	1 = Invert RX data after SerDesBridge 0 = Normal operation	R/W	0
2	wa_bypass	1 = Bypass word alignment 0 = Do not invert RX data after SerDesBridge(Note: Loopback data is inverted)	R/W	0
3	dec_bypass	1 = Bypass 8b10b decoder 0 = Normal operation	R/W	0
4	ctc_bypass	1 = Bypass clock toleration compensation 0 = Normal operation	R/W	0
5	rx_gear_bypass	1 = Bypass PCS Rx gear box 0 = Normal operation	R/W	0
6	signal_detect	1 = Force enabling the Rx link state machine 0 = Dependent of ffc_signal_detect to enable the Rx link state machine	R/W	0
7	lsm_disable	1 = Disable Rx link state machine 0 = Enable Rx link state machine	R/W	0

Table 56. PCS Control Register 6 (CH_05)

If neither match enable bit is set below, single character skip matching is performed using match 4.

Bit	Name	Description	R/W	Default
3:0	Reserved			
4	match_2_enable	1 = Enable two character skip matching (using match 4,3)	R/W	1
5	match_4_enable	1 = Enable four character skip matching (using match 4, 3, 2, 1)	R/W	0
7:6	min_ipg_cnt [1:0]	Minimum IPG to enforce	R/W	2'b11

Table 57. PCS Control Register 7 – CC match 1 LO (CH_06)

Bit	Name	Description	R/W	Default
7:0	cc_match_1 [7:0]	Lower bits of user defined clock compensator skip pattern 1	R/W	8'h00

Table 58. PCS Control Register 8 – CC match 2 LO (CH_07)

Bit	Name	Description	R/W	Default
7:0	cc_match_2[7:0]	Lower bits of user defined clock compensator skip pattern 2	R/W	8'h00

Table 59. PCS Control Register 9 – CC match 3 LO (CH_08)

Bit	Name	Description	R/W	Default
7:0	cc_match_3[7:0]	Lower bits of user defined clock compensator skip pattern 3	R/W	8'hBC

Table 60. PCS Control Register 10 – CC match 4 LO (CH_09)

Bit	Name	Description	R/W	Default
7:0	cc_match_4[7:0]	Lower bits of user defined clock compensator skip pattern 3	R/W	8'h50

Table 61. PCS Control Register 11 – CC match HI (CH_0A)

Bit	Name	Description	R/W	Default
1:0	cc_match_1 [9:8]	Upper bits of user defined clock compensator skip pattern 1 [9] = Disparity error [8] = K control	R/W	2'b00
2:3	cc_match_2 [9:8]	Upper bits of user defined clock compensator skip pattern 2 [9] = Disparity error [8] = K control	R/W	2'b00
4:5	cc_match_3 [9:8]	Upper bits of user defined clock compensator skip pattern 3 [8] = K control [9] = Disparity error	R/W	2'b01
7:6	cc_match_4 [9:8]	Upper bits of user defined clock compensator skip pattern 4 [8] = K control [9] = Disparity error	R/W	2'b01

Table 62. PCS Control Register 12 – UDF Comma Mask LO (CH_0B)

Bit	Name	Description	R/W	Default
7:0	udf_comma_mask [7:0]	Lower bits of user defined comma mask	R/W	8'hFF

Table 63. PCS Control Register 13 – UDF comma a LO (CH_0C)

Bit	Name	Description	R/W	Default
7:0	udf_comma_a [7:0]	Lower bits of user defined comma character 'a'	R/W	8'h83

Table 64. PCS Control Register 14 – UDF comma b LO (CH_0D)

Bit	Name	Description	R/W	Default
7:0	udf_comma_b [7:0]	Lower bits of user defined comma character 'b'	R/W	8'h7c

Table 65. PCS Control Register 15 – UDF comma HI (CH_0E)

