CertusPro-NX SerDes/PCS User Guide

Technical Note

FPGA-TN-02245-1.2

July 2023

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults and associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of vocabulary used in this document.

Vocabulary	Definition
AC	Alternating Current
AN	Auto Negotiation
ASIC	Application Specific Integrated Circuit
BER	Bit Error Ratio
BIST	Built In Self-Test
CDR	Clock and Data Recovery
CE	Consumer Electronic
CIS	CMOS Image Sensor
CoaXPress	An interface to connect devices (typical cameras) to host (typical frame grabbers)
COMMA	The seven bit comma string is defined as either 7'b0011111 (comma+) or 7'b1100000 (comma-)
CTC	Clock Tolerance Compensation
CTLE	Continuous Time Linear Equalizer
DC	Direct Current
DDR	Double Data Rate
DFE	Decision Feedback Equalization
DFF	D-Flip Flop
DP	DisplayPort
DSP	Digital Signal Processor
eDP	Embedded DisplayPort
EI1	Electrical Idle 1
EI2	Electrical Idle 2
EI4	Electrical Idle 4
EIEOS	Electrical Idle Exit Ordered Set
EMI	Electro-Magnetic Interference
EPCS	External PCS
ESR	Equivalent Series Resistance
FEC	Forward Error Correction
FFE	Feed Forward Equalizer
FIFO	First Input First Output
FIR	Finite Impulse Response
FOM	Figure of Merit
FPGA	Field-Programmable Gate Array
FS	Full Swing
FSM	Finite State Machine
HBR	High Bit Rate
HBR2	High Bit Rate 2
HBR3	High Bit Rate 3
Gen1/Gen2/Gen3	Generation 1/Generation 2/Generation 3
GMII	Gigabit Media Independent Interface
GPLL	General Purpose PLL
GUI	Graphic User Interface
HCSL	High Speed Current Steering Logic, or Host Controller Signal Level
IP	Intellectual Property
IPG	Inter-Packet Gap

Vocabulary	Definition
ISI	Inter-Symbol Interference
I/F	Interface
JIIA	Japan Industrial Imaging Association
JTAG	Joint Test Action Group
L2	A low power state inside PCIe LTSSM
LF	Low Frequency
LMMI	Lattice Memory Mapped Interface
LOS	Loss of Signal
LSByte/LSB	Last Significant Byte
LTSSM	Link Training and Status State Machine
LVDS	Low-Voltage Differential Signaling
MAC	Media Access Control
MPCS	Multi-protocol Physical Coding Sublayer
MSByte/MSB	Most Significant Byte
NL	Number of Lane
OOB	Out Of Band
PC	Personal Computer
PCI	Peripheral Component Interconnect
PCIe / PCI-E	PCI Express
PCLK	Primary Clock
PCS	Physical Coding Sublayer
PFD	Phase Frequency Detector
PMA	Physical Media Attachment
PMD	Physical Medium Dependent
PIPE	PHY Interface for PCI Express
PISO	Parallel In Serial Out
PLL	Phase Locked Loop
PPM	Parts Per Million
PRBS	Pseudo Random Bit Sequence
PRD	Pseudo Random Data
QSGMII	Quad Serial Gigabit Media Independent Interface
RBR	Reduced Bit Rate
RC	Root Complex
RS	Reconciliation Sublayer
Rx	Receiver
RxDP	Receiver Data Positive pin
RxDN	Receiver Data Negative pin
RXAUI	Reduced 10 Gigabit Attachment Unit Interface
SATA	Serial Advanced Technology Attachment
SDR	Single Data Rate
SerDes	Serializer/Deserializer
SIPO	Serial In Parallel Out
SGMII	Serial Gigabit Media Independent Interface
SKP	SKIP order sets defined by PCI Express
SLVS-EC	Scalable Low Voltage Signaling with Embedded Clock
SRIO	Serial Rapid IO

Vocabulary	Definition
SSC	Spread Spectrum Clocking
SATA	Serial ATA
Sync	Synchronous
Tx	Transmitter
TxDP	Transmitter Data Positive pin
TxDN	Transmitter Data Negative pin
UI	Unit Interval
USB	Universal Serial Bus
USB3	USB SuperSpeed
VC0_RX	Virtual Channel 0 Receiver
VC0_TX	Virtual Channel 0 Transmitter
VESA	Video Electronics Standards Association
WAKE#	A wake up signal defined by PCIe
WIS	WAN Interface Sublayer
XAUI	10 Gigabit Ethernet Attachment Unit Interface
XGMII	10 Gigabit Media Independent Interface

1. Introduction

The Lattice Semiconductor CertusPro-NX device family has up to 8 channels embedded SerDes with associated Physical Coding Sublayer (PCS) logic, which supports PCI Express Gen1/2/3 hard IP Core, DisplayPort, 10GBASE-R, 1000BASE-X, SGMII, QSGMII, XAUI, SLVS-EC, and CoaXPress protocols. There are two kinds of Physical Coding Sublayer logic inside the CertusPro-NX device: one is for PCI Express only; the other is Multi-Protocol Physical Coding Sublayer (MPCS) for protocols other than PCI Express.

Each channel of MPCS contains dedicated transmit and receive logic for high-speed, full-duplex serial data transfer at data rates up to 10.3125 Gb/s. The MPCS logic in each channel can be configured to support corresponding protocols. In addition, the protocol-based logic can be fully or partially bypassed in a number of configurations to provide the flexibility of designing user's own high-speed data interface. SerDes channel input can be independently AC-coupled or DC-coupled to meet the requirements from different protocols.

2. Features

- Single Channel MPCS Functionalities
  - From 625 Mb/s up to 10.3125 Gb/s per channel
  - Word alignment and link synchronization FSM
  - Supports DisplayPort, SLVS-EC, CoaXPress, and Ethernet 1000BASE-X/SGMII/XAUI/QSGMII protocols
  - Supports popular 8B/10B PCS-based protocols
  - Supports user-specified generic 8B/10B mode
  - Supports Ethernet 64B/66B coding and decoding
  - Supports per channel configuration
  - PMA-only mode allows direct 8-bit or 10-bit interface to FPGA logic
- Multiple Channel Alignment
  - Up to eight channels of the same protocol
  - Minimized Transmitter (Tx) lane-to-lane skew
  - Receiver (Rx) multiple lane de-skew (up to 100 UI)
- Multiple Protocol Clock Tolerance Compensation (CTC) logic
  - Compensates for frequency difference between reference clock and received data rate
  - Allows user-defined SKIP pattern of 1, 2, or 4 bytes in length
- Integrated Loopback Modes for System Debugging
  - Two loopback modes are provided by SerDes (PMA)
  - Two loopback modes are provided by MPCS 8B/10B mode
  - Three loopback modes are provided by MPCS 64B/66B mode
- Rx Eye Monitor
  - Integrated 2D eye-scan for each Rx channel
  - First on-chip SerDes debug tool in Lattice general purpose FPGAs
- SerDes Power Supply Monitor
  - Integrated real-time monitor for SerDes power supply
- Easy-to-Use Design Interface in Lattice Radiant™ Design Tool
  - Easy-to-use Graphic User Interface (GUI)
  - Some easy-to-use soft IP cores and example projects available

3. Using This Technical Note

This technical note provides a thorough description of the complete functionality of the embedded SerDes and associated PCS logic, including the description of:

- Architecture of the CertusPro-NX SerDes/PCS module and interface
- SerDes/PCS function
- Clock and reset
- SerDes/PCS debug capabilities
- SerDes/PCS register access
- SerDes/PCS usage for different protocols
- SerDes/PCS block latency
- SerDes/PCS generation in Lattice Radiant software
- The status and control registers associated with the SerDes and PCS logic, which can be accessed through the Lattice Memory Mapped Interface (LMMI).

The electrical and timing characteristics of the embedded SerDes and the package pinout information are provided in CertusPro-NX Family Data Sheet (FPGA-DS-02086).

The Lattice Radiant design tools support all modes of PCS. Most modes are dedicated to applications for specific industry standard data protocols listed in the Features section. Other modes are more general purpose in nature in order to let user customize user's own application settings. Radiant design tools allow the user to define the mode for each quad in the design. This technical note describes the operation of the SerDes and PCS for all modes supported by Lattice Radiant software.

4. Standards Supported

The supported standards are listed in Table 4.1. Note that only flip-chip package based device can support standards with data rate higher than 6.25 Gbps; refer to CertusPro-NX Family Data Sheet (FPGA-DS-02086) for more details.

Table 4.1. Standards Supported by the SerDes/PCS

Standard	Data Rate (Mb/s)	System Reference Clock (MHz)	FPGA Clock (MHz)	Number of Link Width	Encoding Style
PCI Express Gen1[1]	2500	100	125	×1, ×2, ×4	8b10b
PCI Express Gen2[1]	5000	100	125	×1, ×2, ×4	8b10b
PCI Express Gen3[1]	8000	100	250	×1, ×2, ×4	128b130b
Ethernet 1000BASE-X	1250	125	125	x1	8b10b
Ethernet SGMII	1250	125	125	×1	8b10b
Ethernet XAUI	3125	156.25	156.25	×4	8b10b
Ethernet QSGMII	5000	125	125	×1	8b10b
Ethernet 10GBASE-R[2]	10312.5	161.1328125	161.1328125	×1	64b66b
SLVS-EC Grade1	~1250	—	~125	×1, ×2, ×4, ×6, ×8	8b10b
SLVS-EC Grade2	~2500	—	~125	×1, ×2, ×4, ×6, ×8	8b10b
SLVS-EC Grade3	~5000	—	~125	×1, ×2, ×4, ×6, ×8	8b10b
CoaXPress CXP-1	1250	125	125	×1~×8	8b10b
CoaXPress CXP-2	2500	125	125	×1~×8	8b10b
CoaXPress CXP-3	3125	156.25	156.25	×1~×8	8b10b
CoaXPress CXP-5	5000	125	125	×1~×8	8b10b
CoaXPress CXP-6	6250	156.25	156.25	×1~×8	8b10b
DP/eDP RBR	1620	108	162	×1, ×2, ×4	8b10b
DP/eDP HBR	2700	135	135	×1, ×2, ×4	8b10b
DP/eDP HBR2	5400	135	135	×1, ×2, ×4	8b10b
DP/eDP HBR3	8100	135	202.5	×1, ×2, ×4	8b10b
10-bit/20-bit/40-bit SerDes	625 – 8100	—	—	×1~×8	N/A
8-bit/16-bit/32-bit SerDes	625 – 8100	—	—	×1~×8	N/A
Generic 8b10b	625 – 8100	—	—	×1~×8	8b10b

Notes:

1. CertusPro-NX supports a maximum of four lanes PCIe with hard IP, refer to the SerDes/PCS Architecture section and PCI Express Architecture section for more details.
2. CertusPro-NX SerDes does not support Ethernet 10GBASE-KR. Moreover, not all channels can support 10GBASE-R, refer to MPCS Architecture section for more details.

5. Architecture Overview

5.1. Device Architecture

The SerDes/PCS block is arranged in quads containing logic for four full-duplex data channels. Figure 5.1 shows the arrangement of SerDes/PCS Quads on the CertusPro-NX 100k device.

Figure 5.1. CertusPro-NX 100k Device Block Diagram

Table 5.1 shows the maximum number of available SerDes/PCS channels for each CertusPro-NX device. Refer to CertusPro-NX Family Data Sheet (FPGA-DS-02086) for more details on the actual number of channels varying from package to package.

Table 5.1. Maximum Number of SerDes/PCS Channels per CertusPro-NX Device

Package	CertusPro-NX 50k	CertusPro-NX 100k
SerDes/PCS Quads	1	2
SerDes/PCS Channels	4	8

5.2. SerDes/PCS Architecture

Each CertusPro-NX SerDes/PCS quad includes four PMA channels, one PCI Express PCS Quad, one MPCS Quad, and related glue logic. Each PMA channel integrates CDR for Receiver and PLL for Transmitter. The PCI Express PCS is designed only for PCI Express, while the Multi-Protocol PCS (MPCS) is designed for other protocols.

CertusPro-NX device also integrates one PCI Express Link Layer Quad, which contains one PCIe ×1 block and one PCIe ×4 block. The PCIe ×4 PCI Express Link Layer block can be configured as ×1, ×2 or ×4 mode. The PCI Express Link Layer block, PCI Express PCS channels, and PMA channels constitute the complete PCI Express Hard IP block. Table 5.2 shows the maximum number of available PCI Express blocks information for the CertusPro-NX device. Refer to CertusPro-NX Family Data Sheet (FPGA-DS-02086) for more details on the actual number of channels varying from package to package.

Table 5.2. Maximum Number of PCI Express Blocks per CertusPro-NX Device

Package	CertusPro-NX 50k	CertusPro-NX 100k
PCI Express Hard IP	1	1
PCI Express PCS Quad	1	2
PCI Express Link Layer ×1 Block	1	1
PCI Express Link Layer ×4 Block	1	1

Figure 5.2 shows CertusPro-NX device SerDes/PCS quad architecture. For protocols other than PCI Express, the PCI Express PCS can be bypassed. The user can implement MPCS and PMA for Ethernet SGMII, XAUI, QSGMII, XGMII, SLVS-EC, CoaXpress, DP/eDP or Generic 8B/10B applications. The MPCS can also be bypassed so that the SerDes/PCS module works in PMA-only mode. Note that only SerDes/PCS Quad0 integrates PCIe Link Layer Hard IP (PCI Express Link Layer).

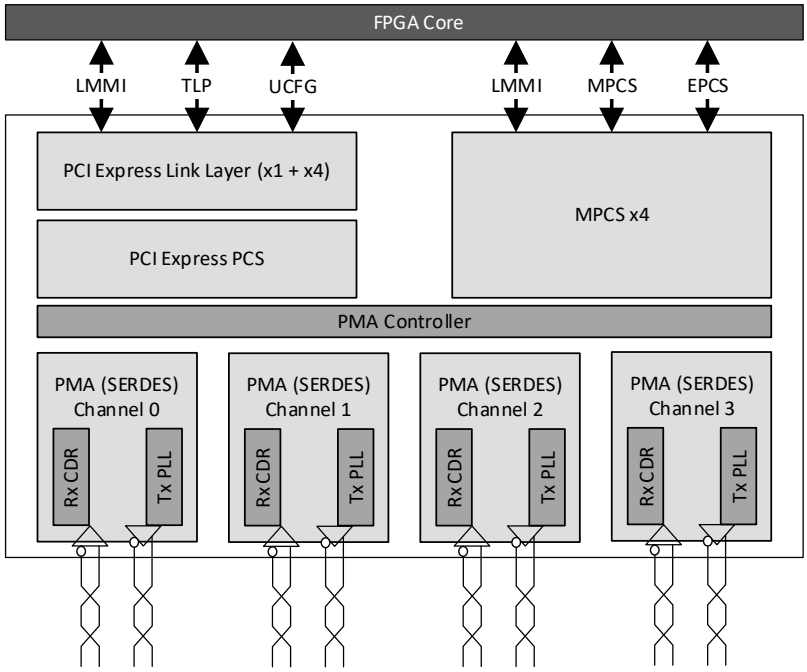

Figure 5.2. CertusPro-NX SerDes/PCS Quad Architecture

Table 5.3 shows the specific block usage for the corresponding SerDes/PCS mode.

Table 5.3. Block Usage for the Corresponding SerDes/PCS Mode

SerDes/PCS Mode	PCI Express	MPCS	PMA Only
PMA	√	√	√
PMA Controller	√	√	√
PCI Express PCS	√	—	—
PCI Express Link Layer Quad	√	—	—
MPCS	—	√	—

5.3. PCI Express Architecture

The CertusPro-NX PCI Express Link Layer block is a hard IP, which supports PCI Express Gen1, Gen2, and Gen3, and is compatible with PCI-SIG PCI Express Base Spec version 3.1a. The PCI Express Link Layer Block implements PHY Layer adaption, Data Link Layer, and Transaction Layer. The PCI Express Link Layer block together with PCI Express PCS, PMA Controller and PMA, constitute the complete PCI Express Hard IP block.

The PCI Express Hard IP supports both Endpoint and Root Complex modes. It supports up to four physical functions. Each of the four functions has independent PCI Express configurations space. It also supports ECC and parity data path protection. The LMMI interface is provided in the top wrapper that is to be used in user-controlled IP core configuration.

The PCI Express Link Layer Quad for all CertusPro-NX devices has four PIPE interfaces that can be used in different number of links. This can provide maximum flexibility, depending on the bandwidth of the application requirements. Figure 5.3 shows the architecture of the CertusPro-NX PCI Express Hard IP. There are four physical functions in the Link Layer ×4 block. There are four physical functions in the Link Layer ×1 block. Total there are eight physical functions in one Link Layer Quad. PCI Express Configuration Space access and Error Reporting are done through the dedicated interface (PCI Express Configuration Register Interface, UCFG). The core Configuration and Status Registers can be accessed using the corresponding LMMI interface per Link Layer block. PCI Express Link Layer block interface to PCI Express PCS block using the PIPE interface with up to 128-bit data width. The Link Layer block operates at the clock frequency of 250 MHz, while the user interface is at 125 MHz in Gen1/Gen2 speed and 250 MHz in Gen3 speed. The user interface, TLP interface, can be from 32 bits up to 128 bits wide depending on the core configuration. 32 bits of the TLP interface is used for ×1 mode, 64 bits of TLP interface is used for ×2 mode, and 128 bits of the TLP interface is used for ×4 mode.

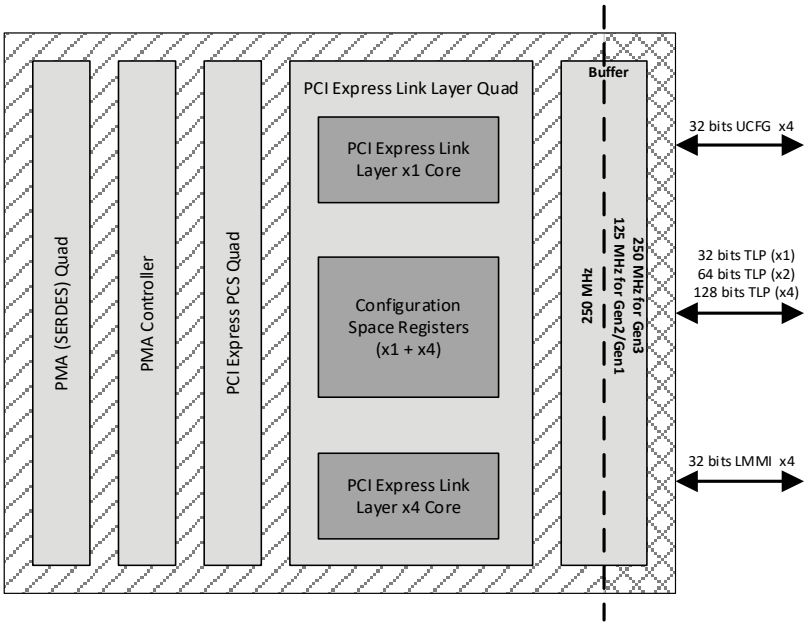

Figure 5.3. PCI Express Hard IP Architecture

PCI Express Link Layer Quad0 can be configured as x4 mode, x2 mode, x1 mode, x2_x1 mode, and x1_x1 mode. In x2_x1 mode, the Link Layer Quad can support two independent PCI Express applications, one in ×2 mode and the other in ×1 mode. Table 5.5 describes the SerDes/PCS lane mapping details.

Table 5.5. PCI Express Link Layer Quad Lane Mapping

Mode Name	Description
×4	All 4 lanes are used.
×2	Lane 0 and Lane 1 are used.
×1	Lane 0 is used.
×2_×1	Lane 0 and Lane 1 are used for ×2. Lane 3 is used for ×1.
×1_×1	Lane 0 and Lane 3 are used for ×1.

Figure 5.4 shows the PCI Express Link Layer block functional diagram. For more detailed information on CertusPro-NX PCI Express features, function descriptions, and IP usage, refer to PCIe X4 IP Core - Lattice Radiant Software (FPGA-IPUG-02126).

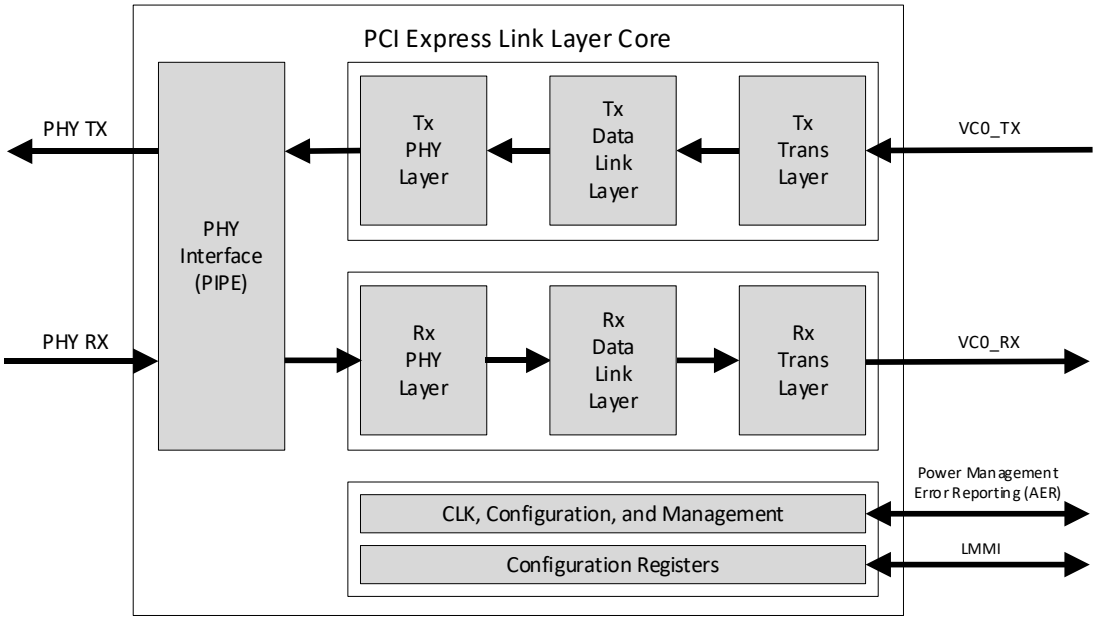

Figure 5.4. PCI Express Link Layer Functional Diagram

5.4. MPCS Architecture

CertusPro-NX Multi-Protocol PCS is designed for popular serial protocols other than PCI Express, such as Ethernet SGMII, XAUI, QSGMII, 10GBASE-R, SLVS-EC, CoaXpress, and DP/eDP. MPCS can be configured as Generic 8B/10B mode, for user-defined serial protocols other than those listed in the Features section.

Each MPCS Quad includes four channels. One Quad Common block communicates with each channel within the Quad to implement the multiple lane alignment function for both transmitter and receiver. The Quad Common block also talks to neighboring Quads to implement the lane alignment across Quad boundary. Figure 5.5 shows the simplified block diagram of MPCS Quad.

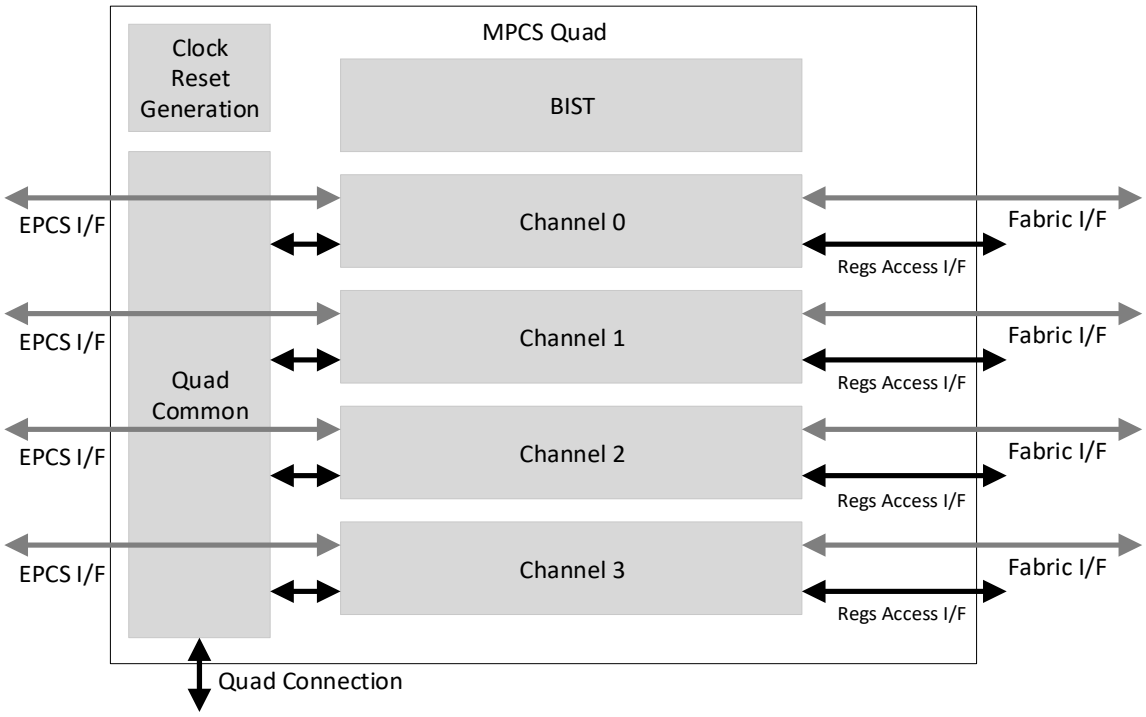

Figure 5.5. MPCS Quad Block Diagram

Four MPCS channels can work independently or together to compose multi-lane link. Each channel has one independent PMA interface, one LMMI interface, and one FPGA fabric interface. The internal functional block diagram of the channel is illustrated in Figure 5.6. Every Quad can be programed into one of the serval protocol-based modes.

Each Quad can be programmed with selected protocols that have the same reference clock source. For example, SGMII channel and QSGMII channel can share the same Quad using the same reference clock. When a Quad shares a PCI Express ×1 channel with a non-PCI Express channel, the reference clock for the Quad must be compatible with all protocols within the Quad. For example, a PCI Express spread spectrum reference clock is not compatible with most Gigabit Ethernet applications.

Since each Quad has its own reference clock, different Quads can support different standards on the same chip. This feature makes the CertusPro-NX family ideal for bridging between different standards. MPCS Quads are not dedicated solely to industry standard protocols. Each Quad and each channel within a Quad 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.

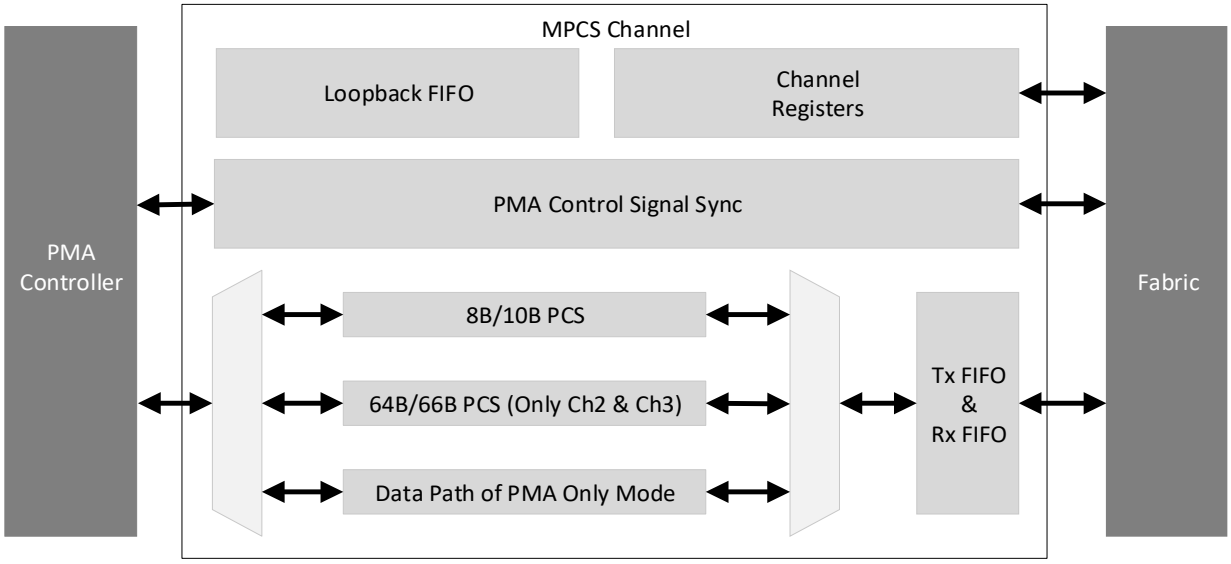

Figure 5.6. MPCS Channel Functional Block Diagram

Each MPCS channel has all the following functional blocks (Figure 5.6):

- The 8B/10B PCS: Supports popular 8B/10B PCS-based packet protocols.
- The 64B/66B PCS: Supports IEEE802.3 10GBASE-R protocol.
- PMA-only mode: Allows user logic to access PMA with very low latency.
- Channel register: Self-contained configuration and status register.
- PMA Control Signal Sync: Synchronize PMA control signals between PMA controller and user logic.
- Loopback FIFO: Internal FIFO for the corresponding loopback mode.
- Tx FIFO and Rx FIFO: Transmitter FIFO and Receiver FIFO, which are used for gearing and clock phase difference elimination.

However, in each quad, only lane 2 and lane 3 of the MPCS channel contains 64B/66B PCS. So, Ethernet 10GBASE-R is only supported by lane 2 and lane 3 of each quad, while other protocols can be supported by all lanes. Refer to Table 5.6 for more details. Lane 0 and lane 1 have no 64B/66B PCS path. So, related registers cannot be accessed by user logic.

Table 5.6. 10GBASE-R Lane Mapping

	Quad0				**Quad1**			
Lane ID	0	1	2	3	0	1	2	3
Supported Protocol	All protocols other than 10GE.							
10GE Supported	N/A	N/A	10GE	10GE	N/A	N/A	10GE	10GE

5.5. Reference Clock Architecture

Each PMA Quad has four PMA channels. Each PMA channel has one independent Tx PLL and one independent CDR PLL. However, all PMA channels share the same reference clock source. Each Quad requires its own reference clock, which can be sourced externally from package pins, SDQx_REFCLKP/SDQx_REFCLKN, or from the FPGA internally.

Figure 5.7 shows CertusPro-NX 100k device SerDes/PCS reference clock architecture. The Clock Tree block is designed for balancing the skew between different Quads based on one reference clock source, and the skew between different clock sources. The reference clock can source from General-purpose PLL (GPLL) output or Primary Clock (PCLK). However, it is not suggested to use the clock from GPLL output which contains too much jitter. With the Clock Tree, different PMA Quads can use the same reference clock source, which allows the user to implement more than four lanes multi-lane serial protocols based on two or three PMA Quads.

Each device also has two dedicated external reference clock input package pins, SD_EXTx_REFCLKP, SD_EXTx_REFCLKN, which allow the user to have more choices about the reference clock sources. Reference clock source from these two dedicated pins are needed when the number of lanes is larger than four (applications across the Quad).

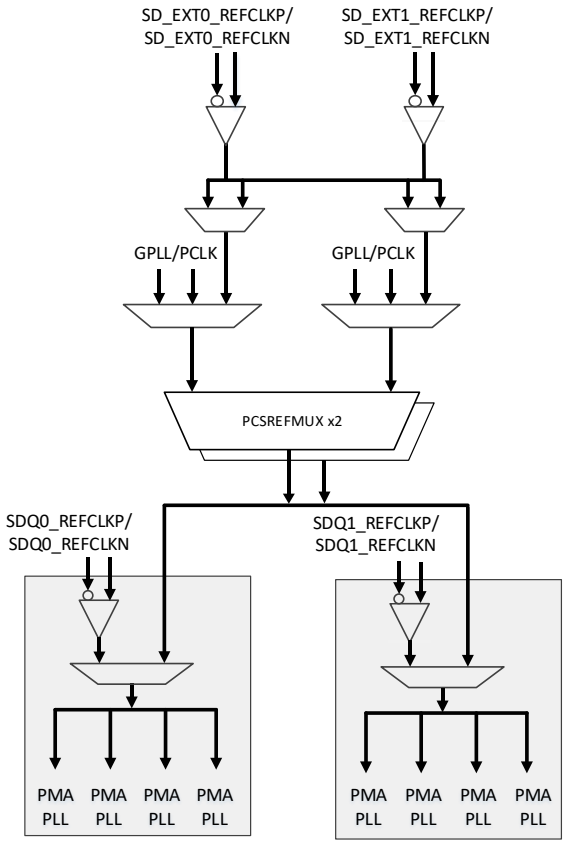

Figure 5.7. CertusPro-NX 100k Device SerDes Reference Clock Architecture

5.6. Multi-Protocol Design Consideration

Different combinations of protocols with a SerDes/PCS quad are permitted subject to certain conditions. One of the most basic requirements for two or more protocols sharing the same SerDes/PCS quad is that these protocols must have the same reference clock frequency. This restriction is due to these protocols share the same reference clock source.

Table 5.7 lists the support of mixed protocols within a CertusPro-NX SerDes/PCS quad. The reference clock must be without spread spectrum when PCI Express and other protocols share the same SerDes/PCS quad.

Table 5.7. CertusPro-NX Mixed Protocols within a Quad

Protocol		Protocol
PCI Express	&	1000BASE-X (GigE)
PCI Express	&	SGMII
PCI Express	&	QSGMII
PCI Express	&	CoaXPress CXP-1/CXP-2/CXP-5
1000BASE-X (GigE)	&	SGMII
1000BASE-X (GigE)	&	QSGMII
1000BASE-X (GigE)	&	CoaXPress CXP-1/CXP-2/CXP-5
SGMII	&	QSGMII
SGMII	&	CoaXPress CXP-1/CXP-2/CXP-5
QSGMII	&	CoaXPress CXP-1/CXP-2/CXP-5

5.7. SerDes/PCS Block Signal Interface

For PCI Express Hard IP mode, the LMMI interface, TLP interface, and UCFG interface are accessible (Figure 5.8).

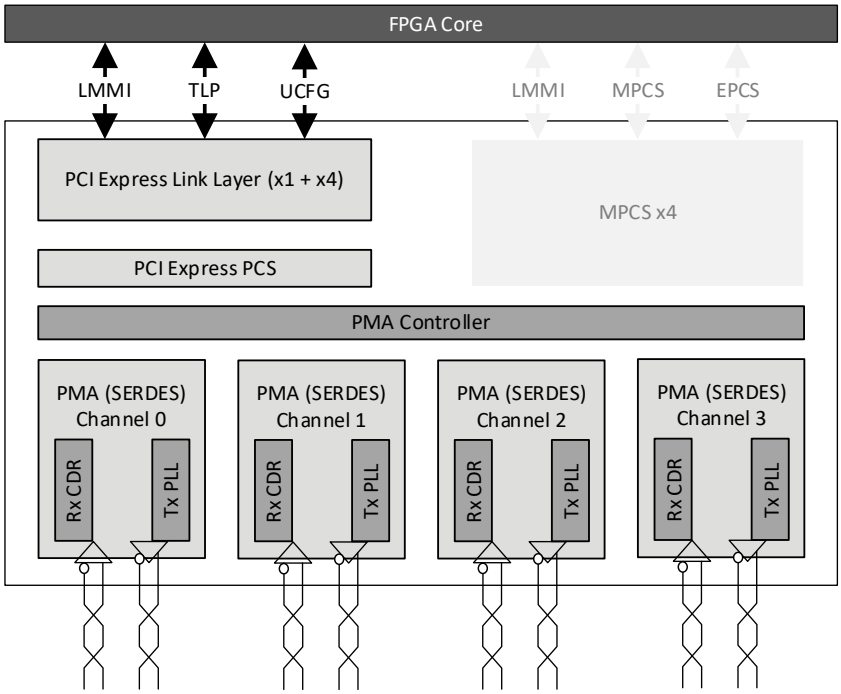

Figure 5.8. PCI Express Hard IP Mode

For MPCS mode, the MPCS interface and LMMI interface are accessible (Figure 5.9).

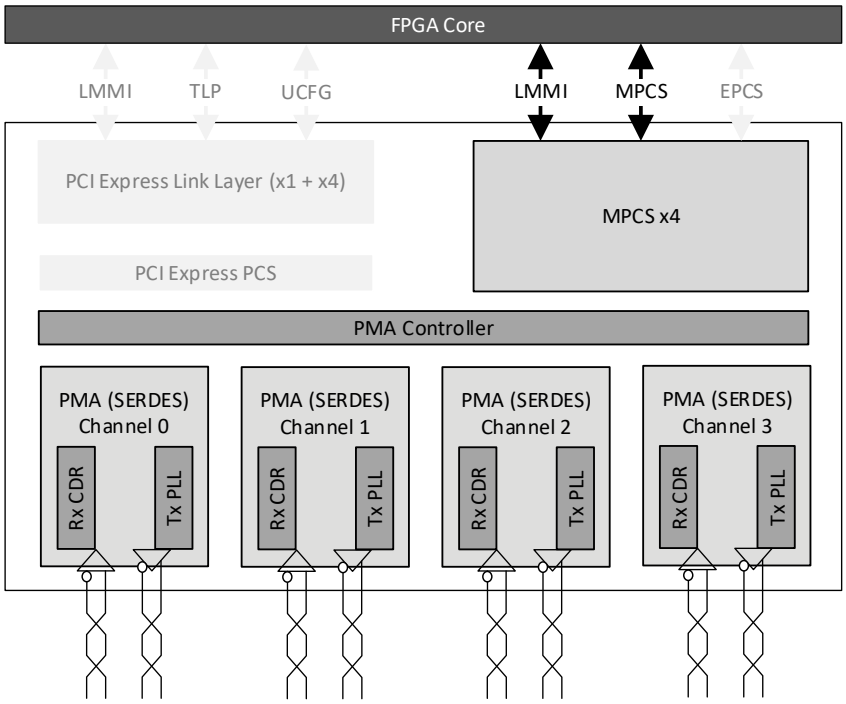

Figure 5.9. MPCS Mode

For PMA only mode, the EPCS interface and LMMI interface are accessible (Figure 5.10).

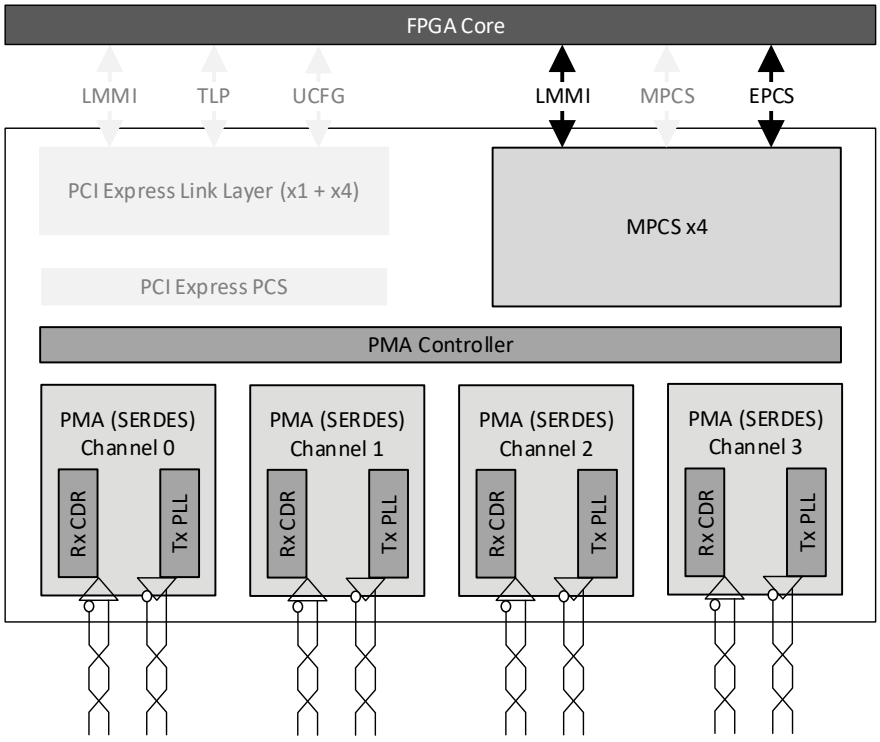

Figure 5.10. PMA Only Mode

5.8. Detailed Interface Descriptions

5.8.1. MPCS Interface

MPCS interface is accessible when SerDes/PCS is configured as MPCS mode. MPCS mode is designed for applications other than PCIe or PMA Only modes. Table 5.8 shows the detailed MPCS interface descriptions. All the signals listed in this table are per lane, and NL means the number of lanes.

Table 5.8. MPCS Interface

Port Name	I/O	Width	Description
Clock and Reset			
mpcs_rx_usr_clk_i	In	NL	User interface Rx clock input.
mpcs_tx_usr_clk_i	In	NL	User interface Tx clock input.
mpcs_tx_pcs_rstn_i	In	NL	Active low signal used to reset the Tx path of MPCS module. This signal must be released only when PMA has completed calibration.
mpcs_rx_pcs_rstn_i	In	NL	Active low signal used to reset the Rx path of MPCS module. This signal must be released only when PMA has completed calibration.
mpcs_cc_ clk_i	In	NL	Input clock for Clock Frequency Compensation. CTC clock input.
mpcs_rx_out_clk_o	Out	NL	PCS Rx output clock.
mpcs_tx_out_clk_o	Out	NL	PCS TX output clock.
mpcs_perstn_i	In	NL	Fundamental reset. Triggers PCS auto calibration.
mpcs_clkin_i	In	NL	This low-speed clock drives all calibration logic inside PMA Controller. The recommended frequency range is 100-300 MHz. Note that this clock should be stable and continuous after power on.
Tx/Rx FIFO Signals			
mpcs_tx_ch_din_i	In	80*NL	For the signal mapping of this port, refer to Table 5.12.
mpcs_tx_fifo_st_o	Out	4*NL	For the signal mapping of this port, refer to Table 5.12.
mpcs_rx_ch_dout_o	Out	80*NL	For the signal mapping of this port, refer to Table 5.12.
mpcs_rx_fifo_st_o	Out	4*NL	For the signal mapping of this port, refer to Table 5.12.
Elastic Buffer Signals (8B/10B PCS)			
mpcs_ebuf_empty_o	Out	NL	Elastic Buffer Empty output port. • 1'b1 – the frequency compensation buffer, Elastic Buffer, is empty. • 1'b0 – the frequency compensation buffer, Elastic Buffer, is not empty.
mpcs_ebuf_full_o	Out	NL	Elastic Buffer Full output port. • 1'b1 – the frequency compensation buffer, Elastic Buffer, is full. • 1'b0 – the frequency compensation buffer, Elastic Buffer, is not full.
mpcs_anxmit_i	In	NL	In GigE application case, the high level of this signal indicates the current state is GigE Auto-negotiation. In this state, replace /C1/, /C2/ with /I2/ ordered sets periodically so that the following stage (CTC) gets opportunity to perform clock frequency compensation.
Word Aligner Signals (8B/10B PCS)			
mpcs_walign_en_i	In	NL	Word alignment enabling input port. This function is useful if the automatic synchronization is not enabled. The rising edge of this signal triggers the word alignment operation.
mpcs_get_lsync_o	Out	NL	Link Synchronization output port. • 1'b1 – link synchronization is acquired. • 1'b0 – loss of link synchronization.

Port Name	I/O	Width	Description
Lane-to-Lane Deskew Signals (8B/10B PCS)			
mpcs_rx_get_lalign_o	Out	NL	Receive Lane align output port. • 1'b1 – alignment acquired. • 1'b0 – loss of alignment.
mpcs_rx_deskew_en_i	In	NL	Receive Deskew enable port. The rising edge of this signal triggers the lane-to-lane deskew operation.
BER Monitor (64B/66B PCS)			
mpcs_rx_hi_ber_o	Out	NL	The high level of this signal indicates the high bit error ratio is indicated.
Block Aligner (64B/66B PCS)			
mpcs_rx_blk_lock_o	Out	NL	The high level of this signal indicates the block lock is achieved.
PMA Control and Status Signals			
mpcs_pwrdn_i	In	2*NL	This signal is used to put PMA in powerdown mode. This signal has only three states. This signal is required to be clocked on mpcs_clkin_i. • 2'b11 – deep low-power state. • 2'b10 – low-power state. • 2'b00 – operational state.
mpcs_txhiz_i	In	NL	This signal is used to load Electrical Idle III in the Tx driver of the PMA macro.
mpcs_rxidle_o	Out	NL	This port is used to signal the Electrical Idle condition detected by the PMA control logic. This signal is driven by mpcs_clkin_i.
mpcs_rxerr_i	In	NL	This signal is used to report to PMA control logic that error data is detected by the PCS logic. Asserting this signal leads CDR PLL switch back to the process of frequency lock acquisition.
mpcs_fomreq_i	In	NL	This signal is used to request a Figure of Merit (FOM) evaluation.
mpcs_fomack_o	Out	NL	This signal is used to handshake the FOM evaluation request in MPCS mode. It is asserted for a single clock cycle by the PMA Controller. This signal is synchronous to mpcs_tx_out_clk_o.
mpcs_fomrslt_o	Out	8*NL	This signal is the evaluated FOM result. This signal is synchronous to mpcs_tx_out_clk_o.
mpcs_rate_i	In	2*NL	MPCS rates: • 2'b10 – Rate2 • 2'b01 – Rate1 • 2'b00 – Rate0
mpcs_speed_o	Out	2*NL	MPCS current speeds: • 2'b10 – Rate2 • 2'b01 – Rate1 • 2'b00 – Rate0
mpcs_txval_i	In	NL	PHY transmit valid: this signal is used to transmit valid data. If deasserted, the PMA macro is put in Electrical Idle 1. It can be used for protocol requiring Electrical Idle SATA and must also be deasserted as long as mpcs_ready_o is not asserted. This signal is also required to be generated one clock cycle earlier than the corresponding mpcs_tx_data_i signals. mpcs_txval_i = 0 causes the PMA Tx driver to generate the Electrical Idle condition 22 tx_pcs_clk cycles later. If mpcs_txval_i = 1, this operation causes the PMA Tx driver to exit Electrical Idle condition 22 tx_pcs_clk cycles later.

Port Name	I/O	Width	Description
mpcs_rxval_o	Out	NL	PHY receive valid: this signal is used to signal receive valid data. It corresponds to the two condition completed by the PMA control logic: • Receiver detects incoming data (not in Electrical Idle); • CDR PLL is locked to input bitstream in fine grain state.
mpcs_phyrdy_o	Out	NL	When asserted, this signal tells the user that the PHY is ready to transmit while using mpcs_txval_i. This signal is deasserted for following conditions: • During calibration. • When Tx common mode voltage is not guaranteed. • PHY in low power states. • During rate change negotiation. This signal is driven by mpcs_clkin_i.
mpcs_ready_o	Out	NL	PHY ready: this signal is asserted when the PHY has completed the calibration sequence for each specific lane. This signal is driven by mpcs_clkin_i.
mpcs_rxoob_i	In	NL	This signal configures the activity detector to detect out-of-band (OOB) accurately.
mpcs_txdeemp_i	In	NL	When asserted high, programmed de-emphasis is applied to the transmitter driver.
mpcs_pwrst_o	Out	2*NL	This signal is used to report the PHY current power state. ***This signal is driven by*** mpcs_clkin_i.
mpcs_skipbit_i	In	NL	The rising edge of this signal triggers the input parallel data to shift one UI inside PMA.

5.8.2. EPCS Interface

EPCS interface is accessible when SerDes/PCS is configured as PMA only mode. EPCS mode is designed for applications that do not need encoding and decoding. Table 5.9 shows the detailed EPCS interface descriptions. All the signals listed in this table are per lane, and NL means the number of lanes.

Table 5.9. EPCS Interface

Port Name	I/O	Width	Description
Clock and Reset Signals			
epcs_rx_usr_clk_i	In	NL	User interface Rx clock input.
epcs_tx_usr_clk_i	In	NL	User interface Tx clock input.
epcs_tx_pcs_rstn_i	In	NL	Active low signal used to reset the Tx path of MPCS module.
epcs_rx_pcs_rstn_i	In	NL	Active low signal used to reset the Rx path of MPCS module.
epcs_rx_clk_o	Out	NL	PCS Rx output clock.
epcs_tx_clk_o	Out	NL	PCS Tx output clock.
epcs_rstn_i	In	NL	Fundamental reset that triggers PCS auto calibration.
epcs_clkin_i	In	NL	This low-speed clock drives all calibration logic inside PMA Controller. The recommended frequency range is 100-300 MHz. Note that this clock should be stable and continuous after power on.
Tx/Rx FIFO Signals			
epcs_txdata_i	In	80*NL	For the signal mapping of this port, refer to Table 5.12.
epcs_rxdata_o	Out	80*NL	For the signal mapping of this port, refer to Table 5.12.
PMA Control and Status Signals			
epcs_pwrdn_i	In	2*NL	This signal is used to put the PMA in powerdown mode. This signal has only three states. This signal is required to be clocked on mpcs_clkin_i.

Port Name	I/O	Width	Description
			• 2'b11 – deep low-power state. • 2'b10 – low-power state. • 2'b00 – operational state.
epcs_txhiz_i	In	NL	This signal is used to load Electrical Idle III in the Tx driver of the PMA macro.
epcs_rxidle_o	Out	NL	This port is used to signal the Electrical Idle condition detected by the PMA control logic. This signal is driven by epcs_clkin_i.
epcs_rxerr_i	In	NL	This signal is used to report to PMA control logic where error data is detected by the PCS logic. Asserting this signal leads CDR PLL switch back to the process of frequency lock acquisition.
epcs_fomreq_i	In	NL	This signal is used to request the FOM evaluation.
epcs_fomack_o	Out	NL	This signal is used to handshake the FOM evaluation request in EPCS mode. It is asserted for a single clock cycle by the PMA Controller. This signal is synchronous to epcs_tx_out_clk_o.
epcs_fomrslt_o	Out	8*NL	This signal is the evaluated FOM result. This signal is synchronous to epcs_tx_out_clk_o.
epcs_rate_i	In	2*NL	EPCS rates: • 2'b10 – Rate2 • 2'b01 – Rate1 • 2'b00 – Rate0
epcs_speed_o	Out	2*NL	EPCS current speeds: • 2'b10 – Rate2 • 2'b01 – Rate1 • 2'b00 – Rate0
epcs_txval_i	In	NL	PHY transmit valid. This signal is used to transmit valid data. If deasserted, the PMA macro is put in Electrical Idle 1. It can be used for protocol requiring Electrical Idle, SATA, and must also be deasserted as long as epcs_ready_o is not asserted. This signal is also required to be generated one clock cycle earlier than those corresponding epcs_tx_data_i signals. epcs_txval_i = 0 causes the PMA Tx driver to generate the Electrical Idle condition after 22 tx_pcs_clk cycles. If epcs_txval_i = 1, this operation causes the PMA Tx driver to exit Electrical Idle condition 22 tx_pcs_clk cycles later.
epcs_rxval_o	Out	NL	PHY receive valid: this signal is used to signal receive valid data. It corresponds to the two-condition completed by the PMA control logic: • Receiver detects incoming data (not in Electrical Idle); • CDR PLL is locked to input bitstream in fine grain state.
epcs_phyrdy_o	Out	NL	When asserted, this signal tells the user that the PHY is ready to transmit while using epcs_txval_i. This signal is deasserted for following conditions: • during calibration. • when Tx common mode voltage is not guaranteed. • PHY in low power states. • during rate change negotiation. This signal is driven by mpcs_clkin_i.
epcs_ready_o	Out	NL	PHY ready. This signal is used to release the reset for the external PCS and controller, and to start transmitting data to PMA. This signal is driven by epcs_clkin_i.
epcs_rxoob_i	In	NL	This signal configures the activity detector to detect out-of-band (OOB) accurately.
epcs_txdeemp_i	In	NL	When asserted high, programmed de-emphasis is applied to the

Port Name	I/O	Width	Description
			transmitter driver.
epcs_pwrst_o	Out	2*NL	This signal is used to report the PHY current power state. This signal is driven by epcs_clkin_i.
epcs_skipbit_i	In	NL	The rising edge of this signal triggers the input parallel data to shift one UI inside PMA.

5.8.3. LMMI Interface

LMMI is a simple memory mapped interface proposed by Lattice, and is mainly used for register access. Table 5.10 shows the detailed LMMI interface descriptions. All these signals are per lane, and NL means the number of lanes.

Table 5.10. LMMI Interface

Port Name	I/O	Width	Description
lmmi_clk_i	In	NL	LMMI clock input.
lmmi_resetn_i	In	NL	Active low LMMI reset.
lmmi_request_i	In	NL	Start transaction.
lmmi_wr_rdn_i	In	NL	Write = HIGH, Read = LOW
lmmi_offset_i	In	9*NL	Address/Offset, the highest bit is used to select register space. • 1'b1 – MPCS register space. • 1'b0 – PMA register space.
lmmi_wdata_i	In	8*NL	Write data.
lmmi_rdata_valid_o	Out	NL	Valid data indicator.
lmmi_ready_o	Out	NL	Slave ready signal.
lmmi_rdata_o	Out	8*NL	Read data.

5.8.4. Other Signals

Table 5.11 shows signals other than MPCS, EPCS, PIPE and LMMI interfaces per quad. NL means the number of lanes. For detailed descriptions about Quad-to-Quad signals, refer to the Clocks and Reset section.

Table 5.11. Other Signals

Port Name	I/O	Width	Description
PMA Serial I/O[1]			
sdq_refclkp_i	In	1	Reference Clock of SerDes PLL in one Quad.
sdq_refclkn_i	In	1	Reference Clock of SerDes PLL in one Quad.
sd[n]rxp_i	In	NL	Analog Rx differential IO.
sd[n]rxn_i	In	NL	Analog Rx differential IO.
sd[n]txp_o	Out	NL	Analog Tx differential IO.
sd[n]txn_o	Out	NL	Analog Tx differential IO.
sd[n]_rext_i	In	NL	External Resistance.
sd[n]_refret_i	In	NL	Analog reference return for PMA PLL.
Reference Clock			
sd_ext_0_refclk_i	In	1	Reference Clock from SD_EXT0_REFCLKP, SD_EXT0_REFCLKN.
sd_ext_1_refclk_i	In	1	Reference Clock from SD_EXT1_REFCLKP, SD_EXT1_REFCLKN.
pll_0_refclk_i	In	1	Reference Clock from left top GPLL.
pll_1_refclk_i	In	1	Reference Clock from right top GPLL.
sd_pll_refclk_i	In	1	Reference Clock from FPGA PCLK, only for test purpose.
diffioclksel_i	In	1	Dynamic clock source selection: • 1'b1 – sd_ext_1_refclk_i. • 1'b0 – sd_ext_0_refclk_i.
clksel_i	In	2	Dynamic clock source selection: • 2'b11 – sd_pll_refclk_i. • 2'b10 – sd_ext_0_refclk_i or sd_ext_1_refclk_i. • 2'b01 – pll_1_refclk_i. • 2'b00 – pll_0_refclk_i.
use_refmux_i	In	1	Dynamic clock source selection: • 1'b1 – clock from PCSREFMUX output. • 1'b0 – clock from per quad source (sdq_refclkp_i, sdq_refclkn_i).
Quad-Quad Interface			
tx_lalign_clk_out_o	Out	1	The outputted clock for sharing among Quads are hooked up during Quad integration.
rx_lalign_clk_out_o	Out	1	The outputted clock for sharing among Quads are hooked up during Quad integration.
tx_lalign_clk_in_i	In	1	The input clock is shared by all channels within a Quad to implement lane alignment.
rx_lalign_clk_in_i	In	1	The input clock is shared by all channels within a Quad to implement lane alignment.
lalign_out_up_o	Out	8	The connection signals between Quads for lane alignment across Quad boundary.
lalign_in_up_i	In	8	The connection signals between Quads for lane alignment across Quad boundary.
lalign_out_down_o	Out	8	The connection signals between Quads for lane alignment across Quad boundary.
lalign_in_down_i	In	8	The connection signals between Quads for lane alignment across Quad boundary.

Port Name	I/O	Width	Description
JTAG Interface			
acjtag_mode_i	In	1	When asserted, this signal activates the ACJTAG controller of the PMA control logic which now controls the PMA hard macro. The PMA hard macro is thus disconnected from the PMA control logic: This signal is used for two purposes: • Select the multiplexer between ACJTAG controller and functional logic at the PMA interface directly. • Put out of reset the ACJTAG controller. This signal is used as reset input for the embedded ACJTAG controller of the PMA.
acjtag_enable_i	In	1	This signal configures the PMA in ACTAG test mode. By default, it resets the Tx and Rx driver and receiver followed by loading the driver and receiver with default settings. The PMA receiver is by default in DC test mode and the transmitter is driving 0.
acjtag_acmode_i	In	1	When acjtag_enable_i == 1'b1, this signal selects either AC mode (1'b1) or DC mode (1'b0).
acjtag_drive1_i	In	1	When acjtag_enable_i == 1'b1, this signal selects the differential value the transmitter drives.
acjtag_highz_i	In	1	When acjtag_mode_i == 1'b1, assertion of this signal puts the PMA driver in high impedance.
acjtagpout_i	In	1	ACJTAG Output Data.
acjtagnout_i	In	1	ACJTAG Output Data.

Note:

1. For multiple ports, [n] indicates lane/channel number.

5.8.5. Data Bus Sharing and Mapping

Table 5.12 defines the detailed usage for different bits of MPCS/EPCS data bus.

Table 5.12. Data Bus Sharing and Mapping

MPCS Module Port	MPCS Mode Protocol != "10GE"	MPCS Mode Protocol == "10GE"	EPCS Mode
Tx Path			
mpcs_tx_ch_din_i[39:0]/ epcs_txdata_i[39:0]	tx_data[39:0] 4-byte output data • bit[39:30]: byte_3 • bit[39]: cordisp bit or data bit[9] when being used in 10b mode. • bit[38]: control character when being used in 8B/10B mode; or data bit[8] when being used in 10b mode. • bit[37:30]: data bit[7:0] • bit[29:20]: byte_2 • bit[19:10]: byte_1 • bit[9:0]: byte_0	tx_data_64b[39:0] Input data.	40-bit input data.
mpcs_tx_ch_din_i[43:40]/! epcs_txdata_i[43:40]	tx_frcdisp[3:0] • 1: use "force running disparity" mode; in this mode, the running disparity is indicated by "tx_dispval" signal. • 0: do not use "force running disparity" mode.	tx_data_64b[43:40] Input data.	—

MPCS Module Port	MPCS Mode Protocol != “10GE”	MPCS Mode Protocol == “10GE”	EPCS Mode
mpcs_tx_ch_din_i[47:44]/ epcs_txdata_i[47:44]	tx_dispval[3:0] In “force running disparity” mode, this signal indicates the forced running disparity: • 1: force positive running disparity. • 0: force negative running disparity. If not in “force running disparity” mode): • 1: revert running disparity. • 0: use running disparity calculated by 8B/10B encoder.	tx_data_64b[47:44] Input data.	—
mpcs_tx_ch_din_i[51:48]/ epcs_txdata_i[51:48]	tx_frcdata[3:0] • 1: force 8B/10B encoder to output “tx_data”. • 0: do not force the output data of 8B/10B encoder.	tx_data_64b[51:48] Input data.	—
mpcs_tx_ch_din_i[63:52]/ epcs_txdata_i[63:52]	—	tx_data_64b[63:52] Input data.	—
mpcs_tx_ch_din_i[71:64]/ epcs_txdata_i[71:64]	—	tx_control[7:0]/ tx_header[1:0] 8-bit control indication • bit[0] is the control signal for tx_data_64b[7:0]. • bit[1] is the control signal for tx_data_64b[15:8]. • … • bit[7] is the control signal for tx_data_64b[63:56]. bit[1:0] of this signal can also be used as 2-bit block header. If "force data" signal is asserted ("tx_frcpkt" is driven high), or if 64B/66B encoder is bypassed, bit[1:0] of this signal, along with the 64-bit payload (coming from "tx_data_64b"), forms a 66-bit block which is usually generated by the Encoder. In this case, bit[7:2] of this signal are unused.	—
mpcs_tx_ch_din_i[72]/ epcs_txdata_i[72]	—	tx_frcpkt When this signal is asserted high, the 64-bit data (tx_data_64b) and 2-bit header (tx_control[1:0]) on the same clock cycle is used to drive TX Gear Box directly.	—

MPCS Module Port	MPCS Mode Protocol != “10GE”	MPCS Mode Protocol == “10GE”	EPCS Mode
		The Encoder and Scrambler are bypassed in this case. Once de-asserted, the 64-bit data and 8-bit control are encoded and scrambled before they go to TX Gear Box. • 1'b1: on the same cycle, use 64-bit data and 2-bit header coming from user logic to feed the Gear Box. • 1'b0: process the 64-bit data and 8-bit control as normal (go through encoder, scrambler, and gear box).	
mpcs_tx_ch_din_i[78:73]/ epcs_txdata_i[78:73]	—	—	—
mpcs_tx_ch_din_i[79]/ epcs_txdata_i[79]	—	tx_fifo_wr • 1: write 64-bit data and 8-bit control to TX FIFO. User logic should monitor the FIFO status and properly control the writing operation to avoid the FIFO overflow or underflow.	—
mpcs_tx_fifo_st[3:0]	tx_fifo_status[3:0] • bit[0]: FIFO is almost empty. • bit[1]: FIFO is almost full. • bit[2]: FIFO underflow. • bit[3]: FIFO overflow.	tx_fifo_status[3:0] • bit[0]: FIFO is almost empty. • bit[1]: FIFO is almost full. • bit[2]: FIFO underflow. • bit[3]: FIFO overflow.	tx_fifo_status[3:0] • bit[0]: FIFO is almost empty. • bit[1]: FIFO is almost full. • bit[2]: FIFO underflow. • bit[3]: FIFO overflow.
Rx Path			
mpcs_rx_ch_dout_o[39:0]/ epcs_rxdata_o[39:0]	rx_data[39:0] 4-byte input data • bit[39:30]: byte_3 • bit[39]: rundisp bit or data bit[9] when being used in 10b mode • bit[38]: control character when being used in 8B/10B mode; or data bit[8] when being used in 10b mode • bit[37:30]: data bit[7:0] • bit[29:20]: byte_2 • bit[19:10]: byte_1 • bit[9:0]: byte_0	rx_data_64b[39:0] Input data.	40-bit output data.
mpcs_rx_ch_dout_o[43:40]/ epcs_rxdata_o[43:40]	rx_errdisp[3:0] • 1: disparity error. • 0: no error.	rx_data_64b[43:40] Input data.	—
mpcs_rx_ch_dout_o[47:44]/	rx_errcode[3:0]	rx_data_64b[47:44]	—

MPCS Module Port	MPCS Mode Protocol != “10GE”	MPCS Mode Protocol == “10GE”	EPCS Mode
epcs_rxdata_o[47:44]	• 1: invalid code group. • 0: valid code group.	Input data.	
mpcs_rx_ch_dout_o[51:48]/ epcs_rxdata_o[51:48]	rx_skp_add[3:0] • 1: added byte indication given by Elastic Buffer. • 0: normal byte.	rx_data_64b[51:48] Input data.	—
mpcs_rx_ch_dout_o[55:52]/ epcs_rxdata_o[55:52]	rx_skp_del[3:0] • 1: deleted byte indication given by Elastic Buffer. • 0: normal byte.	rx_data_64b[55:52] Input data.	—
mpcs_rx_ch_dout_o[63:56]/ epcs_rxdata_o[63:56]	—	rx_data_64b[63:56] Input data.	—
mpcs_rx_ch_dout_o[71:64]/ epcs_rxdata_o[71:64]	—	rx_control[7:0]/ rx_header[1:0] 8-bit control indication. • bit[0] is the control signal for rx_data_64b[7:0]; • bit[1] is the control signal for rx_data_64b[15:8]; • … • bit[7] is the control signal for rx_data_64b[63:56]. In 64B/66B decoder bypass mode, the bit [1:0] of this signal is used to carry "rx_header[1:0]" of a 66-bit block. Other bits are not used in this mode.	—
mpcs_rx_ch_dout_o[73:72]/ epcs_rxdata_o[73:72]	—	rx_fifo_add[1:0] The indication of insertion for clock frequency difference compensation. • bit[1] corresponds to 4-byte idles which are carried by bit[63:32] of data bus (rx_data_64b). • bit[0] corresponds to 4-byte idles which are carried by bit[31:0] of data bus (rx_data_64b).	—
mpcs_rx_ch_dout_o[75:74]/ epcs_rxdata_o[75:74]	—	rx_fifo_del[1:0] The indication of deletion for clock frequency difference compensation. • bit[1] corresponds to 4-byte idles or sequence ordered-set which are carried by bit[63:32] of data bus (rx_data_64b).	—

MPCS Module Port	MPCS Mode Protocol != “10GE”	MPCS Mode Protocol == “10GE”	EPCS Mode
		• bit[0] corresponds to 4-byte idles or sequence ordered-set which are carried by bit[31:0] of data bus (rx_data_64b).	
mpcs_rx_ch_dout_o[78:76]/ epcs_rxdata_o[78:76]	—	—	—
mpcs_rx_ch_dout_o[79]/ epcs_rxdata_o[79]	—	rx_data_valid • 1: the output data is valid. • 0: no valid data on the output of RX FIFO.	—
mpcs_rx_fifo_st_o[3:0]	rx_fifo_status[3:0] • bit[0]: FIFO is almost empty. • bit[1]: FIFO is almost full. • bit[2]: FIFO underflow. • bit[3]: FIFO overflow.	rx_fifo_status[3:0] • bit[0]: Reserved. • bit[1]: Reserved. • bit[2]: FIFO underflow. • bit[3]: FIFO overflow.	rx_fifo_status[3:0] • bit[0]: Reserved. • bit[1]: Reserved. • bit[2]: FIFO underflow. • bit[3]: FIFO overflow.

