Cyclone V Device Handbook

Volume 2: Transceivers

Subscribe

Send Feedback

CV-5V3
2018.10.24

101 Innovation Drive
San Jose, CA 95134
www.altera.com

ALTERA® now part of Intel

Contents

Transceiver Architecture in Cyclone V Devices

1

2016.01.28

CV-53001 Subscribe Send Feedback

Describes the Cyclone® V transceiver architecture, clocking, channels, channel bonding, and transmitter and receiver channel datapaths.

Altera® 28-nm Cyclone V devices provide transceivers with the lowest power requirement at 3.125 and 6.144 Gigabits per second (Gbps). These transceivers comply with a wide range of protocols and data rate standards; however, 6.144 Gbps support is limited to the common public radio interface (CPRI) protocol only.

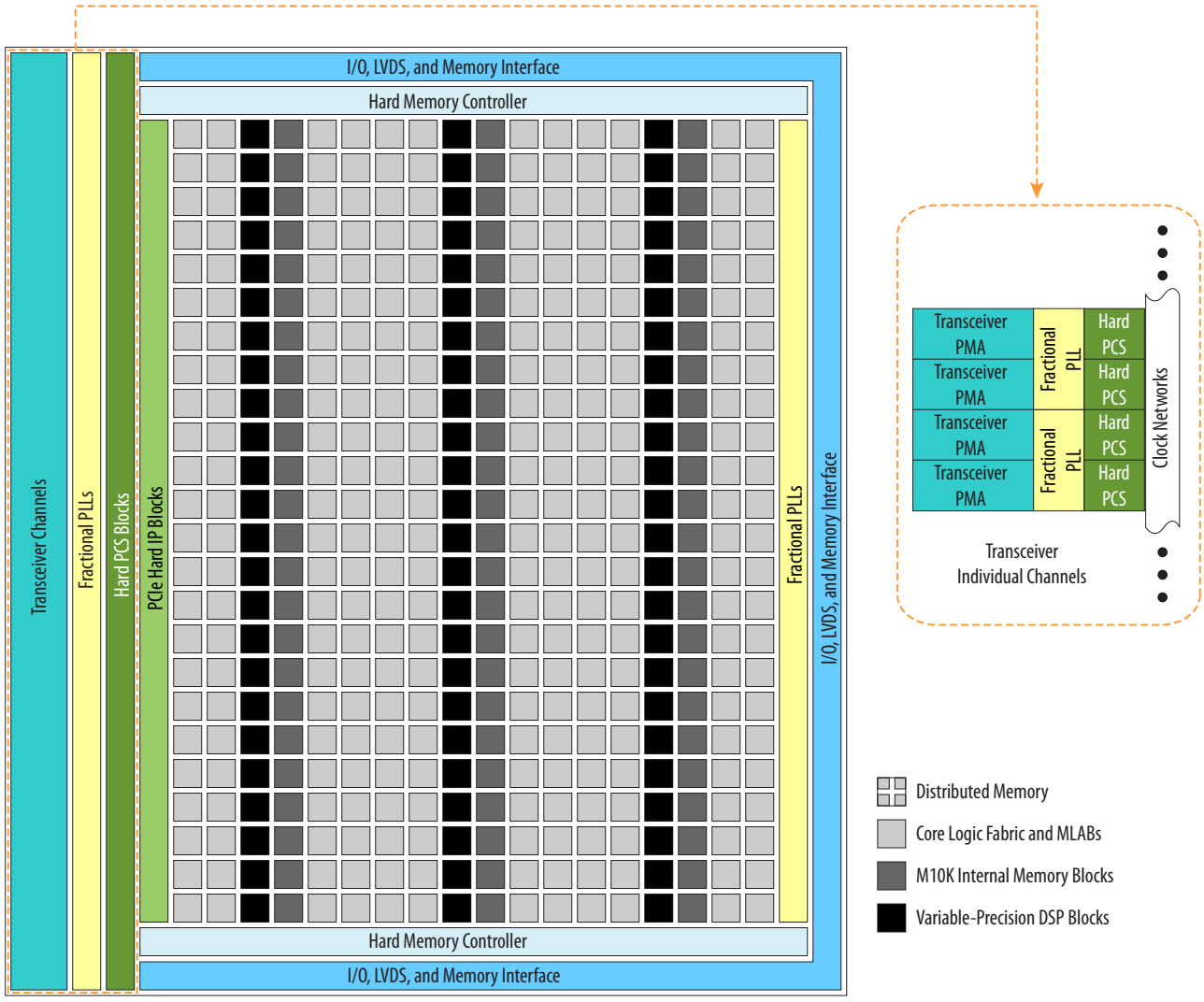
Cyclone V devices have up to 12 transceiver channels with serial data rates between 614 megabits per second (Mbps) and 6.144 Gbps and have backplane-capable transceiver support for PCI Express® (PCIe®) Base Specification 2.0 Gen1 and Gen2 up to x4 bonded channels.

Cyclone V transceiver channels are full-duplex and clock data recovery (CDR)–based with physical coding sublayer (PCS) and physical medium attachment (PMA) layers.

Architecture Overview

Figure 1-1: Basic Layout of Transceivers in a Cyclone V Device

This figure represents a Cyclone V device with transceivers. Other Cyclone V devices may have a different floor plan than the one shown here.



The embedded high-speed clock networks in Cyclone V devices provide dedicated clocking connectivity for the transceivers. You can also use the fractional phase-locked loop (fPLL) between the PMA and PCS to clock the transceivers.

The embedded PCIe hard intellectual property (IP) of Cyclone V devices implements the following PCIe protocol stacks:

- Physical interface/media access control (PHY/MAC) layer
- Data link layer
- Transaction layer

The embedded hard IP saves significant FPGA resources, reduces design risks, and reduces the time required to achieve timing closure. The hard IP complies with the PCIe Base Specification 2.0 for Gen1 and Gen2 signaling data rates.

Related Information

- **Cyclone V Device Handbook: Known Issues**
  Lists the planned updates to the *Cyclone V Device Handbook* chapters.
- **Transceiver Clocking in Cyclone V Devices**
- **IP Compiler for PCI Express User Guide**
- **For more information about the PCIe hard IP block architecture, see the Cyclone V Hard IP for PCI Express User Guide.**

Transceiver Banks

Cyclone V transceivers are grouped in transceiver banks of three channels. Some Cyclone V devices support four or five transceiver channels.

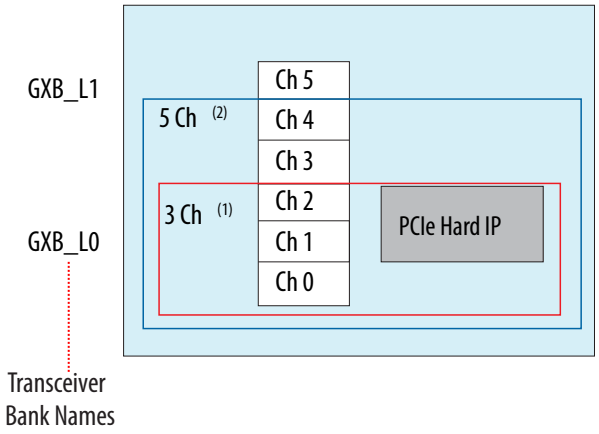
Every transceiver bank is comprised of three channels (ch 0, ch 1, and ch 2, or ch 3, ch 4 , and ch 5). The Cyclone V device family has a total of four transceiver banks (for the largest density family) namely, GXB_L0, GXB_L1, GXB_L2 and GXB_L3.

The location of the transceiver bank boundaries are important for clocking resources, bonding channels, and fitting.

In some package variations, the total transceiver count is reduced.

Figure 1-2: GX/GT Devices with Three or Five Transceiver Channels and One PCIe Hard IP Block

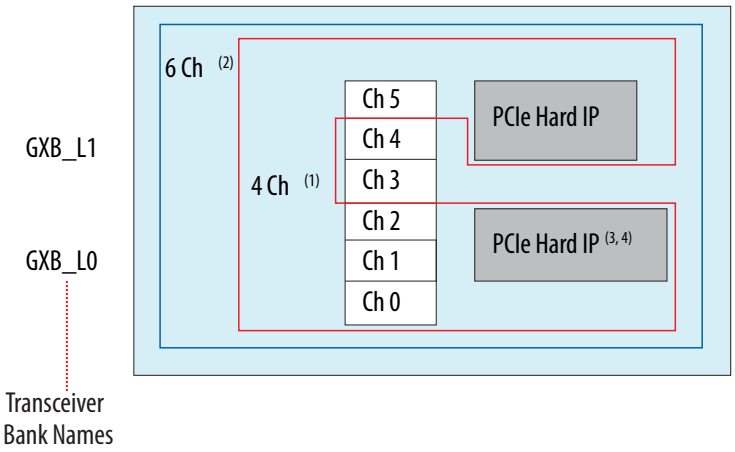
The PCIe Hard IP block is located across Ch 1 and Ch 2 of banks GXB_L0.



Notes:
1. 3-channel device transceiver channels are located on bank L0.
2. 5-channel device transceiver channels are located on bank L0, and Ch 3 and Ch 4 of bank L1.

Figure 1-3: GX/GT Devices with Four or Six Transceiver Channels and Two PCIe Hard IP Blocks

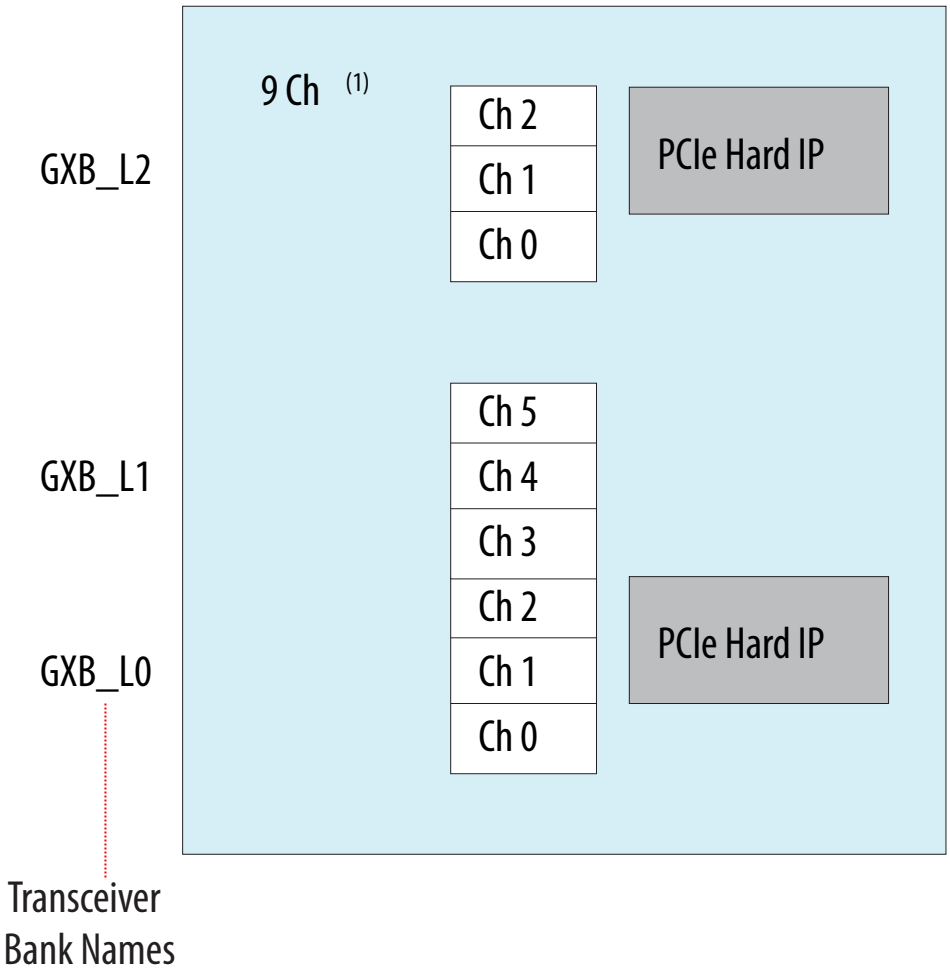
The PCIe Hard IP blocks are located across Ch 1 and Ch 2 of bank GXB_L0, and Ch 4 and Ch 5 of bank GXB_L1.



Notes:
1. 4-channel device transceiver channels are located on bank L0, and Ch 5 of bank L1.
2. 6-channel device transceiver channels are located on banks L0 and L1.
3. 5CGXC3 and5CGXC7 devices in the U19 package and 5CGXC3, 5CGXC7, and 5CGXC9 devices in the F23 package only support one PCIe Hard IP due to bonding restrictions.
4. 5CGTD7 devices in the U19 package and 5CGTD9 devices in the F23 package only support one PCIe Hard IP due to bonding restrictions.

Figure 1-4: GX/GT/SX/ST Devices with Nine Transceiver Channels and Two PCIe Hard IP Blocks

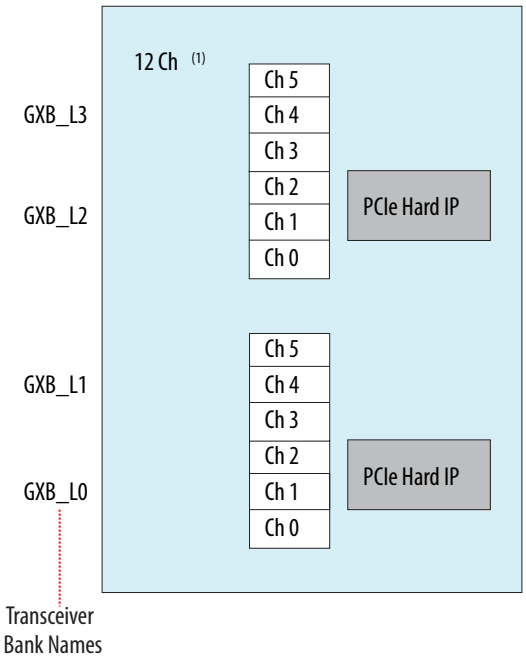
The PCIe Hard IP blocks are located across Ch 1 and Ch 2 of bank GXB_L0, and Ch 1 and Ch 2 of bank GXB_L2.



Note:
1. 9-channel device transceiver channels are located on banks L0, L1, and L2.

Figure 1-5: GX/GT Devices with 12 Transceiver Channels and Two PCIe Hard IP Blocks

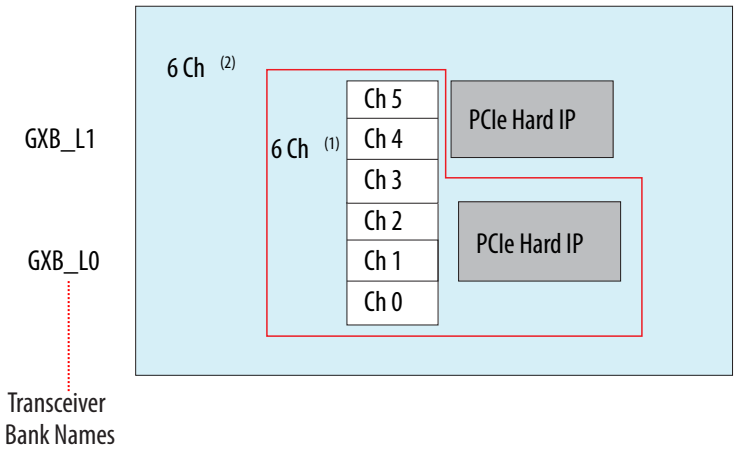
The PCIe Hard IP blocks are located across Ch 1 and Ch 2 of bank GXB_L0, and Ch 1 and Ch 2 of bank GXB_L2.



Note:
1. 12-channel device transceiver channels are located on banks L0, L1, L2, and L3.

Figure 1-6: SX Device with Six Transceiver Channels and One or Two PCIe Hard IP Blocks

The PCIe Hard IP blocks are located across Ch 1 and Ch 2 of bank GXB_L0, and Ch 4 and Ch 5 of bank GXB_L1.



Notes:
1. 6 transceiver channels with one PCIe Hard IP block.
2. 6 transceiver channels with two PCIe Hard IP blocks.

Usage Restrictions on Specific Channels

Channels next to PCIe Hard IP block are not timing-optimized for the 6.144 Gbps CPRI data rate. Avoid placing the 6.144 Gbps CPRI channels in affected channels. The affected channels can still be used as a CMU to clock the CPRI channels.

Table 1-1: Usage Restrictions on Specific Channels Across Device Variants

Channels	Channel Bank Location	Usage Restriction
Ch 1, Ch 2	GXB_L0	• No 6.144 Gbps CPRI support • No support for PCS with phase compensation FIFO in registered mode
Ch 4, Ch 5	GXB_L1(1)	
Ch 1, Ch 2	GXB_L2(1)	

Cyclone V GX transceiver channels are comprised of a transmitter and receiver that can operate individually and simultaneously—providing a full-duplex physical layer implementation for high-speed serial interfacing.

(1) Impacted only if the device has PCIe Hard IP block located next to this bank.

The transmitter and receiver in a channel are structured into PMA and PCS sections:

- PMA—converts serial data to parallel data and vice versa for connecting the FPGA to a serial transmission medium.
- PCS—prepares the parallel data for transmission across a physical medium or restores the data to its original form using hard digital logic implementation.

6.144 Gbps CPRI Support Capability in GT Devices

You can configure Cyclone V GT devices to support 6.144 Gbps for CPRI protocol only. The Cyclone V GT device supports up to three full duplex channels that are compliant to the 6.144 Gbps CPRI protocol for every two transceiver banks. The transceivers are grouped in transceiver banks of three channels.

Altera recommends that you increase the `VCCE_GXBL` and `VCCL_GXBL` to a nominal value of 1.2 V in order to be compliant to the 6.144Gbps CPRI protocol. The reference clock frequency for the 6.144 Gbps CPRI channel must be ≥ 307.2 MHz.

Related Information

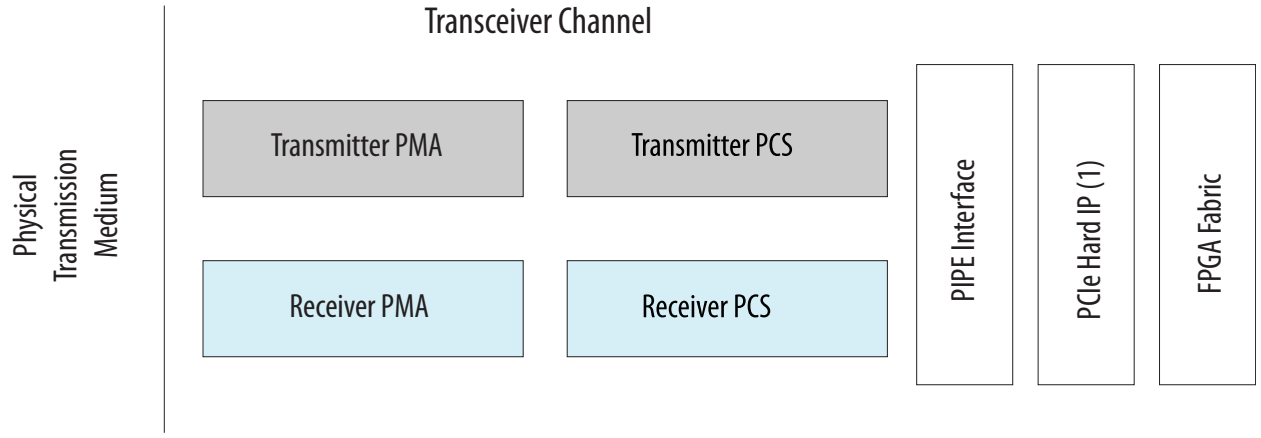
Transceiver Protocol Configurations in Cyclone V Devices

Transceiver Channel Architecture

Cyclone V transceiver channels support the following interface methods with the FPGA fabric:

- Directly—bypassing the PIPE interface for the PCIe interface and PCIe hard IP block
- Through the PIPE interface and PCIe hard IP block—for hard IP implementation of the PCIe protocol stacks (PHY/MAC, data link layer, and transaction layer)

Figure 1-7: Transceiver Channel Block Diagram in Cyclone V Devices



Note:
1. Only certain transceiver channels support interfacing to the PCIe hard IP block.

You can bond multiple channels to implement a multilane link.

Related Information

Transceiver Banks on page 1-3

PMA Architecture

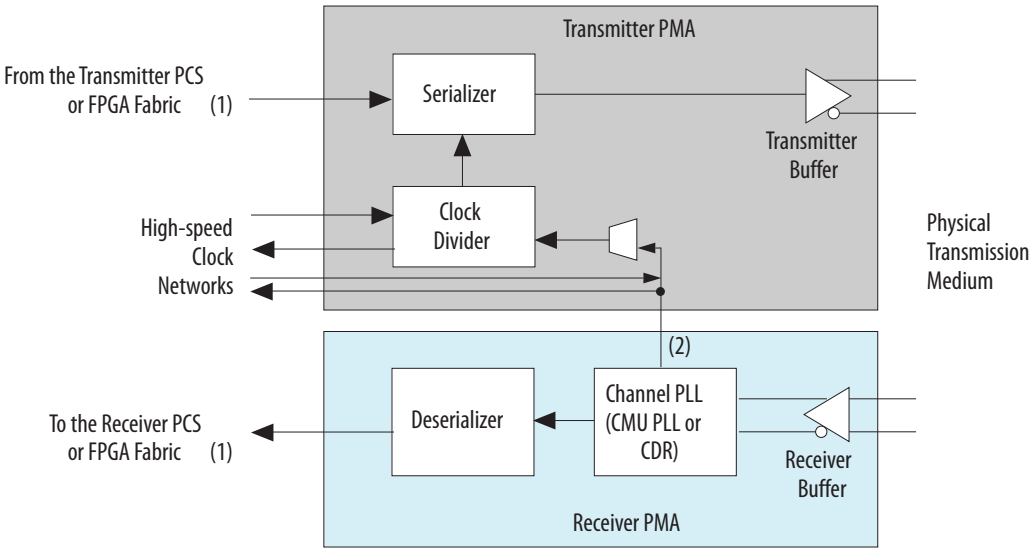
The PMA includes the transmitter and receiver datapaths, clock multiplier unit (CMU) PLL—configured from the channel PLL—and the clock divider. The analog circuitry and differential on-chip termination (OCT) in the PMA requires the calibration block to compensate for process, voltage, and temperature variations (PVT).

Each transmitter channel has a clock divider. There are two types of clock dividers, depending on the channel location in a transceiver bank:

- Channels 0, 2, 3, and 5—local clock divider
- Channels 1 and 4—central clock divider

Using clocks from the clock lines and CMU PLL, the clock divider generates the parallel and serial clock sources for transmitter and optionally for the receiver PCS. The central clock divider can additionally feed the clock lines used to bond channels compared to the local clock divider.

Figure 1-8: PMA Block Diagram of a Transceiver Channel in Cyclone V Devices



Notes:
1. The channel PLL provides the serial clock when configured as a CMU PLL.
2. The channel PLL recovers the clock and serial data stream when configured as a CDR.

Related Information

- **Altera Transceiver PHY IP Core User Guide**
- **IP Compiler for PCI Express User Guide**

Transmitter PMA Datapath

Table 1-2: Functional Blocks in the Transmitter PMA Datapath

Block	Functionality
Serializer	• Converts the incoming low-speed parallel data from the transmitter PCS to high-speed serial data and sends the data LSB first to the transmitter buffer. • Supports 8-, 10-, 16-, and 20-bit serialization factors. • Supports the optional polarity inversion and bit reversal features.
Transmitter Buffer	• The 1.5-V PCML output buffer conditions the high-speed serial data for transmission into the physical medium. • Programmable differential output voltage (VOD) • Programmable pre-emphasis • Programmable V_{CM} current strength • Programmable slew rate • On-chip biasing for common-mode voltage (TX V_{CM}) • Differential OCT (85, 100, 120, and 150 Ω) • Transmitter output tristate • Receiver detect (for the PCIe receiver detection function)

Related Information

Transmitter Buffer Features and Capabilities on page 1-14

Serializer

The serializer converts the incoming low-speed parallel data from the transceiver PCS to high-speed serial data and sends the data to the transmitter buffer.

The serializer supports 8, 10, 16, and 20 bits of serialization factors. The serializer block sends out the LSB of the input data first. The transmitter serializer also has polarity inversion and bit reversal capabilities.

Transmitter Polarity Inversion

The positive and negative signals of a serial differential link might accidentally be swapped during board layout. The transmitter polarity inversion feature is provided to correct this situation without requiring a board re-spin or major updates to the logic in the FPGA fabric.

A high value on the `tx_invpolarity` port inverts the polarity of every bit of the input data word to the serializer in the transmitter datapath. Because inverting the polarity of each bit has the same effect as swapping the positive and negative signals of the differential link, correct data is sent to the receiver. The dynamic `tx_invpolarity` signal might cause initial disparity errors at the receiver of an 8B/10B encoded link. The downstream system must be able to tolerate these disparity errors.

Caution: If the polarity inversion is asserted midway through a serializer word, the word will be corrupted.

Bit Reversal

You can reverse the transmission bit order to achieve MSB-to-LSB ordering using the bit reversal feature at the transmitter.

Table 1-3: Bit Reversal Feature

	Transmission Bit Order	
Bit Reversal Option	8- or 10-bit Serialization Factor	16- or 20-bit Serialization Factor
Disabled (default)	LSB to MSB	LSB to MSB
Enabled	MSB to LSB For example: 8-bit—`D[7:0]` rewired to `D[0:7]` 10-bit—`D[9:0]` rewired to `D[0:9]`	MSB to LSB For example: 16-bit—`D[15:0]` rewired to `D[0:15]` 20-bit—`D[19:0]` rewired to `D[0:19]`

Transmitter Buffer

Transmitter buffers support the programmable analog settings (differential output voltage and pre-emphasis), common-mode voltage (TX V_{CM}), and OCT.

The transmitter buffer includes additional circuitry to improve integrity, such as the programmable differential output voltage (VOD), programmable three-tap pre-emphasis circuitry, internal termination circuitry, and PCIe receiver detect capability to support a PCIe configuration.

Modifying programmable values withing transmitter output buffers can be performed by a single reconfiguration controller for the entire FPGA, or multiple reconfiguration controllers if desired. Within each transceiver bank (three-transceiver channels), a maximum of one reconfiguration controller is allowed. There is only one slave interface to all PLLs and PMAs within each transceiver bank. Therefore, many transceiver banks can be connected to a single reconfiguration controller, but only one reconfiguration controller can be connected to the transceiver bank (three-transceiver channels).

Note: A maximum of one reconfiguration controller is allowed per transceiver bank.

Figure 1-9: Transmitter Buffer Block Diagram in Cyclone V Devices

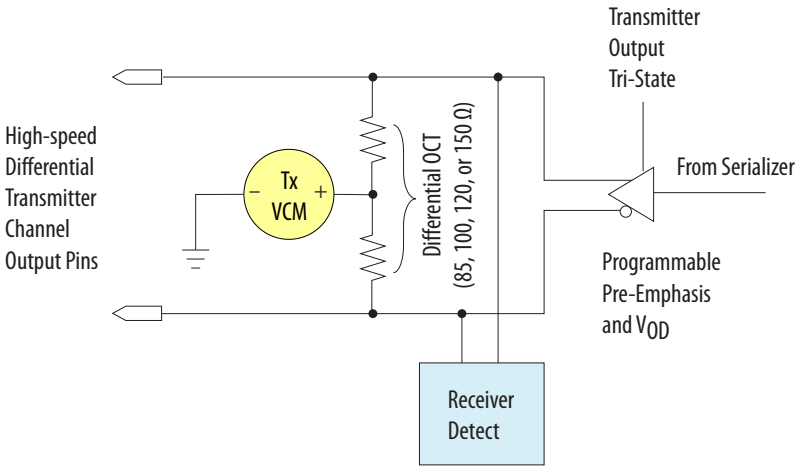


Table 1-4: Description of the Transmitter Buffer Features

Features provided by the Pseudo Current Mode Logic (PCML) output buffer to the integrated circuitry.

Category	Features	Description
Improve Signal Integrity	Programmable Differential Output Voltage (V_{OD})	Controls the current mode drivers for signal amplitude to handle different trace lengths, various backplanes, and receiver requirements. The actual V_{OD} level is a function of the current setting and the transmitter termination value.
	Programmable Pre-Emphasis	Boosts the high-frequency components of the transmitted signal, which may be attenuated when propagating through the transmission medium. The physical transmission medium can be represented as a low-pass filter in the frequency domain. Variation in the signal frequency response that is caused by attenuation significantly increases the data-dependent jitter and other intersymbol interference (ISI) effects at the receiver end. Use the pre-emphasis feature to maximize the data opening at the far-end receiver.
	Programmable Slew Rate	Controls the rate of change for the signal transition.

Category	Features	Description
Save Board Space and Cost	On-Chip Biasing	Establishes the required transmitter common-mode voltage (TX V_{CM}) level at the transmitter output. The circuitry is available only if you enable OCT. When you disable OCT, you must implement off-chip biasing circuitry to establish the required TX V_{CM} level.
	Differential OCT	The termination resistance is adjusted by the calibration circuitry, which compensates for PVT. You can disable OCT and use external termination. However, you must implement off-chip biasing circuitry to establish the required TX V_{CM} level. TX V_{CM} is tri-stated when you use external termination.
Reduce Power	Programmable V_{CM} Current Strength	Controls the impedance of V_{CM}. A higher impedance setting reduces current consumption from the on-chip biasing circuitry.
Protocol-Specific Function	Transmitter Output Tri-State	Enables the transmitter differential pair voltages to be held constant at the same value determined by the TX V_{CM} level with the transmitter in the high impedance state. This feature is compliant with differential and common-mode voltage levels and operation time requirements for transmitter electrical idle, as specified in the PCI Express Base Specification 2.0 for Gen1 and Gen2 signaling rates.
	Receiver Detect	Provides link partner detection capability at the transmitter end using an analog mechanism for the receiver detection sequence during link initialization in the Detect state of the PCIe Link Training and Status State Machine (LTSSM) states. The circuit detects if there is a receiver downstream by changing the transmitter common-mode voltage to create a step voltage and measuring the resulting voltage rise time. For proper functionality, the series capacitor (AC-coupled link) and receiver termination values must comply with the PCI Express Base Specification 2.0 for Gen1 and Gen2 signaling rates. The circuit is clocked using `fixedclk` and requires an enabled transmitter OCT with the output tri-stated.

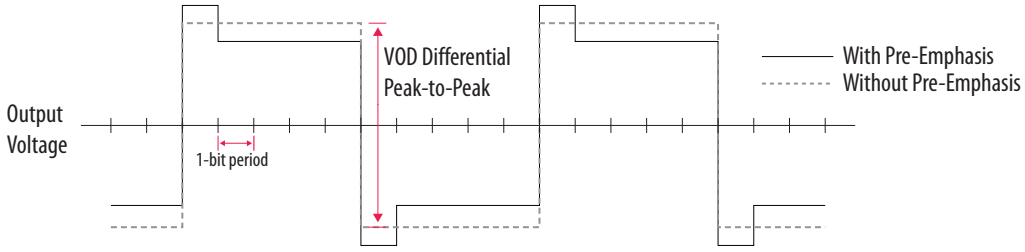
Transmitter Buffer Features and Capabilities

Table 1-5: Transmitter Buffer Features

Feature	Capability
Programmable Differential Output Voltage (V_{OD})	Up to 1200 mV of differential peak-to-peak output voltage
Programmable Pre-Emphasis	Support first Updated the Post Tap Pre-emphasis setting
On-Chip Biasing for Common-Mode Voltage (TX V_{CM})	0.65 V
Differential OCT	85, 100, 120, and 150 Ω
Transmitter Output Tri-State	Supports the electrical idle function at the transmitter as required by the PCIe Base Specification 2.0 for Gen1 and Gen2 signaling rates
Receiver Detect	Supports the receiver detection function as required by the PCIe Base Specification 2.0 for Gen1 and Gen2 signaling rates

Figure 1-10: Example of the Pre-Emphasis Effect on Signal Transmission at the Transmitter Output

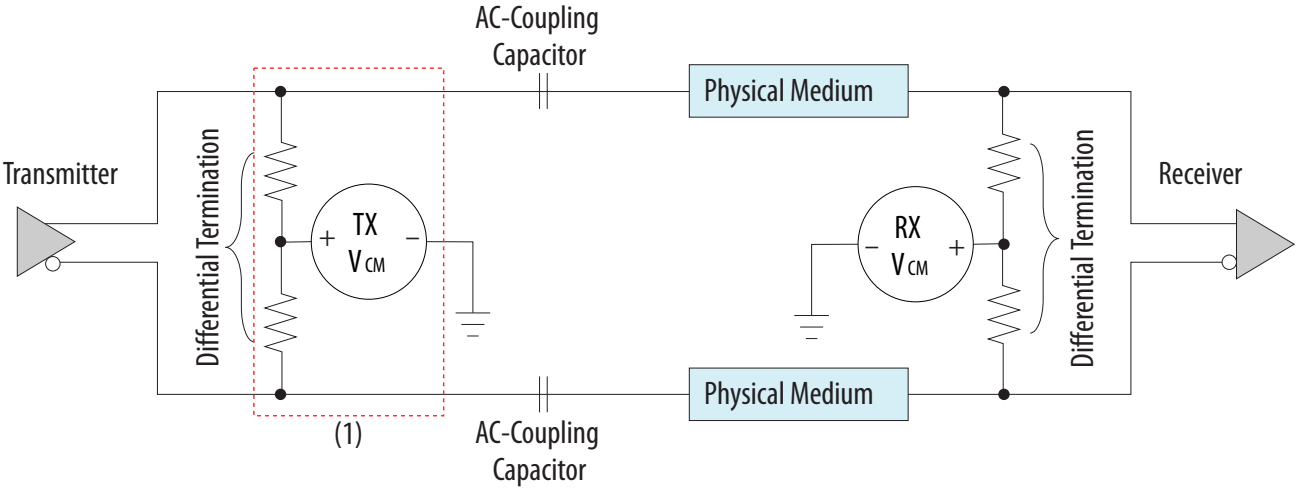
Signal transmission at the transmitter output with and without applying pre-emphasis post-tap for a 3.125 Gbps signal with an alternating data pattern of five 1s and five 0s.