Bit	Name	Description	R/W	Default
1:0	Reserved			
3:2	udf_comma_mask [9:8]	Upper bits of user defined comma mask	R/W	2'b11
5:4	udf_comma_b [9:8]	Upper bits of user defined comma character 'b'	R/W	2'b01
7:6	udf_comma_a [9:8]	Upper bits of user defined comma character 'a'	R/W	2'b10

Table 66. PCS Interrupt Control Register 16 (CH_0F)

Bit	Name	Description	R/W	Default
0	fb_tx_fifo_error_int_ctl	1 = Enable interrupt on empty/full condition in the transmit FPGA bridge FIFO.	R/W	0
1	fb_rx_fifo_error_int_ctl	1 = Enable interrupt on empty/full condition in the receive FPGA bridge FIFO.	R/W	0
2	cc_overrun_int_ctl	1 = Enable interrupt for cc_overrun 0 = Disable interrupt for cc_overrun.	R/W	0
3	cc_underrun_int_ctl	1 = Enable interrupt for cc_underrun 0 = Disable interrupt for cc_underrun.	R/W	0
7:4	Reserved		R/W	0

Per Channel SerDes Control Register Details

Table 67. SERDES Control Register 1 (CH_10)

Bit	Name	Description	R/W	Default
0	tpwdnb	0 = Power down transmit channel 1 = Power up transmit channel	R/W	0
1	rate_mode_tx	0 = Full rate selection for transmit 1 = Half rate selection for transmit	R/W	0
2	tx_div11_sel	0 = Full rate selection for transmit (high definition SMPTE) 1 = Divide by 11 selection for transmit (standard definition SMPTE)	R/W	0
3	ldr_core2tx_sel	1 = Select low speed serial data from FPGA core	R/W	0
4	tdrv_pre_en	1 = TX driver de-emphasis enable 0 = TX driver de-emphasis disable.	R/W	0
5	tdrv_post_en	1 = TX driver post-emphasis enable 0 = TX post-emphasis disable	R/W	0
6	tx_pre_sign	1 = TX preemphasis not inverted 0 = TX preemphasis inverted	R/W	0
7	tx_post_sign	1 = TX post emphasis not inverted 0 = TX post emphasis inverted	R/W	0

Table 68. SERDES Control Register 2 (CH_11)

Bit	Name	Description	R/W	Default
4:0	rterm_tx [4:0]	TX resistor termination select. Disabled when PCIe feature is enabled (Ohms) 00000 = 5K 00001 = 80 00100 = 75 00110 = 70 01011 = 60 10011 = 50 11001 = 46 All other values are RESERVED	R/W	4'b0000
6:5	tx_cm_sel[1:0]	Select TX output common mode voltage 00 = power down 01 = 0.6 V 10 = 0.55 V 11 = 0.5 V	R/W	2'b00
7	Internal use only			

Table 69. SERDES Control Register 3 (CH_12)

Bit	Name	Description	R/W	Default
1:0	tdrv_slice0_sel[1:0]	TX drive slice enable for slice 0: 00 = Power Down 01 = Select Main Data 10 = Select Pre Data 11 = Select Post Data	R/W	2'b00
3:2	tdrv_slice1_sel[1:0]	TX drive slice enable for slice 1: 00 = Power Down 01 = Select Main Data 10 = Select Pre Data 11 = Select Post Data	R/W	2'b00
5:4	tdrv_slice2_sel[1:0]	TX drive slice enable for slice 2: 00 = Power Down 01 = Select Main Data 10 = Select Pre Data 11 = Select Post Data	R/W	2'b00
7:6	tdrv_slice3_sel[1:0]	TX drive slice enable for slice 3: 00 = Power Down 01 = Select Main Data 10 = Select Pre Data 11 = Select Post Data	R/W	2'b00

Table 70. SERDES Control Register 4 (CH_13)