5.9. Control and Status Signals

Table 5.14 describes the control and status signals for 8B/10B PCS.

Table 5.15 describes the control and status signals for 64B/66B PCS.

Table 5.14. Control and Status Signals Functions (8B/10B PCS)

Signal Name	Description
Transmit Control Signals	
tx_data_8b[39|29|19|9]	The cordisp bit, used in GigE protocol to replace /I2/ symbol with /I1/, if the current running disparity is detected as “positive”.
tx_data_8b[38|28|18|8]	Active-high control character indicator.
tx_frcdisp[3:0]	Active-high signal which instructs the PCS to accept disparity value from tx_disval[3:0] input.
tx_dispval[3:0]	Disparity value supplied from FPGA logic. Valid when tx_frcdisp[3:0] is high.
tx_frcdata[3:0]	Force 8B/10B encoder to output tx_data_8b data. When this signal is high, the corresponding channel forces the 8B/10B encoder to output data from the input directly. In other words, the output is the same as input. Note that the input of 8B/10B encoder is always 10 bits, the highest 2 bits are unused when 8B/10B encoder is enabled and tx_frcdata[3:0] is low. This functionality is mainly implemented for applications like SLVS-EC that needs the ability to send illegal 10b codes.
tx_fifo_status[3:0]	Tx FIFO status: • bit[3] – FIFO overflow. • bit[2] – FIFO underflow. • bit[1] – FIFO is almost full. • bit[0] – FIFO is almost empty.
Receive Status Signals	
rx_data_8b[39|29|19|9]	The rundisp bit. Shows the running disparity status.
rx_data_8b[38|28|18|8]	Active-high control character indicator.
rx_errdisp[3:0]	Active-high signal driven by the PCS to indicate a disparity error was detected with the associated data.
rx_errcode[3:0]	Code violation signal to indicate an error was detected with the associated data.
rx_skp_add[3:0]	SKP added indication given by Elastic Buffer.
rx_skp_del[3:0]	SKP deleted indication given by Elastic Buffer.
rx_fifo_status[3:0]	Rx FIFO status: • bit[3] – FIFO overflow. • bit[2] – FIFO underflow. • bit[1] – FIFO is almost full. • bit[0] – FIFO is almost empty.

Table 5.15. Control and Status Signals Functions (64B/66B PCS)

Signal Name	Description
Transmit Control Signals	
tx_control[7:0]/ tx_header[1:0]	8-bit control indication: • bit[0] – the control signal for tx_data_64b[7:0]. • bit[1] – the control signal for tx_data_64b[15:8]. • … • bit[7] – the control signal for tx_data_64b[63:56]. bit[1:0] of this signal can also be used as 2-bit block header. If "force data" signal is asserted, or if 64B/66B encoder is bypassed, bit[1:0] of this signal, along with the 64-bit payload forms a 66-bit block which is usually generated by the 64B/66B encoder. In this case, bit[7:2] of this signal are not used.
tx_frcpkt	When this signal is asserted high, the 64-bit data (tx_data_64b) and 2-bit header (tx_control[1:0]) on the same clock cycle are used to drive Tx Gear Box directly. The Encoder and Scrambler are bypassed in this case. Once de-asserted, the 64-bit data and 8-bit control are encoded and scrambled before they go to Tx Gear Box. • 1'b1 – on the same cycle, use 64-bit data and 2-bit header coming from user logic to feed the Gear Box. • 1'b0 – process the 64-bit data and 8-bit control as normal (go through encoder, scrambler, and gear box).
tx_fifo_wr	Write 64-bit data and 8-bit control to Tx FIFO. User logic should monitor the FIFO status and properly control the writing operation to avoid the FIFO overflow or underflow.
tx_fifo_status[3:0]	Tx FIFO status: • bit[0] – FIFO is almost empty. • bit[1] – FIFO is almost full. • bit[2] – FIFO underflow. • bit[3] – FIFO overflow.
Receive Status Signals	
rx_control[7:0]/ rx_header[1:0]	8-bit control indication: • bit[0] – the control signal for rx_data_64b[7:0]. • bit[1] – the control signal for rx_data_64b[15:8]. • … • bit[7] – the control signal for rx_data_64b[63:56]. In 64B/66B decoder bypass mode, bit[1:0] of this signal is used to carry rx_header[1:0] of a 66-bit block. Other bits are not used in this mode.
rx_fifo_add[1:0]	The indication of insertion for clock frequency difference compensation: • bit[1] – corresponds to 4-byte idles which are carried by bit[63:32] of data bus (rx_data_64b). • bit[0] – corresponds to 4-byte idles which are carried by bit[31:0] of data bus (rx_data_64b).
rx_fifo_del[1:0]	The indication of deletion for clock frequency difference compensation: • bit[1] – corresponds to 4-byte idles or sequence ordered-set which are carried by bit[63:32] of data bus (rx_data_64b). • bit[0] – corresponds to 4-byte idles or sequence ordered-set which are carried by bit[31:0] of data bus (rx_data_64b).
rx_data_valid	Output indicator: • 1'b1 – the output data is valid. • 1'b0 – no valid data on the output of Rx FIFO.
rx_fifo_status[3:0]	Rx FIFO status: • bit[0] – FIFO is almost empty. • bit[1] – FIFO is almost full. • bit[2] – FIFO underflow. • bit[3] – FIFO overflow.

5.10. Detailed Channel Block Diagram

Detailed block diagrams in this section are intended to show the major functionality in a single channel of the CertusPro-NX SerDes/PCS. The user can find all the major blocks, clock, and data flow in these diagrams. PCI Express PCS channel is hidden in all the three modes, considering that this channel can only be used by PCI Express protocol.

5.10.1. MPCS-8B/10B Mode

Figure 5.12 shows the detailed channel block diagram of MPCS-8B/10B mode. In this mode, only 8B/10B PCS channel is expanded. 64B/66B PCS channel and PMA Only channel are hidden.

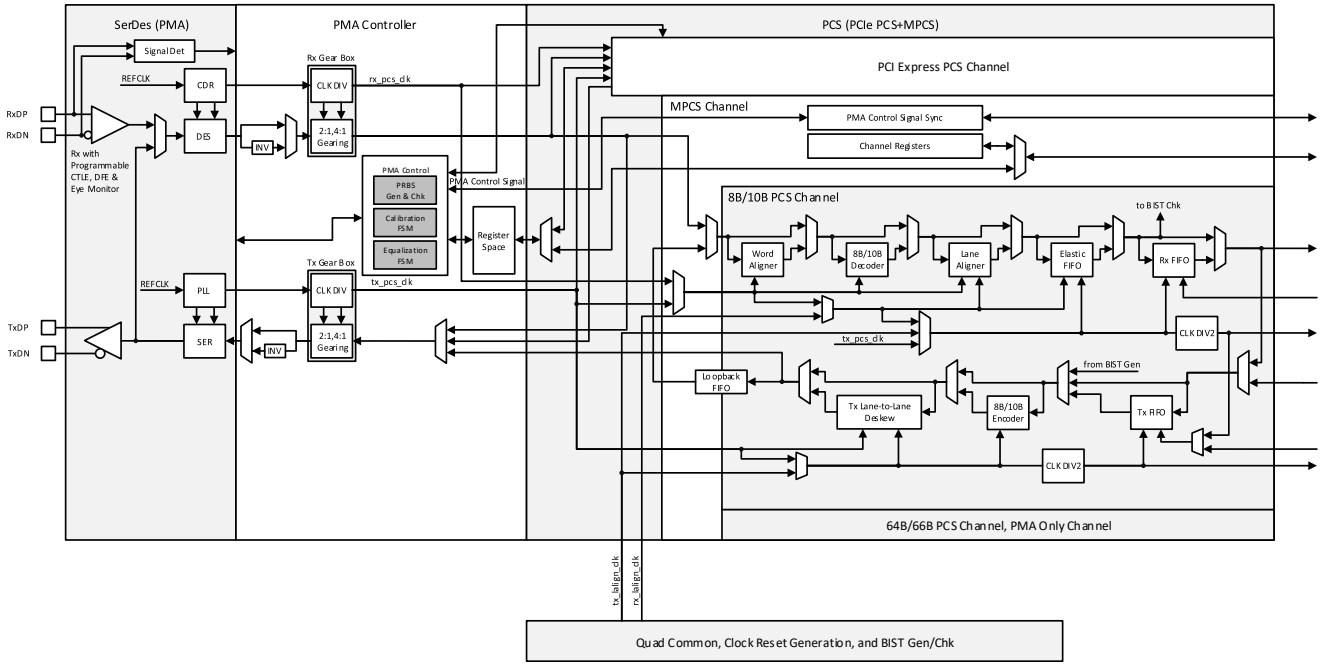

Figure 5.12. Detailed Channel Block Diagram of MPCS-8B/10B Mode

5.10.2. MPCS-64B/66B Mode

Figure 5.13 shows the detailed channel block diagram in PMCS-64B/66B mode. In this mode, only 64B/66B PCS channel is expanded, 8B/10B PCS channel and PMA Only channel are hidden.

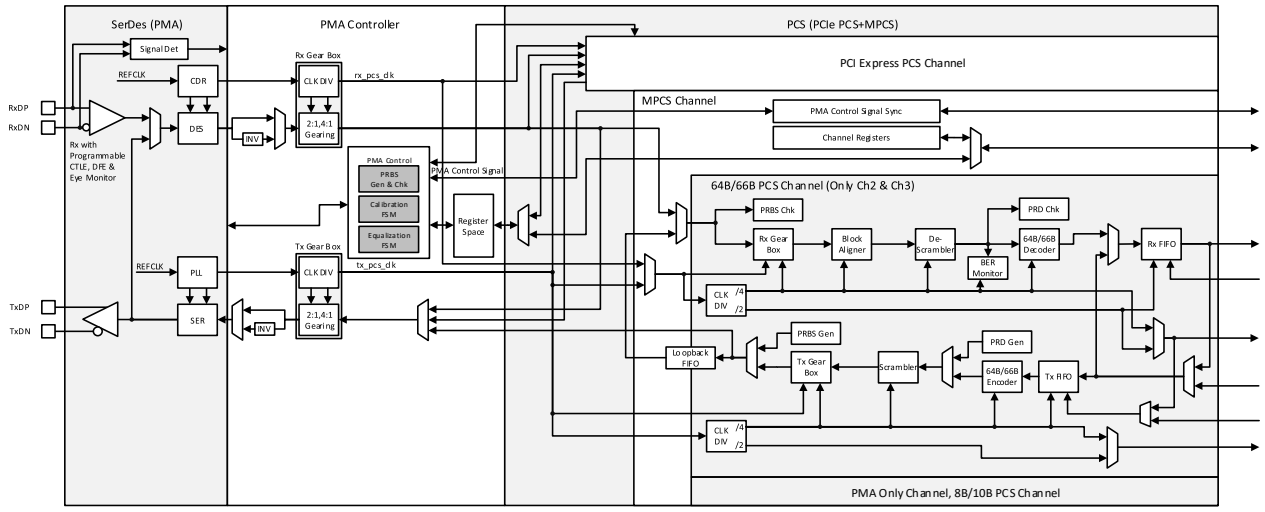

Figure 5.13. Detailed Channel Block Diagram of MPCS-64B/66B Mode

Note that only channel 2 and channel 3 of each MPCS quad contains 64B/66B PCS channel.

5.10.3. MPCS-PMA Only Mode

Figure 5.14 shows the detailed channel block diagram in PMCS-PMA Only mode. In this mode, only PMA Only channel is expanded. 8B/10B PCS channel and 64B/66B PCS channel are hidden.

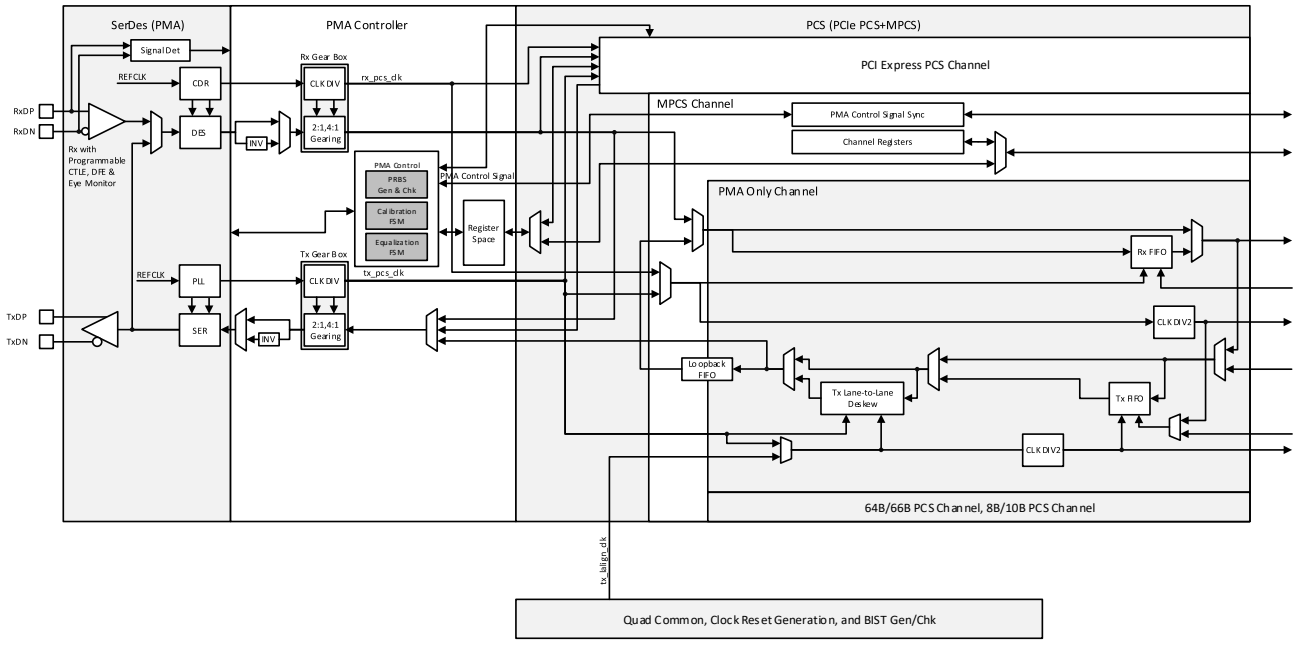

Figure 5.14. Detailed Channel Block Diagram of MPCS-PMA Only Mode

6. SerDes/PCS Function Description

CertusPro-NX devices have one to three Quads of embedded SerDes/PCS logic. Each Quad, in turn, supports four independent full-duplex data channels. A single channel can support a data link, and each Quad can support up to four such channels.

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

6.1. SerDes (PMA)

Figure 6.1 shows a simplified functional block diagram of the SerDes (PMA) macro. Each of the SerDes macros includes three main sub-functions, Transmitter (Tx), Receiver (Rx), and PLL Clock blocks. The Tx block receives 5, 8, 10, 16, or 20 bits of transition-encoded data synchronous with a Tx clock, serializes it into a single stream of differential transmitted data and transmits to the lane. The transmitter supports multi-level output driver, multi-level transition emphasis, and multi-level Common Mode levels.

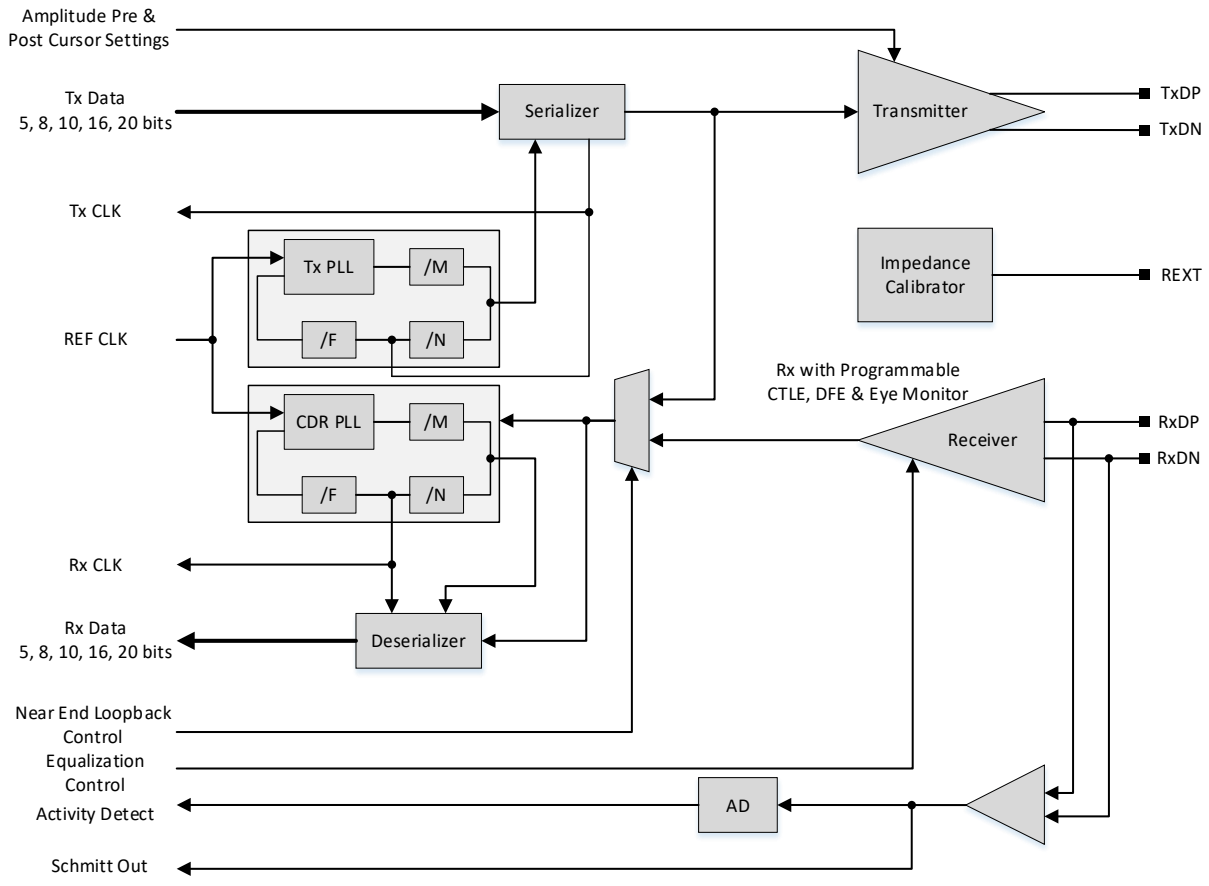

Figure 6.1. Simplified Block Diagram of the SerDes (PMA)

Output amplitude control range nominally spans from V_{diff_pp} = 0 to V_{diff_pp} = VCCSDx nominal, with a 1/128 typical resolution at full-scale. A 3-tap FIR based Tx equalizer is implemented to support de-emphasis and pre-shoot. The de-emphasis control range nominally spans from 0 dB to beyond 20 dB. Pre-cursor and post-cursor coefficients are negative, with each coefficient able to be quantized to typically 1/128 resolution with respect to normalized full-scale.

The Rx block receives the serialized data from the lane, de-serializes this into 5, 8, 10, 16, or 20 bits digital data and provides the de-serialized data and the recovered link clock. An Rx continuous time linear equalizer (CTLE) is provided with programmable gain and a single tap decision feedback equalization circuit (DFE) is provided with programmable tap weight.

The SerDes includes an eye monitor to map the post-equalized eye density. Eye X and Y coordinates can be controlled, and the error density can be calculated at each coordinate. The X coordinate can be set by a 6-bit bus in 1.5mV$_{diff}$ (minimum) increments with a separate sign bit, and the Y coordinate by a 6-bit bus in UI/64 increments.

The PLL Clock blocks generate the required Tx and Rx link clocks and high frequency internal clocks from the provided reference clock source, and contain all the control-feedback loops. Multiple single-lane macros may be formed into wider transmit and/or receive links with support from the PCS as required by the relevant standards.

6.1.1. AC Coupling

For most serial protocols, each PMA channel needs to be AC coupled externally. Suitable values for AC coupling capacitors have to be used to maximize link signal quality, and conform to respective standards specifications.

Take PCI Express as example, the AC coupling capacitors should be placed near Tx side (Figure 6.2).

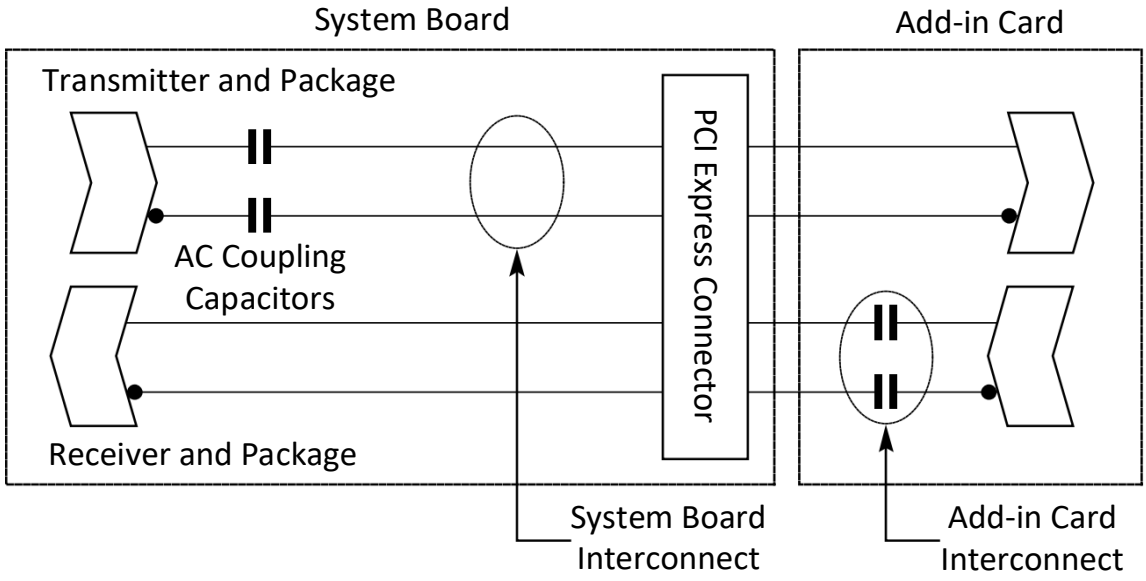

Figure 6.2. PCI Express AC Coupling Capacitors Location

6.1.2. PLL Clock Setting

Tx and Rx rates can be different frequencies but limited to certain allowed integer ratios. The architecture of Tx PLL and CDR PLL is similar. The user can configure PLL VCO frequency FVCO, PMA internal clock frequency F_{bit}, and PMA parallel output clock frequency FPMA by accessing related PMA registers.

The following equation shows the relationship between F_{VCO}, F_{bit}, F_{PMA}, and PMA reference clock frequency F_{Ref}. For detailed PLL setting for specific protocols, refer to the Clock Frequency section. M can be set as 1, 2, 4, and 8; F can be set as 1, 2, 3, 4, 5, and 6; N can be set as 5, 8, 10, 16, and 20.

$$F_{VCO} = F_{Ref} \times M \times N \times F$$
$$F_{bit} = F_{VCO}/M = F_{Ref} \times N \times F$$
$$F_{PMA} = F_{bit}/N = F_{Ref} \times F$$

Table 6.1 shows the operation range for PMA reference clock frequency F_{Ref}, PLL VCO frequency F_{VCO}, PMA internal clock frequency F_{bit}, and PMA parallel output clock frequency F_{PMA}.

Table 6.1. Operation Range for F_{Ref}, F_{VCO}, F_{bit}, and F_{PMA}

	Minimum (MHz)	Maximum (MHz)
PMA reference clock frequency F_{Ref}	74.25	162
PLL VCO frequency F_{VCO}	5000	10312.5
PMA internal clock frequency F_{bit}	625	10312.5
PMA parallel output clock frequency F_{PMA}	—	500 (644.53 for 64B/66B PCS only)

6.2. PMA Controller

PMA Controller block is shared by all modes, as shown in Figure 6.3. PCI Express PCS is connected to PMA Controller directly, while MPCS is connected to PMA Controller through External PCS (EPCS) interface. The Tx Gear Box and Rx Gear Box can be used as 2:1 gearing buffer, when the PMA data width is set to no more than 10 bits. PMA Control Logic controls the calibration and equalization procedures, and monitors the PMA status. PMA Control Logic also implements the PRBS generation and check blocks that support PRBS7, PRBS11, PRBS23 and PRBS31 patterns. Some registers inside PMA Controller can be accessed through Lattice Memory Mapped Interface (LMMI). PMA Control Signals are connected to both PCI Express PCS and MPCS. MPCS implements some synchronization logic between PMA Controller and FPGA Fabric, considering that PMA Control Signals from PMA Controller are asynchronous. The Data Polarity Invert modules between PMA and Tx/Rx Gear Box are implemented to invert the data polarity for Tx/Rx path, if necessary.

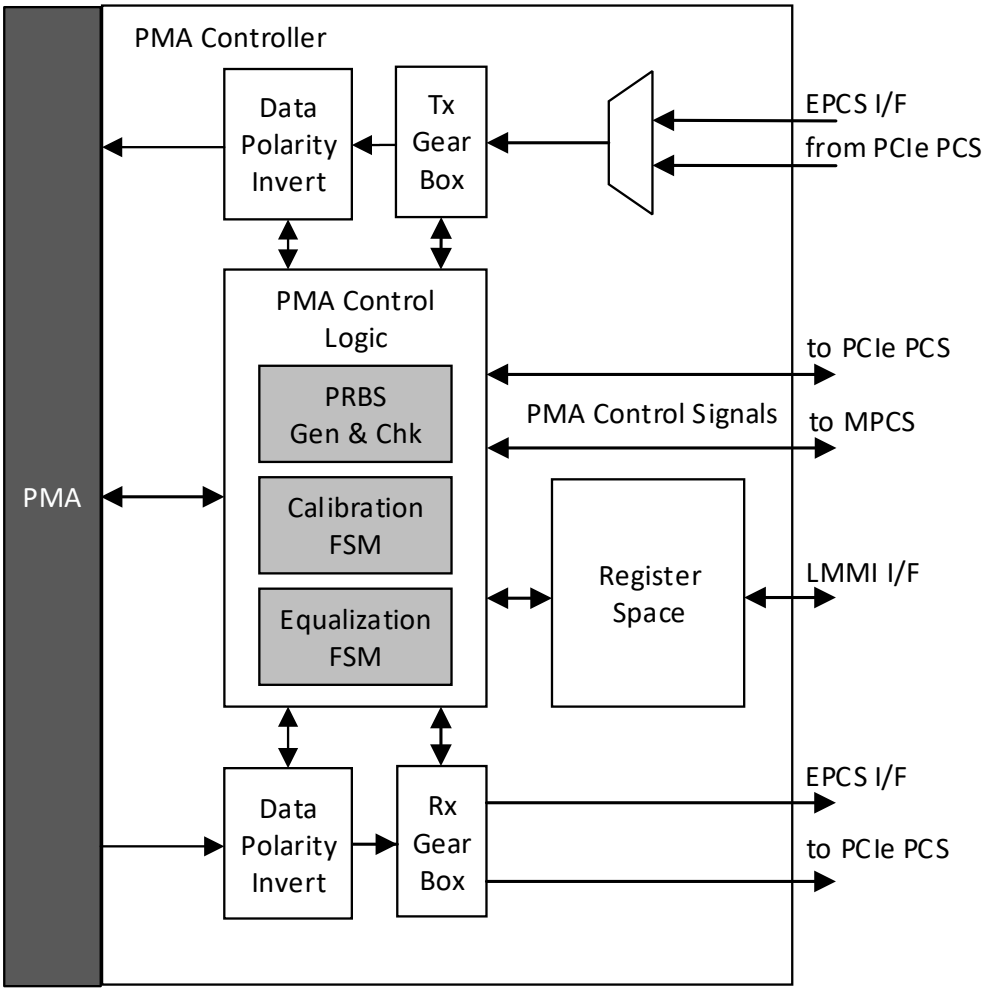

Figure 6.3. PMA Controller Block Diagram

6.3. PCI Express PCS

Figure 6.4 is the detailed block diagram of the PCI Express PCS. PCI Express PCS is designed for PCI Express protocol only. The Tx Gear Box and Rx Gear Box inside PCI Express PCS are designed for Gen3 speed only, which implements 2:1 gearing logic. 8B/10B Encoder and Decoder are designed for Gen1 and Gen2 speed. 128B/130B is designed for Gen3 speed only. Word Aligner module is for Gen1 and Gen2 speed, and Block Aligner is designed for Gen3 speed. Elastic Buffer (CTC FIFO) is implemented to compensate the clock difference between the recovered clock and the transmit clock. PCI Express related registers can be accessed through LMMI.

PCI Express PCS, PMA Controller and PMA constitute PCI Express Media Access Control (MAC) Layer which can be connected to PCI Express Link Layer Hard IP block through the PIPE interface.

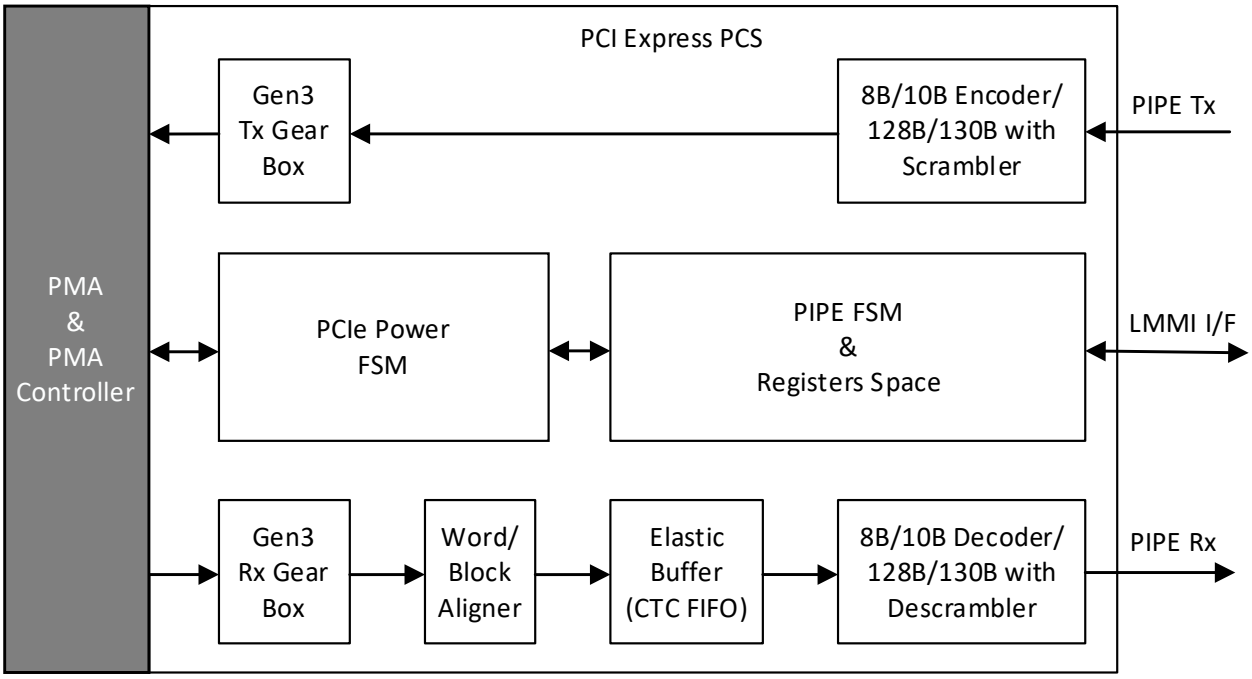

Figure 6.4. PCI Express PCS & PMA Controller Block Diagram

6.4. MPCS

6.4.1. 8B/10B PCS

Figure 6.5 shows the block diagram of the 8B/10B PCS inside MPCS channel. This block implements the most common functionalities required by serial protocols based on 8B/10B style PCS:

- 8B/10B Encoder and Decoder
- Tx Lane-to-lane Deskew
- Tx FIFO and Rx FIFO
- Word Aligner
- Rx Lane Aligner
- Elastic Buffer

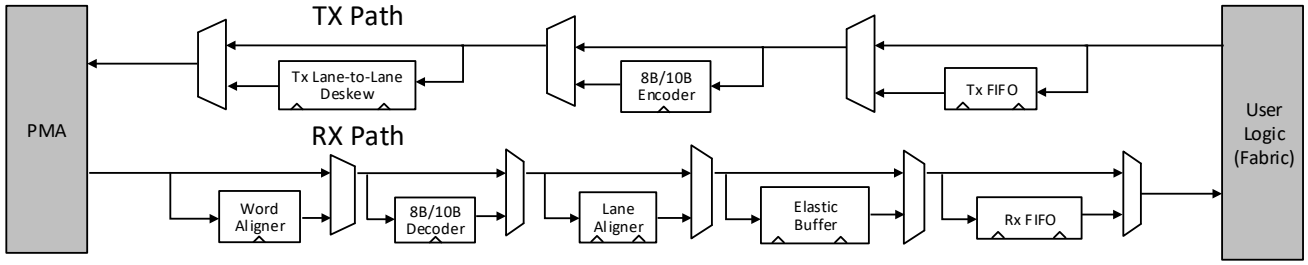

Figure 6.5. MPCS 8B/10B PCS Block Diagram

6.4.1.1. Data Bus Description

The bit mapping and transfer order of 40-bit Transmitter/Receiver data bus is described as follows. The data flowing over the data bus can be either 8b data or 10b data.

- 8b data: the 8-bit raw data before 8B/10B encoding or after 8B/10B decoding.
- 10b data: the 10-bit DC-balanced code before 8B/10B decoding or after 8B/10B encoding.

The MPCS uses either 1-byte or 2-byte width internal data path, depending on protocol data rate. Moreover, the MPCS-Fabric interface width can be configured as 1-byte, 2-byte, or 4-byte mode. Check the Clock Frequency section for more information about the relationship between data bus width and data rate.

- 1-byte mode: receive, transmit or process 1-byte data per clock cycle; in this mode, only bit[9:0] of the data bus are used to carry valid data; other 30 bits are left unused.
- 2-byte mode: receive, transmit or process 2-byte data per clock cycle; only bit[19:0] are valid in this mode.
- 4-byte mode: receive, transmit or process 4-byte data per clock cycle; all 40 bits are used in this mode.

The data flowing to/from fabric can be in 1-byte, 2-byte or 4-byte mode, but data inside MPCS are only in 1-byte, 2-byte mode. There is 2:1 gearing logic in MPCS Tx path to convert 4-byte data flowing from fabric to 2-byte data for MPCS internal processing. Similarly, a 1:2 gearing exists in Rx path to convert 2-byte data at high speed to 4-byte before they flow to fabric.

The bit mapping of Tx path data bus is described in Table 6.2. The bit mapping of Rx path data bus is described in Table 6.3. Transfer ordering represents the serial order of sending.

Table 6.2. Bit Mapping of Tx Data Bus

	8b Data	10b Data	Byte	1-byte Mode	2-byte Mode	4-byte Mode	Transfer Ordering
tx_data[39]	cordisp_3	data_3[9]	byte_3	unused	unused	used	last
tx_data[38]	ctrl_3	data_3[8]					
tx_data[37]	data_3[7]	data_3[7]					
tx_data[36]	data_3[6]	data_3[6]					
tx_data[35]	data_3[5]	data_3[5]					
tx_data[34]	data_3[4]	data_3[4]					
tx_data[33]	data_3[3]	data_3[3]					
tx_data[32]	data_3[2]	data_3[2]					
tx_data[31]	data_3[1]	data_3[1]					
tx_data[30]	data_3[0]	data_3[0]					^
tx_data[29]	cordisp_2	data_2[9]	byte_2				|
tx_data[28]	ctrl_2	data_2[8]					|
tx_data[27]	data_2[7]	data_2[7]					|
tx_data[26]	data_2[6]	data_2[6]					|
tx_data[25]	data_2[5]	data_2[5]					|
tx_data[24]	data_2[4]	data_2[4]					|
tx_data[23]	data_2[3]	data_2[3]					|
tx_data[22]	data_2[2]	data_2[2]					|
tx_data[21]	data_2[1]	data_2[1]					|
tx_data[20]	data_2[0]	data_2[0]					|
tx_data[19]	cordisp_1	data_1[9]	byte_1		used		|
tx_data[18]	ctrl_1	data_1[8]					|
tx_data[17]	data_1[7]	data_1[7]					|
tx_data[16]	data_1[6]	data_1[6]					|
tx_data[15]	data_1[5]	data_1[5]					|
tx_data[14]	data_1[4]	data_1[4]					|
tx_data[13]	data_1[3]	data_1[3]					|
tx_data[12]	data_1[2]	data_1[2]					|
tx_data[11]	data_1[1]	data_1[1]					|
tx_data[10]	data_1[0]	data_1[0]					|
tx_data[9]	cordisp_0	data_0[9]	byte_0	used			|
tx_data[8]	ctrl_0	data_0[8]					|
tx_data[7]	data_0[7]	data_0[7]					|
tx_data[6]	data_0[6]	data_0[6]					|
tx_data[5]	data_0[5]	data_0[5]					|
tx_data[4]	data_0[4]	data_0[4]					|
tx_data[3]	data_0[3]	data_0[3]					|
tx_data[2]	data_0[2]	data_0[2]					|
tx_data[1]	data_0[1]	data_0[1]					first
tx_data[0]	data_0[0]	data_0[0]					

Table 6.3. Bit Mapping of Rx Data Bus

	8b Data	10b Data	Byte	1-byte Mode	2-byte Mode	4-byte Mode	Transfer Ordering
rx_data[39]	rundisp_3	data_3[9]	byte_3	unused	unused	used	last
rx_data[38]	ctrl_3	data_3[8]					
rx_data[37]	data_3[7]	data_3[7]					
rx_data[36]	data_3[6]	data_3[6]					
rx_data[35]	data_3[5]	data_3[5]					
rx_data[34]	data_3[4]	data_3[4]					
rx_data[33]	data_3[3]	data_3[3]					
rx_data[32]	data_3[2]	data_3[2]					
rx_data[31]	data_3[1]	data_3[1]					
rx_data[30]	data_3[0]	data_3[0]					^
rx_data[29]	rundisp_2	data_2[9]	byte_2				|
rx_data[28]	ctrl_2	data_2[8]					|
rx_data[27]	data_2[7]	data_2[7]					|
rx_data[26]	data_2[6]	data_2[6]					|
rx_data[25]	data_2[5]	data_2[5]					|
rx_data[24]	data_2[4]	data_2[4]					|
rx_data[23]	data_2[3]	data_2[3]					|
rx_data[22]	data_2[2]	data_2[2]					|
rx_data[21]	data_2[1]	data_2[1]					|
rx_data[20]	data_2[0]	data_2[0]					|
rx_data[19]	rundisp_1	data_1[9]	byte_1		used		|
rx_data[18]	ctrl_1	data_1[8]					|
rx_data[17]	data_1[7]	data_1[7]					|
rx_data[16]	data_1[6]	data_1[6]					|
rx_data[15]	data_1[5]	data_1[5]					|
rx_data[14]	data_1[4]	data_1[4]					|
rx_data[13]	data_1[3]	data_1[3]					|
rx_data[12]	data_1[2]	data_1[2]					|
rx_data[11]	data_1[1]	data_1[1]					|
rx_data[10]	data_1[0]	data_1[0]					|
rx_data[9]	rundisp_0	data_0[9]	byte_0	used			|
rx_data[8]	ctrl_0	data_0[8]					|
rx_data[7]	data_0[7]	data_0[7]					|
rx_data[6]	data_0[6]	data_0[6]					|
rx_data[5]	data_0[5]	data_0[5]					|
rx_data[4]	data_0[4]	data_0[4]					|
rx_data[3]	data_0[3]	data_0[3]					|
rx_data[2]	data_0[2]	data_0[2]					|
rx_data[1]	data_0[1]	data_0[1]					first
rx_data[0]	data_0[0]	data_0[0]					

6.4.1.2. Rx Channel Data Transfer

Signals of Fabric-MPCS Rx channel interface fall into two categories:

- Signals “rx_errdisp”, “rx_errcode”, “skp_add” and “skp_del” are strictly synchronous to the data bus (“rx_data”). These signals indicate that “disparity error”, “code error”, “added SKIP pattern” and “deleted SKIP pattern” belong to relative byte of data, separately.
- Other signals are not aligned to a certain byte of data, for example, FIFO empty and full. However, these signals last at least one clock cycle so that user logic can correctly sample them.

6.4.1.3. Tx Channel Data Transfer

Signals of Fabric-MPCS Tx channel interface fall into two categories:

- Signals “tx_frcdisp”, “tx_dispval”, and “tx_frcdata” are strictly synchronous to the data bus (“tx_data”). These signals are used to force disparity, reverse disparity, and force data to the exact corresponding byte of data.
- Other signals are not aligned to a certain byte of data, for example, FIFO empty and full. However, these signals last at least one clock cycle so that user logic can correctly sample them.

6.4.1.4. Function Block Description

Tx FIFO

The Tx FIFO module serves two purposes:

- Provide user logic input data with 2:1 gearing to internal MPCS data path.
- Clock phase compensation FIFO to ease MPCS-Fabric interface timing closure.

The 2:1 gearing can be optionally enabled to convert 4-byte data to 2-byte. Figure 6.6 shows how this conversion proceeds.

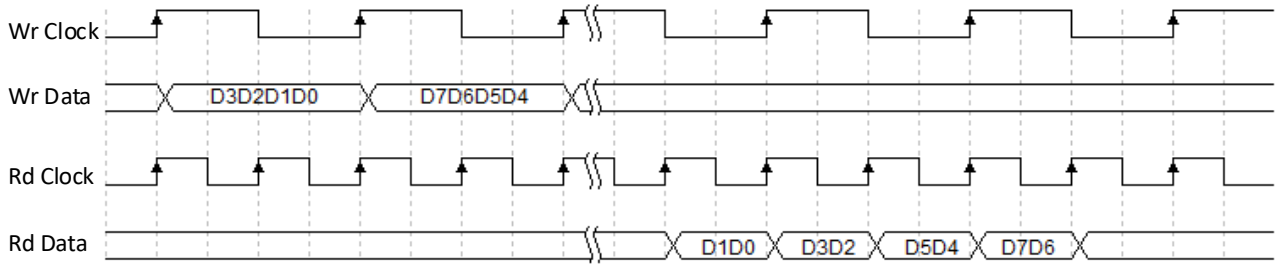

Figure 6.6. Tx Gearing Case I

The 2:1 gearing logic can also be used to convert 2-byte data to 1-byte. Figure 6.7 shows how this conversion proceeds.

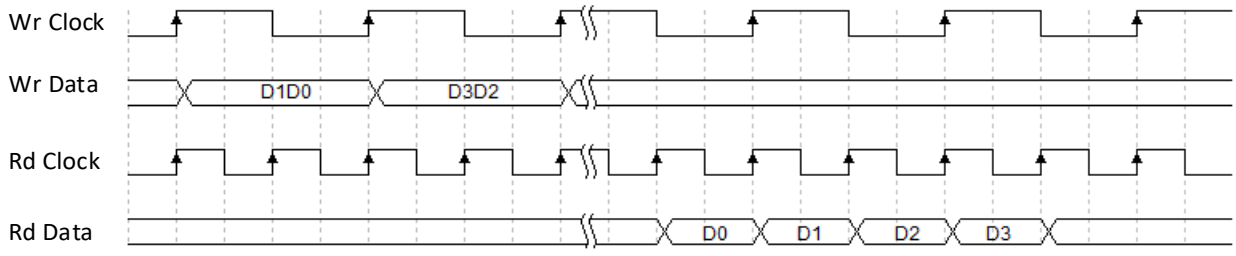

Figure 6.7. Tx Gearing Case II

The phase compensation FIFO resolves clock phase difference between its read side and write side. The FIFO can signal its overflow and underflow status once they happen. Write side and read side clock should not have frequency difference. Otherwise, the FIFO overflow or underflow occurs.

The phase compensation FIFO can also be bypassed. When being bypassed, it works as synchronous D-Flip Flop (DFF) to capture the data from fabric. Table 6.4 is the comparison between using phase compensation FIFO and bypassing the FIFO.

Table 6.4. Tx FIFO Usage

	Using FIFO	Bypassing FIFO
Easy use	It is robust and easy to use. Timing closure is easy to be achieved.	Strict timing constraint must be applied to the Fabric-MPCS interface.
Latency	Bigger and uncertain latency is introduced by this FIFO.	Must use this mode if low latency or deterministic latency are required.
Lane-to-lane skew	Extra lane-to-lane skew between every two bonded channels may be introduced.	No extra lane-to-lane skew is introduced.

In the multiple-channel mode, the Tx Lane-to-lane Deskew FIFO is optionally enabled on all channels to eliminate the lane-to-lane skew introduced by the uncertain latency of phase compensation FIFO.
Note: Refer to the 8B/10B PCS Multiple Lane Tx Path section for more details about the multiple-channel alignment mode.

Rx FIFO

The Rx FIFO module also serves two purposes:

- 1:2 gearing MPCS data before forwarding to fabric.
- Clock phase compensation FIFO to ease MPCS-Fabric interface timing closure.

The 1:2 gearing can be optionally enabled to convert 2-byte data to 4-byte. Figure 6.8 shows how this conversion proceeds.

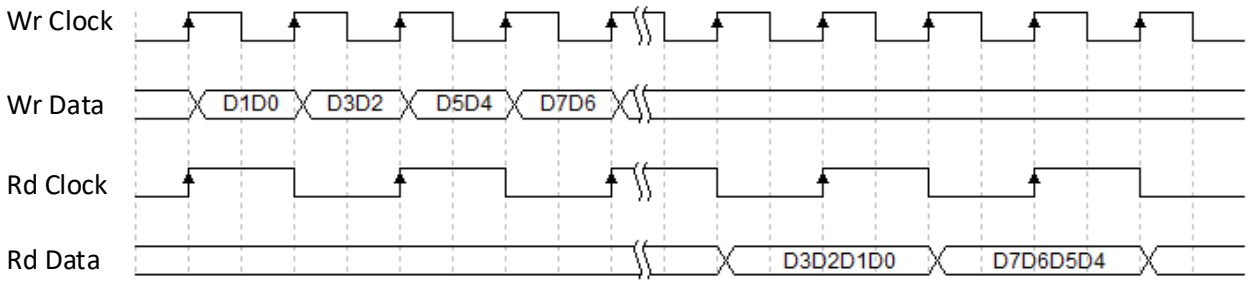

Figure 6.8. Rx Gearing Case I

The 1:2 gearing logic can convert 1-byte data to 2-byte, too. Figure 6.9 shows how this conversion proceeds.

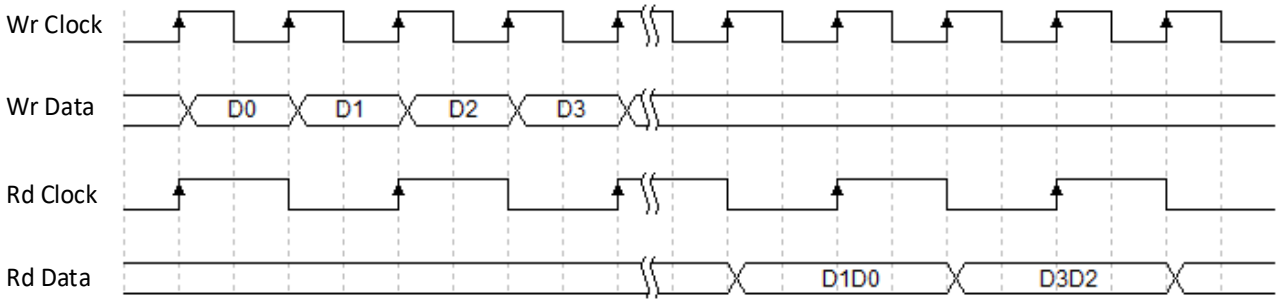

Figure 6.9. Rx Gearing Case II

This module can put the word alignment pattern (usually the control symbol) to the byte_0 (LSByte) of the 2-byte or 4-byte data bus. This byte-shifting feature can optionally be enabled. When byte-shifting feature is enabled, the module reports if the byte shifting happens. The byte-shifting feature is implemented for some protocols which always have the special symbol to be aligned on an even-numbered code-group boundary.

Figure 6.10 shows how this feature work. A is the word alignment pattern. X is padding byte.

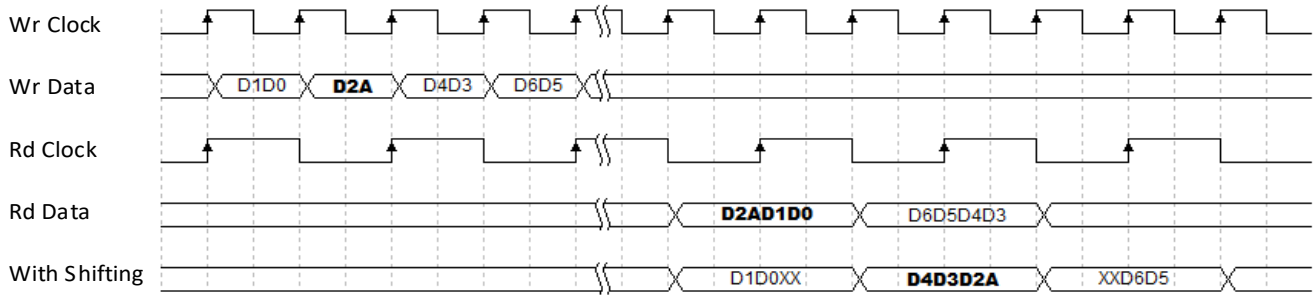

Figure 6.10. Byte Shifting for Word Alignment Pattern

The phase compensation FIFO resolves clock phase difference between its read side and write side. The write side and read side clock should not have frequency difference. Otherwise, the FIFO overflow or underflow occurs. The phase compensation FIFO can be bypassed. When bypassing, it works as synchronous DFF to launch the data to fabric.

8B/10B Encoder

This encoder performs the 8-bit to 10-bit code conversion along with maintaining the running disparity rules specified. The functionality of this encoder is compatible with most 8B/10B PCS based protocols.

The 8B/10B encoder implements two modes: 1-byte mode and 2-byte mode. In 1-byte mode, encoder encodes one 8-bit input data in one clock cycle; in 2-byte mode, encoder encodes two 8-bit input data in one clock cycle.

Some protocols require the transmitter to start up with a certain disparity. The module allows the user to replace the current running disparity calculated by this module with the required disparity. The forced disparity can be used as the start to calculate disparity for the subsequent 8b data.

The “Force Disparity” and “Invert Disparity” are controlled by two input signals: “tx_frcdisp” and “tx_dispval”. The combinations of their values are listed in Table 6.5 below.

Table 6.5. Disparity Combinations

tx_frcdisp	tx_dispval	Disparity
1	0	Force negative disparity.
1	1	Force positive disparity.
0	0	Calculated by the 8B/10B encoder.
0	1	Invert the current disparity.

Some protocols require to output certain patterns with invalid 10b code. This can be achieved by forcing data mode. In forcing data mode, the 8B/10B encoder module can output the input 10-bit data (1-byte mode) or 20-bit data (2-byte mode) directly. This forcing data mode is implemented for protocol like SLVS-EC that needs 8B/10B encoding/decoding, but needs to send some invalid 10b code sometimes.

Note: The “force data” function takes precedence over “force disparity” and “invert disparity” if they are enabled simultaneously.

The IEEE 802.3 standard, clause 36, specifies two idle order sets (/I1/ and /I2/) for the transmitter. The /I1/ ordered set consists of a negative disparity /K28.5/ (10’h283) followed by a /D5.6/ code group. (A /D5.6/ has the same value, 10’h1A5, for the positive and negative disparity versions, and has a balanced 10-bit code.) The /I1/ ordered set should be transmitted only once, if the running disparity before the idle is positive. The /I2/ ordered set consists of a positive disparity /K28.5/ (10’h17C) followed by a negative disparity /D16.2/ (10’h289) code group. The /I2/ ordered set can start the idle sequence, if the disparity before the idle sequence is negative. Otherwise, /I2/ follows a /I1/ ordered set and is continually transmitted, maintaining a negative running disparity until the end of the Inter-Packet Gap (IPG).

However, from the FPGA soft-logic side, the current disparity state of the transmitter is unknown. This is where the inline bit “cordisp” comes into play. If the “cordisp” is asserted for one clock cycle upon entering an IPG, while “cntl”=0, “txdata”=0x50, it forces this module to output a /I1/ ordered-set into the transmit data stream, if the current disparity is positive. However, if the current disparity is negative, no change is made to the transmitted data stream. For the remainder of the IPG, /I2/s should be driven into the MPCS and the “cordisp” should remain de-asserted.

The /I2/ Ordered set is replaced with /I1/ automatically, when MPCS is working in GigE (1000BASE-X) mode.

This 8B/10B encoder also supports interleaving mode when performing 8B/10B encoding. In this mode, the current input 8b data is used to calculate a new value of running disparity. The new value can then be used as running disparity for encoding the input data after the next data. However, the data stream seems to be separated into two streams, encoded independently and then merged to one steam again. Finally, the encoded data are transmitted to PMA with the same order as that they enter the 8B/10B encoder. An example is given below.

The input byte sequence is:

B0 (first input byte) -> B1 -> B2 -> B3 -> B4 -> B5 ...

During encoding,

B2 uses running disparity from B0 (rather than B1);
B3 uses running disparity from B1;
B4 uses running disparity from B2;
…

Since the disparity rule is violated, this mode is only used for protocols, like RXAUI, in which merging two data streams takes place after 8B/10B encoding.

8B/10B Decoder

The decoder performs the 10-bit to 8-bit code conversion along with verifying the running disparity. Corresponding to the encoder, the 8B/10B Decoder implements two mode: 1-byte mode and 2-byte mode. In 1-byte mode, the decoder decodes one 10b data in one clock cycle; in 2-byte mode, the decoder decodes two 10b data in one clock cycle.

This decoder supports running disparity error detection feature based on the 10-bit code it received. The current running disparity is based on the disparity calculation of the last code received. If negative disparity is calculated for the last 10-bit code, a neutral or positive disparity 10-bit code is expected. If the decoder does not receive a neutral or positive disparity 10-bit code, the “rx_errdisp” signal goes high. If a positive disparity is calculated, a neutral or negative disparity 10-bit code is expected. In this situation, the rx_errdisp signal goes high, if the code received is not what are expected.

If the code received is not a valid encoded data character or control character, the “rx_errcode” signal goes high. This signal is aligned to the invalid code word received at fabric.

The inline bit “rundisp” represents the current running disparity:

- 1 = positive
- 0 = negative

The decoder also supports the interleaving mode when performing 8B/10B decoding. In this mode, the current input 8b data is used to calculate a new value of running disparity. The new value can then be used as running disparity for decoding the input data after the next data. This is symmetric function to the 8B/10B encoder interleaving mode. Refer to 8B/10B encoder interleaving mode for more detailed description and potential application scenario.

Word Aligner

The word boundary of the upstream transmitter is lost upon deserialization, considering that the data is serialized before transmission and then de-serialized at the receiver. The word aligner receives parallel data from the De-Serializer and restores the word boundary.

To make alignment possible, transmitters send a recognizable sequence (usually a COMMA) periodically. The receiver searches for the COMMA in the incoming data. Once the receiver finds the COMMA, it moves the COMMA to a byte boundary, and the received parallel words match the transmitted parallel words.

Figure 6.11 shows the block diagram of the Word Aligner module.

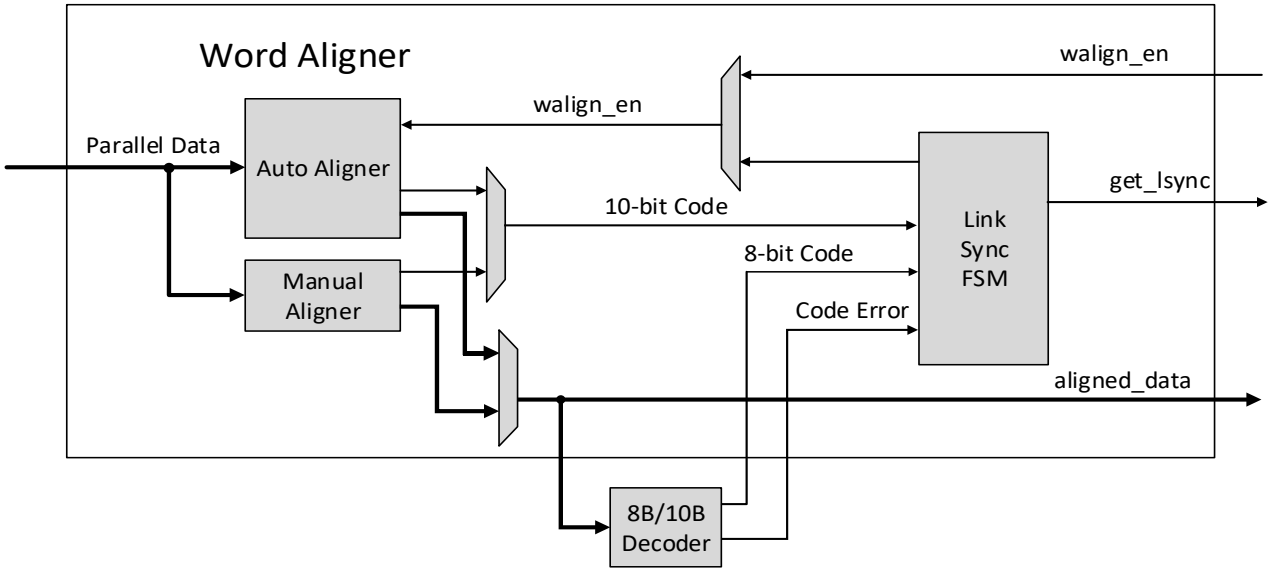

Figure 6.11. Word Aligner Block Diagram

The Word Aligner module supports automatic alignment mode. This module searches for word alignment pattern in the input parallel data, then move the matched pattern to byte boundary and align the output data to the boundary. The rising edge of walign_en signal triggers the word alignment operation. Otherwise, the module locks the current boundary and avoid re-alignment. The walign_en signal can be driven either by user logic or by Link Synchronization FSM which is a state machine to detect the loss of link synchronization and signal the re-alignment request.

To support applications which need deterministic and lower latency, CertusPro-NX SerDes/PCS also supports manual alignment mode (or bit slip mode in some documents). In manual mode, the rising edge of skipbit signal triggers the input parallel data to shift one UI inside PMA. User logic can count how many rising edges of skipbit signal to know the total number of shifted bits inside PMA.