You can AC-couple the transmitter to a receiver. In an AC-coupled link, the AC-coupling capacitor blocks the transmitter common-mode voltage. At the receiver end, the termination and biasing circuitry restores the common-mode voltage level that is required by the receiver.

Figure 1-11: AC-Coupled Link with a Cyclone V Transmitter

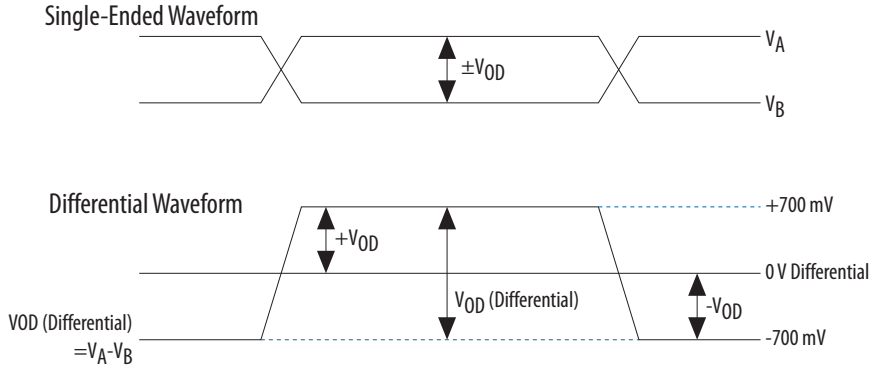
The PCIe spec requires that you enable transmitter OCT for receiver detect operation.



Notes:
1. When you disable OCT, you must implement external termination and off-chip biasing circuitry to establish the required TX V_{CM} level.

Programmable Transmitter Analog Settings

Figure 1-12: V_{OD} (Differential) Signal Level



Each transmit buffer has programmable pre-emphasis circuits that boost high frequencies in the transmit data signal, which might be attenuated in the transmission media. Using pre-emphasis can maximize the data eye opening at the far-end receiver. The pre-emphasis circuitry provides first post-tap settings with up to 6 dB of high-frequency boost.

Programmable Transmitter V_{CM}

The transmitter buffers have on-chip biasing circuitry to establish the required V_{CM} at the transmitter output. The circuitry supports a V_{CM} setting of 0.65 V.

Note: On-chip biasing circuitry is available only if you select one of the Termination logic options in order to configure OCT. If you select external termination, you must implement off-chip biasing circuitry to establish the V_{CM} at the transmitter output buffer.

Programmable Transmitter Differential OCT

The transmitter buffers support optional differential OCT resistances of 85, 100, 120, and 150 Ω . The resistance is adjusted by the on-chip calibration circuit during calibration, which compensates for PVT changes. The transmitter buffers are current mode drivers. Therefore, the resultant V_{OD} is a function of the transmitter termination value.

Transmitter Protocol Specific

There are two PCIe features in the transmitter PMA section—receiver detect and electrical idle.

- PCIe Receiver Detect—The transmitter buffers have a built-in receiver detection circuit for use in PCIe configurations for Gen1 and Gen2 data rates. This circuit detects whether there is a receiver downstream by sending out a pulse on the common mode of the transmitter and monitoring the reflection.
- PCIe Electrical Idle—The transmitter output buffers support transmission of PCIe electrical idle (or individual transmitter tri-state).

Related Information

Altera Transceiver PHY IP Core User Guide

Receiver PMA Datapath

There are three blocks in the receiver PMA datapath—the receiver buffer, channel PLL configured for clock data recovery (CDR) operation, and deserializer.

Table 1-6: Functional Blocks in the Receiver PMA Datapath

Block	Functionality

Receiver Buffer	• Receives the serial data stream and feeds the stream to the channel PLL if you configure the channel PLL as a CDR
	• Supports the following features:
	  • Programmable CTLE (Continuous Time Linear Equalization)
	  • Programmable DC gain
	  • Programmable V_{CM} current strength
	  • On-chip biasing for common-mode voltage (RX V_{CM})
	  • I/O standard (1.5 V **PCML**, 2.5 V **PCML**, **LVDS**, **LVPECL**)
	  • Differential OCT (85, 100, 120 and 150 Ω)
	  • Signal detect

Block	Functionality
Channel PLL	• Recovers the clock and serial data stream if you configure the channel PLL as a CDR. • Requires offset cancellation to correct the analog offset voltages. • If you do not use the channel PLL as a CDR, you can configure the channel PLL as a CMU PLL for clocking the transceivers.
Deserializer	• Converts the incoming high-speed serial data from the receiver buffer to low-speed parallel data for the receiver PCS. • Receives serial data in LSB-to-MSB order. • Supports 8-, 10-, 16-, and 20-bit deserialization factors. • Supports the optional clock-slip feature for applications with stringent latency uncertainty requirement.

Receiver Buffer

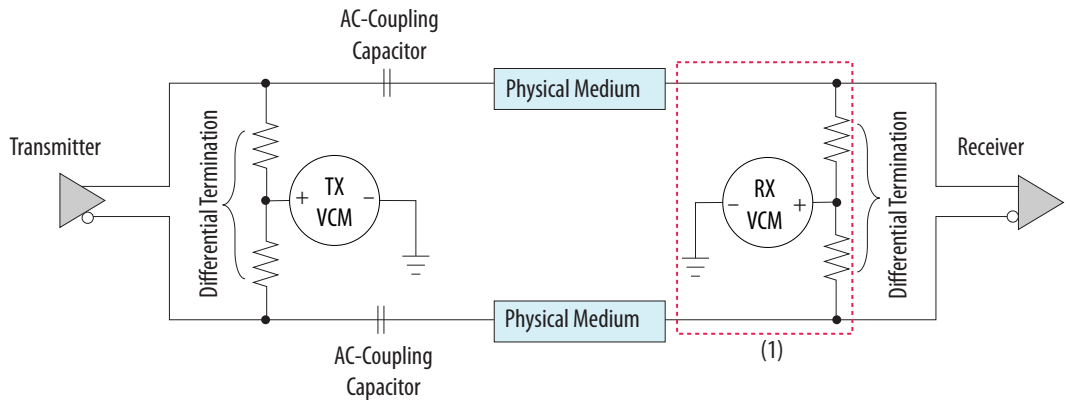
Table 1-7: Cyclone Receiver Buffer Features

Category	Features	Description
Improve Signal Integrity	Programmable Continuous Time Linear Equalization (CTLE)	Boosts the high-frequency components of the received signal, which may be attenuated when propagating through the transmission medium. The physical transmission medium can be represented as a low-pass filter in the frequency domain. Variation in the signal frequency response that is caused by attenuation leads to data-dependent jitter and other ISI effects—causing incorrect sampling on the input data at the receiver. The amount of the high-frequency boost required at the receiver to overcome signal attenuation depends on the loss characteristics of the physical medium.
	Programmable DC Gain	Provides equal boost to the received signal across the frequency spectrum.
Save Board Space and Cost	On-Chip Biasing	Establishes the required receiver common-mode voltage (RX V_{CM}) level at the receiver input. The circuitry is available only if you enable OCT. When you disable OCT, you must implement off-chip biasing circuitry to establish the required RX V_{CM} level.
	Differential OCT	The termination resistance is adjusted by the calibration circuitry, which compensates for PVT. You can disable OCT and use external termination. However, you must implement off-chip biasing circuitry to establish the required RX V_{CM} level. RX V_{CM} is tri-stated when you use external termination.
Reduce Power	Programmable V_{CM} Current Strength	Controls the impedance of V_{CM}. A higher impedance setting reduces current consumption from the on-chip biasing circuitry.

Category	Features	Description
Protocol-Specific Function	Signal Detect	Senses if the signal level present at the receiver input is above or below the threshold voltage that you specified. The detection circuitry has a hysteresis response that asserts the status signal only when a number of data pulses exceeding the threshold voltage are detected and deasserts the status signal when the signal level below the threshold voltage is detected for a number of recovered parallel clock cycles. The circuitry requires the input data stream to be 8B/10B-coded. Signal detect is compliant to the threshold voltage and detection time requirements for electrical idle detection conditions as specified in the PCI Express Base Specification 2.0 for Gen1 and Gen2 signaling rates. Signal detect is also compliant to SATA/SAS protocol up to 3 Gbps support.

You can AC-couple the receiver to a transmitter. In an AC-coupled link, the AC-coupling capacitor blocks the transmitter common-mode voltage. At the receiver end, the termination and biasing circuitry restores the common-mode voltage level that is required by the receiver.

Figure 1-13: AC-Coupled Link with a Cyclone V Receiver

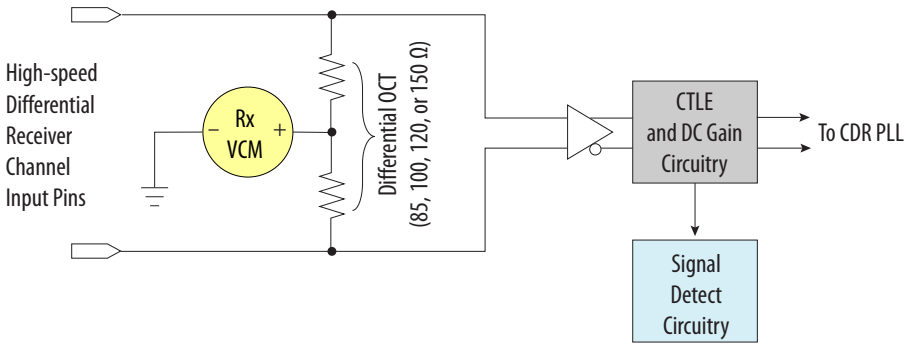


Note:
1. When you disable OCT, you must implement external termination and off-chip biasing circuitry to establish the required RX V_{CM} level.

The receiver buffers support the programmable analog settings (CTLE and DC gain), programmable common mode voltage (RX V_{CM}), OCT, and signal detect function.

The receiver input buffer receives serial data from the high-speed differential receiver channel input pins and feeds the serial data to the channel PLL configured as a CDR unit.

Figure 1-14: Receiver Buffer Block Diagram in Cyclone V Devices



Modifying programmable values within receiver input buffers can be performed by a single reconfiguration controller for the entire FPGA, or multiple reconfiguration controllers if desired. Within each transceiver bank (three-transceiver channels) a maximum of one reconfiguration controller is allowed. There is only one slave interface to all PLLs and PMAs within each transceiver bank. Therefore, many transceiver banks can be connected to a single reconfiguration controller, but only one reconfiguration controller can be connected to the transceiver bank (three-transceiver channels).

Note: A maximum of one reconfiguration controller is allowed per transceiver bank.

Programmable CTLE and DC Gain

Each receiver buffer has a single-tap programmable equalization circuit that boosts the high-frequency gain of the incoming signal, thereby compensating for the low-pass filter effects of the physical medium. The amount of high-frequency gain required depends on the loss characteristics of the physical medium. The equalization circuitry provides up to 4 dB of high-frequency boost.

Each receiver buffer also supports the programmable DC gain circuitry that provides an equal boost to the incoming signal across the frequency spectrum. The DC gain circuitry provides up to 3 dB of gain setting.

Programmable Receiver V_{CM}

The receiver buffers have on-chip biasing circuitry to establish the required V_{CM} at the receiver input. The circuitry supports a V_{CM} setting of 0.8 V.

On-chip biasing circuitry is available only if you select one of the termination logic options in order to configure OCT. If you select external termination, you must implement off-chip biasing circuitry to establish V_{CM} at the receiver input buffer.

Programmable Receiver Differential On-Chip Termination

The receiver buffers support optional differential OCT resistances of 85, 100, 120, and 150 Ω . The resistance is adjusted by the on-chip calibration circuit during calibration, which compensates for PVT changes.

Signal Threshold Detection Circuitry

In a PCIe and SATA/SAS configuration, the signal threshold detection circuitry will be enabled to detect the present of incoming signal.

The signal threshold detection circuitry senses whether the signal level present at the receiver input buffer is above the signal detect threshold voltage you specified.

Deserializer

The deserializer block clocks in serial input data from the receiver buffer using the high-speed serial recovered clock and deserializes the data using the low-speed parallel recovered clock. The deserializer forwards the deserialized data to the receiver PCS.

The deserializer supports 8, 10, 16, and 20 bits of deserialization factors.

Figure 1-15: Deserializer Operation with a 10-bit Deserialization Factor

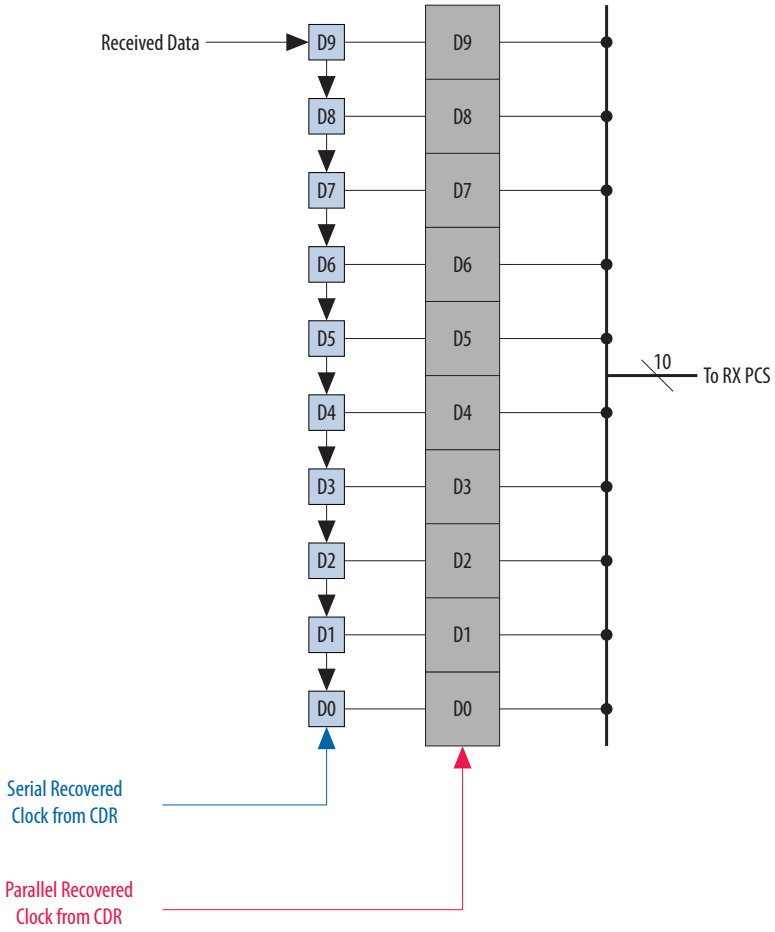
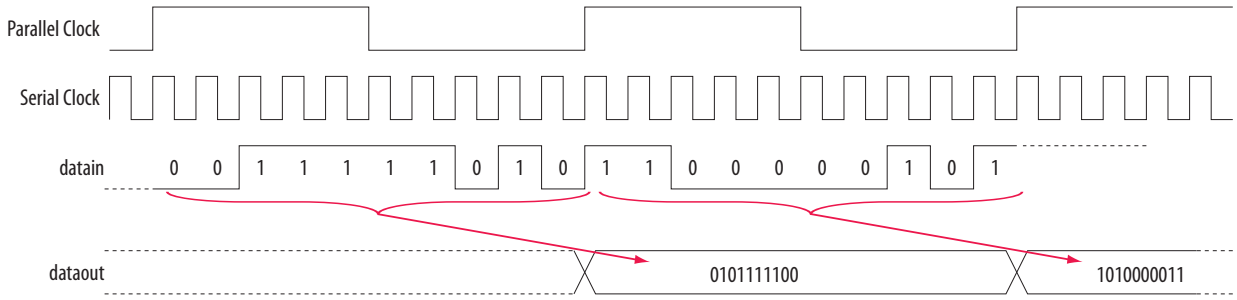


Figure 1-16: Deserializer Bit Order with 10-bit Deserialization Factor

The serial stream (0101111100) is deserialized to a 10'h17C value. The serial data is received LSB to MSB.



Clock-slip

Word alignment in the PCS may contribute up to one parallel clock cycle of latency uncertainty. The clock-slip feature allows word alignment operation with a reduced latency uncertainty by performing the word alignment function in the deserializer. Use the clock slip feature for applications that require deterministic latency.

The deterministic latency state machine in the word aligner from the PCS automatically controls the clock-slip operation. After completing the clock-slip process, the deserialized data is word-aligned into the receiver PCS.

Transmitter PLL

In Cyclone V GX/GT/SX/ST devices, there are two transmitter PLL sources: CMU PLL (channel PLL) and fPLL. The channel PLL can be used as CMU PLL to clock the transceivers or as clock data recovery (CDR) PLL.

Note: Cyclone V transceiver channels support full-duplex operation. The CMU PLL is sourced from the channel PLL of channels 1 or 4.

Table 1-8: Transmitter PLL Capability and Availability

Transmitter PLL	Serial Data Range	Availability
CMU PLL	0.611 Gbps to 6.144 Gbps	Every channel when not used as receiver CDR
fPLL	0.611 Gbps to 3.125 Gbps	Two per transceiver bank

Related Information

Transceiver Clocking in Cyclone V Devices

Channel PLL Architecture

In LTR mode, the channel PLL tracks the input reference clock. The PFD compares the phase and frequency of the voltage controlled oscillator (VCO) output and the input reference clock. The resulting PFD output controls the VCO output frequency to half the data rate with the appropriate counter (M or L) value given an input reference clock frequency. The lock detect determines whether the PLL has achieved lock to the phase and frequency of the input reference clock.

In LTD mode, the channel PLL tracks the incoming serial data. The phase detector compares the phase of the VCO output and the incoming serial data. The resulting phase detector output controls the VCO output to continuously match the phase of the incoming serial data.

The channel PLL supports operation in either LTR or LTD mode.

Note: Use the LTR/LTD controller only when the channel PLL is configured as a CDR PLL.

Table 1-9: Channel PLL Counters

The Quartus® II software automatically selects the appropriate counter values for each transceiver configuration.

Counter	Description	Values
N	Pre-scale counter to divide the input reference clock frequency to the PFD by the N factor	1, 2, 4, 8
M	Feedback loop counter to multiply the VCO frequency above the input reference frequency to the PFD by the M factor	1, 4, 5, 8, 10, 12, 16, 20, 25
L (PFD)	VCO post-scale counter to divide the VCO output frequency by the L factor in the LTR loop	1, 2, 4, 8
L (PD)	VCO post-scale counter to divide the VCO output frequency by the L factor in the LTD loop	1, 2, 4, 8

Channel PLL as CDR PLL

When configured as a receiver CDR, each channel PLL independently recovers the clock from the incoming serial data. The serial and parallel recovered clocks are used to clock the receiver PMA and PCS blocks.

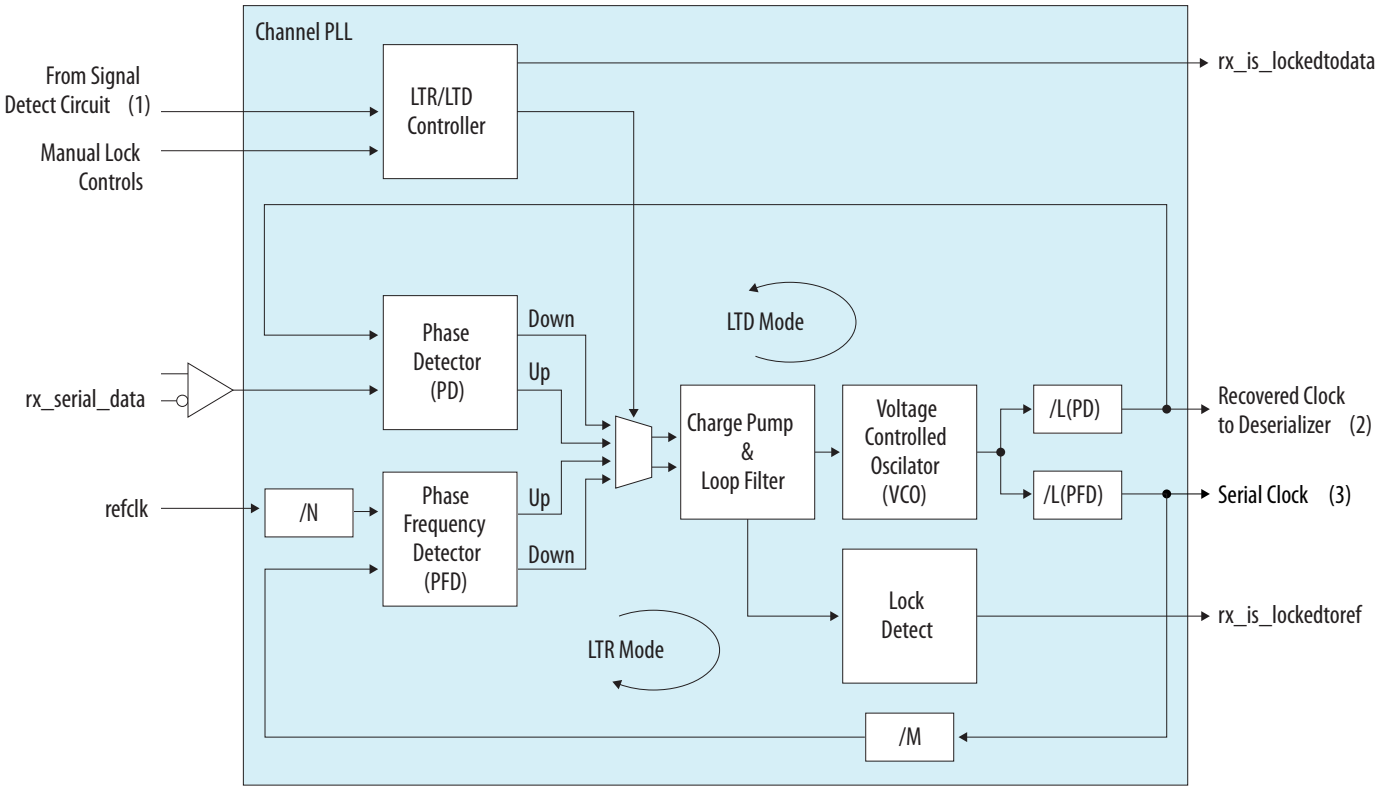
The CDR supports the full range of data rates. The voltage-controlled oscillator (VCO) operates at half rate. The L-counter dividers (PD) after the VCO extend the CDR data rate range. The Quartus II software automatically selects these settings.

The CDR operates in either lock-to-reference (LTR) or lock-to-data (LTD) mode. In LTR mode, the CDR tracks the input reference clock. In LTD mode, the CDR tracks the incoming serial data.

The time needed for the CDR PLL to lock to data depends on the transition density and jitter of the incoming serial data and the PPM difference between the receiver input reference clock and the upstream transmitter reference clock. You must hold the receiver PCS in reset until the CDR PLL locks to data and produces a stable recovered clock.

After the receiver power up and reset cycle, you must keep the CDR in LTR mode until the CDR locks to the input reference clock. When locked to the input reference clock, the CDR output clock is trained to the configured data rate. The CDR then switches to LTD mode to recover the clock from the incoming data. The LTR/LTD controller controls the switch between the LTR and LTD modes.

Figure 1-17: Channel PLL Block Diagram



Notes:
1. Applicable in a PCIe configuration and custom mode configuration, for example SATA/SAS.
2. Applicable when configured as a CDR PLL.
3. Applicable when configured as a CMU PLL.

Lock-to-Reference Mode

In LTR mode, the phase frequency detector (PFD) in the CDR tracks the receiver input reference clock. The PFD controls the charge pump that tunes the VCO in the CDR. Depending on the data rate and the selected input reference clock frequency, the Quartus II software automatically selects the appropriate /M and /L divider values so the CDR output clock frequency is half the data rate. The `rx_is_lockedtoref` status signal is asserted to indicate that the CDR has locked to the phase and frequency of the receiver input reference clock.

The phase detector is inactive in LTR mode and `rx_is_lockedtodata` is ignored.

Lock-to-Data Mode

During normal operation, the CDR must be in LTD mode to recover the clock from the incoming serial data. In LTD mode, the phase detector in the CDR tracks the incoming serial data at the receiver buffer. Depending on the phase difference between the incoming data and the CDR output clock, the phase detector controls the CDR charge pump that tunes the VCO.

Note: The PFD output is invalid in LTD mode. The `rx_is_lockedtoref` signal may toggle randomly and has no significance in LTD mode.

After switching to LTD mode, the `rx_is_lockedtodata` status signal is asserted. It can take a maximum of 1 ms for the CDR to lock to the incoming data and produce a stable recovered clock. The actual lock time depends on the transition density of the incoming data and the parts per million (ppm) difference between the receiver input reference clock and the upstream transmitter reference clock. The receiver PCS logic must be held in reset until the CDR produces a stable recovered clock.

CDR PLL in Automatic Lock Mode

In automatic lock mode, the LTR/LTD controller directs the transition between the LTR and LTD modes when a set of conditions are met to ensure proper CDR PLL operation. The mode transitions are indicated by the `rx_is_lockedtodata` status signal.

After power-up or reset of the receiver PMA, the CDR PLL is directed into LTR mode. The controller transitions the CDR PLL from LTR to LTD mode when all the following conditions are met:

- The frequency of the CDR PLL output clock and input reference clock is within the configured ppm frequency threshold setting
- The phase of the CDR PLL output clock and input reference clock is within approximately 0.08 unit interval (UI) of difference
- In PCIe configurations only—the signal detect circuitry must also detect the presence of the signal level at the receiver input above the threshold voltage specified in the PCI Express Base Specification 2.0. (Signal detect is an optional signal in Custom or Native PHY IP. Use the Assignment Editor to select the threshold voltage.)

The controller transitions the CDR PLL from LTD to LTR mode when either of the following conditions are met:

- The difference in between frequency of the CDR PLL output clock and input reference clock exceeds the configured ppm frequency threshold setting
- In PCIe configurations only—the signal detect circuitry detects the signal level at the receiver input below the threshold voltage specified in the PCI Express Base Specification 2.0

After switching to LTD mode, the `rx_is_lockedtodata` status signal is asserted. Lock to data takes a minimum of 4 µs, however the actual lock time depends on the transition density of the incoming data and the parts per million (PPM) difference between the receiver input reference clock and the upstream transmitter reference clock. The receiver PCS logic must be held in reset until the CDR produces a stable recovered clock.

If there is no transition on the incoming serial data for an extended duration, the CDR output clock may drift to a frequency exceeding the configured PPM threshold when compared with the input reference clock. In such a case, the LTR/LTD controller transitions the CDR PLL from LTD to LTR mode.

CDR PLL in Manual Lock Mode

In manual lock mode, the LTR/LTD controller directs the transition between the LTR and LTD modes based on user-controlled settings in the `pma_rx_set_locktodata` and `pma_rx_set_locktoref` registers. Alternatively you can control it using the `rx_set_locktodata` and `rx_set_locktoref` ports available in the transceiver PHY IPs.

In LTR mode, the phase detector is not active. When the CDR PLL locks to the input reference clock, you can switch the CDR PLL to LTD mode to recover the clock and data from the incoming serial data.

In LTD mode, the PFD output is not valid and may cause the lock detect status indicator to toggle randomly. When there is no transition on the incoming serial data for an extended duration, you must switch the CDR PLL to LTR mode to wait for the read serial data.

Manual lock mode provides the flexibility to manually control the CDR PLL mode transitions bypassing the PPM detection as required by certain applications that include, but not limited to, the following:

- Link with frequency differences between the upstream transmitter and the local receiver clocks exceeding the CDR PLL ppm threshold detection capability. For example, a system with asynchronous spread-spectrum clocking (SSC) downspread of –0.5% where the SSC modulation results in a PPM difference of up to 5000.
- Link that requires a faster CDR PLL transition to LTD mode, avoiding the duration incurred by the PPM detection in automatic lock mode.

In manual lock mode, your design must include a mechanism—similar to a PPM detector—that ensures the CDR PLL output clock is kept close to the optimum recovered clock rate before recovering the clock and data. Otherwise, the CDR PLL might not achieve locking to data. If the CDR PLL output clock frequency is detected as not close to the optimum recovered clock rate in LTD mode, direct the CDR PLL to LTR mode.

Related Information

Transceiver Reset Control and Power Down in Cyclone V Devices

Channel PLL as a CMU PLL

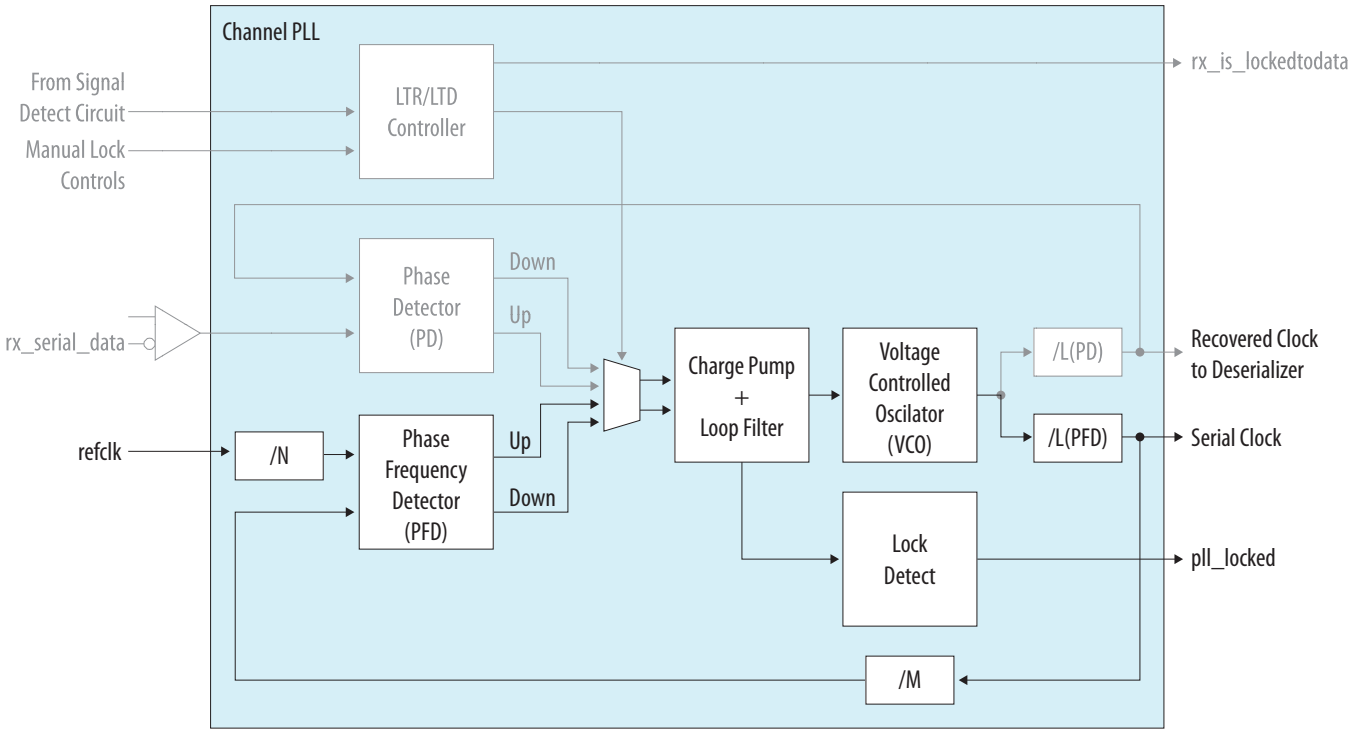
When you use the channel PLL as the CMU PLL, you can configure the transceiver channel only as a transmitter.

The CMU PLL operates in LTR mode only and supports the full range of data rates.

The VCO of the PLL operates at half rate and the L-counter dividers (PFD), after the VCO, extend the PLL data rate range.

Note: CDR functionality for the receiver is not available when you configure the channel PLL as a CMU PLL—you can use the transceiver channel only as a transmitter.

Figure 1-18: CMU PLL in Cyclone V Devices



The CMU PLL output serial clock, with a frequency that is half of the data rate, feeds the clock divider that resides in the transmitter of the same transceiver channel. The CMU PLLs in channels 1 and 4 feed the x1 and x6 clock lines.

Related Information

Transceiver Clocking in Cyclone V Devices

fPLL as a Transmitter PLL

In addition to CMU PLL, the fPLL located adjacent to the transceiver banks are available for clocking the transmitters for serial data rates up to 3.125 Gbps.