Bit	Name	Description	R/W	Default
3:0	Internal use only			
5:4	tdrv_slice3_cur[1:0]	Controls the output current for slice 3. Every 100 uA increases the output amplitude by 100 mV. 00 = 800 uA 01 = 1600 uA 10 = 2400 uA 11 = 3200 uA	R/W	2'b00
7:6	tdrv_slice4_cur[1:0]	Controls the output current for slice 4. Every 100 uA increases the output amplitude by 100 mV. 00 = 800 uA 01 = 1600 uA 10 = 2400 uA 11 = 3200 uA	R/W	0

Table 71. SERDES Control Register 5 (CH_14)

Bit	Name	Description	R/W	Default
2:0	tdrv_slice0_cur[2:0]	TX driver slice 0 current settings. Increases the output current of slice 0 by 100 uA which corresponds to a 100 mV differential increase in output amplitude. 000 = default swing amplitude (100 uA) 001 = 200 uA 010 = 300 uA 011 = 400 uA 100 = 500 uA 101 = 600 uA 110 = 700 uA 111 = 800 uA	R/W	2'b00
5:3	tdrv_slice1_cur[2:0]	TX driver slice 1 current settings. Increases the output current of slice 1 by 100 uA which corresponds to a 100 mV differential increase in output amplitude. 000 = default swing amplitude (100 uA) 001 = 200 uA 010 = 300 uA 011 = 400 uA 100 = 500 uA 101 = 600 uA 110 = 700 uA 111 = 800 uA	R/W	2'b00
7:6	tdrv_slice2_cur[1:0]	Controls the output current for slice 2. Every 100 uA increases the output amplitude by 100 mV. 00 = 800 uA 01 = 1600 uA 10 = 2400 uA 11 = 3200 uA	R/W	2'b00

Table 72. SERDES Control Register 6 (CH_15)

Bit	Name	Description	R/W	Default
3:0	lb_ctl[3:0]	Serial loopback control, only one bit should be selected: [3] = slb_r2t_dat_en serial RX to TX LB enable (CDR data) [2] = slb_r2t_ck_en serial RX to TX LB enable (CDR clock) [1] = slb_eq2t_en serial loopback from equalizer to driver enable [0] = slb_t2r_en serial TX to RX LB enable	R/W	4'b0000
5:4	tdrv_dat_sel [1:0]	Driver output select: 00 = Data from Serializer muxed to driver (normal operation) 01 = Data rate clock from Serializer muxed to driver 10 = Serial Rx to Tx LB (data) if slb_r2t_dat_en='1' 10 = Serial Rx to Tx LB (clock) if slb_r2t_ck_en='1' 11 = Serial LB from equalizer to driver if slb_eq2t_en='1'	R/W	2'b00
7:6	tdrv_slice5_cur[1:0]	Controls the output current for slice 5. Every 100 uA increases the output amplitude by 100mV. 00 = 800 uA 01 = 1600 uA 10 = 2400 uA 11 = 3200 uA	R/W	2'b00

Table 73. SERDES Control Register 7 (CH_16)

Bit	Name	Description	R/W	Default
0	rpwdnb	0 = Power down receiver channel 1 = Power up receiver channel	R/W	0
1	rate_mode_rx	0 = Full rate selection for receive 1 = Half rate selection for receive	R/W	0
2	rx_div11_sel	0 = Full rate selection for receive (high definition SMPTE) 1 = Divide by 11 selection for receive (standard definition SMPTE)	R/W	0
3	ldr_rx2core_sel	Enables boundary scan input path for routing the high speed RX inputs to a lower speed Serdes in the FPGA (for out of band application)	R/W	0
4	rx_refck_sel	Rx CDR Reference Clock Select 0 = rx_refck_local 1 = ck_core_rx		
5	rcv_dcc_en	1 = Receiver DC coupling enable. 0 = AC coupling (default)	R/W	0
7:6	rxterm_cm[1:0]	Command mode voltage for RX input termination: 00 = RX Input Supply 01 = Floating (AC Ground) 10 = GND 11 = RX Input Supply	R/W	0