The maximum allowed alignment pattern length is 20-bit. The user can define the length as either 10-bit or 20-bit. 8-bit and 16-bit patterns are implemented by applying mask code to 10-bit and 20-bit pattern respectively. Word aligner module provides programmable primary alignment pattern. The total length of the alignment pattern register is 20-bit. Only bit[9:0] are to be used when 10-bit pattern is used.

To support certain protocols or other user-defined protocols, Word Aligner module also provides programmable secondary alignment pattern. The length of the alignment pattern is 20-bit, and bit[9:0] are used for 10-bit length pattern. This secondary alignment pattern can optionally be enabled. If the secondary alignment pattern is enabled, matching either primary or secondary pattern is allowed during detecting alignment pattern.

The programmable alignment pattern mask code is 20-bit length, which has one-to-one bitwise correspondence to the 20-bit alignment pattern. It is important to note that the mask code can be applied to both primary and secondary alignment patterns. The corresponding pattern bit is applied to the pattern matching process when the related bit value is set to 0. The corresponding pattern bit is ignored in the pattern matching process when the related bit value is set to 1.

The internal data bus width can be configured as 1-byte mode or 2-byte mode. In 2-byte mode, the module can optionally move the matched data bytes to the 2-byte boundary.

Some protocols always have the special symbol to be aligned on an even-numbered code-group boundary. The two-byte alignment pattern defined in this example is “align pattern byte0” (LSByte) and “align patter byte1” (MSByte). The data bus before 2-byte boundary alignment is shown in Figure 6.12.

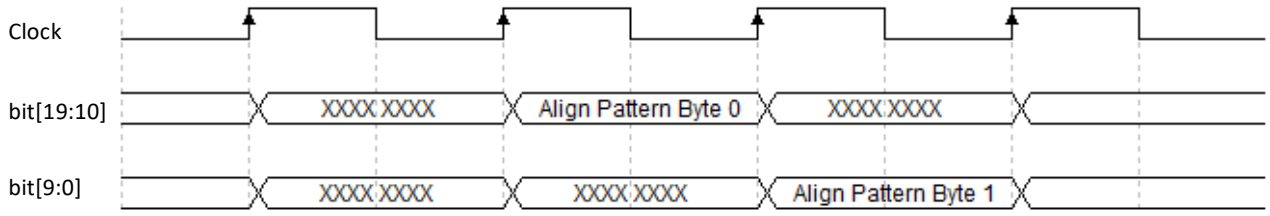

Figure 6.12. Before 2-byte Boundary Alignment

After 2-byte boundary alignment, the LSByte of alignment pattern always appears at bit[9:0] of the 20-bit data bus, as shown in Figure 6.13.

Figure 6.13. After 2-byte Boundary Alignment

Word aligner module can report the number of bits slipped during alignment operations. This feature helps in calculating the latency through the receiver data path. The waveform in Figure 6.14 and Figure 6.15 show how the word aligner locates the data boundary in the input parallel data and align outputted data to the boundary. In this example, the alignment pattern is defined as K28.5=0101111100.

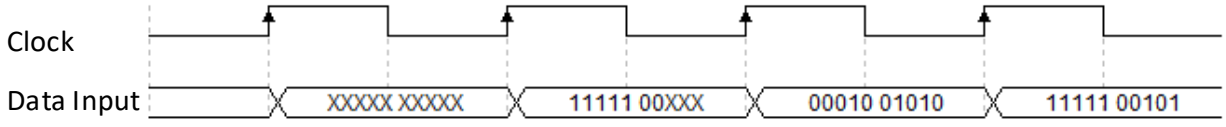

Figure 6.14. Data Stream before Word Alignment

After locking the data boundary, this module starts outputting aligned data: K28.5 followed by D16.2 (1010001001). Seven bits are slipped in this example.

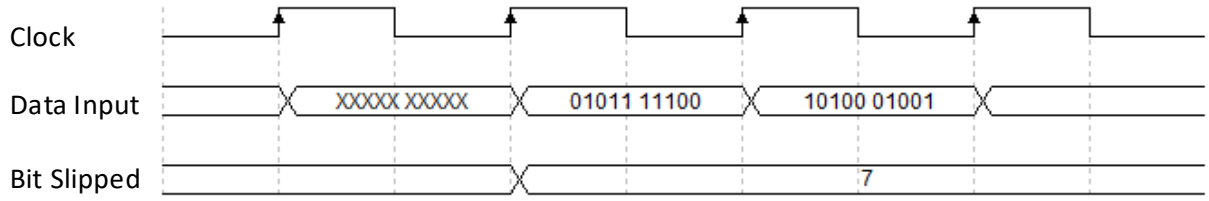

Figure 6.15. Data Stream after Word Alignment

However, the number of bit slipped is always zero in manual alignment mode considering that the bit shifting is operated by PMA and controlled by fabric.

The link synchronization is an extension of the word aligner. The state machine implements hysteresis during link synchronization. The number of synchronization code groups that the link must receive to acquire synchronization and the number of erroneous code groups that it must receive to fall out of synchronization are programmable. Figure 6.16 shows the Link Synchronization FSM.

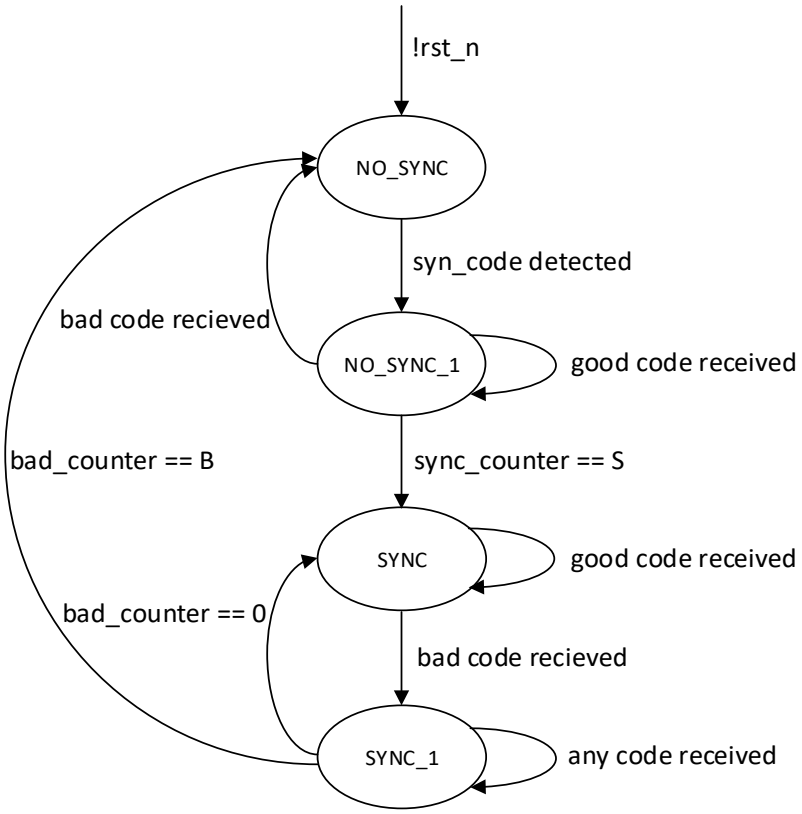

Figure 6.16. Link Synchronization FSM

After reset, Link Synchronization FSM enters the default state: NO_SYNC. In NO_SYNC state, the FSM tries to find the synchronization code in the input data stream. The FSM assigns 1 to sync_counter, then moves from NO_SYNC state to NO_SYNC_1 state whenever valid synchronization code is found. Otherwise, the sync_counter, good_counter, and bad_counter are assigned to the default values 0.

In NO_SYNC_1 state, the FSM continues to detect the synchronization code. The sync_counter is added 1 after new synchronization is detected. The FSM moves from NO_SYNC_1 state to SYNC state when sync_counter equals S (one configurable parameter). However, the FSM moves back to NO_SYNC state whenever any bad code (invalid code in 8B/10B coding) is detected.

Data stream is aligned successfully in SYNC state. The Auto Aligner module locks the current boundary and avoids re-alignment. In SYNC state, the FSM monitors continuously if there is any bad code appearing. The bad_counter is added 1 when bad code is detected, meanwhile the FSM moves from SYNC state to SYNC_1 state.

In SYNC_1 state, the bad_counter is added 1 and the good_counter is cleared whenever any bad code is detected. Otherwise, the good_counter is added 1 when valid code is received. The bad_counter is minus 1 whenever the good_counter equals G (one configurable parameter), and meanwhile the good_counter is cleared. The FSM moves back to SYNC state when bad_counter equals 0, or moves to NO_SYNC state to make a re-alignment request when the bad_counter equals B (another configurable parameter).

Following parameters are configurable:

- Number of valid synchronization code groups or ordered sets received to achieve synchronization (S).
  - The range of this parameter is 3 to 255.
- Number of bad code groups received to lose synchronization (B).
  - The range of this parameter is 3 to 63.
- Number of continuous good code groups received to reduce the error count by one (G).
  - The range of this parameter is 3 to 255.

Table 6.6 is the example settings about parameter S, B and G for XAUI and GigE protocols.

Table 6.6. Example Settings for XAUI and GigE

	GigE	XAUI
S	3	4
B	4	4
G	4	4

This Link Synchronization FSM can optionally be enabled or disabled. If disabled, user logic should implement the state machine to meet some criteria of synchronization or loss of synchronization and drive walign_en to request the re-alignment operation once loss of synchronization is detected.

The length of synchronization code (or named synchronization detect pattern) can be configured as 1-byte, 2-byte or 4-byte. All bits of synchronization code are configurable and maskable. Once masked, the corresponding bit of synchronization code is ignored during pattern matching. Primary and secondary synchronization code are provided separately, and the secondary synchronization code can optionally be disabled.

The Link Synchronization FSM supports both 10-bit mode (bypassing 8B/10B decoder) and 8-bit mode (after 8B/10B decoding) of input data. The bit mapping of 8-bit mode and 10-bit mode is shown in Figure 6.17.

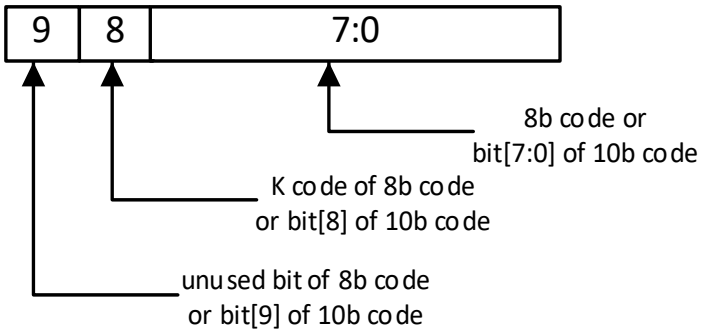

Figure 6.17. Bit Mapping of Input Data

The first byte of synchronization code must appear on N-byte boundary, where N is the synchronization code length. Once the synchronization code is detected, the Link Synchronization FSM applies this criterion to the validity check of subsequent incoming data. Figure 6.18 shows the location where the first byte of synchronization code is expected to appear.

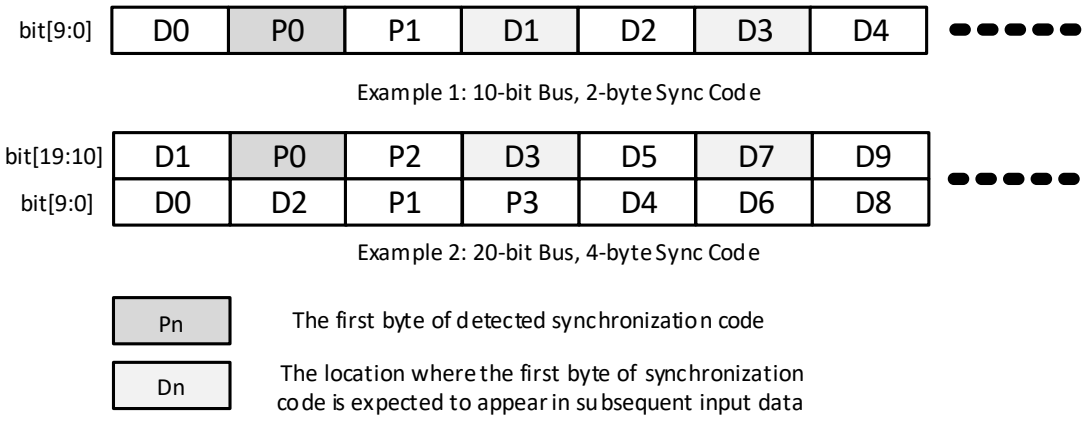

Figure 6.18. The Expected Location of Synchronization Code

Tx Lane-to-lane Deskew

One clock cycle lane-to-lane skew may be introduced by the uncertain latency of Tx FIFO (works as phase compensation FIFO). The Tx Lane-to-lane Deskew FIFO is implemented to eliminate this skew between two different Tx channels by using the common clock from Quad Common module.

The common clock has no frequency difference with respect to each channel's tx_pcs_clk, but can have a big phase difference. The Quad Common module selects one channel's tx_pcs_clk as the write clock of all channels' Tx Lane-to-Lane Deskew FIFO, while the read clock of each channel is the original tx_pcs_clk generated by each channel's PMA. Check the MPCS Quad Clock Detail section for more details about this common clock.

The Tx Lane-to-lane Deskew FIFO can be disabled in single lane application or Tx FIFO is bypassed.

Elastic Buffer

This Elastic Buffer (or named as Clock Compensation Buffer in some documents) performs clock frequency adjustment between the recovered receive clock domain and the local system clock domain. The Elastic Buffer performs clock compensation by inserting or deleting bytes at the position where SKIP pattern is detected, without causing loss of packet data. A 32-byte Elastic FIFO (CTC FIFO) is implemented to temporarily buffer coming data from recovered receive clock domain and transfer the data to local system clock domain.

The Elastic Buffer can optionally be enabled or disabled. Figure 6.19 shows the format of common SKIP pattern used by most 8B/10B PCS based protocols: one COM byte optionally followed by 1 to 3 SKP bytes.

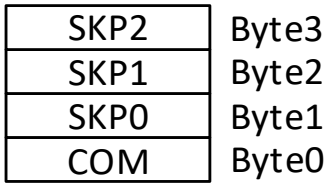

Figure 6.19. SKIP Pattern Format

The length of SKIP pattern can be set to 1, 2, or 4 byte. In 1-byte length mode, only Byte 0 is used; in 2-byte length mode, Byte 0 and Byte 1 are used, and Byte 0 holds the pattern byte received firstly (usually a COM byte).

This Elastic Buffer provides two SKIP patterns: primary and secondary. Primary SKIP pattern is always enabled, and secondary SKIP pattern can optionally be enabled. This module matches either of these two SKIP patterns, if the secondary SKIP pattern is enabled. However, only primary SKIP pattern can be used in 8-bit data mode.

A mask code is provided to allow partially matching the SKIP pattern. Refer to Figure 6.20. The mask code has 4-bit length, each bit corresponds to one byte of the SKIP pattern. Value 1 means the corresponding byte is masked, and value 0 means not masked. SKIP byte being masked is ignored during SKIP matching. It is important to note that the mask code is applicable for both primary and secondary SKIP patterns.

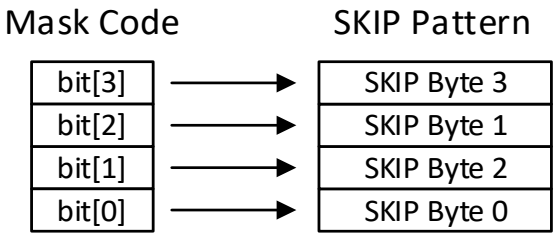

Figure 6.20. SKIP Pattern Mask Code

This Elastic Buffer supports both 10-bit data mode (bypassing 8B/10B Decoder) and 8-bit data mode (after 8B/10B decoding). The bit mapping of 8-bit mode and 10-bit mode is shown in Figure 6.17.

The clock frequency compensation operation is automatic, but the deletion/insertion of SKIP pattern can be observed by user logic. “skp_add” is driven high to indicate the insertion of SKIP pattern, and “skp_del” is driven high to indicate the deletion of SKIP pattern. “skp_add” and “skp_del” signal the Elastic FIFO overrun and underrun status. “ebuf_empty” is driven high when this elastic FIFO is empty, and “ebuf_full” is driven high when elastic buffer is full.

Programmable high-water line and low water line are shown in Figure 6.21. The module starts to send data out when the FIFO is half-full (byte number gets Middle threshold). When the byte number of data residing in the FIFO exceeds the high-water line, this module removes incoming SKIP pattern to prevent overflow. When the byte number of data residing in the FIFO drops below the low water line, this module adds additional SKIP pattern from the incoming SKIP pattern it matches to prevent underflow.

The minimum number of SKIP patterns after deletion can be set as one, two or three. This feature is useful when the minimum IPG needs to be guaranteed in protocols like GigE.

The Elastic Buffer can control the frequency of clock frequency correction operation (SKIP insertion and deletion), which is implemented by defining the minimum clock cycles required between two clock correction operations.

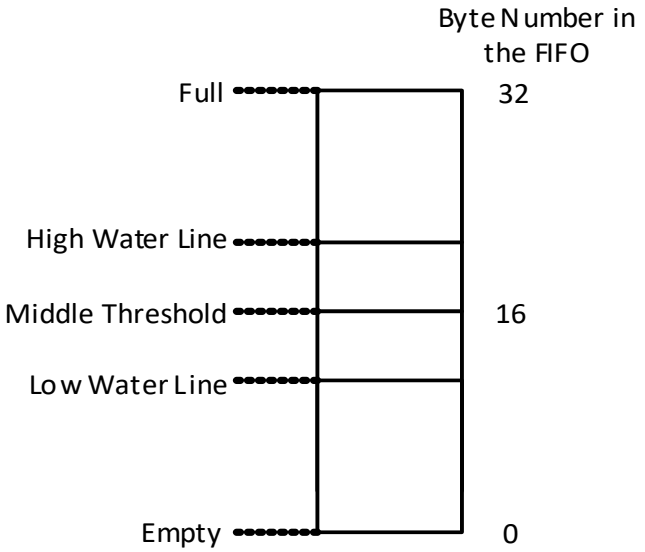

Figure 6.21. FIFO High/Low Water Line

Lane Aligner

Lane Aligner module is implemented to transfer the input data in respective recovered clock domains into a common clock domain, and get all lanes aligned before forwarding data to next stage at the same time. This module should be disabled in single-lane applications.

The Lane Aligner module looks for the predefined alignment pattern in the coming data and writes the data into a FIFO. Once the first alignment pattern is received on a certain lane, the de-skew FIFO read-and-write pointers are both reset to zero. The write pointer is then incremented for subsequent input data. The read pointer keeps zero and output data of the FIFO read side is repeated for the last read-out data. Until alignment pattern is detected by all lanes, the read pointer of the FIFO starts incrementing for each read clock cycle, aligning all lanes.

Usually, the write clock is from Rx PMA of each channel, while the read clock is from the Quad Common module. Check the MPCS Quad Clock Detail section for more details about the common clock.

The internal data bus width can be set as 1-byte or 2-byte mode. The input data of this module can be from 8B/10B Decoder or from Word Aligner (8B/10B Decoder bypass mode). Both primary lane alignment pattern and secondary lane alignment pattern are up to 4-byte length, the secondary one is optional and can be disabled. If the secondary lane alignment pattern is enabled, the input data can match either primary or secondary pattern.

The alignment pattern length can be defined as 1, 2, or 4 bytes. If the length is 1, only byte 0 of lane alignment pattern (primary and secondary) are used; if the length is 2, both byte 0 and byte 1 are used; if the length is 4, all four bytes of alignment pattern are used.

A 4-bit mask code is provided to allow partially matching the alignment pattern (Figure 6.22). Each bit of the mask code corresponds to one byte of the alignment pattern, value 1 means the corresponding byte is masked. Alignment pattern byte being masked is ignored during matching. The mask code is applicable for both primary and secondary alignment patterns.

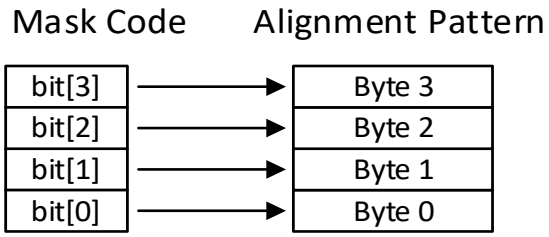

Figure 6.22. Lane Alignment Pattern Mask Code

The maximum lane-to-lane skew can be handled by this module is 10-byte clock cycles (or 100 UI), but user can reconfigure this value to a lower value as per protocols. It is recommended to set the real max skew plus one or two, considering that the extra skew is introduced by Quad-to-Quad handshaking (one for 1-byte mode, two for 2-byte mode).

In 2-byte internal data bus width mode, this module shifts the coming data by one byte towards LSByte of the data bus so that the first byte of the alignment pattern always appears at byte_0 (LSByte) of the data bus. When a data group that matches the alignment pattern is found in the coming data, the first byte of the matching pattern can occur at byte_0 or byte_1 of a 2-byte data bus. This introduces an extra lane-to-lane skew. During performing lane alignment operation, this module can move the data to 2-byte boundary to remove this skew. Figure 6.23 shows how the data is shifted. {A0, A1, A2, A3} is the 4-byte alignment pattern, and A0 is the first byte.

Figure 6.23. Data Shifting for Alignment

The lane alignment operation is triggered by the rising edge of signal rx_deskew_en. The rx_deskew_en should keep asserted until next time the lane alignment operation is required. The module signals the achievement of lane alignment by driving rx_get_lalign high as long as rx_deskew_en is asserted. The rx_get_lalign signal is driven low once the de-assertion of rx_deskew_en signal is detected. Toggling rx_deskew_en (driven low and then high) can trigger another session of lane alignment.

The rx_deskew_en signal can be driven low once the rising edge of rx_get_lalign is detected. Another choice is to keep the rx_deskew_en high until the next lane alignment is required.

User logic can implement lane alignment state machine to monitor the alignment status and control Lane Aligner module to perform re-alignment operation once loss of lane alignment is detected by user logic.

6.4.2. 64B/66B PCS

Figure 6.24 shows the block diagram of the 64B/66B PCS inside MPCS channel. This block implements the 10GBASE-R PCS defined by IEEE802.3, which contains following sub-blocks:

- Tx Gear Box and Rx Gear Box
- PRBS Generator and Checker
- Scrambler and Descrambler
- PRD Pattern Generator and Checker
- 64B/66B Encoder and Decoder
- Block Aligner
- BER Monitor
- Tx FIFO and Rx FIFO

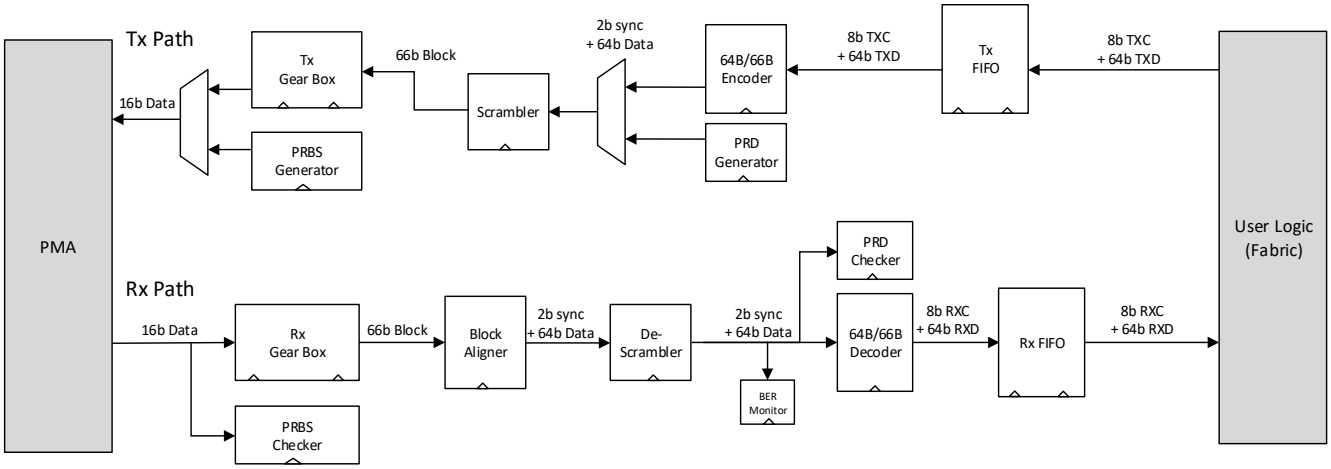

Figure 6.24. 64B/66B PCS Channel Block Diagram

6.4.2.1. Data Bus Description

The 10 Gigabit Media Independent Interface (XGMII) is a Double Data Rate (DDR) interface between MAC and PCS, defined by IEEE802.3 specification. CertusPro-NX device implements a Single Data Rate (SDR) MPCS-Fabric interface to replace the standard XGMII, considering that it is difficult to implement the DDR interface with FPGA fabric. Figure 6.25 shows the comparison between XGMII and CertusPro-NX MPCS- Fabric interface.

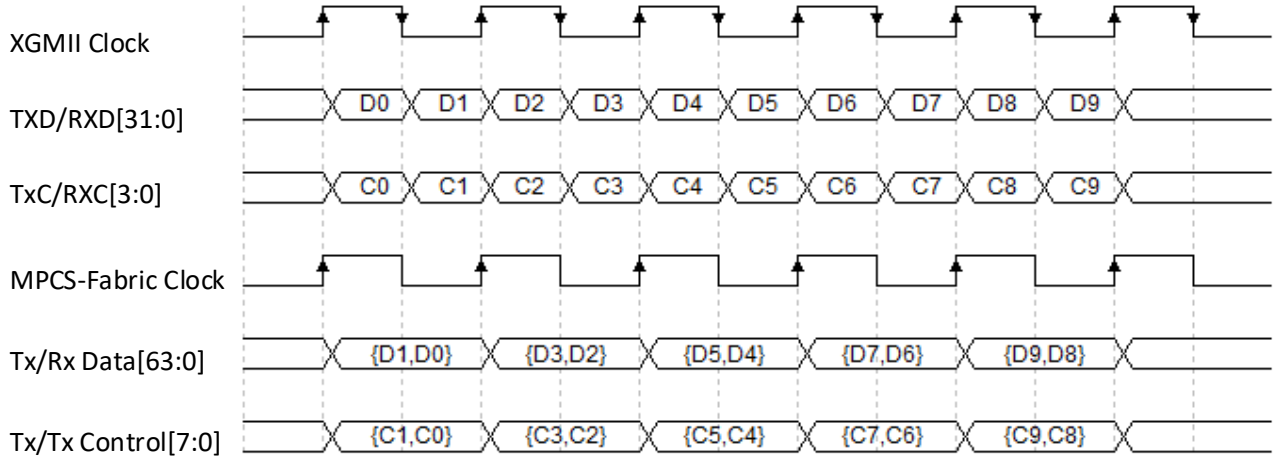

Figure 6.25. XGMII vs. MPCS-Fabric Interface

6.4.2.2. Function Block Description

Tx/Rx Gear Box

The Tx/Rx Gear Box module implements 4:1 gearing functionality in order to adapt the data path between the 66-bit width of 64B/66B block and the 16-bit width of PMA data bus. The PCS internal data bus (between Tx/Rx Gear Box and fabric) bandwidth is one bit higher than the PMA data bus bandwidth, considering PCS internal data bus using the div-by-4 clock from PMA Controller:

$$16 \times f_{16bit} = 64 \times f_{64bit} < 66 \times f_{64bit}$$

To solve this bandwidth mismatch issue, the Tx/Rx Gear Box processes 32 66-bit data blocks in 33 clock cycles. An internal signal, which is driven low for one clock cycle in every 33 cycles and is used to indicate the valid 66-bit data block.

Tx Gear Box and Rx Gear Box are implemented for 64B/66B PCS channel, and cannot be used by 8B/10B PCS channel or PMA only channel.

PRBS Generator and Checker

The PRBS Generator module can generate bit sequences of square wave, PRBS9 pattern, and PRBS31 pattern for debugging purpose. The PRBS Checker module checks the corresponding pattern enabled at the PRBS Generator module and performs error counting.

The square wave pattern is intended to aid in conducting certain transmitter tests required by IEEE802.3 10GBASE-R PCS. It is not intended for receiver tests, and the PRBS Checker module is not expected to receive this square wave pattern. When square wave pattern is selected, the PRBS Generator sends a repeating pattern of n ones followed by n zeros where n can be 4, 6, 8 or 11.

Scrambler and Descrambler

The Scrambler module scrambles the 64-bit block payload data using $x^{58} + x^{39} + 1$ polynomial specified by IEEE802.3 specification. The Descrambler module descrambles the 64-bit block payload data received from PMA. Expect for the two sync header bits, the entire 64-bit payload data of a 66-bit data block is scrambled and descrambled.

Both the Scrambler module and Descrambler module can be optionally bypassed for test purposes. If bypassed, the scrambler directly sends the 66-bit input data to the output ports.

PRD Generator and Checker

The Pseudo Random Data (PRD) Generator module generates the pseudo random pattern for test purpose. This pseudo random data is named as pseudo-random pattern in IEEE802.3 specification section 49.2.8.

The pattern generated by PRD Generator module can be set as 64-bit payload data in a 66-bit data block, then is forwarded to Scrambler module. The seed of the pattern can be configured as seed A or seed B by accessing 10GBASE-R Test Pattern Control registers. Both seed A and seed B can be changed by accessing related registers.

The PRD Generator is loaded with a seed or its inverse at the start of a block every 128 blocks. Either 64-bit zeros or the 64-bit encoding for two Local Fault ordered sets can be selected as the data pattern, depending on seed settings. The sync header is fixed to the control sync header, 2'b10. Thus, the pseudo-random pattern is a series of blocks with the control sync header and a pseudo-random payload. The characteristics of the pseudo-random pattern can be varied by varying the seed values and data input.

The PRD Checker module checks the pseudo-random pattern and counts the mismatch error, which can be recorded in 10GBASE-R Test Pattern Error Counter registers.

64B/66B Encoder

The 64B/66B Encoder module encodes 64-bit XGMII data and 8-bit XGMII control into the 66-bit data block. This module can be optionally bypassed for test purpose.

In force packet mode (tx_frcpkt is asserted), the 64-bit data and 2-bit control from user logic are sent to Tx Gear Box directly. Both 64B/66B Encoder and Scrambler modules are bypassed.

64B/66B Decoder

The 64B/66B Decoder module reverses the 64B/66B Encoder process, which converts the 66-bit data block to 64-bit XGMII data and 8-bit XGMII control. This module can be optionally bypassed for test purpose.

The decoder performs functions such as sending local faults to the Media Access Control (MAC)/Reconciliation Sublayer (RS) under reset and substituting error codes when the 10GBASE-R PCS rules are violated.

Tx FIFO

The Tx FIFO module is implemented to adapt Tx path clock frequency and phase difference between 64B/66B PCS channel and fabric. It serves in following two application cases:

Case I (with GPLL)

In this case, GPLL is required to generate a 156.25 MHz clock as the write clock of Tx FIFO. User logic can write 64-bit XGMII data and 8-bit XGMII control to Tx FIFO continuously, and the tx_fifo_wr can be tied to high. User logic does not need to monitor the FIFO status in this case. Refer to Figure 6.26 for the timing diagram of this case.

For more details about this clocking scheme of 64B/66B PCS path, refer to the 64B/66B PCS with GPLL section.

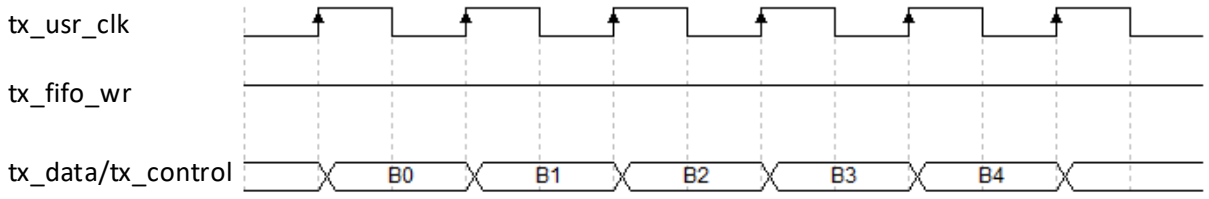

Figure 6.26. 64B/66B PCS Tx FIFO Write Operation Case I

Case II (without GPLL)

In this case, user logic takes the clock originated from PMA as the write clock of Tx FIFO. The FIFO status needs to be monitored, considering that the frequency of write clock and read clock is the same but the bandwidth between the write side and the read side is different. The read side should be adapted to the bandwidth of 64B/66B Encoder module 64bit@161.1328125MHz, and the write side is 64bit@161.1328125MHz.

User logic can implement FIFO write control logic by monitoring the almost full status of Tx FIFO. Once the almost full flag is asserted, writing FIFO should be stopped immediately. Alternatively, user logic can make a pause of writing for one clock cycle in every 33 cycles without monitoring the almost full status of Tx FIFO. For more details about this clocking scheme of this case, refer to the 64B/66B PCS without GPLL section.

In Figure 6.27, the tx_fifo_wr is controlled properly so that there is neither overflow nor underflow. The almost full and almost empty flags can be reset by accessing 64B/66B PCS Tx FIFO Almost Full Setting register and 64B/66B PCS Tx Almost Empty Setting register.

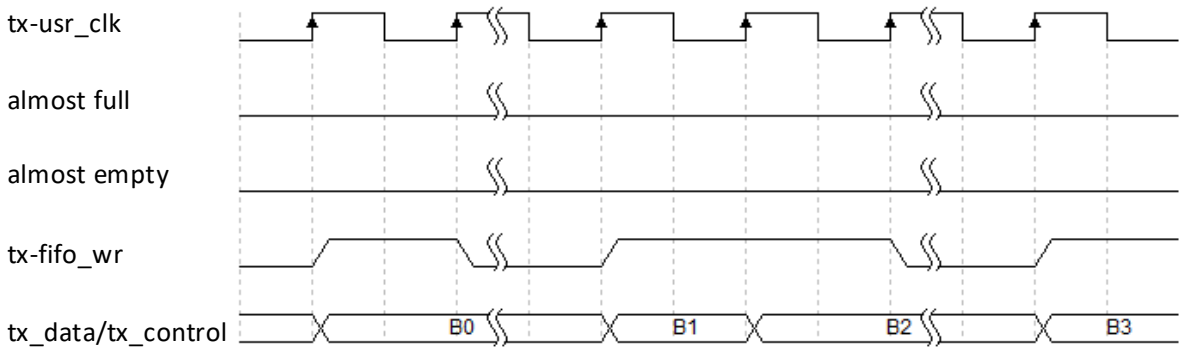

Figure 6.27. 64B/66B PCS Tx FIFO Write Operation Case II

Rx FIFO

The Rx FIFO module is implemented to adapt Rx path clock frequency and phase difference between 64B/66B PCS and fabric. It also serves in the following two application cases.

Case I (with GPLL)

In this case, GPLL is required to generate a 156.25 MHz clock as the read clock of Rx FIFO. The rx_data_valid is continuous because the bandwidth between the write side and read side is the same, 64bit@161.1328125MHz and 66bit@156.25MHz. Refer to Figure 6.28 for the timing diagram of this case. For more details about this clocking scheme of this case, refer to the 64B/66B PCS with GPLL section.

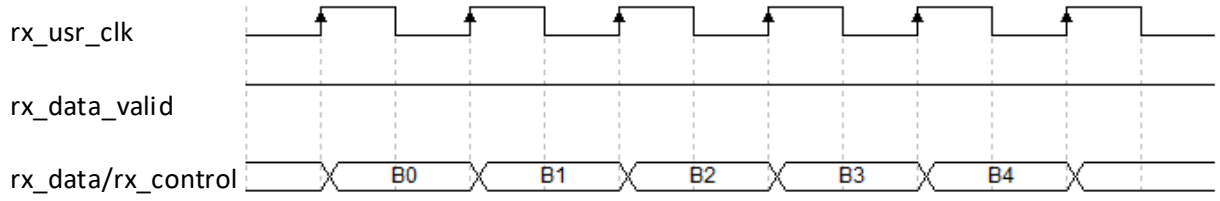

Figure 6.28. 64B/66B PCS Rx FIFO Read Operation Case I

Case II (without GPLL)

In this case, user logic takes the clock originated from PMA as the read clock of Rx FIFO. The data valid signal for the data out of Rx FIFO becomes discontinuous, considering that the frequency of write clock and read clock is the same but the bandwidth between the write side and the read side is different. The write side should be adapted to the bandwidth of 64B/66B Decoder module, 64bit@161.1328125MHz, and the read side is 64bit@161.1328MHz. Refer to Figure 6.29 for the timing diagram of this case. For more details about this clocking scheme of this case, refer to the 64B/66B PCS without GPLL section.

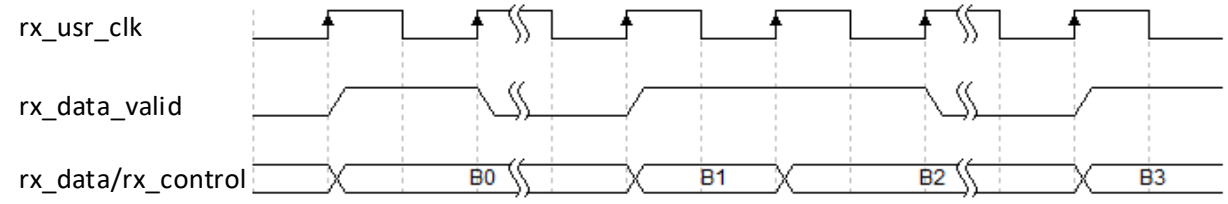

Figure 6.29. 64B/66B PCS Rx FIFO Read Operation Case II

The Rx FIFO also implements Clock Tolerance Compensation (CTC) functionality to compensate clock frequency up to ± 100 PPM difference. This functionality can compensate up to 1 clock cycle difference every 5000 clock cycles in Rx path by monitoring Rx FIFO status and inserting or deleting characters specified by IEEE802.3, if pre-defined criteria is met. What should be noted is that this CTC functionality can only be used in the case with GPLL.

Character deleting occurs, if the number of data residing in Rx FIFO exceeds high water line. Character inserting occurs, if the number of data residing in Rx FIFO is lower than low water line.

Idle control character (/I/) deleting/inserting rules include:

- /I/s are inserted or deleted in groups of 4.
- /I/s may be added following idle or ordered sets.
- /I/s shall not be added while data is being received.
- When deleting /I/s, the first four characters after a /T/ cannot be deleted.

The Sequence ordered sets (/O/) deleting rules include:

- Only Sequence ordered set can be deleted.
- Deleting only when two consecutive sequence ordered sets are received; and only one of the two consecutive sequence ordered set can be deleted.
- Signal ordered sets are not deleted for clock compensation.

Block Aligner

The Block Aligner module detects the boundary of a 66-bit data block received from Rx Gear Box. The incoming 66-bit data stream is slipped one bit at a time until the required number of valid synchronization header (bit 0 and bit 1) is detected in the received data stream.

This module implements block lock FSM specified in Clause 49 of IEEE802.3, which shifts the incoming data stream and detects the block boundary. The bit number of shifting can be checked through related status registers. The block lock is achieved when rx_blk_lock signal is driven high.

This module can optionally be bypassed for test purpose.

BER Monitor

The BER Monitor module is implemented in accordance with the 10GBASE-R protocol specified by IEEE802.3. The BER Monitor starts to count the number of invalid synchronization headers within a 125-µs period.

If more than 16 invalid synchronization headers are observed in a 125-µs period, the BER Monitor asserts the hi_ber signal, indicating a high bit error ratio condition. The BER Monitor counts the number of times the BER FSM has entered the BER_BAD_SH state, and the number of times the BER FSM has entered the RX_E state. These counter values can be checked through 10GBASE-R BER Counter register.

Note: The BER_BAD_SH state and RX_E state are defined by IEEE802.3.

6.4.3. PMA Only Mode

In this mode, the PMA EPCS data bus is accessed by user logic with very low latency. Only Tx/Rx FIFO and Tx Lane-to-lane Deskew modules are between the PMA Controller and fabric. Figure 6.30 shows the block diagram of PMA Only mode.

The functionality of these three modules in PMA Only mode are the same as those in 8B/10B PCS mode. Check the Function Block Description section for detailed function descriptions about these modules.

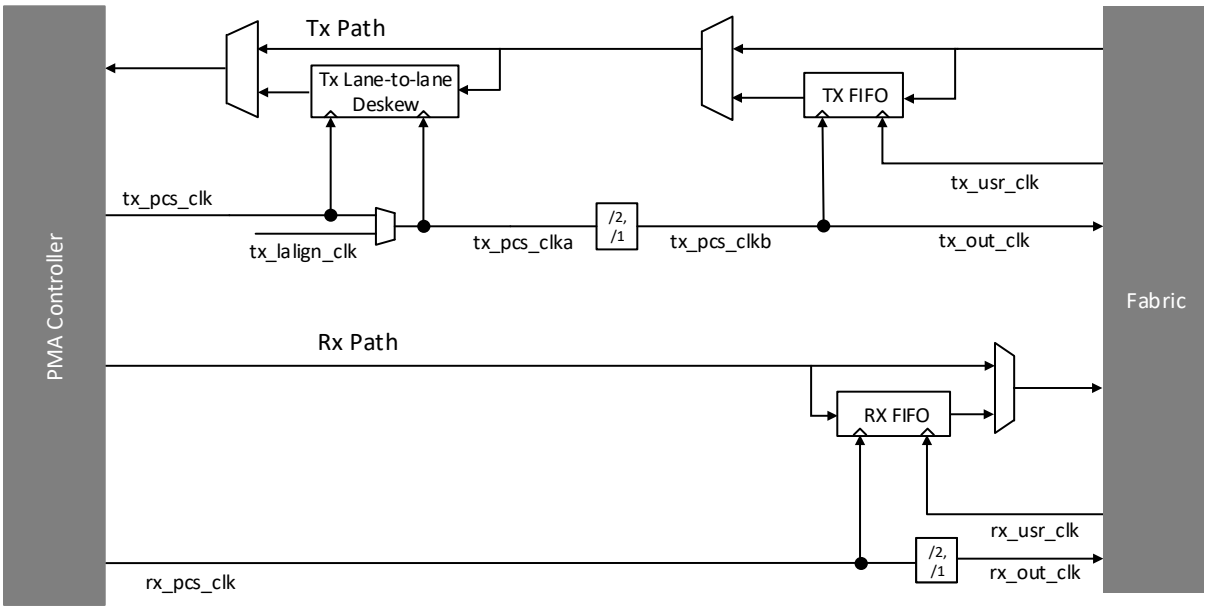

Figure 6.30. PMA Only Mode Block Diagram

6.4.3.1. Data Bus Description

The bit mapping of user logic data bus to PMA data bus is shown in Table 6.7. The MPCS internal data bus width can be 8-bit, 10-bit, 16-bit, and 20-bit, while the data bus width between MPCS and fabric can be 8-bit, 10-bit, 16-bit, 20-bit, 32-bit and 40-bit.

Table 6.7. PMA Only Mode Data Bus

PMA Data Bus Width	Mode	MPCS Internal Data Bus Width	Gearing	MPCS-Fabric Data Bus Mapping[1]	Transfer Ordering[2,3]
20-bit	Mode 1-a	20-bit	1:1	usr_dbus[19:0]	T0: usr_dbus[19:0]
	Mode 1-b		2:1	usr_dbus[39:0]	T0: usr_dbus[19:0] T1: usr_dbus[39:20]
10-bit	Mode 2-a	10-bit	1:1	usr_dbus[9:0]	T0: usr_dbus[9:0]
	Mode 2-b		2:1	usr_dbus[19:0]	T0: usr_dbus[9:0] T1: usr_dbus[19:0]
	Mode 2-c	20-bit	1:1	usr_dbus[19:0]	T0: usr_dbus[9:0] T1: usr_dbus[19:0]
	Mode 2-d		2:1	usr_dbus[39:0]	T0: usr_dbus[9:0] T1: usr_dbus[19:10] T2: usr_dbus[29:20] T3: usr_dbus[39:30]
5-bit	Mode 3-a	10-bit	1:1	usr_dbus[9:0]	T0: usr_dbus[4:0] T1: usr_dbus[9:5]
	Mode 3-b		2:1	usr_dbus[19:10]	T0: usr_dbus[4:0] T1: usr_dbus[9:5] T2: usr_dbus[14:10] T3: usr_dbus[19:15]
16-bit	Mode 4-a	16-bit	1:1	usr_dbus[15: 0]	T0: usr_dbus[15:0]
	Mode 4-b		2:1	usr_dbus[35: 20] usr_dbus[15: 0]	T0: usr_dbus[15:0] T1: usr_dbus[35:20]
8-bit	Mode 5-a	8-bit	1:1	usr_dbus[7:0]	T0: usr_dbus[7:0]
	Mode 5-b		2:1	usr_dbus[17:10] usr_dbus[7:0]	T0: usr_dbus[7:0] T1: usr_dbus[17:10]

Notes:

1. The “usr_dbus” is named as epcs_txdata_i in Tx path, or epcs_rxdata_o in Rx path.
2. The bit[0] is transmitted/received first.
3. Transfer ordering description:
   - T0: the first PMA clock cycle to capture/launch data in a data block transfer.
   - T1: the second PMA clock cycle to capture/launch data.
   - T2: the third PMA clock cycle to capture/launch data.
   - T3: the fourth PMA clock cycle to capture/launch data.

6.5. Quad Common

This per Quad module communicates with each channel within the Quad to implement the multiple lane alignment function for both transmitter and receiver. It also talks to neighboring Quads to implement the lane alignment across Quad boundary.

Within a Quad, the supported alignment modes are listed in Table 6.8. In Mode 1, Lane2 and Lane3 work independently (not aligned). In Mode 2, Lane0 and Lane1 work independently (not aligned).

Table 6.8. Channel Alignment within One Quad

Mode	Quad			
	Lane0	Lane1	Lane2	Lane3
Mode 1	2-lane alignment		—	—
Mode 2	—	—	2-lane alignment	
Mode 3	2-lane alignment		2-lane alignment	
Mode 4	4-lane alignment			

Together with adjacent Quad, this module supports up to 8-lane alignment. The allowed alignment modes are listed in Table 6.9. Each of these two Quads supports all single Quad alignment modes. In Mode 5, Lane2 and Lane3 in Quad_1 work independently (not aligned). In Mode 6, Lane0 and Lane1 in Quad_0 work independently (not aligned).

Table 6.9. Channel Alignment between Two Quads

Mode	Quad_0				Quad_1			
	Lane0	Lane1	Lane2	Lane3	Lane0	Lane1	Lane2	Lane3
Mode 5	6-lane alignment						—	—
Mode 6	—	—	6-lane alignment					
Mode 7	6-lane alignment						2-lane alignment	
Mode 8	2-lane alignment		6-lane alignment					
Mode 9	8-lane alignment							

6.6. Reference Clock

All four PMA channels inside one Quad share the same reference clock source. This reference clock can source from per-quad package pins (SDQx_REFCLKP/N), GPLL output, PCLK and dedicated two package pins (SD_EXT0_REFCLKP/N and SD_EXT1_REFCLKP/N). Refer to Figure 6.31 for the block diagram of reference clock source per quad.

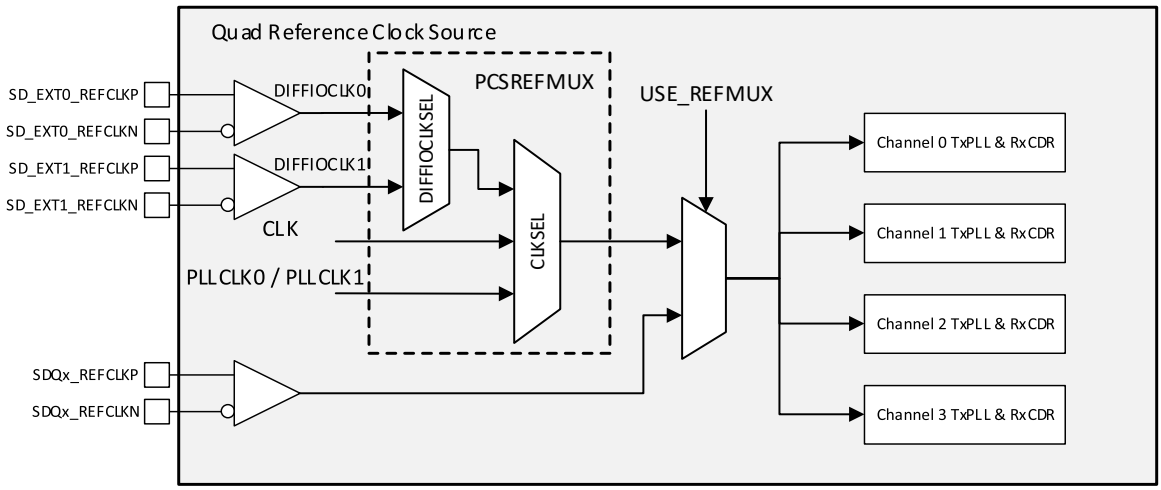

Figure 6.31. Quad Reference Clock Source

Both SDQx_REFCLKP/N receive buffer and SD_EXTx_REFCLKP/N receive buffer can convert differential input to single-ended output to drive the clock tree. However, the detailed implementation of these two receive buffer are different.

The SDQx_REFCLKP/N receive buffer is unterminated input and must be DC coupled out of chip. This SDQx_REFCLKP/N receive buffer supports voltage signal input or current signal, such as High Speed Current Steering Logic (HCSL). The SD_EXTx_REFCLKP/N receive buffer supports both voltage signal input and current signal input, such as LVDS, HCSL. SD_EXTx_REFCLKP/N receive buffer can also be AC coupled out of chip. However, it is recommended to use LVDS source to drive SD_EXTx_REFCLKP/N receive buffer.

For electrical and timing characteristics of SDQx_REFCLKP/N receive buffer and SD_EXTx_REFCLKP/N receive buffer, check CertusPro-NX Family Data Sheet (FPGA-DS-02086).

The reference clock is selected dynamically by a clock MUX block. Table 6.10 shows the usage of this clock MUX primitive (PCSREFMUX). When PCSREFMUX is enabled, CertusPro-NX SerDes/PCS chooses the output clock of PCSREFMUX instead of the default clock source from per-quad package pins (SDQx_REFCLKP/N).

Note that both per-quad package pins (SDQx_REFCLKP/N) and dedicated two package pins (SD_EXT0_REFCLKP/N and SD_EXT1_REFCLKP/N) can only be used as the reference clock for SerDes. It is impossible to use these clock sources to drive FPGA fabric directly.

Table 6.10. PCSREFMUX Usage

Port Name	Descriptions
DIFFIOCLK0	Source from SD_EXT0_REFCLKP/N.
DIFFIOCLK1	Source from SD_EXT1_REFCLKP/N.
PLLCLK0	Source from GPLL output 0 (Left Top GPLL).
PLLCLK1	Source from GPLL output 1 (Right Top GPLL).
CLK	Source from fabric (Only for test).
DIFFIOCLKSEL	Dynamic selection between DIFFIOCLK0 and DIFFIOCLK1. • 1'b1 – DIFFIOCLK1 • 1'b0 – DIFFIOCLK0
CLKSEL	Dynamic clock source selection. • 2'b11 – CLK • 2'b10 – DIFFIOCLK0 or DIFFIOCLK1 • 2'b01 – PLLCLK1 • 2'b00 – PLLCLK0
CLKOUT	Reference clock output.

7. Clocks and Reset

7.1. MPCS Channel Clock Detail

This section describes the detailed implementation of the clock distribution inside MPCS channel. 8B/10B PCS channel, 64B/66B PCS channel and PMA Only channel are discussed separately.

7.1.1. 8B/10B PCS Clock

Figure 7.1 shows the 8B/10B PCS channel clock diagram.

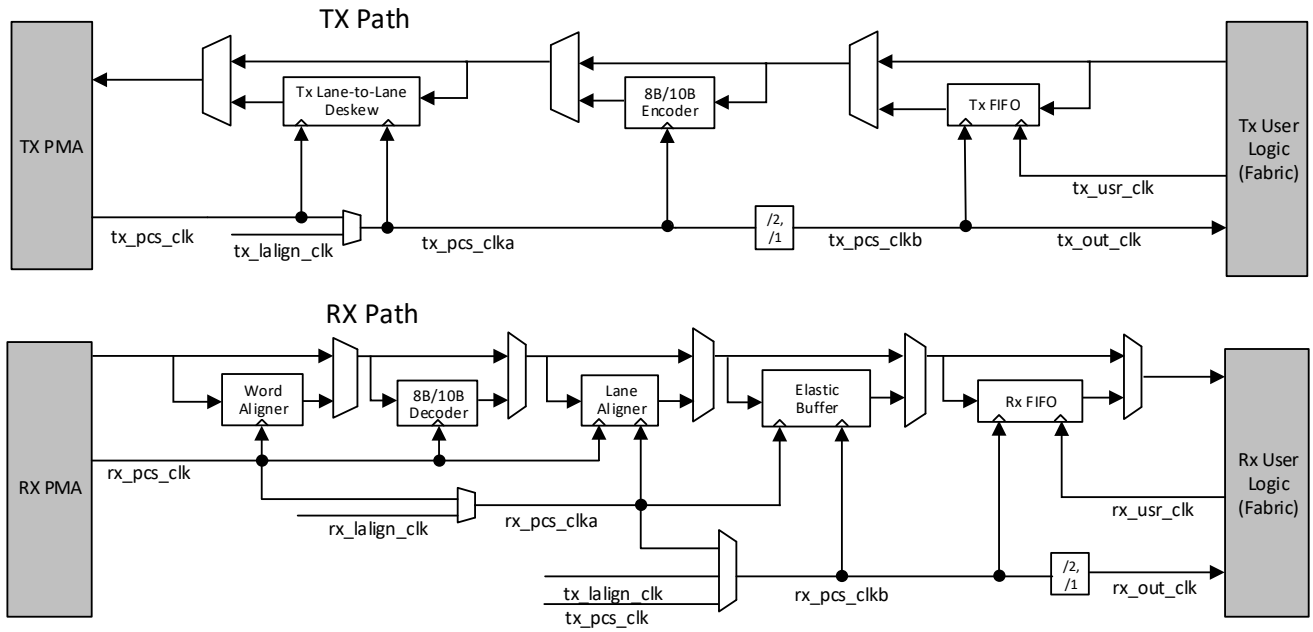

Figure 7.1. 8B/10B PCS Channel Clock Diagram

Table 7.1 shows the detailed 8B/10B PCS Channel clock descriptions.

Table 7.1. 8B/10B PCS Channel Clock

Clock	Direction	Description
PMA Interface (Hardened Connection)		
tx_pcs_clk	N/A	This parallel data clock is generated by PMA Tx macro and used by MPCS to drive Tx data bus. The source of this clock is Tx PLL.
rx_pcs_clk	N/A	This parallel data clock is generated by PMA Rx macro and used by MPCS to receive data from Rx data bus. The source of this clock is the recovered clock from Rx CDR.
Fabric Interface		
tx_out_clk	Output	This clock is directly connected to FPGA global buffer, and thus can drive FPGA clock tree. In multiple-channel cases, channels may share the same clock tree with each other. Therefore, not every tx_out_clk in channels is really used. The source of this clock is selected by MPCS, depending on application cases. This clock can be the divided-by-two version of its source clock.
rx_out_clk	Output	This clock is directly connected to FPGA global buffer and thus can drive FPGA clock tree. In multiple-channel cases, channels may share the same clock tree with each other. Therefore, not every rx_out_clk in channels is really used. The source of this clock is selected by MPCS, depending on application cases. This clock can be the divided-by-two version of its source clock.

Clock	Direction	Description
tx_usr_clk	Input	This clock is a node of fabric clock tree. The data sent by user logic to MPCS is synchronous to this clock.
rx_usr_clk	Input	This clock is a node of fabric clock tree. The data received by user logic from MPCS is synchronous to this clock.
Channel Internal Clock		
tx_pcs_clka	N/A	This clock drives most part of Tx path logics. It may come from tx_pcs_clk in single-channel application or from tx_lalign_clk in multiple-channel application where all channels need to be aligned.
tx_pcs_clkb	N/A	This clock has the same frequency as tx_pcs_clka or half frequency of tx_pcs_clka, depending on input data rate and bus width. This clock and tx_pcs_clka are positive edge aligned, with as little skew as possible between them so that output data from this clock domain can be sampled by tx_pcs_clka directly.
rx_pcs_clka	N/A	This clock may come from rx_lalign_clk, a common clock for all channels in a multi-channel application case. In that case, all channels may need to be aligned before performing clock frequency compensation. In single-channel application, this clock is derived from rx_pcs_clk. This clock drives channel alignment logic and the write side of Elastic Buffer.
rx_pcs_clkb	N/A	This clock may come from rx_pcs_clka, if the Elastic Buffer is bypassed. When Elastic Buffer is enabled, the source of this clock can be tx_pcs_clk that comes from the Tx path of the same channel, or tx_lalign_clk which is the common clock within bonded channels. rx_pcs_clka is used in single-channel case, and tx_pcs_clk is for multiple-channel usage.
Clocks within a Quad		
tx_lalign_clk	N/A	This is a common clock shared by channels within Quad. It is used in multiple-channel application cases. In Tx path, a common clock is used to align all channels in order to minimize the lane-to-lane skew. In Rx path, this common clock is used as local clock to perform clock frequency compensation.
rx_lalign_clk	N/A	This is a common clock shared by channels within Quad. It is used in multiple-channel application cases. In Rx path, a common clock is required to perform channel alignment as well as clock frequency compensation.

7.1.2. 64B/66B PCS Clock

Figure 7.2 shows the 64B/66B PCS Channel clock diagram. Table 7.2 shows the 64B/66B PCS Channel clock detailed descriptions.

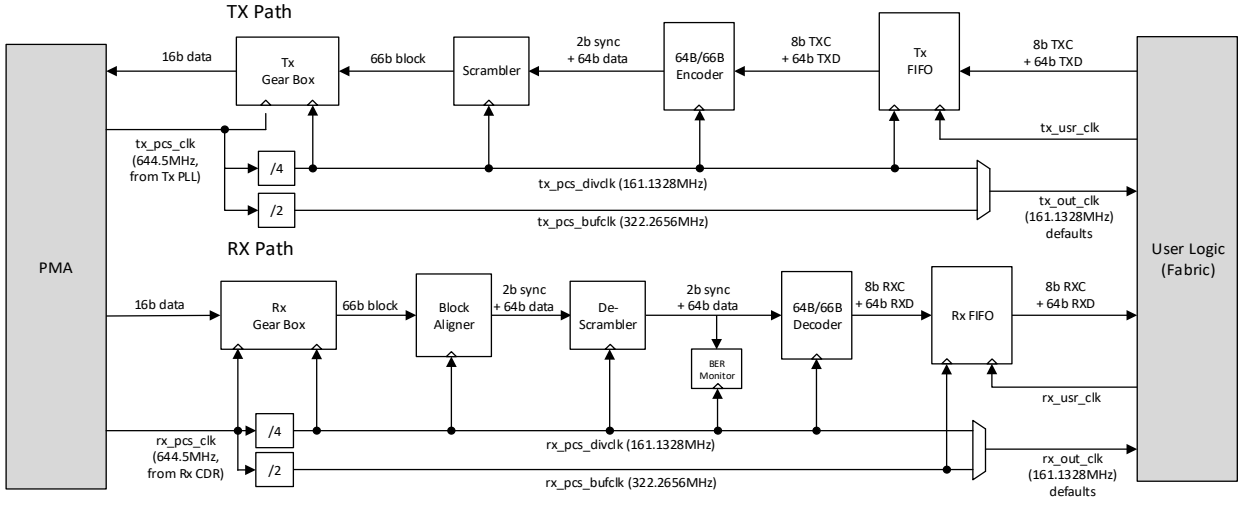

Figure 7.2. 64B/66B PCS Channel Clock Diagram

Table 7.2. 64B/66B PCS Channel Clock

Clock	Direction	Description
PMA Interface (Hardened Connection)		
tx_pcs_clk	N/A	This parallel data clock is generated by PMA Tx macro and used by MPCS to drive Tx data bus. The source of this clock is Tx PLL.
rx_pcs_clk	N/A	This parallel data clock is generated by PMA Rx macro and used by MPCS to receive data from Rx data bus. The source of this clock is the recovered clock from Rx CDR.
Fabric Interface		
tx_out_clk	Output	This clock is directly connected to FPGA global buffer, and thus can drive FPGA clock tree. The source of this clock is selected by MPCS, depending on application cases. This clock can be the divided-by-two version of its source clock.
rx_out_clk	Output	This clock is directly connected to FPGA global buffer, and thus can drive FPGA clock tree. The source of this clock is selected by MPCS, depending on application cases. This clock can be the divided-by-two version of its source clock.
tx_usr_clk	Input	This clock is a node of fabric clock tree. The data sent by user logic to MPCS is synchronous to this clock.
rx_usr_clk	Input	This clock is a node of fabric clock tree. The data received by user logic from MPCS is synchronous to this clock.
Channel Internal Clock		
tx_pcs_divclk	N/A	This clock is divided-by-four version of tx_pcs_clk. It is used to drive most part of the 64B/66B PCS Tx path sub-modules. This clock and tx_pcs_clk are positive edge aligned, with as little skew as possible between them, so that output data from this clock domain can be sampled by tx_pcs_clk directly.
rx_pcs_divclk	N/A	This clock is divided-by-four version of rx_pcs_clk. It is used to drive most part of the 64B/66B PCS Rx path sub-modules. This clock and rx_pcs_clk are positive edge aligned, with as little skew as possible between them, so that output data from rx_pcs_clk clock domain can be sampled by rx_pcs_divclk directly.
tx_pcs_bufclk	N/A	This clock is divided-by-two version of tx_pcs_clk.
rx_pcs_bufclk	N/A	This clock is divided-by-two version of rx_pcs_clk.

7.1.3. PMA Only Mode Clock

Figure 7.3 shows the PMA Only mode clock diagram. Table 7.3 shows the PMA Only mode channel clock detailed descriptions.