Related Information

Clock Networks and PLLs in Cyclone V Devices

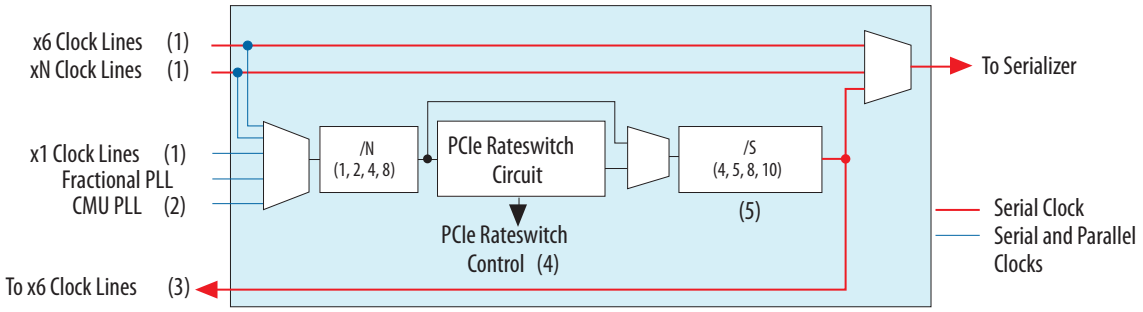
Clock Divider

Each Cyclone V transmitter channel has a clock divider.

There are two types of clock dividers, depending on the channel location in a transceiver bank:

- Local clock divider—channels 0, 2, 3, and 5 provide serial and parallel clocks to the PMA
- Central clock divider—channels 1 and 4 can drive the x6 and xN clock lines

Figure 1-19: Clock Divider for a Transceiver Channel in Cyclone V Devices



Notes:
1. For information about the x1, x6, and xN clock lines, refer to Transceiver Clocking in Cyclone V Devices.
2. Only from the channel PLL in the same transceiver channel configured as a CMU PLL.
3. Applicable for central clock dividers only (clock dividers in channels 1 and 4).
4. The PCIe rateswitch circuit allows dynamic switching between Gen2 and Gen1 line rates in a PCIe Gen2 design.
5. The divider settings are configured automatically depending on the serialization factor. The selected divider setting is half of the serialization factor.

Both types of clock dividers can divide the serial clock input to provide the parallel and serial clocks for the serializer in the channel if you use clocks from the clock lines or transmit PLLs. The central clock divider can additionally drive the x6 clock lines used to bond multiple channels.

In bonded channel configurations, both types of clock dividers can feed the serializer with the parallel and serial clocks directly, without dividing them from the x6 or xN clock lines.

Related Information

Transceiver Clocking in Cyclone V Devices

Calibration Block

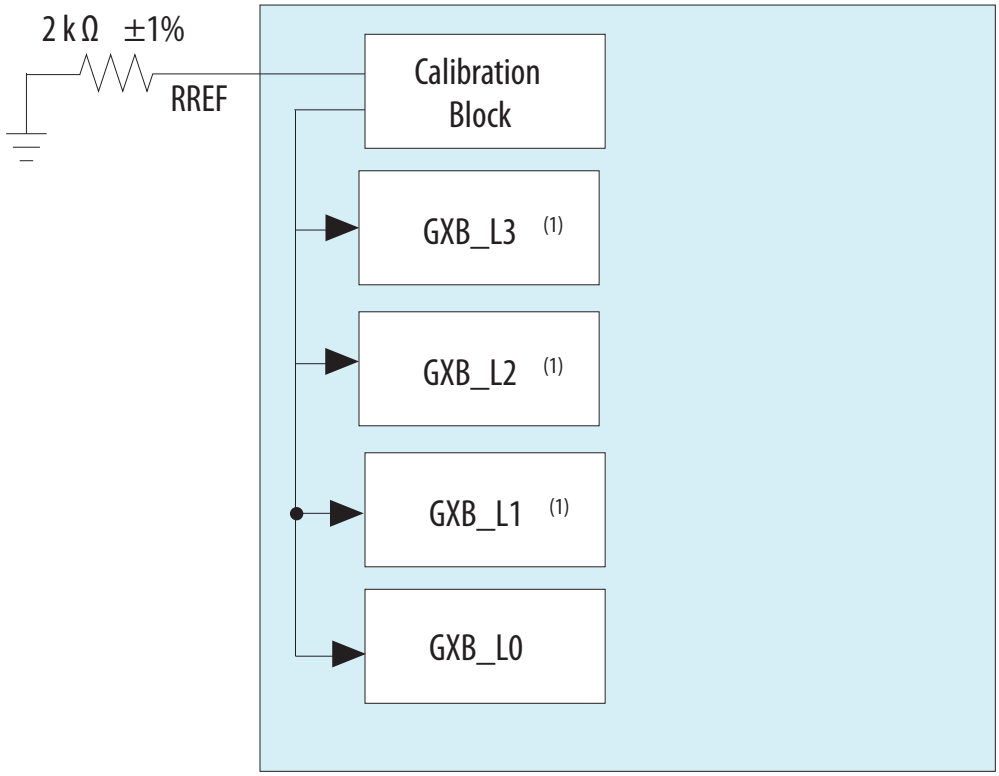
The calibration block calibrates the differential OCT resistance and analog circuitry in the transceiver PMA to ensure the functionality is independent of PVT.

It is also used for duty cycle calibration of the clock line at serial data rates ≥ 4.9152 Gbps.

There is only one calibration block available for the Cyclone V transceiver PMA. It is located on the top left of the device (same side as the transceiver channels).

Figure 1-20: Calibration Block Location and Connections in Cyclone V Devices

Transceivers are on the left side of the device only.



Note:
1. GXB_L1, GXB_L2, and GXB_L3 banks are only available in some device variants.

The calibration block internally generates a constant internal reference voltage, independent of PVT variations. The block uses the internal reference voltage and external reference resistor to generate constant reference currents.

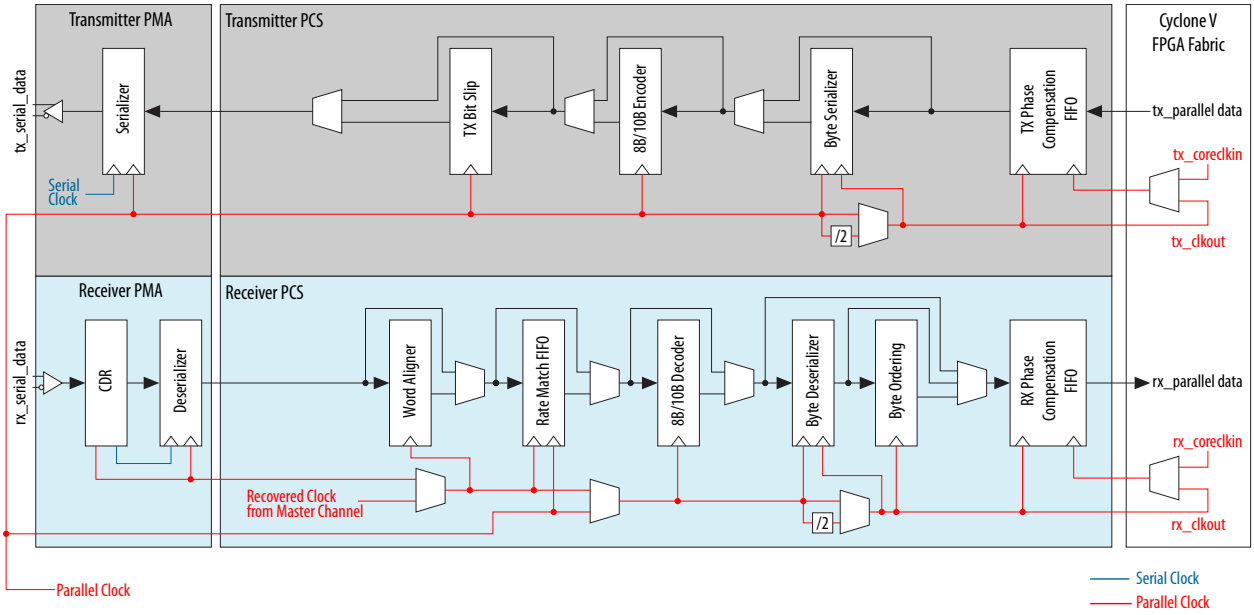
Note: You must connect the external reference resistor to the RREF pin.

These reference currents are used by the analog block calibration circuit to calibrate the transceiver banks. You must connect a separate 2 kΩ (tolerance max ± 1%) external resistor on each RREF pin to ground. To ensure the calibration block operates properly, the RREF resistor connection in the board must be free from external noise.

PCS Architecture

Figure 1-21: PCS Block Diagram of a Transceiver Channel in a Cyclone V Device

The serial and parallel clocks are sourced from the clock divider.



The transceiver channel PCS datapath is categorized into two configurations—single-width and double-width, based on the transceiver channel PMA-PCS width (or serialization/deserialization factor).

Table 1-10: PCS Datapath Configurations

Parameters	Single-Width	Double-Width
PMA–PCS Interface Width	8 or 10 bit	16 or 20 bit
FPGA Fabric–Transceiver Interface Width	8 or 10 bit 16 or 20 bit(2)	16 or 20 bit 32 or 40 bit(2)

(2) The byte serializer and deserializer are enabled.

Transmitter PCS Datapath

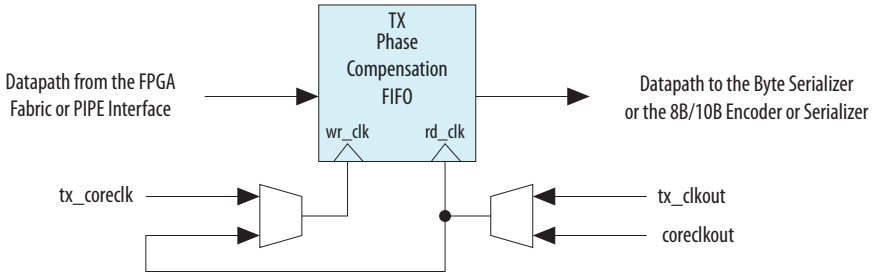
Table 1-11: Blocks in the Transmitter PCS Datapath

Block	Functionality
Transmitter Phase Compensation FIFO	• Compensates for the phase difference between the low-speed parallel clock and the FPGA fabric interface clock, when interfacing the transmitter PCS with the FPGA fabric directly or with the PCIe hard IP block • Supports operation in phase compensation and registered modes
Byte Serializer	• Halves the FPGA fabric–transceiver interface frequency at the transmitter channel by doubling the transmitter input datapath width • Allows the transmitter channel to operate at higher data rates with the FPGA fabric–transceiver interface frequency that is within the maximum limit • Supports operation in single- and double-width modes
8B/10B Encoder	• Generates 10-bit code groups from 8-bit data and the 1-bit control identifier in compliance with Clause 36 of the IEEE 802.3 specification • Supports operation in single- and double-width modes and running disparity control
Transmitter Bit-Slip	• Enables user-controlled, bit-level delay in the data prior to serialization for serial transmission • Supports operation in single- and double-width modes

Transmitter Phase Compensation FIFO

The transmitter phase compensation FIFO is four words deep and interfaces with the transmitter channel PCS and the FPGA fabric or PCIe hard IP block. The transmitter phase compensation FIFO compensates for the phase difference between the low-speed parallel clock and the FPGA fabric interface clock.

Figure 1-22: Transmitter Phase Compensation FIFO



The transmitter phase compensation FIFO supports two operations:

- Phase compensation mode with various clocking modes on the read clock and write clock
- Registered mode with only one clock cycle of datapath latency

Registered Mode

To eliminate the FIFO latency uncertainty for applications with stringent datapath latency uncertainty requirements, bypass the FIFO functionality in registered mode to incur only one clock cycle of datapath latency when interfacing the transmitter channel to the FPGA fabric. Configure the FIFO to registered mode when interfacing the transmitter channel to the FPGA fabric or PCIe hard IP block to reduce datapath latency. In registered mode, the low-speed parallel clock that is used in the transmitter PCS clocks the FIFO.

Phase Compensation Mode

The transmitter phase compensation FIFO compensates for any phase difference between the read and write clocks for the transmitter control and data signals. The low-speed parallel clock feeds the read clock, while the FPGA fabric interface clock feeds the write clock. The clocks must have 0 ppm difference in frequency or a FIFO underrun or overflow condition may result.

The FIFO supports various clocking modes on the read and write clocks depending on the transceiver configuration.

Related Information

Transceiver Clocking in Cyclone V Devices

Byte Serializer

The byte serializer divides the input datapath by two to run the transceiver channel at higher data rates while keeping the FPGA fabric interface frequency within the maximum limit.

The byte serializer supports operation in single- and double-width modes. The datapath clock rate at the output of the byte serializer is twice the FPGA fabric–transmitter interface clock frequency. The byte serializer forwards the least significant word first followed by the most significant word.

Note: You must use the byte serializer in configurations that exceed the maximum frequency limit of the FPGA fabric–transceiver interface.

Byte Serializer in Single-Width Mode

The byte serializer forwards the LSByte first, followed by the MSByte. The input data width to the byte serializer depends on the channel width option. For example, in single-width mode with a channel width of 20 bits, the byte serializer sends out the least significant word `tx_parallel_data[9:0]` of the parallel data from the FPGA fabric, followed by `tx_parallel_data`[19:10].

Table 1-12: Input and Output Data Width of the Byte Serializer in Single-Width Mode for Cyclone V Devices

Mode	Input Data Width to the Byte Serializer	Output Data Width from the Byte Serializer	Byte Serializer Output Ordering
Single-width	16	8	Least significant 8 bits of the 16-bit output first
	20	10	Least significant 10 bits of the 20-bit output first

Byte Serializer in Double-Width Mode

The operation in double-width mode is similar to that of single-width mode. For example, in double-width mode with a channel width of 32 bits, the byte serializer forwards `tx_parallel_data[15:0]` first, followed by `tx_parallel_data[31:16]`.

Table 1-13: Input and Output Data Width of the Byte Serializer in Double-Width Mode for Cyclone V Devices

Mode	Input Data Width to the Byte Serializer	Output Data Width from the Byte Serializer	Byte Serializer Output Ordering
Double-width	32	16	Least significant 16 bits of the 32-bit output first
	40	20	Least significant 20 bits of the 40-bit output first

If you select the **8B/10B Encoder** option, the 8B/10B encoder uses the output from the byte serializer. Otherwise, the byte serializer output is forwarded to the serializer.

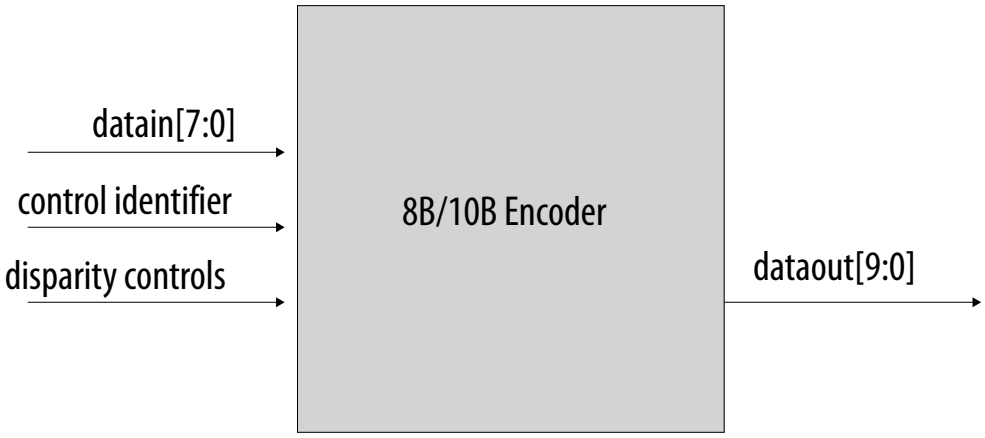
8B/10B Encoder

The 8B/10B encoder supports operation in single- and double-width modes with the running disparity control feature.

8B/10B Encoder in Single-Width Mode

In single-width mode, the 8B/10B encoder generates 10-bit code groups from 8-bit data and 1-bit control identifier with proper disparity according to the PCS reference diagram in the Clause 36 of the IEEE 802.3 specification. The 10-bit code groups are generated as valid data code-groups (/Dx.y/) or special control code-groups (/Kx.y/), depending on the 1-bit control identifier.

Figure 1-23: 8B/10B Encoder in Single-Width Mode

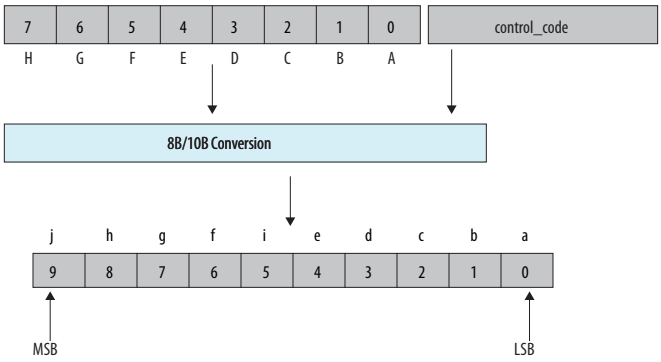


The IEEE 802.3 specification identifies only 12 sets of 8-bit characters as `/Kx.y/`. If other sets of 8-bit characters are set to encode as special control code-groups, the 8B/10B encoder may encode the output 10-bit code as an invalid code (it does not map to a valid /Dx.y/ or /Kx.y/ code), or unintended valid /Dx.y/ code, depending on the value entered.

In single-width mode, the 8B/10B encoder translates the 8-bit data to a 10-bit code group (control word or data word) with proper disparity. If the `tx_datak` input is high, the 8B/10B encoder translates the input `data[7:0]` to a 10-bit control word. If the `tx_datak` input is low, the 8B/10B encoder translates the input `data[7:0]` to a 10-bit data word.

Figure 1-24: 8B/10B Conversion Format

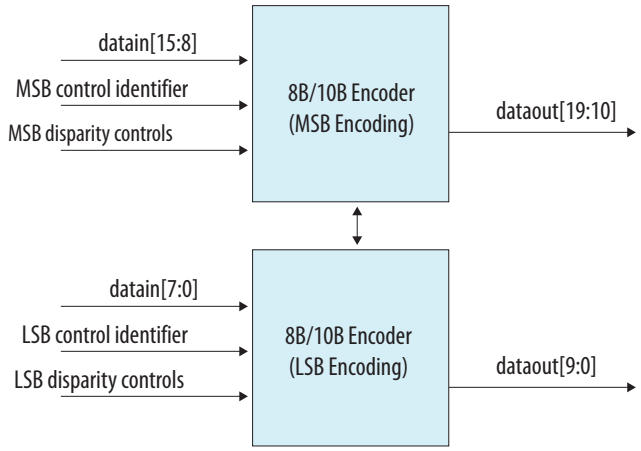
The LSB is transmitted first.



8B/10B Encoder in Double-Width Mode

In double-width mode, two 8B/10B encoders are cascaded to generate two sets of 10-bit code groups from 16-bit data and two 1-bit control identifiers. When receiving the 16-bit data, the 8-bit LSByte is encoded first, followed by the 8-bit MSByte.

Figure 1-25: 8B/10B Encoder in Double-Width Mode



Running Disparity Control

The 8B/10B encoder automatically performs calculations that meet the running disparity rules when generating the 10-bit code groups. The running disparity control feature provides user-controlled signals (`tx_dispval` and `tx_forcedisp`) to manually force encoding into a positive or negative current running disparity code group. When you enable running disparity control, the control overwrites the current running disparity value in the encoder based on the user-controlled signals, regardless of the internally-computed current running disparity in that cycle.

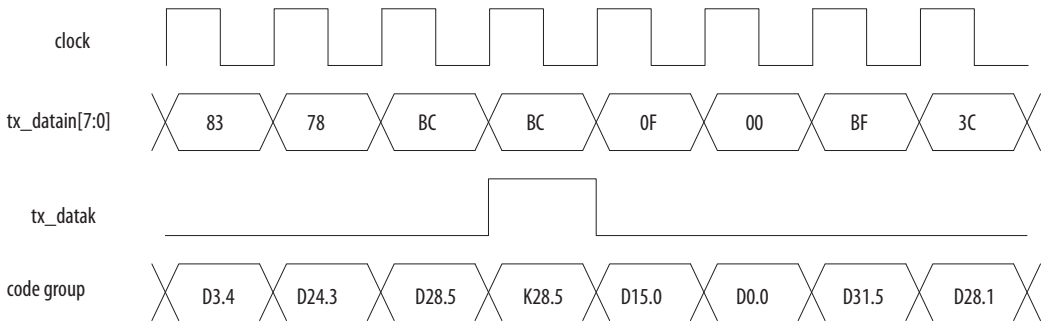
Note: Using running disparity control may temporarily cause a running disparity error at the receiver.

Control Code Encoding

The 8B/10B block provides the `tx_datak` signal to indicate whether the 8-bit data at the `tx_parallel_data` signal should be encoded as a control word (Kx.y) or a data word (Dx.y). When `tx_datak` is low, the 8B/10B encoder block encodes the byte at the `tx_parallel_data` signal as data (Dx.y). When tx_datak is high, the 8B/10B encoder encodes the input data as a Kx.y code group. The rest of the `tx_parallel_data` bytes are encoded as a data word (Dx.y).

Figure 1-26: Control Word and Data Word Transmission

The second 0xBC is encoded as a control word (K28.5).



Note: The IEEE802.3 8B/10B encoder specification identifies only a set of 8-bit characters for which you must assert `tx_datak`. If you assert `tx_datak` for any other set of bytes, the 8B/10B encoder might encode the output 10-bit code as an invalid code (it does not map to a valid Dx.y or Kx.y code), or unintended valid Dx.y code, depending on the value entered. It is possible for a downstream 8B/10B decoder to decode an invalid control word into a valid Dx.y code without asserting code error flags.

Reset Condition

The `reset_tx_digital` signal resets the 8B/10B encoder. During reset, the running disparity and data registers are cleared. Also, the 8B/10B encoder outputs a K28.5 pattern from the RD– column continuously until `reset_tx_digital` is deasserted. The input data and control code from the FPGA fabric is ignored during the reset state. After reset, the 8B/10B encoder starts with a negative disparity (RD–) and transmits three K28.5 code groups for synchronization before it starts encoding and transmitting the data on its output.

Note: While `reset_tx_digital` is asserted, the downstream 8B/10B decoder that receives the data might observe synchronization or disparity errors.

Encoder Output During Reset Sequence

When in reset (`reset_tx_digital` is high), a K28.5- (K28.5 10-bit code group from the RD– column) is sent continuously until `reset_tx_digital` is low. Because of some pipelining of the transmitter channel PCS, some "don't cares" (10'hxxx) are sent before the three synchronizing K28.5 code groups. User data follows the third K28.5 code group.

Figure 1-27: 8B/10B Encoder Output During and After Reset Conditions

8B/10B encoder output during and after reset conditions in both single- and double-width modes.

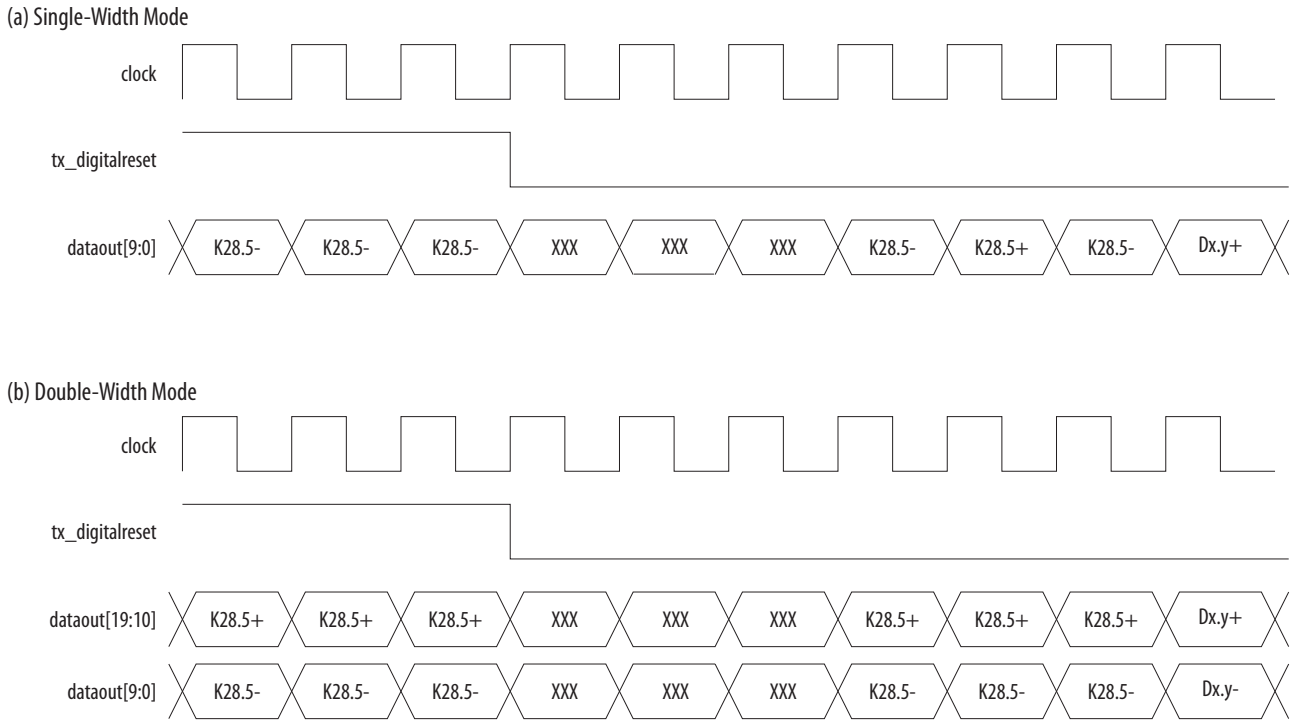


Table 1-14: 8B/10B Encoder Output During and After Reset Conditions

Operation Mode	During 8B/10B Reset	After 8B/10B Reset Release
Single Width	Continuously sends the /K28.5/ code from the RD– column	Some "don't cares" are seen due to pipelining in the transmitter channel, followed by three /K28.5/ codes with proper disparity—starts with negative disparity—before sending encoded 8-bit data at its input.
Double Width	Continuously sends the /K28.5/ code from the RD– column on the LSByte and the /K28.5/ code from the RD+ column on the MSByte	Some "don't cares" are seen due to pipelining in the transmitter channel, followed by: • Three /K28.5/ codes from the RD– column before sending encoded 8-bit data at its input on LSByte. • Three /K28.5/ codes from the RD+ column before sending encoded 8-bit data at its input on MSByte.

Transmitter Bit-Slip

The transmitter bit-slip allows you to compensate for the channel-to-channel skew between multiple transmitter channels by slipping the data sent to the PMA. The maximum number of bits slipped is controlled from the FPGA fabric and is equal to the width of the PMA-PCS minus 1.

Table 1-15: Bits Slip Allowed with the `tx_bitslipboundaryselect` signal

Operation Mode	Maximum Bit-Slip Setting
Single width (8 or 10 bit)	9
Double width (16 or 20 bit)	19

Receiver PCS Datapath

The sub-blocks in the receiver PCS datapath are described in order from the word aligner to the receiver phase compensation FIFO block.

Table 1-16: Blocks in the Receiver PCS Datapath

Block	Functionality
Word Aligner	• Searches for a predefined alignment pattern in the deserialized data to identify the correct boundary and restores the word boundary during link synchronization • Supports an alignment pattern length of 7, 8, 10, 16, 20, or 32 bits • Supports operation in four modes—manual alignment, bit-slip, automatic synchronization state machine, and deterministic latency state machine—in single- and double-width configurations • Supports the optional programmable run-length violation detection, polarity inversion, bit reversal, and byte reversal features
Rate Match FIFO	• Compensates for small clock frequency differences of up to ±300 parts per million (ppm)—600 ppm total—between the upstream transmitter and the local receiver clocks by inserting or deleting skip symbols when necessary • Supports operation that is compliant to the clock rate compensation function in supported protocols
8B/10B Decoder	• Receives 10-bit data and decodes the data into an 8-bit data and a 1-bit control identifier—in compliance with Clause 36 of the IEEE 802.3 specification • Supports operation in single- and double-width modes
Byte Deserializer	• Halves the FPGA fabric–transceiver interface frequency at the receiver channel by doubling the receiver output datapath width • Allows the receiver channel to operate at higher data rates with the FPGA fabric–transceiver interface frequency that is within maximum limit • Supports operation in single- and double-width modes
Byte Ordering	• Searches for a predefined pattern that must be ordered to the LSByte position in the parallel data going to the FPGA fabric when you enable the byte deserializer
Receiver Phase Compensation FIFO	• Compensates for the phase difference between the low-speed parallel clock and the FPGA fabric interface clock when interfacing the receiver PCS with the FPGA fabric directly or with the PCIe hard IP block • Supports operation in phase compensation and registered modes

Word Aligner

Parallel data at the input of the receiver PCS loses the word boundary of the upstream transmitter from the serial-to-parallel conversion in the deserializer. The word aligner receives parallel data from the

deserializer and restores the word boundary based on a pre-defined alignment pattern that must be received during link synchronization.

The word aligner searches for a pre-defined alignment pattern in the deserialized data to identify the correct boundary and restores the word boundary during link synchronization. The alignment pattern is pre-defined for standard serial protocols according to the respective protocol specifications for achieving synchronization. For proprietary protocol implementations, you can specify a custom word alignment pattern specific to your application.

In addition to restoring the word boundary, the word aligner implements the following features:

- Synchronization state machine
- Programmable run length violation detection (for all transceiver configurations)
- Receiver polarity inversion (for all transceiver configurations except PCIe)
- Receiver bit reversal (for custom single- and double-width configurations only)
- Receiver byte reversal (for custom double-width configuration only)

The word aligner operates in one of the following three modes:

- Manual alignment
- Automatic synchronization state machine
- Bit-slip
- Deterministic latency state machine

Except for bit-slip mode, after completing word alignment, the deserialized data is synchronized to have the word alignment pattern at the LSB portion of the aligned data.

When the 8B/10B encoder/decoder is enabled, the word aligner detects both positive and negative disparities of the alignment pattern. For example, if you specify a /K28.5/ (b'0011111010) pattern as the comma, `rx_patterndetect` is asserted if b'0011111010 or b'1100000101 is detected in the incoming data.

Word Aligner Options and Behaviors

The operation mode and alignment pattern length support varies depending on the word aligner configurations.

Table 1-17: Word Aligner Options and Behaviors

PMA-PCS Interface Width (bits)	Word Alignment Mode	Word Alignment Pattern Length (bits)	Word Alignment Behavior
8	Manual Alignment	16	User-controlled signal starts the alignment process. Alignment happens once unless the signal is reasserted.
	Bit-Slip	16	User-controlled signal shifts data one bit at a time.

PMA-PCS Interface Width (bits)	Word Alignment Mode	Word Alignment Pattern Length (bits)	Word Alignment Behavior
10	Manual Alignment	7 and 10	User-controlled signal starts the alignment process. Alignment happens once unless the signal is reasserted.
	Bit-Slip	7 and 10	User-controlled signal shifts data one bit at a time.
	Automatic Synchronized State Machine	7 and 10	Data is required to be 8B/10B encoded. Aligns to selected word aligner pattern when pre-defined conditions are satisfied.
	Deterministic Latency State Machine	10	User-controlled signal starts the alignment process. After the pattern is found and the word boundary is identified, the state machine controls the deserializer to clock-slip the boundary-indicated number of serial bits.
16	Manual Alignment	8, 16, and 32	Alignment happens automatically after RX PCS reset. User-controlled signal starts the alignment process thereafter. Alignment happens once unless the signal is reasserted.
	Bit-Slip	8, 16, and 32	User-controlled signal shifts data one bit at a time.
20	Manual Alignment	7, 10, and 20	Alignment happens automatically after RX PCS reset. User-controlled signal starts the alignment process thereafter. Alignment happens once unless the signal is reasserted.
	Bit-Slip	7, 10, and 20	User-controlled signal shifts data one bit at a time.
	Deterministic Latency State Machine	10 and 20	User-controlled signal starts the alignment process. After the pattern is found and the word boundary is identified, the state machine controls the deserializer to clock-slip the boundary-indicated number of serial bits.

Word Aligner in Manual Alignment Mode

In manual alignment mode, the word alignment operation is manually controlled with the `rx_std_wa_patternalign` input signal or the `rx_enapatternalign` register.

Depending on the configuration, controlling the `rx_std_wa_patternalign` signal enables the word aligner to look for the predefined word alignment pattern in the received data stream. A value 1 at the `rx_patterndetect` register indicates that the word alignment pattern is detected . A value 1 at the `rx_syncstatus` register indicates that the word aligner has successfully synchronized to the new word boundary.

Manual word alignment can be also triggered by writing a value 1 to `rx_enapatternalign` register. The word alignment is triggered in the next parallel clock cycle when a 0 to 1 transition occurs on the `rx_enapatternalign` register.

After `rx_syncstatus` is asserted and if the incoming data is corrupted causing an invalid code group, `rx_syncstatus` remains asserted. The `rx_errdetect` register will be set to 1 (indicating RX 8B/10B error detected). When this happens, the manual alignment mode is not be able to de-assert the `rx_syncstatus` signal. You must manually assert `rx_digitalreset` or manually control `rx_std_wa_patternalign` to resynchronize a new word boundary search whenever `rx_errdetect` shows an error.