Table 74. SERDES Control Register 8 (CH_ 17)

Bit	Name	Description	R/W	Default
4:0	rterm_rx [4:0]	RX input termination setting (Ohms): 00000 = 5K 00001 = 80 00100 = 75 00110 = 70 01011 = 60 10011 = 50 11001 = 46 All other values are RESERVED	R/W	4’b0000
5	Reserved			
7:6	rxin_cm[1:0]	Common mode voltage for equalizer input in ac coupling: 00 = 0.7 V 01 = 0.65 V 10 = 0.75 V 11 = CMFB	R/W	2’b00

Table 75. Serdes Control Register 9 (CH_18)

Bit	Name	Description	R/W	Default
3:0	Reserved			
4	Reserved		R/W	0
5	Reserved		R/W	0
6	fc2dco_floop	1 = Force DCO lock to the frequency loop	R/W	0
7	fc2dco_dloop	1 = Force DCO lock to the data loop	R/W	0

Table 76. Serdes Control Register 10 (CH_19)

Bit	Name	Description	R/W	Default
0	req_en	1 = Receiver equalization enable 0 = Receiver equalization disable	R/W	0
4:1	rx_rate_sel [3:0]	Equalizer pole position select: 0000 = TBD 0001 = TBD 0010 = TBD 0011 = TBD 0100 = TBD 0101 = TBD 0110 = TBD 0111 = TBD 1000 = TBD 1001 = TBD 1010 = TBD 1011 = TBD 1100 = TBD 1101 = TBD 1110 = TBD 1111 = TBD,	R/W	4'h0
6:5	req_lvl_set[1:0]	Level setting for equalization: 00 = 6 dB 01 = 9 dB 10 = 12 dB 11 = not used	R/W	2'b00
7	Reserved			

Table 77. Serdes Control Register 11 (CH_1A)

Bit	Name	Description	R/W	Default
2:0	rx_dco_ck_div[2:0]	VCO output frequency select: 00x = Divided by 1 01x = Divided by 2 100 = Divided by 4 101 = Divided by 8 110 = Divided by 16 111 = Divided by 32	R/W	3'b000
3	pden_sel	This signal is used to disable the phase detector in the CDR during electrical idle. When set to 1, checks if los is set. If los is 0 then disables the phase detector (or data loop).	R/W	0
4	dco_facq_rst	Reset signal to trigger the DCO frequency acquisition process when it's necessary	RW	0
5	dco_calib_rst	Reset signal to trigger the DCO calibration process when it's necessary	RW	0
6	en_recalib	Enable recalibration: 0 = Disable 1 = Enable re-calibration	R/W	0
7	band_calib_mode	Calibration mode: 0 = Disable 1 = Enable	R/W	0

Table 78. Serdes Control Register 12 (CH_1B)

Bit	Name	Description	R/W	Default
2:0	rx_los_lvl[2:0]	Sets differential p-p threshold voltage for loss of signal detection: 000 = TBD 001 = TBD 010 = TBD 011 = TBD 100 = TBD 101 = TBD 110 = TBD 111 = TBD	R/W	3'b000
4:3	rx_los_ceq[1:0]	Sets the equalization value at input stage of LOS detector: 00 = TBD 01 = TBD 10 = TBD 11 = TBD	R/W	2'b00
5	rx_los_hyst_en	Enables hysteresis in detection threshold level	R/W	0
6	rx_los_en	Enables loss of signal detector: 0 = Disabled 1 = Enabled	R/W	0
7	rlos_sel	Enables LOS detector output before enabling CDR phase detector after calibration: 0 = Disabled 1 = Enabled (If the channel is being used, this bit should be set to 1)	R/W	0

Table 79. Serdes Interrupt Control Register 1 (CH_ 1E)