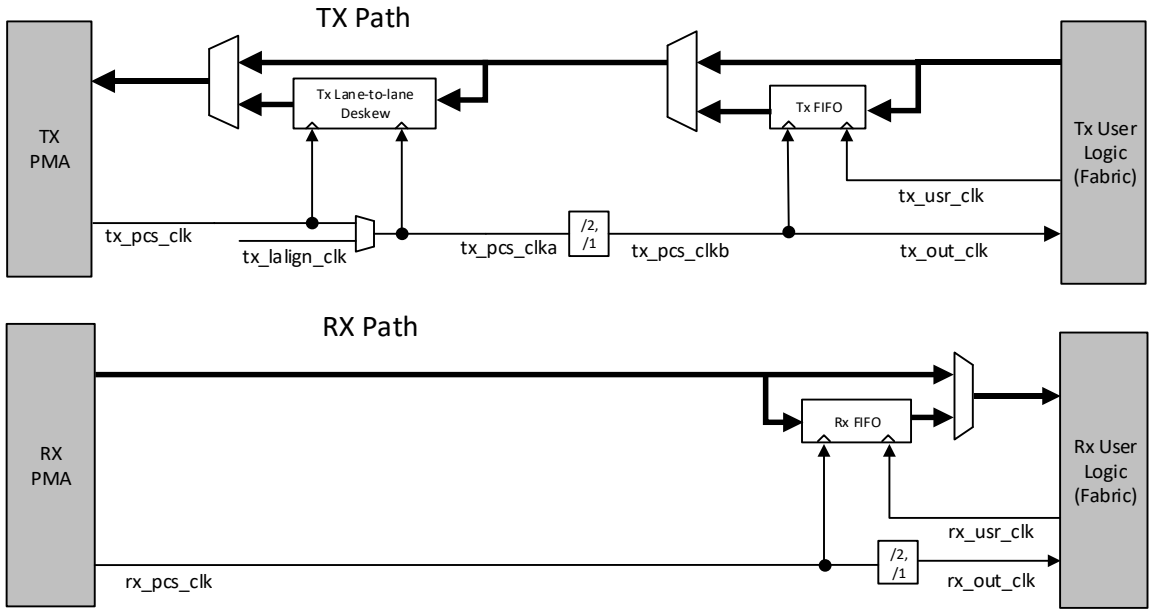

Figure 7.3. PMA Only Mode Clock Diagram

Table 7.3. PMA Only Mode Channel Clock

Clock	Direction	Description
PMA interface (Hardened Connection)		
tx_pcs_clk	N/A	This parallel data clock is generated by PMA Tx macro and used by MPCS to drive Tx data bus. The source of this clock is Tx PLL.
rx_pcs_clk	N/A	This parallel data clock is generated by PMA Rx macro and used by MPCS to receive data from Rx data bus. The source of this clock is the recovered clock from Rx CDR.
Fabric Interface		
tx_out_clk	Output	This clock is directly connected to FPGA global buffer, and thus can drive FPGA clock tree. In multiple-channel cases, channels may share the same clock tree with each other. Therefore, not every tx_out_clk in channels is really used. The source of this clock is selected by MPCS, depending on application cases. This clock can be the divided-by-two version of its source clock.
rx_out_clk	Output	This clock is directly connected to FPGA global buffer, and thus can drive FPGA clock tree. In multiple-channel cases, channels may share the same clock tree with each other. Therefore, not every rx_out_clk in channels is really used. The source of this clock is selected by MPCS, depending on application cases. This clock can be the divided-by-two version of its source clock.
tx_usr_clk	Input	This clock is a node of fabric clock tree. The data sent by user logic to MPCS is synchronous to this clock.
rx_usr_clk	Input	This clock is a node of fabric clock tree. The data received by user logic from MPCS is synchronous to this clock.

Clock	Direction	Description
Channel Internal Clock		
tx_pcs_clka	N/A	This clock drives most part of Tx path logics. It may come from tx_pcs_clk in single-channel application or from tx_lalign_clk in multiple-channel application where all channels need to be aligned.
tx_pcs_clkb	N/A	This clock has the same frequency as tx_pcs_clka or half frequency of tx_pcs_clka, depending on input data rate and bus width. This clock and tx_pcs_clka are positive edge aligned, with as little skew as possible between them so that output data from this clock domain can be sampled by tx_pcs_clka directly.
Clocks within a Quad		
tx_lalign_clk	N/A	This is a common clock shared by channels within Quad. It is used in multiple-channel application cases. In Tx path, a common clock is used to align all channels in order to minimize the lane-to-lane skew.

7.2. MPCS Quad Clock Detail

This section describes the clock distribution inside one Quad in detail. Figure 7.4 shows the MPCS Quad clock distribution diagram. Table 7.4 is the detailed Quad clock description. The Quad Common module implements the clock multiplexing and distributing functionalities for multiple-channel applications.

The tx_pcs_clk and rx_pcs_clk from channel 0 are connected to Quad Common module and outside of MPCS Quad. The tx_pcs_clk[0] and rx_pcs_clk[0] can be connected to all four channels inside the Quad. The tx_pcs_clk and rx_pcs_clk from channel 2 are connected to Quad Common module, too, but tx_pcs_clk[2] and rx_pcs_clk[2] can be connected to channel 2 and channel 3 only by Quad Common module. These clock connections bring some limitation on the implementation for multiple-channel applications. Check the Quad Common section for more details.

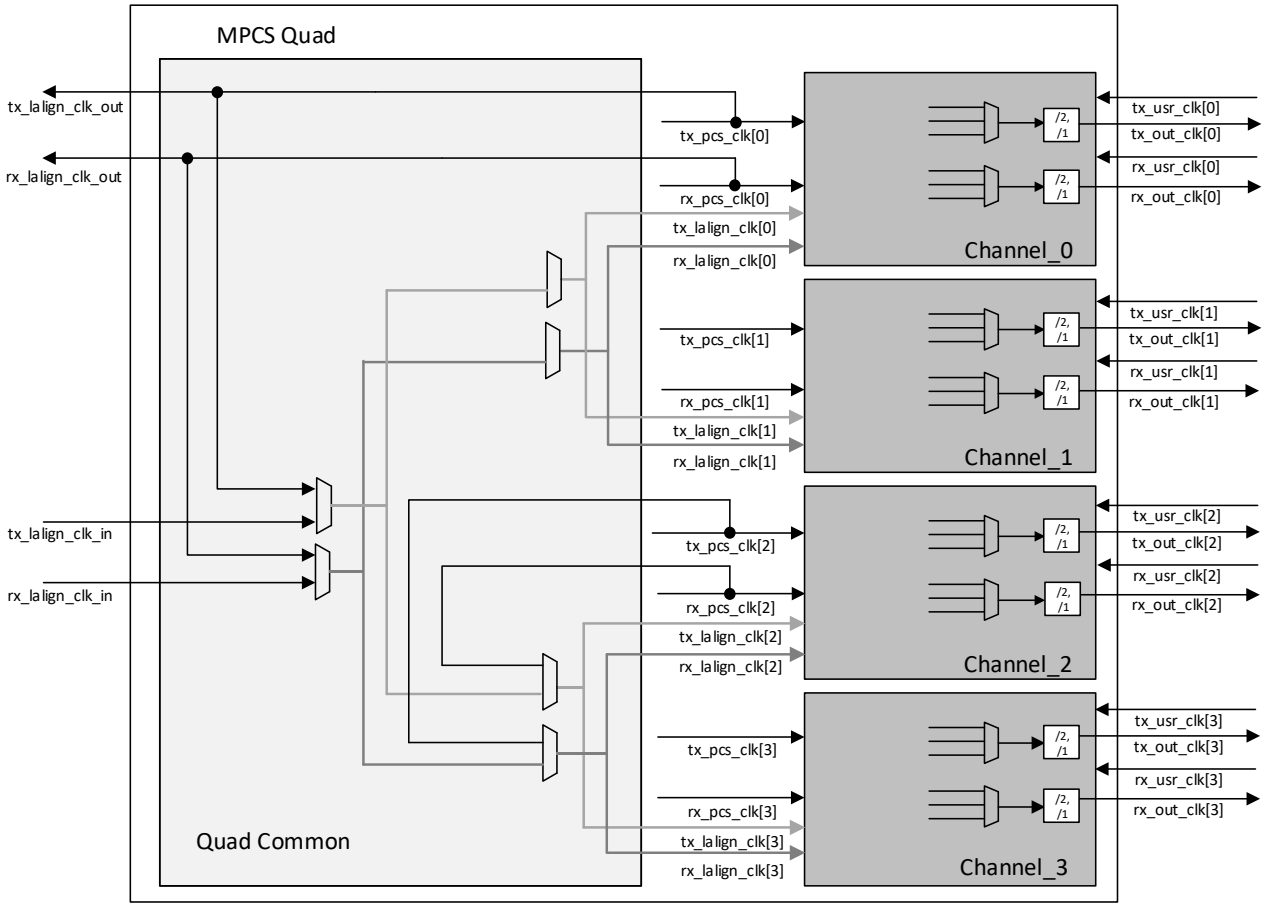

Figure 7.4. Quad Clock Distribution Diagram

Table 7.4. Quad Clock

Clock	Direction	Description
Input and Output		
tx_lalign_clk_out	N/A	This clock is used for lane alignment across Quad boundary. It can drive multiple Quads tx_lalign_clk tree in the application case where more than four lanes (and up to twelve lanes) are combined to form a single link. The source of this clock is always tx_pcs_clk from channel_0 of this Quad. This introduces a usage limitation that channel_0 must always be included in multiple-channel link.
rx_lalign_clk_out	N/A	This clock is used for lane alignment across Quad boundary. It can drive multiple Quads rx_lalign_clk tree in the application case where more than four lanes (and up to twelve lanes) are combined to form a single link. The source of this clock is always rx_pcs_clk from channel_0 of this Quad. This introduces a usage limitation that channel_0 must be included in the multiple-channel link.
tx_lalign_clk_in	N/A	This clock is shared by two adjacent Quads. It can be used to drive local tx_lalign_clk tree.
rx_lalign_clk_in	N/A	This clock is shared by two adjacent Quads. It can be used to drive local rx_lalign_clk tree.

7.3. MPCS Quad-to-Quad Clock Connection

Figure 7.5 shows the clock connection between two adjacent Quads. The output alignment clocks (rx_lalign_clk_out and tx_lalign_clk_out) from Quad 0 drive these two Quads (Quad 0 and Quad 1). Output clocks of Quad 1 are floating and internally gated.

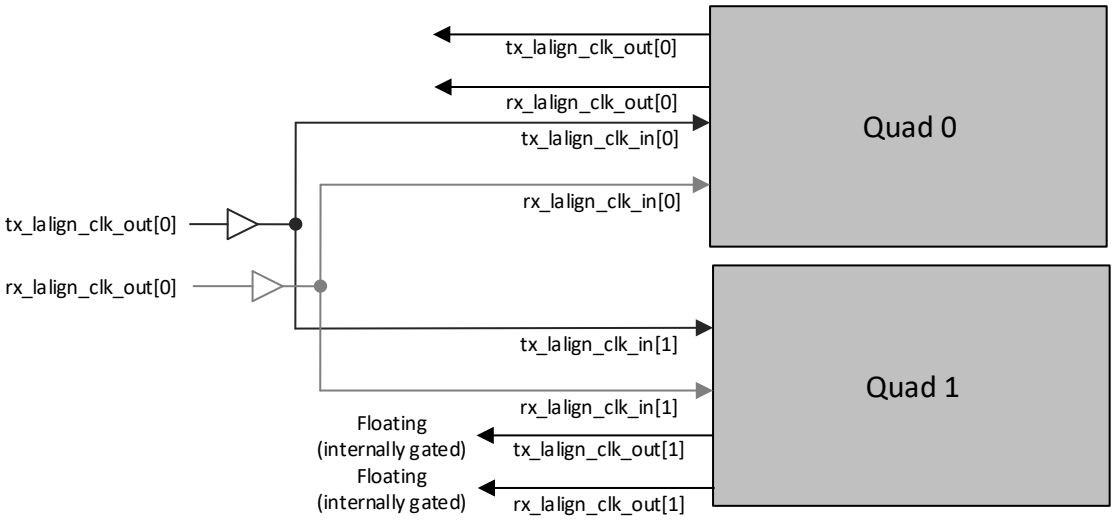

Figure 7.5. Two Quads Clock Connection

For single Quad applications, the output alignment clocks (rx_lalign_clk_out and tx_lalign_clk_out) are floating and input clocks are tied low or high. Refer to Figure 7.6 for more details.

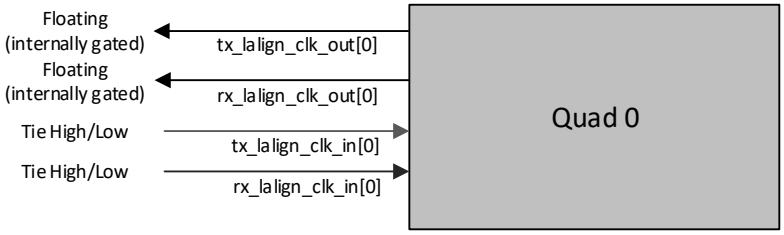

Figure 7.6. Single Quad Clock Connection

7.4. Clock Frequency

This section lists the recommended reference clock frequency, the PLL settings, PMA/PCS internal clock frequency, interface data bus width and clock frequency for Ethernet, SLVS-EC, CoaXPress, DP/eDP and PCIe protocols. Refer to Table 7.5 for the recommended settings for these protocols.

The PLL settings are the same for Tx PLL and Rx CDR, considering that the architecture of Tx PLL and Rx CDR is similar. Refer to the SerDes (PMA) section for the Tx PLL and Rx CDR implementation details inside PMA channel.

Table 7.5. Recommend Settings for Some Protocols

Protocol		Rate (Gbps)	F_{Ref} (MHz)	PLL Setting			PMA Internal Clock		PMA Output	PCS Clk Div	F_{PCS}[7]	PCS Bus Width	Out Clk Div	F_{OUT}[8]	User Data Bus Width
				M[1]	F[2]	N[3]	F_{VCO}[4]	F_{bit}[5]	F_{PMA}[6]						
Ethernet	10GBASE-R	10.3125	161.1328	1	4	16	10312.5	10312.5	644.53	1	644.53	16	4	161.1328	64
	QSGMII	5	125	2	4	10	10000	5000	500	2	250	20	2	125	40
	XAUI	3.125	156.25	2	2	10	6250	3125	312.5	1	312.5	10	2	156.25	20
	SGMII	1.25	125	8	1	10	10000	1250	125	1	125	10	1	125	10
	1KBASE-X	1.25	125	8	1	10	10000	1250	125	1	125	10	1	125	10
SLVS-EC	Grade 3	5[9]	—	2	4	10	~10000	~5000	~500	2	~250	20	2	~125	40
	Grade 2	2.5[9]	—	4	2	10	~10000	~2500	~250	1	~250	10	2	~125	20
	Grade 1	1.25[9]	—	8	1	10	~10000	~1250	~125	1	~125	10	1	~125	10
CoaXPress	—	6.25	156.25	1	2	20	6250	6250	312.5	1	312.5	20	2	156.25	40
	—	5	125	2	4	10	10000	5000	500	2	250	20	2	125	40
	—	3.125	156.25	2	2	10	6250	3125	312.5	1	312.5	10	2	156.25	20
	—	2.5	125	4	2	10	10000	2500	250	1	250	10	2	125	20
	—	1.25	125	8	1	10	10000	1250	125	1	125	10	1	125	10
DP/eDP	HBR3	8.1	135	1	3	20	8100	8100	405	1	405	20	2	202.5	40
	HBR2	5.4	135	1	2	20	5400	5400	270	1	270	20	2	135	40
	HBR	2.7	135	2	2	10	5400	2700	270	1	270	10	2	135	20
	RBR	1.62	108	4	3	5	6480	1620	324	2	162	10	1	162	10
PCIe	Gen3	8	100	1	5	16	8000	8000	500	N/A	N/A	N/A	N/A	N/A	N/A
	Gen2	5	100	2	5	10	10000	5000	500	N/A	N/A	N/A	N/A	N/A	N/A
	Gen1	2.5	100	4	5	5	10000	2500	500	N/A	N/A	N/A	N/A	N/A	N/A

Notes:

1. M can be set as 1, 2, 4 and 8.
2. F can be set as 1, 2, 3, 4, 5 and 6.
3. N can be set as 5, 8, 10, 16 and 20.
4. $F_{VCO}=F_{Ref}*M*F*N$
5. $F_{bit}=F_{Ref}*F*N$
6. $F_{PMA}=F_{Ref}*F$
7. F_{PCS} presents the clock frequency of tx_pcs_clk and rx_pcs_clk.
8. F_{OUT} presents the clock frequency of tx_out_clk and rx_out_clk.
9. The setting in this row is applicable to the whole range of this baud rate. The actual bit rate is determined by the reference clock, which can vary in a small range.

7.5. Application Case

7.5.1. 8B/10B PCS Single Lane Tx Path

Case I-a: Use Tx FIFO

As shown in Figure 7.7, the Tx Lane-to-lane Deskew module is bypassed, considering this a single lane application. The tx_out_clk is used to drive FPGA global clock tree, and a leaf node of this clock tree returning to MPCS is used as the read clock of Tx FIFO. Tx FIFO module works as asynchronous FIFO to eliminate the phase difference between tx_pcs_clkb and tx_usr_clk.

The clock tx_pcs_clkb and tx_out_clk can be the divide-by-2 version of tx_pcs_clka.

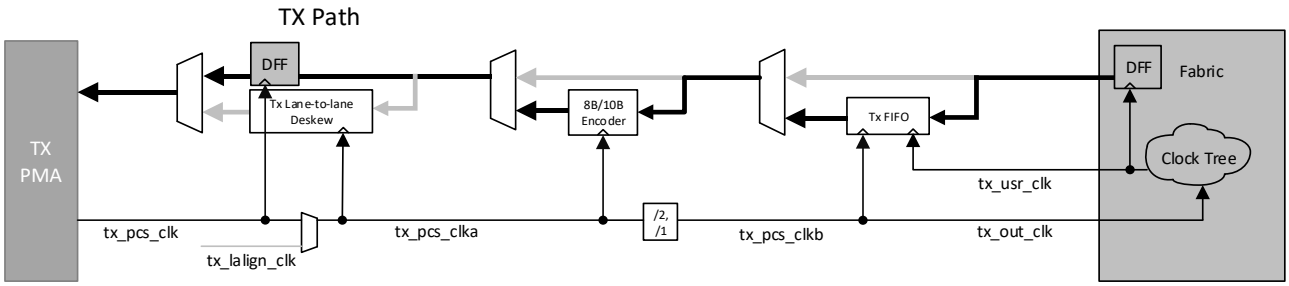

Figure 7.7. Case I-a Clock Structure

Case I-b: Bypass Tx FIFO

As shown in Figure 7.8, in this case, the Tx FIFO module is bypassed to achieve low latency or deterministic latency. A synchronous DFF is used to capture the input data instead of Tx FIFO. The tx_out_clk drives FPGA global clock tree, and a leaf node of this clock tree is used to drive user logic.

What should be noted is that the critical timing constraint must be applied to Fabric-MPCS interface, considering the phase difference between tx_out_clk and tx_usr_clk. The Tx FIFO must be enabled, if the timing analysis results show that the timing constraint cannot be met.

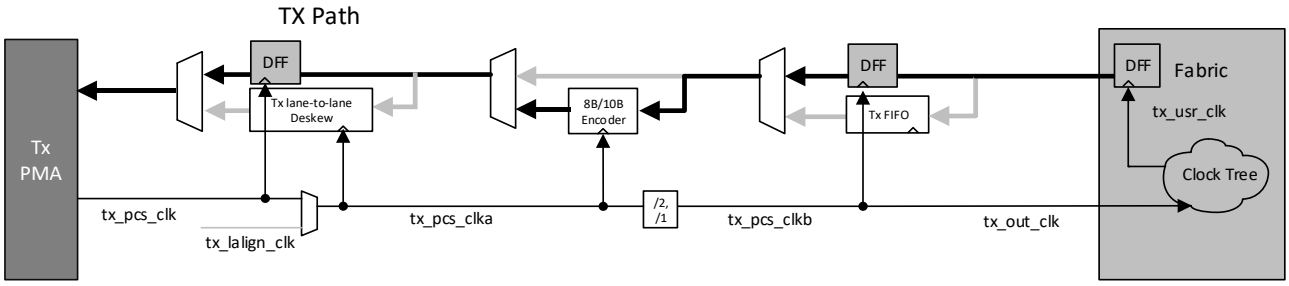

Figure 7.8. Case I-b Clock Structure

7.5.2. 8B/10B PCS Single Lane Rx Path

Case II-a: Use Rx FIFO

As shown in Figure 7.9, the source of rx_pcs_clk is the Rx recovered clock from PMA Rx CDR. The Elastic Buffer is bypassed, because the returned clock rx_usr_clk has exactly the same frequency as the source clock (or half frequency if the clock divider is enabled). The Lane Aligner module is also bypassed, considering it a single lane application. The Rx FIFO module is enabled to eliminate clock phase difference between rx_pcs_clkb and rx_usr_clk.

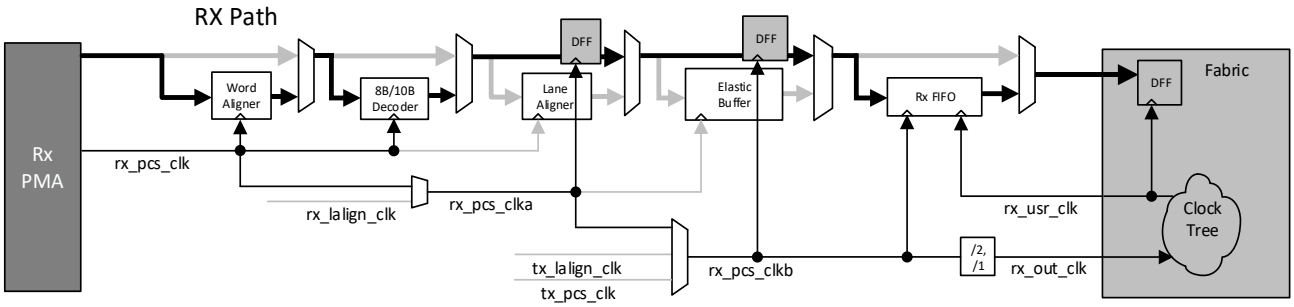

Figure 7.9. Case II-a Clock Structure

Case II-b: Bypass Rx FIFO

As shown in Figure 7.10, in this case, the Rx FIFO module is bypassed to achieve low latency or deterministic latency. A synchronous DFF is used to capture the input data instead of Rx FIFO. The rx_out_clk drives FPGA global clock tree, and a leaf node of this clock tree returning to MPCS is used to drive user logic.

What should be noted is that the critical timing constraint must be applied to Fabric-MPCS interface, considering the phase difference between rx_out_clk and rx_usr_clk. The Tx FIFO must be enabled, if the timing analysis results show that the timing constraint cannot be met.

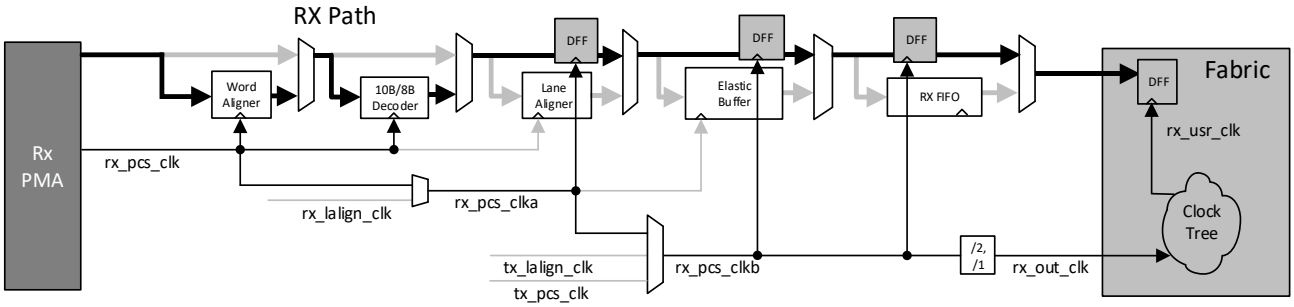

Figure 7.10. Case II-b Clock Structure

Case II-c: Use Elastic Buffer

As shown in Figure 7.11, the rx_pcs_clk is the recovered clock from PMA Rx CDR, the tx_pcs_clk is the generated clock from PMA Tx PLL. The rx_out_clk is used to drive FPGA global clock tree, and the rx_usr_clk is a leaf node of this clock tree returning to MPCS. Rx FIFO module works as asynchronous FIFO to eliminate the phase difference between rx_out_clk and rx_usr_clk.

Lane Aligner module is bypassed, considering this a single lane application. The Elastic Buffer module works as CTC FIFO, to compensate the frequency difference between rx_pcs_clk and tx_pcs_clk.

This mode is mainly designed for applications like PCIe and Ethernet, the rx_pcs_clk is discontinuous in certain scenario. Some protocols also specify this clock tolerance compensation feature, as the rx_pcs_clk may contain too much noise.

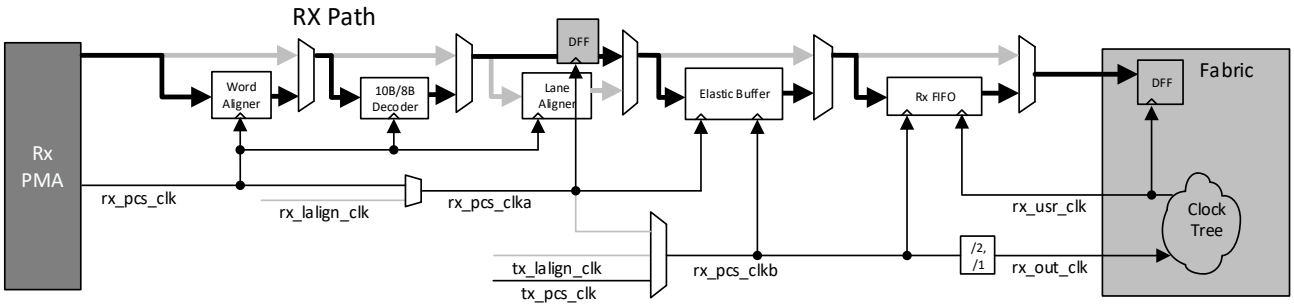

Figure 7.11. Case II-c Clock Structure

7.5.3. 8B/10B PCS Multiple Lane Tx Path

Case III-a: Use Tx FIFO

As shown in Figure 7.12, the tx_pcs_clk0 and tx_pcs_clk1 are the generated clock from the corresponding channel PMA Tx PLL. Tx FIFO modules are enabled to eliminate the clock phase difference between tx_usr_clk and each channel internal clock (tx_out_clk0 and tx_out_clk1). The tx_out_clk0 is used to drive FPGA global clock tree, and a leaf node of this clock tree returning to MPCS is used as the read clock of Tx FIFO.

However, an uncertain one clock cycle latency is introduced by Tx FIFO module. A common clock (tx_lalign_clk) from Quad Common module is used by all lanes to replace tx_pcs_clk to drive the Tx path. The Tx Lane-to-lane Deskew module works as asynchronous FIFO to eliminate this uncertain latency and the clock phase difference between tx_lalign_clk and tx_pcs_clk.

Figure 7.12 shows the two-lane application. The clock structure of multiple-lane applications (more than two) is similar to this two-lane application.

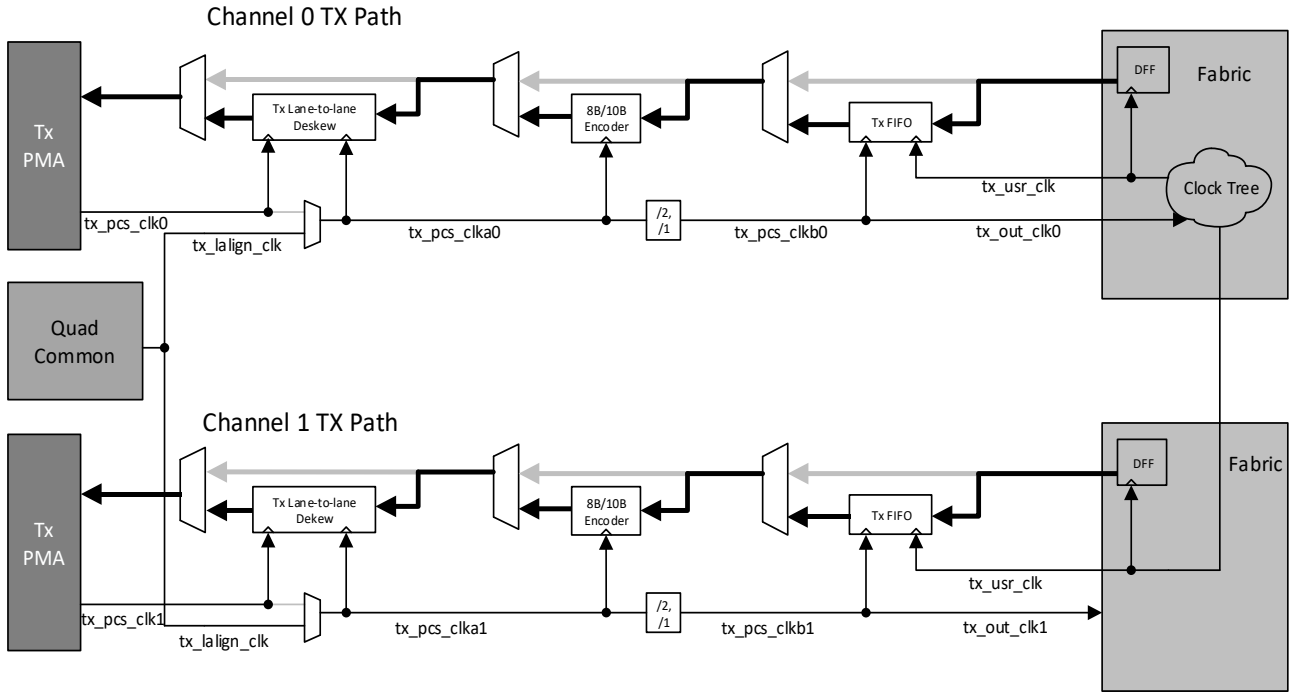

Figure 7.12. Case III-a Clock Structure

7.5.4. 8B/10B PCS Multiple Lane Rx Path

Case IV-a: Bypass Elastic Buffer

As shown in Figure 7.13, the rx_pcs_clk0 and rx_pcs_clk1 are the recovered clock from the corresponding channel PMA Rx CDR. Rx FIFO modules are enabled to eliminate the clock phase difference between rx_usr_clk and each channel internal clock (rx_out_clk0 and rx_out_clk1). The rx_out_clk0 is used to drive FPGA global clock tree, and a leaf node of this clock tree returning to MPCS is used as the read clock of Rx FIFO.

The rx_lalign_clk from Quad Common module is used by all lanes to replace each channel rx_pcs_clk to drive the Rx path. The Lane Aligner modules are enabled for Rx lane-to-lane Deskew and for eliminating the phase difference between rx_pcs_clk and rx_lalign_clk.

Elatstic Buffer module is bypassed, considering no frequency difference between rx_pcs_clka and rx_usr_clk.

Figure 7.13 shows the two-lane application. The clock structure of multiple-lane applications (more than two) is similar to this two-lane application.

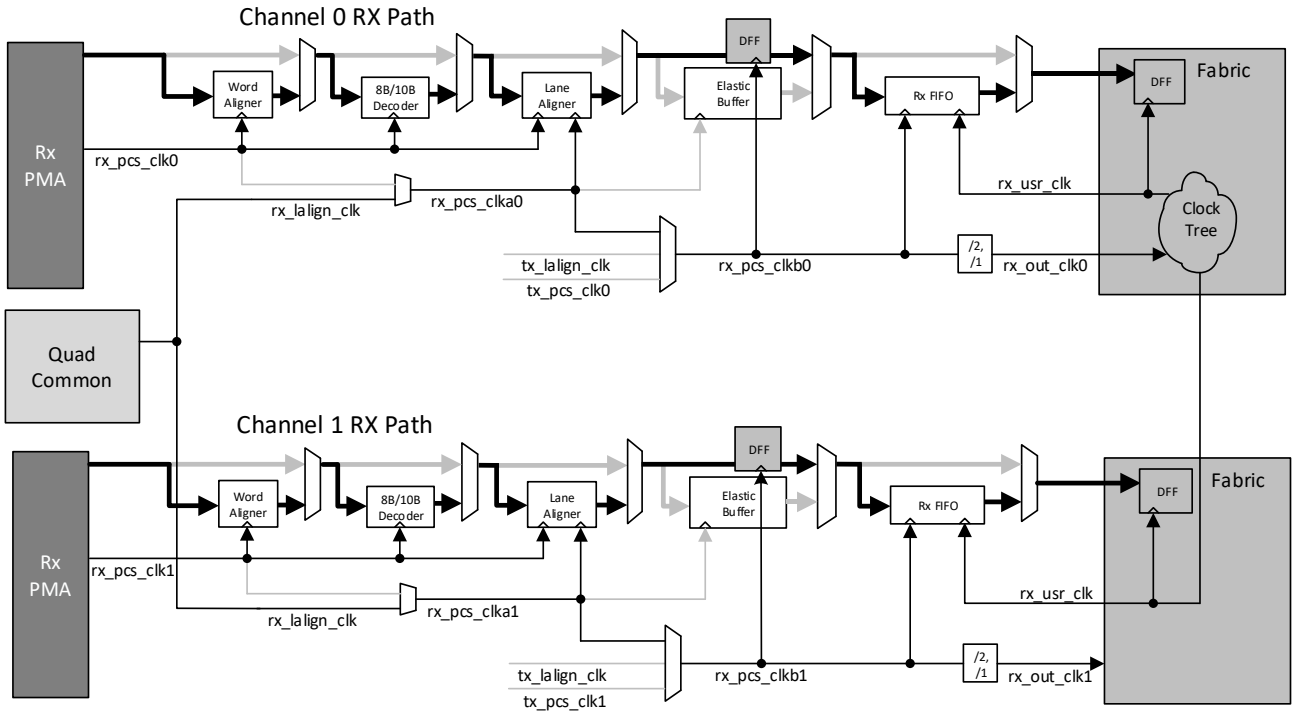

Figure 7.13. Case IV-a Clock Structure

Case IV-b: Use Elastic Buffer

As shown in Figure 7.14, the rx_pcs_clk0 and rx_pcs_clk1 are the recovered clock from corresponding channel PMA Rx CDR. Rx FIFO modules are enabled to eliminate the clock phase difference between rx_usr_clk and each channel internal clock (rx_out_clk0 and rx_out_clk1). The rx_out_clk0 is used to drive FPGA global clock tree, and a leaf node of this clock tree returning to MPCS is used as the read clock of Rx FIFO.

The rx_lalign_clk from Quad Common module is used by all lanes to replace each channel rx_pcs_clk to drive the Rx path. The Lane Aligner modules are enabled for Rx lane-to-lane Deskew and for eliminating the phase difference between rx_pcs_clk and rx_lalign_clk.

The Elastic Buffer module works as CTC FIFO to compensate the frequency difference between rx_pcs_clka and tx_lalign_clk. The tx_lalign_clk is from Quad Common module and originated from certain channel PMA Tx PLL.

This mode is mainly designed for applications like PCIe and Ethernet. The rx_pcs_clk is discontinuous in certain scenario. Some protocols also specify this clock tolerance compensation feature, considering the rx_pcs_clk containing too much noise.

Figure 7.14 shows the two-lane application. The clock structure of multiple-lane applications (more than two) is similar to this two-lane application.

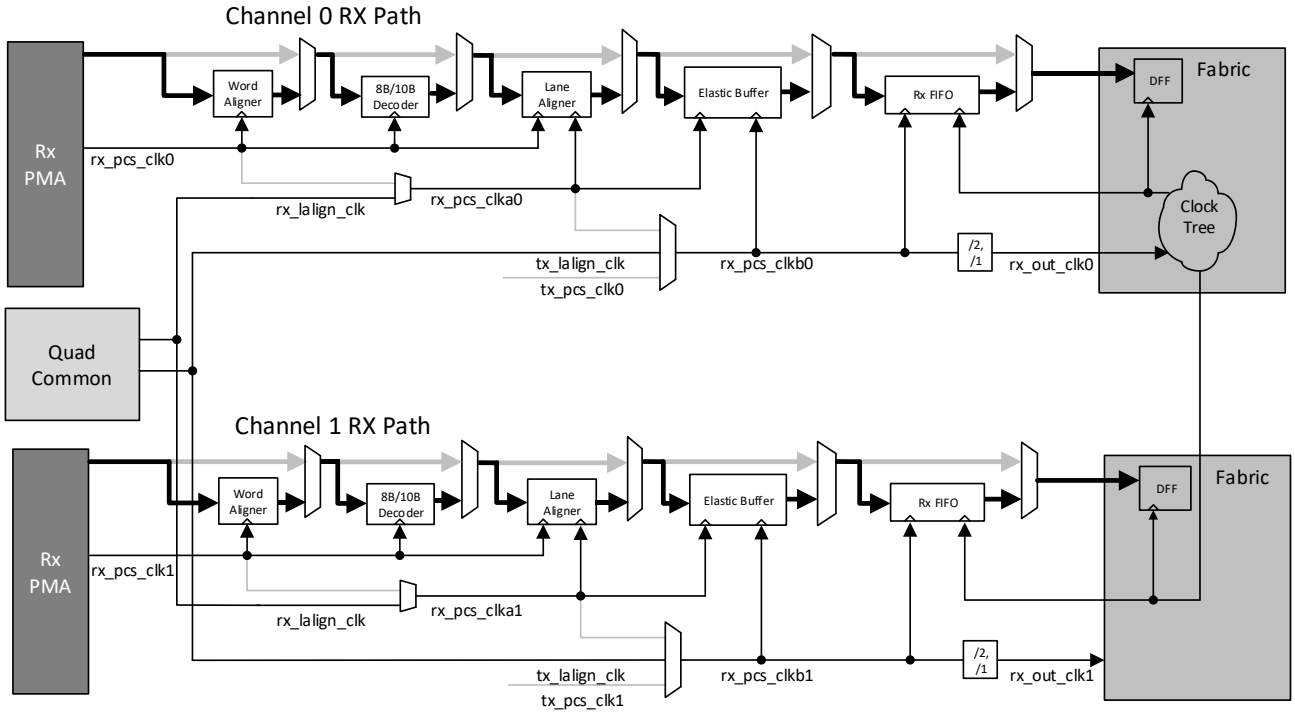

Figure 7.14. Case IV-b Clock Structure

7.5.5. 64B/66B PCS with GPLL

Case V-a: 64B/66B PCS with GPLL

As shown in Figure 7.15, the tx_pcs_clk is generated clock from PMA Tx PLL, and the rx_pcs_clk is recovered clock from PMA Rx CDR. The source of tx_out_clk and rx_out_clk can be divided-by-two version or divided-by-four of tx_pcs_clk and rx_pcs_clk respectively.

A General PLL (GPLL) is recommended to be used in fabric to generate the 156.25 MHz clock to drive both Tx XGMII data path and Rx XGMII data path.

All modules inside 64B/66B PCS channel are recommended to be enabled, considering this 64B/66B PCS being designed for Ethernet 10GBASE-R only. However, certain module can be disabled or bypassed only for test purposes.

Figure 7.15 shows an example of all modules being enabled. The tx_out_clk is used to drive GPLL to generate 156.25 MHz tx_usr_clk and rx_usr_clk. Tx FIFO is enabled to eliminate the clock frequency difference between tx_pcs_divclk and tx_usr_clk. Rx FIFO is enabled to eliminate the clock frequency difference between rx_pcs_divclk and rx_usr_clk. Check Tx FIFO section for the Tx FIFO write timing diagram, and the Rx FIFO section for the Rx FIFO read timing diagram.

The Rx FIFO also works as CTC FIFO, to implements the clock tolerance compensation required by Ethernet 10GBASE-R protocol.

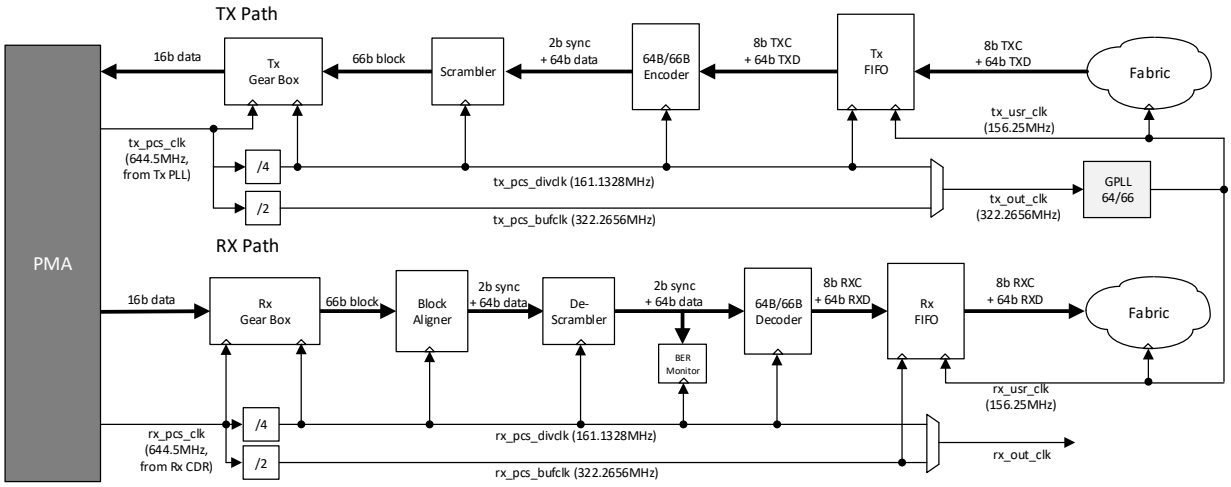

Figure 7.15. 64B/66B PCS with Using GPLL

7.5.6. 64B/66B PCS without GPLL

Case VI-a: Rx Path Clock Compensation is Enabled

Figure 7.16 shows an example of GPLL not being used. The tx_out_clk is used to drive both tx_usr_clk and rx_usr_clk directly. Check the Tx FIFO section for the Tx FIFO write timing diagram, and the Rx FIFO section for the Rx FIFO read timing diagram. The Rx FIFO also works as CTC FIFO, to implements the functionality of clock tolerance compensation.

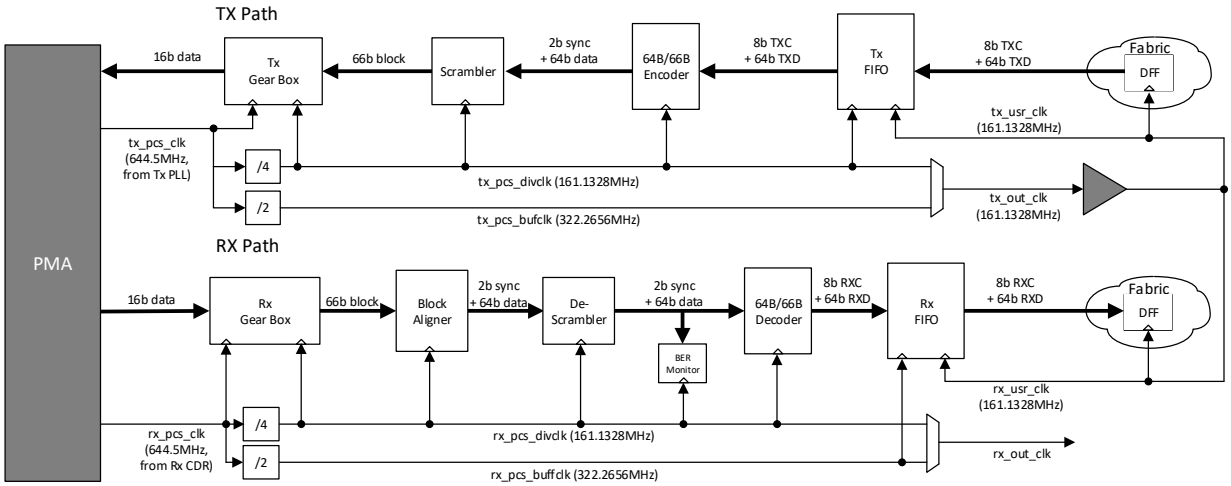

Figure 7.16. 64B/66B PCS without using GPLL (Case I)

Case VI-b: Rx Path Clock Compensation is Disabled

Figure 7.17 shows an example of GPLL not being used and CTC functionality being disabled. The tx_out_clk is used to drive tx_usr_clk, and rx_out_clk is used to drive rx_usr_clk respectively. Check the Tx FIFO section for the Tx FIFO write timing diagram, and the Rx FIFO section for the Rx FIFO read timing diagram. The clock tolerance compensation is unnecessary, considering no clock frequency difference between rx_out_clk and rx_usr_clk.

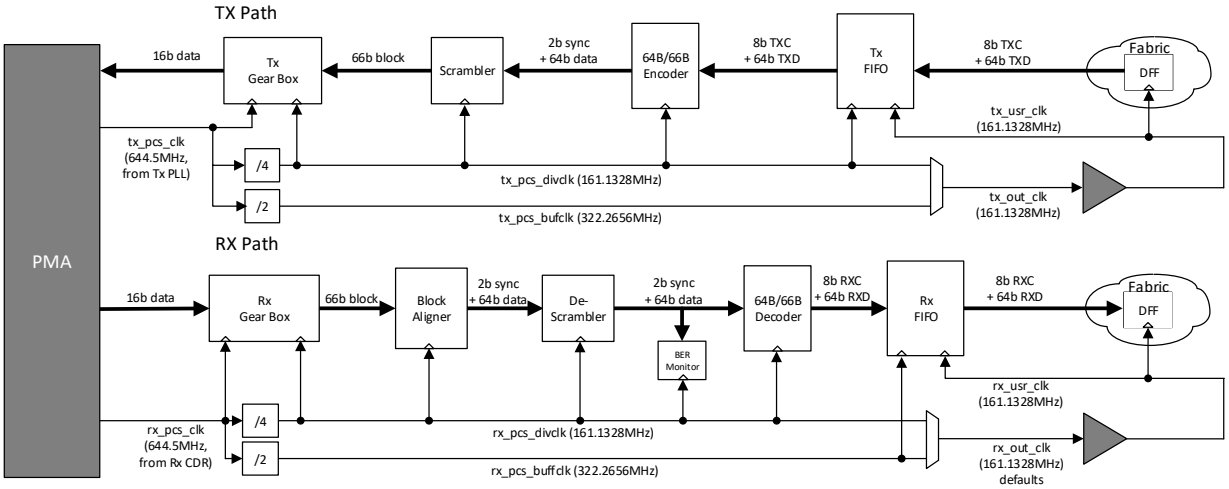

Figure 7.17. 64B/66B PCS without Using GPLL (Case II)

7.6. PMA Clock Divider

Inside the PMA Controller, two PMA clock dividers are implemented for Tx/Rx gearbox. The gearing can be 1:1, 2:1 or 4:1 for both Tx path and Rx path, with considerations below:

- PMA data width is 16-bit or 20-bit.
- Gearing must be 1:1.
- PMA data width is 8-bit or 10-bit.
- Gearing can be 1:1 or 2:1.
- PMA data width is 5-bit.
- Gearing can be 1:1, 2:1, or 4:1.

These PMA clock dividers can be configured through IP GUI, primitive parameter or register accessing.

7.7. SerDes/PCS Reset

7.7.1. Reset and Power Control Signals

Table 7.6 shows the reset and power control signals for CertusPro-NX SerDes/PCS. NL represents the number of lanes.

Table 7.6. Reset and Power Control Signals

Port Name	Width	Description
Reset Signals		
mpcs_tx_pcs_rstn_i	NL	Active low signal used to reset the Tx path of MPCS channel. Exists only in MPCS mode.
mpcs_rx_pcs_rstn_i	NL	Active low signal used to reset the Rx path of MPCS channel. Exists only in MPCS mode.
mpcs_perstn_i	NL	Fundamental reset. Triggers PCS auto calibration. Exists only in MPCS mode.
epcs_tx_pcs_rstn_i	NL	Active low signal used to reset the Tx path of MPCS channel. Exists only in PMA Only mode.
epcs_rx_pcs_rstn_i	NL	Active low signal used to reset the Rx path of MPCS channel. Exists only in PMA Only mode.
epcs_rstn_i	NL	Fundamental reset that triggers PCS auto calibration. Exists only in PMA Only mode.

Port Name	Width	Description
lmmi_resetn_i	NL	Active Low LMMI Reset.
Power Control Signals		
mpcs_pwrdn_i	2*NL	This signal is used to put PMA in power down mode. This signal has only three states. Exists only in MPCS mode. • 2'b11 – deep low-power state. • 2'b10 – low-power state. • 2'b00 – operational state.
epcs_pwrdn_i	2*NL	This signal is used to put PMA in power down mode. This signal has only three states. Exists only in PMA Only mode. • 2'b11 – deep low-power state. • 2'b10 – low-power state. • 2'b00 – operational state.

7.7.2. Reset Generation

Reset generation and distribution are handled by the clocks and resets block. The reset signals from user logic are active low and asynchronous. The reset logic is shown in Figure 7.18 and the corresponding table in Table 7.7. Before device configuration done and the SerDes power stable is stable, CertusPro-NX SerDes/PCS is always in reset conditions. Note that the power supply for channel 1 (in each Quad) must be always on and stable, even though channel 1 may not be used in this Quad. Refer to Unused Quad/Channel and Power Supply section for detail.

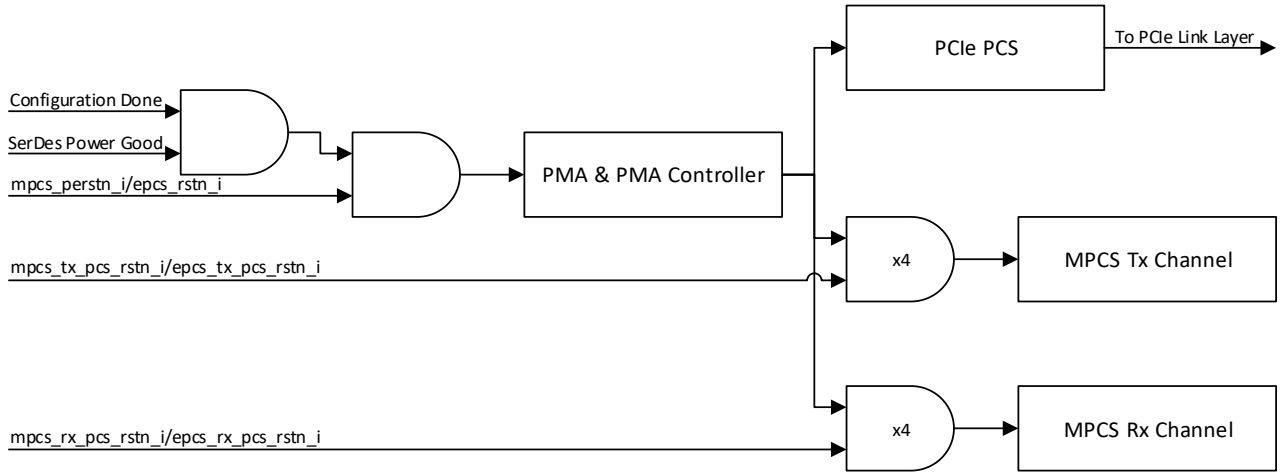

Figure 7.18. CertusPro-NX SerDes/PCS Reset Generation

Table 7.7. Reset Table for SerDes/PCS Quad

Reset Signal	PMA	PMA Controller	PMA Tx PLL	PMA Rx PLL (CDR)	MPCS Tx	MPCS Rx
Configuration Done	√	√	√	√	√	√
SerDes Power Good	√	√	√	√	√	√
mpcs_perstn_i	√	√	√	√	√	√
epcs_rstn_i	√	√	√	√	√	√
txpllrst[3:0] (PMA reg66 bit4)	—	—	√	—	√	—
rxpllrst[3:0] (PMA reg66 bit5)	—	—	—	√	—	√
mpcs_tx_pcs_rstn_i[3:0]	—	—	—	—	√	—

Reset Signal	PMA	PMA Controller	PMA Tx PLL	PMA Rx PLL (CDR)	MPCS Tx	MPCS Rx
epcs_tx_pcs_rstn_i[3:0]	—	—	—	—	√	—
mpcs_rx_pcs_rstn_i[3:0]	—	—	—	—	—	√
epcs_rx_pcs_rstn_i[3:0]	—	—	—	—	—	√
tx_mpcs_rst[3:0] (PCS reg10 bit1)	—	—	—	—	√	—
rx_mpcs_rst[3:0] (PCS reg20 bit1)	—	—	—	—		√

7.7.3. Reset Sequence

This section covers the recommended reset sequence for MPCS mode and PMA Only mode. For reset sequence required by PIPE mode and PCI Express Hard IP mode, refer to PCIe X4 IP Core - Lattice Radiant Software (FPGA-IPUG-02126).

7.7.3.1. MPCS Mode

Figure 7.19 shows the MPCS mode reset sequence timing diagram for Tx path.

After power up, the valid and stable clocks should be connected to lmmi_clk_i and mpcs_clkin_i ports. Then lmmi_resetn_i should be released by user logic to allow INIT bus to start PMA and PCS register initialization. During stage A, main power is applied to the SerDes/PCS and no clock is valid yet. All reset signals should be asserted by user logic.

User logic can release mpcs_perstn_i to inform SerDes/PCS module that user logic is ready and no more registers need to be configured from user logic, which requires PMA and PMA Controller under the reset state. During stage B, PMA, PMA Controller, and MPCS are under the reset state, but register access bus is working. After mpcs_perstn_i is released by user logic, SerDes/PCS module checks whether all the following conditions are satisfied. Then, SerDes/PCS module can decide if PMA and PMA Controller reset should be released.

- Channel PMA analog powers (VCCSDx and VCCPLLSDx) are stable.
- Global Set/Reset (GSR) has been released by configuration module.
- Reference clock is ready and stable.

During stage C, mpcs_tx_out_clk can be available but not accurate when Tx PLL locks on the reference clock. PMA Controller works with PMA on calibration, mpcs_ready_o asserts when PMA calibration is done. PMA calibration may fail if there are some issues on the external PMA PLL filter. Refer to PMA PLL Filter for detailed requirements about the PMA PLL filter. What should be noted is that it is forbidden to write any PMA registers during stage C. The mpcs_phyrdy_o signal is released when below conditions have been met:

- PMA calibration is done.
- Tx common mode voltage is guaranteed.
- PMA is not in low power states.
- PMA not in rate change negotiation stage.

Once mpcs_phyrdy_o is asserted, user logic can assert mpcs_txval_i to let PMA quit electrical idle state. Finally, user logic can release mpcs_tx_pcs_rstn_i and put valid data onto mpcs_tx_ch_din_i port.

During stage F, PMA, PMA Controller and MPCS are in the normal working mode and are ready to transmit data.

Figure 7.20 shows the MPCS mode reset sequence timing diagram for Rx path.

Reset sequence timing for Rx path is the same as the timing for Tx path during stage A and stage B.

During stage C, mpcs_rx_out_clk can be available but not accurate when Rx CDR PLL locks on the reference clock. During stage D, Rx CDR PLL is trying to lock on input data stream. mpcs_rxval can be asserted once Rx CDR locks on the input data stream.

User logic can release mpcs_rx_pcs_rstn_i when mpcs_rxval_o has been asserted to receive valid data from mpcs_rx_ch_dout_o port.

During stage F, PMA, PMA Controller and MPCS are in the normal working mode and are ready to receive data.

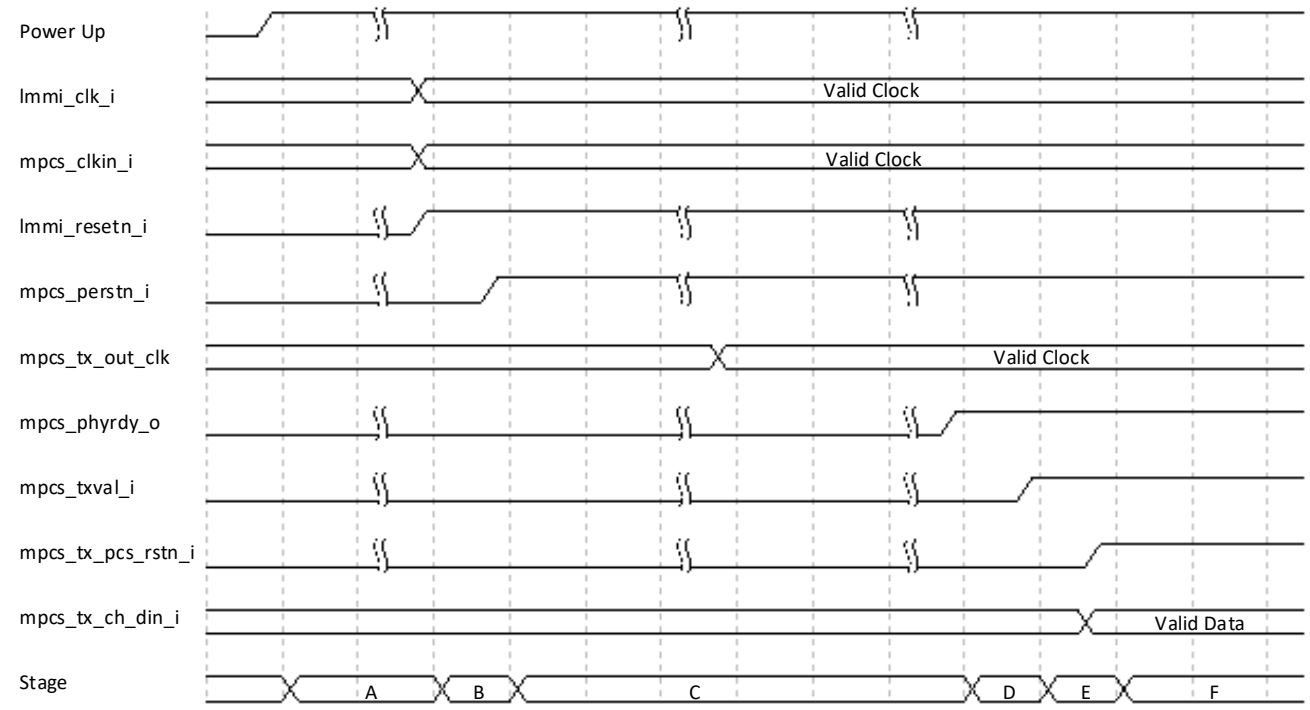

Figure 7.19. MPCS Mode Reset Sequence (Tx Path)

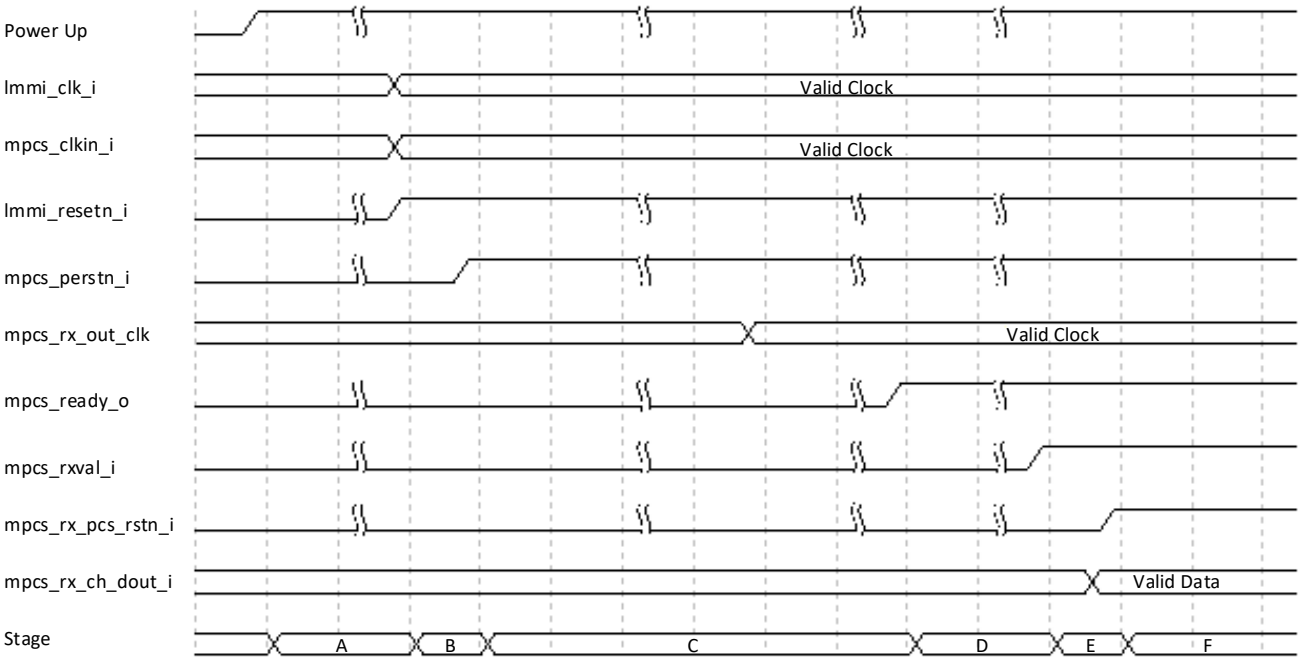

Figure 7.20. MPCS Mode Reset Sequence (Rx Path)

7.7.3.2. PMA Only Mode

EPCS mode reset sequence timing diagram is similar to that of the MPCS mode, but several signal names are different. Refer to Figure 7.21 and Figure 7.22 for detailed timing diagrams.