Table 1-18: Word Aligner Operations in Manual Alignment Mode

PCS Mode	PMA–PCS Interface Width (bits)	Word Alignment Operation
Single Width	8	1. After the `rx_digitalreset` signal deasserts, a 0-to-1 transition on the `rx_enapatternalign` register triggers the word aligner to look for the predefined word alignment pattern in the received data stream and automatically synchronize to the new word boundary. 2. Any alignment pattern found thereafter in a different word boundary does not cause the word aligner to resynchronize to this new word boundary because there is a lack of a preceding 0-to-1 transition on the `rx_enapatternalign` register. 3. To resynchronize to the new word boundary, create a 0-to-1 transition in the `rx_enapatternalign` register. 4. If you set the `rx_enapatternalign` register to 1 before the deassertion of the `rx_digitalreset` signal, the word aligner updates the word boundary when the first alignment pattern is found, even though a 0-to-1 transition was not explicitly generated. 5. To resynchronize to the new word boundary, create a 0-to-1 transition to the `rx_enapatternalign` register. 6. When the word aligner synchronized to the new word boundary, the `rx_patterndetect` and `rx_syncstatus` signals will assert for one parallel clock cycle.
	10	1. After the `rx_digitalreset` signal deasserts, setting the `rx_enapatternalign` register to 1 triggers the word aligner to look for the predefined word alignment pattern, or its complement in the received data stream, and automatically synchronize to the new word boundary. 2. Any alignment pattern found thereafter in a different word boundary causes the word aligner to resynchronize to this new word boundary if the `rx_enapatternalign` register remains set to 1. 3. If you set the `rx_enapatteralign` register to 0, the word aligner maintains the current word boundary even when it finds the alignment pattern in a new word boundary. 4. When the word aligner synchronized to the new word boundary, the `rx_patterndetect` and `rx_syncstatus` signals will assert for one parallel clock cycle.

PCS Mode	PMA–PCS Interface Width (bits)	Word Alignment Operation
Double Width	16 20	1. After the rx_digitalreset signal deasserts, regardless of the setting in the rx_enapatternalign register, the word aligner synchronizes to the first predefined alignment pattern found. 2. Any alignment pattern found thereafter in a different word boundary does not cause the word aligner to resynchronize to this new word boundary. 3. To resynchronize to the new word boundary, create a 0-to-1 transition in the rx_enapatternalign register. 4. When the word aligner synchronized to the new word boundary, the rx_patterndetect and rx_syncstatus signals will assert for one parallel clock cycle. The rx_syncstatus signal will deassert if the next rising edge of rx_enapatternalign is detected.

Word Aligner in Bit-Slip Mode

In bit-slip mode, the word aligner is controlled by the rx_bitslip bit of the pcs8g_rx_wa_control register. At every 0-1 transition of the rx_bitslip bit of the pcs8g_rx_wa_control register, the bit-slip circuitry slips one bit into the received data stream, effectively shifting the word boundary by one bit. Also in bit-slip mode, the word aligner pcs8g_rx_wa_status register bit for rx_patterndetect is driven high for one parallel clock cycle when the received data after bit-slipping matches the 16-bit word alignment pattern programmed.

To achieve word alignment, you can implement a bit-slip controller in the FPGA fabric that monitors the rx_parallel_data signal, the rx_patterndetect signal, or both, and controls them with the rx_bitslip signal.

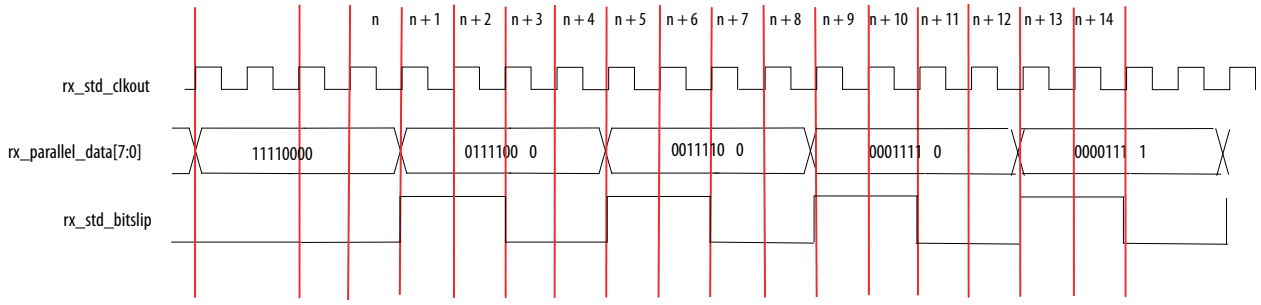
Table 1-19: Word Aligner in Bit-Slip Mode

PCS Mode	PMA–PCS Interface Width (bits)	Word Alignment Operation
Single Width	8 10	1. At every rising edge to the rx_bitslip signal, the word aligner slips one bit into the received data. 2. When bit-slipping shifts a complete round of the data bus width, the word boundary is back to the original boundary. 3. Check the received data, rx_parallel_data after every bit slip operation whether the predefined word alignment pattern is visible in the new word boundary. When the word alignment pattern is visible in the new word boundary, the alignment process is completed and the bit-slip operation can be stopped.
Double Width	16 20	

Note: For every bit slipped in the word aligner, the earliest bit received is lost.

For this example, consider that 8'b11110000 is received back-to-back and 16'b0000111100011110 is the predefined word alignment pattern. A rising edge on the `rx_std_bitslip` signal at time n + 1 slips a single bit 0 at the MSB position, forcing the `rx_parallel_data` to 8'b01111000. Another rising edge on the `rx_std_bitslip` signal at time n + 5 forces `rx_parallel_data` to 8'b00111100. Another rising edge on the `rx_std_bitslip` signal at time n + 9 forces `rx_parallel_data` to 8'b00011110. Another rising edge on the `rx_std_bitslip` signal at time n + 13 forces the `rx_parallel_data` to 8'b00001111. At this instance, `rx_parallel_data` in cycles n + 12 and n + 13 is 8'b00011110 and 8'b00001111, respectively, which matches the specified 16-bit alignment pattern 16'b0000111100011110.

Figure 1-28: Word Aligner Configured in Bit Slip Mode



Note: Bit slip operation can also be triggered by a 0 to 1 transition in the `rx_bitslip` register.

Word Aligner in Automatic Synchronization State Machine Mode

In automatic synchronization state machine mode, a programmable state machine determines the moment that the word aligner has either achieved synchronization or lost synchronization.

You can configure the state machine to provide hysteresis control during link synchronization and throughout normal link operation. Depending on your protocol configurations, the state machine parameters are automatically configured so they are compliant with the synchronization state machine in the respective protocol specification.

Table 1-20: Programmable Parameters for the Word Aligner in Synchronization State Machine Mode

Parameter	Values
Number of valid synchronization code groups or ordered sets received to achieve synchronization	1–256
Number of erroneous code groups received to lose synchronization	1–64
Number of continuous good code groups received to reduce the error count by one	1–256

Table 1-21: Word Aligner Operation in Automatic Synchronization State Machine Mode

PCS Mode	PMA–PCS Interface Width	Word Alignment Operation
Single Width	10 bits	1. After the `rx_digitalreset` signal deasserts, the word aligner starts looking for the predefined word alignment pattern, or its complement, in the received data stream and automatically aligns to the new word boundary. 2. Synchronization is achieved only after the word aligner receives the programmed number of valid synchronization code groups in the same word boundary and is indicated with the assertion of the `rx_syncstatus` signal. 3. After assertion and achieving synchronization, the `rx_syncstatus` signal remains asserted until the word aligner loses synchronization. 4. Loss of synchronization occurs when the word aligner receives the programmed number of erroneous code groups without receiving the intermediate good code groups and is indicated with the deassertion of the `rx_syncstatus` signal. 5. The word aligner may achieve synchronization again after receiving a new programmed number of valid synchronization code groups in the same word boundary.

Word Aligner in Automatic Synchronization State Machine Mode with a 10-Bit PMA-PCS Interface Configuration

Protocols such as PCIe require the receiver PCS logic to implement a synchronization state machine to provide hysteresis during link synchronization. Each of these protocols defines a specific number of synchronization code groups that the link must receive to acquire synchronization and a specific number of erroneous code groups that it must receive to fall out of synchronization.

In PCIe configurations, the word aligner in automatic synchronization state machine mode automatically selects the word alignment pattern length and pattern as specified by each protocol. The synchronization state machine parameters are fixed for PCIe configurations as specified by the respective protocol.

Table 1-22: Word Aligner in Synchronization State Machine Modes for a PCIe Configuration

Mode	PCIe
Number of valid synchronization code groups or ordered sets received to achieve synchronization	4
Number of erroneous code groups received to lose synchronization	17
Number of continuous good code groups received to reduce the error count by one	16

After deassertion of the `reset_rx_digital` signal in automatic synchronization state machine mode, the word aligner starts looking for the word alignment pattern or synchronization code groups in the received data stream. When the programmed number of valid synchronization code groups or ordered sets is

received, the rx_syncstatus status bit is driven high to indicate that synchronization is acquired. The rx_syncstatus status bit is constantly driven high until the programmed number of erroneous code groups is received without receiving intermediate good groups; after which rx_syncstatus is driven low. The word aligner indicates loss of synchronization (rx_syncstatus remains low) until the programmed number of valid synchronization code groups are received again.

Word Aligner Operations in Deterministic Latency State Machine Mode

In deterministic latency state machine mode, word alignment is achieved by performing a clock-slip in the deserializer until the deserialized data coming into the receiver PCS is word-aligned. The state machine controls the clock-slip process in the deserializer after the word aligner has found the alignment pattern and identified the word boundary. Deterministic latency state machine mode offers a reduced latency uncertainty in the word alignment operation for applications that require deterministic latency.

After rx_syncstatus is asserted and if the incoming data is corrupted causing an invalid code group, rx_syncstatus remains asserted. The rx_errdetect register will be set to 1 (indicating RX 8B/10B error detected). When this happens, the manual alignment mode is not be able to de-assert the rx_syncstatus signal. You must manually assert rx_digitalreset or manually control rx_std_wa_patternalign to resynchronize a new word boundary search whenever rx_errdetect shows an error.

Table 1-23: Word Aligner Operations in Deterministic Latency State Machine Mode

PCS Mode	PMA–PCS Interface Width	Word Alignment Operation
Single Width	10 bits	1. After `rx_digitalreset` deasserts, the word aligner starts looking for the predefined word alignment pattern, or its complement, in the received data stream and automatically aligns to the new word boundary. 2. After the pattern is found and the word boundary is identified, the state machine controls the clock-slip process in the deserializer. 3. When the clock-slip is complete, the deserialized data coming into the receiver PCS is word-aligned and is indicated by the value 1 in the `rx_syncstatus` register until `rx_digitalreset` is asserted. 4. To resynchronize to the new word boundary, the Avalon-MM register `rx_enapatternalign` (not available as a signal) must be reasserted to initiate another pattern alignment. Asserting `rx_enapatternalign` may cause the extra shifting in the RX datapath if `rx_enablepatternalign` is asserted while bit slipping is in progress. Consequently, `rx_enapatternalign` should only be asserted under the following conditions: • `rx_syncstatus` is asserted • `rx_bitslipboundaryselectout` changes from a non-zero value to zero or 1 5. When the word aligner synchronizes to the new word boundary, `rx_syncstatus` has a value of 1 until `rx_digitalreset` is deasserted or `rx_enapatternalign` is set to 1. `rx_patterndetect` has a value of 1 whenever a word alignment pattern is found for one parallel clock cycle regardless of whether or not the word aligner is triggered to align to the new word boundary.
Double Width	20 bits	

Programmable Run-Length Violation Detection

The programmable run-length violation detection circuit resides in the word aligner block and detects if consecutive 1s or 0s in the received data exceed the user-specified threshold.

If the data stream exceeds the preset maximum number of consecutive 1s or 0s, the violation is signified by the assertion of the `rx_rlv` status bit.

Table 1-24: Detection Capabilities of the Run-Length Violation Circuit

PCS Mode	PMA–PCS Interface Width (bits)	Run-Length Violation Detector Range	
		Minimum	Maximum
Single Width	8	4	128
	10	5	160
Double Width	16	8	512
	20	10	640

Receiver Polarity Inversion

The positive and negative signals of a serial differential link might erroneously be swapped during board layout. Solutions such as board re-spin or major updates to the PLD logic can be expensive. The polarity inversion feature at the receiver corrects the swapped signal error without requiring board re-spin or major updates to the logic in the FPGA fabric. The polarity inversion feature inverts the polarity of every bit at the input to the word aligner, which has the same effect as swapping the positive and negative signals of the serial differential link.

Inversion is controlled dynamically with the `rx_invpolarity` register. When you enable the polarity inversion feature, initial disparity errors may occur at the receiver with the 8B/10B-coded data. The receiver must be able to tolerate these disparity errors.

Caution: If you enable polarity inversion midway through a word, the word will be corrupted.

Bit Reversal

By default, the receiver assumes a LSB-to-MSB transmission. If the transmission order is MSB-to-LSB, the receiver forwards the bit-flipped version of the parallel data to the FPGA fabric on `rx_parallel_data`. To reverse the bit order at the output of the word aligner to receive a MSB-to-LSB transmission, use the bit reversal feature at the receiver.

Table 1-25: Bit Reversal Feature

Bit Reversal Option	Received Bit Order	
	Single-Width Mode (8 or 10 bit)	Double-Width Mode (16 or 20 bit)
Disabled (default)	LSB to MSB	LSB to MSB

Enabled	MSB to LSB
	For example:
	`8-bit—D[7:0] rewired to D[0:7]`
	`10-bit—D[9:0] rewired to D[0:9]`
		MSB to LSB
		For example:
		`16-bit—D[15:0] rewired to D[0:15]`
		`20-bit—D[19:0] rewired to D[0:19]`

Note: When receiving the MSB-to-LSB transmission, the word aligner receives the data in reverse order. The word alignment pattern must be reversed accordingly to match the MSB first incoming data ordering.

You can dynamically control the bit reversal feature to use the `rx_bitreversal_enable` register with the word aligner in bit-slip mode. When you dynamically enable the bit reversal feature in bit-slip mode, ignore the pattern detection function in the word aligner because the word alignment pattern cannot be dynamically reversed to match the MSB first incoming data order.

Receiver Byte Reversal

In double-width mode, two symbols of incoming data at the receiver may be accidentally swapped during transmission. For a 16-bit input data width at the word aligner, the two symbols are bits`[15:8]` and bits`[7:0]`. For a 20-bit input data width at the word aligner, the two symbols are bits`[19:10]` and bits`[9:0]`. The byte reversal feature at the word aligner output corrects the swapped signal error by swapping the two symbols in double-width mode at the word aligner output, as listed in **Table 1-26**.

Table 1-26: Byte Reversal Feature

Byte Reversal Option	Word Aligner Output	
	16-bit Data Width	20-bit Data Width
Disabled	`D[15:0]`	`D[19:0]`
Enabled	`D[7:0], D[15:8]`	`D[9:0], D[19:10]`

The reversal is controlled dynamically using the `rx_bytereversal_enable` register, and when you enable the receiver byte reversal option, this may cause initial disparity errors at the receiver with 8B/10B-coded data. The receiver must be able to tolerate these disparity errors.

Note: When receiving swapped symbols, the word alignment pattern must be byte-reversed accordingly to match the incoming byte-reversed data.

Rate Match FIFO

The Rate Match FIFO compensates for the small clock frequency differences between the upstream transmitter and the local receiver clocks.

In a link where the upstream transmitter and local receiver can be clocked with independent reference clock sources, the data can be corrupted by any frequency differences (in ppm count) when crossing the data from the recovered clock domain—the same clock domain as the upstream transmitter reference clock—to the local receiver reference clock domain.

The rate match FIFO is 20 words deep, which compensates for the small clock frequency differences of up to ±300 ppm (600 ppm total) between the upstream transmitter and the local receiver clocks by performing symbol insertion or deletion, depending on the ppm difference on the clocks.

The rate match FIFO requires that the transceiver channel is in duplex configuration (both transmit and receive functions) and has a predefined 20-bit pattern (that consists of a 10-bit control pattern and a 10-bit skip pattern). The 10-bit skip pattern must be chosen from a code group with neutral disparity.

The rate match FIFO operates by looking for the 10-bit control pattern, followed by the 10-bit skip pattern in the data, after the word aligner has restored the word boundary. After finding the pattern, the rate match FIFO performs the following operations to ensure the FIFO does not underflow or overflow:

- Inserts the 10-bit skip pattern when the local receiver reference clock frequency is greater than the upstream transmitter reference clock frequency
- Deletes the 10-bit skip pattern when the local receiver reference clock frequency is less than the upstream transmitter reference clock frequency

The rate match FIFO supports operations in single-width mode. The 20-bit pattern can be user-defined for custom configurations. For protocol configurations, the rate match FIFO is automatically configured to support a clock rate compensation function as required by the following specifications:

- The PCIe protocol per clock tolerance compensation requirement, as specified in the PCI Express Base Specification 2.0 for Gen1 and Gen2 signaling rates
- The Gbps Ethernet (GbE) protocol per clock rate compensation requirement using an idle ordered set, as specified in Clause 36 of the IEEE 802.3 specification

In asynchronous systems, use independent reference clocks to clock the upstream transmitter and local receiver. Frequency differences in the order of a few hundred ppm can corrupt the data when latching from the recovered clock domain (the same clock domain as the upstream transmitter reference clock) to the local receiver reference clock domain.

The rate match FIFO deletes SKP symbols or ordered sets when the upstream transmitter reference clock frequency is higher than the local receiver reference clock frequency and inserts SKP symbols or ordered sets when the local receiver reference clock frequency is higher than the upstream transmitter reference clock frequency.

Note: For the Gigabit Ethernet protocol, if you enabled rate match FIFO in the autonegotiation state machine in an FPGA core, refer to the "Rate Match FIFO" section in the "Gigabit Ethernet" section in the *Transceiver Protocol Configurations in Cyclone V Devices* chapter.

Related Information

- **Transceiver Custom Configurations in Cyclone V Devices**
- **Transceiver Protocol Configurations in Cyclone V Devices**

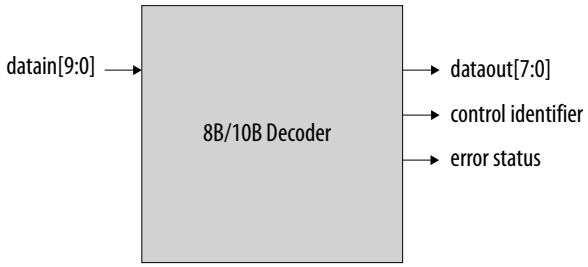
8B/10B Decoder

The receiver channel PCS datapath implements the 8B/10B decoder after the rate match FIFO. In configurations with the rate match FIFO enabled, the 8B/10B decoder receives data from the rate match FIFO. In configurations with the rate match FIFO disabled, the 8B/10B decoder receives data from the word aligner. The 8B/10B decoder supports operation in single- and double-width modes.

8B/10B Decoder in Single-Width Mode

In single-width mode, the 8B/10B decoder decodes the received 10-bit code groups into an 8-bit data and a 1-bit control identifier, in compliance with Clause 36 in the IEEE 802.3 specification. The 1-bit control identifier indicates if the decoded 8-bit code is a valid data or special control code. The decoded data is fed to the byte deserializer or the receiver phase compensation FIFO (if the byte deserializer is disabled).

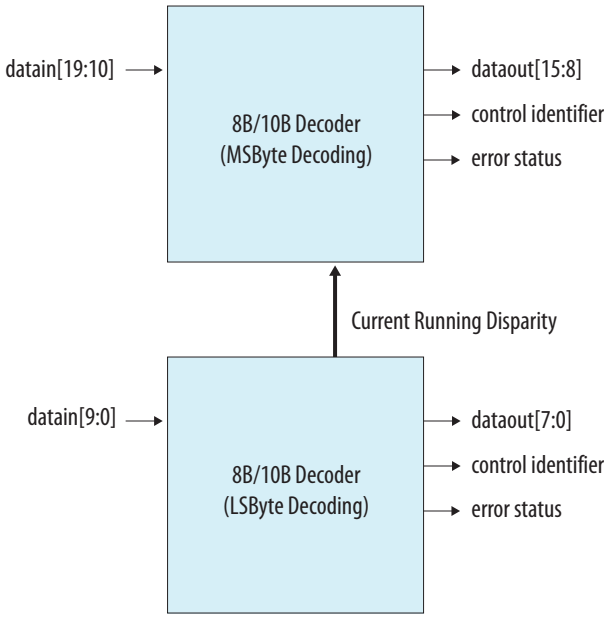
Figure 1-29: 8B/10B Decoder in Single-Width Mode



8B/10B Decoder in Double-Width Mode

In double-width mode, two 8B/10B decoders are cascaded to decode the 20-bit code groups into two sets of 8-bit data and two 1-bit control identifiers. When receiving the 20-bit code group, the 10-bit LSByte is decoded first and the ending running disparity is forwarded to the other 8B/10B decoder for decoding the 10-bit MSByte.

Figure 1-30: 8B/10B Decoder in Double-Width Mode



Control Code Group Detection

The 8B/10B decoder indicates whether the decoded 8-bit code group is a data or a control code group on the `rx_datak` signal. If the received 10-bit code group is one of the 12 control code groups (`/Kx.y/`) specified in the IEEE802.3 specification, the `rx_datak` signal is driven high. If the received 10-bit code group is a data code group (`/Dx.y/`), the `rx_datak` signal is driven low.

Byte Deserializer

The FPGA fabric-transceiver interface frequency has an upper limit. In configurations that have a receiver PCS frequency greater than the upper limit stated, the parallel received data and status signals cannot be

forwarded directly to the FPGA fabric because it violates this upper limit for the FPGA fabric-transceiver interface frequency. In such configurations, the byte deserializer is required to reduce the FPGA fabric-transceiver interface frequency to half while doubling the parallel data width.

Note: The byte deserializer is required in configurations that exceed the FPGA fabric-transceiver interface clock upper frequency limit. It is optional in configurations that do not exceed the FPGA fabric-transceiver interface clock upper frequency limit.

The byte deserializer supports operation in single- and double-width modes. The datapath clock rate at the input of the byte deserializer is twice the FPGA fabric–receiver interface clock frequency. After byte deserialization, the word alignment pattern may be ordered in the MSByte or LSByte position.

The data is assumed to be received as LSByte first—the least significant 8 or 10 bits in single-width mode or the least significant 16 or 20 bits in double-width mode.

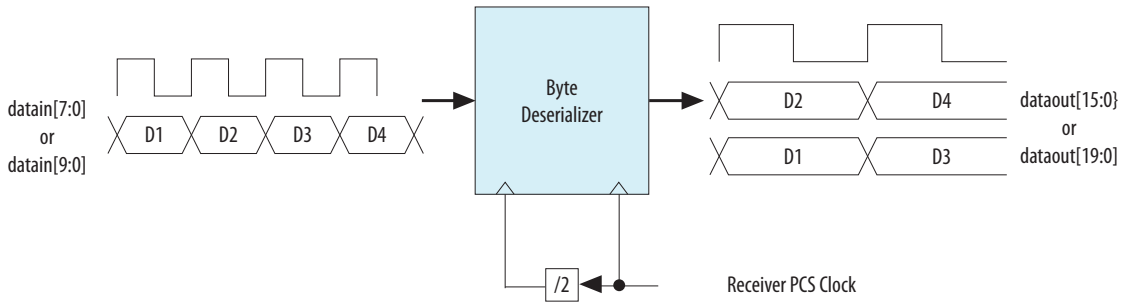
Table 1-27: Byte Deserializer Input Datapath Width Conversion

Mode	Byte Deserializer Input Datapath Width	Receiver Output Datapath Width
Single Width	8	16
	10	20
Double Width	16	32
	20	40

Byte Deserializer in Single-Width Mode

In single-width mode, the byte deserializer receives 8-bit wide data from the 8B/10B decoder or 10-bit wide data from the word aligner (if the 8B/10B decoder is disabled) and deserializes it into 16- or 20-bit wide data at half the speed.

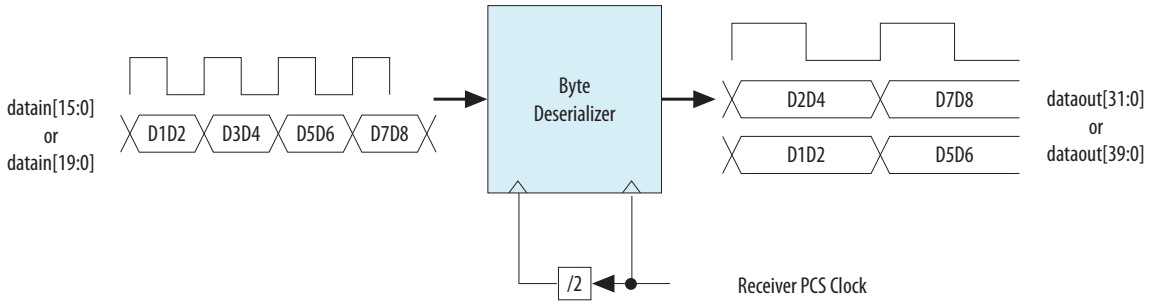
Figure 1-31: Byte Deserializer in Single-Width Mode



Byte Deserializer in Double-Width Mode

In double-width mode, the byte deserializer receives 16-bit wide data from the 8B/10B decoder or 20-bit wide data from the word aligner (if the 8B/10B decoder is disabled) and deserializes it into 32- or 40-bit wide data at half the speed.

Figure 1-32: Byte Deserializer in Double-Width Mode



Byte Ordering

When you enable the byte deserializer, the output byte order may not match the originally transmitted ordering. For applications that require a specific pattern to be ordered at the LSByte position of the data, byte ordering restores the proper byte order of the byte-deserialized data before forwarding it to the FPGA fabric.

Byte ordering operates by inserting a predefined pad pattern to the byte-deserialized data if the predefined byte ordering pattern found is not in the LSByte position.

Byte ordering requires the following:

- A receiver with the byte deserializer enabled
- A predefined byte ordering pattern that must be ordered at the LSByte position of the data
- A predefined pad pattern

Byte ordering supports operation in single- and double-width modes. Both modes support operation in word aligner-based and manual ordering modes.

Byte Ordering in Single-Width Mode

Byte ordering is supported only when you enable the byte deserializer.

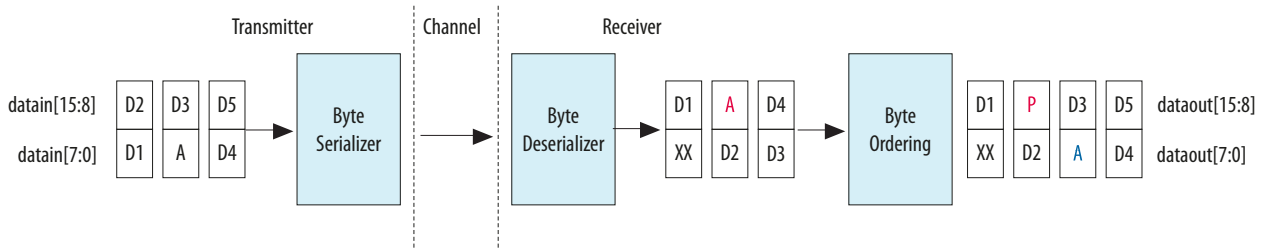
Table 1-28: Byte Ordering Operation in Single-Width Mode

PMA–PCS Interface Width	FPGA Fabric–Transceiver Interface Width	8B/10B Decoder	Byte Ordering Pattern Length	Pad Pattern Length
8 bits	16 bits	Disabled	8 bits	8 bits
10 bits	16 bits	Enabled	9 bits(3)	9 bits (3)
	20 bits	Disabled	10 bits	10 bits

(3) The MSB of the 9-bit pattern represents the 1-bit control identifier of the 8B/10B-decoded data. The lower 8 bits represent the 8-bit decoded code.

Figure 1-33: Byte Ordering Operation Example in Single-Width Mode

An example of a byte ordering operation in single-width mode (8-bit PMA-PCS interface width) where A is the predefined byte ordering pattern and P is the predefined pad pattern.



Byte Ordering in Double-Width Mode

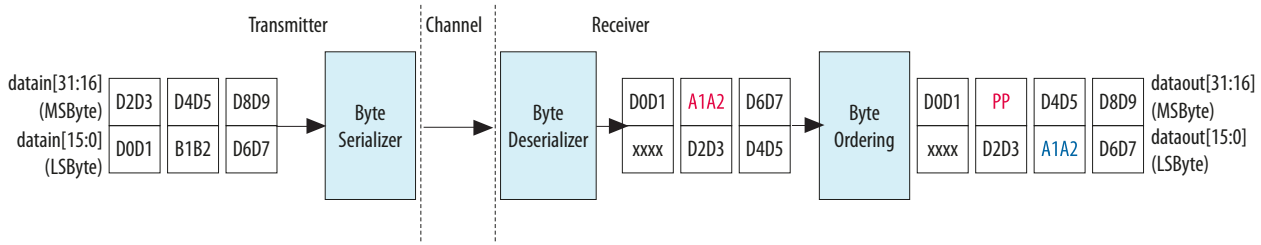
Byte ordering is supported only when you enable the byte deserializer.

Table 1-29: Byte Ordering Operation in Double-Width Mode

PMA–PCS Interface Width	FPGA Fabric–Transceiver Interface Width	8B/10B Decoder	Byte Ordering Pattern Length	Pad Pattern Length
16 bits	32 bits	Disabled	8 or 16 bits	8 bits
20 bits	32 bits	Enabled	9(4) or 18 bits(5)	9 bits(4)
	40 bits	Disabled	10 or 20 bits	10 bits

Figure 1-34: Byte Ordering Operation Example in Double-Width Mode

An example of a byte ordering operation in double-width mode (16-bit PMA-PCS interface width) where A1A2 is the predefined byte ordering pattern and P is the predefined pad pattern.



Word Aligner-Based Ordering Mode

In word aligner-based ordering mode, the byte ordering operation is controlled by the word aligner synchronization status signal, `rx_syncstatus`.

(4) The MSB of the 9-bit pattern represents the 1-bit control identifier of the 8B/10B-decoded data. The lower 8 bits represent the 8-bit decoded code.

(5) The 18-bit pattern consists of two sets of 9-bit patterns, individually represented as in the previous note.

After a rising edge on the rx_syncstatus signal, byte ordering looks for the byte ordering pattern in the byte-deserialized data.

When the first data byte that matches the byte ordering pattern is found, the byte ordering performs the following operations:

- If the pattern is not in the LSByte position—byte ordering inserts the appropriate number of pad patterns to push the byte ordering pattern to the LSByte position and indicates the byte alignment.
- If the pattern is in the LSByte position—byte ordering indicates the byte alignment.

Any byte misalignment found thereafter is ignored unless another rising edge on the rx_syncstatus signal, indicating resynchronization, is observed.

Manual Ordering Mode

In manual ordering mode, the byte ordering operation is controlled using the rx_enabyteord signal.

A rising edge on the rx_enabyteord signal triggers byte ordering to look for the byte ordering pattern in the byte-deserialized data.

When the first data byte that matches the byte ordering pattern is found, the byte ordering performs the following operations:

- If the pattern is not in the LSByte position—byte ordering inserts the appropriate number of pad patterns to push the byte ordering pattern to the LSByte position and indicates the byte alignment.
- If the pattern is in the LSByte position—byte ordering indicates the byte alignment.

Any byte misalignment found thereafter is ignored unless another rising edge on the rx_enabyteord signal is observed.

Receiver Phase Compensation FIFO

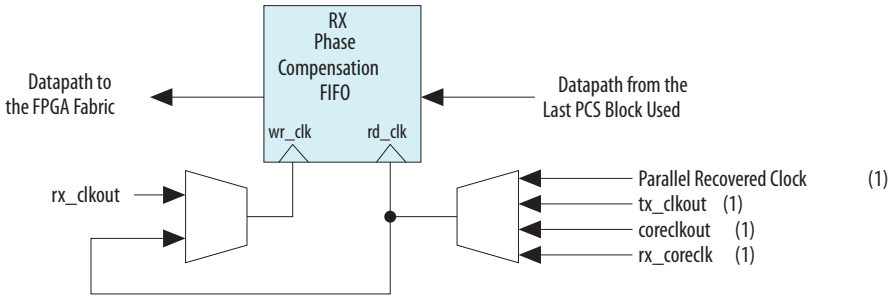
The receiver phase compensation FIFO is four words deep and interfaces the status and data signals between the receiver PCS and the FPGA fabric or the PCIe hard IP block.

The low-speed parallel clock feeds the write clock, while the FPGA fabric interface clock feeds the read clock. The clocks must have 0 ppm difference in frequency or a receiver phase compensation FIFO underrun or overflow condition may result.

The FIFO supports the following operations:

- Phase compensation mode with various clocking modes on the read clock and write clock
- Registered mode with only one clock cycle of datapath latency

Figure 1-35: Receiver Phase Compensation FIFO



Note:
1. These clocks may have been divided by 2 if you used a byte deserializer.

Related Information

Transceiver Clocking in Cyclone V Devices.