Bit	Name	Description	R/W	Default
0	~rlol_int_ctl	1= Enable interrupt when receiver is locked	R/W	0
1	rlol_int_ctl	1= Enable interrupt for receiver loss of lock	R/W	0
2	Reserved			
3	Reserved			
4	~rlos_int_ctl	1 = Enable interrupt for receiver Rx Loss of Signal when input level meets or is greater than programmed LOW threshold	R/W	0
5	rlos_int_ctl	1 = Enable interrupt for Rx Loss of Signal when input levels fall below the programmed LOW threshold (using rlos_set)	R/W	0
6	pcie_det_done_int_ctl	1 = Enable interrupt for detection of a far-end receiver for PCI Express	R/W	0
7	Reserved			

Per Channel Reset and Clock Control Register Details

Table 80. Reset and Clock Control Register 1 (CH_1F)

Bit	Name	Description	R/W	Default
0	sel_sd_rx_clk	1 = FPGA Bridge write clock and Elastic Buffer read clock driven by serdes recovered clock 0 = FPGA Bridge write clock and Elastic Buffer read clock driven by ff_ebrd_clk	R/W	0
1	ff_rx_h_clk_en	1 = Enable ff_rx_h_clk	R/W	0
2	ff_rx_f_clk_dis	1 = Disable ff_rx_f_clk	R/W	0
3	ff_tx_h_clk_en	1 = Enable ff_tx_h_clk	R/W	0
4	ff_tx_f_clk_dis	1 = Disable ff_tx_f_clk	R/W	0
5	lane_tx_rst	1 = Assert reset signal to transmit logic	R/W	0
6	lane_rx_rst	1 = Assert reset signal to receve logic	R/W	0
7	rrst	1 = Rx reset	RW	0

Per Channel PCS Status Register Details

Table 81. PCS Status Register 1 (CH_30)

Bit	Name	Description	R/W	Int
0	fb_tx_fifo_error	1 = FPGA bridge (FB) TX FIFO overrun 0 = FB TX FIFO not overrun	RO	Y
1	fb_rx_fifo_error	1 = FPGA bridge (FB) RX FIFO overrun 0 = FB RX FIFO not overrun	RO	Y
2	cc_overrun	1 = CC FIFO overrun 0 = CC FIFO not overrun	RO	Y
3	cc_underrun	1 = CC FIFO underrun 0 = CC FIFO not underrun	RO	Y
4	pfifo_error	1 = Parallel FIFO error 0 = No Parallel FIFO error	RO	Y
7:5	Reserved			

Table 82. PCS Status Register 2 (CH_31)

Bit	Name	Description	R/W	Int
7:0	prbs_error_cnt[7:0]	Count of the number of PRBS errors. Clears to zero on read. Sticks at FF.	RO CR	N

Table 83. PCS Status Register 3 (CH_32)

Bit	Name	Description	R/W	Int
3:0	wa_offset[3:0]	Word Aligner Offset	RO	N
7:4	Reserved			

Table 84. PCS General Interrupt Status Register 4 (CH _33)

Bit	Name	Description	R/W	Int
0	fb_tx_fifo_error_int	1 = Interrupt generated on fb_tx_fifo_error 0 = Interrupt not generated fb_tx_fifo_error.	RO CR	Y
1	fb_rx_fifo_error_int	1 = Interrupt generated on fb_rx_fifo_error. 0 = Interrupt not generated fb_rx_fifo_error.	RO CR	Y
2	cc_overrun_int	1 = Interrupt generated on cc_overrun 0 = Interrupt not generated on cc_overrun	RO CR	Y
3	cc_underrun_int	1 = Interrupt generated on cc_underrun 0 = Interrupt not generated on cc_underrun		
			RO CR	Y
7:4	Reserved			

Table 85. PCS Status Register 5 (CH_34)