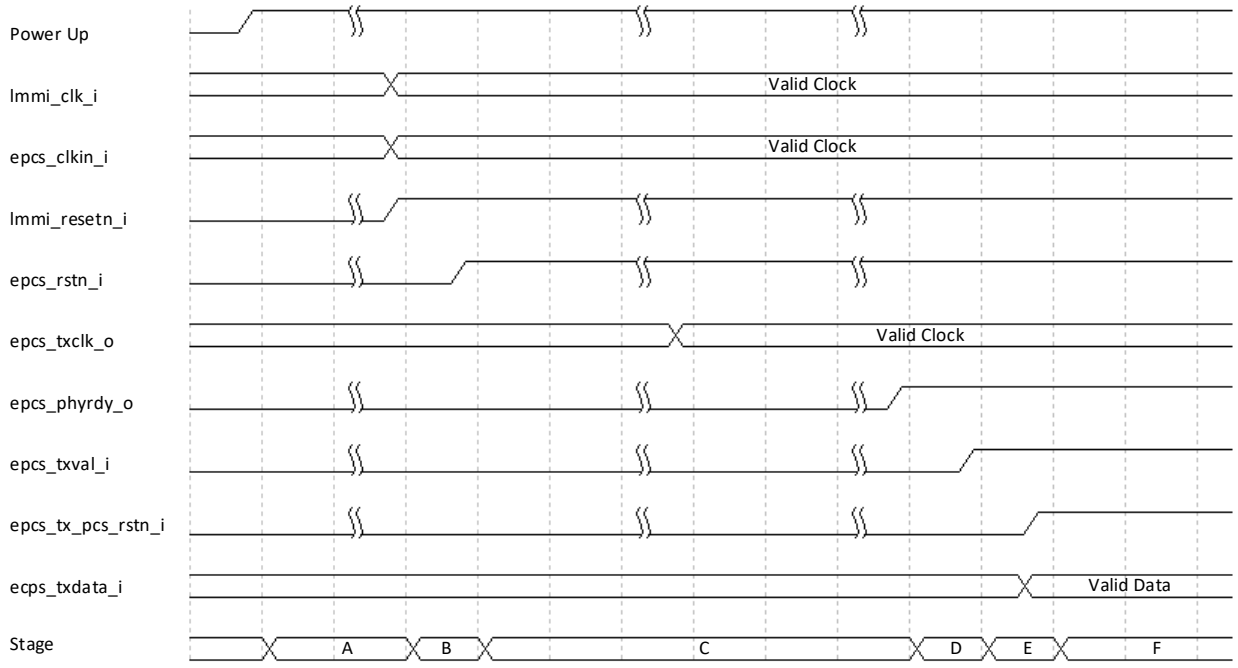

Figure 7.21. EPCS Mode Reset Sequence (Tx Path)

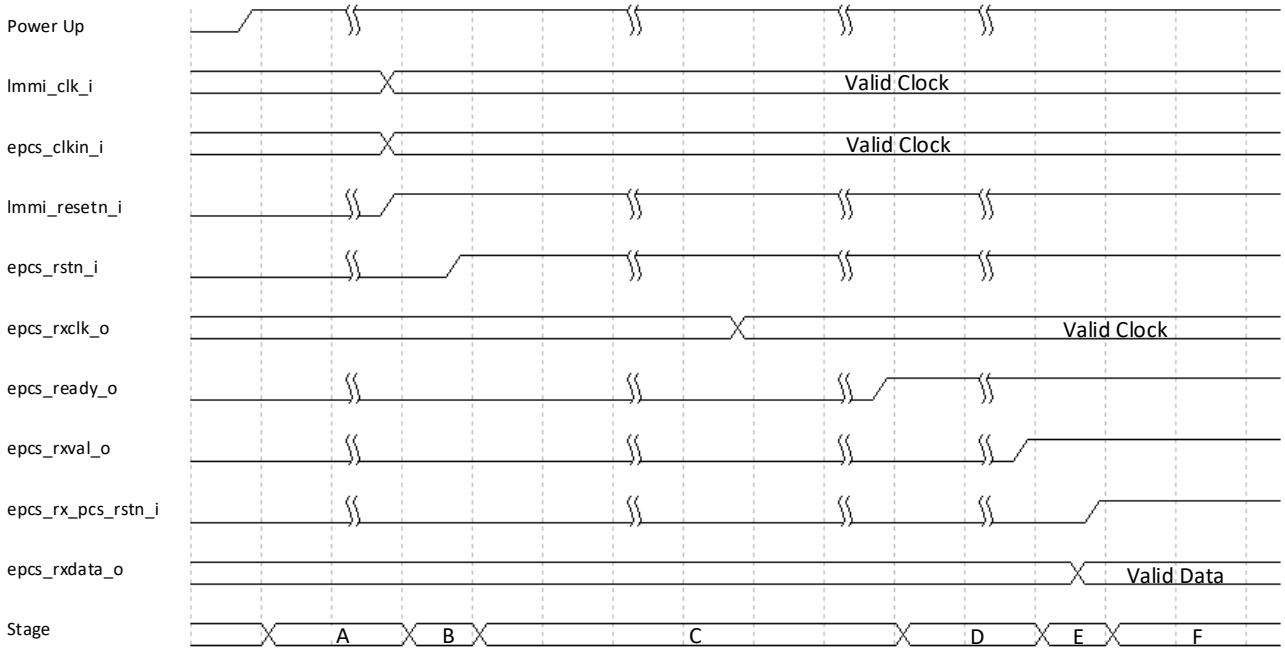

Figure 7.22. EPCS Mode Reset Sequence (Rx Path)

8. SerDes Equalization

The interconnection between the transmitter and receiver device acts as a filter at typical baud rates, and distorts the serial data signal to varying extents. Figure 8.1 shows the signal distortion for the typical backplane applications. The signal out from the transmitter side is a clean digital signal, but the signal waveform is significantly distorted at the receiver side. The frequency response function for this interconnection is a low-pass filter, considering the insertion loss increases with the increase of the frequency of signal. Signal distortion occurs because the signal baud rate is above the cut-off frequency for this low-pass filter.

As data rates get higher, the Unit Interval (UI, or bit time) becomes smaller, with the result that it is increasingly to avoid having the value in one bit time affect the value in another bit time. When a signal has been held at the same voltage for several bit times, and when sending several bits in a row of the same polarity, the channel has more time to approach the target voltage. The resulting higher voltage makes it difficult to change to the opposite value within the required time when the polarity does change. AC coupling sometimes exacerbates this phenomenon, considering the charging effect of capacitance. What is more, the interconnection low-pass filter makes it more difficult for the receiver to recognize the real value of the signal, because the high frequency component of the repeating value signal is less than the continuous flip signal. This problem of previous bits affecting subsequent bits is referred to as Inter-Symbol Interference (ISI).

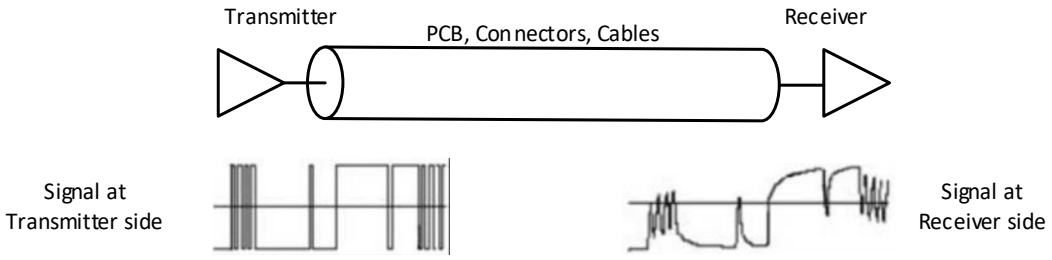

Figure 8.1. Signal Distortion for Typical Backplane Application

To solve these Signal Integrity (SI) problems, data signal should be equalized at the transmitter side. Equalization at the receiver side is also necessary sometimes even though equalization at the transmitter side is always required for high-speed digital signals. De-emphasis and pre-emphasis are common equalization technologies used by most high-speed serial applications. De-emphasis reduces the voltage for repeated bits in a data bitstream. Pre-emphasis is similar to de-emphasis, which intentionally overdrives at the first bit for repeated bits in a data bitstream. Both de-emphasis and pre-emphasis can be implemented by 2-tap Finite Impulse Response (FIR). In some documents, this FIR is named as Feed Forward Equalizer (FFE). For data rate higher than 6 Gbps, the 3-tap FIR based equalizer is usually used by a transmitter. The Continuous-Time Linear Equalization (CTLE) and Decision Feedback Equalization (DFE) are common implementations in receiver equalizer designs.

Figure 8.2 shows the typical backplane application with Tx equalizer. Tx equalizer distorts the signal with de-emphasis technology, such that the resulting signal at the receiver input is a clean waveform and is easier to be recognized.

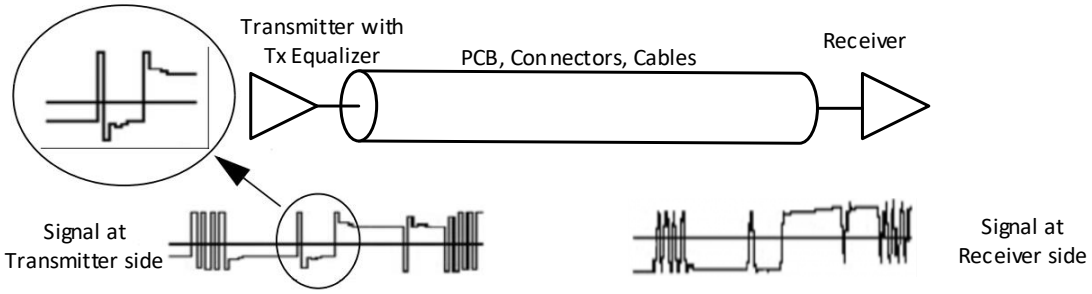

Figure 8.2. Typical Backplane Application with Tx Equalizer

Figure 8.3 shows the typical backplane application with Rx equalizer. Despite the signal distortion at the input of the receiver, Rx equalizer corrects for the distortion and produces a clean waveform. CTLE reduces the low frequency component from the received signal, which attenuates by a lower amount on the transmission line. The high frequency component of the received signal can also be amplified by CTLE sometimes. The DFE is similar to the Tx equalizer, usually implemented by multi-tap FIR. The combination of CTLE and DFE tolerates more channel loss than either equalizer alone when they have been tuned properly.

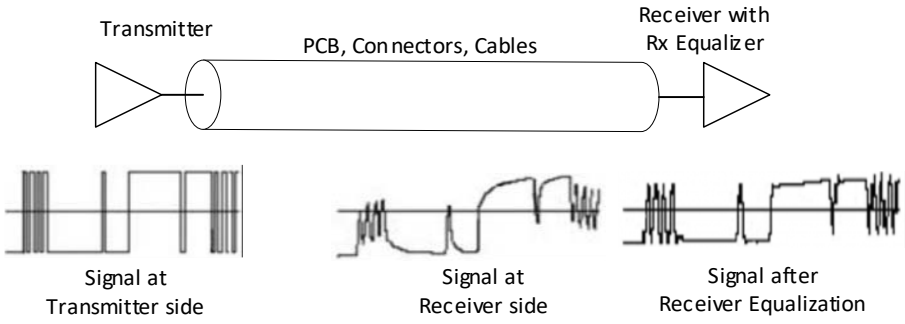

Figure 8.3. Typical Backplane Application with Rx Equalizer

CertusPro-NX device implements both Tx equalizer and Rx equalizer for multiple protocols supporting. Tx equalizer is based on 3-tap FIR, while the Rx equalizer is based on CTLE and 1-tap DFE. Below sections descript the detailed usage of CertusPro-NX Tx equalizer and Rx equalizer.

8.1. Tx Equalization

Figure 8.4 shows the block diagram of transmit equalizer. Both pre-cursor ratio (C_{-1}) and post-cursor ratio (C_{+1}) are negative and $|C_{-1}| + C_0 + |C_{+1}| = 1,\ |C_{-1}| + |C_{+1}| < 0.5$

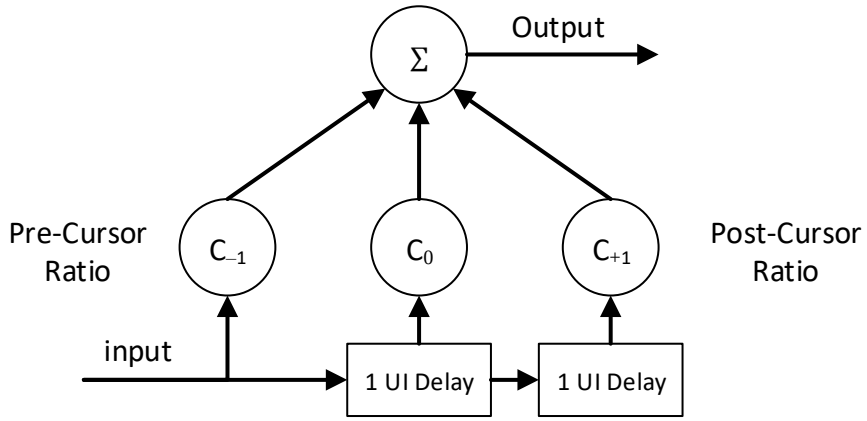

Figure 8.4. Transmit Equalizer Block Diagram

Table 8.1. Description of Tx Voltage Levels

Tx Voltage Levels	Definition	Normalized Amplitude
Va	The amplitude of the first bit of a repeated bitstream (given 00001111 pattern).	$1 + 2 \times C_{-1}$
Vb	The amplitude between the first and the last bits in a repeated bitstream.	$1 + 2 \times C_{-1} + 2 \times C_{+1}$
Vc	The amplitude of the last bit of a repeated bitstream (given 00001111 pattern).	$1 + 2 \times C_{+1}$
Vd	Full-scale amplitude (given 01010101 pattern).	1

Given a differential signal with symmetric swings above and below 0, it can be seen from Figure 8.5 that there are four possible output values in each positive or negative directions. Table 8.1 shows the positive normalized voltage levels.

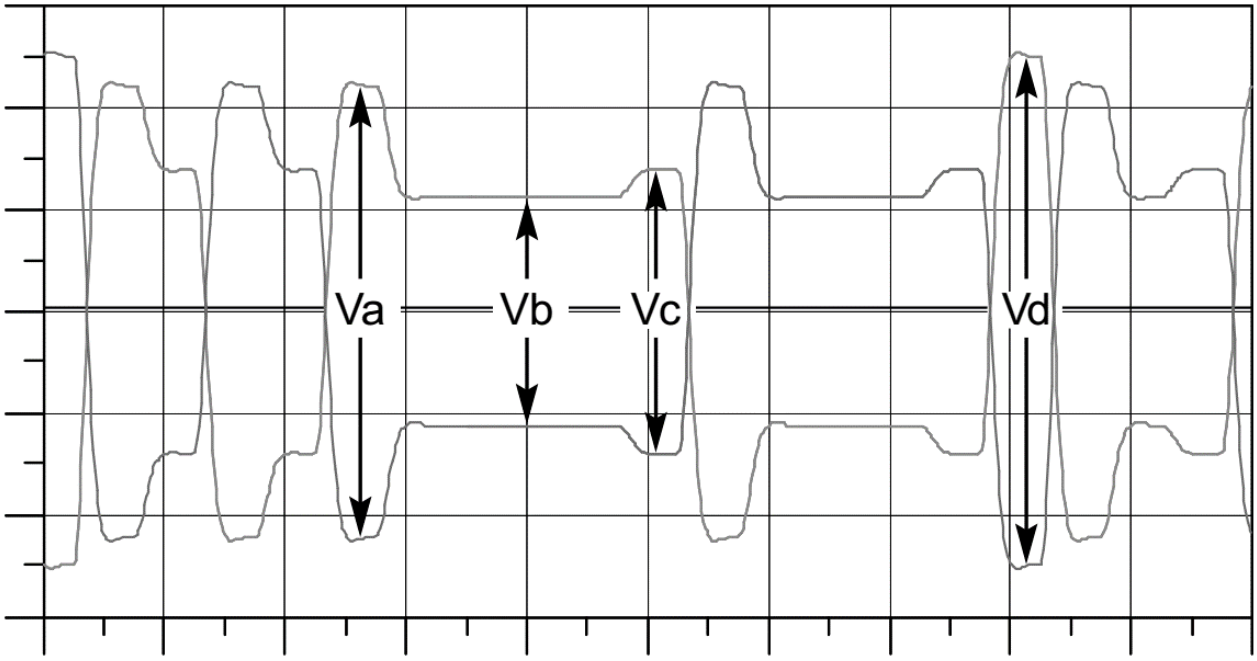

Figure 8.5. Definition of Tx Voltage Levels

The de-emphasis is defined as

$$\text{Deemphasis (dB)} = 20\log_{10}(V_b/V_a)\text{ (dB)}.$$

Moreover, the pre-shoot is defined as

$$\text{Preshoot(dB)} = 20\log_{10}(V_c/V_b)\text{ (dB)}.$$

Table 8.2 shows the Tx preset ratios and corresponding coefficient values defined by PCI Express Specification. For protocols that only require de-emphasis feature, the pre-cursor ratio (C_{-1}) should be fixed as zero. For example, to achieve –3.5 dB de-emphasis, post-cursor ratio (C_{+1}) should be set as –0.167 and pre-cursor ratio (C_{-1}) should be set as 0.000.

The pre-cursor ratio (C_{-1}) and post-cursor ratio (C_{+1}) can be configured through PMA registers (reg0a and reg0b). After these registers have been updated, the mpcs_txdeemp_i or epcs_txdeemp_i port should be asserted and then writes 1 to the bit 0 of PMA register (reg80) to trigger a new computation of PMA settings based on the current value in related registers.

Table 8.2. PCIe Tx Preset Ratios and Corresponding Coefficient Values

Preset #	Preshoot (dB)	De-emphasis (dB)	C_{-1}	C_{+1}	Va/Vd	Vb/Vd	Vc/Vd
P4	0.0	0.0	0.000	0.000	1.000	1.000	1.000
P1	0.0	–3.5	0.000	–0.167	1.000	0.668	0.668
P0	0.0	–6.0	0.000	–0.250	1.000	0.500	0.500
P9	3.5	0.0	–0.166	0.000	0.668	0.668	1.000
P8	3.5	–3.5	–0.125	–0.125	0.750	0.500	0.750
P7	3.5	–6.0	–0.100	–0.200	0.800	0.400	0.600
P5	1.9	0.0	–0.100	0.000	0.800	0.800	1.000
P6	2.5	0.0	–0.125	0.000	0.750	0.750	1.000
P3	0.0	–2.5	0.000	–0.125	1.000	0.750	0.750
P2	0.0	–4.4	0.000	–0.200	1.000	0.600	0.600

8.2. Rx Equalization

CertusPro-NX device implements programmable single pole-zero Continuous Timer Linear Equalizer (CTLE) at the receiver side, followed by Decision Feedback Equalization (DFE) and circuitry to support adaptation of the CTLE and DFE. Figure 8.6 shows the block diagram of receive equalizer. CTLE comprises an active high pass filter that has the effect of amplifying higher frequency components that have been more severely attenuated by the interconnection or attenuating the lower frequency components to a greater degree than the higher frequency components. DFE comprises samplers offset by a programmable voltage value and a decision selection circuit that has the effect of varying the sample threshold voltage based on the sampled data stream.

The CTLE and DFE can be tuned to compensate for more severe inter-symbol interference (ISI) than either circuit performs alone.

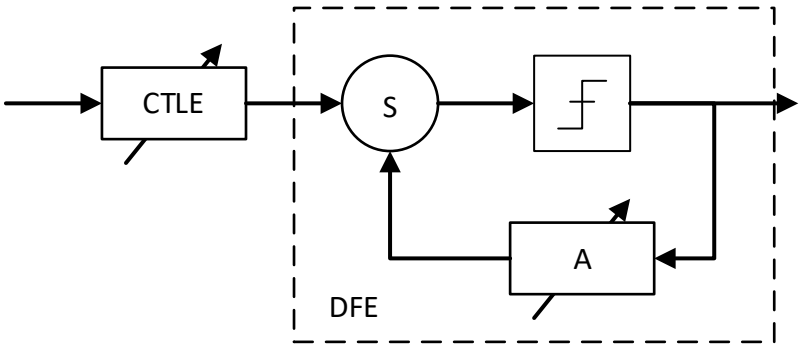

Figure 8.6. Receive Equalizer Block Diagram

The CTLE can be considered as a cascade of two frequency dependent equalizers. One primarily controls low frequency attenuation and the other primarily controls the boost at high frequency.

Figure 8.7 shows the illustrative CTLE frequency response as a function of RxA2gain and RxCTLEgain3db. The low frequency attenuation level A_{LF} is programmable and can be set by adaptation control to maximize the signal quality of the receiver for achieving the highest possible BER. The gain A_V is defined as

$$A_V = 20 \times log_{10}(\frac{V_{OUT}}{V_{IN}})$$

A_{PK-HF}, the peak gain, is typically 3-4 dB. A_{LF} is the gain of the low frequency equalizer with respect to the high frequency gain. A_{LF} typically ranges from 4 dB (RxA2gain code 0) to -12 dB (RxA2gain code 63). F_{PK}, the frequency associated with A_{PK-HF}, is typically around 3.2 GHz.

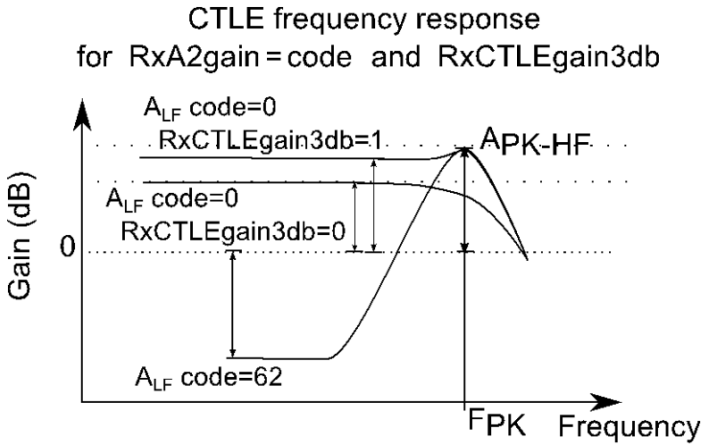

Figure 8.7. CTLE Frequency Response

It is important to note that CTLE has potential variation over process temperature and voltage. This CTLE was designed to be adapted by the PMA Controller using an adaptation scheme. Thus, it is recommended that when not employed as part of an adaptation scheme that A_{LF} is set to the maximum possible value by selecting RxA2gain=32'd0 (code 0). This setting represents the minimum CTLE equalization effect (closest to flat response).

Table 8.3 shows the CTLE gain code table, the applied RxA2gain code value can be read from PMA register (rege6).

Table 8.3. CTLE Gain Code (RxA2gain)

Code	A_{LF} (dB)	A_{PK} (dB)	F_{PK} (GHz)	BW (GHz)	RxA2gain
0	4.72	4.72	N/A	8.95	32'b00000000000000000000000000000000
1	4.55	4.55	N/A	9.31	32'b00000000000000000000000000000001
2	4.31	4.31	N/A	9.83	32'b00000000000000000000000000000011
3	3.95	3.96	1.20	10.58	32'b00000000000000000000000000000010
4	3.38	3.74	3.02	11.11	32'b00000000000000000000000000000110
5	3.31	3.73	3.09	11.13	32'b00000000000000000000000000000100
6	3.23	3.73	3.16	11.15	32'b00000000000000000000000000001100
7	3.15	3.73	3.24	11.16	32'b00000000000000000000000000001000
8	3.06	3.73	3.31	11.17	32'b00000000000000000000000000011000
9	2.97	3.73	3.39	11.18	32'b00000000000000000000000000010000
10	2.87	3.73	3.39	11.18	32'b00000000000000000000000000110000
11	2.76	3.74	3.47	11.18	32'b00000000000000000000000000100000
12	2.64	3.75	3.47	11.17	32'b00000000000000000000000001100000
13	2.52	3.76	3.47	11.16	32'b00000000000000000000000001000000
14	2.40	3.77	3.55	11.15	32'b00000000000000000000000011000000
15	2.27	3.78	3.55	11.13	32'b00000000000000000000000010000000
16	2.15	3.80	3.55	11.11	32'b00000000000000000000000110000000
17	2.01	3.81	3.55	11.10	32'b00000000000000000000000100000000
18	1.88	3.83	3.55	11.07	32'b00000000000000000000001100000000
19	1.74	3.84	3.55	11.05	32'b00000000000000000000001000000000
20	1.60	3.86	3.55	11.03	32'b00000000000000000000011000000000
21	1.46	3.88	3.55	11.00	32'b00000000000000000000010000000000
22	1.31	3.89	3.55	10.98	32'b00000000000000000000110000000000
23	1.15	3.91	3.55	10.95	32'b00000000000000000000100000000000
24	1.00	3.93	3.55	10.93	32'b00000000000000000001100000000000
25	0.84	3.95	3.55	10.90	32'b00000000000000000001000000000000
26	0.67	3.97	3.55	10.87	32'b00000000000000000011000000000000
27	0.51	3.98	3.55	10.84	32'b00000000000000000010000000000000
28	0.34	4.00	3.47	10.82	32'b00000000000000000110000000000000
29	0.16	4.02	3.47	10.79	32'b00000000000000000100000000000000
30	-0.02	4.04	3.47	10.76	32'b00000000000000001100000000000000
31	-0.20	4.05	3.47	10.73	32'b00000000000000001000000000000000
32	-0.40	4.07	3.47	10.70	32'b00000000000000011000000000000000
33	-0.59	4.09	3.39	10.67	32'b00000000000000010000000000000000
34	-0.79	4.11	3.39	10.65	32'b00000000000000110000000000000000
35	-1.00	4.12	3.39	10.62	32'b00000000000000100000000000000000
36	-1.21	4.14	3.39	10.59	32'b00000000000001100000000000000000
37	-1.44	4.16	3.39	10.56	32'b00000000000001000000000000000000
38	-1.66	4.18	3.31	10.53	32'b00000000000011000000000000000000
39	-1.90	4.19	3.31	10.51	32'b00000000000010000000000000000000

Code	A_{LF} (dB)	A_{PK} (dB)	F_{PK} (GHz)	BW (GHz)	RxA2gain
40	-2.13	4.21	3.31	10.48	32'b00000000000011000000000000000000
41	-2.39	4.23	3.31	10.45	32'b00000000000010000000000000000000
42	-2.65	4.24	3.24	10.42	32'b00000000001100000000000000000000
43	-2.92	4.26	3.24	10.40	32'b00000000001000000000000000000000
44	-3.20	4.27	3.24	10.37	32'b00000000011000000000000000000000
45	-3.49	4.29	3.24	10.34	32'b00000000010000000000000000000000
46	-3.79	4.31	3.16	10.32	32'b00000000110000000000000000000000
47	-4.11	4.32	3.16	10.29	32'b00000000100000000000000000000000
48	-4.43	4.34	3.16	10.26	32'b00000001100000000000000000000000
49	-4.77	4.35	3.09	10.24	32'b00000001000000000000000000000000
50	-5.12	4.37	3.09	10.21	32'b00000011000000000000000000000000
51	-5.50	4.38	3.09	10.19	32'b00000010000000000000000000000000
52	-5.89	4.40	3.09	10.16	32'b00000110000000000000000000000000
53	-6.30	4.41	3.02	10.14	32'b00000100000000000000000000000000
54	-6.73	4.43	3.02	10.11	32'b00001100000000000000000000000000
55	-7.19	4.44	3.02	10.09	32'b00001000000000000000000000000000
56	-7.68	4.45	2.95	10.07	32'b00011000000000000000000000000000
57	-8.20	4.47	2.95	10.04	32'b00010000000000000000000000000000
58	-8.76	4.48	2.95	10.02	32'b00110000000000000000000000000000
59	-9.36	4.49	2.95	10.00	32'b00100000000000000000000000000000
60	-10.01	4.51	2.88	9.98	32'b01100000000000000000000000000000
61	-10.72	4.52	2.88	9.95	32'b01000000000000000000000000000000
62	-11.48	4.53	2.88	9.93	32'b11000000000000000000000000000000
63	-12.33	4.55	2.82	9.91	32'b10000000000000000000000000000000

The CTLE is designed to support adaptation on live data by changing only RxA2gain to match the frequency loss in the channel. Note that the RxA2gain must be changed between adjacent codes, only single bit toggles high or low at any one time.

The DFE circuit adjusts the decision threshold of the present bit by the resolved value of the prior bit. If the prior data is a 0, then the low threshold is chosen. If the prior data is 1, then the high threshold is chosen. The DFE helps equalize the effect of inter-symbol interference (ISI) by subtracting the pulse response amplitude of prior symbol (bit) from the current symbol (bit).

The magnitude of the DFE feedback is determined by the asynchronous control bus RxA1gain, which is defined by Table 8.4. The applied RxA1gain code value can be read from PMA register (rege5).

Table 8.4. DFE Feedback Magnitude (RxA1gain)

Code	DAC Value (mV)	RxA1gain
0	0	32'b00000000000000000000000000000000
1	1.5	32'b00000000000000000000000000000001
2	3	32'b00000000000000000000000000000011
3	4.5	32'b00000000000000000000000000000010
4	6	32'b00000000000000000000000000000110
5	7.5	32'b00000000000000000000000000000100
6	9	32'b00000000000000000000000000001100
7	10.5	32'b00000000000000000000000000001000
8	12	32'b00000000000000000000000000011000
9	13.5	32'b00000000000000000000000000010000

Code	DAC Value (mV)	RxA1gain
10	15	32'b00000000000000000000000000110000
11	16.5	32'b00000000000000000000000000100000
12	18	32'b00000000000000000000000001100000
13	19.5	32'b00000000000000000000000001000000
14	21	32'b00000000000000000000000011000000
15	22.5	32'b00000000000000000000000010000000
16	24	32'b00000000000000000000000110000000
17	25.5	32'b00000000000000000000000100000000
18	27	32'b00000000000000000000001100000000
19	28.5	32'b00000000000000000000001000000000
20	30	32'b00000000000000000000011000000000
21	31.5	32'b00000000000000000000010000000000
22	33	32'b00000000000000000000110000000000
23	34.5	32'b00000000000000000000100000000000
24	36	32'b00000000000000000001100000000000
25	37.5	32'b00000000000000000001000000000000
26	39	32'b00000000000000000011000000000000
27	40.5	32'b00000000000000000010000000000000
28	42	32'b00000000000000000110000000000000
29	43.5	32'b00000000000000000100000000000000
30	45	32'b00000000000000001100000000000000
31	46.5	32'b00000000000000001000000000000000
32	48	32'b00000000000000011000000000000000
33	49.5	32'b00000000000000010000000000000000
34	51	32'b00000000000000110000000000000000
35	52.5	32'b00000000000000100000000000000000
36	54	32'b00000000000001100000000000000000
37	55.5	32'b00000000000001000000000000000000
38	57	32'b00000000000011000000000000000000
39	58.5	32'b00000000000010000000000000000000
40	60	32'b00000000000110000000000000000000
41	61.5	32'b00000000000100000000000000000000
42	63	32'b00000000001100000000000000000000
43	64.5	32'b00000000001000000000000000000000
44	66	32'b00000000011000000000000000000000
45	67.5	32'b00000000010000000000000000000000
46	69	32'b00000000110000000000000000000000
47	70.5	32'b00000000100000000000000000000000
48	72	32'b00000001100000000000000000000000
49	73.5	32'b00000001000000000000000000000000
50	75	32'b00000011000000000000000000000000
51	76.5	32'b00000010000000000000000000000000
52	78	32'b00000110000000000000000000000000
53	79.5	32'b00000100000000000000000000000000
54	81	32'b00001100000000000000000000000000
55	82.5	32'b00001000000000000000000000000000

Code	DAC Value (mV)	RxA1gain
56	84	32'b00011000000000000000000000000000
57	85.5	32'b00010000000000000000000000000000
58	87	32'b00110000000000000000000000000000
59	88.5	32'b00100000000000000000000000000000
60	90	32'b01100000000000000000000000000000
61	91.5	32'b01000000000000000000000000000000
62	93	32'b11000000000000000000000000000000
63	94.5	32'b10000000000000000000000000000000

Note that the RxA1gain must be changed between adjacent codes, only single bit toggles high or low at any one time.

In order to support the adaptation of the CTLE and the DFE, a programmable error sampler is implemented by PMA. The error sampler is designed to provide information for sign-sign least mean square adaption (SS-LMS) or related adaptation schemes. This error sampler uses the same clock edge as the DFE sampler and has a programmable voltage threshold which is controlled by two magnitude inputs RxA0gain and RxA1gain. Table 8.5 shows the error sampler threshold calculation.

Table 8.5. Error Sampler Threshold Calculation

Error Sampler Threshold	A0DIR	A1DIR	Description
−RxA0gain − RxA1gain	0	0	non-transition 0 voltage target
+RxA0gain − RxA1gain	1	0	transition 1 voltage target
+RxA0gain + RxA1gain	1	1	non-transition 1 voltage target
−RxA0gain + RxA1gain	0	1	transition 0 voltage target

Table 8.6 defines the RxA0gain, and the applied RxA0gain code value can be read from PMA register (rege4). Note that the RxA0gain must be changed between adjacent codes, only single bit toggles high or low at any one time.

Table 8.6. Error Sampler Amplitude (RxA0gain)

Code	DAC Value (mV)	RxA0gain
0	0	32'b00000000000000000000000000000000
1	3	32'b00000000000000000000000000000001
2	6	32'b00000000000000000000000000000011
3	9	32'b00000000000000000000000000000010
4	12	32'b00000000000000000000000000000110
5	15	32'b00000000000000000000000000000100
6	18	32'b00000000000000000000000000001100
7	21	32'b00000000000000000000000000001000
8	24	32'b00000000000000000000000000011000
9	27	32'b00000000000000000000000000010000
10	30	32'b00000000000000000000000000110000
11	33	32'b00000000000000000000000000100000
12	36	32'b00000000000000000000000001100000
13	39	32'b00000000000000000000000001000000
14	42	32'b00000000000000000000000011000000
15	45	32'b00000000000000000000000010000000
16	48	32'b00000000000000000000000110000000
17	51	32'b00000000000000000000000100000000
18	54	32'b00000000000000000000001100000000
19	57	32'b00000000000000000000001000000000
20	60	32'b00000000000000000000011000000000

Code	DAC Value (mV)	RxA0gain
21	63	32'b00000000000000000000010000000000
22	66	32'b00000000000000000000110000000000
23	69	32'b00000000000000000001100000000000
24	72	32'b00000000000000000001110000000000
25	75	32'b00000000000000000010000000000000
26	78	32'b00000000000000000011000000000000
27	81	32'b00000000000000000100000000000000
28	84	32'b00000000000000000110000000000000
29	87	32'b00000000000000001000000000000000
30	90	32'b00000000000000001100000000000000
31	93	32'b00000000000000010000000000000000
32	96	32'b00000000000000011000000000000000
33	99	32'b00000000000000100000000000000000
34	102	32'b00000000000000110000000000000000
35	105	32'b00000000000001000000000000000000
36	108	32'b00000000000001100000000000000000
37	111	32'b00000000000010000000000000000000
38	114	32'b00000000000011000000000000000000
39	117	32'b00000000000100000000000000000000
40	120	32'b00000000000110000000000000000000
41	123	32'b00000000001000000000000000000000
42	126	32'b00000000001100000000000000000000
43	129	32'b00000000010000000000000000000000
44	132	32'b00000000110000000000000000000000
45	135	32'b00000001000000000000000000000000
46	138	32'b00000001100000000000000000000000
47	141	32'b00000010000000000000000000000000
48	144	32'b00000011000000000000000000000000
49	147	32'b00000100000000000000000000000000
50	150	32'b00000110000000000000000000000000
51	153	32'b00001000000000000000000000000000
52	156	32'b00001100000000000000000000000000
53	159	32'b00010000000000000000000000000000
54	162	32'b00011000000000000000000000000000
55	165	32'b00100000000000000000000000000000
56	168	32'b00110000000000000000000000000000
57	171	32'b01000000000000000000000000000000
58	174	32'b01100000000000000000000000000000
59	177	32'b10000000000000000000000000000000
60	180	32'b11000000000000000000000000000000
61	183	32'b00000000000000000000000000000000
62	186	32'b00000000000000000000000000000000
63	189	32'b00000000000000000000000000000000

It is recommended to use the adaptive scheme directly, other than to configure CTLE and DFE modules manually. The Rx adaptive equalization related parameters can be set through IPGUI of PMA register (regd9).

9. SerDes/PCS Debug Capabilities

9.1. Loopback Mode

CertusPro-NX SerDes/PCS supports up to seven loopback modes for different test purposes:

- PMA Near-End Serial Loopback Mode
- PMA Far-End Parallel Loopback Mode
- 8B/10B PCS Near-End Parallel Loopback Mode
- 8B/10B PCS Far-End Parallel Loopback Mode
- 64B/66B PCS Loopback Path A
- 64B/66B PCS Loopback Path B
- 64B/66B PCS Loopback Path C

9.1.1. PMA Loopback

Figure 9.1 shows the two loopback modes implemented by PMA and PMA Controller, PMA Near-End Serial Loopback Mode and PMA Far-End Parallel Loopback Mode.

In the near end serial loopback mode, the serial data from Tx path is fed back to the Rx PMA (CDR and DES). The Rx PMA extracts clock and data from the bit stream generated by Tx path. This loopback mode can be enabled or disabled by the bit[1] of PMA register reg64.

In the far end parallel loopback mode, the parallel data received from Rx PMA is transmitted to Tx PMA directly. This mode requires no clock frequency difference between Rx path and Tx path. Otherwise, user logic must implement bit skip functionality with the Rx PMA.

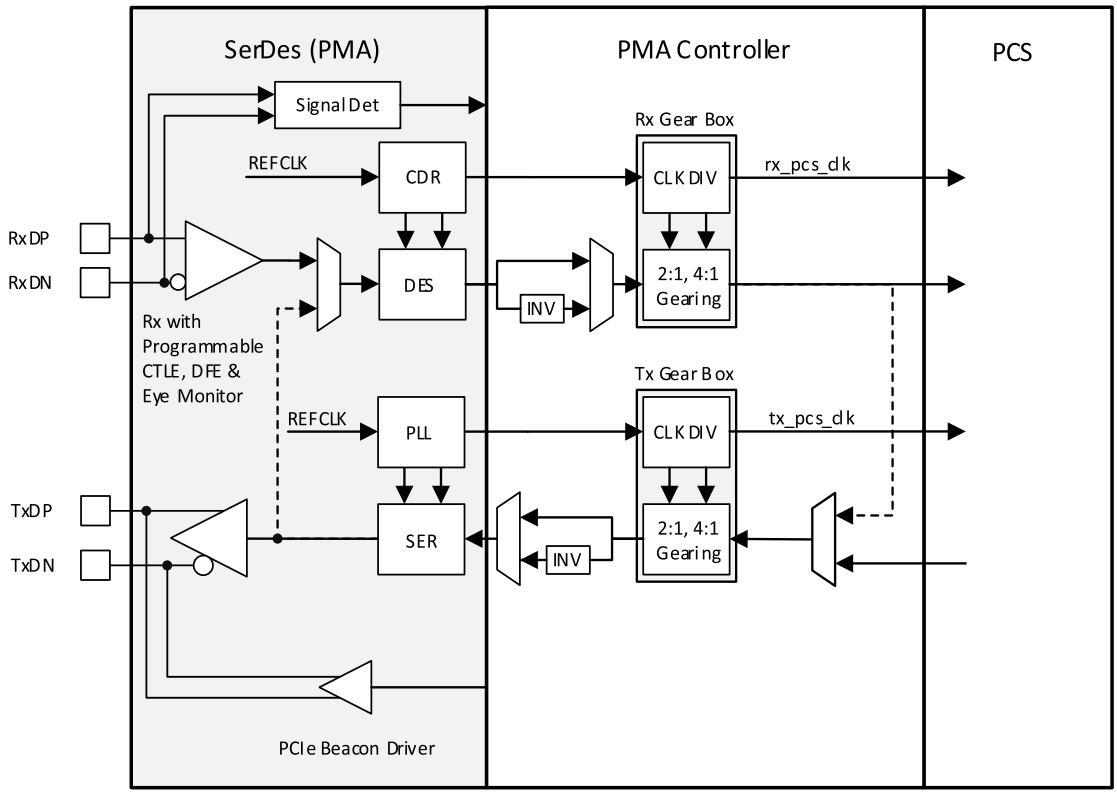

Figure 9.1. PMA Loopback Mode

9.1.2. 8B/10B PCS Loopback

The 8B/10B PCS implements two loopback modes, 8B/10B PCS Near-End Parallel Loopback Mode and 8B/10B PCS Far-End Parallel Loopback Mode.

Figure 9.2 shows the block diagram of 8B/10B PCS Near-End Parallel Loopback Mode. In this mode, the 8B/10B PCS output data passes through a phase compensation FIFO (Loopback FIFO) to Rx path. In addition, the tx_pcs_clk is used to drive Rx path. This loopback mode can be enabled or disabled by the bit[0] of MPCS register rege0.

Figure 9.3 shows the block diagram of 8B/10B PCS Far-End Parallel Loopback Mode. In this mode, the 8B/10B PCS Rx output data is looped back to Tx path near fabric side. In addition, the rx_out_clk is used to drive the writing side of the Tx FIFO. What should be noted is that the Tx and Rx path clock must have no frequency difference. In other words, the Tx path and Rx path should share the same reference clock source. This loopback mode can be enabled or disabled by the bit[1] of MPCS register rege0.

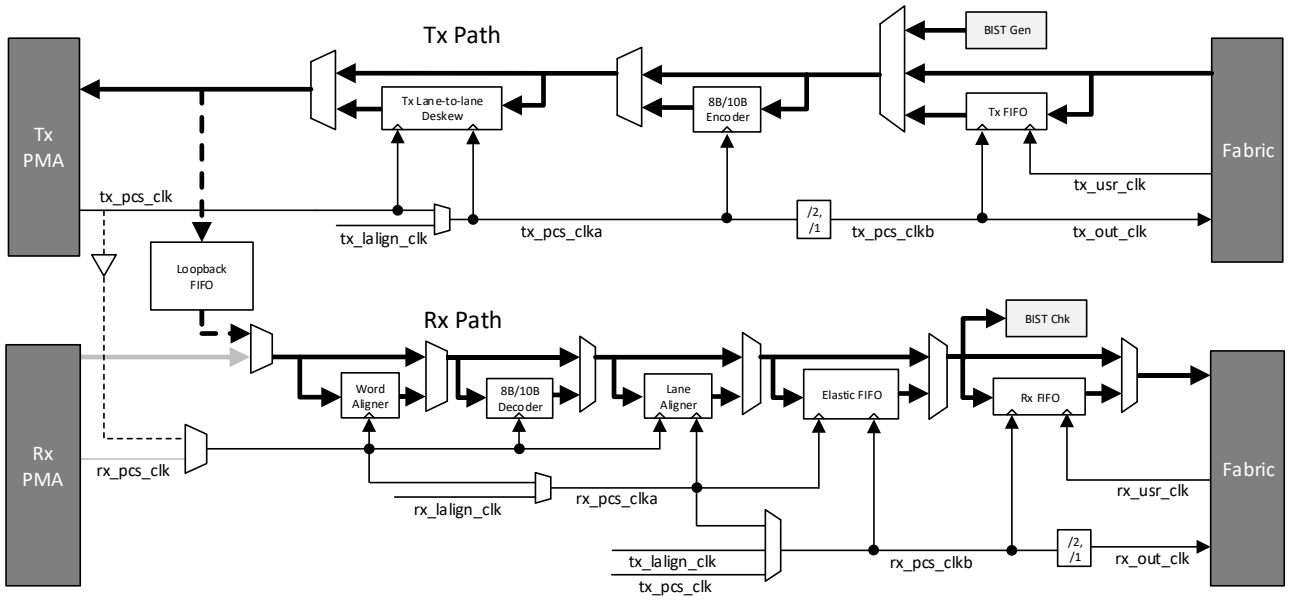

Figure 9.2. 8B/10B PCS Near-End Parallel Loopback Mode

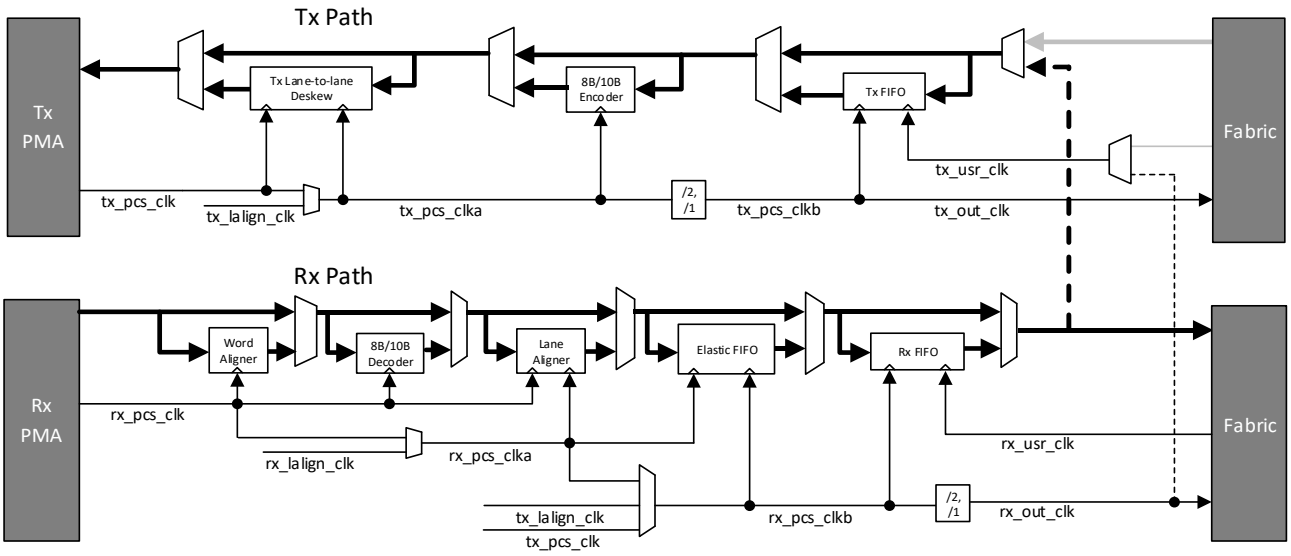

Figure 9.3. 8B/10B PCS Far-End Parallel Loopback Mode

9.1.3. 64B/66B PCS Loopback

Figure 9.4 shows three loopback modes implemented by 64B/66B PCS: Loopback Path A, Loopback Path B and Loopback Path C.

In loopback path A mode, the 16-bit input data of Rx path comes from Tx path. In addition, the tx_pcs_clk is used to drive both Tx path and Rx path. The Loopback FIFO is used for clock phase compensation between Tx path and Rx path. This mode can be enabled or disabled by the bit[0] of MPCS register rege0 when MPCS works as 64B/66B PCS mode.

In loopback path B mode, the Tx XGMII 64-bit data and 8-bit control from fabric are fed back to Rx FIFO directly. This mode can be enabled or disabled by the bit[2] of MPCS register rege0 when MPCS works as 64B/66B PCS mode.

In loopback path C mode, the received XGMII 64-bit data and 8-bit control from Rx path are fed back to Tx FIFO directly. This mode can be enabled or disabled by the bit[1] of MPCS register rege0 when MPCS works as 64B/66B PCS mode.

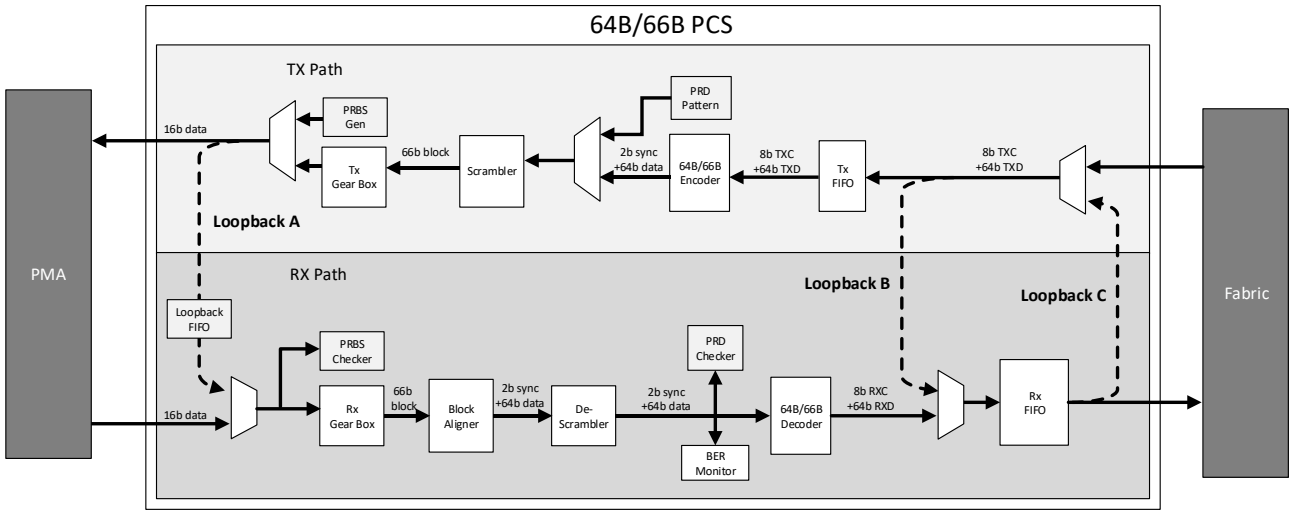

Figure 9.4. 64B/66B PCS Loopback Mode

9.1.4. PMA Only Loopback

The PMA Only mode, same as 8B/10B PCS, also supports loopback modes. The near end parallel loopback mode can be enabled or disabled by the bit[0] of MPCS register rege0 when MPCS works as PMA Only mode. The far end parallel loopback mode can be enabled or disabled by the bit [1] of MPCS register rege0 when MPCS works as PMA Only mode.

9.2. Signal Detector

Each channel contains a signal detector circuit at receiver input, as shown in Figure 9.5. The signal detector picks up the data signal from channel receiver input, and compares its differential amplitude with the threshold voltage. If the input signal differential amplitude is higher than the threshold voltage, the detector asserts the TranDet output.

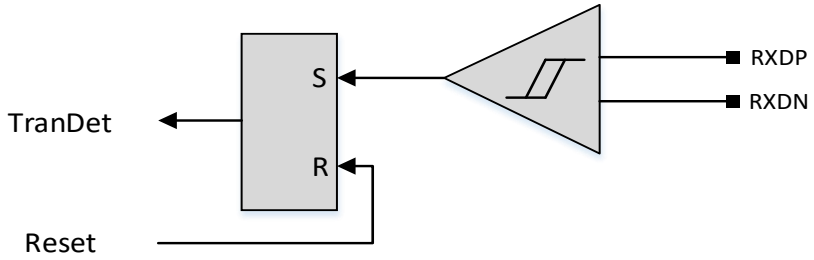

Figure 9.5. Signal Detector

The threshold voltage can be configured as 125 mV or 180 mV, by accessing the bit[1] of PMA Control Register 0 [PMA reg00]. The signal detection condition is defined in Table 9.1. After reset, if TranDet is not asserted, it means no transition exceeding the threshold voltage is detected at RxDP/RxDN. If TranDet is asserted, it means at least one transition exceeding threshold voltage is detected at RxDP/RxDN.

The signal detector output can be found by accessing the bit[2] of PMA Status Register [PMA reg7f].

Table 9.1. Signal Detection Conditions

Threshold Voltage	Signal Detection Conditions
125 mV	Rx differential amplitude less than 125 mV.
180 mV	Rx differential amplitude less than 180 mV.

Note that this signal detector is mainly implemented for electrical idle detection, which is required by PCIe. CertusPro-NX SerDes does not support the Loss of Signal (LOS) functionality, which is supported by Lattice ECP3 and ECP5 SerDes.

9.3. Loss of Lock

Both Tx PLLs and CDR PLLs have the loss-of-lock detectors for PMA debug purpose.

If the Tx PLL loses lock, the loss-of-lock for the PLL is asserted and remains asserted until the PLL reacquires lock state. When the PLL loses lock, it is likely to be caused by a reference clock problem. The Tx PLL loss-of-lock detector output can be found by accessing the bit[4] of PMA Status Register [PMA reg7f].

For CDR PLLs, there are two steps to get the lock state:

- Frequency lock or lock with reference-to-reference clock, is a frequency lock operation whereby CDR PLL locks to the reference clock through the Phase Frequency Detector (PFD) operation. Sampling clock at the receiver is not aligned to the center of the data eye during this step.
- Phase lock or lock with reference to input bitstream, is a phase lock operation whereby CDR PLL acquires phase and small frequency deviation lock to the bitstream. Sampling clock at the receiver is aligned to the center of the data eye after this step. It is imperative that the bitstream be valid upon entering phase lock. There are two further steps for the phase lock:
  - Coarse phase lock, which has higher range of frequency acquisition (±5000 ppm of static frequency difference). This step is always a transient step before embarking on to fine phase lock.
  - Fine phase lock, which has a lower range of frequency acquisition (±300 ppm static frequency difference). Note that fine phase lock can acquire to spread spectrum modulated signals (dynamic frequency difference) when the slow-moving frequency of the input intersects the current frequency of CDR PLL. Thereafter, fine phase lock continues to track spread spectrum modulation barring abrupt frequency jumps requiring reacquisition in coarse phase lock.

The CDR PLL loss-of-lock detector output can be found by accessing the bit[3] and bit[5] of PMA Status Register [PMA reg7f].

Figure 9.6 shows CDR PLL locking flow. Table 9.2 shows the typical duration time for each lock step.

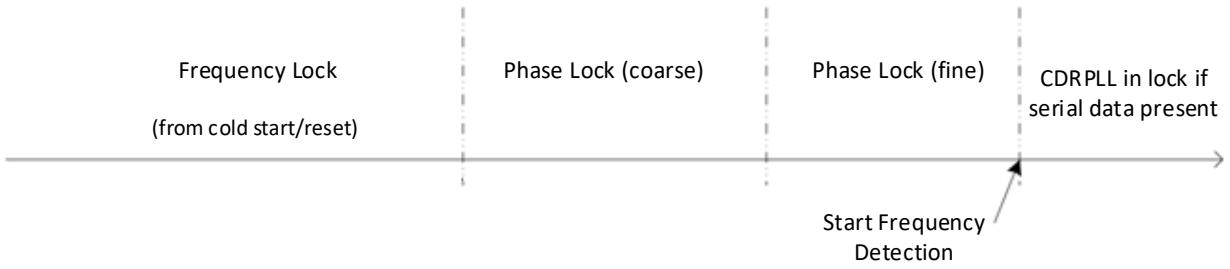

Figure 9.6. CDR PLL Locking Flow

Table 9.2. Typical Duration Time for Each Lock Step

Description	Frequency Lock	Coarse Phase Lock	Fine Phase Lock
Duration when training	15 µs	0.5 µs – 50 µs	0.25 µs – 25 µs
Duration when relocking	15 µs	0.5 µs	0.25 µs

PMA Controller begins frequency detection (or comparison of frequency difference between TxClk and RxClk) after being in fine phase lock state for 0.25 µs or more.

In the absence of input data bitstream or other externally induced factors which causes loss of lock of CDR PLL, the frequency of the CDR PLL drifts and as the deviation of frequency between RxClk and TxClk exceeds preset threshold as determined by the PMA Controller, PMA Controller goes back to frequency lock state and repeats the acquisition process.

As long as the CDR PLL and associated datapath are enabled, and reference clock is present, irrespective of the presence or absence of input data bitstream, the CDR PLL produces RxClk. In the absence of input data bitstream (due to Electrical Idle or other conditions), the frequency of CDR PLL can drift anywhere within the range specified by the bit[2] of Control Register 0 [PMA reg00]. In the presence of input data bitstream, CDR PLL acquires phase and frequency lock to input data bitstream within 2500-bit times.

The bit[3] of Control Register 0 [PMA reg00] shows the status of CDR PLL internal frequency detector. Table 9.3 shows the conditions when this signal becomes asserted, where assertion is equated to “show mismatch”.

Table 9.3. CDR PLL Frequency Detector Operation Bounds

bit[3] of Control Register 0	WILL NOT show mismatch	MAY show mismatch	WILL show mismatch
0	frequency difference < 0.39%	0.39% < frequency difference < 0.59%	frequency difference > 0.59%
1	frequency difference < 0.78%	0.78% < frequency difference < 1.17%	frequency difference < 1.17%

If the input data bitstream is lost for whatever reason after CDR PLL is locked to input data bitstream, the CDR PLL would not continue to receive timing correction signals from the input data bitstream. The frequency of CDR PLL drifts, and this drift is dependent on the leakage currents, mismatch of leakage currents, noise, and other ambient conditions. The amount of drift of frequency of CDR PLL over a given duration of time is indeterminate, and it is entirely possible that the frequency do not drift significantly enough to trigger frequency mismatch indication for the given duration.

If after the CDR PLL has acquired lock to the input data bitstream, and the frequency of the input data bitstream suddenly shifts in average (as opposed to a slow-moving frequency shift caused due to spread spectrum input) to the frequency difference lager than the operation bounds, the CDR PLL frequency detector may report the mismatch.

9.4. ECO Editor

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

ECO Editor is a graphical representation of all the user configurable settings and allows user to change the settings in that environment. Care must be taken when making the changes. The design software tools cannot check the correctness of the changes that the user made.

ECO Editor changes only settings inside the SerDes. Changes in the SerDes that affect the external connection requires user to re-run the software flow. Changes made in ECO Editor that do not change external connection just requires user to re-generate the bitstream with the same place-and-routing done in the core.

Refer to the ECO Editor User Guide in Lattice Radiant software for more information.

9.5. Eye Monitor

Eye monitor is an on-chip scope to visualize post-equalization signal quality at SerDes/PMA Rx side. In some documents, this eye monitor is also named as 2D eye scan.

Figure 9.7 shows the block diagram of eye monitor in CertusPro-NX SerDes/PMA. RxDATA is recovered from the equalized differential waveform by sampling after the Rx equalizer (CTLE and DFE). The horizontal sampling position for data sampler is determined by the CDR function, while the vertical sampling position for data sampler is fixed as differential zero. To enable eye scan functionality, an additional sampler is provided with programmable horizontal and vertical offsets from the data sampling position. Figure 9.8 shows the example of data sample and offset sample.

Eye monitor accumulates the number of data samples (sample counter) and the number of times that the offset sample disagreed with the data sample (error counter). The bit error ratio (BER) at the programmed vertical and horizontal offset is defined as the ration of the error count to the sampler count. Repeating such BER measurements for the full array of horizontal and vertical offsets (or a subsampled set of offsets) to produce a BER map as shown in Figure 9.9.

Note that this BER map is commonly referred to as a statistical eye diagram, where the color map represents $\log_{10}$(BER). This eye is apparently smaller than a traditional oscilloscope view because it has been closed by very low probability jitter and noise that do not show up in the much lower number of samples of an oscilloscope.

For the detailed usage of CertusPro-NX eye monitor, refer to Radiant Help > User Guide > Testing and Debugging On-Chip > Performing Logic Analysis > Debugging with Reveal Controller > Running Reveal Eye-Opening Monitor.

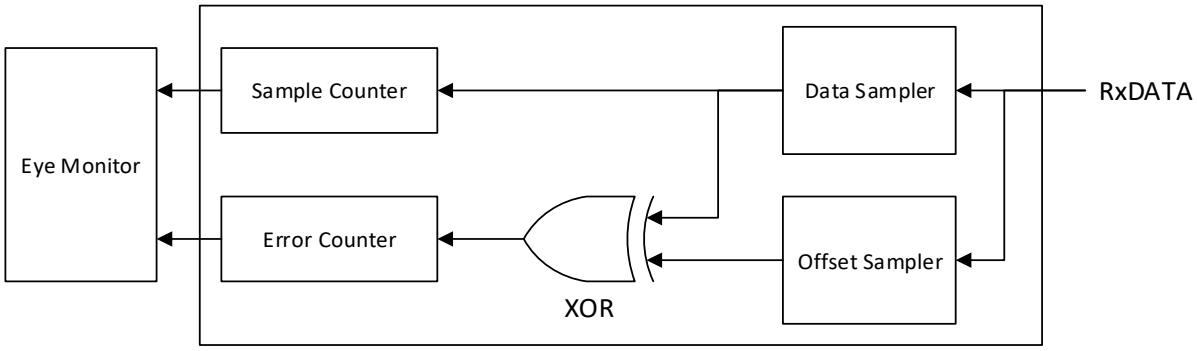

Figure 9.7. Eye Monitor Block Diagram

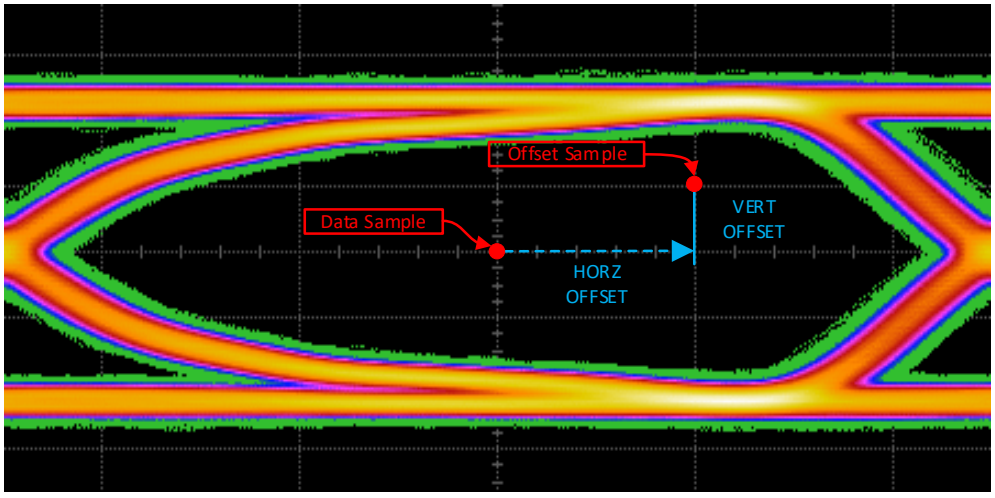

Figure 9.8. Data Sample and Offset Sample

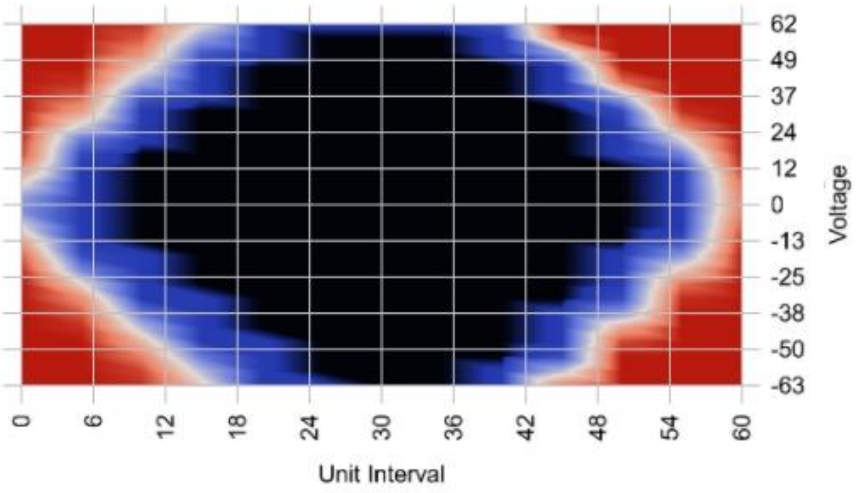

Figure 9.9. Eye Monitor Output (BER Map)

10. SerDes/PCS Register Access

10.1. Register Access Bus

Each SerDes/PCS channel has independent register access bus: Lattice Memory Mapped Interface (LMMI). It has the following features:

- 8-bit width of write and read data.
- 9-bit address (offset), the highest bit is used to select register space.
  - offset[8] == 1'b1, MPCS register space
  - offset[8] == 1'b0, PMA register space
- Support single byte read and write.
- Support burst read and write.
- Read-back data validity indication.
- Ready signal to support wait state.