Registered Mode

To eliminate the FIFO latency uncertainty for applications with stringent datapath latency uncertainty requirements, bypass the FIFO functionality in registered mode to incur only one clock cycle of datapath latency when interfacing the transmitter channel to the FPGA fabric. Configure the FIFO to registered mode when interfacing the transmitter channel to the FPGA fabric or PCIe hard IP block to reduce datapath latency. In registered mode, the low-speed parallel clock that is used in the transmitter PCS clocks the FIFO.

Channel Bonding

The high-speed serial clock and low-speed parallel clock skew between channels and unequal latency in the transmitter phase compensation FIFO contribute to transmitter channel-to-channel skew. Bonded transmitter datapath clocking provides low channel-to-channel skew when compared with non-bonded channel configurations.

- Bonded channel configurations—the serial clock and parallel clock for all bonded channels are generated by the transmit PLL and central clock divider, resulting in lower channel-to-channel clock skew.

  The transmitter phase compensation FIFO in all bonded channels share common pointers and control logic generated in the central clock divider, resulting in equal latency in the transmitter phase compensation FIFO of all bonded channels. The lower transceiver clock skew and equal latency in the transmitter phase compensation FIFOs in all channels provide lower channel-to-channel skew in bonded channel configurations.

- Non-bonded channel configurations—the parallel clock in each channel are generated independently by its local clock divider, resulting in higher channel-to-channel clock skew.

  The transmitter phase compensation FIFO in each non-bonded channel has its own pointers and control logic that can result in unequal latency in the transmitter phase compensation FIFO of each channel. The higher transceiver clock skew and unequal latency in the transmitter phase compensation FIFO in each channel can result in higher channel-to-channel skew.

Related Information

Transceiver Clocking in Cyclone V Devices

PLL Sharing

In a Quartus II design, you can merge two different protocol configurations to share the same CMU PLL resources. The protocol configurations and the two CMU PLLs to be merged must fulfill the following conditions:

- The transceiver channels must fit in the same transceiver bank.
- The CMU PLLs must have identical configurations, with identical output frequencies.
- The CMU PLLs must share a common REFCLK input.
- The CMU PLLs must share a common reset input.

Do not merge a CMU PLL with one that is used for PCI Express. The PCIe Hard IP needs complete control of the CMU PLL and its reset for compliance.

Document Revision History

The revision history for this chapter.

Table 1-30: Document Revision History

Date	Version	Changes
October 2018	2018.10.24	Made the following change: • Added "The CMU PLLs must share a common reset input" to the *PLL Sharing* requirements.
January 2016	2016.01.19	Made the following changes: • Changed the notes in the "GX/GT Devices with Three or Five Transceiver Channels and One PCIe Hard IP Block" figure.

Date	Version	Changes

September 2014	2014.09.30	• Updated the *6.144 Gbps CPRI Support Capability in GT Devices* section.
		• Added notes to the *Cyclone V GX/GT Devices with Four or Six Transceiver Channels and Two PCIe Hard IP Blocks* figure.
		• Updated the *Architecture Overview* section.
		• Added a note to the *Rate Match FIFO* section.
		• Updated the *Word Aligner* section.
		• *Word Aligner in Bit Slip Mode*
		  • Updated the *Word Aligner in Bit-Slip Mode* table.
		  • Added the *Word Aligner Configured in Bit Slip Mode* figure.
		  • Added an example of word aligner behavior when bit slipping is enabled.
		• Updated the *Word Aligner in Deterministic Latency State Machine Mode* section.
		• Added a description of `rx_syncstatus` to the *Word Aligner in Manual Alignment Mode* section.
		• Changed the description of deterministic latency state machine mode and added a description of `rx_syncstatus` to the *Word Aligner Operations in Deterministic Latency State Machine Mode* section.

May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base • Updated the *Transceiver Architecture in Cyclone V Devices* section. • Updated the *Architecture Overview* section. • Updated the *Automatic Lock Mode* section. • Updated the *Transmitter Buffer* section. • Updated Table 1-2. • Updated Table 1-5. • Updated the *Rate Match FIFO* section. • Updated the *Transceiver Banks* section. • Added the *Channel Variants* section. • Updated the *Transceiver Channel Architecture* section. • Updated Figure 1-7. • Updated the *Transmitter PLL* section. • Updated the *Channel PLL Architecture* section. • Updated the *Channel PLL as CDR PLL* section. • Updated the *CDR PLL in Automatic Lock Mode* section.

Date	Version	Changes
		• Added the *CDR PLL in Manual Lock Mode* section. • Updated the *Channel PLL as a CMU PLL* section. • Added the *fPLL as a Transmitter PLL* section. • Updated the *Clock Divider* section. • Updated the *Receiver PMA Datapath* section. • Updated the *Receiver Buffer* section. • Updated the *Programmable Receiver V_{CM}* section. • Updated the *Transmitter PMA Datapath* section. • Added the *Bit Reversal* section. • Updated the *Transmitter Buffer* section. • Updated the *Transmitter Buffer Features and Capabilities* section. • Updated the *Transmitter Protocol Specific* section. • Updated the *Calibration Block* section. • Updated the *PCS Architecture* section. • Updated the *Transmitter Phase Compensation FIFO* section. • Added the *Registered Mode* section. • Updated the *Byte Serializer* section. • Updated the *8B/10B Encoder in Single-Width Mode* section. • Updated the *Word Aligner* section. • Updated the *Word Aligner Options and Behaviors* section. • Updated the *Word Aligner in Manual Alignment Mode* section. • Updated the *Programmable Run-Length Violation Detection* section. • Updated the *Rate Match FIFO* section. • Updated the *8B/10B Decoder* section. • Updated the *Byte Deserializer* section. • Updated the *Byte Ordering in Single-Width Mode* section. • Updated the *Byte Ordering in Double-Width Mode* section. • Added the *Word Aligner-Based Ordering Mode* section. • Added the *Manual Ordering Mode* section. • Updated the *Receiver Phase Compensation FIFO* section. • Updated the *Channel Bonding* section. • Updated the *PLL Sharing* section.

Date	Version	Changes
December 2012	2012.12.03	Clarified note to Figure 1-6 to indicate only certain transceiver channels support interfacing to PCIe. Removed DC-Coupling information from Transmitter Buffer Features and Capabilities and PMA Receiver Buffer.
November 2012	2012.11.19	Reorganized content and updated template
June 2012	1.1	Added in contents of Transceiver Basics for Cyclone V Devices. Updated “Architecture Overview”, “PMA Architecture” and “PCS Architecture” sections. Updated Table 1–11. Updated Figure 1–36.

Transceiver Clocking in Cyclone V Devices

2

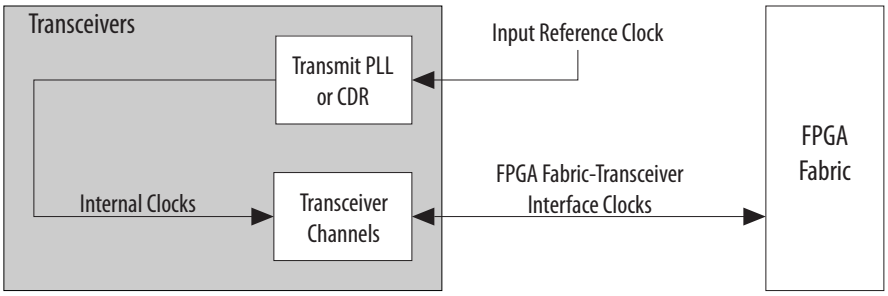
2016.01.28

CV-53002

Subscribe Send Feedback

This chapter provides information about the Cyclone® V transceiver clocking architecture. The chapter describes the clocks that are required for operation, internal clocking architecture, and clocking options when the transceiver interfaces with the FPGA fabric.

Figure 2-1: Transceiver Clocking Architecture Overview



Related Information

Cyclone V Device Handbook: Known Issues
Lists the planned updates to the *Cyclone V Device Handbook* chapters.

Input Reference Clocking

This section describes how the reference clock for the transmitter PLL and CDR is provided to generate the clocks required for transceiver operation.

Table 2-1: Input Reference Clock Sources

Sources	Transmitter PLL	CDR	Jitter Performance [6]
	CMU PLL		
Dedicated `refclk` pin	Yes	Yes	1
REFCLK network	Yes	Yes	2

[6] The lower number indicates better jitter performance.

Sources	Transmitter PLL	CDR	Jitter Performance [6]
	CMU PLL		
Dual-purpose RX / refclk pin	Yes	Yes	3
Fractional PLL	Yes	Yes	4
Generic CLK pin	No	No	5
Core clock network (GCLK, RCLK, PCLK)	No	No	6

Dedicated Reference Clock Pins

Cyclone V devices have one dedicated reference clock (refclk) pin for each bank of three transceiver channels.

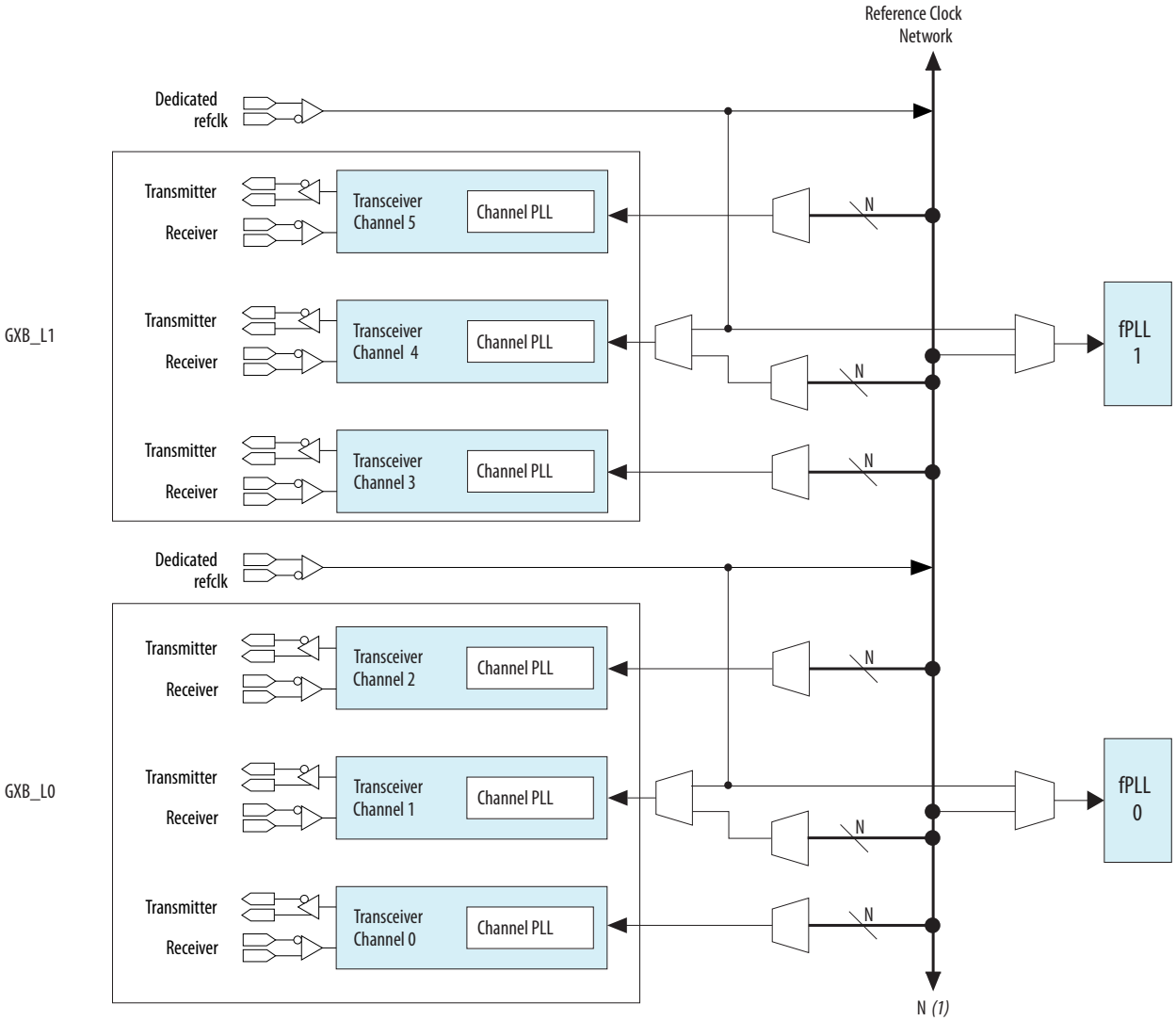
The dedicated reference clock pins drive the channel PLL in channel 1 or 4 directly. This option provides the best quality of input reference clock to the transmitter PLL and CDR.

Note: For specifications about the input frequency supported by the refclk pins, refer to the *Cyclone V Device Datasheet*.

As shown in the following figure the dedicated refclk pin direct connection to the channel PLL (which can be configured either as a CMU PLL or CDR) is only available in channel 1 of a transceiver bank and channel 4 of the neighboring transceiver bank.

[6] The lower number indicates better jitter performance.

Figure 2-2: Input Reference Clock Sources for Transceiver Channels



Note:
(1) N is the number of dedicated refclk pins, which equals the number of transceiver channels on a side divided by 3.

Related Information

Cyclone V Device Datasheet

Dedicated refclk Using the Reference Clock Network

Designs that use multiple channel PLLs with the same clock frequency can use the same dedicated `refclk` pin. Each dedicated `refclk` pin can drive any channel PLL (CMU PLL/CDR) and the fractional PLL through the reference clock network.

Figure 2-2 shows the input reference clock sources for six channel PLLs across two transceiver banks. For six transceiver channels, the total number of clock lines in the reference clock network is 2 (N = 6/3).

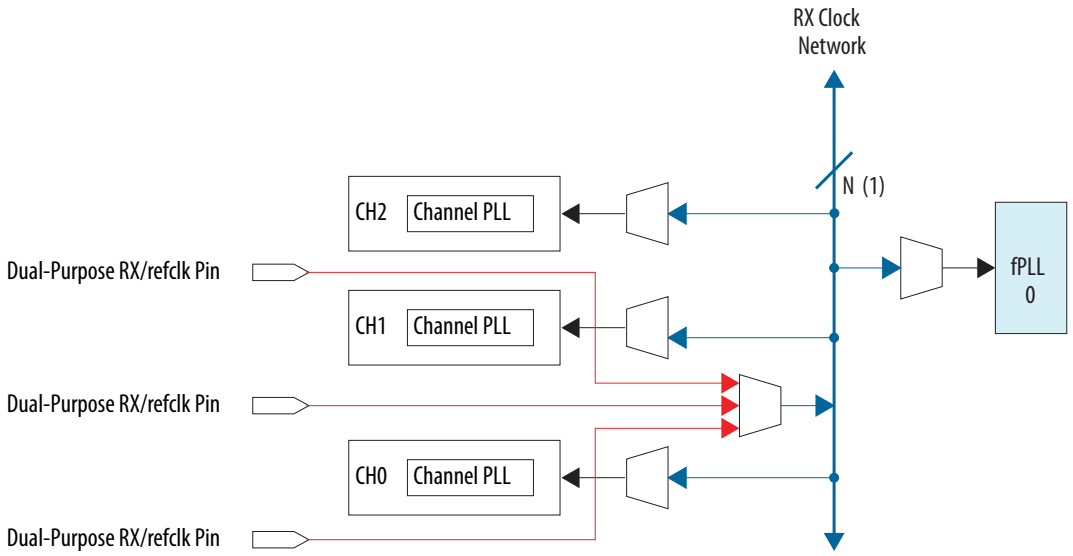
Dual-Purpose RX/refclk Pins

When not used as a receiver, an RX differential pair can be used as an additional input reference clock source.

The clock from the RX pins feed the RX clock network that spans all the channels on one side of the device. Only one RX differential pair for every three channels can be used as input reference clock at a time. The following figure shows the use of dual-purpose RX/refclk differential pin as input reference clock source and the RX clock network.

Note:
- An RX differential pair from another bank can be used as an input reference clock pin on the same side of the device.
- refclk switching cannot be performed when dual-purpose RX differential pins are used refclk pins.

Figure 2-3: Dual-Purpose RX/refclk Pin as an Input Reference Clock



Note:
(1) N is the number of transceiver channels on a side divided by 3.

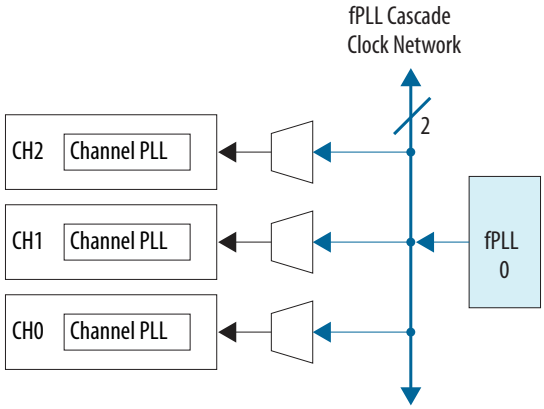
Fractional PLL (fPLL)

The fPLL clock output can be used as input reference clock source to transmitter PLL or CDR.

Cascading the fPLL to transmitter PLL or CDR enables you to use an input reference clock that is not supported by the transmitter PLL or CDR. The fPLL synthesizes a supported input reference clock for the transmitter PLL or CDR.

A fPLL is available for each bank of three transceiver channels. Each fPLL drives one of two fPLL cascade clock network lines that can provide an input reference clock to any transmitter PLL or CDR on the same side of a device.

Figure 2-4: fPLL Clock Output as Input Reference Clock



Note: It is not recommended to use fractional PLL in fractional mode for transceiver applications as a TX PLL or for PLL cascading.

Internal Clocking

This section describes the clocking architecture internal to Cyclone V transceivers.

Different physical coding sublayer (PCS) configurations and channel bonding options result in various transceiver clock paths.

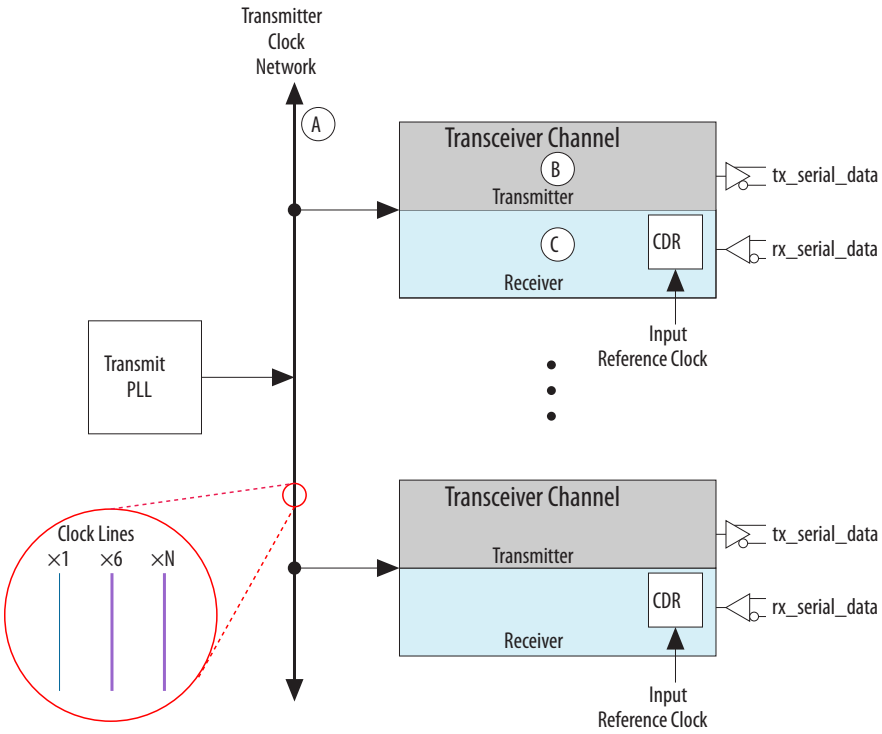
Note: The Quartus II software automatically performs the internal clock routing based on the transceiver configuration that you select.

The labels listed in the following table and figure mark the three sections of the transceiver internal clocking.

Table 2-2: Internal Clocking Subsections

Label	Scope	Description
A	Transmitter Clock Network	Clock distribution from transmitter PLLs to channels
B	Transmitter Clocking	Clocking architecture within transmitter channel datapath
C	Receiver Clocking	Clocking architecture within receiver channel datapath

Figure 2-5: Internal Clocking



Transmitter Clock Network

The transmitter PLL is comprised of the CMU PLL. All CMU PLLs are identical, but the usage varies depending on channel location due to the availability of access to the clock distribution network.

Table 2-3: Usage Capability of Each CMU PLL Within Two Transceiver Banks

CMU PLL Location in Two Transceiver Banks	Clock Network Access	Usage Capability
CH 0	No	Clock transmitter within same channel only
CH 1	Yes	Clock transmitter within same channel only and other channels via clock network
CH 2	No	Clock transmitter within same channel only
CH 3	No	Clock transmitter within same channel only
CH 4	Yes	Clock transmitter within same channel and other channels via clock network
CH 5	No	Clock transmitter within same channel only

The transmitter clock network routes the clock from the transmitter PLL to the transmitter channel. As shown in the previous figure, the transmitter clock network routes the clock from the transmit PLL to the transmitter channel. A clock divider provides two clocks to the transmitter channel:

- Serial clock—high-speed clock for the serializer
- Parallel clock—low-speed clock for the serializer and the PCS

Cyclone V transceivers support non-bonded and bonded transceiver clocking configurations:

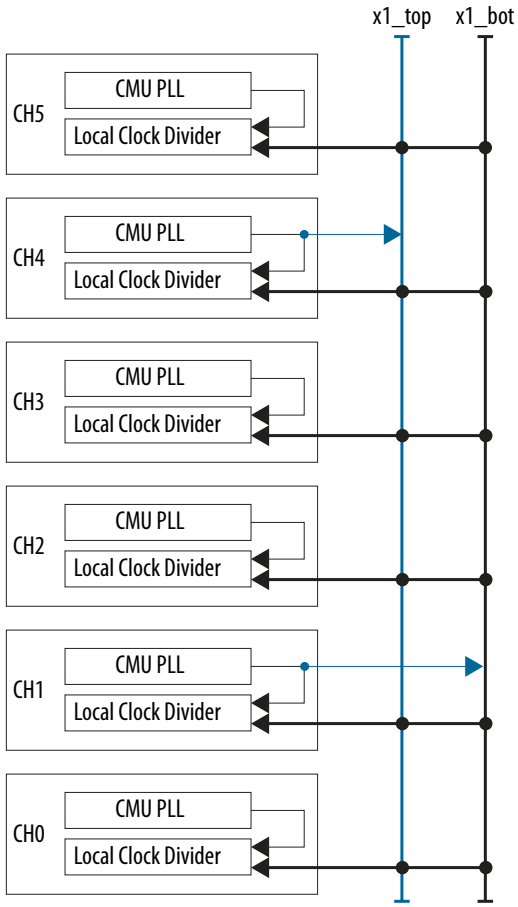
- Non-bonded configuration—Only the serial clock from the transmit PLL is routed to the transmitter channel. The clock divider of each channel generates the local parallel clock.
- Bonded configuration—Both the serial clock and parallel clock are routed from the central clock divider in channel 1 or 4 to the bonded transmitter channels.

The transmitter clock network is comprised of x1 (x1 and x1_fPLL), x6 and xN clock lines.

Table 2-4: Characteristics of x1, x6, and xN Clock Lines

Characteristics	x1	x1_fPLL	x6	xN
Clock Source	CMU PLL from CH 1 or CH 4 in two banks (serial clock only)	fPLL adjacent to transceivers (serial clock only)	Central clock divider from CH 1 or Ch 4 in two banks (serial and parallel clock)	x6 clock lines (serial and parallel clock)
Maximum Data Rate (Gbps)	5.0 (GT and ST), 3.125 (GX and SX)	3.125	5.0 (GT and ST), 3.125 (GX and SX)	3.125
Clock Line Span	Within two transceiver banks	Within a group of 3 channels (0, 1, 2 or 3, 4, 5)	Within two transceiver banks	Across all channels on the same side of the device
Non-bonded Configuration	Yes	Yes	Yes	Yes
Bonded Configuration	No	No	Yes	Yes

Figure 2-6: x1 Clock Line Architecture

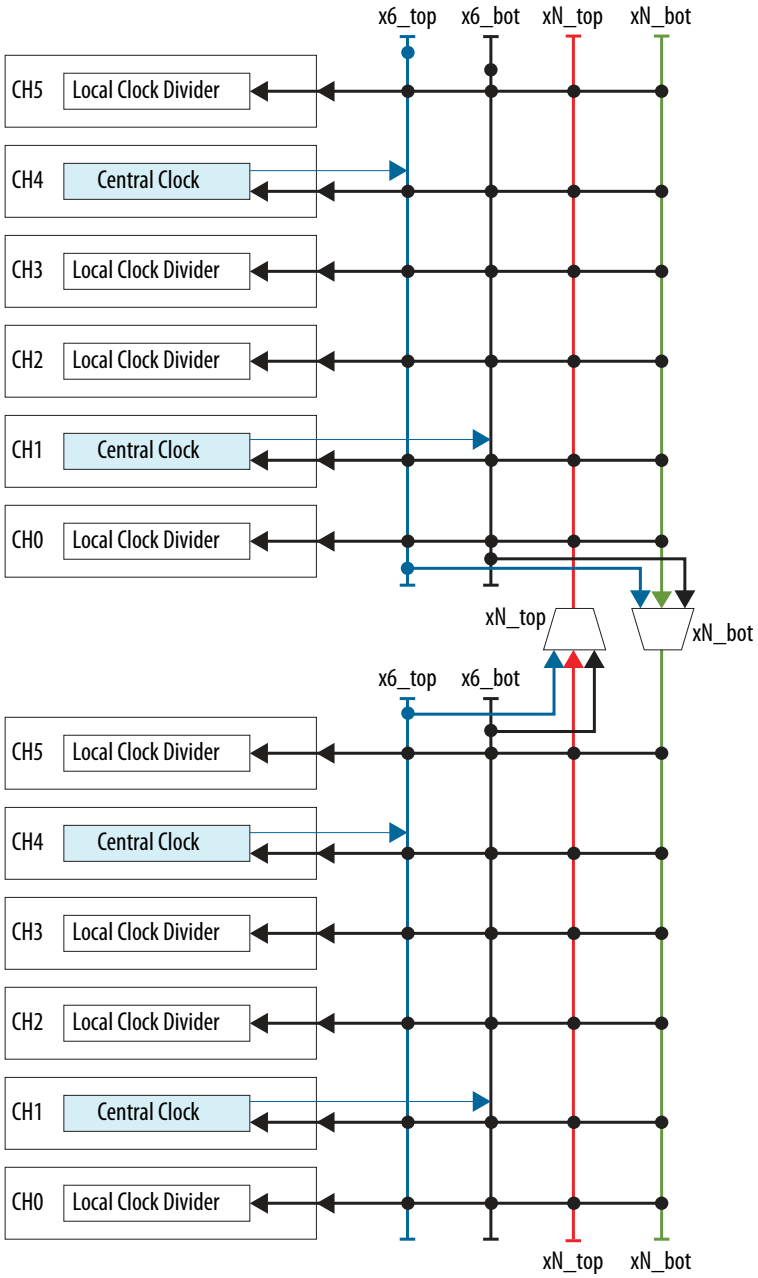


Note: All clock lines shown in this figure carry the serial clock only.

The x1 clock lines are driven by serial clocks of CMU PLLs from channels 1 and 4. The serial clock in the x1 clock line is then distributed to the local and central clock dividers of every channel within both the neighboring transceiver banks.

Note: When you configure the channel PLL as a CMU PLL to drive the local clock divider, or the central clock divider of its own channel, you cannot use the channel PLL as a CDR. Without a CDR, you can use the channel only as a transmitter channel.

Figure 2-7: x6 and xN Clock Line Architecture



Note: All the clock lines shown in this figure carry both the serial and parallel clocks.

The x6 clock lines are driven by serial and parallel clocks from the central clock divider in channels 1 and 4. For channels 0 to 5 within the 2 transceiver banks, the serial and parallel clocks in the x6 clock line are then distributed to every channel in both the transceiver banks.

The xN clock lines extend the clocking reach of the x6 clock line to all channels on the same side of the device. To reach a xN clock line, the clocks must be provided on the x6 clock line. The serial and parallel clocks in the x6 clock line are distributed to every channel within the two transceiver banks. The serial and

parallel clocks are distributed to other channels beyond the two banks or the six channels using the xN clock line.

In bonded configurations, serial and parallel clocks from the x6 or xN clock lines are received by the clock divider of every bonded channel and fed directly to the serializer. In a non-bonded configuration, the clock divider of every non-bonded channel receives the serial clock from the x6 or xN clock lines and generates the individual parallel clock to the serializer.

Note: In a bonded configuration, bonded channels must be placed contiguously without leaving a gap between the channels, except when the gap channel is a CMU PLL used for the bonded channels. The contiguous channels may span across many banks for bonded mode. For example, in x8 bonding, the bonded channels may span across 3 to 4 banks with the condition that there must be no gap between the channels except when the gap is due to the channel used for CMU PLL.

Transmitter Clocking

Transmitter (TX) clocking refers to the clocking architecture that is internal to the TX channel of a transceiver.

As shown in the following figure, the clock divider provides the serial clock to the serializer, and the parallel clock to the serializer and TX PCS. When the byte serializer is not used, the parallel clock clocks all the blocks up to the read side of the TX phase compensation FIFO. For configurations with the byte serializer, the parallel clock is divided by a factor of two for the byte serializer and the read side of the TX phase compensation FIFO. The read side clock of the TX phase compensation FIFO is also forwarded to the FPGA fabric to interface the FPGA fabric with the transceiver.

Figure 2-8: Clocking Architecture for Transmitter PCS and PMA Configuration

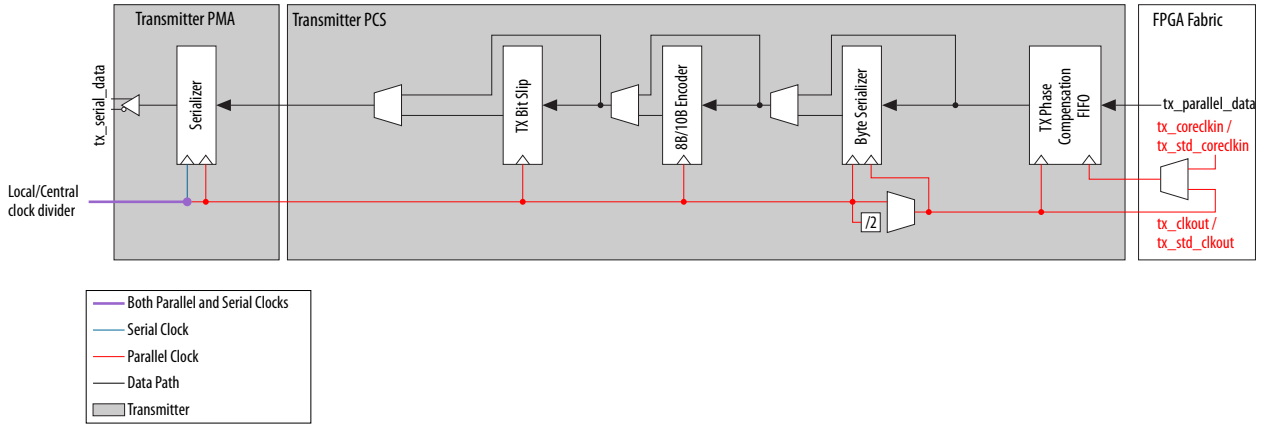


Table 2-5: Clock Sources for All TX PCS Blocks

PCS Block	Side	Clock Source
TX Phase Compensation FIFO	Write	FPGA fabric write clock, driven either by tx_clkout or tx_coreclkin
	Read	Parallel clock (divided). Clock forwarded to FPGA fabric as tx_clkout
Byte Serializer	Write	Parallel clock (divided) either by factor of 1 (not enabled), or factor of 2 (enabled)
	Read	Parallel clock

PCS Block	Side	Clock Source
8B/10B Encoder	—	Parallel clock
TX Bit Slip	—	Parallel clock

Non-Bonded Channel Configurations

This section describes the clock path for non-bonded configurations.

The following table describes the clock path for non-bonded configuration with the CMU PLL and fPLL as TX PLL using various clock lines.

Table 2-6: Clock Path for Non-Bonded Configurations

Clock Line	Transmitter PLL	Clock Path
x1	CMU	CMU PLL » x1 » individual clock divider » serializer
x6, xN	CMU	CMU PLL » central clock divider » x6 » xN » individual clock divider » serializer (7)
	fPLL	fPLL » x1_fPLL » central clock divider » x6 » individual clock divider » serializer (7)

(7) Non-bonded channels within the neighboring two banks or within the six channels of TX PLL are driven by clocks from x6 clock line. Channels in other banks outside the six channels are driven by the xN clock line.