Bit	Name	Description	R/W	Int
0	cc_we_o	1 = Elastic FIFO write enable 0 = Elastic FIFO write disable	RO	N
1	cc_re_o	1 = Elastic FIFO read enable 0 = Elastic FIFO read disable	RO	N
3:2	Reserved			
4	fb_txrst_o	1 = FPGA bridge Tx Normal Operation 0 = FPGA bridge Tx reset	RO	N
5	fb_rxrst_o	1 = FPGA bridge Rx Normal Operation 0 = FPGA bridge Rx reset	RO	N
6	ffs_ls_sync_status	1 = Sync in the link state machine. 0 = Not sync in the LSM.	RO	N
7	Reserved			

Table 86. PCS Status Register 6 (CH_35)

Bit	Name	Description	R/W	Default
7:0	Reserved			

Per Channel SerDes Status Register Details

Table 87. SERDES Status Register 1 (CH_36)

Bit	Name	Description	R/W	Int
0	~rlol	1 = Indicates that CDR has locked to data.	RO	Y
1	rlol	1 = Indicates CDR loss of lock to data. CDR is locked to reference clock.	RO	Y
3:2	Reserved		RO CR	Y
4	~rlos	1 = Indicates that the input signal detected by receiver is greater than or equal to the programmed LOW threshold	RO CR	Y
5	rlos	1 = Indicates that the input signal detected by receiver is below the programmed LOW threshold	RO CR	Y
6	pci_det_done	1 = Receiver detection process completed by SerDes transmitter. 0 = Receiver detection process not completed by SerDes transmitter.	RO CR	Y
7	Reserved			

Table 88. SERDES Status Register 2 (CH_37)

Bit	Name	Description	R/W	Int
0	pci_connect	1 = Receiver detected by SerDes transmitter (at the transmitter device). 0 = Receiver not detected by SerDes transmitter (at the transmitter device).	RO	N
3:1	Reserved			
4	DCO_FACQ_DONE	1 = indicates DCO frequency acquisition done	RO	N
5	DCO_FACQ_ERR	1 = indicates DCO frequency acquisition error (>300ppm)	RO	N
6	DCO_CALIB_DONE	1 = indicates DCO calibration done	RO	N
7	DCO_CALIB_ERR	1 = indicates DCO calibration might be wrong (L/H boundary band selected)	RO	N

Table 89. SERDES Interrupt Status Register 5 (CH _3A

Bit	Name	Description	R/W	Int
0	~rlol_int	1 = Interrupt generated for ~rlol	CO CR	Y
1	rlol_int	1 = Interrupt generated for rlol	CO CR	Y
3:2	Reserved			
4	~rlos_int	1 = Interrupt generated for ~rlos	CR RO	Y
5	rlos_int	1 = Interrupt generated for rlos	CR RO	Y
6	pci_det_done_int	1= Interrupt generated for pci_det_done	CR RO	Y
7	Reserved			

Appendix B. Register Settings for Various Standards

Table 90. Per Dual Register Settings for Different Standards

Register	1GbE	10GbE	RapidIO 1x	RapidIO 4x	PCI-Ex 1x	PCI-Ex 4x
comma_a_lo	hex 03	hex 03	hex 03	hex 03	hex 03	hex 03
comma_b_lo	hex fc	hex fc	hex fc	hex fc	hex fc	hex fc
comma_mask_lo	hex 7f	hex 7f	hex 7f	hex 7f	hex 7f	hex 7f

Table 91. Per Channel Register Settings for Different Standards

Character	1GbE	10GbE	PCI-Ex	RapidIO
K23.7 (F7)	Carrier extend		PAD	
K27.7 (FB)	SOP	ST	Start TLP	A (align)
K28.0 (1C)		SKIP R	SKIP	SC
K28.1 (3C)			FTS	
K28.2 (5C)		SoS	Start DLLP	
K28.3 (7C)		ALIGN A	IDLE	PD
K28.4 (9C)		SEQ		
K28.5 (BC)	+ D5.6 or D16.2			
= IDLE	SYNC K	COMMA	K	
K28.6 (DC)				
K28.7 (FC)				
K29.7 (FD)	EOP	T	END	R (skip)
K30.7 (FE)	ERR	ERR	END BAD	