When the LMMI master is ready to request a transaction, it asserts lmmi_request_i and drives the request data (lmmi_wr_rdn_i, lmmi_offset_i, and lmmi_wdata_i) at the same time and starts sampling lmmi_ready_o on the positive edge clock. The LMMI master holds the lmmi_request_i and the requested data constantly until the lmmi_ready_o is asserted. When both lmmi_request_i and lmmi_ready_o are asserted on the same positive clock edge, the LMMI slave latches the requested data and starts transaction. Once the requested data has been latched, the LMMI master can immediately start the next transaction request as its own discretion. If the LMMI slave is ready to start another transaction in the next clock cycle, it holds lmmi_ready_o asserted. Otherwise, the LMMI slave de-asserts lmmi_ready_o until it is ready to accept a new transaction. Figure 10.1, Figure 10.2, and Figure 10.3 show the examples for different transactions.

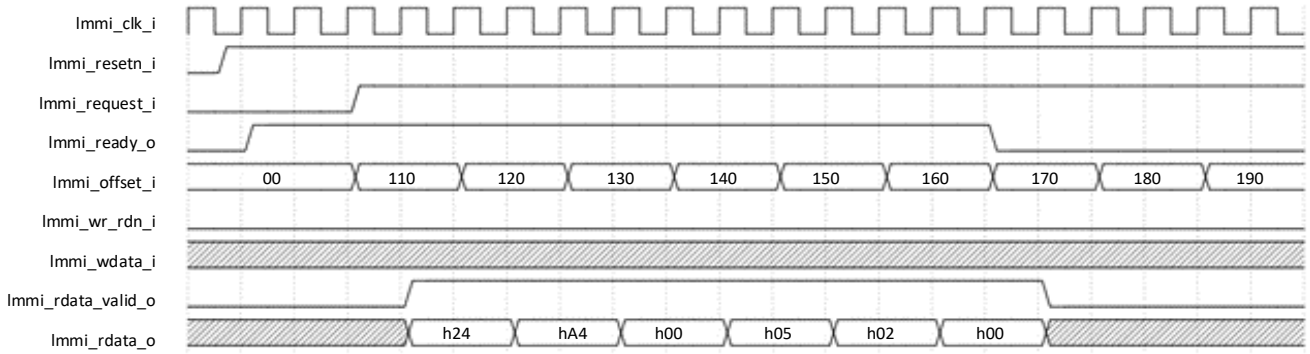

Figure 10.1. Burst Read Transaction

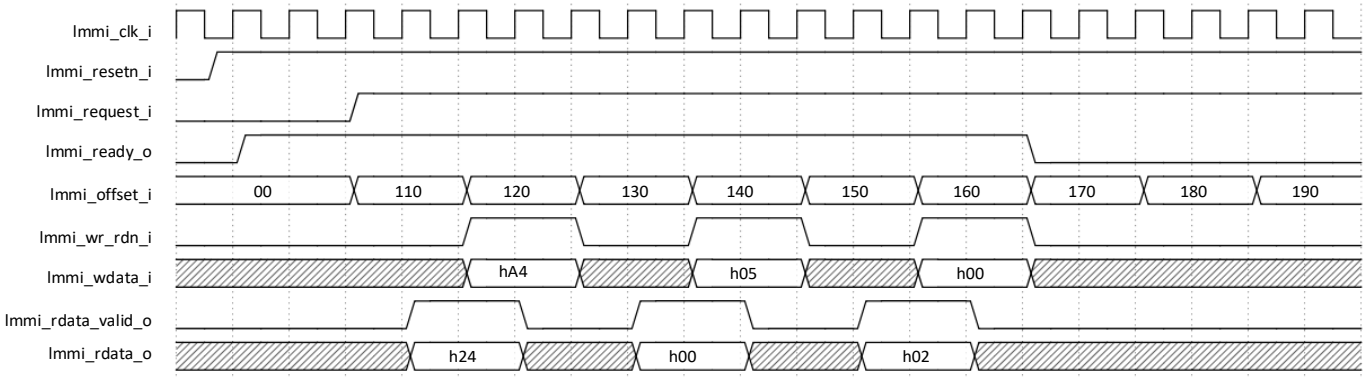

Figure 10.2. Back-to-Back Read and Write Transaction

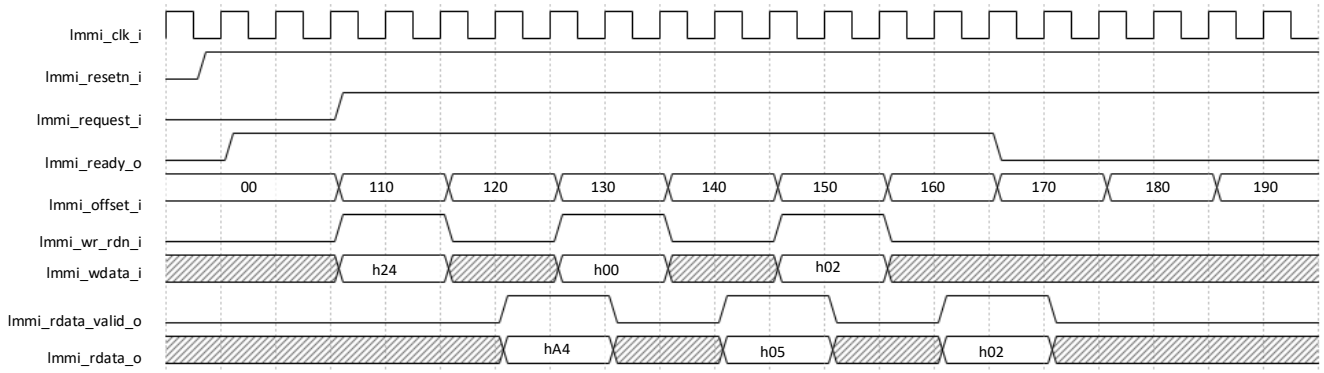

Figure 10.3. Back-to-Back Write and Read Transaction

Note that all one in lmmi_offset_i and all one in lmmi_wdata_i triggers the software reset in LMMI protocol. This software reset puts registers back to the reset states.

10.2. Register Space

Each SerDes/PCS channel has independent register space which includes following category of registers:

- Configuration registers
- Control registers
- Status registers
- Status registers with clear-on-read

All register bits in Appendix A. Configure Registers are shown with bit7 as the significant bit on the left. Table 10.1 shows the abbreviations that are used to indicate what type of register access for each is supported.

Table 10.1. Access Type Definition

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value.	Ignores write access.
WO	Returns 0.	Updates register value.
RW	Returns register value.	Updates register value.
RW1C	Returns register value.	Writing 1'b1 on register bit clears the bit to 1'b0. Writing 1'b0 on register bit is ignored.
W/A	Returns register value.	Writing 1'b1 to clear related bit. Writing 1'b0 is ignored.
CR	Clear on a read.	Ignores write access.
RSVD	Returns 0.	Ignores write access.

11. Protocol Mode

11.1. PCI Express Mode

PCI Express is a high performance, fully scalable, well-defined standard for a wide variety of computing and communications platforms. Being a packet based serial technology, PCI Express greatly reduces the number of required pins and simplifies board routing and manufacturing. PCI Express is a point-to-point technology, as opposed to the multi-drop bus in PCI. Each PCI Express device has the advantage of full duplex communication with its link partner to greatly increase overall system bandwidth.

In CertusPro-NX, only Quad0 integrates PCIe Link Layer hard IP, which works with PCI Express PCS, PMA Controller and PMA. The channel cannot be configured as MPCS mode or EPCS mode when this channel has been configured as PCI Express mode, considering all these modes share the PMA Controller and PMA channels.

PCIe Base Specification requires external AC-coupling. The recommended Capacitance are shown in Table 11.1. The detected state is the initial state after PCIe reset or power up. In this state, a device electrically detects if a receiver device is present at the far end of the link by observing the rate of change. And the device compares the time the line voltage to charge-up and the expected time. If a receiver device is attached, the charge time is much longer. The value and location of this AC-coupling capacitor affects the charge time during PCIe detect state. That is why the recommended capacitance range is so narrow.

Table 11.1. PCI Express Recommend AC Capacitance

PCIe Generation	Minimum	Typical	Maximum
Gen1	75 nF	—	265 nF
Gen2	75 nF	—	265 nF
Gen3	176 nF	—	265 nF

PCI Express Base Specification defines a beacon signal inside the Link Training and Status State Machine (LTSSM). While a PCIe link is in L2 power state, its main power source and clock are turned off. An auxiliary voltage (V_{aux}) keeps a small part of the device working, including the wake-up logic. To signal a wake-up event, a downstream device can drive the beacon signal upstream to start the L2 exit sequence. A switch or bridge receiving a beacon signal on its downstream port must forward notification upstream by sending the beacon signal on its upstream port or by asserting the WAKE# pin.

However, CertusPro-NX SerDes does not support the auxiliary power required by PCIe LTSSM L2 state. It is meaningless to support PCIe beacon signal or WAKE# signal generation and detection for unsupported LTSSM L2 state.

For more detailed information on CertusPro-NX PCI Express features, function descriptions, and IP usage, refer to PCIe X4 IP Core - Lattice Radiant Software (FPGA-IPUG-02126).

11.2. Generic 8B/10B Mode

The Generic 8B/10B mode of CertusPro-NX SerDes/PCS is intended for applications requiring 8B/10B encoding and decoding without the need for additional protocol-specific data manipulation. CertusPro-NX SerDes/PCS can support Generic 8B/10B applications from 625 Mbps to 8.1 Gbps per channel.

In Generic 8B/10B mode, 8B/10B Encoder module and 8B/10B Decoder module are enabled. Tx Lane-to-lane Deskew module, Tx FIFO module, Word Aligner module, Rx Lane Aligner module, Elastic Buffer module and Rx FIFO module can be enabled or bypassed per user design. Refer to the 8B/10B PCS section for detailed information about these modules inside CertusPro-NX SerDes/PCS 8B/10B Channel PCS.

11.3. PMA Only Mode

The PMA Only mode of CertusPro-NX SerDes/PCS is intended for applications without 8B/10B coding and decoding. In PMA Only mode, the PMA EPCS data bus is accessed by user logic with very low latency. Only Tx FIFO, Rx FIFO and Rx Lane-to-lane Deskew modules are implemented between the PMA Controller and fabric.

The width of user data bus in PMA Only mode can be configured as 8-bit, 10-bit, 16-bit or 20-bit data. Refer to Table 6.7 for detailed information about PMA Only mode data bus.

11.4. Ethernet Mode

11.4.1. 1000BASE-X (GigE) Mode

The 1000BASE-X (or Gigabit Ethernet, GigE) mode of CertusPro-NX SerDes/PCS fully supports from the serial I/O to the GMII/SGMII interface of the IEEE 802.3 1000BASE-X Gigabit Ethernet standard.

In 1000BASE-X mode, CertusPro-NX SerDes/PCS supports the clock compensation and auto-negotiation features. The auto-negotiation control logic should be implemented in FPGA fabric to work with CertusPro-NX SerDes/PCS.

Idle pattern insertion is required for clock compensation and auto-negotiation. The mpcs_anxmit_i signal should be asserted by auto-negotiation control logic to indicate the current state is GigE auto-negotiation state. In this state, the /C1/ and /C2/ ordered sets are replaced by /I2/ ordered sets periodically at Rx path, so that the Elastic Buffer gets the opportunity to perform clock compensation functionality. While auto-negotiating, the link partner transmits /C1/ and /C2/ ordered sets continuously.

The cordisp bit of mpcs_tx_data is used on the transmit side of the PCS to ensure that an Inter-Packet Gap (IPG) 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 content of the Ethernet frame.

However, from the FPGA fabric side of the PCS, the current disparity state of the PCS transmitter is unknown. This is where the cordisp bit signal comes into play. If the cordisp bit signal is asserted for one clock cycle upon entering an IPG, it forces the Tx path PCS to insert a /I1/ ordered set into the transmit data stream when the current disparity is positive. If the current disparity is negative, then no change is made to the transmit data stream.

From the FPGA fabric side of the PCS, the IPG is typically characterized by the continuous transmission of the /I2/ ordered set.

Note that in the PCS channel, /I2/ ordered set means the current disparity is to be preserved, and /I1/ ordered set means the current disparity state should be flipped. Therefore, it is possible to ensure that the IPG begins in a negative disparity state. If the disparity state before the IPG is negative, then a continuous stream of /I2/ ordered sets are transmitted during the IPG. If the disparity state before the IPG is positive, then followed by a continuous stream of /I2/ ordered sets, a single /I1/ ordered set is transmitted.

At the FPGA fabric side of the PCS, the IPG is always driven into the PCS with /I2/ ordered sets. The cordisp bit signal is asserted for one clock cycle, when the IPG first begins. If necessary, the PCS converts this /I2/ ordered set into a /I1/ ordered set. For the remainder of the IPG, /I2/ ordered sets should be driven into the PCS and the cordisp bit signal should remain de-asserted.

For example, if a continuous stream of 512 bytes of Ethernet frames and 512 bytes of /I/ ordered sets are set, the following can be observed:

- During the first IPG, all negative disparity /I2/ ordered sets are seen.
- During the next IPG, the period begins with positive disparity /I1/ ordered set. Then all the remaining ordered sets are negative disparity /I2/ ordered sets.
- During the next IPG, all negative disparity /I2/ ordered sets are seen.
- During the next IPG, all period begins with positive disparity /I1/ ordered set. Then all remaining ordered sets are negative disparity /I2/ ordered sets.

Table 11.2 shows the definition of the configuration ordered sets and IDLE ordered sets for Ethernet 1000BASE-X (GigE).

Table 11.2. GigE Configuration and IDLE Ordered Sets Definition

Configuration/IDLE	Code	Ordered Sets	Number of Code	Encoding
Configuration /C/	/C1/	Configuration 1	4	/K28.5//D21.5//Config_Reg/
	/C2/	Configuration 2	4	/K28.5//D2.2//Config_Reg/
IDLE /I/	/I1/	IDLE 1	2	/28.5//D5.6/
	/I2/	IDLE 2	2	/28.5//D16.2/

There is no specific requirement for GigE on AC-coupling or DC-coupling implementation. CertusPro-NX SerDes/PCS supports both AC-coupling and DC-coupling link. A 100 nF AC-coupling capacitor is recommended to be used in an AC coupled link.

11.4.2. XAUI Mode

The XAUI mode of CertusPro-NX SerDes/PCS is intended for the optional interface specified by IEEE802.3 between 10G Ethernet MAC and PHY. In XAUI mode, CertusPro-NX SerDes/PCS is configured as four 2.5 Gbps lanes based on 8B/10B PCS. With Lattice XAUI IP core, the XAUI mode of CertusPro-NX SerDes/PCS fully supports from serial I/O to the XGMII interface.

In XAUI mode, the transmit state machine inside 8B/10B PCS performs translation of XGMII idles to proper ‖A‖, ‖K‖, and ‖R‖ ordered sets according to the IEEE802.3 specifications. Table 11.3 shows the definition of the IDLE ordered sets for Ethernet XAUI.

Table 11.3. XAUI IDLE Ordered Sets Definition

Code	Ordered Sets	Number of Code	Encoding
‖A‖	Sync Column	4	/28.5//28.5//28.5//28.5/
‖K‖	Skip Column	4	/28.0//28.0//28.0//28.0/
‖R‖	Align Column	4	/28.3//28.3//28.3//28.3/

The XAUI receiver needs be AC-coupled to the XAUI, and a 100 nF AC-coupling capacitor is recommended to be used in an AC-coupled link.

11.4.3. SGMII Mode and QSGMII Mode

The Serial Gigabit Media Independent Interface (SGMII) designed by Cisco is to convey network data, and port speed between a 10/100/1000 PHY and a MAC with significantly fewer signal pins than required for GMII. The SGMII interface can operate in both half and full duplex and at all ports speeds. The Quad Serial Gigabit Media Independent Interface (QSGMII) is designed to convey four ports of network data between Ethernet PHY and MAC like SGMII.

The SGMII mode and QSGMII mode of CertusPro-NX SerDes/PCS fully supports from serial I/O to GMII interface. In SGMII mode, SerDes/PCS block is configured as single lane at 1.25 Gbps, based on 8B/10B PCS. In QSGMII mode, SerDes/PCS block is configured as single lane at 5 Gbps, based on 8B/10B PCS.

SGMII and QSGMII support both AC and DC coupled operation. CertusPro-NX SerDes/PCS supports both AC-coupling and DC-coupling link. A 100nF AC-coupling capacitor is recommended to be used in an AC coupled link.

11.4.4. 10GBASE-R Mode

The 10GBASE-R mode of CertusPro-NX SerDes/PCS is intended for the connection between XGMII and 10 Gigabit serial I/O, in 10 Gigabit Ethernet applications specified by IEEE802.3. In 10GBASE-R mode, CertusPro-NX SerDes/PCS is configured as single lane at 10.3125 Gbps, based on 64B/66B PCS.

The 64B/66B PCS encodes eight data octets or control characters, and 2-bit synchronization header into a block. Blocks containing control characters also contain a block type field. Data octets are labeled from D_0 to D_7. Control characters other than ordered sets (/O/), start control character (/S/) and termination control character (/T/) are labeled from C_0 to C_7. The control character for ordered sets are labeled as O_0 or O_4, as it is only valid on the first octet of the XGMII. The control character for start is labeled as S_0 or S_4 for the same reason. The control character for terminate is labeled from T_0 to T_7.

The first two bits of a block are the synchronization header (sync header). Blocks are either data blocks or control blocks. The sync header is 2'b01 for data blocks and 2'b10 for control blocks. Thus, there is always a transition between the first two bits of a block. The remainder of the block contains the payload. The payload is scrambled, and the sync header bypasses the scrambler. Therefore, the sync header is the only position in the block that always contains a transition. This feature of the code is used to obtain block synchronization.

Data blocks contain eight data characters. Control blocks begin with an 8-bit block type field that indicates the format of the remainder of the block. To control blocks containing a Start or Terminate character, the character is implied in the block type field. Other control characters are encoded in a 7-bit control code or a 4-bit /O/ Code. Each control block contains eight characters. Table 11.4 shows the format of the 64B/66B blocks.

Table 11.4. 64B/66B Blocks Formats

Input Data	64B/66B Block								
Data Block Format	**Sync**	**Block Payload**							
$D_0D_1D_2D_3D_4D_5D_6D_7$	01	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
Control Block Format	**Sync**	**Type**	**Payload**						
$C_0C_1C_2C_3C_4C_5C_6C_7$	10	8'h1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
$C_0C_1C_2C_3O_4D_5D_6D_7$	10	8'h2D	C_0	C_1	C_2	C_3	O_4	D_5	D_6	D_7
$C_0C_1C_2C_3S_4D_5D_6D_7$	10	8'h33	C_0	C_1	C_2	C_3		D_5	D_6	D_7
$O_0D_1D_2D_3S_4D_5D_6D_7$	10	8'h66	D_1	D_2	D_3	O_0		D_5	D_6	D_7
$O_0D_1D_2D_3O_4D_5D_6D_7$	10	8'h55	D_1	D_2	D_3	O_0	O_4	D_5	D_6	D_7
$S_0D_1D_2D_3O_4D_5D_6D_7$	10	8'h78	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$O_0D_1D_2D_3C_4C_5C_6C_7$	10	8'h4B	D_1	D_2	D_3	O_0	C_4	C_5	C_6	C_7
$T_0C_1C_2C_3C_4C_5C_6C_7$	10	8'h87		C_1	C_2	C_3	C_4	C_5	C_6	C_7
$D_0T_1C_2C_3C_4C_5C_6C_7$	10	8'h99	D_0		C_2	C_3	C_4	C_5	C_6	C_7
$D_0D_1T_2C_3C_4C_5C_6C_7$	10	8'hAA	D_0	D_1		C_3	C_4	C_5	C_6	C_7
$D_0D_1D_2T_3C_4C_5C_6C_7$	10	8'hB4	D_0	D_1	D_2		C_4	C_5	C_6	C_7
$D_0D_1D_2D_3T_4C_5C_6C_7$	10	8'hCC	D_0	D_1	D_2	D_3		C_5	C_6	C_7
$D_0D_1D_2D_3D_4T_5C_6C_7$	10	8'hD2	D_0	D_1	D_2	D_3	D_4		C_6	C_7
$D_0D_1D_2D_3D_4D_5T_6C_7$	10	8'hE1	D_0	D_1	D_2	D_3	D_4	D_5		C_7
$D_0D_1D_2D_3D_4D_5D_6T_7$	10	8'hFF	D_0	D_1	D_2	D_3	D_4	D_5	D_6	

CertusPro-NX SerDes/PCS 10GBASE-R mode does not support the Auto-Negotiation (AN), Training and Forward Error Correction (FEC) features required by the 10GBASE-KR backplane links. The optional WAN Interface Sublayer (WIS) part of the 10GBASE-R standard is not implemented by CertusPro-NX SerDes/PCS.

11.5. SLVS-EC Mode

The Scalable Low Voltage Signaling with Embedded Clock (SLVS-EC) is a high-speed serial interface technology developed by Sony for the next generation high resolution CMOS Image Sensor (CIS). The SLVS-EC interface provides a unidirectional wide band pixel data transfer from CIS to a Digital Signal Processor (DSP) or other digital devices.

CertusPro-NX SerDes/PCS supports up to eight lanes and up to 5 Gbps baud rate SLVS-EC receiver applications. Table 11.5 shows the baud grades supported by CertusPro-NX SerDes/PCS. In SLVS-EC mode, SerDes/PCS block is configured as specific link rate within Baud Grade 1, 2 or 3, based on 8B/10B PCS.

Table 11.5. SLVS-EC Baud Rate

Baud Grade 1	Baud Grade 2	Baud Grade 3
1152 ~ 1250 Mbps	2304 ~ 2500 Mbps	4608 ~ 5000 Mbps

SLVS-EC, a kind of low-swing serial protocol, the minimum receiver input differential swing is 40 mV for baud grade 3. The interconnection between transmitter and receiver should be DC coupled, considering the differential swing is quite small. Table 11.6 shows the general parameters of receiver characteristics for SLVS-EC.

Table 11.6. General Parameters of Receiver Characteristics for SLVS-EC

Baud Rate	Parameters	Min	Typical	Max
Baud Grade 1 and 2	Receiver Input Differential Zero-Peak Voltage	65 mV	—	285 mV
	Receiver Input Common Mode Voltage	25 mV	—	300 mV
	Receiver Input Differential Resistance	80 Ω	—	110 Ω
Baud Grade 3	Receiver Input Differential Zero-Peak Voltage	40 mV	—	285 mV
	Receiver Input Common Mode Voltage	25 mV	—	300 mV
	Receiver Input Differential Resistance	80 Ω	—	110 Ω

11.6. DisplayPort Mode

DisplayPort (DP) is an industry standard to accommodate the growing broad adoption of digital display technology within the Personal Computer (PC) and Consumer Electronic (CE) industries. It consolidates internal and external connection methods to reduce device complexity, supports necessary features for key cross industry applications, and provides performance scalability to enable the next generation of displays featuring higher color depths, refresh rates, and display resolutions.

The DisplayPort mode of CertusPro-NX SerDes/PCS supports up to four different DisplayPort link rate: RBR, HBR, HBR2, and HBR3. In DisplayPort mode, SerDes/PCS block is configured as specific link rate, based on 8B/10B PCS.

VESA DisplayPort Specification requires external AC-coupling. The recommended Capacitance are shown in Table 11.7. All DisplayPort Main-Link lanes need be AC-coupled, and the AC-coupling capacitors need be placed on the transmitter side. The AC-coupling capacitors may also be placed on the receiver side for better signal performance.

Table 11.7. DisplayPort Recommend AC Capacitance

Link Rate	Minimum	Typical	Maximum
All	75 nF	—	265 nF

11.7. CoaXPress Mode

CoaXPress is defined by Japan Industrial Imaging Association (JIIA) to connect cameras and frame grabbers. It combines the simplicity of coaxial cable with state-of-art high speed data technology, allowing up to 12.5 Gbps data rate per cable, plus device control and power in the same cable.

CoaXPress is a point-to-point scalable interface that consists of one master connection and optional extension connections. Each connection is associated with a coaxial cable. Each connection provides:

- high speed serial downstream connection at up to 12.5 Gbps;
- low speed serial upstream connection at up to 41.6 Mbps (or 20.83 Mpbs).

In other words, both high speed serial downstream connection and low speed serial upstream connection share the same coaxial cable.

The CoaXPress mode of CertusPro-NX SerDes/PCS supports up to five different CoaXPress link rate: CXP-1, CXP-2, CXP-3, CXP-5, and CXP-6. In CoaXPress mode, SerDes/PCS is configured as specific link rate, based on 8B/10B PCS.

12. SerDes/PCS Block Latency

Table 12.1 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 12.1. Transmit/Receive SerDes/PCS Latency

Tx/Rx	SerDes/PCS	Description	Bypass	Min	Max	Unit
Tx Path	8B/10B PCS	Tx Interface FIFO	2	8	12	EPCS clock
		8B/10B Encoder	1	1	1	
		Phase Matching FIFO[1]	1	7	11	
Rx Path		Word Aligner	1	1	3	
		8B/10B Decoder	1	1	1	
		Lane Aligner[1]	1	5	10[2]	
		Clock Compensation FIFO	1	16	20	
		Rx Interface FIFO	2	8	12	
Tx Path	64B/66B PCS	Tx Interface FIFO	1	4	9	EPCS div4 clock
		64B66B Encoder	1	3	3	
		Scrambler	1	1	1	
		Tx Gear Box	N/A	2[3]	3[4]	
Rx Path		Block Aligner	1	1	1	
		Descrambler	1	1	1	
		64B66B Decoder	1	3	3	
		Rx Interface FIFO (with CTC functionality)	2	10	12	
		Rx Gear Box	N/A	2[3]	3[4]	
Tx Path	PMA Only	Tx Interface FIFO	2	8	12	EPCS clock
		Phase Matching FIFO[1]	1	7	11	
Rx Path		Rx Interface FIFO	2	8	12	
Tx Path	SerDes/PMA	Serializer	N/A	3	4	UI
		Gearing Box	N/A	2	2	
Rx Path		Deserializer	N/A	4	5	
		Gearing Box	N/A	1[5]	1[5]	

Notes:

1. Enabled in multi-lane mode. Otherwise, disabled.
2. The actual value should be added with the max_skew between two different lanes.
3. The actual value should be added with one EPCS clock cycle.
4. The actual value should be added with three ECPS clock cycles.
5. The actual value should be added with one EPCS clock cycle.

13. Other Design Considerations

13.1. Simulation of the SerDes/PCS

All SerDes/PCS simulation models are located in the installation directory, under ….\cae_library\simulation\blackbox directory.

13.2. PMA PLL Filter

To achieve a reasonable level of long-term jitter, it is vital to deliver an analog-grade power supply to the PLL. Typically, an R-C or R-L-C filter is usually used, with the “C” being composed of multiple devices to achieve a wide spectrum of noise absorption. Although the circuit is simple, there are specific board layout requirements for CertusPro-NX SerDes/PCS.

- The PLL filter should include series ferrite bead with Equivalent Series Resistance (ESR) approximately equal to 0.28 Ω. The ESR of the ferrite bead is critical. Too high ferrite bead ESR leads to a large IR drop. Too low ferrite bead ESR leads to resonance in the board filter.
- The series resistance of this filter is limited for DC reasons. Generally, it is recommended to see << 5% voltage drop across this device under worst-case conditions.
- The recommended ferrite bead is Murata Bead BLM15EG221SN1 [ESR = 0.28 Ω].
- To achieve good low-frequency cut off, there should be a main ceramic capacitor (~47 µF) in the filter design. Ceramic capacitors are preferred due to their lower parasitic resistance. As the filter also needs to sustain its attenuation into moderately high frequencies, additionally, there is at least one low-ESL and low-ESR capacitor in parallel (~470 nF for quad lane VCCPLLSDx). The routes from the high frequency capacitor(s) to the chip must be kept short, and the capacitor must preferably be placed right underneath the chip on the reverse side of the board. Cursory analysis suggests that a third, very high frequency, capacitor should help reduce noise. But experimental data has not shown any jitter benefit in real applications.
- Board layout around the high-frequency capacitor and the path from there to the pads is critical. It is vital that the quiet ground and power are treated like analog signals.
- The entire VCCPLLSDx and SDx_REFRET wiring path must not couple with any signal aggressors, especially any high swing high slew rate signals such as TTL, CMOS, SSTL signals used in DDR buses. Trace shielding or line spacing management is also required.
- The SDx_REFRET pin serves as the local on-chip ground return path for VCCPLLSDx, so the external board ground must not short with SDx_REFRET under any circumstance.
- The power and ground traces should be short, and run close and parallel as far as possible, with large spacings to adjacent traces. On no account should any connection be made from VCCPLLSDx or SDx_REFRET directly to board power planes; only connect as described above and depicted in Figure 13.1.

A high precision 976 Ω resistor (for 85 Ω differential impedance) is required to be used in 0402 or 0201 package for the external reference resistor connected between SDx_REXT and SDx_REFRET. Better than or equal to 1% precision is required. Table 13.1 shows the recommended external reference resistor for serval differential impedance applications.

- The power dissipation through this resistor is less than 1 mW during calibration. Dissipation is null, if calibrator code is parked at all-zeroes.
- The routes from the reference resistor to the chip must be kept short and the resistor must preferably be placed right underneath the chip on the reverse side of the board.
- Note that the capacitance of signal trace routes in FR4 is typically 4pF/inch. Each additional inch of trace increases RC settling time constant by about 5 ns and hence add extra 20 ns for each step of the calibration process.

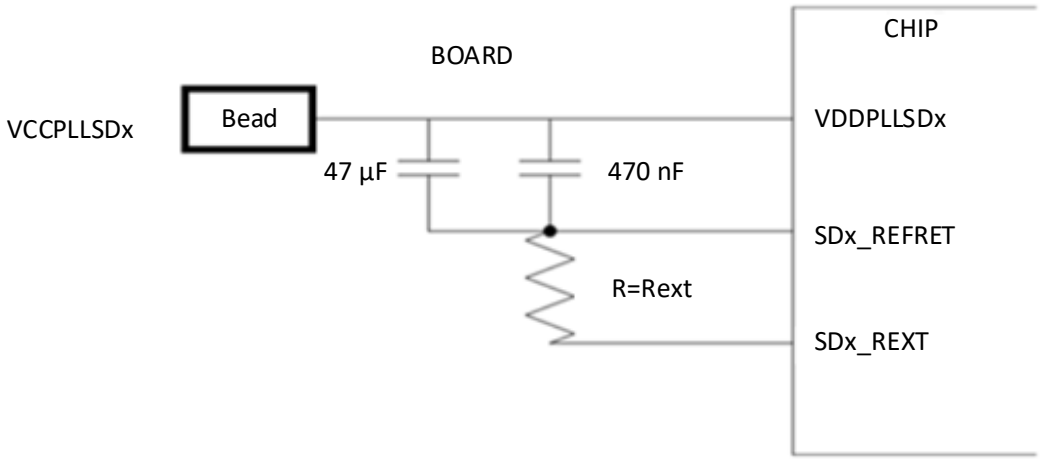

Figure 13.1. Example Connection to Analog Power and Reference Pins

Table 13.1. Recommended External Reference Resistor for Serval Differential Impedance Applications

External Reference Resistor (Rext)	Differential Impedance
909 Ω	80 Ω
976 Ω	85 Ω
1.02 kΩ	90 Ω
1.15 kΩ	100 Ω

13.3. Reference Clock Source Selection

The reference clock can source from per-quad package pins (SDQx_REFCLKP/N), GPLL output, PCLK and dedicated two package pins (SD_EXT0_REFCLKP/N and SD_EXT1_REFCLKP/N). However, there are some design considerations on the selection of reference clock source for specific applications.

When the clock from GPLL output is used as the reference clock for CertusPro-NX SerDes/PCS, the reference clock to the GPLL should be assigned to the dedicated GPLL input pad. However, the GPLL output jitter may not meet system specifications. Reference clock source from GPLL output is not recommended in most applications, because the TxPLL and RxCDR cannot maintain a stable lock state with GPLL output reference clock.

In PCI Express Hard IP mode, the reference clock source from per-quad package pins (SDQx_REFCLKP/N) need be used.

For applications with more than four lanes, the reference clock source from one of the two dedicated package pins (SD_EXT0_REFCLKP/N or SD_EXT1_REFCLKP/N) is recommended.

For applications with several different speed grades and the frequency of the reference clock is different. The solution is to use two reference clock sources with dynamic selection.

13.4. Spread Spectrum Clocking Support

Spread Spectrum Clocking (SSC) is a technique used in electronics design to intentionally modulate the ideal position of the clock edge such that the resulting signal spectrum is spread around the ideal frequency of the clock. In timing circuits, this technique has the advantage of reducing Electromagnetic Interference (EMI) associated with the fundamental frequency of the signal. The amount of EMI a system is allowed to generate is set by various regulatory bodies to ensure systems not interfacing with one another. System spectrum clocking is often used to help meet the regulated EMI requirements. A spread spectrum signal has the disadvantage of having much higher jitter than the un-modulated signal.

Spread spectrum clocking is commonly used for microprocessor clocks, USB and PCI Express reference clocks to reduce EMI. CertusPro-NX SerDes/PCS supports the following optional SSC features defined by PCI Express Base Specification:

- The reference clock can be modulated by +0% to -0.5% from nominal (5000 ppm), referred to as down spreading.
- The modulation rate must be between 30 KHz and 33 KHz.

In PCI Express applications, the Root Complex (RC) is responsible for spreading the reference clock. The Endpoint uses the same clock to pass back the spectrum through the Transmitter. What should be noted is that the reference clock from RC needs to be used in PCI Express Endpoint applications based on CertusPro-NX SerDes/PCS, when the SSC feature has been enabled. Otherwise, reference clock source from local OSC can be used for CertusPro-NX SerDes/PCS. However, using a local OSC as reference clock is not recommended on account of the system compatibility.

The Tx PLL and CDR PLL inside PMA have no capability to generate SSC clock but can be compatible with SSC clock.

Although CertusPro-NX SerDes allows the mixing of a PCI Express channel and other protocol channels within the same quad, using the PCI Express reference clock with SSC can cause a violation of the Gigabit Ethernet, Serial Rapid IO (SRIO), SGMII and QSGMII transmit jitter specifications.

13.5. Unused Quad/Channel and Power Supply

Some power supply pins can be left floating for power saving purpose, when specific quads or channels are unused. Follow guidelines below for unused quads or channels power pins connections:

- For unused quads:
  - Connect all VSSDQ pins to board ground.
  - Leave corresponding VCCAUXSDQx [x = 0, 1] pins floating (not connected).
  - Leave corresponding VCCSDx [x = 0 ~ 7] and VCCPLLSDx [x = 0 ~ 7] pins floating (not connected).
- For unused channels:
  - Connect all VSSDQ pins to board ground.
  - Connect corresponding VCCAUXSDQx [x = 0, 1] to a clean and stable power rail.
  - Connect corresponding VCCSDx [x = 1, 5] and VCCPLLSDx [x = 1, 5] pins to a clean and stable power rail.
  - Connect used VCCSDx [x = 0 ~ 7] and VCCPLLSDx [x = 0 ~ 7] pins to a clean and stable power rail.
  - Leaves other unused VCCSDx [x = 0 ~ 7] and VCCPLLSDx [x = 0 ~ 7] pins floating (not connected).

Note: Channel 1 of the used quad must be powered even though channel 1 is unused in this quad.

Table 13.2 shows the numbering for CertusPro-NX SerDes power pins.

Table 13.2. SerDes Power Pins Numbering

Quad Number	Channel Number	Quad Power Pins	Channel Power Pins	Channel PLL Power Pins
Quad0	Channel 0	VCCAUXSDQ0	VCCSD0	VCCPLLSD0
	Channel 1		VCCSD1	VCCPLLSD1
	Channel 2		VCCSD2	VCCPLLSD2
	Channel 3		VCCSD3	VCCPLLSD3
Quad1	Channel 0	VCCAUXSDQ1	VCCSD4	VCCPLLSD4
	Channel 1		VCCSD5	VCCPLLSD5
	Channel 2		VCCSD6	VCCPLLSD6
	Channel 3		VCCSD7	VCCPLLSD7

13.6. Electrical Idle

Electrical Idle is a steady state condition where the transmitter TxDP and TxDN voltages are held constant with the same value. Electrical Idle is primarily used in power savings and inactive states. CertusPro-NX SerDes/PCS supports three types of Electrical Idle: EI1, EI2, and EI4.

Electrical Idle I (EI1): In this case, both the transmitter output pins (TxDP and TxDN) are held to a steady-state common-mode value and are held to have controlled low output impedance. This mode consumes more power than EI2 and EI4 but requires low recovery time since the AC-Coupling capacitor charge on the board is undisturbed.

Electrical Idle 2 (EI2): In this case, both the transmitter output pins (TxDP and TxDN) are held at 0 potential, but the impedance of the transmitter is still as what is calibrated. DC current consumption of the driver stage is null. This has the consequence of changing the Common Mode voltage from that of normal operation, and hence takes at least 100 µs to settle to or recover form assuming up to 200 nF AC-Coupling capacitor. This mode is mainly designed for PCIe and SATA protocols.

Electrical Idle 4 (EI4): In this case, both the transmitter output pins (TxDP and TxDN) are held to a steady-state common-mode value and are with controlled medium output impedance (about four times higher than EI1). This mode consumes about a quarter of the power used by EI1 but requires more recovery time.

Table 13.3 shows the electrical idle usage in different SerDes/PCS modes.

Table 13.3. Electrical Idle Related Signals

SerDes/PCS Mode	Electrical Idle Operation	Signal Name	Descriptions
MPCS	Transmit at Tx	epcs_txval_i	Deasserted to enable Electrical Idle transmit
	Detection at Rx	epcs_rxidle_o	—
PMA Only	Transmit at Tx	epcs_txval_i	Deasserted to enable Electrical Idle transmit
	Detection at Rx	epcs_rxidle_o	—

13.7. Multiple Data Rate Support

There are only three pre-defined register groups for specific data rate (speed grade). These pre-defined register groups are designed for PCI Express Gen3 mainly.

For protocols having more than three data rates and requiring dynamic switching such as CoaXPress, DisplayPort and embedded DisplayPort, user logic needs update the register setting if the target data rate is not in the pre-defined register groups. However, it may have some impacts on the total time consumed by data rate switching under this condition.

14. SerDes/PCS Generation in Radiant Software

CertusPro-NX SerDes/PCS can be configured and implemented by MPCS foundational IP that is provided by IP Catalog tool in Lattice Radiant software. There are three types of primitives named PCSX1, PCSX2, and PCSX4. PCSX1 is for single channel applications based on CertusPro-NX SerDes/PCS block. PCSX2 and PCSX4 are for multiple channel applications. MPCS foundational IP provides some parameters default settings for specific protocols and implements necessary connections between two or more quads for multiple channel applications (more than four channels).

Implementing SerDes/PCS block with the MPCS foundational IP provided by Radiant IP Catalog tool is recommended.

14.1. Configuration GUI

Figure 14.1 shows the configuration GUI of MPCS foundational IP.

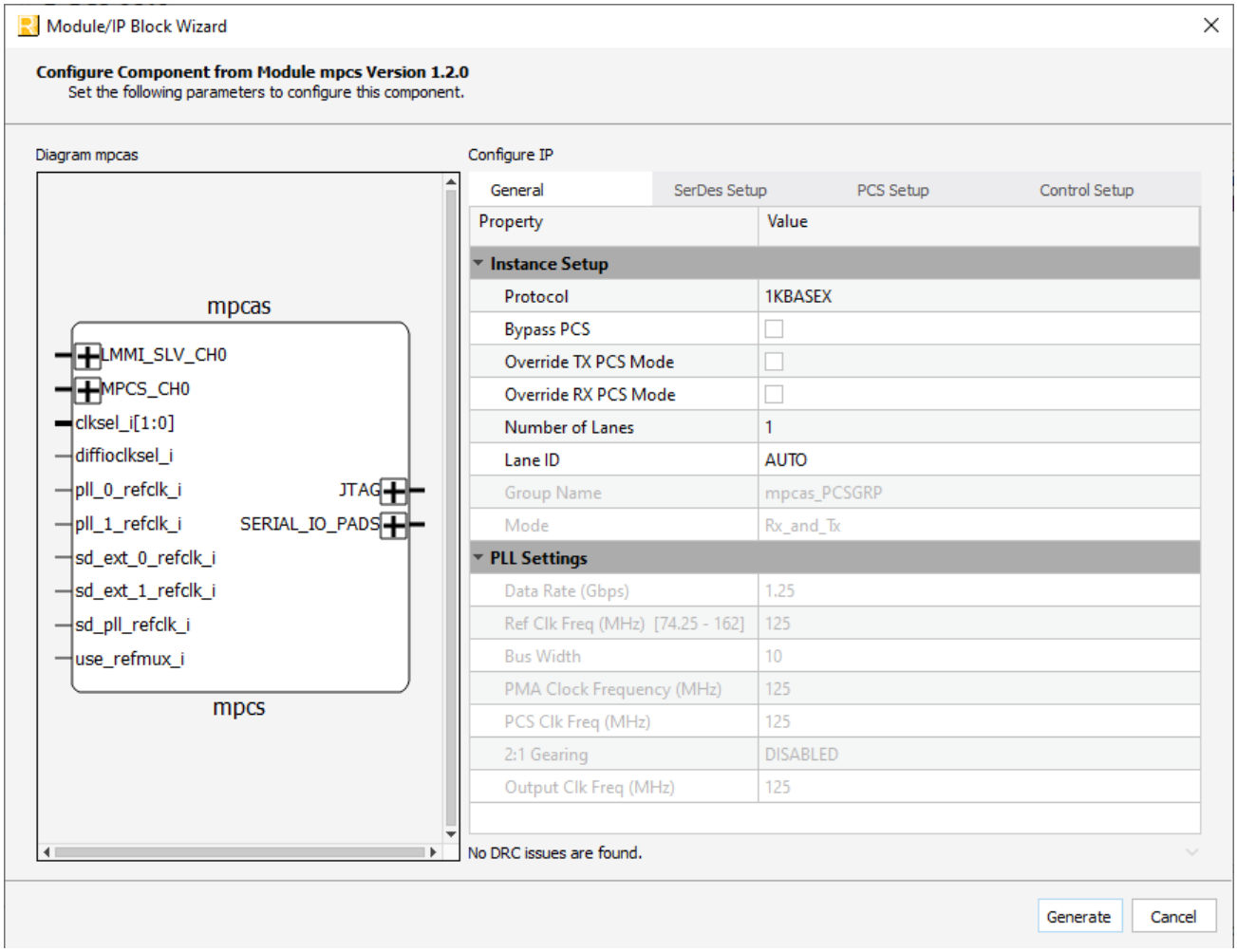

Figure 14.1. MPCS Configuration GUI

Table 14.1 lists the available options for the Protocol. Refer to MPCS Module – Lattice Radiant Software User Guide (FPGA-IPUG-02118) for detailed information about this MPCS foundational IP.

Table 14.1. Protocol Descriptions

Protocol Name	Descriptions
1KBASEX	Ethernet 1000BASE-X or GigE
10GE	Ethernet 10GBASE-R
COAXPRESS	CoaXPress
DP	DisplayPort
eDP	Embedded DisplayPort
QSGMII	QSGMII
PCIE	PCI Express Hard IP mode (not for user implementation)
PCIE-PCS	PCI Express PIPE mode
SGMII	SGMII
SLVS_EC	Sony SLVS-EC
XAUI	XAUI
G8B10B	Generic 8B/10B

14.2. Primitive

Table 14.2 shows the pin-to-pin connection between CertusPro-NX SerDes/PCS primitive and MPCS foundational IP.

Table 14.2. Pin-to-Pin Connection[1]

Primitive Ports	MPCS Foundational IP Port – MPCS/EPCS
CH[3:0]_PMACLKIN	mpcs_clkin_i/ epcs_clkin_i
CH[3:0]_TXCLK	mpcs_tx_usr_clk_i/ epcs_tx_usr_clk_i
CH[3:0]_RXCLK	mpcs_rx_usr_clk_i/ epcs_rx_usr_clk_i
CH[3:0]_CCCLK	mpcs_cc_clk_i
CH[3:0]_PIPE_PCS_TXCLKOUT	mpcs_tx_out_clk_o/ epcs_tx_clk_o
CH[3:0]_RXOUTCLK	mpcs_rx_out_clk_o/ epcs_rx_clk_o
TX_ALIGN_CLKIN	tx_lalign_clkin_i
RX_ALIGN_CLKIN	rx_lalign_clkin_i
TX_ALIGN_CLKOUT	tx_lalign_clk_out_o
RX_ALIGN_CLKOUT	rx_lalign_clk_out_o
CH[3:0]_PERSTN	mpcs_perstn_i/ epcs_rstn_i
CH[3:0]_MPCS_TX_URSTN	mpcs_tx_pcs_rstn_i/ epcs_tx_pcs_rstn_i
CH[3:0]_MPCS_RX_URSTN	mpcs_rx_pcs_rstn_i/ epcs_rx_pcs_rstn_i
CH[3:0]_LMMICLK	lmmi_clk_i
CH[3:0]_LMMIRESETN	lmmi_resetn_i
CH[3:0]_LMMIREQUEST	lmmi_request_i
CH[3:0]_LMMIWR_RDN	lmmi_wr_rdn_i
CH[3:0]_LMMIOFFSET	lmmi_offset_i
CH[3:0]_LMMIWDATA	lmmi_wdata_i

Primitive Ports	MPCS Foundational IP Port – MPCS/EPCS
CH[3:0]_LMMIRDATA_VALID	lmmi_rdata_valid_o
CH[3:0]_LMMIREADY	lmmi_ready_o
CH[3:0]_LMMIRDATA	lmmi_rdata_o
CH[3:0]_SDRXP	sd[n]_rxp_i
CH[3:0]_SDRXN	sd[n]_rxn_i
CH[3:0]_SDTXP	sd[n]_txp_o
CH[3:0]_SDTXN	sd[n]_txn_o
CH[3:0]_SDREXT	sd[n]_rext_i
CH[3:0]_SDREFRET	sd[n]_refret_i
SDQ_REFCLKP	sdq_refclkp_i
SDQ_REFCLKN	sdq_refclkn_i
CH[3:0]_AUXCLK	—
CH[3:0]_CLKREQI	—
CH[3:0]_CLKREQO	—
CH[3:0]_CLKREQOE	—
LALIGN_IN_UP	lalign_in_up_i
LALIGN_IN_DOWN	lalign_in_down_i
LALIGN_OUT_UP	lalign_out_up_o
LALIGN_OUT_DOWN	lalign_out_down_o
CH[3:0]_MISC_CTRL	—
CH[3:0]_PIPE_POWER_DOWN	mpcs_pwrdn_i/ epcs_pwrdn_i
CH[3:0]_PIPE_RATE	mpcs_rate_i/ epcs_rate_i
CH[3:0]_PIPE_TXDETECTRX	—
CH[3:0]_PIPE_RX_EQEVAL	—
CH[3:0]_PIPE_INVALID_REQUEST	—
CH[3:0]_PIPE_RX_PRESET_HINTEN	—
CH[3:0]_PIPE_RX_PRESET_HINT	—
CH[3:0]_PIPE_RX_POL	—
CH[3:0]_PIPE_TX_CLKREQ	mpcs_rxoob_i/ epcs_rxoob_i
CH[3:0]_PIPE_TXCOMM_DIS	mpcs_txdeemp_i/ epcs_txdeemp_i
CH[3:0]_PIPE_RXEI_DIS	mpcs_skipbit_i/ epcs_skipbit_i
CH[3:0]_PIPE_L1PMSSEN	—
CH[3:0]_PIPE_L1X_ENTRY_REQ	mpcs_fomreq_i/ epcs_fomreq_i
CH[3:0]_PIPE_TX_SWING	mpcs_walign_en_i
CH[3:0]_PIPE_TX_MARGIN	—
CH[3:0]_PIPE_LOCAL_GET_PRESET_COEF	mpcs_anxmit_i
CH[3:0]_PIPE_TX_EI	mpcs_txhiz_i/ epcs_txhiz_i
CH[3:0]_PIPE_TX_DEEMPH	16’b0
CH[3:0]_PIPE_TX_DEEMPH_PCS_TX_DATA	mpcs_tx_ch_din_i[79:78]/ epcs_txdata_i[79:78]

Primitive Ports	MPCS Foundational IP Port – MPCS/EPCS
CH[3:0]_ PIPE_RX_PRESET_PCS_TX_DATA	mpcs_tx_ch_din_i [77:74]/ epcs_txdata_i [77:74]
CH[3:0]_PIPE_TX_DATA_VLD	mpcs_tx_ch_din_i [73:56]/ epcs_txdata_i [73:56]
CH[3:0]_ PIPE_REMOTE_FS_PCS_TX_DATA	mpcs_tx_ch_din_i [53:48]/ epcs_txdata_i [53:48]
CH[3:0]_ PIPE_REMOTE_LF_PCS_TX_DATA	{mpcs_tx_ch_din_i [55:54], mpcs_tx_ch_din_i [74:44]}/ {epcs_txdata_i [55:54], epcs_txdata_i [74:44]}
CH[3:0]_ PIPE_PRESET_INDX_PCS_TX_DATA	mpcs_tx_ch_din_i [43:40]/ epcs_txdata_i [43:40]
CH[3:0]_PIPE_TX_SYNC_HDR_PCS_TX_DATA	mpcs_tx_ch_din_i [39:38]/ epcs_txdata_i [39:38]
CH[3:0]_PIPE_TX_START_BLK_PCS_TX_DATA	mpcs_tx_ch_din_i [37]/ epcs_txdata_i [37]
CH[3:0]_PIPE_TX_COMP_PCS_TX_DATA	mpcs_tx_ch_din_i [36]/ epcs_txdata_i [36]
CH[3:0]_PIPE_TX_DATAK_PCS_TX_DATA	mpcs_tx_ch_din_i [35:32]/ epcs_txdata_i [35:32]
CH[3:0]_PIPE_TX_DATA_PCS_TX_DATA	mpcs_tx_ch_din_i[31:0]/ epcs_txdata_i[31:0]
CH[3:0]_PIPE_BLKALGNCTRL	mpcs_rx_deskew_en_i
CH[3:0]_PIPE_WIDTH_2G5	mpcs_rx_fifo_st_o[1:0]
CH[3:0]_PIPE_WIDTH_5G0	mpcs_pwrst_o
CH[3:0]_PIPE_WIDTH_8G0	mpcs_speed_o
CH[3:0]_PIPE_RXEQ_FOM	mpcs_fomrslt_o
CH[3:0]_PIPE_PHY_STATUS	mpcs_phyrdy_o
CH[3:0]_PIPE_RX_EI	mpcs_rxidle_o
CH[3:0]_PIPE_RX_CLKREQ	mpcs_ready_o
CH[3:0]_PIPE_L1X_ENTRY_ACK	mpcs_fomack_o
CH[3:0]_PIPE_RX_START_BLOCK	—
CH[3:0]_PCS_RX_DATA	mpcs_rx_ch_dout_o[79:74]
CH[3:0]_PIPE_TX_PRESET_COEF_PCS_RX_DATA	mpcs_rx_ch_dout_o[69:56]
CH[3:0]_PIPE_LOCAL_GET_TX_PRESET_COEF	{mpcs_ebuf_empty_o, mpcs_ebuf_full_o, mpcs_get_lsync_o, mpcs_rx_get_lalign_o}
CH[3:0]_ PIPE_RX_DATAK_PCS_RX_DATA	mpcs_rx_ch_dout_o [55:52]
CH[3:0]_ PIPE_RXEQ_DIR_PCS_RX_DATA	mpcs_rx_ch_dout_o [51:48]
CH[3:0]_PIPE_LOCAL_LF_PCS_RX_DATA	{mpcs_rx_ch_dout_o[73:72], mpcs_rx_ch_dout_o[47:44]}
CH[3:0]_PIPE_LOCAL_FS_PCS_RX_DATA	{mpcs_rx_ch_dout_o[71:70], mpcs_rx_ch_dout_o[43:40]}
CH[3:0]_PIPE_RX_SYNC_HDR_PCS_RX_DATA	mpcs_rx_ch_dout_o[39:38]
CH[3:0]_PIPE_RX_DATA_VLD_PCS_RX_DATA	mpcs_rx_ch_dout_o[37]
CH[3:0]_PIPE_RX_VLD_PCS_RX_DATA	mpcs_rx_ch_dout_o[36]
CH[3:0]_PIPE_RX_DATA_EN_PCS_RX_DATA	mpcs_rx_ch_dout_o[35]
CH[3:0]_PIPE_RX_STATUS_PCS_RX_DATA	mpcs_rx_ch_dout_o[34:32]
CH[3:0]_PIPE_RX_DATA_PCS_RX_DATA	mpcs_rx_ch_dout_o[31:0]
CH[3:0]_PIPE_RXEQ_DIR_PCS_RX_FIFO_ST	mpcs_rx_fifo_st_o[3:2]
CH[3:0]_PCS_RX_HI_BER	mpcs_rx_hi_ber_o
CH[3:0]_PCS_RX_BLK_LOCK	mpcs_rx_blk_lock_o
CH[3:0]_PCS_TX_FIFO_ST	mpcs_tx_fifo_st_o[3:2]
CH[3:0]_ PIPE_TX_DATA_EN	mpcs_tx_fifo_st_o[1]

Primitive Ports	MPCS Foundational IP Port – MPCS/EPCS
CH[3:0]_ PIPE_LOCAL_GET_TX_COEF_VLD	mpcs_tx_fifo_st_o[0]
CH[3:0]_PIPE_WIDTH_2G5_LL	—
CH[3:0]_PIPE_WIDTH_5G0_LL	—
CH[3:0]_PIPE_WIDTH_8G0_LL	—
CH[3:0]_PIPE_POWER_DOWN_LL	—
CH[3:0]_PIPE_RATE_LL	—
CH[3:0]_PIPE_TX_DETECT_RX_LOOPBACK_LL	—
CH[3:0]_PIPE_RX_EQ_EVAL_LL	—
CH[3:0]_PIPE_INVALID_REQUEST_LL	—
CH[3:0]_PIPE_RX_EQ_EVAL_FEEDBACK_FOM_LL	—
CH[3:0]_PIPE_RX_EQ_EVAL_FEEDBACK_DIR_LL	—
CH[3:0]_PIPE_RX_PRESET_HINT_ENABLE_LL	—
CH[3:0]_PIPE_RX_PRESET_HINT_LL	—
CH[3:0]_PIPE_PHY_STATUS_LL	—
CH[3:0]_PIPE_RX_POLARITY_LL	—
CH[3:0]_PIPE_RX_ELEC_IDLE_LL	—
CH[3:0]_PIPE_RX_CLKREQ_N_LL	—
CH[3:0]_PIPE_TX_CLKREQ_N_LL	—
CH[3:0]_PIPE_TX_CM_DISABLE_LL	—
CH[3:0]_PIPE_RX_EI_DISABLE_LL	—
CH[3:0]_PIPE_L1PMSSEN_LL	—
CH[3:0]_PIPE_TX_SWING_LL	—
CH[3:0]_PIPE_TX_MARGIN_LL	—
CH[3:0]_PIPE_TX_DEEMPH_LL	—
CH[3:0]_PIPE_TX_DEEMPH_WINDOW_LL	—
CH[3:0]_PIPE_LOCAL_FS_LL	—
CH[3:0]_PIPE_LOCAL_LF_LL	—
CH[3:0]_PIPE_LOCAL_GET_PRESET_COEF_LL	—
CH[3:0]_PIPE_LOCAL_GET_PRESET_INDEX_LL	—
CH[3:0]_PIPE_LOCAL_GET_TX_COEF_VALID_LL	—
CH[3:0]_PIPE_LOCAL_GET_TX_PRESET_COEF_LL	—
CH[3:0]_PIPE_REMOTE_FS_LL	—
CH[3:0]_PIPE_REMOTE_LF_LL	—
CH[3:0]_PIPE_REMOTE_EQ_RX_DEEMPH_LL	—
CH[3:0]_PIPE_REMOTE_EQ_RX_PRESET_LL	—
CH[3:0]_PIPE_TX_DATA_ENABLE_LL	—
CH[3:0]_PIPE_TX_DATA_VALID_LL	—
CH[3:0]_PIPE_TX_START_BLOCK_LL	—
CH[3:0]_PIPE_TX_SYNC_HEADER_LL	—
CH[3:0]_PIPE_TX_DATA_LL	—
CH[3:0]_PIPE_TX_DATAK_LL	—
CH[3:0]_PIPE_TX_COMPLIANCE_LL	—
CH[3:0]_PIPE_TX_ELEC_IDLE_LL	—
CH[3:0]_PIPE_RX_DATAEN_LL	—
CH[3:0]_PIPE_RX_DATA_VALID_LL	—
CH[3:0]_PIPE_RX_START_BLOCK_LL	—

Primitive Ports	MPCS Foundational IP Port – MPCS/EPCS
CH[3:0]_PIPE_RX_SYNC_HEADER_LL	—
CH[3:0]_PIPE_BLOCK_ALIGN_CONTROL_LL	—
CH[3:0]_PIPE_RX_DATA_LL	—
CH[3:0]_PIPE_RX_DATAK_LL	—
CH[3:0]_PIPE_RX_VALID_LL	—
CH[3:0]_PIPE_RX_STATUS_LL	—
PIPE_PCLKOUT_LL	—

Note:

1. [n] indicates lane/channel number, and n can be 0 ~ 11.

Appendix A. Configuration Registers

A.1. PMA Registers

A.1.1. Register Address

Table A.1. Register Address

Offset LMMI	Register Name	Access Type	Description
8'd0	reg00	RW	Control register 0.
8'd1	reg01	RW	Clock count for error counter decrement.
8'd2	reg02	RW	Error counter threshold – Rx idle detect maximum latency.
8'd3	reg03	RW	Rx impedance ratio.
8'd4	reg04	RW	Tx PLL F settings and PCLK ratio.
8'd5	reg05	RW	Tx PLL M & N settings.
8'd6	reg06	RW	Rx PLL F settings and PCLK ratio.
8'd7	reg07	RW	Rx PLL M & N settings.
8'd8	reg08	RW	250 ns timer base count.
8'd9	reg09	RW	Tx impedance ratio.
8'd10	reg0a	RW	Tx Post-Cursor ratio.
8'd11	reg0b	RW	Tx Pre-Cursor ratio.
8'd14	reg0e	RW	Power down feature.
8'd25	reg18	RW	Tx amplitude ratio.
8'd33	reg21	RW	CDR PLL frequency comparator maximum difference.
8'd34	reg22	RW	CDR PLL frequency comparator counter.
8'd35	reg23	RW	EI4 mode register.
8'd48	reg30	RO	PMA controller status.
8'd100	reg64	RW	PRBS control register.
8'd101	reg65	RO	PRBS error counter register.
8'd102	reg66	RW	PHY reset override register.
8'd103	reg67	RW	PHY power override register.
8'd116	reg74	RW	PCS loopback control.
8'd117	reg75	RW	Transmit PLL current charge pump.
8'd118	reg76	RW	Receive PLL current charge pump.
8'd121	reg79	RW	CDR PLL manual control.
8'd127	reg7f	RO	PMA Status.
8'd128	reg80	WO	Update settings command register.
8'd217	regd9	RW	Adaptive equalization enable register.
8'd228	rege4	RO	Applied Rx equalization A0 gain.
8'd229	rege5	RO	Applied Rx equalization A1 gain.
8'd230	rege6	RO	Applied Rx equalization A2 gain.

A.1.2. Register Description

Table A.2. Control Register 0 [reg00][1]

Field	Name	Access	Width	Reset	Description
[7]	auto shift	RW	1	1'b1	Defines whether or not Electrical Idle 1 pattern is automatically shifted in SerDes macro after loading the drive pattern. • 1'b1 – automatically shifted. • 1'b0 – not automatically shifted.
[6]	force Rx detect	RW	1	1'b0	Forces the result of PCIe receiver detect operation to be always detected. • 1'b1 – the result is always detected. • 1'b0 – normal operation (this bit should be set to 1'b0 for other protocols).
[5:4]	Reserved	RW	2	2'b00	—
[3]	reserved	RSVD	1	1'b0	__
[2]	CDR PLL delta	RW	1	1'b0	Defines the frequency comparator threshold value. • 1'b1 – Rx clock and Tx clock must be in 0.78% difference range. • 1'b0 – 0.39%
[1]	signal detect threshold	RW	1	1'b0	Defines the Schmitt trigger signal detection threshold used to detect Electrical Idle on Rx. • 1'b1 – threshold is 180 mV (±33%) • 1'b0 – threshold is 125 mV (±40%)
[0]	Tx select Rx feedback	RW	1	1'b0	Tx PLL reference clock source selection. • 1'b1 – driven by CDR PLL loop feedback clock. • 1'b0 – normal operation.