Figure 2-9: Three Non-Bonded Transmitter Channels Driven by CMU PLL using x1 Clock Line Within a Transceiver Bank

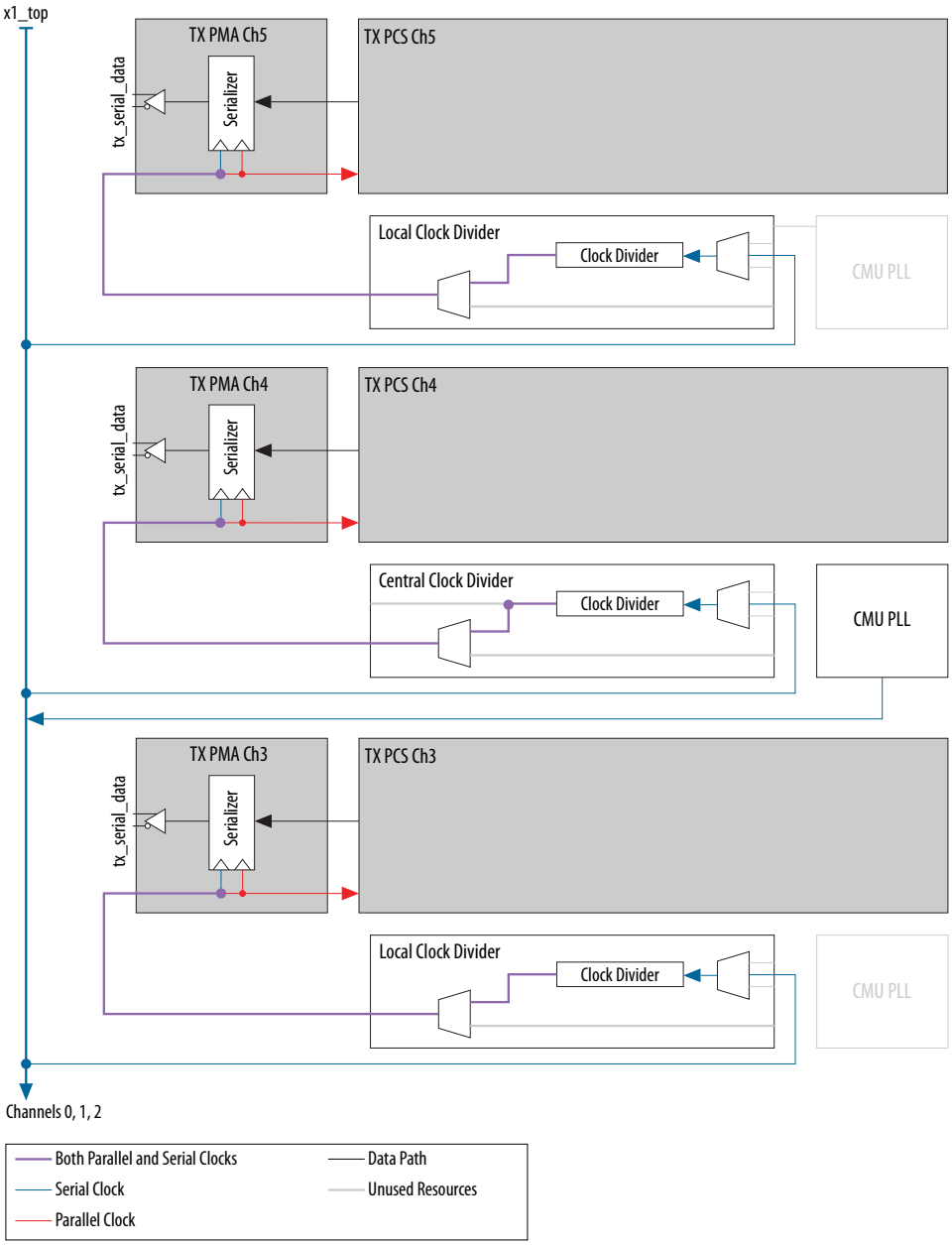
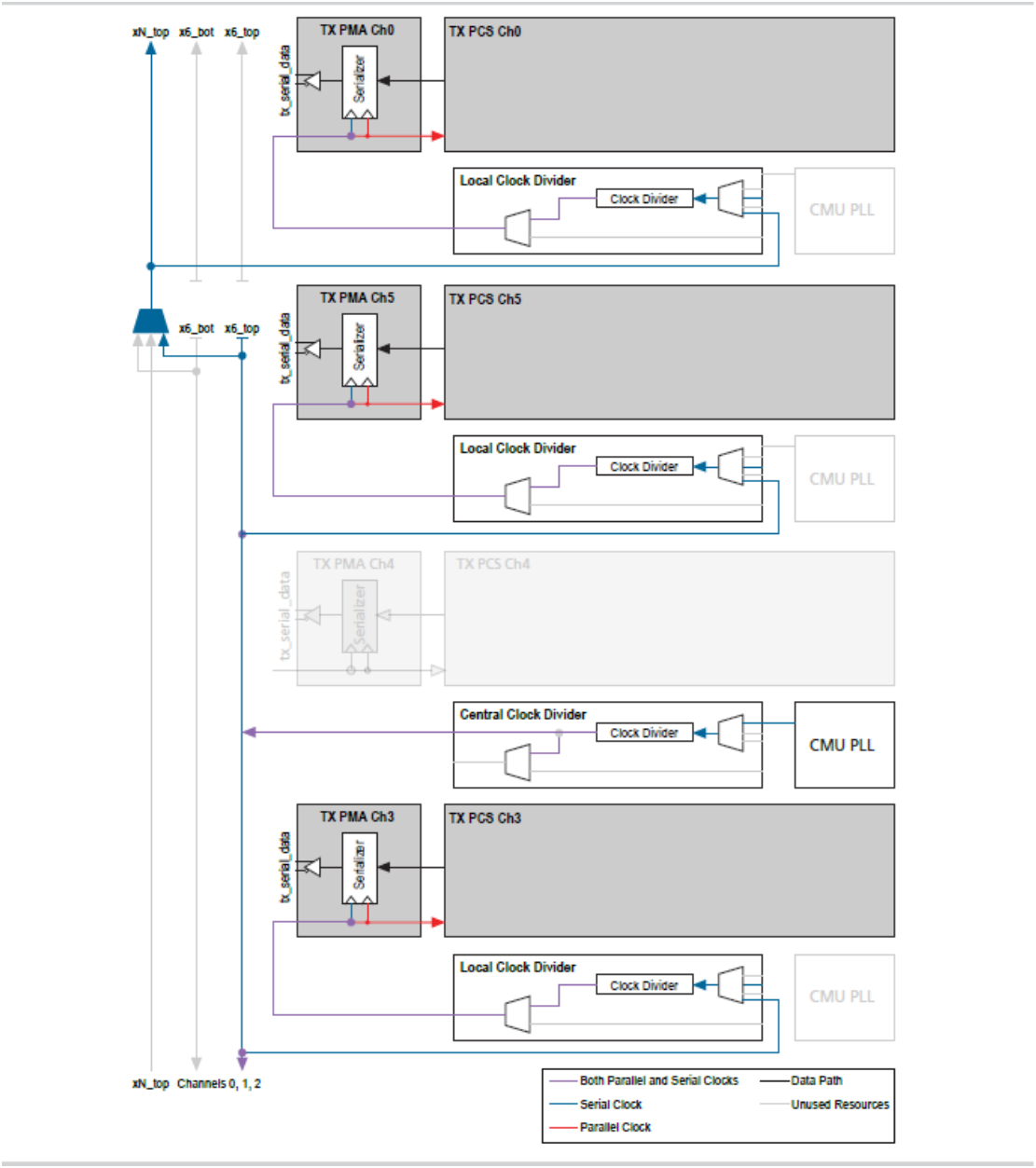


Figure 2-10: Three Non-Bonded Transmitter Channels Driven by CMU PLL using x6 and xN Clock Lines Across Multiple Transceiver Banks



Bonded Channel Configurations

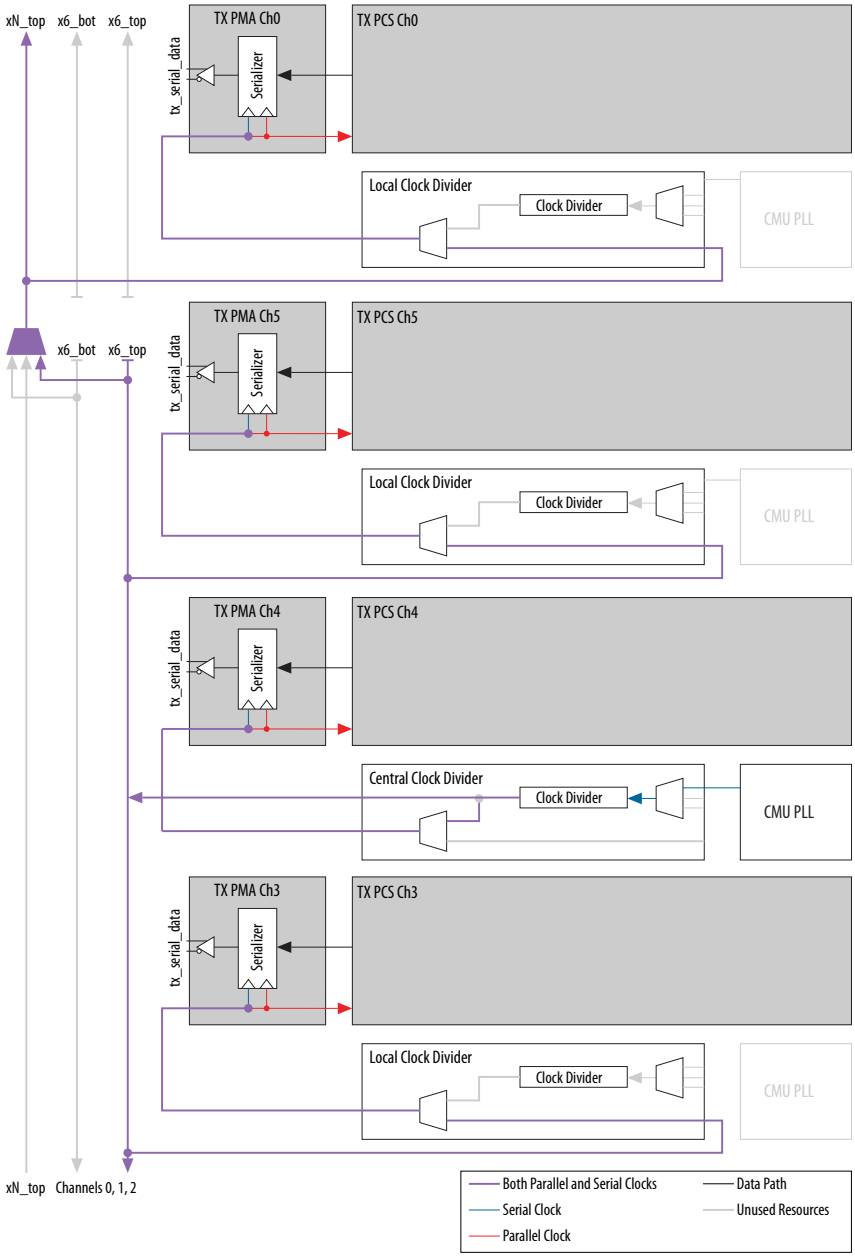
This section describes the clock path for bonded configurations.

The following table describes the clock path for bonded configurations with the CMU PLL as TX PLL using various clock lines.

Table 2-7: Clock Path for Bonded Configurations

Clock Line	Transmitter PLL	Clock Path
x6, xN	CMU	CMU PLL » central clock divider » x6 » xN » serializer [8]

Figure 2-11: Four Bonded Transmitter Channels Driven by CMU PLL using x6 and xN Clock Line Across Multiple Transceiver Banks



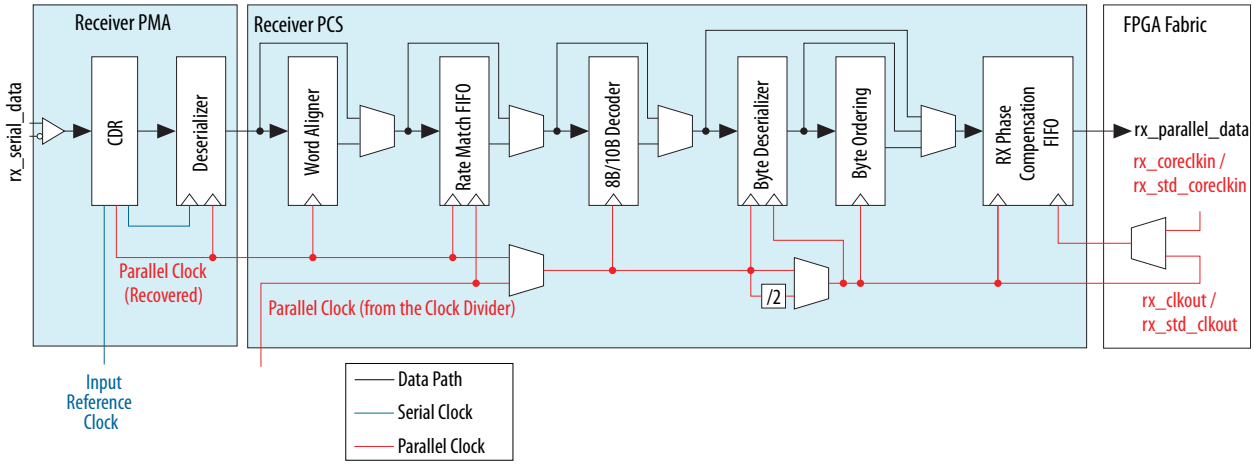
[8] Bonded channels within the neighboring two banks or within the six channels of TX PLL are driven by clocks from x6 clock line. Channels in other banks outside the 6 channels are driven by the xN clock line.

Note: When a channel PLL is configured as a CMU PLL to drive the local clock divider or the central clock divider of its own channel, the channel PLL cannot be used as a CDR. Without a CDR, the channel can be used only as a transmitter.

Receiver Clocking

This section describes receiver clocking, which is the clocking architecture internal to the receiver channel of a transceiver.

Figure 2-12: Clocking Architecture for Receiver PCS and PMA Configuration



The CDR in the PMA of each channel recovers the serial clock from the incoming data and generates the parallel clock (recovered) by dividing the serial clock (recovered). The deserializer uses both clocks. The receiver PCS can use the following clocks depending on the configuration of the receiver channel:

- Parallel clock (recovered) from the CDR in the PMA
- Parallel clock from the clock divider that is used by the channel's transmitter PCS

Table 2-8: Clock Sources for All Receiver PCS Blocks

Block	Side	Clock Source
Word aligner	-	Parallel clock (recovered)
Rate match FIFO	Write	Parallel clock (recovered)
	Read	Parallel clock from the clock divider
8B/10B decoder	-	• Rate match FIFO is not used-Parallel clock (recovered) • Rate match FIFO is used-Parallel clock from the clock divider

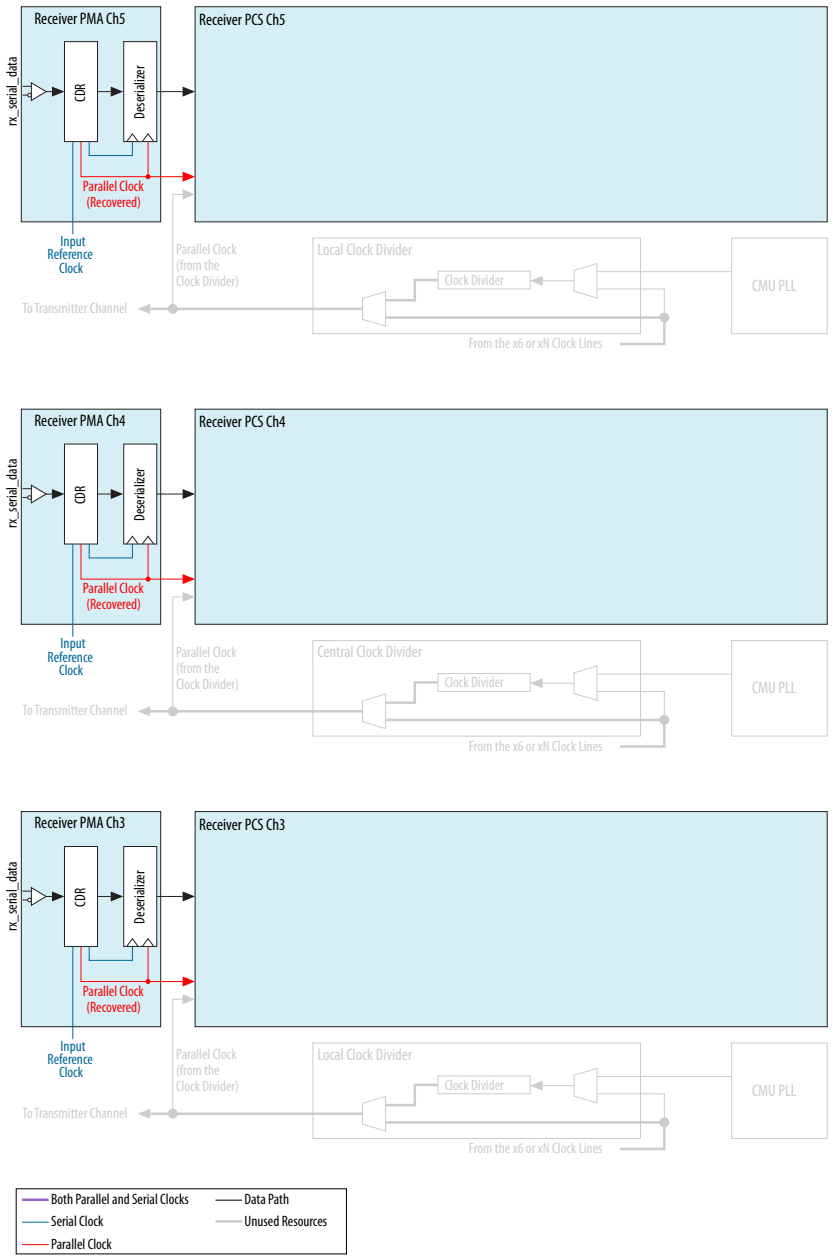
Block	Side	Clock Source
Byte deserializer	Write	• Rate match FIFO is not used-Parallel clock (recovered) • Rate match FIFO is used-Parallel clock from the clock divider
	Read	Divided down version of the write side clock depending on the deserialization factor of 1 or 2, also called the parallel clock (divided)
Byte ordering	-	Parallel clock (divided)
Receiver (RX) phase compensation FIFO	Write	Parallel clock (divided). This clock is also forwarded to the FPGA fabric.
	Read	Clock sourced from the FPGA fabric

Receiver Non-Bonded Channel Configurations

This section describes the receiver non-bonded channel configurations.

The receiver clocking in non-bonded mode varies depending on whether the rate match FIFO is enabled. When the rate match FIFO is not enabled, the receiver PCS in every channel uses the parallel recovered clock. When the rate match FIFO is enabled, the receiver PCS in every channel uses both the parallel recovered clock and parallel clock from the clock divider.

Figure 2-13: Three Non-Bonded Receiver Channels without Rate Match FIFO Enabled

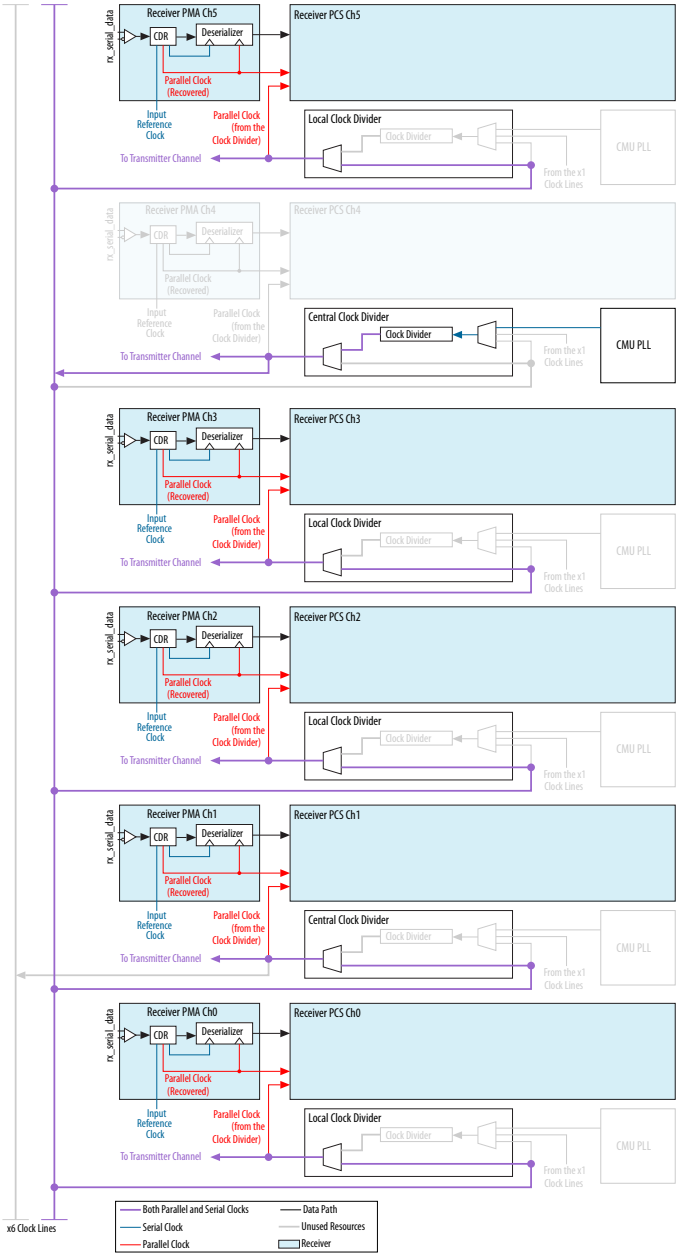


Receiver Bonded Channel Configurations

Receiver channels can only be bonded in configurations where rate match FIFOs are enabled. When bonded, the receiver PCS requires the parallel clock (recovered) and the parallel clock from the central clock divider in channel 1 or 4.

Note: For more information about the clocking scheme used in different configurations, refer to the *Transceiver Protocol Configurations in Cyclone V Devices* and *Transceiver Custom Configurations in Cyclone V Devices* chapters.

Figure 2-14: Five Bonded Receiver Channels with Rate Match FIFO Enabled



Related Information

- Transceiver Protocol Configurations in Cyclone V Devices
- Transceiver Custom Configurations in Cyclone V Devices

FPGA Fabric–Transceiver Interface Clocking

This section describes the clocking options available when the transceiver interfaces with the FPGA fabric.

The FPGA fabric-transceiver interface clocks consist of clock signals from the FPGA fabric to the transceiver blocks and clock signals from the transceiver blocks to the FPGA fabric. These clock resources use the clock networks in the FPGA core, including the global (GCLK), regional (RCLK), and periphery (PCLK) clock networks.

The FPGA fabric–transceiver interface clocks can be subdivided into the following three categories:

- Input reference clocks—Can be an FPGA fabric–transceiver interface clock. This may occur when the FPGA fabric-transceiver interface clock is forwarded to the FPGA fabric, where it can then clock logic.
- Transceiver datapath interface clocks—Used to transfer data, control, and status signals between the FPGA fabric and the transceiver channels. The transceiver channel forwards the `tx_clkout` signal to the FPGA fabric to clock the data and control signals into the transmitter. The transceiver channel also forwards the recovered `rx_clkout` clock (in configurations without the rate matcher) or the `tx_clkout` clock (in configurations with the rate matcher) to the FPGA fabric to clock the data and status signals from the receiver into the FPGA fabric.
- Other transceiver clocks—The following transceiver clocks form a part of the FPGA fabric–transceiver interface clocks:
  - `mgmt_clk`—Avalon®-MM interface clock used for controlling the transceivers, dynamic reconfiguration, and calibration
  - `fixed_clk`—the 125 MHz fixed-rate clock used in the PCIe (PIPE) receiver detect circuitry

Table 2-9: FPGA Fabric–Transceiver Interface Clocks

Clock Name	Clock Description	Interface Direction	FPGA Fabric Clock Resource Utilization
tx_pll_refclk, rx_cdr_refclk	Input reference clock used for clocking logic in the FPGA fabric	Transceiver-to-FPGA fabric	GCLK, RCLK, PCLK
tx_clkout, tx_pma_clkout	Clock forwarded by the transceiver for clocking the transceiver datapath interface		
rx_clkout, rx_pma_clkout	Clock forwarded by the receiver for clocking the receiver datapath interface		
tx_coreclkin	User-selected clock for clocking the transmitter datapath interface	FPGA fabric-to-transceiver	
rx_coreclkin	User-selected clock for clocking the receiver datapath interface		
fixed_clk	PCIe receiver detect clock		
mgmt_clk [9]	Avalon-MM interface management clock		

[9] The mgmt_clk is a free-running clock that is not derived from the transceiver blocks.

Note: For more information about the GCLK, RCLK, and PCLK resources available in each device, refer to the *Clock Networks and PLLs in Cyclone V Devices* chapter.

Related Information

Clock Networks and PLLs in Cyclone V Devices

Transceiver Datapath Interface Clocking

There are two types of design considerations for clock optimization when interfacing the transceiver datapath to the FPGA fabric:

- PCS with FIFO in phase compensation mode – share clock network for identical channels
- PCS with FIFO in registered mode or PMA direct mode – refer to *AN 580: Achieving Timing Closure in Basic (PMA Direct) Functional Mode*, for additional timing closure techniques between transceiver and FPGA fabric

The following sections describe design considerations for interfacing the PCS transmitter and PCS receiver datapath to the FPGA fabric with FIFO in phase compensation mode.

Related Information

AN 580: Achieving Timing Closure in Basic (PMA Direct) Functional Mode

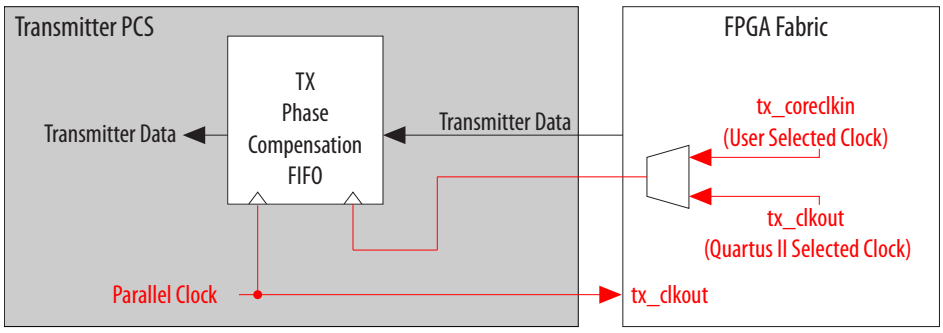
Transmitter Datapath Interface Clocking

The write side of the TX phase compensation FIFO makes up the transmitter datapath interface. The transmitter datapath interface clock clocks this interface.

The following figure shows the transmitter datapath interface clocking. The transmitter PCS forwards the following clocks to the FPGA fabric:

- `tx_clkout`—for each transmitter channel in a non-bonded configuration
- `tx_clkout[0]`—for all transmitter channels in a bonded configuration

Figure 2-15: Transmitter Datapath Interface Clocking for Transceivers



All configurations that use the PCS channel must have a 0 parts per million (ppm) difference between write and read clocks of the transmitter phase compensation FIFO.

Note: For more information about interface clocking for each configuration, refer to the T*ransceiver Custom Configuration in Cyclone V Devices* and *Transceiver Protocol Configurations in Cyclone V Devices* chapters.

You can clock the transmitter datapath interface with one of the following options:

- The Quartus II-selected transmitter datapath interface clock
- The user-selected transmitter datapath interface clock

Note: To reduce GCLK, RCLK, and PCLK resource utilization in your design, you can select the user-selection option to share the transceiver datapath interface clocks.

Related Information

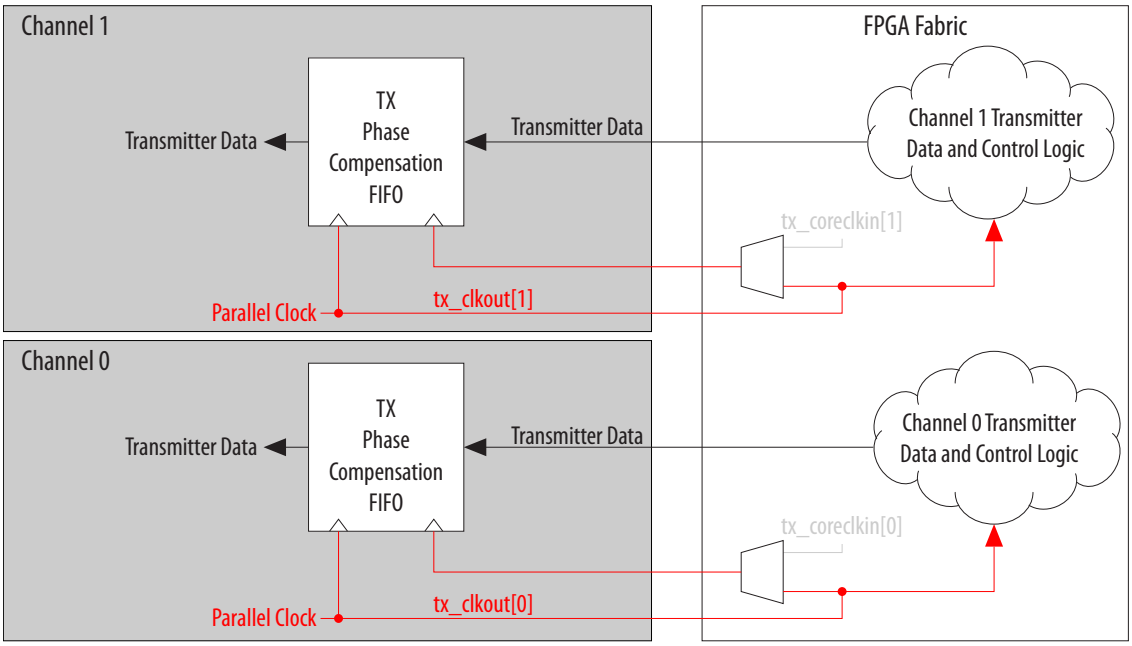
- **Transceiver Custom Configurations in Cyclone V Devices**
- **Transceiver Protocol Configurations in Cyclone V Devices**

Quartus II-Software Selected Transmitter Datapath Interface Clock

The Quartus II software automatically selects the appropriate clock from the FPGA fabric to clock the transmitter datapath interface.

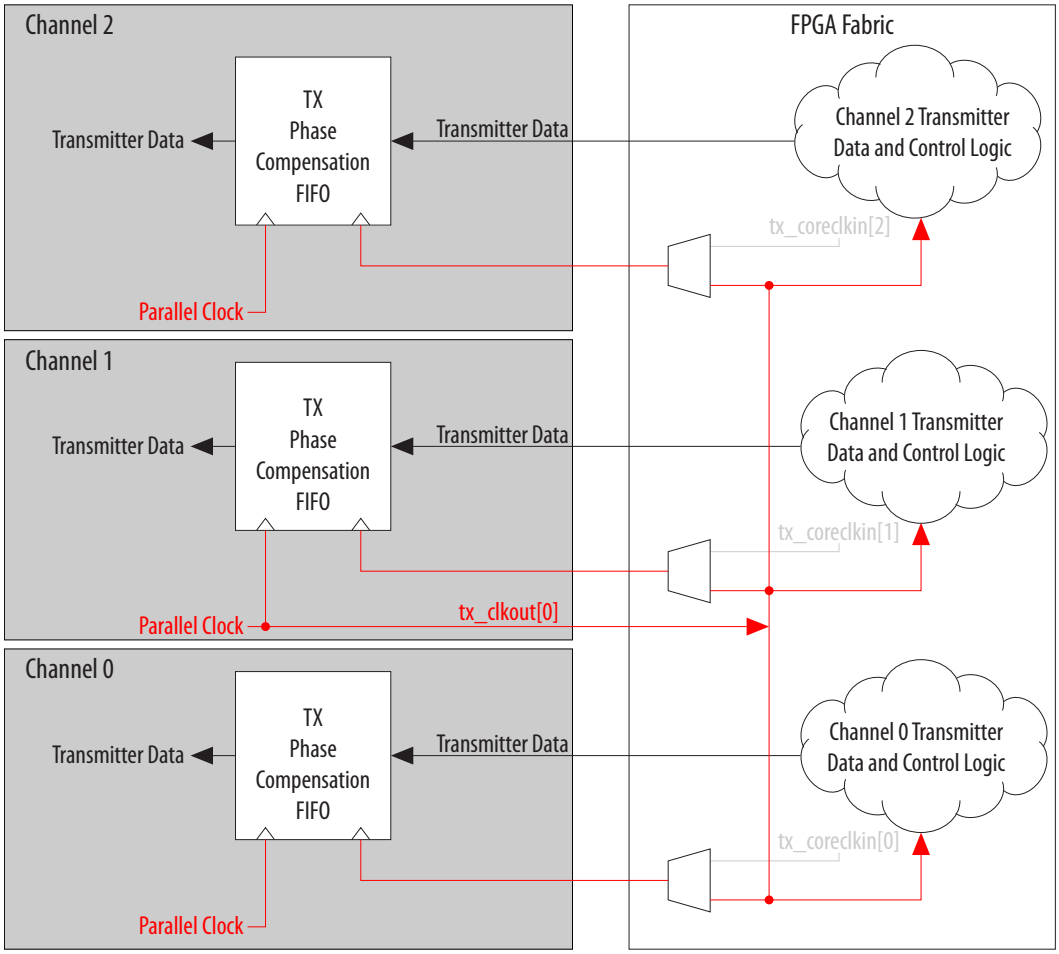
The following figure shows the transmitter datapath interface of two transceiver non-bonded channels clocked by their respective transmitter PCS clocks, which are forwarded to the FPGA fabric.

Figure 2-16: Transmitter Datapath Interface Clocking for Non-Bonded Channels



The following figure shows the transmitter datapath interface of three bonded channels clocked by the `tx_clkout[0]` clock. The `tx_clkout[0]` clock is derived from the central clock divider of channel 1 or 4 of the two transceiver banks.

Figure 2-17: Transmitter Datapath Interface Clocking for Three Bonded Channels



Selecting a Transmitter Datapath Interface Clock

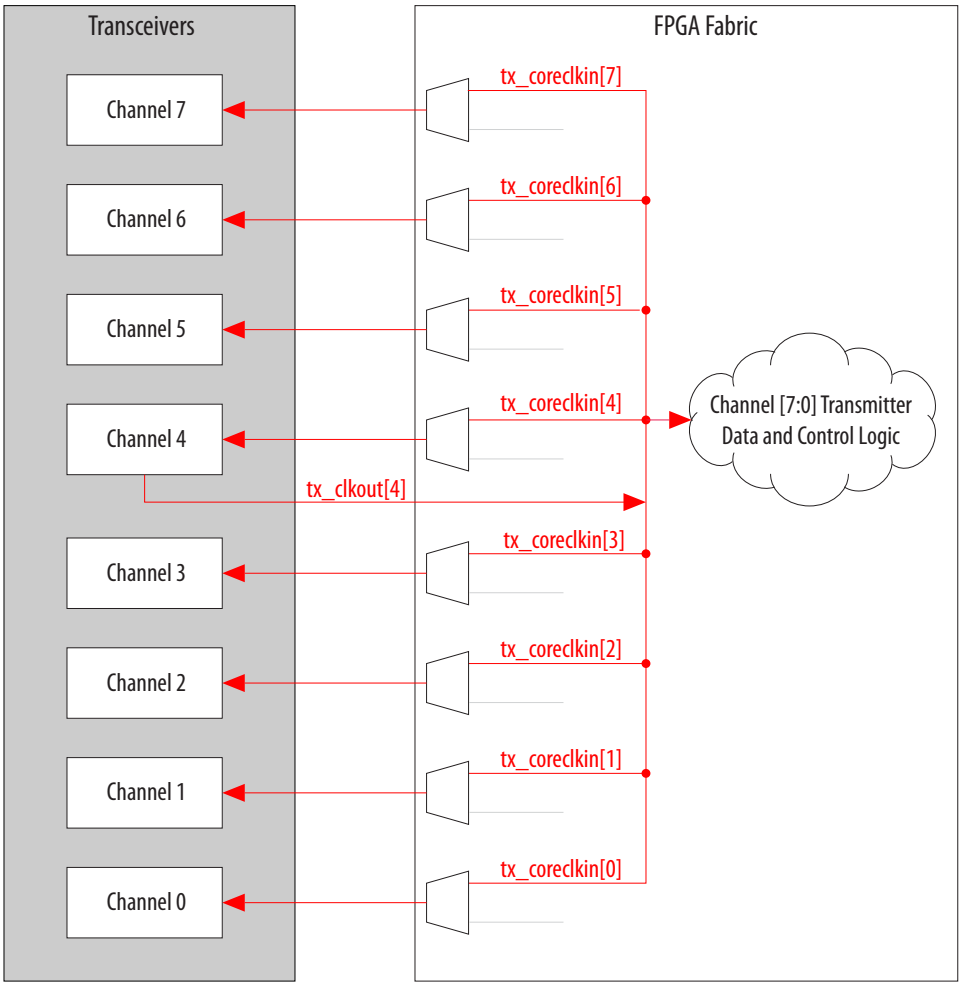
Multiple non-bonded transmitter channels use a large portion of GCLK, RCLK, and PCLK resources. Selecting a common clock driver for the transmitter datapath interface of all identical transmitter channels saves clock resources.

Multiple transmitter channels that are non-bonded lead to high utilization of GCLK, RCLK, and PCLK resources (one clock resource per channel). You can significantly reduce GCLK, RCLK, and PCLK resource use for transmitter datapath clocks if the transmitter channels are identical.

Note: Identical transmitter channels have the same input reference clock source, transmit PLL configuration, transmitter PMA, and PCS configuration, but may have different analog settings, such as transmitter voltage output differential (V_{OD}), transmitter common-mode voltage (V_{CM}), or pre-emphasis settings.

To achieve the clock resource savings, select a common clock driver for the transmitter datapath interface of all identical transmitter channels. The following figure shows six identical channels clocked by a single clock (`tx_clkout` of channel 4).

Figure 2-18: Six Identical Channels with a Single User-Selected Transmitter Interface Clock



To clock six identical channels with a single clock, perform these steps:

1. Instantiate the `tx_coreclkin` port for all the identical transmitter channels (`tx_coreclkin[5:0]`).
2. Connect `tx_clkout[4]` to the `tx_coreclkin[5:0]` ports.
3. Connect `tx_clkout[4]` to the transmitter data and control logic for all six channels.

Note: Resetting or powering down channel 4 causes a loss of the clock for all six channels.

The common clock must have a 0 ppm difference for the read side of the transmitter phase compensation FIFO of all the identical channels. A frequency difference causes the FIFO to under run or overflow, depending on whether the common clock is slower or faster, respectively.

You can drive the 0 ppm common clock by one of the following sources:

- `tx_clkout` of any channel in non-bonded channel configurations
- `tx_clkout[0]` in bonded channel configurations
- Dedicated `refclk` pins

Note: The Quartus II software does not allow gated clocks or clocks that are generated in the FPGA logic to drive the `tx_coreclkin` ports.

You must ensure a 0 ppm difference. The Quartus II software is unable to ensure a 0 ppm difference because it allows you to use external pins, such as dedicated `refclk` pins.

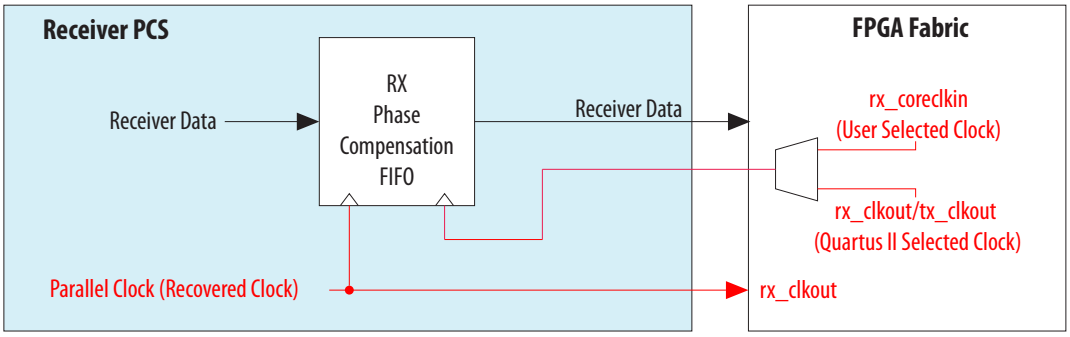
Receiver Datapath Interface Clock

The read side of the RX phase compensation FIFO makes up the 6-Gbps receiver datapath interface. The receiver datapath interface clock clocks this interface.

The receiver PCS forwards the following clocks to the FPGA fabric:

- `rx_clkout`—for each receiver channel in a non-bonded configuration when you do not use a rate matcher
- `tx_clkout`—for each receiver channel in a non-bonded configuration when you use a rate matcher
- single `rx_clkout[0]`—for all receiver channels in a bonded configuration

Figure 2-19: Receiver Datapath Interface Clocking



All configurations that use the PCS channel must have a 0 ppm difference between the receiver datapath interface clock and the read side clock of the RX phase compensation FIFO.

Note: For more information about interface clocking for each configuration, refer to the *Transceiver Custom Configuration in Cyclone V Devices* and *Transceiver Protocol Configurations in Cyclone V Devices* chapters.

You can clock the receiver datapath interface with one of the following options:

- The Quartus II-selected receiver datapath interface clock
- The user-selected receiver datapath interface clock

Note: To reduce GCLK, RCLK, and PCLK resource utilization in your design, you can select the user-selection option to share the transceiver datapath interface clocks.

Related Information

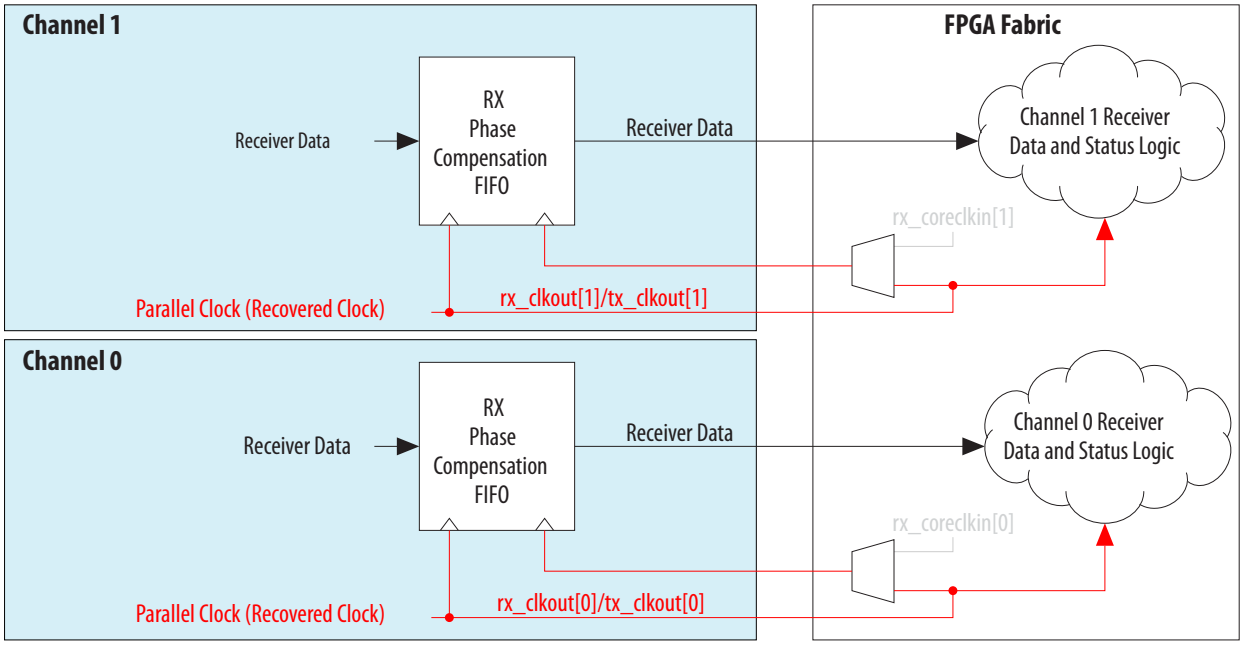
- **Transceiver Custom Configuration in Cyclone V Devices**
- **Transceiver Protocol Configurations in Cyclone V Devices**

Quartus II Software-Selected Receiver Datapath Interface Clock

Quartus II automatically selects the appropriate clock from the FPGA fabric to clock the receiver datapath interface.

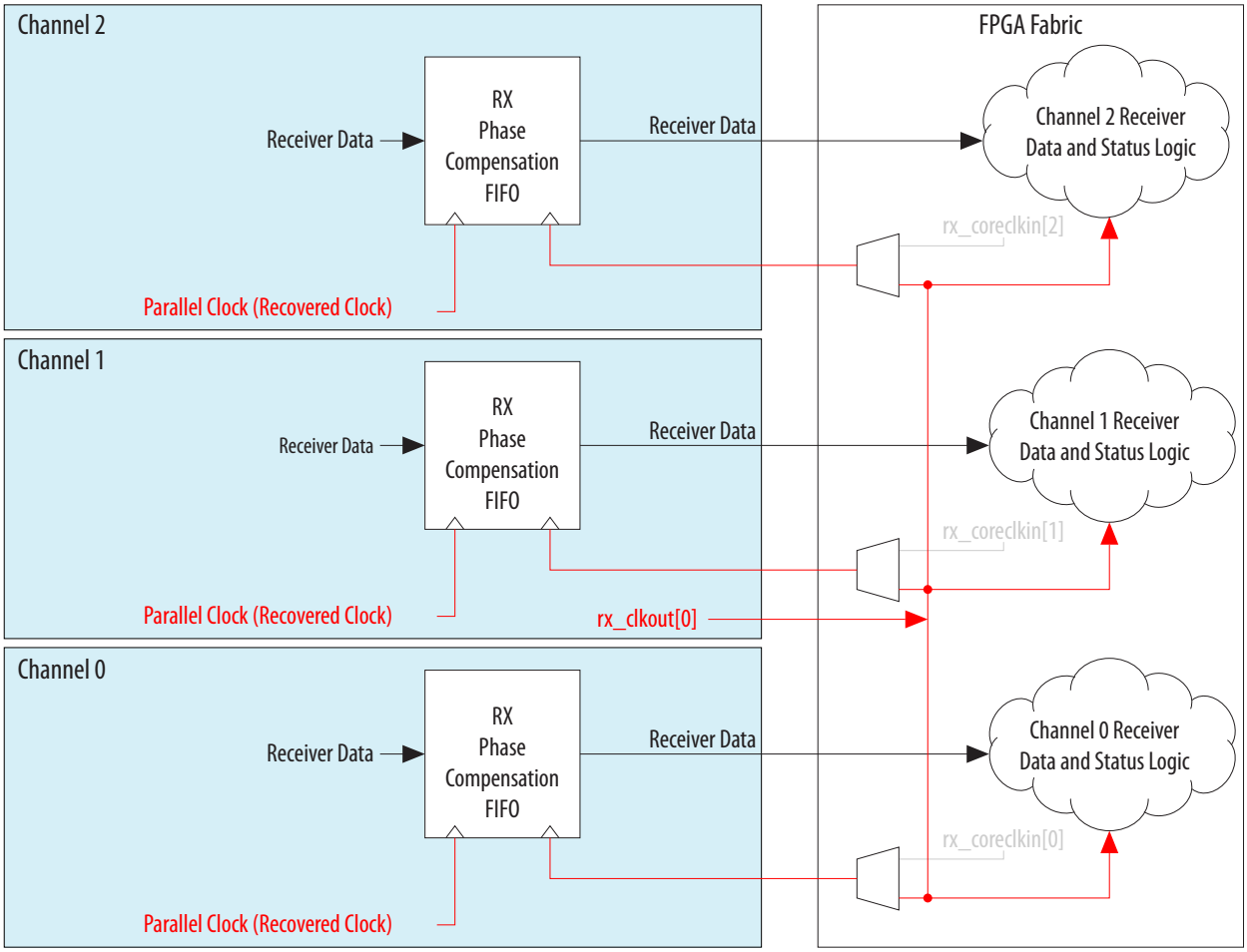
The following figure shows the receiver datapath interface of two transceiver non-bonded channels clocked by their respective receiver PCS clocks, which are forwarded to the FPGA fabric.

Figure 2-20: Receiver Datapath Interface Clocking for Non-Bonded Channels



The following figure shows the receiver datapath interface of three bonded channels clocked by the `tx_clkout[0]` clock. The `tx_clkout[0]` clock is derived from the central clock divider of channel 1 or 4 of the two transceiver banks.

Figure 2-21: Receiver Datapath Interface Clocking for Three Bonded Channels



Selecting a Receiver Datapath Interface Clock

Multiple non-bonded receiver channels use a large portion of GCLK, RCLK, and PCLK resources. Selecting a common clock driver for the receiver datapath interface of all identical receiver channels saves clock resources.

Non-bonded multiple receiver channels lead to high utilization of GCLK, RCLK, and PCLK resources (one clock resource per channel). You can significantly reduce GCLK, RCLK, and PCLK resource use for the receiver datapath clocks if the receiver channels are identical.

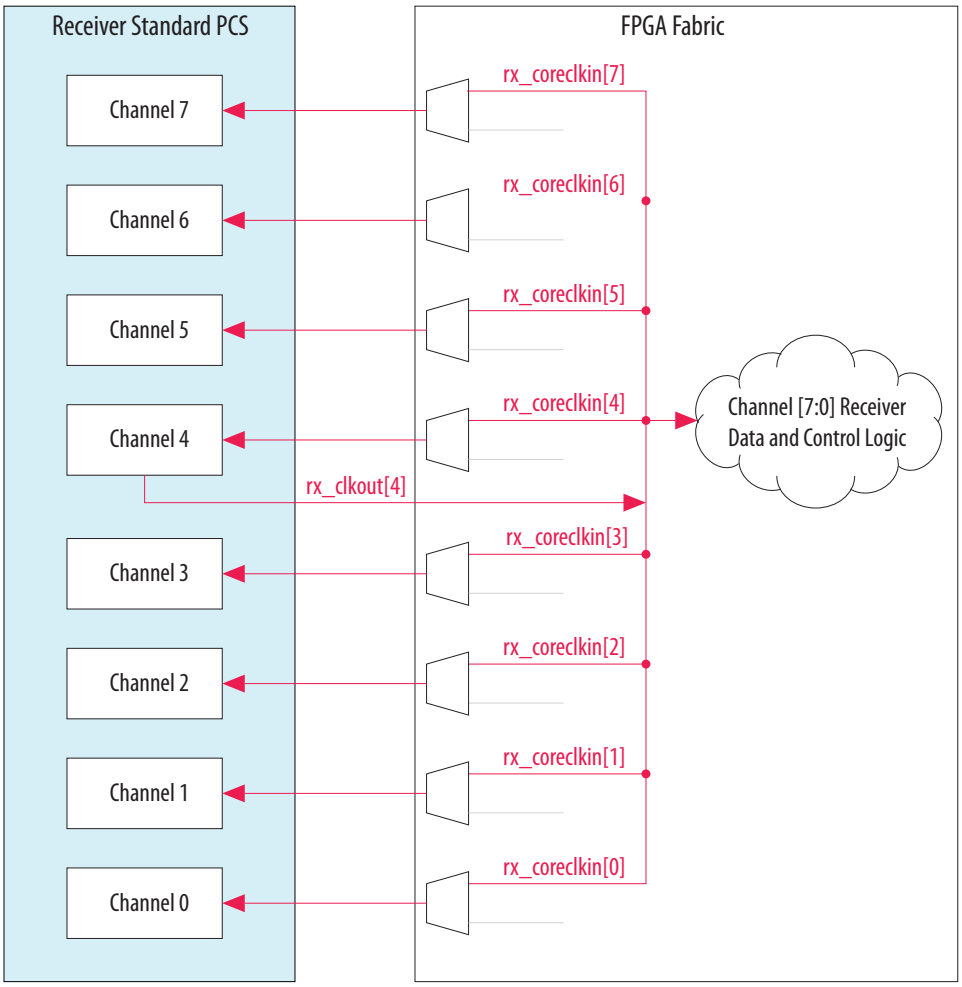
Note: Identical receiver channels are defined as channels that have the same input reference clock source for the CDR and the same receiver PMA and PCS configuration. These channels may have different analog settings, such as receiver common mode voltage (V_{ICM}), equalization, or DC gain setting.

To achieve clock resource savings, select a common clock driver for the receiver datapath interface of all identical receiver channels. To select a common clock driver, perform these steps:

1. Instantiate the `rx_coreclkin` port for all the identical receiver channels
2. Connect the common clock driver to their receiver datapath interface, and receiver data and control logic.

The following figure shows six identical channels that are clocked by a single clock (`rx_clkout[0]`).

Figure 2-22: Six Identical Channels with a Single User-Selected Receiver Interface Clock



To clock six identical channels with a single clock, perform these steps:

- Instantiate the `rx_coreclkin` port for all the identical receiver channels (`rx_coreclkin[5:0]`).
- Connect `rx_clkout[4]` to the `rx_coreclkin[5:0]` ports.
- Connect `rx_clkout[4]` to the receiver data and control logic for all six channels.

Note: Resetting or powering down channel 4 leads to a loss of the clock for all six channels.

The common clock must have a 0 ppm difference for the write side of the RX phase compensation FIFO of all the identical channels. A frequency difference causes the FIFO to under run or overflow, depending on whether the common clock is faster or slower, respectively.

You can drive the 0 ppm common clock driver from one of the following sources:

- `tx_clkout` of any channel in non-bonded receiver channel configurations with the rate matcher
- `rx_clkout` of any channel in non-bonded receiver channel configurations without the rate matcher
- `tx_clkout[0]` in bonded receiver channel configurations
- Dedicated `refclk` pins

Note: The Quartus II software does not allow gated clocks or clocks generated in the FPGA logic to drive the `rx_coreclkin` ports.

Note: You must ensure a 0 ppm difference. The Quartus II software is unable to ensure a 0 ppm difference because it allows you to use external pins, such as dedicated `refclk` pins.

Document Revision History

The table below lists the revision history for this chapter.

Table 2-10: Document Revision History

Date	Version	Changes
January 2016	2016.01.28	• Added fPLL information to the figure "Input Reference Clock Sources for Transceiver Channels". • Added fPLL information to table "Clock Path for Non-Bonded Configurations".
September 2014	2014.09.30	• Updated the chapter to indicate that it is not recommended to use fractional PLL in fractional mode as a TX PLL or for PLL cascading. • Updated *Table: Characteristics of x1, x6, and xN Clock Lines* with x1_fPLL characteristics.
May 2013	2013.05.06	• Updated Content for Quartus II software version 13.0 • Updated figure "Input Reference Clock Sources for Transceiver Channels" • Added a section on "Dual-Purpose RX/refclk Pins". • Added link to the known document issues in the Knowledge Base.
November 2012	2012.11.19	• Reorganized content and updated template. • Updated for the Quartus II software version 12.1.
June 2012	1.1	Minor editorial changes.
October 2011	1.0	Initial release.

Transceiver Reset Control in Cyclone V Devices

3

2016.01.28

CV-53003

Altera's recommended reset sequence ensures that both the physical coding sublayer (PCS) and physical medium attachment (PMA) in each transceiver channel are initialized and functioning correctly.

Related Information

Cyclone V Device Handbook: Known Issues
Lists the planned updates to the *Cyclone V Device Handbook* chapters.

PHY IP Embedded Reset Controller

The embedded reset controller in the PHY IP enables you to initialize the transceiver physical coding sublayer (PCS) and physical medium attachment (PMA) blocks.

To simplify your transceiver-based design, the embedded reset controller provides an option that requires only one control input to implement an automatic reset sequence. Only one embedded reset controller is available for all the channels in a PHY IP instance.

The embedded reset controller automatically performs the entire transceiver reset sequence whenever the `phy_mgmt_clk_reset` signal is triggered. In case of loss-of-link or loss-of-data, the embedded reset controller asserts the appropriate reset signals. You must monitor `tx_ready` and `rx_ready`. A high on these status signals indicates the transceiver is out of reset and ready for data transmission and reception.

Note: Deassert the `mgmt_rst_reset` signal of the transceiver reconfiguration controller at the same time as `phy_mgmt_clk_reset` to start calibration.

Note: The PHY IP embedded reset controller is enabled by default in all transceiver PHY IP cores except the Native PHY IP core.

Embedded Reset Controller Signals

The following figure shows the embedded reset controller and signals in the PHY IP instance. These signals reset your transceiver when you use the embedded reset controller.

Figure 3-1: Embedded Reset Controller

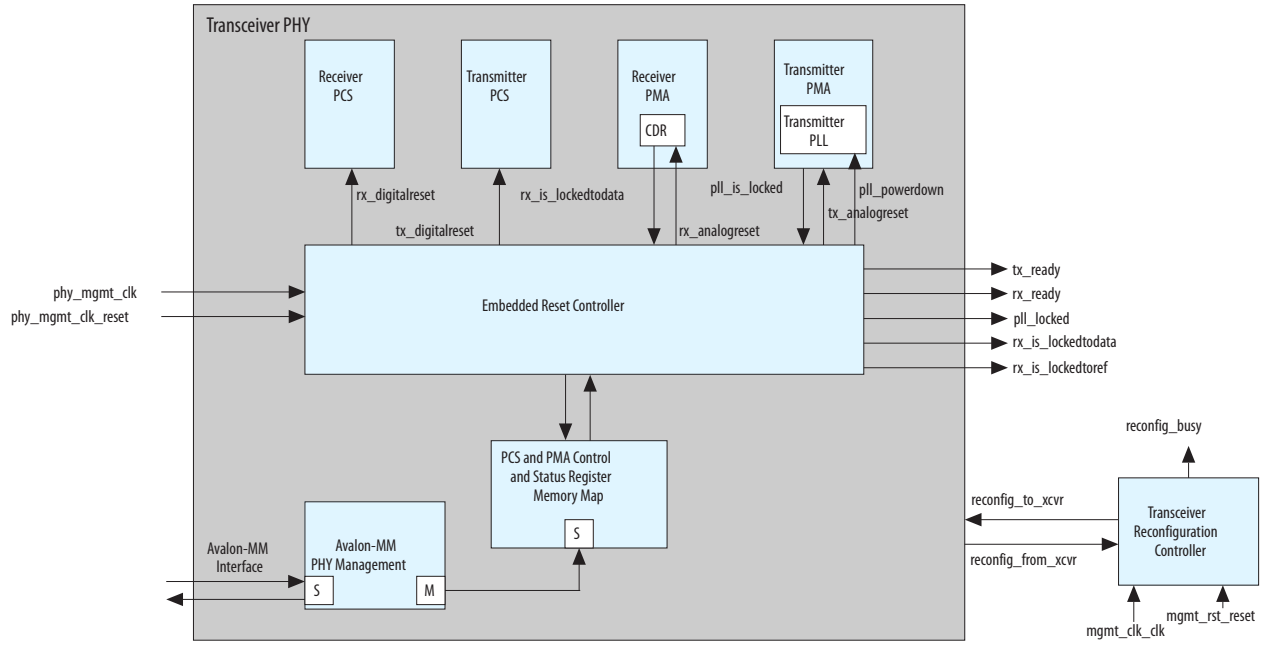


Table 3-1: Embedded Reset Controller Reset Control and Status Signals

Signal Name	Signal	Description
phy_mgmt_clk	Control Input	Clock for the embedded reset controller.
phy_mgmt_clk_reset	Control Input	A high-to-low transition of this asynchronous reset signal initiates the automatic reset sequence control. Hold this signal high to keep the reset signals asserted.
tx_ready	Status Output	A continuous high on this signal indicates that the transmitter (TX) channel is out of reset and is ready for data transmission. This signal is synchronous to phy_mgmt_clk.
rx_ready	Status Output	A continuous high on this signal indicates that the receiver (RX) channel is out of reset and is ready for data reception. This signal is synchronous to phy_mgmt_clk.

Signal Name	Signal	Description
reconfig_busy	Status Output	An output from the Transceiver Reconfiguration Controller block indicates the status of the dynamic reconfiguration controller. At the first mgmt_clk_clk clock cycle after power-up, reconfig_busy remains low. This signal is asserted from the second mgmt_clk_clk clock cycle to indicate that the calibration process is in progress . When the calibration process is completed, the reconfig_busy signal is deasserted. This signal is also routed to the embedded reset controller by the Quartus® II software by embedding the signal in the reconfig_to_xcvr bus between the PHY IP and the Transceiver Reconfiguration Controller.
pll_locked	Status Output	This signal is asserted when the TX PLL achieves lock to the input reference clock. When this signal is asserted high, the embedded reset controller deasserts the tx_digitalreset signal.
rx_is_lockedtodata	Status Output	This signal is an optional output status port. When asserted, this signal indicates that the CDR is locked to the RX data and the CDR has changed from lock-to-reference (LTR) to lock-to-data (LTD) mode.
rx_is_lockedtoref	Status Output	This is an optional output status port. When asserted, this signal indicates that the CDR is locked to the reference clock.
mgmt_clk_clk	Clock	Clock for the Transceiver Reconfiguration Controller. This clock must be stable before releasing mgmt_rst_reset.
mgmt_rst_reset	Reset	Reset for the Transceiver Reconfiguration Controller

Resetting the Transceiver with the PHY IP Embedded Reset Controller During Device Power-Up

Follow this reset sequence to ensure a reliable link initialization after the initial power-up.

The numbers in the following figure correspond to the following numbered list, which guides you through the transceiver reset sequence during device power-up.

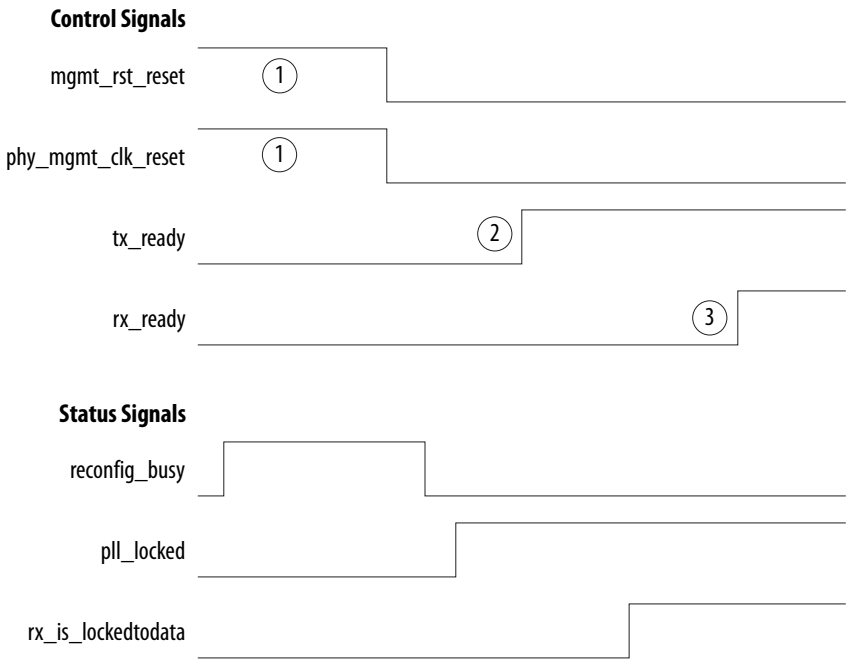
1. During device power-up, mgmt_rst_reset and phy_mgmt_clk_reset must be asserted to initialize the reset sequence. phy_mgmt_clk_reset holds the transceiver blocks in reset and mgmt_rst_reset is required to start the calibration IPs. Both these signals should be held asserted for a minimum of two phy_mgmt_clk clock cycles. If phy_mgmt_clk_reset and mgmt_rst_reset are driven by the same

source, deassert them at the same time. If the two signals are not driven by the same source, `phy_mgmt_clk_reset` must be deasserted before `mgmt_rst_reset`.

2. After the transmitter calibration and reset sequence are complete, the `tx_ready` status signal is asserted and remains asserted to indicate that the transmitter is ready to transmit data.
3. After the receiver calibration and reset sequence are complete, the `rx_ready` status signal is asserted and remains asserted to indicate that the receiver is ready to receive data.

Note: If the `tx_ready` and `rx_ready` signals do not stay asserted, the reset sequence did not complete successfully and the link will be down.

Figure 3-2: Reset Sequence Timing Diagram Using Embedded Reset Controller during Device Power-Up



Resetting the Transceiver with the PHY IP Embedded Reset Controller During Device Operation

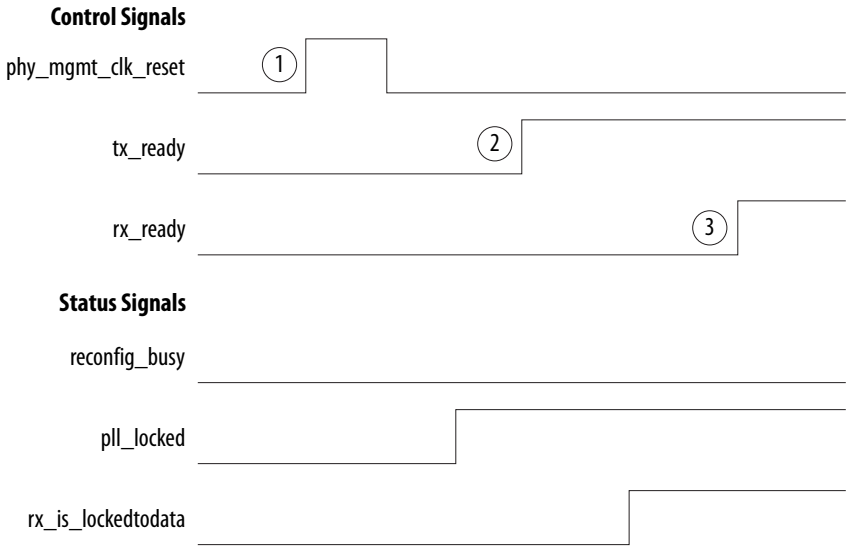
Follow this reset sequence to reset the entire transceiver at any point during the device operation, to re-establishing a link, or after certain dynamic reconfigurations.

The numbers in the following figure correspond to the numbered list, which guides you through the transceiver reset sequence during device operation.

1. Assert `phy_mgmt_clk_reset` for two `phy_mgmt_clk` clock cycles to re-start the entire transceiver reset sequence.
2. After the transmitter reset sequence is complete, the `tx_ready` status signal is asserted and remains asserted to indicate that the transmitter is ready to transmit data.
3. After the receiver reset sequence is complete, the `rx_ready` status signal is asserted and remains asserted to indicate that the receiver is ready to receive data.

Note: If the `tx_ready` and `rx_ready` signals do not stay asserted, the reset sequence did not complete successfully and the link will be down.

Figure 3-3: Reset Sequence Timing Diagram Using Embedded Reset Controller during Device Operation



Note: To reset the transmitter and receiver analog and digital blocks separately without repeating the entire reset sequence, use the Avalon Memory Map registers.