Note:

1. This register can be reprogrammed when the PHY is under reset or when calibration has completed (PMA is ready).

Table A.3. Clock Count for Error Counter Decrement [reg01] [1]

Field	Name	Access	Width	Reset	Description
[7:0]	errcnt_dec	RW	8	8'h20	In PCS driven mode, the PMA control logic counts the number of errors detected by the PCS logic in order to decide how to switch back to frequency lock mode of CDR PLL. This counter is used to decrement the error counter every 16*errcnt_dec[7:0] TxClk clock cycles.

Note:

1. This register can be reprogrammed when the PHY is under reset or when calibration has completed (PMA is ready).

Table A.4. Error Counter Threshold – Rx Idle Detect Max Latency [reg02]

Field	Name	Access	Width	Reset	Description
[7:4]	rxidle_max	RW	4	4'hF	Defines the number of clock cycles required by the Signal Detector to report either electrical idle or valid input data conditions. The least value is 4'h3, because the Signal Detector output is considered as metastable by the PCS logic.
[3:0]	errcnt_thr	RW	4	4'h8	Defines the error counter threshold value after which the CDR PLL switches back to frequency lock.

Table A.5. Rx Impedance Ratio [reg03][1]

Field	Name	Access	Width	Reset	Description
[7:0]	rx_imped_ratio	RW	8	8'h80	Tunes the Rx impedance ratio of the PMA. • 8'h80 – 100 Ω • 8'h55 – 150 Ω corresponds to 2/3 ratio

Note:

1. This register can be reprogrammed when the PHY is under reset or when calibration has completed (PMA is ready).

Table A.6. Tx PLL F Settings and PCLK Ratio [reg04][1]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	2'b00	—
[5:4]	tx_div_mode0	RW	2	2'b10	Defines the ratio between PCLK and TxClk (PMA Clock Divider). PCLK is used by the PCIe PCS logic as well as by the majority of the PMA control logic, and thus is also useful for other protocol in order to reduce the amount of logic requiring high TxClk frequency. • 2'b11 – divided by 4 • 2'b10 – divided by 2 • 2'b00 – divided by 1
[3:0]	TxF	RW	4	4'h4	Defines the Tx PLL F setting. • 4'b0101 – 6 • 4'b0100 – 5 • 4'b0011 – 4 • 4'b0010 – 3 • 4'b0001 – 2 • 4'b0000 – 1

Note:

1. This register can be reprogrammed when the PHY is under reset or when both CDR PLL and Tx PLL are under reset.

Table A.7. Tx PLL M & N Settings [reg05] [1]

Field	Name	Access	Width	Reset	Description
[7]	cnt250ns_max	RSVD	1	1'b0	Internal usage.
[6:5]	TxM	RW	2	2'b10	Defines the Tx PLL M setting. • 2'b11 – 8 • 2'b10 – 4 • 2'b01 – 2 • 2'b00 – 1
[4:0]	TxN	RW	5	5'h04	Defines the Tx PLL N setting. • 5'b10011 – 20 • 5'b01111 – 16 • 5'h01001 – 10 • 5'b00111 – 8 • 5'b00100 – 5

Note:

1. This register can be reprogrammed when the PHY is under reset or when Tx PLL is under reset.

Table A.8. Rx PLL F Settings and PCLK Ratio [reg06] [1]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	2'b00	—
[5:4]	rx_div_mode0	RW	2	2'b10	Defines the ratio between internal Rx CLK and RxClk (PMA Clock Divider). • 2'b11 – divided by 4

Field	Name	Access	Width	Reset	Description
					• 2'b10 – divided by 2 • 2'b00 – divided by 1
[3:0]	TxF	RW	4	4'h4	Defines the Tx PLL F setting. • 4'b0101 – 6 • 4'b0100 – 5 • 4'b0011 – 4 • 4'b0010 – 3 • 4'b0001 – 2 • 4'b0000 – 1

Note:

1. This register can be reprogrammed when the PHY is under reset or when CDR PLL is under reset.

Table A.9. Rx PLL M & N Settings [reg07] [1]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6:5]	RxM	RW	2	2'b10	Defines the CDRPLL M setting. • 2'b11 – 8 • 2'b10 – 4 • 2'b01 – 2 • 2'b00 – 1
[4:0]	RxN	RW	5	5'h04	Defines the CDR PLL N setting. • 5'b10011 – 20 • 5'b01111 – 16 • 5'h01001 – 10 • 5'b00111 – 8 • 5'b00100 – 5

Note:

1. This register can be reprogrammed when the PHY is under reset or both CDR PLL and Tx PLL are under reset.

Table A.10. 250ns Timer Base Count [reg08] [1]

Field	Name	Access	Width	Reset	Description
[7:0]	cnt250ns_max	RW	8	8'h7C	This register defines the base count of a 250 ns event based on Tx CLK (Frequency is F_{PMA}) clock. This counter is used by the CDR PLL and PMA Controller for operations such as receiver detect, Electrical Idle. In case of non-integer value, the base count should be rounded up. Note that this register must be set correctly for all protocols. This register aims to generate an internal timing event every 250 ns or more internally for PMA Controller to calibrate, CDR PLL lock to reference clock and data, receive detect operation. For example, for PCIe Gen1/Gen2/Gen3, Tx CLK is 500 MHz. So that the value of this register should be 8'h7C ((124 + 1) * 2ns = 250ns).

Note:

1. This register can be reprogrammed when the PHY is under reset.

Table A.11. Tx Impedance Ratio [reg09] [1]

Field	Name	Access	Width	Reset	Description
[7:0]	tx_imped_ratio	RW	8	8'h80	Tunes the Tx impedance ratio of the PMA. • 8'h80 – 100 Ω • 8'h55 – 150 Ω corresponds to 2/3 ratio

Note:

1. This register can be reprogrammed when the PHY is under reset or when calibration has completed (PMA is ready).

Table A.12. Tx Post-Cursor Ratio [reg0a] [1]

Field	Name	Access	Width	Reset	Description
[7:0]	tx_pst_ratio	RW	8	8'h15	This register defines the Tx post-cursor ratio for PCI Gen1 speed or non-PCIe protocols. A value of 8'd128 corresponds to 100% (full voltage) whereas a value of 8'd0 corresponds to 0%.

Note:

1. This register can be reprogrammed during normal operation, but effect only appears when the parameter for SerDes (PMA) transmitter has been updated (at the end of calibration, on entry or exit of Tx electrical idle or when reg80 bit 1 has been programmed).

Table A.13. Tx Pre-Cursor Ratio [reg0b] [1]

Field	Name	Access	Width	Reset	Description
[7:0]	tx_pre_ratio	RW	8	8'h00	This register defines the Tx pre-cursor ratio for PCI Gen1 speed or non-PCIe protocols. A value of 8'd128 corresponds to 100% (full voltage) whereas a value of 8'd0 corresponds to 0%.

Note:

1. This register can be reprogrammed during normal operation, but effect only appears when the parameter for SerDes (PMA) transmitter has been updated (at the end of calibration, on entry or exit of Tx electrical idle or when reg80 bit 1 has been programmed).

Table A.14. Power down Feature [reg0e] [1]

Field	Name	Access	Width	Reset	Description
[7:6]	rxidle_msb	RW	2	2'b11	Used as the MSB bits of the activity detector logic, enabling to specify that no activity is detected during up to 61 TxClkp clock cycles. These bits are the 2 MSB bits whereas the rxidle_max[3:0] field of reg02 represent the LSB part.
[5]	force_signal	RW	1	1'b0	Disables the Idle detection circuitry and forces signal detection on the receiver. • 1'b1 – disabling. • 1'b0 – normal operation.
[4]	force_idle	RW	1	1'b0	Disables the Idle detection circuitry and forces Electrical Idle detection on the receive side. • 1'b1 – disabling. • 1'b0 – normal operation.
[3]	NO_FCMP	RW	1	1'b0	Disables the frequency comparator logic of the PCS-driven CDR PLL control logic. • 1'b1 – the frequency comparator logic is not any part of the condition for going from fine-grain lock state to frequency acquisition. • 1'b0 – normal operation.
[2]	PMFF_ALL	RW	1	1'b0	Intended for disabling the function which waits for every active lane to have valid data to transmit to generate a global read enable. • 1'b1 – each lane might start transmitting data with one 500 MHz clock uncertainty (corresponding to 5-bit or 10-bit time depending on the speed of the link).

Field	Name	Access	Width	Reset	Description
					• 1'b0 – normal operation.
[1]	CDR_ERR	RW	1	1'b0	Intended for internally disabling the error counter of the CDR PLL state machine which switches back the CDR PLL to frequency mode acquisition when the number of errors counted is higher than the predefined error threshold. • 1'b1 – disables the error counter. • 1'b0 – normal operation.
[0]	CDR_P1	RW	1	1'b0	Defines the state of the CDR PLL when the PHY is in P1 low power mode. Note that this bit must not be set for application which removes reference clock in P1 mode. • 1'b1 – the CDR PLL is kept alive in frequency lock mode in P1 state which enables a faster recovery time from P1 state, but which also consumes more power (all Rx logic is kept alive and consumes power in P1 state). • 1'b0 – the CDR PLL is put in reset and low-power, enabling to save maximum power.

Note:

1. This register can be reprogrammed when the PHY is under reset or when calibration has completed (PMA is ready), except for bit-2 which can only be modified under reset condition.

Table A.15. Tx Amplitude Ratio [reg18]

Field	Name	Access	Width	Reset	Description
[7:0]	tx_amp_ratio	RW	8	8'h80	This register implements the Tx amplitude ratio used by the Tx driver. A value of 128 corresponds to 100% (full voltage) whereas a value of 0 corresponds to 0%. Values higher than 128 are forbidden. This register is used in PCIe Gen1/Gen2/Gen3 and non-PCIe mode.

Table A.16. CDR PLL Frequency Comparator Maximum Difference [reg21]

Field	Name	Access	Width	Reset	Description
[7:0]	cdrpll_cmp_max	RW	8	8'h14	This register defines the threshold after which the frequency comparator considers that the TxClk and RxClk are too much difference in order to consider that the CDR PLL has lost lock and thus decide to relock the CDR PLL in frequency mode. PPM_threshold = (reg21 – 2) * reg22 (ppm). Example: reg21 = 8'h8, and reg22 = 8'h4, then PPM_threshold = (8 – 2) * (4 * 256) (ppm) = 6144(ppm).

Table A.17. CDR PLL Frequency Comparator Counter [reg22]

Field	Name	Access	Width	Reset	Description
[7:0]	cdrpll_cnt_max	RW	8	8'h4	This register defines the number of 256 clock cycles for evaluating the frequency difference between TxClk and RxClk.

Table A.18. EI4 Mode Register [reg23]

Field	Name	Access	Width	Reset	Description
[7]	EI4	RW	1	1'b0	Electrical Idle selection. • 1'b1 – EI4 instead of EI1. • 1'b0 – EI1.
[6:3]	reserved	RSVD	4	4'b0000	—
[2]	rstcdr_idl	RW	1	1'b0	Reset CDR whenever Electrical Idle is detected on Rx (LOS). • 1'b1 – the CDR is put into reset each time that Electrical

Field	Name	Access	Width	Reset	Description
					Idle is detected on Rx. This bit must not be set in protocol where EI is actively used and requires short exit time (PCIe and USB3). • 1'b0 – do not reset CDR.
[1]	rstcdr_frq	RW	1	1'b1	Reset CDR whenever frequency comparator triggers an error, and direct back the CDR to frequency lock mode (LOL). • 1'b1 – reset. • 1'b0 – do not reset.
[0]	rstcdr_err	RW	1	1'b1	Reset CDR whenever the error gathering function reports an error and direct back the CDR to frequency lock mode (LOL). • 1'b1 – reset. • 1'b0 – do not reset.

Table A.19. PMA Controller Status [reg30]

Field	Name	Access	Width	Reset	Description
[7]	pma_rdy	RO	1	NA	Defines whether PMA has completed its internal calibration sequence after power-up and PHY reset de-assertion. • 1'b1 – completed. • 1'b0 – not yet.
[6]	reserved	RSVD	1	1'b0	—
[5]	arxctle_err	RO	1	NA	Defines Rx CTLE calibration has reached a minimum or maximum value. • 1'b1 – reached. • 1'b0 – not yet reached.
[4]	asch_err	RO	1	NA	Defines if Schmitt offset calibration has reached a minimum or maximum value. • 1'b1 – reached. • 1'b0 – not yet reached.
[3]	arxes_err	RO	1	NA	Defines Rx Offset Dp calibration has reached a minimum or maximum value. • 1'b1 – reached. • 1'b0 – not yet reached.
[2]	arxdm_err	RO	1	NA	Defines Rx Offset Dm calibration has reached a minimum or maximum value. • 1'b1 – reached. • 1'b0 – not yet reached.
[1]	arxdp_err	RO	1	NA	Defines Rx Offset Error Sampler calibration has reached a minimum or maximum value. • 1'b1 – reached. • 1'b0 – not yet reached.
[0]	arxt_err	RO	1	NA	Defines Rx Offset T calibration has reached a minimum or maximum value. • 1'b1 – reached. • 1'b0 – not yet reached.

Table A.20. PRBS Control Register [reg64] [1]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6]	prbs_chk	RW	1	1'b0	PRBS pattern checker control. • 1'b1 – enabled. • 1'b0 – disabled.
[5:4]	reserved	RSVD	2	2'b00	Internal usage.
[3:2]	prbs_typ	RW	2	2'b00	Defines the type of PRBS pattern which is applied. • 2'b11 – PRBS31 • 2'b10 – PRBS23 • 2'b01 – PRBS11 • 2'b00 – PRBS7
[1]	lpbk_en	RW	1	1'b0	Near-End loopback (serial loopback from Tx back to Rx) control. • 1'b1 – the PMA is put in Near-End loopback. • 1'b0 – the PMA is not put in Near-End loopback.
[0]	prbs_gen	RW	1	1'b0	PRBS pattern transmission control. • 1'b1 – starts the PRBS pattern transmission. • 1'b0 – not start the PRBS pattern transmission.

Note:

1. This register can be reprogrammed any time but has functional impact on the functionality as it can configure the SerDes in loopback or generate the PRBS pattern.

Table A.21. PRBS Error Counter Register [reg65]

Field	Name	Access	Width	Reset	Description
[7:0]	prbs_errcnt	RO	8	8'h00	Reports the number of PRBS error detected when the PRBS test is applied. This register is automatically cleared when the prbs_chk bit gets cleared. The PRBS error counter saturates at 254 errors, the 255-count value corresponding to an error code where the CDR PLL is not locked to incoming data. In case of such error code detected, the PRBS test must either wait for longer time for CDR PLL to synchronize on input data before enabling PRBS checker, or simply times out reporting that no data is received at all. Note that the PRBS error counter logic is also with count error, when the PRBS invariant (all zero value) is obtained, considering input data as error data.

Table A.22. PHY Reset Override Register [reg66]

Field	Name	Access	Width	Reset	Description
[7]	rxhf_clkdn	RW	1	1'b0	CDR PLL VCO control. • 1'b1 – disables CDR PLL VCO and shuts down the Rx de-serializer circuitry and RxClk while still maintaining CDR PLLs in lock for intermediate power savings. • 1'b0 – normal operation.
[6]	txhf_clkdn	RW	1	1'b0	Tx PLL VCO control • 1'b1 – disables CDR PLL VCO and shuts down the Tx high frequency trees and TxClk while still maintaining Tx PLLs in lock for intermediate power savings. • 1'b0 – normal operation.
[5]	rxpllrst	RW	1	1'b0	• 1'b1 – reset the CDR PLL. • 1'b0 – normal operation.
[4]	txpllrst	RW	1	1'b0	• 1'b1 – reset the Tx PLL. • 1'b0 – normal operation.

Field	Name	Access	Width	Reset	Description
[3]	rxpll_init	RW	1	1'b0	• 1'b1 – initializes the CDR PLL settings. • 1'b0 – normal operation.
[2]	txpll_init	RW	1	1'b0	• 1'b1 – initializes the Tx PLL settings. • 1'b0 – normal operation.
[1]	rx_hiz	RW	1	1'b0	• 1'b1 – forces Rx driver to HiZ. • 1'b0 – normal operation.
[0]	tx_hiz	RW	1	1'b0	• 1'b1 – forces Tx driver to HiZ. • 1'b0 – normal operation.

Table A.23. PHY Power Override Register [reg67]

Field	Name	Access	Width	Reset	Description
[7:1]	reserved	RSVD	7	7'h0	—
[0]	rx_pwrdn	RW	1	1'b0	Force the Rx PMA logic to be in power-down mode. • 1'b1 – power down Rx PMA. • 1'b0 – normal operation.

Table A.24. Transmit PLL Current Charge Pump [reg75] [1]

Field	Name	Access	Width	Reset	Description
[7:3]	reserved	RSVD	5	5'h0	—
[2:0]	txicp_rate	RW	3	3'b101	These bits define the Tx PLL charge pump current when the PMA is running in PCIe Gen1 speed or in any other non-PCIe protocols.

Note:

1. This register can be reprogrammed when the PHY is under reset.

Table A.25. Receive PLL Current Charge Pump [reg76] [1]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6:4]	rxcdricp_rate	RW	3	3'b011	These bits define the Rx CDR PLL charge pump current when the CDR PLL is frequency locked and running in PCIe Gen1 speed or in any other non-PCIe protocols.
[3]	reserved	RSVD	1	1'b0	—
[2:0]	rxicp_rate	RW	3	3'b011	These bits define the Rx CDR PLL charge pump current when the CDR PLL is phase locked and running in PCIe Gen1 speed or in any other non-PCIe protocols.

Note:

1. This register can be reprogrammed when the PHY is under reset.

Table A.26. PCS Loopback Control [reg74]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6]	rx_polinv	RW	1	1'b0	Rx data polarity control. • 1'b1 – inverts the polarity of data on the received data. • 1'b0 – normal operation.
[5]	tx_polinv	RW	1	1'b0	Tx data polarity control. • 1'b1 – inverts the polarity of data on the transmitted data. • 1'b0 – normal operation.
[4]	eff_lpbk	RW	1	1'b0	Far-end loopback control. • 1'b1 – activates the far-end loopback through PCIe elastic

Field	Name	Access	Width	Reset	Description
					buffer fifo, assuming 0ppm between recovered clock and transmitted clock. In case of different reference clock, the PMA Tx PLL reference clock can use the CDR clock, allowing to re-transmit loopback data through the elastic buffer FIFO put in transparent mode. • 1'b0 – normal operation.
[3]	meso_sync	RO	1	1'b0	Reports if mesochronous clock alignment state machine has completed its process, thus having aligned CDR receive clock to transmit clock. • 1'b1 – completed. • 1'b0 – not yet completed.
[2]	meso_lpbk	RW	1	1'b0	Enables the mesochronous loopback mode which forces PMA received data to be re-transmitted on the PMA Tx interface. This mode requires that no PPM exists between Rx data and Tx data (thus that both sides of the link use the same reference clock), and performs alignment of CDR clock to transmit clock using the PMA CDR PLL skip bit functionality. This alignment is automatically performed by a state machine when this loopback register is set. • 1'b1 – enabling the mesochronous loopback mode, • 1'b0 – normal operation.
[1]	par_lpbk	RW	1	1'b0	Enables the parallel loopback mode which forces the transmitted PIPE 10-bit encoded data to be loopback to the receiver PIPE Rx interface. • 1'b1 – enabling the parallel loopback mode. • 1'b0 – normal operation.
[0]	plesio_lpbk	RW	1	1'b0	Enables the plesiochronous loopback mode which forces the PCS to loopback data from Rx back to Tx after the PCIe elastic buffer function. It is equivalent to PCIe slave loopback except that it is forced by a register instead of controlled by the PCIe MAC layer. • 1'b1 – enabling the plesiochronous loopback mode. • 1'b0 – normal operation.

Table A.27. CDR PLL Manual Control [reg79]

Field	Name	Access	Width	Reset	Description
[7:3]	reserved	RSVD	5	5'h00	—
[2]	fine_grain	RW	1	1'b0	Forces the CDR PLL state machine to fine grain state. • 1'b1 – forces the CDR PLL state machine to fine grain state. • 1'b0 – normal operation.
[1]	coarse_grain	RW	1	1'b0	Forces the CDR PLL state machine to coarse grain mode. In this state, the CDR PLL performs a coarse grain lock on received data enabling to adjust its clock up to 5000 ppm. • 1'b1 – forces the CDR PLL state machine to coarse grain state. • 1'b0 – normal operation.
[0]	freq_lock	RW	1	1'b0	Forces the CDR PLL state machine to frequency lock state. In this state, the CDR PLL does not lock on received data but on reference clock. • 1'b1 – forces the CDR PLL state machine to frequency lock state.

Field	Name	Access	Width	Reset	Description
					• 1'b0 – normal operation.

Table A.28. PMA Status [reg7f]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	2'b00	—
[5]	rxclkstable	RO	1	1'b0	CDR PLL status. • 1'b1 – locked. • 1'b0 – loss of lock.
[4]	txclkstable	RO	1	1'b0	Tx PLL status. • 1'b1 – locked. • 1'b0 – loss of lock.
[3]	cdrdiagout	RO	1	1'b0	CDR PLL internal frequency detector. • 1'b1 – the frequencies of TxClk and RxClk are not within operation bounds. • 1'b0 – the frequencies of TxClk and RxClk are within operation bounds.
[2]	trandet	RO	1	1'b0	Rx activity detection. • 1'b1 – Rx activity detected. • 1'b0 – Electrical idle or no valid signal.
[1]	cdrpllrstb	RO	1	1'b0	CDR PLL reset status. • 1'b1 – reset is released. • 1'b0 – CDR PLL is under reset.
[0]	txpllrstb	RO	1	1'b0	Tx PLL reset status. • 1'b1 – reset is released. • 1'b0 – Tx PLL is under reset.

Table A.29. Update Settings Command Register [reg80]

Field	Name	Access	Width	Reset	Description
[7:1]	reserved	RSVD	7	7'h0	—
[0]	reg_apply	WO	1	1'b0	This bit is transient (read always report 0) where writing 1'b1 to this bit triggers a new computation of PMA settings based on values written into related PMA registers.

Table A.30. Adaptive Equalization Enable Register [regd9]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6:4]	rxeq_algo	RW	3	3'b111	These bits define which equalization algorithm to apply for each data rate. The rxeq_algo[2] is for data rate 2, rxeq_algo[1] is for data rate 1, and rxeq_algo[0] is for data rate 0. • 1'b1 – SS_LMS. • 1'b0 – RL2plus.
[3]	mode_bff	RW	1	1'b1	When this bit is set, the adaptive equalization restart from the A2/A1/A0 gain settings corresponding to the best FOM found during equalization training on exit from electrical idle or other conditions. Otherwise, it restarts from the oldest settings of the history buffer.
[2:0]	rxeq_enable	RW	3	3'b100	These bits enable the adaptive equalization for each data rate. The rxeq_enable[2] is for data rate 2, rxeq_enable[1] is for data rate 1, and rxeq_enable[0] is for data rate 0.

Field	Name	Access	Width	Reset	Description
					• 1'b1 – enable adaptive equalization. • 1'b0 – disable adaptive equalization.

Table A.31. Applied Rx Equalization A0 Gain Register [rege4]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	2'b00	—
[5:0]	RxA0gain	RW	6	—	These bits report the current setting applied to A0 gain setting of the PMA.

Table A.32. Applied Rx Equalization A1 Gain Register [rege5]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	2'b00	—
[5:0]	RxA1gain	RW	6	—	These bits report the current setting applied to A1gain setting of the PMA.

Table A.33. Applied Rx Equalization A2 Gain Register [rege6]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	2'b00	—
[5:0]	RxA2gain	RW	6	—	These bits report the current setting applied to A2 gain setting of the PMA.

A.2. MPCS Registers

A.2.1. Register Address

Table A.34. Register Address

Offset LMMI	Register Name	Access Type	Description
8'h00	reg00	RW	MPCS Data Path Selection.
8'h10	reg10	RW	MPCS Tx Path Control.
8'h11	reg11	RW	8B/10B Encoder Control.
8'h20	reg20	RW	MPCS Rx Path Control.
8'h21	reg21	RW	MPCS Rx Path Status.
8'h22	reg22	RW	8B/10B Decoder Control.
8'h30	reg30	RW	Word Alignment Control.
8'h31	reg31	RW	Primary Word Alignment Pattern Byte 0.
8'h32	reg32	RW	Primary Word Alignment Pattern Byte 1.
8'h33	reg33	RW	Primary Word Alignment Pattern MSB.
8'h34	reg34	RW	Secondary Word Alignment Pattern Byte 0.
8'h35	reg35	RW	Secondary Word Alignment Patter Byte 1.
8'h36	reg36	RW	Secondary Word Alignment Pattern MSB.
8'h37	reg37	RW	Word Alignment Pattern Mask Code Byte 0.
8'h38	reg38	RW	Word Alignment Pattern Mask Code Byte 1.
8'h39	reg39	RW	Word Alignment Pattern Mask Code MSB.
8'h3a	reg3a	RW	Sync_Det FSM Configuration 0.
8'h3b	reg3b	RW	Sync_Det FSM Configuration 1.

Offset LMMI	Register Name	Access Type	Description
8'h3c	reg3c	RW	Sync_Det FSM Configuration 2.
8'h3d	reg3d	RW	Sync_Det FSM Configuration 3.
8'h3e	reg3e	RO	Number of Bit Slipped During Word Alignment.
8'h3f	reg3f	RW	Primary Sync_Det Pattern Byte 0.
8'h40	reg40	RW	Primary Sync_Det Pattern Byte 1.
8'h41	reg41	RW	Primary Sync_Det Pattern Byte 2.
8'h42	reg42	RW	Primary Sync_Det Pattern Byte 3.
8'h43	reg43	RW	Primary Sync_Det Pattern MSB.
8'h44	reg44	RW	Secondary Sync_Det Pattern Byte 0.
8'h45	reg45	RW	Secondary Sync_Det Pattern Byte 1.
8'h46	reg46	RW	Secondary Sync_Det Pattern Byte 2.
8'h47	reg47	RW	Secondary Sync_Det Pattern Byte 3.
8'h48	reg48	RW	Secondary Sync_Det Pattern MSB.
8'h49	reg49	RW	Sync_Det Pattern Mask Code Byte 0.
8'h4a	reg4a	RW	Sync_Det Pattern Mask Code Byte 1.
8'h4b	reg4b	RW	Sync_Det Pattern Mask Code Byte 2.
8'h4c	reg4c	RW	Sync_Det Pattern Mask Code Byte 3.
8'h4d	reg4d	RW	Sync_Det Pattern Mask Code MSB.
8'h50	reg50	RW	Lane Alignment Control.
8'h51	reg51	RW	Maximum Lane-to-lane Skew.
8'h52	reg52	RW	Primary Lane Alignment Pattern Byte 0.
8'h53	reg53	RW	Primary Lane Alignment Pattern Byte 1.
8'h54	reg54	RW	Primary Lane Alignment Pattern Byte 2.
8'h55	reg55	RW	Primary Lane Alignment Pattern Byte 3.
8'h56	reg56	RW	Primary Lane Alignment Pattern MSB.
8'h57	reg57	RW	Secondary Lane Alignment Pattern Byte 0.
8'h58	reg58	RW	Secondary Lane Alignment Pattern Byte 1.
8'h59	reg59	RW	Secondary Lane Alignment Pattern Byte 2.
8'h5a	reg5a	RW	Secondary Lane Alignment Pattern Byte 3.
8'h5b	reg5b	RW	Secondary Lane Alignment Pattern MSB.
8'h5c	reg5c	RW	Lane Alignment Pattern Mask Code.
8'h60	reg60	RW	Clock Frequency Compensation Control.
8'h61	reg61	RW	SKIP Pattern Insertion/Deletion Control.
8'h62	reg62	RW	Elastic FIFO High Water Line.
8'h63	reg63	RW	Elastic FIFO Low Water Line.
8'h64	reg64	RW	Primary SKIP Pattern Byte 0.
8'h65	reg65	RW	Primary SKIP Pattern Byte 1.
8'h66	reg66	RW	Primary SKIP Pattern Byte 2.
8'h67	reg67	RW	Primary SKIP Pattern Byte 3.
8'h68	reg68	RW	Primary SKIP Pattern Byte MSB.
8'h69	reg69	RW	Secondary SKIP Pattern Byte 0.
8'h6a	reg6a	RW	Secondary SKIP Pattern Byte 1.
8'h6b	reg6b	RW	Secondary SKIP Pattern Byte 2.
8'h6c	reg6c	RW	Secondary SKIP Pattern Byte 3.
8'h6d	reg6d	RW	Secondary SKIP Pattern Byte MSB.
8'h6e	reg6e	RW	SKIP Pattern Mask Code.

Offset LMMI	Register Name	Access Type	Description
8’h80	reg80	RW	64B/66B PCS Tx Path Control.
8’h81	reg81	RW	64B/66B PCS Tx FIFO Almost Full Setting.
8’h82	reg82	RW	64B/66B PCS Tx FIFO Almost Empty Setting.
8’h83	reg83	RO	64B/66B PCS Rx Path Control.
8’h84	reg84	RW	64B/66B PCS CTC High Water Line.
8’h85	reg85	RW	64B/66B PCS CTC Low Water Line.
8’h86	reg86	RO	64B/66B PCS Block Align Shift.
8'h90	reg90	RO	10GBASE-R BER Counter.
8'h91	reg91	RO	10GBASE-R Block Error Counter.
8'h92	reg92	RW	10GBASE-R Test Pattern Seed A Byte0.
8'h93	reg93	RW	10GBASE-R Test Pattern Seed A Byte1.
8'h94	reg94	RW	10GBASE-R Test Pattern Seed A Byte2.
8'h95	reg95	RW	10GBASE-R Test Pattern Seed A Byte3.
8'h96	reg96	RW	10GBASE-R Test Pattern Seed A Byte4.
8'h97	reg97	RW	10GBASE-R Test Pattern Seed A Byte5.
8'h98	reg98	RW	10GBASE-R Test Pattern Seed A Byte6.
8’h99	reg99	RW	10GBASE-R Test Pattern Seed A Byte7.
8'h9a	reg9a	RW	10GBASE-R Test Pattern Seed B Byte0.
8'h9b	reg9b	RW	10GBASE-R Test Pattern Seed B Byte1.
8'h9c	reg9c	RW	10GBASE-R Test Pattern Seed B Byte2.
8'h9d	reg9d	RW	10GBASE-R Test Pattern Seed B Byte3.
8'h9e	reg9e	RW	10GBASE-R Test Pattern Seed B Byte4.
8'h9f	reg9f	RW	10GBASE-R Test Pattern Seed B Byte5.
8'ha0	rega0	RW	10GBASE-R Test Pattern Seed B Byte6.
8'ha1	rega1	RW	10GBASE-R Test Pattern Seed B Byte7.
8'ha2	rega2	RO	10GBASE-R Test Pattern Control 0.
8'ha3	rega3	RO	10GBASE-R Test Pattern Control 1.
8'ha4	rega4	RO	10GBASE-R Test Pattern Error Counter Byte0.
8'ha5	rega5	RO	10GBASE-R Test Pattern Error Counter Byte1.
8’hc6	regc6	RO	PMA Controller Status.
8’hc7	regc7	RW	PMA Control.
8’he0	rege0	RW	Loopback Mode Control.
8’he1	rege1	RW	MPCS BIST Control 0.
8’he2	rege2	RW	MPCS BIST Control 1.
8’he3	rege3	RW	User Defined BIST Constant 1 Byte_0.
8’he4	rege4	RW	User Defined BIST Constant 1 Byte_1.
8’he5	rege5	RW	User Defined BIST Constant 1 MSByte.
8’he6	rege6	RW	User Defined BIST Constant 2 Byte_0.
8’he7	rege7	RW	User Defined BIST Constant 2 Byte_1.
8’he8	rege8	RW	User Defined BIST Constant 2 MSByte.
8’he9	rege9	RO	BIST Status 0.
8’hea	regea	RO	BIST Status 1.
8’hfe	regfe	RW1C/RO	PMA and PIPE PHY Status.

A.2.2. Register Description

Table A.35. MPCS Data Path Selection [reg00]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6:5]	rx_pcsmode_sel	RW	2	2'b00	Rx Path Source Selector. Specifies the selected source/interface for the Rx Path. • 2'b11 – PMA only. • 2'b10 – Reserved. • 2'b01 – PMA+64B/66B PCS. • 2'b00 – PMA+8B/10B PCS.
[4]	rx_pcsmode_ovrd	RW	1	1'b0	Rx Path override. Specifies whether or not to override the selected source/interface for the Rx Path. • 1'b1 – use rx_pcsmode_sel to override the MODESEL to set the MPCS mode for the Rx path. • 1'b0 – use the MODESEL to set the MPCS mode for the Rx path.
[3]	reserved	RSVD	1	1'b0	—
[2:1]	tx_pcsmode_sel	RW	2	2'b00	Tx Path Source Selector. Specifies the selected source/interface for the Tx Path. • 2'b11 – PMA only. • 2'b10 – Reserved. • 2'b01 – PMA+64B/66B PCS. • 2'b00 – PMA+8B/10B PCS.
[0]	tx_pcsmode_ovrd	RW	1	1'b0	TX Path override. Specifies whether or not to override the selected source/interface for the Tx Path. • 1'b1 – use rx_pcsmode_sel to override the MODESEL to set the MPCS mode for the Tx path. • 1'b0 – use the MODESEL to set the MPCS mode for the Tx path.

Table A.36. Tx Path Control [reg10]

Field	Name	Access	Width	Reset	Description
[7]	tx_pmfifo_dis	RW	1	1'b0	Phase Matching FIFO. Specifies the Phase Matching FIFO is enabled or disabled (Tx lane-to-lane Deskew). • 1'b1 – Phase Matching FIFO disabled. • 1'b0 – Phase Matching FIFO enabled.
[6]	tx_bond_mask	RW	1	1'b0	Bond Mask. The banded channel mode is defined by mc1_tx_bond_mode signal. • 1'b1 – exclude this channel from bonded channel group. • 1'b0 – do not exclude this channel.
[5]	tx_dbus_20b	RW	1	1'b0	Bus Width. Specifies the internal data bus width for Tx Path. • 1'b1 – internal data bus is 20-bit width. • 1'b0 – internal data bus is 10-bit width. **Note**: this bit is not applicable for 64B/66B PCS mode.
[4]	enc_8b10b_dis	RW	1	1'b0	8B/10B Encoding Enable. Specifies the 8B/10B encoding is enabled or disabled. • 1'b1 – Encoding disabled. • 1'b0 – Encoding enabled.

Field	Name	Access	Width	Reset	Description
[3]	tx_fifo_dis	RW	1	1'b0	Tx FIFO Enable. Specifies the Tx phase compensation FIFO is enabled or disabled. • 1'b1 – Tx phase compensation FIFO disabled. • 1'b0 – Tx phase compensation FIFO enabled.
[2]	tx_gear_en	RW	1	1'b0	Gearing Enable. Specifies the 2:1 Gearing is enabled or disabled. This bit controls the 8B/10B path and PMA Only path gearing logic, as well as the "tx_out_clk" for all PCS modes. • 1'b1 – Gearing is enabled. • 1'b0 – Gearing is disabled. **Note**: In 8B/10B PCS mode and PMA Only mode, the output clock (tx_out_clk) is driven by the divided-by-two clock, if this bit is set to "1". In 64B/66B PCS mode, the output clock (tx_out_clk) is driven by the clock with doubled rate of default output clock, if this bit is set to "1".
[1]	tx_mpcs_rst	RW	1	1'b0	Soft resets the Tx Path not including the register space. • 1'b1 – reset the Tx MPCS path. • 1'b0 – do not reset the Tx MPCS path.
[0]	tx_mpcs_dis	RW	1	1'b0	TX Path Enable. Specifies the MPCS channel Tx Path is enabled or disabled. • 1'b1 – MPCS channel Tx Path is disabled. • 1'b0 – MPCS channel Tx Path is enabled.

Table A.37. 8B/10B Encoder Control [reg11]

Field	Name	Access	Width	Reset	Description
[7:1]	reserved	RSVD	7	—	—
[0]	enc_8b10b_interleave	RW	1	1'b0	8B/10B Encoder Interleave. Specifies whether the 8B/10B Encoder Interleave is enabled or disabled. • 1'b1 – enabled. • 1'b0 – disabled.

Table A.38. Rx Path Control [reg20]

Field	Name	Access	Width	Reset	Description
[7]	rfifo_com_align	RE	1	1'b0	COMMA Byte alignment. Specifies the feature of putting the COMMA byte to LSByte (Byte_0) of the data bus is enabled or disabled. • 1'b1 – disable this feature. • 1'b0 – enable this feature.
[6]	rx_bond_mask	RW	1	1'b0	Bond Mask. The banded channel mode is defined by mc1_rx_bond_mode signal. • 1'b1 – exclude this channel from bonded channel group. • 1'b0 – do not exclude this channel.
[5]	rx_dbus_20b	RW	1	1'b0	Bus Width. Specifies the internal data bus width for Rx Path. • 1'b1 – internal data bus is 20-bit width. • 1'b0 – internal data bus is 10-bit width. **Note**: this bit is not applicable for 64B/66B PCS mode.
[4]	dec_8b10b_dis	RW	1	1'b0	8B/10B Decoding Enable. Specifies the 8B/10B decoding is enabled or disabled. • 1'b1 – decoding disabled. • 1'b0 – decoding enabled.

Field	Name	Access	Width	Reset	Description
[3]	rx_fifo_dis	RW	1	1'b0	Rx FIFO Enable. Specifies the Rx phase compensation FIFO is enabled or disabled. • 1'b1 – Rx phase compensation FIFO disabled. • 1'b0 – Rx phase compensation FIFO enabled.
[2]	rx_gear_en	RW	1	1'b0	Gearing Enable. Specifies the 1:2 Gearing is enabled or disabled. This bit controls the 8B/10B path and PMA Only path gearing logic, as well as the "tx_out_clk" for all PCS modes. • 1'b1 – gearing is enabled. • 1'b0 – gearing is disabled. **Note**: In 8B/10B PCS mode and PMA Only mode, the output clock (rx_out_clk) is driven by the divided-by-two clock if this bit is set to "1". In 64B/66B PCS mode, the output clock (rx_out_clk) is driven by the clock with doubled rate of default output clock, if this bit is set to "1".
[1]	rx_mpcs_rst	RW	1	1'b0	Soft resets the Rx Path not including the register space. • 1'b1 – reset the Rx MPCS path. • 1'b0 – do not reset the Rx MPCS path.
[0]	rx_mpcs_dis	RW	1	1'b0	Rx Path Enable. Specifies the MPCS channel Rx Path is enabled or disabled. • 1'b1 – MPCS channel Rx Path is disabled. • 1'b0 – MPCS channel Rx Path is enabled.

Table A.39. MPCS Rx Path Status [reg21]

Field	Name	Access	Width	Reset	Description
[7:1]	reserved	RSVD	7	—	—
[0]	rfifo_byte_shift	RW	1	1'b0	Byte shift. Specifies the data shift caused by COMMA byte alignment operation. • 1'b1 – there is one byte data shift. • 1'b0 – there is no byte shift.

Table A.40. 8B/10B Decoder Control [reg22]

Field	Name	Access	Width	Reset	Description
[7:1]	reserved	RSVD	7	—	—
[0]	dec_8b10b interleave	RW	1	1'b0	8B/10B decoder interleaving mode control • 1'b0 – disabled. • 1'b1 – enabled.

Table A.41. Word Alignment Control [reg30]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RW	2	—	—
[5]	wa_dis	RW	1	1'b0	Word Alignment. Specifies the Word Alignment Module is enabled or disabled. • 1'b1 – Word Alignment Module is disabled and the input data is bypassed. • 1'b0 – Word Alignment Module is enabled.
[4]	align_2byte_dis	RW	1	1'b0	2-Byte Alignment Pattern. Specifies where the pattern appears in the data bus. • 1'b1 – the pattern can appear at either [9:0] or [19:10] of the data bus. • 1'b0 – in a 2-byte internal data bus width mode, always put the LSByte of the word alignment pattern to [9:0] of the data bus.
[3]	sec_waptn_dis	RW	1	1'b0	Pattern Matching Enable. Specifies whether the secondary word alignment pattern matching is enabled or not. • 1'b1 – Secondary Word Alignment Pattern Matching is disabled. • 1'b0 – Secondary Word Alignment Pattern Matching is enabled.
[2]	wa_ptn_20b	RW	1	1'b0	Word Align Pattern Bit Width. Specifies the bit width that is to be used for the word alignment pattern. • 1'b1 – Word Alignment Pattern is 20-bit width. • 1'b0 – Word Alignment Pattern is 10-bit width; only [9:0] of the 20-bit pattern and mask code are used.
[1]	syncdet_fsm_dis	RW	1	1'b0	Sync_Det FSM. Specifies the "sync_det" FSM is to be used or is to be disabled. • 1'b1 – disable "sync_det" FSM. • 1'b0 – enable "sync_det" FSM.
[0]	auto_wa_dis	RW	1	1'b0	Word Align Mode. Specifies the use of automatic word alignment or use of manual word alignment. • 1'b1 – use manual word alignment mode. • 1'b0 – use automatic word alignment mode.

Table A.42. Primary Word Alignment Pattern Byte 0 [reg31]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_wa_ptn [7:0]	RW	8	8'h7C	Specifies the 20-bit primary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.

Table A.43. Primary Word Alignment Pattern Byte 1 [reg32]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_wa_ptn [17:10]	RW	8	8'h0	Specifies the 20-bit primary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.

Table A.44. Primary Word Alignment Pattern MSB [reg33]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3:2]	pri_wa_ptn[19:18]	RW	2	2'h0	Specifies the 20-bit primary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.
[1:0]	pri_wa_ptn[9: 8]	RW	2	2'h1	Specifies the 20-bit primary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.

Table A.45. Secondary Word Alignment Pattern Byte 0 [reg34]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_wa_ptn [7:0]	RW	8	8'h83	Specifies the 20-bit secondary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.

Table A.46. Secondary Word Alignment Pattern Byte 1 [reg35]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_wa_ptn [17:10]	RW	8	8'h0	Specifies the 20-bit secondary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.

Table A.47. Secondary Word Alignment Pattern MSB [reg36]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3:2]	sec_wa_ptn[19:18]	RW	2	2'h0	Specifies the 20-bit secondary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.
[1:0]	sec_wa_ptn[9: 8]	RW	2	2'h2	Specifies the 20-bit secondary word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied.

Table A.48. Word Alignment Pattern Mask Code Byte 0 [reg37]

Field	Name	Access	Width	Reset	Description
[7:0]	wa_mask_code [7:0]	RW	8	8'h0	Word Align Mask Mode. Specifies the 20-bit word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied. • 1'b1 – the corresponding bit of word alignment pattern is ignored during alignment pattern matching. • 1'b0 – the corresponding bit of word alignment pattern is not ignored during alignment pattern matching.

Table A.49. Word Alignment Pattern Mask Code Byte 1 [reg38]

Field	Name	Access	Width	Reset	Description
[7:0]	wa_mask_code [17:10]	RW	8	8'h0	Word Align Mask Mode. Specifies the 20-bit word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied. • 1'b1 – the corresponding bit of word alignment pattern is ignored during alignment pattern matching. • 1'b0 – the corresponding bit of word alignment pattern is not ignored during alignment pattern matching.

Table A.50. Word Alignment Pattern Mask Code MSB [reg39]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3:2]	wa_mask_code[19:18]	RW	2	2'h0	Word Align Mask Mode. Specifies the 20-bit word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied. • 1'b1 – the corresponding bit of word alignment pattern is ignored during alignment pattern matching. • 1'b0 – the corresponding bit of word alignment pattern is not ignored during alignment pattern matching.
[1:0]	wa_mask_code [9: 8]	RW	2	2'h0	Word Align Mask Mode. Specifies the 20-bit word alignment pattern. In 10-bit width mode, only bits 9 to 0 are applied. • 1'b1 – The corresponding bit of word alignment pattern is ignored during alignment pattern matching. • 1'b0 – The corresponding bit of word alignment pattern is not ignored during alignment pattern matching.

Table A.51. Sync_Det FSM Configuration 0 [reg3a]

Field	Name	Access	Width	Reset	Description
[7:0]	num_cal_sync	RW	8	8'd3	Specifies the number of valid synchronization code groups or ordered sets that "sync_det" FSM must receive to achieve synchronization state.

Table A.52. Sync_Det FSM Configuration 1 [reg3b]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	2	—	—
[5:0]	num_bad_code	RW	6	6'd4	Specifies the number of bad code groups received by "sync_det" FSM to conclude the loss of synchronization.

Table A.53. Sync_Det FSM Configuration 2 [reg3c]

Field	Name	Access	Width	Reset	Description
[7:0]	num_good_code	RW	8	8'd4	Specifies the continuous good code groups received by "sync_det" FSM to reduce the error count by one.

Table A.54. Sync_Det FSM Configuration 3 [reg3d]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	3	—	—
4	set_sync_ptn_dis	RW	1	1'b0	Secondary Sync Detect Pattern. Specifies the secondary detect pattern is enabled or disabled. • 1'b1 – disable secondary sync_det pattern and do not use it for matching. • 1'b0 – use the secondary sync_det pattern for matching.
3	sync_ptn_10b	RW	1	1'b1	Sync Detect Pattern. Specifies the pattern code. • 1'b1 – the sync_det pattern is 10b code. • 1'b0 – the sync_det pattern is 8b code.

Field	Name	Access	Width	Reset	Description
2	sync_ptn_align	RW	1	1'b0	Pattern Alignment check. Specifies the pattern alignment check is enabled or disabled. • 1'b1 – the first byte of sync_det pattern must appear on N-byte boundary, where N is the sync_det pattern length defined by sync_ptn_len register. Once a sync_det pattern is detected, the sync_det FSM applies this creation to the validity check of subsequent incoming data. • 1'b0 – disable the sync detect pattern alignment check.
[1:0]	sync_ptn_len	RW	2	2'b00	Pattern Length. Specifies the length of the pattern. • 2'b1x – the length is 4. • 2'b01 – the length is 2. • 2'b00 – the length is 1.

Table A.55. Number of Bit Slipped during Word Alignment [reg3e]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	3	—	—
[4:0]	num_bit_slipped	RO	5	5'b0	Code Group Length. Specifies the length of the code group. • 5'b10011 – 19 bits are slipped. *19 is the maximum bit number.* • … • 5'b00010 – 2 bits are slipped. • 5'b00001 – 1 bit is slipped. • 5'b00000 – there is no bit slipped.

Table A.56. Primary Sync_Det Pattern Byte 0 [reg3f]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_sdptn_byte0	RW	8	8'h83	Primary Sync_Det Pattern Byte 0.

Table A.57. Primary Sync_Det Pattern Byte 1 [reg40]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_sdptn_byte1	RW	8	8'h0	Primary Sync_Det Pattern Byte 1.

Table A.58. Primary Sync_Det Pattern Byte 2 [reg41]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_sdptn_byte2	RW	8	8'h0	Primary Sync_Det Pattern Byte 2.

Table A.59. Primary Sync_Det Pattern Byte 3 [reg42]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_sdptn_byte3	RW	8	8'h0	Primary Sync_Det Pattern Byte 3.

Table A.60. Primary Sync_Det Pattern Byte MSB [reg43]

Field	Name	Access	Width	Reset	Description
[7:6]	pri_sdptn_byte3 [9:8]	RW	2	2'h0	Primary Sync_Det Pattern MSB Register reflects the bits 9 to 8 of primary Sync_Det pattern byte 3 to 0.
[5:4]	pri_sdptn_byte2 [9:8]	RW	2	2'h0	
[3:2]	pri_sdptn_byte1 [9:8]	RW	2	2'h0	
[1:0]	pri_sdptn_byte0 [9:8]	RW	2	2'h2	

Table A.61. Secondary Sync_Det Pattern Byte 0 [reg44]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_sdptn_byte0	RW	8	8'h7c	Secondary Sync_Det Pattern Byte 0.

Table A.62. Secondary Sync_Det Pattern Byte 1 [reg45]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_sdptn_byte1	RW	8	8'h0	Secondary Sync_Det Pattern Byte 1.

Table A.63. Secondary Sync_Det Pattern Byte 2 [reg46]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_sdptn_byte2	RW	8	8'h0	Secondary Sync_Det Pattern Byte 2.

Table A.64. Secondary Sync_Det Pattern Byte 3 [reg47]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_sdptn_byte3	RW	8	8'h0	Secondary Sync_Det Pattern Byte 3.

Table A.65. Secondary Sync_Det Pattern Byte MSB [reg48]

Field	Name	Access	Width	Reset	Description
[7:6]	sec_sdptn_byte3 [9:8]	RW	2	2'h0	Secondary Sync_Det Pattern MSB Register reflects the bits 9 to 8 of secondary Sync_Det pattern byte 3 to 0.
[5:4]	sec_sdptn_byte2 [9:8]	RW	2	2'h0	
[3:2]	sec_sdptn_byte1 [9:8]	RW	2	2'h0	
[1:0]	sec_sdptn_byte0 [9:8]	RW	2	2'h1	

Table A.66. Sync_Det Pattern Mask Code Byte 0 [reg49]

Field	Name	Access	Width	Reset	Description
[7:0]	sdptn_mask_byte0	RW	8	8'h7c	Sync_Det Pattern Mask Code Byte 0.

Table A.67. Sync_Det Pattern Mask Code Byte 1 [reg4a]

Field	Name	Access	Width	Reset	Description
[7:0]	sdptn_mask_byte1	RW	8	8'h0	Sync_Det Pattern Mask Code Byte 1.

Table A.68. Sync_Det Pattern Mask Code Byte 2 [reg4b]

Field	Name	Access	Width	Reset	Description
[7:0]	sdptn_mask_byte2	RW	8	8'h0	Sync_Det Pattern Mask Code Byte 2.

Table A.69. Sync_Det Pattern Mask Code Byte 3 [reg4c]

Field	Name	Access	Width	Reset	Description
[7:0]	sdptn_mask_byte3	RW	8	8'h0	Sync_Det Pattern Mask Code Byte 3.

Table A.70. Sync_Det Pattern Mask Code MSB [reg4d]

Field	Name	Access	Width	Reset	Description
[7:6]	sdptn_mask_byte3[9:8]	RW	2	2'h0	Sync_Det Pattern Mask Code MSB Register reflects the bits 9 to 8 of Sync_Det pattern mask code byte 3 to 0.
[5:4]	sdptn_mask_byte2[9:8]	RW	2	2'h0	
[3:2]	sdptn_mask_byte1[9:8]	RW	2	2'h0	
[1:0]	sdptn_mask_byte0[9:8]	RW	2	2'h0	

Table A.71. Lane Alignment Control [reg50]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	6	—	—
[4]	sec_laptn_en	RW	1	1'b0	Secondary Lane Alignment. Specifies if the secondary lane alignment pattern matching is enabled or disabled. • 1'b1 – Pattern Matching is enabled. • 1'b0 – Pattern Matching is disabled.
[3:2]	lalign_ptn_len	RW	2	2'b0	Lane Alignment Pattern Length. Specifies the lane alignment pattern length in byte. • 2'b1x – 4-byte. • 2'b01 – 2-byte. • 2'b00 – 1-byte.
[1]	lalign_10b	RW	1	1'b0	Lane Alignment Coding Mode. Specifies the Lane Alignment coding scheme of the input data. • 1'b1 – input data is in 10b code mode. • 1'b0 – input data is in 8b code mode.
[0]	lalign_en	RW	1	1'b0	Lane Alignment Enable. Specifies the lane alignment is enabled or disabled. • 1'b1 – Lane Alignment is enabled. • 1'b0 – Lane Alignment is disabled.

Table A.72. Maximum Lane-to-lane Skew [reg51]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	6	—	—
[3:0]	max_lskew	RW	4	4'h0	Maximum Lane-to-lane skew specified in byte. If the number of lane is less or equal to 4, these bits should be configured as expected maximum lane-to-lane skew + 1. If the number of lane is more than 4, these bits should be configured as the expected maximum lane-to-lane skew + 2. • 4'd10 – 10-byte skew. • … • 4'd2 – 2-byte skew. • 4'd1 – 1-byte skew. • 4'd0 – reserved. **Note:** The maximum lane-to-lane skew that can be handled by hardware is 10-byte. These bits must be correctly configured, and hardware uses these bits without any checking.

Table A.73. Primary Lane Alignment Pattern Byte 0 [reg52]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_laptn_byte0	RW	8	8'h7c	Primary Lane Alignment Pattern Byte 0.

Table A.74. Primary Lane Alignment Pattern Byte 1 [reg53]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_laptn_byte1	RW	8	8'h0	Primary Lane Alignment Pattern Byte 1.

Table A.75. Primary Lane Alignment Pattern Byte 2 [reg54]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_laptn_byte2	RW	8	8'h0	Primary Lane Alignment Pattern Byte 2.

Table A.76. Primary Lane Alignment Pattern Byte 3 [reg55]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_laptn_byte3	RW	8	8'h0	Primary Lane Alignment Pattern Byte 3.

Table A.77. Primary Lane Alignment Pattern Byte MSB [reg56]

Field	Name	Access	Width	Reset	Description
[7:6]	pri_laptn_byte3[9:8]	RW	2	2'h0	Primary Lane Alignment Pattern MSB Register reflects the bits 9 to 8 of primary lane alignment pattern byte 3 to 0.
[5:4]	pri_laptn_byte2[9:8]	RW	2	2'h0	
[3:2]	pri_laptn_byte1[9:8]	RW	2	2'h0	
[1:0]	pri_laptn_byte0[9:8]	RW	2	2'h1	

Table A.78. Secondary Lane Alignment Pattern Byte 0 [reg57]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte0	RW	8	8'h7c	Secondary Lane Alignment Pattern Byte 0.

Table A.79. Secondary Lane Alignment Pattern Byte 1 [reg58]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte1	RW	8	8'h0	Secondary Lane Alignment Pattern Byte 1.

Table A.80. Secondary Lane Alignment Pattern Byte 2 [reg59]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte2	RW	8	8'h0	Secondary Lane Alignment Pattern Byte 2.

Table A.81. Secondary Lane Alignment Pattern Byte 3 [reg5a]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte3	RW	8	8'h0	Secondary Lane Alignment Pattern Byte 3.

Table A.82. Secondary Lane Alignment Pattern Byte MSB [reg5b]

Field	Name	Access	Width	Reset	Description
[7:6]	sec_laptn_byte3[9:8]	RW	2	2'h0	Secondary Lane Alignment Pattern MSB Register reflects the bits 9 to 8 of secondary lane alignment pattern byte 3 to 0.
[5:4]	sec_laptn_byte2[9:8]	RW	2	2'h0	
[3:2]	sec_laptn_byte1[9:8]	RW	2	2'h0	
[1:0]	sec_laptn_byte0[9:8]	RW	2	2'h1	

Table A.83. Lane Alignment Pattern Mask Code [reg5c]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3:0]	lalign_mask_code	RW	4	4'h0	Lane Alignment Pattern mask code, bit 3 for pattern byte 3 and bit 0 for pattern byte 0. • 1'b1 – the corresponding byte of lane alignment pattern is ignored during alignment pattern matching. • 1'b0 – the corresponding byte of lane alignment pattern is not ignored during alignment pattern matching.

Table A.84. Clock Frequency Compensation Control [reg60] [1]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	3	—	—
[5]	sec_skip_en	RW	1	1'b0	Enable Secondary Skip. Specifies the secondary skip pattern is enabled or disabled. • 1'b1 – Pattern Matching is enabled. • 1'b0 – Pattern Matching is disabled.
[4:3]	skip_ptn_len	RW	2	2'b00	Skip Pattern Length. Specifies the skip pattern length in bytes. • 2'b1x – 4-byte. • 2'b01 – 2-byte. • 2'b00 – 1-byte.
[2]	clk_comp_10b	RW	1	1'b0	Clock Compensation Coding Mode. Specifies the clock compensation coding scheme for the input data. • 1'b1 – input data is in 10b code mode. • 1'b0 – input data is in 8b code mode.
[1]	ctc_fifo_en	RW	1	1'b0	Clock Compensation FIFO. Specifies the Clock Compensation FIFO is enabled or disabled. • 1'b1 – Clock Compensation FIFO is enabled. • 1'b0 – Clock Compensation FIFO is disabled.
[0]	clk_comp_en	RW	1	1'b0	Enable Clock Frequency Compensation. Specifies the clock frequency compensation is enabled or disabled. • 1'b1 – Clock Frequency Compensation is disabled. • 1'b0 – Clock Frequency Compensation is enabled.

Note:

1. The combination of "ctc_fifo_en" and "clk_comp_en" are listed below: {ctc_fifo_en, clk_comp_en}
   - 2'b0x – bypass the input date of this module.
   - 2'b10 – the module works as asynchronous FIFO without clock frequency compensation.
   - 2'b11 – the module performs clock frequency compensation function.

Table A.85. SKIP Pattern Insertion/Deletion Control [reg61] [1]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	—	—
[6:2]	ctc_repeat_wait	RW	5	5'h0	Clock Correction Frequency Control. It defines the minimum number of clock cycles between two clock correction events, including SKIP deletion and insertion. • x(x=1 to 31) – The next time of SKIP deletion or insertion must be x clock cycles away from the current SKIP deletion or insertion. • 1'b0 – on constraint on SKIP deletion and insertion.
[1:0]	min_ipg_cnt	RW	2	2'h0	Minimum SKIP pattern count. Used to guarantee the minimum number of bytes between packets (that is the minimum Inter Packet Gap) after SKIP deletion. • 2'b11 – at least three SKIP pattern is kept in IPG. • 2'b10 – at least two SKIP pattern is kept in IPG. • 2'b01 – at least one SKIP pattern is kept in IPG. • 2'b00 – no constraint on SKIP deletion.

Note:

1. The length in byte of SKIP pattern is defined in "skp_ptn_len" domain register "Clock Frequency Compensation Control".

Table A.86. Elastic FIFO High Water Line [reg62]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	3	—	—
[4:0]	high_water_line	RW	5	5'b10110	Specifies the Clock Compensation FIFO high water line. Mean is 5'b10000.

Table A.87. Elastic FIFO Low Water Line [reg63]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	3	—	—
[4:0]	low_water_line	RW	5	5'b01010	Specifies the Clock Compensation FIFO low water line. Mean is 5'b10000.

Table A.88. Primary SKIP Pattern Byte 0 [reg64]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_skip_byte0[7:0]	RW	8	8'h7c	Primary SKIP Pattern Byte 0.

Table A.89. Primary SKIP Pattern Byte 1 [reg65]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_skip_byte1[7:0]	RW	8	8'h0	Primary SKIP Pattern Byte 1.

Table A.90. Primary SKIP Pattern Byte 2 [reg66]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_skip_byte2[7:0]	RW	8	8'h0	Primary SKIP Pattern Byte 2.

Table A.91. Primary SKIP Pattern Byte 3 [reg67]

Field	Name	Access	Width	Reset	Description
[7:0]	pri_skip_byte3[7:0]	RW	8	8'h0	Primary SKIP Pattern Byte 3.

Table A.92. Primary SKIP Pattern MSB [reg68]

Field	Name	Access	Width	Reset	Description
[7:6]	pri_skip_byte3[9:8]	RW	2	2'h0	Primary SKIP Pattern MSB Register reflects the bits 9 to 8 of Primary SKIP Pattern byte 3 to 0.
[5:4]	pri_skip_byte2[9:8]	RW	2	2'h0	
[3:2]	pri_skip_byte1[9:8]	RW	2	2'h0	
[1:0]	pri_skip_byte0[9:8]	RW	2	2'h1	

Table A.93. Secondary SKIP Pattern Byte 0 [reg69]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte0[7:0]	RW	8	8'h7c	Secondary SKIP Pattern Byte 0.

Table A.94. Secondary SKIP Pattern Byte 1 [reg6a]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte1[7:0]	RW	8	8'h0	Secondary SKIP Pattern Byte 1.

Table A.95. Secondary SKIP Pattern Byte 2 [reg6b]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte2[7:0]	RW	8	8'h0	Secondary SKIP Pattern Byte 2.

Table A.96. Secondary SKIP Pattern Byte 3 [reg6c]

Field	Name	Access	Width	Reset	Description
[7:0]	sec_laptn_byte3[7:0]	RW	8	8'h0	Secondary SKIP Pattern Byte 3.

Table A.97. Secondary SKIP Pattern MSB [reg6d]

Field	Name	Access	Width	Reset	Description
[7:6]	sec_skip_byte3[9:8]	RW	2	2'h0	Secondary SKIP Pattern MSB Register reflects the bits 9 to 8 of Secondary SKIP Pattern byte 3 to 0.
[5:4]	sec_skip_byte2[9:8]	RW	2	2'h0	
[3:2]	sec_skip_byte1[9:8]	RW	2	2'h0	
[1:0]	sec_skip_byte0[9:8]	RW	2	2'h1	

Table A.98. SKIP Pattern Mask Code [reg6e]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3:0]	skp_mask_code	RW	4	4'h0	SKIP Pattern Mask Code. Specifies whether the SKIP Pattern is ignored or not. • 1'b1 – The corresponding byte of SKIP pattern is ignored during SKIP pattern matching. • 1'b0 – The corresponding byte of SKIP pattern is not ignored during SKIP pattern matching.

Table A.99. 64B/66B PCS Tx Path Control [reg80]

Field	Name	Access	Width	Reset	Description
[7:2]	reserved	RSVD	6	—	—
[1]	enc_64b66b_dis	RW	1	1'b0	Enable 64B/66B Encoder. Specifies the 64B/66B Encoder is enabled or disabled. • 1'b1 – 64B/66B Encoder is disabled. • 1'b0 – 64B/66B Encoder is enabled.
[0]	src_64b66b_dis	RW	1	1'b0	Enable 64B/66B Scrambler. Specifies the 64B/66B Scrambler is enabled or disabled. • 1'b1 – 64B/66B Scrambler is disabled. • 1'b0 – 64B/66B Scrambler is enabled.

Table A.100. 64B/66B PCS Tx FIFO Almost Full Setting Control [reg81]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	6	—	—
[3:0]	tx_fifo_af	RW	4	4'hC	64B/66B Tx FIFO Almost Full. When the number of blocks residing in the FIFO is more than this setting, the "almost full" status is reported.

Table A.101. 64B/66B PCS Tx FIFO Almost Empty Setting Control [reg82]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	6	—	—
[3:0]	tx_fifo_ae	RW	4	4'h4	64B/66B Tx FIFO Almost Empty. When the number of blocks residing in the FIFO is less than this setting, the "almost empty" status is reported.

Table A.102. 64B/66B PCS Rx Path Control [reg83]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	4	—	—
[5]	del_os_cont	RW	1	1'b0	• 1'b1 – Allow continuously delete sequence ordered set when doing clock frequency compensation. • 1'b0 – Only delete one sequence ordered set each time when doing clock frequency compensation.
[4]	pcs_64b66b_nogpll	RW	1	1'b0	• 1'b1 – No GPLL is used to generate user clock. • 1'b0 – GPLL is used to generate user clock.
[3]	balign_64b66b_dis	RW	1	1'b0	Enable 64B/66B Block Aligner. Specifies the 64B/66B block aligner is enabled or disabled. • 1'b1 – 64B/66B block aligner is disabled. • 1'b0 – 64B/66B block aligner is enabled.
[2]	ctc_64b66b_dis	RW	1	1'b0	Enable 64B/66B CTC. Specifies the 64B/66B Clock Frequency Compensation is enabled or disabled. • 1'b1 – 64B/66B Clock Frequency Compensation is disabled. • 1'b0 – 64B/66B Clock Frequency Compensation is enabled.
[1]	dec_64b66b_dis	RW	1	1'b0	Enable 64B/66B Decoder. Specifies the 64B/66B Decoder is enabled or disabled. • 1'b1 – 64B/66B Decoder is disabled. • 1'b0 – 64B/66B Decoder is enabled.

Field	Name	Access	Width	Reset	Description
[0]	descr_64b66b_dis	RW	1	1'b0	Enable 64B/66B Descrambler. Specifies the 64B/66B Descrambler is enabled or disabled. • 1'b1 – 64B/66B Descrambler is disabled. • 1'b0 – 64B/66B Descrambler is enabled.

Table A.103. 64B/66B PCS CTC High Water Line Control [reg84]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	3	—	—
[4:0]	ctc_64b66b_high	RW	5	5'h0C	64B/66B PCS CTC High water line reflects the high water line of clock frequency compensation.

Table A.104. 64B/66B PCS CTC Low Water Line Control [reg85]

Field	Name	Access	Width	Reset	Description
[7:5]	reserved	RSVD	3	—	—
[4:0]	ctc_64b66b_low	RW	5	5'h4	64B/66B PCS CTC Low water line reflects the low water line of clock frequency compensation.

Table A.105. 64B/66B PCS Block Align Shift [reg86]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	—	—
[6:0]	balign_64b66b_shift	RO	7	7'h0	64B/66B PCS Block Align Shift register reflects the bit shifting of block alignment.

Table A.106. 10GBASE-R BER Counter [reg90]

Field	Name	Access	Width	Reset	Description
[7:6]	reserved	RSVD	1	—	—
[5:0]	ber_count	RO/CR	6	6'h0	10GBASE-R BER Counter register reflects the BER counter is a six-bit-count as defined by the ber_count variable in 49.2.14.2 for 10GBASE-R. These bits are reset to all zeros when the register is read or upon execution of the PCS reset. These bits are to be held at all ones in the case of overflow.

Table A.107. 10GBASE-R Block Error Counter [reg91]

Field	Name	Access	Width	Reset	Description
[7:0]	errored_block_count	RO/CR	8	8'h0	10GBASE-R Block Error Counter register reflects the errored blocks counter is an eight-bit count defined by the errored_block_count counter specified in 49.2.14.2 for 10GBASE-R. These bits are to be reset to all zeros when the errored blocks count is read or upon execution of the PCS reset. These bits are to be held at all ones in the case of overflow.

Table A.108. 10GBASE-R Test Pattern Seed A Byte 0 [reg92]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[7:0]	RW	8	8'h0	This register defines byte 0 of 10GBASE-R PCS test pattern seed A.

Table A.109. 10GBASE-R Test Pattern Seed A Byte 1 [reg93]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[15:8]	RW	8	8'h0	This register defines byte 1 of 10GBASE-R PCS test pattern seed A.

Table A.110. 10GBASE-R Test Pattern Seed A Byte 2 [reg94]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[23:16]	RW	8	8'h0	This register defines byte 2 of 10GBASE-R PCS test pattern seed A.

Table A.111. 10GBASE-R Test Pattern Seed A Byte 3 [reg95]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[31:24]	RW	8	8'h0	This register defines byte 3 of 10GBASE-R PCS test pattern seed A.

Table A.112. 10GBASE-R Test Pattern Seed A Byte 4 [reg96]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[39:32]	RW	8	8'h0	This register defines byte 4 of 10GBASE-R PCS test pattern seed A.

Table A.113. 10GBASE-R Test Pattern Seed A Byte 5 [reg97]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[47:40]	RW	8	8'h0	This register defines byte 5 of 10GBASE-R PCS test pattern seed A.

Table A.114. 10GBASE-R Test Pattern Seed A Byte 6 [reg98]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[55:48]	RW	8	8'h0	This register defines byte 6 of 10GBASE-R PCS test pattern seed A.

Table A.115. 10GBASE-R Test Pattern Seed A Byte 7 [reg99]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_a[57:56]	RW	8	8'h0	This register defines byte 7 of 10GBASE-R PCS test pattern seed A.

Table A.116. 10GBASE-R Test Pattern Seed B Byte 0 [reg9a]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[7:0]	RW	8	8'h0	This register defines byte 0 of 10GBASE-R PCS test pattern seed B.

Table A.117. 10GBASE-R Test Pattern Seed B Byte 1 [reg9b]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[15:8]	RW	8	8'h0	This register defines byte 1 of 10GBASE-R PCS test pattern seed B.

Table A.118. 10GBASE-R Test Pattern Seed B Byte 2 [reg9c]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[23:16]	RW	8	8'h0	This register defines byte 2 of 10GBASE-R PCS test pattern seed B.

Table A.119. 10GBASE-R Test Pattern Seed B Byte 3 [reg9d]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[31:24]	RW	8	8'h0	This register defines byte 3 of 10GBASE-R PCS test pattern seed B.

Table A.120. 10GBASE-R Test Pattern Seed B Byte 4 [reg9e]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[39:32]	RW	8	8'h0	This register defines byte 4 of 10GBASE-R PCS test pattern seed B.

Table A.121. 10GBASE-R Test Pattern Seed B Byte 5 [reg9f]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[47:40]	RW	8	8'h0	This register defines byte 5 of 10GBASE-R PCS test pattern seed B.

Table A.122. 10GBASE-R Test Pattern Seed B Byte 6 [rega0]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[55:48]	RW	8	8'h0	This register defines byte 6 of 10GBASE-R PCS test pattern seed B.

Table A.123. 10GBASE-R Test Pattern Seed B Byte 7 [rega1]

Field	Name	Access	Width	Reset	Description
[7:0]	prtp_seed_b[57:56]	RW	8	8'h0	This register defines byte 7 of 10GBASE-R PCS test pattern seed B.

Table A.124. 10GBASE-R Test Pattern Control 0 [rega2]

Field	Name	Access	Width	Reset	Description
[7:5]	tx_sw_pattern	RW	3	3'd0	Defines the square wave repeating pattern of n ones followed by n zeros, where n is configurable between 4 and 11. • 3'd0 – n=4 • 3'd1 – n=6 • 3'd2 – n=8 • 3'd3 – n=11 • others – reserved.
[4]	tx_prbs9_en	RW	1	1'b0	Tx PRBS9 Test-Pattern mode. Specifies the PRBS9 Test-Pattern mode is enabled or disabled. • 1'b1 – PRBS9 test-pattern mode on the Tx path is enabled. • 1'b0 – PRBS9 test-pattern mode on the Tx path is disabled.

Field	Name	Access	Width	Reset	Description
[3]	tx_prbs31_en	RW	1	1'b0	Tx PRBS31 Test-Pattern mode. Specifies the PRBS31 Test-Pattern mode is enabled or disabled. • 1'b1 – PRBS31 test-pattern mode on the Tx path is enabled. • 1'b0 – PRBS31 test-pattern mode on the Tx path is disabled.
[2]	tx_prtp_en	RW	1	1'b0	Tx test pattern. Specifies the Tx test pattern is enabled or disabled. • 1'b1 – transmit test pattern is enabled. • 1'b0 – transmit test pattern is disabled.
[1]	tx_prtp_test_sel	RW	1	1'b0	Tx test pattern Selector. Specifies the selected test pattern on the Tx path. • 1'b1 – square wave test pattern. • 1'b0 – pseudo random test pattern.
[0]	tx_prtp_data_sel	RW	1	1'b0	Tx Data pattern Selector. Specifies the selected data pattern on the Tx path. • 1'b1 – Zeros data pattern. • 1'b0 – LF data pattern.

Table A.125. 10GBASE-R Test Pattern Control 1 [rega3]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3]	rx_prbs31_en	RW	1	1'b0	Rx PRBS31 Test-Pattern mode. Specifies the PRBS31 Test-Pattern mode is enabled or disabled. • 1'b1 – PRBS31 test-pattern mode on the Rx path is enabled. • 1'b0 – PRBS31 test-pattern mode on the Rx path is disabled.
[2]	rx_prtp_en	RW	1	1'b0	Rx test pattern. Specifies the Rx test pattern is enabled or disabled. • 1'b1 – receive test pattern is enabled. • 1'b0 – receive test pattern is disabled.
[1]	rx_prtp_test_sel	RW	1	1'b0	Rx test pattern Selector. Specifies the selected test pattern on the Rx path. • 1'b1 – square wave test pattern. • 1'b0 – pseudo random test pattern.
[0]	rx_prtp_data_sel	RW	1	1'b0	Rx Data pattern Selector. Specifies the selected data pattern on the Rx path. • 1'b1 – Zeros data pattern. • 1'b0 – LF data pattern.

Table A.126. 10GBASE-R Test Pattern Error Counter Byte 0 [rega4]

Field	Name	Access	Width	Reset	Description
[7:0]	tp_error_cnt[7:0]	RO/CR	8	8'h0	10GBASE-R Test Pattern Error Counter Byte 0 register reflects the counter containing the number of errors received during a pattern test. These bits are reset to all zeros when the test-pattern error counter is read or upon execution of the PCS reset. These bits are held at all ones in case of overflow. This counter counts either block errors or bit errors depending on the test mode. **Note:** When accessing this counter, must read the low address byte (this register) first, then read the high address byte (next register).

Table A.127. 10GBASE-R Test Pattern Error Counter Byte 1 [rega5]

Field	Name	Access	Width	Reset	Description
[7:0]	tp_error_cnt[15:8]	RO/RC	8	8'h0	10GBASE-R Test Pattern Error Counter Byte 1 register reflects the counter containing the number of errors received during a pattern test. These bits are reset to all zeros when the test-pattern error counter is read or upon execution of the PCS reset. These bits are held at all ones in case of overflow. This counter counts either block errors or bit errors depending on the test mode.

Table A.128. PMA Control [regc6]

Field	Name	Access	Width	Reset	Description
[7:3]	reserved	RSVD	5	—	—
[3]	pma_rxval	RO	1	1'b0	The value of pma_rxval signal.
[2]	pma_ready	RO	1	1'b0	The value of pma_ready signal.
[1]	pma_phyrdy	RO	1	1'b0	The value of pma_phyrdy signal.

Table A.129. PMA Control [regc7]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	—	—
[3]	pma_rstn_ovrd	RW	1	1'b0	• 1'b1 – use bit[2] of this register to override the PMA control signal "epcs_rstn". • 1'b0 – use "pma_rstn" to drive "epcs_rstn".
[2]	pma_rstn	RW	1	1'b0	If bit[3] of this register is set to "1", use this bit to override the PMA control signal "epcs_rstn". • 1'b1 – drive "epcs_rstn" high if "pma_rstn_ovrd" is set to "1". • 1'b0 – drive "epcs_rstn" low if "pma_rstn_ovrd" is set to "1".
[1]	txval_ovrd	RW	1	1'b0	• 1'b1 – use bit[0] of this register to override the PMA control signal "epcs_txval". • 1'b0 – use "pma_txval" to drive "epcs_txval".
[0]	txval	RW	1	1'b0	If bit[1] of this register is set to "1", use this bit to override the PMA control signal "epcs_txval". • 1'b1 – drive "epcs_txval" high if "txval_ovrd" is set to "1". • 1'b0 – drive "epcs_txval" low if "txval_ovrd" is set to "1".

Table A.130. Loopback Mode Control [rege0]

Field	Name	Access	Width	Reset	Description
[7:3]	reserved	RSVD	5	—	—
[2]	fbrc_lpbk_en	RW	1	1'b0	• 1'b1 – enable 64B/66B PCS loopback B. • 1'b0 – do not enable 64B/66B PCS loopback B.
[1]	far_lpbk_en	RW	1	1'b0	In 8B/10B PCS mode: • 1'b1 – enable far end parallel loopback. • 1'b0 – do not enable far end parallel loopback. In 64B/66B PCS mode: • 1'b1 – enable loopback C. • 1'b0 – do not enable loopback C.
[0]	near_lpbk_en	RW	1	1'b0	In 8B/10B PCS mode: • 1'b1 – enable near end parallel loopback. • 1'b0 – do not enable parallel loopback. In 64B66B PCS mode: • 1'b1 – enable loopback A. • 1'b0 – do not enable loopback A.

Table A.131. MPCS BIST Control 0 [rege1]

Field	Name	Access	Width	Reset	Description
[7:5]	bist_ptn_sel	RW	3	3'h0	BIST Pattern Selection. Specifies the selected BIST Pattern. • 3'b111 – Alternate Constant Values 1 and 2 • 3'b110 – Constant Value 2 • 3'b101 – Constant Value 1 • 3'b100 – PRBS15 • 3'b011 – PRBS31 • 3'b010 – PRBS23 • 3'b001 – PRBS11 • 3'b000 – PRBS7
[4]	bist_rx_data_sel	RW	1	1'b0	Data receive selector. Specifies the selector where the data is to be received. • 1'b1 – data from after 8B/10B decoder. • 1'b0 – data from PMA.
[3]	bist_bypass_tx_gate	RW	1	1'b0	Bypass Transmit gate. • 1'b1 – force to send BIST data no matter the head is found or not. • 1'b0 – start to send BIST data after finding the head.
[2]	bist_bus8bit_sel	RW	1	1'b0	BIST bus width selector. • 1'b1 – 8-bit BIST data. • 1'b0 – 10-bit BIST data.
[1]	bist_mode	RW	1	1'b0	BIST mode. Specifies the BIST mode to be used. • 1'b1 – Continuous BIST mode. • 1'b0 – Timed BIST mode.
[0]	bist_en	RW	1	1'b0	Enable MPCS BIST. Specifies the MPCS BIST is enabled or disabled. • 1'b1 – MPCS BIST is enabled. • 1'b0 – MPCS BIST is disabled.

Table A.132. MPCS BIST Control 1 [rege2]

Field	Name	Access	Width	Reset	Description
[7:6]	bist_ch_sel	RW	2	2'b0	BIST Channel Selector. Specifies the selected BIST Channel. • 2'b11 – MPCS channel_3. • 2'b10 – MPCS channel_2. • 2'b01 – MPCS channel_1. • 2'b00 – MPCS channel_0.
[5:4]	bist_res_sel	RW	2	2'b0	BIST Resolution selector. Specifies the selected BIST resolution. • 2'b11 – Less than 128 errors. • 2'b10 – Less than 16 errors. • 2'b01 – Less than 2 errors. • 2'b00 – No error.
[3:2]	bist_time_sel	RW	2	2'b0	BIST Time selector. Specifies the selected BIST Time. • 2'b11 – 100k cycles. • 2'b10 – 5e+6 cycles. • 2'b01 – 5e+9 cycles. • 2'b00 – 5e+8 cycles.
[1:0]	bist_sync_head_reg	RW	2	2'b0	BIST Sync Header Selector. Specifies the selected BIST sync header counter selection. • 2'b11 – 24 • 2'b10 – 14 • 2'b01 – 8 • 2'b00 – 5

Table A.133. User Defined BIST Constant 1 Byte_0 [rege3]

Field	Name	Access	Width	Reset	Description
[7:0]	udbc[17:10]	RW	8	8'h0	User-defined BIST Constant 1 Byte_0 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.

Table A.134. User Defined BIST Constant 1 Byte_1 [rege4]

Field	Name	Access	Width	Reset	Description
[7:0]	udbc[7:0]	RW	8	8'h0	User-defined BIST Constant 1 Byte_1 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.

Table A.135. User Defined BIST Constant 1 MSByte [rege5]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	4'h0	—
[3:2]	udbc[19:18]	RW	2	2'h0	User-defined BIST Constant 1 Byte_0 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.
[1:0]	udbc[9: 8]	RW	2	2'h0	User-defined BIST Constant 1 Byte_0 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.

Table A.136. User Defined BIST Constant 2 Byte_0 [rege6]

Field	Name	Access	Width	Reset	Description
[7:0]	udbc2[17:10]	RW	8	8'h0	User-defined BIST Constant 1 Byte_0 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.

Table A.137. User Defined BIST Constant 2 Byte_1 [rege7]

Field	Name	Access	Width	Reset	Description
[7:0]	udbc2[7:0]	RW	8	8'h0	User-defined BIST Constant 1 Byte_1 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.

Table A.138. User Defined BIST Constant 2 MSB [rege8]

Field	Name	Access	Width	Reset	Description
[7:4]	reserved	RSVD	4	4'h0	—
[3:2]	udbc[19:18]	RW	2	2'h0	User-defined BIST Constant 1 Byte_0 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.
[1:0]	udbc[9: 8]	RW	2	2'h0	User-defined BIST Constant 1 Byte_0 register reflects the lower 8 bits of the 10b User-defined BIST Constant value pattern 1.

Table A.139. BIST Status 0 [rege9]

Field	Name	Access	Width	Reset	Description
[7:3]	bist_err_cnt[11:8]	RO	5	5'h0	BIST Error Count. Specifies the BIST error count.
[2]	bist_ok	RO	1	1'h0	BIST mode. Specifies the BIST mode to be used. • 1'b1 – No error for bist_res_sel = 0; Less than 2 errors for bist_res_sel = 1; Less than 16 errors for bist_rel_sel = 2 and less than 128 errors for bist_res_rel = 3 • 1'b0 – BIST is timed out or not started.
[1]	bist_done	RO	1	1'h0	BIST Done. Specifies the status of BIST timer. • 1'b1 – BIST timer expires. • 1'b0 – BIST is ongoing or not started.
[0]	bist_time_out	RO	1	1'h0	BIST Time Out. Specifies the BIST timeout status. • 1'b1 – BIST is timed out, but it is still in "sync header detection" stage. • 1'b0 – BIST is ongoing or not started.

Table A.140. BIST Status 1 [regea]

Field	Name	Access	Width	Reset	Description
[7:0]	bist_err_cnt[7:0]	RO	8	8'h0	BIST Status 1 register reflects the BIST error count.

Table A.141. PMA and PIPE PHY Status [regfe]

Field	Name	Access	Width	Reset	Description
[7]	reserved	RSVD	1	1'b0	—
[6]	pipe_phy_status	RW1C	1	1'b1	PIPE PHY status, write 1'b1 to clear. • 1'b1 – PCIe PHY is not ready for transmitting data and thus as long as the calibration has not completed. • 1'b0 – Calibration is done, and PCIe PHY is ready to start initializing the link.
[5]	rxclkstable	RO	1	1'b0	CDR PLL status. • 1'b1 – locked. • 1'b0 – loss of lock.
[4]	txclkstable	RO	1	1'b0	Tx PLL status. • 1'b1 – locked. • 1'b0 – loss of lock.

Field	Name	Access	Width	Reset	Description
[3]	cdrdiagout	RO	1	1'b0	CDR PLL internal frequency detector. • 1'b1 – the frequencies of TxClk and RxClk are not within operation bounds. • 1'b0 – the frequencies of TxClk and RxClk are within operation bounds.
[2]	trandet	RO	1	1'b0	Rx activity detection. • 1'b1 – Rx activity detected. • 1'b0 – Electrical idle or no valid signal.
[1]	cdrpllrstb	RO	1	1'b0	CDR PLL reset status. • 1'b1 – reset is released. • 1'b0 – CDR PLL is under reset.
[0]	txpllrstb	RO	1	1'b0	Tx PLL reset status. • 1'b1 – reset is released. • 1'b0 – Tx PLL is under reset.

Note:

1. This register is similar to the PMA reg7f, except the bit[6].

Appendix B. 8B/10B Symbol Coding

Table B.1 shows the 8B/10B encoding for data characters. Table B.2 shows the 8B/10B encoding for control characters.

Table B.1. 8B/10B Data Symbol Codes

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
D0.0	00	000 00000	100111 0100	011000 1011
D1.0	01	000 00001	011101 0100	100010 1011
D2.0	02	000 00010	101101 0100	010010 1011
D3.0	03	000 00011	110001 1011	110001 0100
D4.0	04	000 00100	110101 0100	001010 1011
D5.0	05	000 00101	101001 1011	101001 0100
D6.0	06	000 00110	011001 1011	011001 0100
D7.0	07	000 00111	111000 1011	000111 0100
D8.0	08	000 01000	111001 0100	000110 1011
D9.0	09	000 01001	100101 1011	100101 0100
D10.0	0A	000 01010	010101 1011	010101 0100
D11.0	0B	000 01011	110100 1011	110100 0100
D12.0	0C	000 01100	001101 1011	001101 0100
D13.0	0D	000 01101	101100 1011	101100 0100
D14.0	0E	000 01110	011100 1011	011100 0100
D15.0	0F	000 01111	010111 0100	101000 1011
D16.0	10	000 10000	011011 0100	100100 1011
D17.0	11	000 10001	100011 1011	100011 0100
D18.0	12	000 10010	010011 1011	010011 0100
D19.0	13	000 10011	110010 1011	110010 0100
D20.0	14	000 10100	001011 1011	001011 0100
D21.0	15	000 10101	101010 1011	101010 0100
D22.0	16	000 10110	011010 1011	011010 0100
D23.0	17	000 10111	111010 0100	000101 1011
D24.0	18	000 11000	110011 0100	001100 1011
D25.0	19	000 11001	100110 1011	100110 0100
D26.0	1A	000 11010	010110 1011	010110 0100
D27.0	1B	000 11011	110110 0100	001001 1011
D28.0	1C	000 11100	001110 1011	001110 0100
D29.0	1D	000 11101	101110 0100	010001 1011
D30.0	1E	000 11110	011110 0100	100001 1011
D31.0	1F	000 11111	101011 0100	010100 1011
D0.1	20	001 00000	100111 1001	011000 1001
D1.1	21	001 00001	011101 1001	100010 1001
D2.1	22	001 00010	101101 1001	010010 1001
D3.1	23	001 00011	110001 1001	110001 1001
D4.1	24	001 00100	110101 1001	001010 1001
D5.1	25	001 00101	101001 1001	101001 1001
D6.1	26	001 00110	011001 1001	011001 1001
D7.1	27	001 00111	111000 1001	000111 1001
D8.1	28	001 01000	111001 1001	000110 1001
D9.1	29	001 01001	100101 1001	100101 1001

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
D10.1	2A	001 01010	010101 1001	010101 1001
D11.1	2B	001 01011	110100 1001	110100 1001
D12.1	2C	001 01100	001101 1001	001101 1001
D13.1	2D	001 01101	101100 1001	101100 1001
D14.1	2E	001 01110	011100 1001	011100 1001
D15.1	2F	001 01111	010111 1001	101000 1001
D16.1	30	001 10000	011011 1001	100100 1001
D17.1	31	001 10001	100011 1001	100011 1001
D18.1	32	001 10010	010011 1001	010011 1001
D19.1	33	001 10011	110010 1001	110010 1001
D20.1	34	001 10100	001011 1001	001011 1001
D21.1	35	001 10101	101010 1001	101010 1001
D22.1	36	001 10110	011010 1001	011010 1001
D23.1	37	001 10111	111010 1001	000101 1001
D24.1	38	001 11000	110011 1001	001100 1001
D25.1	39	001 11001	100110 1001	100110 1001
D26.1	3A	001 11010	010110 1001	010110 1001
D27.1	3B	001 11011	110110 1001	001001 1001
D28.1	3C	001 11100	001110 1001	001110 1001
D29.1	3D	001 11101	101110 1001	010001 1001
D30.1	3E	001 11110	011110 1001	100001 1001
D31.1	3F	001 11111	101011 1001	010100 1001
D0.2	40	010 00000	100111 0101	011000 0101
D1.2	41	010 00001	011101 0101	100010 0101
D2.2	42	010 00010	101101 0101	010010 0101
D3.2	43	010 00011	110001 0101	110001 0101
D4.2	44	010 00100	110101 0101	001010 0101
D5.2	45	010 00101	101001 0101	101001 0101
D6.2	46	010 00110	011001 0101	011001 0101
D7.2	47	010 00111	111000 0101	000111 0101
D8.2	48	010 01000	111001 0101	000110 0101
D9.2	49	010 01001	100101 0101	100101 0101
D10.2	4A	010 01010	010101 0101	010101 0101
D11.2	4B	010 01011	110100 0101	110100 0101
D12.2	4C	010 01100	001101 0101	001101 0101
D13.2	4D	010 01101	101100 0101	101100 0101
D14.2	4E	010 01110	011100 0101	011100 0101
D15.2	4F	010 01111	010111 0101	101000 0101
D16.2	50	010 10000	011011 0101	100100 0101
D17.2	51	010 10001	100011 0101	100011 0101
D18.2	52	010 10010	010011 0101	010011 0101
D19.2	53	010 10011	110010 0101	110010 0101
D20.2	54	010 10100	001011 0101	001011 0101
D21.2	55	010 10101	101010 0101	101010 0101
D22.2	56	010 10110	011010 0101	011010 0101
D23.2	57	010 10111	111010 0101	000101 0101

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
D24.2	58	010 11000	110011 0101	001100 0101
D25.2	59	010 11001	100110 0101	100110 0101
D26.2	5A	010 11010	010110 0101	010110 0101
D27.2	5B	010 11011	110110 0101	001001 0101
D28.2	5C	010 11100	001110 0101	001110 0101
D29.2	5D	010 11101	101110 0101	010001 0101
D30.2	5E	010 11110	011110 0101	100001 0101
D31.2	5F	010 11111	101011 0101	010100 0101
D0.3	60	011 00000	100111 0011	011000 1100
D1.3	61	011 00001	011101 0011	100010 1100
D2.3	62	011 00010	101101 0011	010010 1100
D3.3	63	011 00011	110001 1100	110001 0011
D4.3	64	011 00100	110101 0011	001010 1100
D5.3	65	011 00101	101001 1100	101001 0011
D6.3	66	011 00110	011001 1100	011001 0011
D7.3	67	011 00111	111000 1100	000111 0011
D8.3	68	011 01000	111001 0011	000110 1100
D9.3	69	011 01001	100101 1100	100101 0011
D10.3	6A	011 01010	010101 1100	010101 0011
D11.3	6B	011 01011	110100 1100	110100 0011
D12.3	6C	011 01100	001101 1100	001101 0011
D13.3	6D	011 01101	101100 1100	101100 0011
D14.3	6E	011 01110	011100 1100	011100 0011
D15.3	6F	011 01111	010111 0011	101000 1100
D16.3	70	011 10000	011011 0011	100100 1100
D17.3	71	011 10001	100011 1100	100011 0011
D18.3	72	011 10010	010011 1100	010011 0011
D19.3	73	011 10011	110010 1100	110010 0011
D20.3	74	011 10100	001011 1100	001011 0011
D21.3	75	011 10101	101010 1100	101010 0011
D22.3	76	011 10110	011010 1100	011010 0011
D23.3	77	011 10111	111010 0011	000101 1100
D24.3	78	011 11000	110011 0011	001100 1100
D25.3	79	011 11001	100110 1100	100110 0011
D26.3	7A	011 11010	010110 1100	010110 0011
D27.3	7B	011 11011	110110 0011	001001 1100
D28.3	7C	011 11100	001110 1100	001110 0011
D29.3	7D	011 11101	101110 0011	010001 1100
D30.3	7E	011 11110	011110 0011	100001 1100
D31.3	7F	011 11111	101011 0011	010100 1100
D0.4	80	100 00000	100111 0010	011000 1101
D1.4	81	100 00001	011101 0010	100010 1101
D2.4	82	100 00010	101101 0010	010010 1101
D3.4	83	100 00011	110001 1101	110001 0010
D4.4	84	100 00100	110101 0010	001010 1101
D5.4	85	100 00101	101001 1101	101001 0010

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
D6.4	86	100 00110	011001 1101	011001 0010
D7.4	87	100 00111	111000 1101	000111 0010
D8.4	88	100 01000	111001 0010	000110 1101
D9.4	89	100 01001	100101 1101	100101 0010
D10.4	8A	100 01010	010101 1101	010101 0010
D11.4	8B	100 01011	110100 1101	110100 0010
D12.4	8C	100 01100	001101 1101	001101 0010
D13.4	8D	100 01101	101100 1101	101100 0010
D14.4	8E	100 01110	011100 1101	011100 0010
D15.4	8F	100 01111	010111 0010	101000 1101
D16.4	90	100 10000	011011 0010	100100 1101
D17.4	91	100 10001	100011 1101	100011 0010
D18.4	92	100 10010	010011 1101	010011 0010
D19.4	93	100 10011	110010 1101	110010 0010
D20.4	94	100 10100	001011 1101	001011 0010
D21.4	95	100 10101	101010 1101	101010 0010
D22.4	96	100 10110	011010 1101	011010 0010
D23.4	97	100 10111	111010 0010	000101 1101
D24.4	98	100 11000	110011 0010	001100 1101
D25.4	99	100 11001	100110 1101	100110 0010
D26.4	9A	100 11010	010110 1101	010110 0010
D27.4	9B	100 11011	110110 0010	001001 1101
D28.4	9C	100 11100	001110 1101	001110 0010
D29.4	9D	100 11101	101110 0010	010001 1101
D30.4	9E	100 11110	011110 0010	100001 1101
D31.4	9F	100 11111	101011 0010	010100 1101
D0.5	A0	101 00000	100111 1010	011000 1010
D1.5	A1	101 00001	011101 1010	100010 1010
D2.5	A2	101 00010	101101 1010	010010 1010
D3.5	A3	101 00011	110001 1010	110001 1010
D4.5	A4	101 00100	110101 1010	001010 1010
D5.5	A5	101 00101	101001 1010	101001 1010
D6.5	A6	101 00110	011001 1010	011001 1010
D7.5	A7	101 00111	111000 1010	000111 1010
D8.5	A8	101 01000	111001 1010	000110 1010
D9.5	A9	101 01001	100101 1010	100101 1010
D10.5	AA	101 01010	010101 1010	010101 1010
D11.5	AB	101 01011	110100 1010	110100 1010
D12.5	AC	101 01100	001101 1010	001101 1010
D13.5	AD	101 01101	101100 1010	101100 1010
D14.5	AE	101 01110	011100 1010	011100 1010
D15.5	AF	101 01111	010111 1010	101000 1010
D16.5	B0	101 10000	011011 1010	100100 1010
D17.5	B1	101 10001	100011 1010	100011 1010
D18.5	B2	101 10010	010011 1010	010011 1010
D19.5	B3	101 10011	110010 1010	110010 1010

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
D20.5	B4	101 10100	001011 1010	001011 1010
D21.5	B5	101 10101	101010 1010	101010 1010
D22.5	B6	101 10110	011010 1010	011010 1010
D23.5	B7	101 10111	111010 1010	000101 1010
D24.5	B8	101 11000	110011 1010	001100 1010
D25.5	B9	101 11001	100110 1010	100110 1010
D26.5	BA	101 11010	010110 1010	010110 1010
D27.5	BB	101 11011	110110 1010	001001 1010
D28.5	BC	101 11100	001110 1010	001110 1010
D29.5	BD	101 11101	101110 1010	010001 1010
D30.5	BE	101 11110	011110 1010	100001 1010
D31.5	BF	101 11111	101011 1010	010100 1010
D0.6	C0	110 00000	100111 0110	011000 0110
D1.6	C1	110 00001	011101 0110	100010 0110
D2.6	C2	110 00010	101101 0110	010010 0110
D3.6	C3	110 00011	110001 0110	110001 0110
D4.6	C4	110 00100	110101 0110	001010 0110
D5.6	C5	110 00101	101001 0110	101001 0110
D6.6	C6	110 00110	011001 0110	011001 0110
D7.6	C7	110 00111	111000 0110	000111 0110
D8.6	C8	110 01000	111001 0110	000110 0110
D9.6	C9	110 01001	100101 0110	100101 0110
D10.6	CA	110 01010	010101 0110	010101 0110
D11.6	CB	110 01011	110100 0110	110100 0110
D12.6	CC	110 01100	001101 0110	001101 0110
D13.6	CD	110 01101	101100 0110	101100 0110
D14.6	CE	110 01110	011100 0110	011100 0110
D15.6	CF	110 01111	010111 0110	101000 0110
D16.6	D0	110 10000	011011 0110	100100 0110
D17.6	D1	110 10001	100011 0110	100011 0110
D18.6	D2	110 10010	010011 0110	010011 0110
D19.6	D3	110 10011	110010 0110	110010 0110
D20.6	D4	110 10100	001011 0110	001011 0110
D21.6	D5	110 10101	101010 0110	101010 0110
D22.6	D6	110 10110	011010 0110	011010 0110
D23.6	D7	110 10111	111010 0110	000101 0110
D24.6	D8	110 11000	110011 0110	001100 0110
D25.6	D9	110 11001	100110 0110	100110 0110
D26.6	DA	110 11010	010110 0110	010110 0110
D27.6	DB	110 11011	110110 0110	001001 0110
D28.6	DC	110 11100	001110 0110	001110 0110
D29.6	DD	110 11101	101110 0110	010001 0110
D30.6	DE	110 11110	011110 0110	100001 0110
D31.6	DF	110 11111	101011 0110	010100 0110
D0.7	E0	111 00000	100111 0001	011000 1110
D1.7	E1	111 00001	011101 0001	100010 1110

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
D2.7	E2	111 00010	101101 0001	010010 1110
D3.7	E3	111 00011	110001 1110	110001 0001
D4.7	E4	111 00100	110101 0001	001010 1110
D5.7	E5	111 00101	101001 1110	101001 0001
D6.7	E6	111 00110	011001 1110	011001 0001
D7.7	E7	111 00111	111000 1110	000111 0001
D8.7	E8	111 01000	111001 0001	000110 1110
D9.7	E9	111 01001	100101 1110	100101 0001
D10.7	EA	111 01010	010101 1110	010101 0001
D11.7	EB	111 01011	110100 1110	110100 1000
D12.7	EC	111 01100	001101 1110	001101 0001
D13.7	ED	111 01101	101100 1110	101100 1000
D14.7	EE	111 01110	011100 1110	011100 1000
D15.7	EF	111 01111	010111 0001	101000 1110
D16.7	F0	111 10000	011011 0001	100100 1110
D17.7	F1	111 10001	100011 0111	100011 0001
D18.7	F2	111 10010	010011 0111	010011 0001
D19.7	F3	111 10011	110010 1110	110010 0001
D20.7	F4	111 10100	001011 0111	001011 0001
D21.7	F5	111 10101	101010 1110	101010 0001
D22.7	F6	111 10110	011010 1110	011010 0001
D23.7	F7	111 10111	111010 0001	000101 1110
D24.7	F8	111 11000	110011 0001	001100 1110
D25.7	F9	111 11001	100110 1110	100110 0001
D26.7	FA	111 11010	010110 1110	010110 0001
D27.7	FB	111 11011	110110 0001	001001 1110
D28.7	FC	111 11100	001110 1110	001110 0001
D29.7	FD	111 11101	101110 0001	010001 1110
D30.7	FE	111 11110	011110 0001	100001 1110
D31.7	FF	111 11111	101011 0001	010100 1110

Table B.2. 8B/10B Control Symbol Codes

Data Byte Name	Data Byte Value (Hex)	Data Byte Value (Bin)	Coded Data Value (RD-)	Coded Data Value (RD+)
K28.0	1C	000 11100	001111 0100	110000 1011
K28.1[1]	3C	001 11100	001111 1001	110000 0110
K28.2	5C	010 11100	001111 0101	110000 1010
K28.3	7C	011 11100	001111 0011	110000 1100
K28.4	9C	100 11100	001111 0010	110000 1101
K28.5[1]	BC	101 11100	001111 1010	110000 0101
K28.6	DC	110 11100	001111 0110	110000 1001
K28.7[1]	FC	111 11100	001111 1000	110000 0111
K23.7	F7	111 10111	111010 1000	000101 0111
K27.7	FB	111 11011	110110 1000	001001 0111
K29.7	FD	111 11101	101110 1000	010001 0111
K30.7	FE	111 11110	011110 1000	100001 0111

Note:

1. Contains a COMMA.

Appendix C. Calculating Parameters for SerDes PLL

Table C.1 lists the recommended parameters for SerDes PLL in specific data rate. For detailed information about PLL clock setting, refer to PLL Clock Setting section. N needs be set as 5, 10 or 20, when 8B/10B PCS is required.

Table C.1. Recommended Parameters for SerDes PLL

Bit Rate (Gbps)	Reference Clock (MHz)	M	F	N	F_{VCO} (MHz)	F_{bit} (MHz)	F_{PMA} (MHz)	Note
0.27	135	8	1	8	8640	1080	135	Actual data rate is 1.08 Gbps
0.6144	76.8	8	1	8	4915.2	614.4	76.8	PMA Only
	122.88	8	1	5	4915.2	614.4	122.88	Risk for 4915.2 MHz F_{VCO}
1.001	100.1	8	1	10	8008	1001	100.1	—
	125.125[1]	8	1	8	8008	1001	125.125	PMA Only
1.152	76.8	8	3	5	9216	1152	230.4	—
	115.2	8	1	10	9216	1152	115.2	—
	144	8	1	8	9216	1152	144	PMA Only
1.188	74.25	8	1	16	9504	1188	74.25	PMA Only
	74.25	8	2	8	9504	1188	148.5	PMA Only
	79.2	8	3	5	9504	1188	237.6	—
	118.8	8	1	10	9504	1188	118.8	—
	148.5	8	1	8	9504	1188	148.5	PMA Only
1.2288	76.8	8	1	16	9830.4	1228.8	76.8	PMA Only
	76.8	8	2	8	9830.4	1228.8	153.6	PMA Only
	81.92	8	3	5	9830.4	1228.8	245.76	—
	122.88	8	1	10	9830.4	1228.8	122.88	—
	153.6	8	1	8	9830.4	1228.8	153.6	PMA Only
1.25	78.125	8	1	16	10000	1250	78.125	PMA Only
	78.125	8	2	8	10000	1250	156.25	PMA Only
	83.333	8	3	5	10000	1250	250	—
	125	8	1	10	10000	1250	125	—
	156.25	8	1	8	10000	1250	156.25	PMA Only
1.485	74.25	4	2	10	5940	1485	148.5	—
	74.25	4	4	5	5940	1485	297	—
	92.8125[1]	4	1	16	5940	1485	92.8125	PMA Only
	92.8125[1]	4	2	8	5940	1485	185.625	PMA Only
	99	4	3	5	5940	1485	297	—
	148.5	4	1	10	5940	1485	148.5	—
	148.5	4	2	5	5940	1485	297	—
1.5	75	4	4	5	6000	1500	300	—
	75	4	2	10	6000	1500	150	—
	93.75	4	1	16	6000	1500	93.75	PMA Only
	93.75	4	2	8	6000	1500	187.5	PMA Only
	100	4	3	5	6000	1500	300	—
	150	4	2	5	6000	1500	300	—
	150	4	1	10	6000	1500	150	—

Bit Rate (Gbps)	Reference Clock (MHz)	M	F	N	F_{VCO} (MHz)	F_{bit} (MHz)	F_{PMA} (MHz)	Note
1.62	81	4	1	20	6480	1620	81	—
	81	4	2	10	6480	1620	162	—
	101.25[1]	4	1	16	6480	1620	101.25	PMA Only
	101.25[1]	4	2	8	6480	1620	202.5	PMA Only
	108	4	3	5	6480	1620	324	—
	162	4	1	10	6480	1620	162	—
	162	4	2	5	6480	1620	324	—
2	80	4	5	5	8000	2000	400	—
	83.333	4	3	8	8000	2000	250	PMA Only
	100	4	1	20	8000	2000	100	—
	100	4	2	10	8000	2000	200	—
	125	4	2	8	8000	2000	250	PMA Only
	125	4	1	16	8000	2000	125	PMA Only
	133.333	4	3	5	8000	2000	400	—
2.304	76.8	4	3	10	9216	2304	230.4	—
	76.8	4	6	5	9216	2304	460.8	—
	92.16	4	5	5	9216	2304	460.8	—
	96	4	3	8	9216	2304	288	PMA Only
	115.2	4	2	10	9216	2304	230.4	—
	115.2	4	1	20	9216	2304	115.2	—
	144	4	1	16	9216	2304	144	PMA Only
	144	4	2	8	9216	2304	288	PMA Only
	153.6	4	3	5	9216	2304	460.8	—
2.376	74.25	4	2	16	9504	2376	148.5	PMA Only
	74.25	4	4	8	9504	2376	297	PMA Only
	79.2	4	3	10	9504	2376	237.6	—
	79.2	4	6	5	9504	2376	475.2	—
	95.04[1]	4	5	5	9504	2376	475.2	—
	99	4	3	8	9504	2376	297	PMA Only
	118.8	4	1	20	9504	2376	118.8	—
	118.8	4	2	10	9504	2376	237.6	—
	148.5	4	1	16	9504	2376	148.5	PMA Only
	148.5	4	2	8	9504	2376	297	PMA Only
	158.4[1]	4	3	5	9504	2376	475.2	—
2.4576	76.8	4	2	16	9830.4	2457.6	153.6	PMA Only
	76.8	4	4	8	9830.4	2457.6	307.2	PMA Only
	81.92	4	3	10	9830.4	2457.6	245.76	—
	81.92	4	6	5	9830.4	2457.6	491.52	—
	98.304	4	5	5	9830.4	2457.6	491.52	—
	102.4	4	3	8	9830.4	2457.6	307.2	PMA Only
	122.88	4	1	20	9830.4	2457.6	122.88	—
	122.88	4	2	10	9830.4	2457.6	245.76	—
	153.6	4	1	16	9830.4	2457.6	153.6	PMA Only
	153.6	4	2	8	9830.4	2457.6	307.2	PMA Only

Bit Rate (Gbps)	Reference Clock (MHz)	M	F	N	F_{VCO} (MHz)	F_{bit} (MHz)	F_{PMA} (MHz)	Note
2.5	78.125	4	2	16	10000	2500	156.25	PMA Only
	78.125	4	4	8	10000	2500	312.5	PMA Only
	83.333	4	6	5	10000	2500	500	—
	83.333	4	3	10	10000	2500	250	—
	100	4	5	5	10000	2500	500	—
	104.166	4	3	8	10000	2500	312.5	PMA Only
	125	4	4	5	10000	2500	500	—
	125	4	2	10	10000	2500	250	—
	125	4	1	20	10000	2500	125	—
	156.25	4	1	16	10000	2500	156.25	PMA Only
	156.25	4	2	8	10000	2500	312.5	PMA Only
2.7	84.375[1]	4	2	16	5400	2700	168.75	PMA Only
	84.375[1]	4	4	8	5400	2700	337.5	PMA Only
	90	4	3	10	5400	2700	270	—
	108	2	5	5	5400	2700	540	Risk for 540 MHz F_{PMA}
	112.5	2	3	8	5400	2700	337.5	PMA Only
	135	2	1	20	5400	2700	135	—
	135	2	2	10	5400	2700	270	—
2.97	74.25	2	2	20	5940	2970	148.5	—
	74.25	2	4	10	5940	2970	297	—
	92.8125[1]	2	2	16	5940	2970	185.625	PMA Only
	92.8125[1]	2	4	8	5940	2970	371.25	PMA Only
	99	2	3	10	5940	2970	297	—
	123.75[1]	2	3	8	5940	2970	371.25	PMA Only
	148.5	2	1	20	5940	2970	148.5	—
	148.5	2	2	10	5940	2970	297	—
3	75	2	4	10	6000	3000	300	—
	75	2	2	20	6000	3000	150	—
	93.75	2	2	16	600	300	187.5	PMA Only
	93.75	2	4	8	600	300	375	PMA Only
	100	2	3	10	6000	3000	300	—
	125	2	3	8	6000	3000	375	PMA Only
	150	2	2	10	6000	3000	300	—
	150	2	1	20	6000	3000	150	—
3.072	76.8	2	4	10	6144	3072	230.4	—
	76.8	2	2	20	6144	3072	153.6	—
	96	2	2	16	6144	3072	192	PMA Only
	96	2	4	8	6144	3072	384	PMA Only
	102.4	2	3	10	6144	3072	307.2	—
	128	2	3	8	6144	3072	384	PMA Only
	153.6	2	1	20	6144	3072	153.6	—
	153.6	2	2	10	6144	3072	307.2	—

Bit Rate (Gbps)	Reference Clock (MHz)	M	F	N	F_{VCO} (MHz)	F_{bit} (MHz)	F_{PMA} (MHz)	Note
3.125	78.125	2	4	10	6250	3125	312.5	—
	78.125	2	2	20	6250	3125	156.25	—
	97.65625	2	2	16	6250	3125	195.3125	PMA Only
	97.65625	2	4	8	6250	3125	390.625	PMA Only
	104.166	2	3	10	6250	3125	312.5	—
	130.2083[1]	2	3	8	6250	3125	390.625	PMA Only
	156.25	2	1	20	6250	3125	156.25	—
	156.25	2	2	10	6250	3125	312.5	—
3.75	75	2	5	10	7500	3750	375	—
	78.125	2	3	16	7500	3750	234.375	PMA Only
	78.125	2	6	8	7500	3750	468.75	PMA Only
	93.75	2	4	10	7500	3750	375	—
	93.75	2	2	20	7500	3750	187.5	—
	117.1875[1]	2	2	16	7500	3750	234.375	PMA Only
	117.1875[1]	2	4	8	7500	3750	468.75	PMA Only
	125	2	3	10	7500	3750	375	—
	156.25	2	3	8	7500	3750	468.75	PMA Only
4	80	2	5	10	8000	4000	400	—
	83.333	2	3	16	8000	4000	250	PMA Only
	83.333	2	6	8	8000	4000	500	PMA Only
	100	2	4	10	8000	4000	400	—
	100	2	2	20	8000	4000	200	—
	125	2	2	16	9216	4608	250	PMA Only
	125	2	4	8	9216	4608	500	PMA Only
	133.333	2	3	10	8000	4000	400	—
4.608	76.8	2	3	20	9216	4608	230.4	—
	76.8	2	6	10	9216	4608	460.8	—
	92.16	2	5	10	9216	4608	460.8	—
	96	2	3	16	9216	4608	288	PMA Only
	115.2	2	4	10	9216	4608	460.8	—
	115.2	2	2	20	9216	4608	230.4	—
	144	2	2	16	9216	4608	288	PMA Only
	153.6	2	3	10	9216	4608	460.8	—
4.752	79.2	2	3	20	9504	4752	237.6	—
	79.2	2	6	10	9504	4752	475.2	—
	95.04[1]	2	5	10	9504	4752	475.2	—
	99	2	3	16	9504	4752	297	PMA Only
	118.8	2	4	10	9504	4752	475.2	—
	118.8	2	2	20	9504	4752	237.6	—
	148.5	2	2	16	9504	4752	297	PMA Only
	158.4[1]	2	3	10	9504	4752	475.2	—

Bit Rate (Gbps)	Reference Clock (MHz)	M	F	N	F_{VCO} (MHz)	F_{bit} (MHz)	F_{PMA} (MHz)	Note
4.9152	81.92	2	3	20	9830.4	4915.2	245.76	—
	98.304[1]	2	5	10	9830.4	4915.2	491.52	—
	102.4	2	3	16	9830.4	4915.2	307.2	PMA Only
	122.88	2	4	10	9830.4	4915.2	491.52	—
	122.88	2	2	20	9830.4	4915.2	245.76	—
	153.6	2	2	16	9830.4	4915.2	307.2	PMA Only
5	78.125	2	4	16	10000	5000	312.5	PMA Only
	83.333	2	6	10	10000	5000	500	—
	83.333	2	3	20	10000	5000	250	—
	100	2	5	10	10000	5000	500	—
	104.166	2	3	16	10000	5000	312.5	PMA Only
	125	2	2	20	10000	5000	250	—
	125	2	4	10	10000	5000	500	—
	156.25	2	2	16	10000	5000	312.5	PMA Only
5.4	84.375[1]	1	4	16	5400	5400	337.5	PMA Only
	90	1	3	20	5400	5400	270	—
	108	1	5	10	5400	5400	540	Risk for 540 MHz F_{PMA}
	112.5	1	3	16	5400	5400	337.5	PMA Only
	135	1	2	20	5400	5400	270	—
5.94	74.25	1	4	20	5940	5940	297	—
	74.25	1	5	16	5940	5940	371.25	PMA Only
	92.8125[1]	1	4	16	5940	5940	371.25	PMA Only
	99	1	3	20	5940	5940	297	—
	123.75[1]	1	3	16	5940	5940	371.25	PMA Only
	148.5	1	2	20	5940	5940	297	—
6	75	1	4	20	6000	6000	300	—
	75	1	5	16	6000	6000	375	PMA Only
	93.75	1	4	16	6000	6000	375	PMA Only
	100	1	3	20	6000	6000	300	—
	125	1	3	16	6000	6000	375	PMA Only
	150	1	2	20	6000	6000	300	—
6.144	76.8	1	4	20	6144	6144	307.2	—
	76.8	1	5	16	6144	6144	384	PMA Only
	96	1	4	16	6144	6144	384	PMA Only
	102.4	1	3	20	6144	6144	307.2	—
	128	1	3	16	6144	6144	384	PMA Only
	153.6	1	2	20	6144	6144	307.2	—
6.25	78.125	1	4	20	6250	6250	312.5	—
	78.125	1	5	16	6250	6250	390.625	PMA Only
	97.65625[1]	1	4	16	6250	6250	390.625	PMA Only
	104.166	1	3	20	6250	6250	312.5	—
	130.2083[1]	1	3	16	6250	6250	390.625	PMA Only
	156.25	1	2	20	6250	6250	312.5	—

Bit Rate (Gbps)	Reference Clock (MHz)	M	F	N	F_{VCO} (MHz)	F_{bit} (MHz)	F_{PMA} (MHz)	Note
6.375	79.6875	1	4	20	6375	6375	318.75	—
	79.6875	1	5	16	6375	6375	398.4375	PMA Only
	99.609375[1]	1	4	16	6375	6375	398.4375	PMA Only
	106.25	1	3	20	6375	6375	318.75	—
	132.8125	1	3	16	6375	6375	398.4375	PMA Only
	159.375	1	2	20	6375	6375	318.75	—
6.75	84.375[1]	1	4	20	6750	6750	337.5	—
	84.375[1]	1	5	16	6750	6750	421.875	PMA Only
	105.46875[1]	1	4	16	6750	6750	421.875	PMA Only
	112.5	1	3	20	6750	6750	337.5	—
	140.625	1	3	16	6750	6750	421.875	PMA Only
8	80	1	5	20	8000	8000	400	—
	83.333	1	6	16	8000	8000	500	PCIe Gen3 or PMA Only
	100	1	4	20	8000	8000	400	—
	100	1	5	16	8000	8000	500	PCIe Gen3 or PMA Only
	125	1	4	16	8000	8000	500	PCIe Gen3 or PMA Only
	133.333	1	3	20	8000	8000	400	—
8.1	81	1	5	20	8100	8100	405	—
	101.25[1]	1	4	20	8100	8100	405	—
	135	1	3	20	8100	8100	405	—
10.3125	161.1328125	1	4	16	10312.5	10312.5	644.53	For 64B/66B PCS

Note:

1. This crystal may not be available.

Table C.2 shows some potential serial protocol applications based on CertusPro-NX SerDes/PCS.

Table C.2. Serial Protocol Applications

Protocol	Bit Rate (Gbps)	Reference Clock Available	Coding	SerDes/PCS Mode
PCIe	2.5	83.333, 100, 125	8B/10B	PCIe PCS
	5	83.333, 100, 125	8B/10B	PCIe PCS
	8	83.333, 100, 125	128B/130B	PCIe PCS
DP	1.62	81, 108, 162	8B/10B	8B/10B PCS
	2.7	90, 108, 135	8B/10B	8B/10B PCS
	5.4	90, 108, 135	8B/10B	8B/10B PCS
	8.1	81, 135	8B/10B	8B/10B PCS
1000BASE-X / GigE	1.25	83.333, 125	8B/10B	8B/10B PCS
SGMII	1.25	83.333, 125	8B/10B	8B/10B PCS
XAUI	3.125	78.125, 104.166, 156.25	8B/10B	8B/10B PCS
QSGMII	5	83.333, 100, 125	8B/10B	8B/10B PCS
10GBASE-R	10.3125	161.1328125	64B/66B	64B/66B PCS
CoaXPress	1.25	83.333, 125	8B/10B	8B/10B PCS
	2.5	83.333, 100, 125	8B/10B	8B/10B PCS
	3.125	78.125, 104.166, 156.25	8B/10B	8B/10B PCS
	5	83.333, 100, 125	8B/10B	8B/10B PCS
	6.25	78.125, 104.166, 156.25	8B/10B	8B/10B PCS

Protocol	Bit Rate (Gbps)	Reference Clock Available	Coding	SerDes/PCS Mode
SLVS-EC	1.152	76.8, 115.2	8B/10B	8B/10B PCS
	1.188	79.2, 118.8	8B/10B	8B/10B PCS
	2.304	76.8, 92.16, 115.2, 153.6	8B/10B	8B/10B PCS
	2.376	79.2, 95.04[1],118.8, 158.4[1]	8B/10B	8B/10B PCS
	4.608	76.8, 92.16, 115.2, 153.6	8B/10B	8B/10B PCS
	4.752	79.2, 95.04[1],118.8, 158.4[1]	8B/10B	8B/10B PCS

Note:

1. This crystal may not be available.

References

- CertusPro-NX Family Data Sheet (FPGA-DS-02086)
- Electrical Recommendations for Lattice SERDES (FPGA-TN-02077)
- ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012)
- High-Speed PCB Design Considerations (FPGA-TN-02178)
- LatticeECP3 Family Handbook (HB1009)
- LatticeECP3 Family Data Sheet (FPGA-DS-02074)
- LatticeECP3 SERDES/PCS Usage Guide (FPGA-TN-02190)
- MPCS Module – Lattice Radiant Software User Guide (FPGA-IPUG-02118)
- PCIe X4 IP Core – Lattice Radiant Software (FPGA-IPUG-02126)
- CertusPro-NX FPGA web page
- Clarity Designer Manual
- ECO Editor Manual
- High Speed SerDes Devices and Applications
- PCI Express Base Specification Rev5.0 v1.0
- PCI Express Technology Comprehensive Guide to Generations 1.x, 2.x, and 3.0

Technical Support Assistance

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

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.2, July 2023

Section	Change Summary
Appendix A. Configuration Registers	Replaced register name from “CDR reference" to "Reserved" and deleted description *Defines the CDR PLL reference clock mode. 2’b00 – normal operation* in Table A.2. Control Register 0 [reg00].
References	Added CertusPro-NX web page link in References section.

Revision 1.1, February 2023

Section	Change Summary
All	Changed document title to *CertusPro-NX SerDes/PCS User Guide* and adjusted document type.
Architecture Overview	• Removed the PIPE Interface section and Figure 5.9 PIPE mode. • Updated the PCI Express Architecture section to remove the sentence *The user can configure this IP as PIPE mode, which is compliant to industry PIPE standard version 4.4, by bypassing Link Layer block*. • Updated Table 5.3. Block Usage for the Corresponding SerDes/PCS Mode to remove the column on PIPE.
Protocol Mode	Updated the PCI Express Mode section to remove the sentence *All CertusPro-NX SerDes/PCS quads can be configured as PCI Express PIPE mode to work with PCIe Link Layer soft IP*.
Reference Clock Source Selection	• Updated the Reference Clock Source Selection section to remove the sentence *In PCI Express PIPE mode, the reference clock source from per-quad package pins (SDQx_REFCLKP/N) is also recommended when the number of lane is not larger than four*. • Updated Table 13.3. Electrical Idle Related Signals to remove the PCIe PIPE information.
SerDes/PCS Generation in Radiant Software	Updated Table 14.2. Pin-to-Pin Connection to remove the column on PIPE.

Appendix A. Configuration Registers	• Updated Table A.1. Register Address:
	  • Changed reg68 to reg74 for PCS loopback control
	  • Changed reg69 to reg75 for Transmit PLL Current Charge Pump
	  • Changed reg6a to reg76 for Receive PLL Current Charge Pump
	• Updated the table title to Table A.24. Transmit PLL Current Charge Pump [reg75].
	• Updated the table title to Table A.25. Receive PLL Current Charge Pump [reg76].

Technical Support Assistance	Added link for FAQ website.

Revision 1.0, December 2022

Section	Change Summary
All	• Updated some register name from decimal to hexadecimal. • Updated the descriptions of Signal Detector. • Updated the descriptions of mpcs_clkin_i and epcs_clkin_i signals. • Updated the descriptions of mpcs_skipbit_i and epcs_skipbit_i signals. • Updated the descriptions for the limitation of 64B/66B PCS implementation. • Updated the descriptions for mpcs_phyrdy_o and epcs_phyrdy_o signals. • Updated/Added some descriptions for PMA Only mode. • Updated descriptions for GPLL reference clock related contents. • Updated the read clock of 64B/66B PCS Rx FIFO connections in Figure 5.13. Detailed Channel Block Diagram of MPCS-64B/66B Mode, Figure 7.2. 64B/66B PCS Channel Clock Diagram, Figure 7.15. 64B/66B PCS with Using GPLL, Figure 7.16. 64B/66B PCS without using GPLL (Case I), and Figure 7.17. 64B/66B PCS without Using GPLL (Case II).
Standards Supported	General update to the whole section.

Section	Change Summary

Architecture Overview	• In Table 5.7. MPCS Interface:
	  • updated the description for mpcs_clkin_i port, mpcs_phyrdy_o, and mpcs_skipbit_i ports.
	• In Table 5.8. EPCS Interface:
	  • updated the description for epcs_clkin_i and epcs_phyrdy_o ports.
	• In Table 5.10. LMMI Interface:
	  • removed lmmi_addr_i port.
	• In Table 5.11. Other Signals:
	  • updated description for sd_pll_refclk_i port.
	• Added Note that only the channel 2 and channel 3 of each MPCS quad contains 64B/66B PCS channel to the MPCS-64B/66B Mode section.

SerDes/PCS Function Description	• In the MPCS section:
	  • updated to the rising edge of walign_en signal triggers the word alignment operation and in manual mode, the rising edge of skipbit signal triggers the input parallel data to shift one UI inside PMA in the Word Aligner section;
	  • updated Figure 6.5. MPCS 8B/10B PCS Block Diagram in the 8B/10B PCS section;
	  • rearranged some figures and description orders;
	  • updated Figure 6.24. 64B/66B PCS Channel Block Diagram in the 64B/66B PCS section;
	  • general update to the Data Bus Description in the PMA Only Mode section.

Clocks and Reset	• Updated Figure 7.4. Quad Clock Distribution Diagram in the MPCS Quad Clock Detail section.
	• In the SerDes/PCS Reset section:
	  • newly added Reset Generation;
	  • general update to the Reset Sequence section.

SerDes Equalization	Updated Figure 8.6. Receive Equalizer Block Diagram.
SerDes/PCS Debug Capabilities	• Updated Figure 9.1. PMA Loopback Mode in the Loopback Mode section. • Newly added contents to the Eye Monitor section.
SerDes/PCS Register Access	Added contents about LMMI software reset generation.
Other Design Considerations	General update to the Reference Clock Source Selection section.
SerDes/PCS Generation in Radiant Software	• Made general update to the contents including the figures to keep up with the latest version of MPCS foundational IP. • Removed the original *Attribute Summary* section.

Configuration Registers	• Updated some registers descriptions.
	• Newly added the following registers:
	  • Table A. 30. Adaptive Equalization Enable Register [regd9];
	  • Table A. 31. Applied Rx Equalization A0 Gain Register [rege4];
	  • Table A. 32. Applied Rx Equalization A1 Gain Register [rege5];
	  • Table A. 33. Applied Rx Equalization A2 Gain Register [rege6], and
	  • Table A. 141. PMA and PIPE PHY Status [regfe].

References	Removed *LatticeECP3 Family Handbook (HB1009)*.

Revision 0.84, July 2022

Section	Change Summary
Architecture Overview	Removed XAUI information from the mixed mode support in Table 5.7.

Revision 0.83, January 2022

Section	Change Summary
LMMI Interface	Removed the lmmi_addr_i row from Table 5.10.

Revision 0.82, December 2021

Section	Change Summary
All	Changed document title to *CertusPro-NX SerDes/PCS User Guide*.
Architecture Overview	Updated description of PCI Express Link Layer Quad0 configuration to *PCI Express Link Layer Quad0 can be configured as x4 mode, x2 mode, x1 mode, x2_x1 mode, and x1_x1 mode*.
All	Updated reference to PCIe X4 IP Core - Lattice Radiant Software (FPGA-IPUG-02126).

Revision 0.81, May 2021

Section	Change Summary
All	Initial preliminary release.

Revision 0.70, January 2021

Section	Change Summary
All	Advance release.

www.latticesemi.com