User-Coded Reset Controller

You must implement external reset controller logic (user-coded reset controller) if you disable the embedded reset controller to initialize the transceiver physical coding sublayer (PCS) and physical medium attachment (PMA) blocks.

You can implement a user-coded reset controller with one of the following:

- Using your own Verilog/VHDL code to implement the reset sequence
- Using the Quartus II IP Catalog, which provides a ready-made reset controller IP to place your own Verilog/VHDL code

When using manual mode, you must create a user-coded reset controller to manage the input signals.

Note: You must disable the embedded reset controller before using the user-coded reset controller.

Note: The embedded reset controller can only be disabled for non-protocol transceiver PHY IPs, such as 10GBASE-R PHY, custom PHY, low latency PHY and deterministic latency PHY. Native PHY IP does not have an embedded reset controller, so you must implement your own reset logic.

If you implement your own reset controller, consider the following:

- The user-coded reset controller must be level sensitive (active high)
- The user-coded reset controller does not depend on `phy_mgmt_clk_reset`
- You must provide a clock and reset to the reset controller logic
- The internal signals of the PHY IP embedded reset controller are configured as ports
- You can hold the transceiver channels in reset by asserting the appropriate reset control signals

Related Information

"Transceiver PHY Reset Controller IP Core" chapter of the Altera Transceiver PHY IP Core User Guide.
For information about the transceiver PHY reset controller.

User-Coded Reset Controller Signals

Use the signals in the following figure and table with a user-coded reset controller.

Figure 3-4: Interaction Between the Transceiver PHY Instance, Transceiver Reconfiguration Controller, and the User-Coded Reset Controller

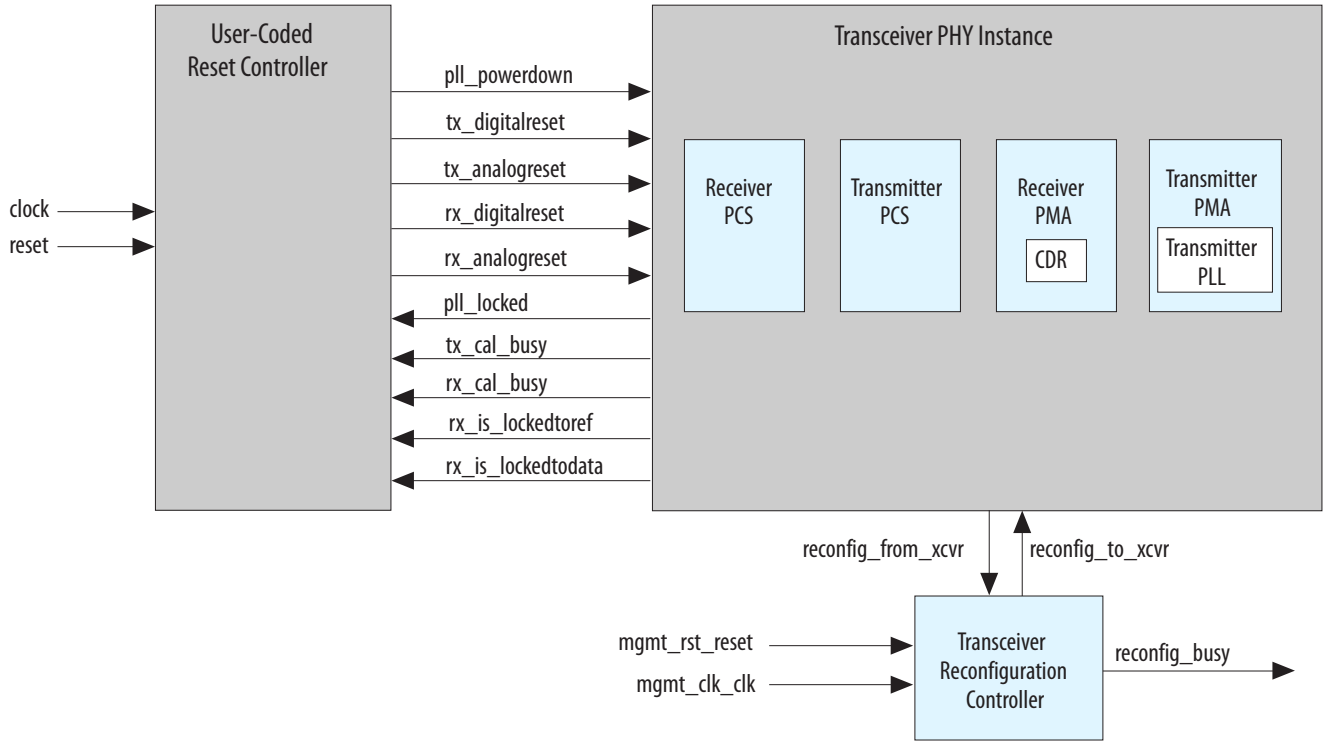


Table 3-2: Signals Used by the Transceiver PHY instance, Transceiver Reconfiguration Controller, and User-Coded Reset Controller

Signal Name	Signal Type	Description
`mgmt_clk_clk`	Clock	Clock for the Transceiver Reconfiguration Controller. This clock must be stable before releasing `mgmt_rst_reset`.

Signal Name	Signal Type	Description
mgmt_rst_reset	Reset	Reset for the Transceiver Reconfiguration Controller
pll_powerdown	Control	Resets the TX PLL when asserted high
tx_analogreset	Control	Resets the TX PMA when asserted high
tx_digitalreset	Control	Resets the TX PCS when asserted high
rx_analogreset	Control	Resets the RX PMA when asserted high
rx_digitalreset	Control	Resets the RX PCS when asserted high
reconfig_busy	Status	A high on this signal indicates that reconfiguration is active
tx_cal_busy	Status	A high on this signal indicates that TX calibration is active
rx_cal_busy	Status	A high on this signal indicates that RX calibration is active
pll_locked	Status	A high on this signal indicates that the TX PLL is locked
rx_is_lockedtoref	Status	A high on this signal indicates that the RX CDR is in the lock to reference (LTR) mode
rx_is_lockedtodata	Status	A high on this signal indicates that the RX CDR is in the lock to data (LTD) mode

Resetting the Transmitter with the User-Coded Reset Controller During Device Power-Up

Follow this reset sequence when designing your User-Coded Reset Controller to ensure a reliable transmitter initialization after the initial power-up.

The numbers in the figure correspond to the following numbered list, which guides you through the transmitter reset sequence during device power-up.

1. To reset the transmitter, begin with:
   - Assert `mgmt_rst_reset` at power-up to start the calibration IPs. Hold `mgmt_rst_reset` active for a minimum of two reset controller clock cycles.
   - Assert and hold `pll_powerdown`, `tx_analogreset`, and `tx_digitalreset` at power-up to reset the transmitter. You can deassert `tx_analogreset` at the same time as `pll_powerdown`.
2. After the transmitter PLL locks, the `pll_locked` status gets asserted after t_{pll_lock}.
3. After the transmitter calibration completes, the `tx_cal_busy` status is deasserted. Depending on the transmitter calibrations, this could happen before or after the `pll_locked` is asserted.
4. Deassert `tx_digitalreset` after the gating conditions occur for a minimum duration of $t_{tx\_digitalreset}$. The gating conditions are:
   - `pll_powerdown` is deasserted
   - `pll_locked` is asserted
   - `tx_cal_busy` is deasserted

The transmitter is out of reset and ready for operation.

Note: During calibration, `pll_locked` might assert and deassert as the calibration IP runs.

Figure 3-5: Reset Sequence Timing Diagram for Transmitter using the User-Coded Reset Controller during Device Power-Up

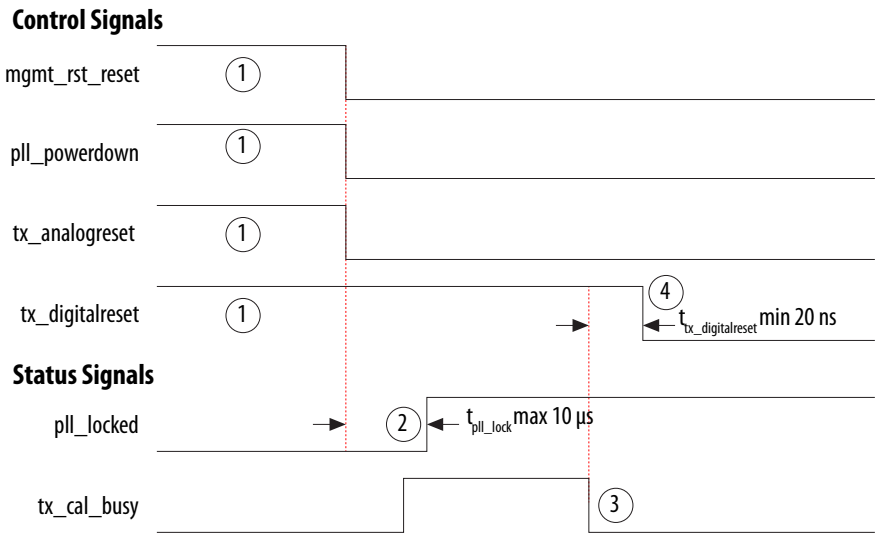


Table 3-3: Guidelines for Resetting the PLL, TX PMA, and TX PCS

To Reset	You Must Reset

PLL	`pll_powerdown`
	`tx_analogreset`
	`tx_digitalreset`

TX PMA	`tx_analogreset` `tx_digitalreset`
TX PCS	`tx_digitalreset`

Resetting the Transmitter with the User-Coded Reset Controller During Device Operation

Follow this reset sequence if you want to reset the PLL, or analog or digital blocks of the transmitter at any point during device operation. This might be necessary for re-establishing a link or after certain dynamic reconfigurations.

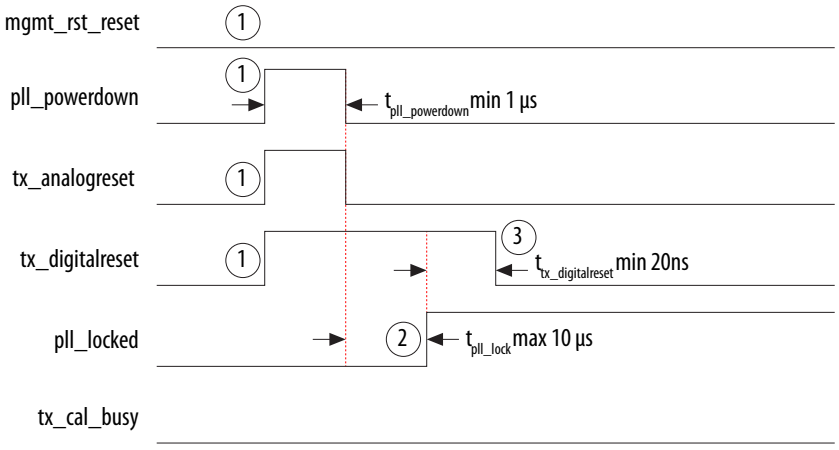
The numbers in the following figure correspond to the following numbered list, which guides you through the transmitter reset sequence during device operation.

1. To reset the transmitter:

- Assert `pll_powerdown`, `tx_analogreset` and `tx_digitalreset`. `tx_digitalreset` must be asserted every time `pll_powerdown` and `tx_analogreset` are asserted to reset the PCS blocks.
- Hold `pll_powerdown` asserted for a minimum duration of $t_{pll\_powerdown}$.
- Deassert `tx_analogreset` at the same time or after `pll_powerdown` is deasserted.

2. After the transmitter PLL locks, the `pll_locked` status is asserted after t_{pll_lock}. While the TX PLL locks, the `pll_locked` status signal may toggle. It is asserted after t_{pll_lock}.
3. Deassert `tx_digitalreset` after a minimum duration of $t_{tx\_digitalreset}$, and after all the gating conditions are removed:
   - `pll_powerdown` is deasserted
   - `pll_locked` is deasserted

Figure 3-6: Reset Sequence Timing Diagram for Transmitter using the User-Coded Reset Controller during Device Operation



Resetting the Receiver with the User-Coded Reset Controller During Device Power-Up Configuration

Follow this reset sequence to ensure a reliable receiver initialization after the initial power-up.

The numbers in the following figure correspond to the following numbered list, which guides you through the receiver reset sequence during device power-up.

1. Assert `mgmt_rst_reset` at power-up to start the calibration IPs. Hold `mgmt_rst_reset` active for a minimum of two `mgmt_clk_clock` cycles. Hold `rx_analogreset` and `rx_digitalreset` active at power-up to hold the receiver in reset. You can deassert them after all the gating conditions are removed.
2. After the receiver calibration completes, the `rx_cal_busy` status is deasserted.
3. Deassert `rx_analogreset` after a minimum duration of $t_{rx\_analogreset}$ after `rx_cal_busy` is deasserted.
4. `rx_is_lockedtodata` is a status signal from the receiver CDR indicating that the CDR is in the lock to data (LTD) mode. Ensure `rx_is_lockedtodata` is asserted and stays asserted for a minimum duration

of t_{LTD} before deasserting `rx_digitalreset`. If `rx_is_lockedtodata` is asserted and toggles, you must wait another additional t_{LTD} duration before deasserting `rx_digitalreset`.

5. Deassert `rx_digitalreset` after a minimum duration of t_{LTD} after `rx_is_lockedtodata` stays asserted. Ensure `rx_analogreset` and `rx_cal_busy` are deasserted before deasserting `rx_digitalreset`.

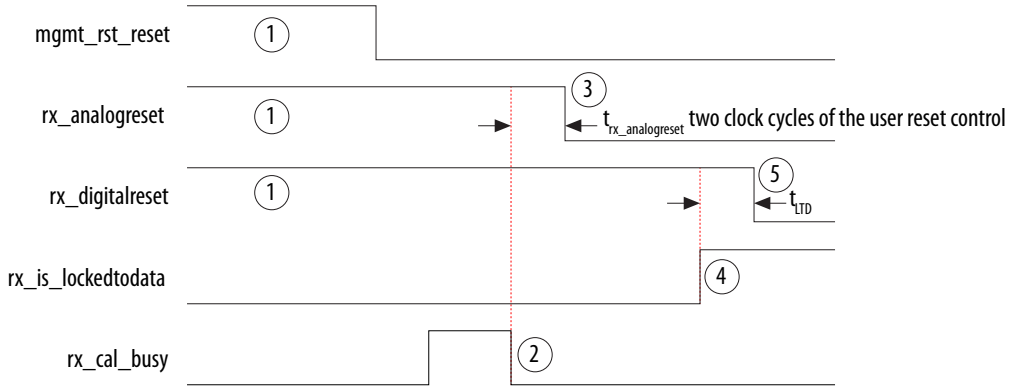
The receiver is now out of reset and ready for operation.

Note: `rx_is_lockedtodata` might toggle when there is no data at the receiver input.

Note: `rx_is_lockedtoref` is a don't care when `rx_is_lockedtodata` is asserted.

Note: `rx_analogreset` must always be followed by `rx_digitalreset`.

Figure 3-7: Reset Sequence Timing Diagram for Receiver using the User-Coded Reset Controller during Device Power-Up



Related Information

Transceiver Architecture in Cyclone V Devices
For information about CDR lock modes.

Resetting the Receiver with the User-Coded Reset Controller During Device Operation

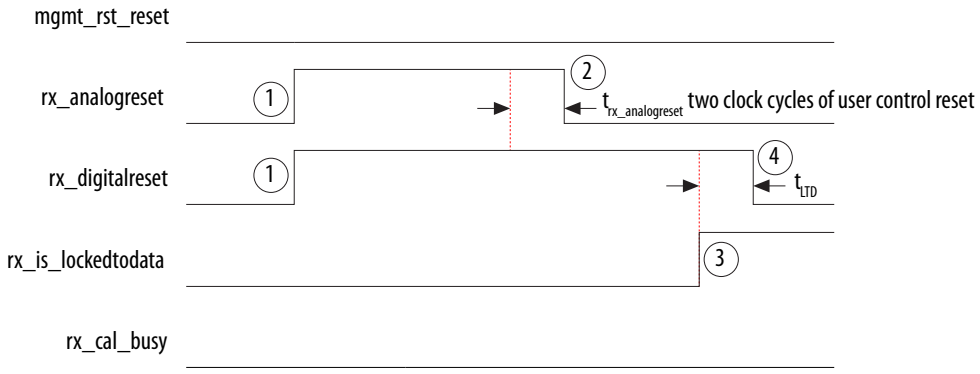
Follow this reset sequence to reset the analog or digital blocks of the receiver at any point during the device operation. This might be necessary for re-establishing a link or after certain dynamic reconfigurations.

The numbers in the following figure correspond to the following numbered list, which guides you through the receiver reset sequence during device operation.

1. Assert `rx_analogreset` and `rx_digitalreset` at any point independently. However, you must assert `rx_digitalreset` every time `rx_analogreset` is asserted to reset the PCS blocks.
2. Deassert `rx_analogreset` after a minimum duration of 40 ns ($t_{rx\_analogreset}$).
3. `rx_is_lockedtodata` is a status signal from the receiver CDR that indicates that the CDR is in the lock to data (LTD) mode. Ensure `rx_is_lockedtodata` is asserted and stays asserted before deasserting `rx_digitalreset`.
4. Deassert `rx_digitalreset` after a minimum duration of t_{LTD} after `rx_is_lockedtodata` stays asserted. Ensure `rx_analogreset` is deasserted.

Note: `rx_is_lockedtodata` might toggle when there is no data at the receiver input. `rx_is_lockedtoref` is a don't care when `rx_is_lockedtodata` is asserted.

Figure 3-8: Reset Sequence Timing Diagram for Receiver using the User-Coded Reset Controller during Device Operation



Related Information

Transceiver Architecture in Cyclone V Devices
For information about CDR lock modes.

Transceiver Reset Using Avalon Memory Map Registers

You can use Memory Map registers within the PHY IP instance to control the reset signals through the Avalon Memory Map interface.

This gives the flexibility of resetting the PLL, and transmitter and receiver analog and digital blocks separately without repeating the entire reset sequence.

Transceiver Reset Control Signals Using Avalon Memory Map Registers

The following table lists the memory map registers for CDR lock mode and channel reset. These signals help you reset your transceiver when you use Memory Map registers within the PHY IP.

Table 3-4: Transceiver Reset Control Using Memory Map Registers

Register Name	Description
pma_rx_set_locktodata	This register is for CDR manual lock mode only. When you set the register to high, the RX CDR PLL is in the lock to data (LTD) mode. The default is low when both registers have the CDR in auto lock mode.
pma_rx_set_locktoref	This register is for CDR manual lock mode only. When you set the register to high, the RX CDR PLL is in the lock to reference (LTR) mode if pma_rx_set_lockedtodata is not asserted. The default is low when both registers have the CDR in auto lock mode.
reset_tx_digital	When you set this register to high, the tx_digitalreset signal is asserted in every channel that is enabled for reset control through the reset_ch_bitmask register. To deassert the tx_digitalreset signal, set the reset_tx_digital register to 0.
reset_rx_analog	When you set this register to high, the rx_analogreset signal is asserted in every channel that is enabled for reset control through the reset_ch_bitmask register. To deassert the rx_analogreset signal, set the reset_rx_analog register to 0.
reset_rx_digital	When you set this register to high, the rx_digitalreset signal is asserted in every channel that is enabled for reset control through the reset_ch_bitmask register. To deassert the rx_digitalreset signal, set the reset_rx_digital register to 0.
reset_ch_bitmask	The registers provide an option to enable or disable certain channels in a PHY IP instance for reset control. By default, all channels in a PHY IP instance are enabled for reset control.
pll_powerdown	When asserted, the TX phase-locked loop (PLL) is turned off.

Related Information

Altera Transceiver PHY IP Core User Guide
For information about register addresses.

Clock Data Recovery in Manual Lock Mode

Use the clock data recovery (CDR) manual lock mode to override the default CDR automatic lock mode depending on your design requirements.

Related Information

"Transceiver PHY Reset Controller IP Core" chapter of the Altera Transceiver PHY IP Core User Guide.

Refer to the description of the `rx_digitalreset` signal in the "Top-Level Signals" table for information about using the manual lock mode.

Control Settings for CDR Manual Lock Mode

Use the following control settings to set the CDR lock mode:

Table 3-5: Control Settings for the CDR in Manual Lock Mode

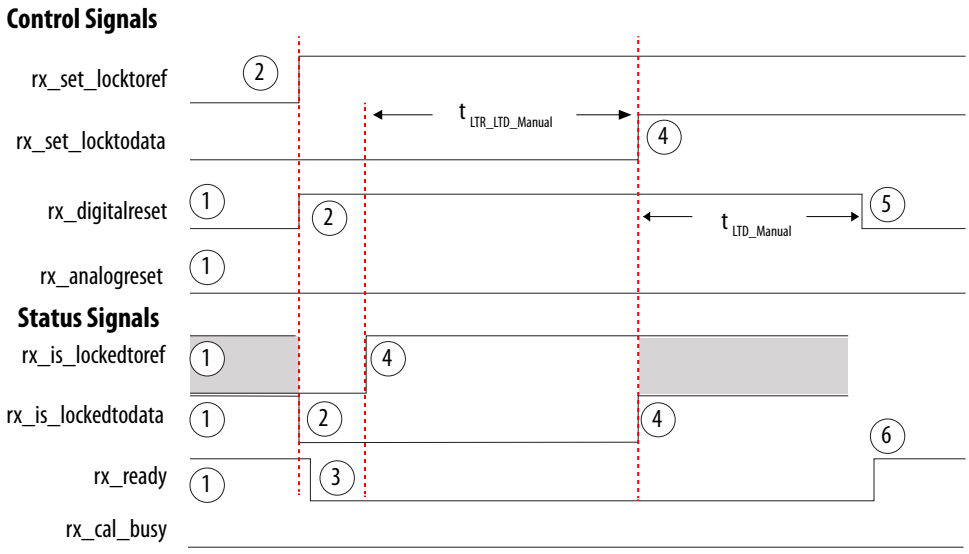
rx_set_locktoref	rx_set_locktodata	CDR Lock Mode
0	0	Automatic
1	0	Manual-RX CDR LTR
X	1	Manual-RX CDR LTD

Resetting the Transceiver in CDR Manual Lock Mode

The numbers in this list correspond to the numbers in the following figure, which guides you through the steps to put the CDR in manual lock mode.

1. Make sure that the calibration is complete (`rx_cal_busy` is low) and the transceiver goes through the initial reset sequence. The `rx_digitalreset` and `rx_analogreset` signals should be low. The `rx_is_lockedtoref` is a don't care and can be either high or low. The `rx_is_lockedtodata` and `rx_ready` signals should be high, indicating that the transceiver is out of reset. Alternatively, you can start directly with the CDR in manual lock mode after the calibration is complete.
2. Assert the `rx_set_locktoref` signal high to switch the CDR to the lock-to-reference mode. The `rx_is_lockedtodata` status signal is deasserted. Assert the `rx_digitalreset` signal high at the same time or after `rx_set_lockedtoref` is asserted if you use the user-coded reset. When the Transceiver PHY reset controller is used, the `rx_digitalreset` is automatically asserted.
3. After the `rx_digitalreset` signal gets asserted, the `rx_ready` status signal is deasserted.
4. Assert the `rx_set_locktodata` signal high, $t_{LTR\_LTD\_Manual}$ (minimum 15 µs) after the CDR is locked to reference. `rx_is_locktoref` should be high and stable for a minimum $t_{LTR\_LTD\_Manual}$ (15 µs), before asserting `rx_set_locktodata`. This is required to filter spurious glitches on `rx_is_lockedtoref`. The `rx_is_lockedtodata` status signal gets asserted, which indicates that the CDR is now set to LTD mode.
   The `rx_is_lockedtoref` status signal can be a high or low and can be ignored after asserting `rx_set_locktodata` high after the CDR is locked to reference.
5. Deassert the `rx_digitalreset` signal after a minimum of t_{LTD_Manual} (4 µs).
6. If you are using the Transceiver PHY Reset Controller, the `rx_ready` status signal gets asserted after the `rx_digitalreset` signal is deasserted. This indicates that the receiver is now ready to receive data with the CDR in manual mode.

Figure 3-9: Reset Sequence Timing Diagram for Receiver when CDR is in Manual Lock Mode



Resetting the Transceiver During Dynamic Reconfiguration

Reset is required for transceiver during dynamic reconfiguration except in the PMA Analog Control Reconfiguration mode.

In general, follow these guidelines when dynamically reconfiguring the transceiver:

1. Hold the targeted channel and PLL in the reset state before dynamic reconfiguration starts.
2. Repeat the sequence as needed after dynamic reconfiguration is complete, which is indicated by deassertion of the `reconfig_busy`, `tx_cal_busy`, `rx_cal_busy` signals.

Guidelines for Dynamic Reconfiguration if Transmitter Duty Cycle Distortion Calibration is Required During Device Operation

If transmitter duty cycle distortion calibration is required during device operation, ensure the general guidelines for transceiver dynamic reconfiguration are followed. Additionally, use the following recommendations:

1. Do not connect `tx_cal_busy` to the transceiver Reset Controller IP.
2. Disable the embedded reset controller and use an external reset controller.

Note: If channel reconfiguration is required before TX DCD calibration, ensure the following:

- The TX PLL, TX channel, and Transceiver Reconfiguration Controller blocks must not be in the reset state during TX DCD calibration. Ensure the following signals are not asserted during TX DCD calibration:
  - `pll_powerdown`
  - `tx_digitalreset`
  - `tx_analogreset`
  - `mgmt_rst_reset`

Repeat the reset sequence when TX DCD calibration is complete.

Transceiver Blocks Affected by the Reset and Powerdown Signals

The following table lists blocks that are affected by specific reset and powerdown signals.

Table 3-6: Transceiver Blocks Affected

Transceiver Block	pll_ powerdown	rx_digital-reset	rx_analogr-eset	tx_digitalreset	tx_analogreset
PLL					
CMU PLL	Yes	—	—	—	—
Receiver Standard PCS					
Receiver Word Aligner	—	Yes	—	—	—
Receiver Deskew FIFO	—	Yes	—	—	—
Receiver Rate Match FIFO	—	Yes	—	—	—
Receiver 8B/10B Decoder	—	Yes	—	—	—
Receiver Byte Deserializer	—	Yes	—	—	—
Receiver Byte Ordering	—	Yes	—	—	—
Receiver Phase Compensa-tion FIFO	—	Yes	—	—	—
Receiver PMA					
Receiver Buffer	—	—	Yes	—	—
Receiver CDR	—	—	Yes	—	—
Receiver Deserializer	—	—	Yes	—	—
Transmitter Standard PCS					
Transmitter Phase Compensation FIFO	—	—	—	Yes	—
Byte Serializer	—	—	—	Yes	—
8B/10B Encoder	—	—	—	Yes	—

Transceiver Block	pll_ powerdown	rx_digital-reset	rx_analogr-eset	tx_digitalreset	tx_analogreset
Transmitter Bit-Slip	—	—	—	Yes	—
Transmitter PMA					
Transmitter Central/Local Clock Divider	—	—	—	—	Yes
Serializer	—	—	—	—	Yes
Transmitter Buffer	—	—	—	—	Yes

Transceiver Power-Down

To maximize power savings, enable PMA hard power-down across all channels on a side of the device where you do not use the transceivers.

The hard power-down granularity control of the transceiver PMA is per side. To enable PMA hard power-down on the left or right side of the device, ground the transceiver power supply of the respective side.

VCCE_GXBL and VCCL_GXBL must be connected either to the required supply or to GND. The VCCH_GXBL pin must always be powered.

Related Information

Cyclone V Device Datasheet
For information about the transceiver power supply operating conditions of the left and right side of Cyclone V devices.

Document Revision History

Date	Version	Changes
January 2016	2016.01.28	• Updated the "Transceiver Power-Down" section about the hard power-down granularity control of the transceiver PMA per side. • Changed heading "User-Controlled Reset Controller" to "User-Coded Reset Controller". • Added statement about using manual mode to the "User-Coded Reset Controller" section. • Added a link to the Related Links in the "Clock Data Recovery in Manual Lock Mode" section.

Date	Version	Changes
September 2014	2014.09.30	• Added information about using signals to the "Resetting the Transceiver with the PHY IP Embedded Reset Controller during Device Power-Up" section. • Changed heading "User-Controlled Reset Controller" to "User-Coded Reset Controller". • Added statement about using manual mode to the "User-Coded Reset Controller" section. • Added a link to the Related Links in the "Clock Data Recovery in Manual Lock Mode" section.
May 2013	2013.05.06	• Updated the guidelines for Dynamic Reconfiguration if TX DCD Calibration is required during device operation. • Added link to the known document issues in the Knowledge Base.
November 2012	2012.11.19	• Rewritten and reorganized content, and updated template • Updated reset sequence procedures • Included sequences for resetting transceiver during device operation
November 2011	1.1	• Added “User-Controlled Reset Controller” section. • Updated Figure 3–1 and Table 3–1.
August 2011	1.0	Initial release.

Transceiver Protocol Configurations in Cyclone V Devices

4

2016.01.28

CV-53004

Subscribe Send Feedback

The dedicated transceiver physical coding sublayer (PCS) and physical medium attachment (PMA) circuitry supports the following communication protocols.

Table 4-1: Transceiver PCS Features for Cyclone V Devices

PCS Support	Data Rates (Gbps)	Transmitter Datapath	Receiver Datapath
PCI Express® (PCIe®) Gen1 (x1, x2, and x4) and Gen2 (x1, x2, and x4)	2.5, 5	PIPE (PHY Interface for the PCIe architecture) interface to the PCIe Hard IP	PIPE interface to the PCIe Hard IP
Gbps Ethernet (GbE)	1.25, 3.125	The same as custom single- and double-width modes	The same as custom single- and double-width modes, plus the rate match FIFO
Serial Digital Interface (SDI)	0.27(10), 1.485, and 2.97	Phase compensation FIFO and byte serializer	Phase compensation FIFO and byte deserializer
SATA, SAS	1.5 and 3.0	Phase compensation FIFO, byte serializer, and 8B/10B encoder	Phase compensation FIFO, byte deserializer, word aligner, and 8B/10B decoder
Common Public Radio Interface (CPRI)	0.6144, 1.2288, 2.4576, 3.072, 4.9152, 6.144 (11)	The same as custom single- and double-width modes, plus the transmitter (TX) deterministic latency	The same as custom single- and double-width modes, plus the receiver (RX) deterministic latency

(10) The 0.27 gigabits per second (Gbps) data rate is supported using oversampling user logic that must be implemented by the user in the FPGA core.

(11) Cyclone V GT devices support data rates greater than 5.0 Gbps only in CPRI.

PCS Support	Data Rates (Gbps)	Transmitter Datapath	Receiver Datapath
OBSAI	0.768, 1.536, 3.072	The same as custom single- and double-width modes, plus the TX deterministic latency	The same as custom single- and double-width modes, plus the RX deterministic latency
XAUI	3.125	Implemented using soft PCS	Implemented using soft PCS

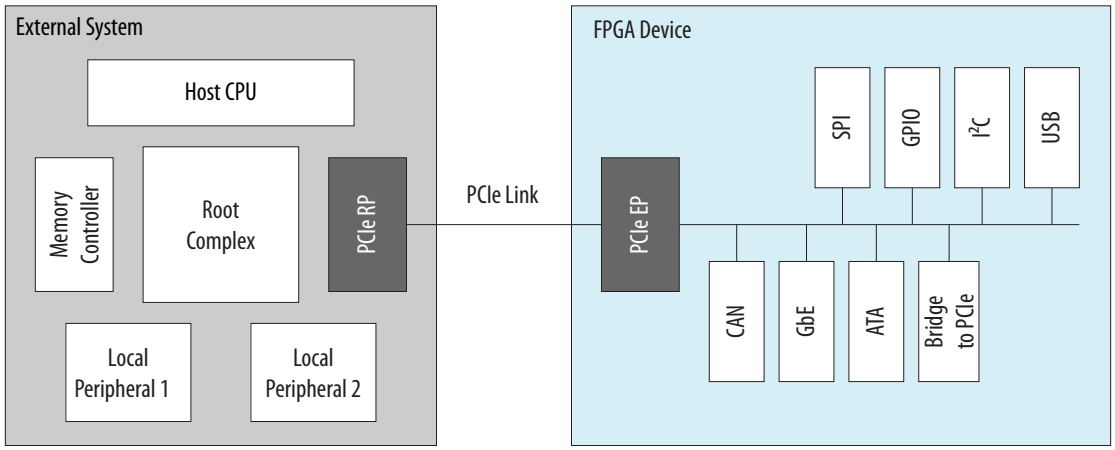
Related Information

- **Cyclone V Device Handbook: Known Issues**
  Lists the planned updates to the *Cyclone V Device Handbook* chapters.
- **Use this chapter along with the Altera Transceiver PHY IP Core User Guide.**
- **Upcoming Cyclone V Device Features**

PCI Express

The Cyclone V devices have PCIe Hard IP that is designed for performance, ease-of-use, and increased functionality. The Hard IP consists of the media access control (MAC) lane, data link, and transaction layers. The PCIe Hard IP supports the PCIe Gen1 end point and root port up to x4 lane configurations. The PCIe endpoint support includes multifunction support for up to eight functions and Gen2 x4 lane configurations.

Figure 4-1: PCIe Multifunction for Cyclone V Devices

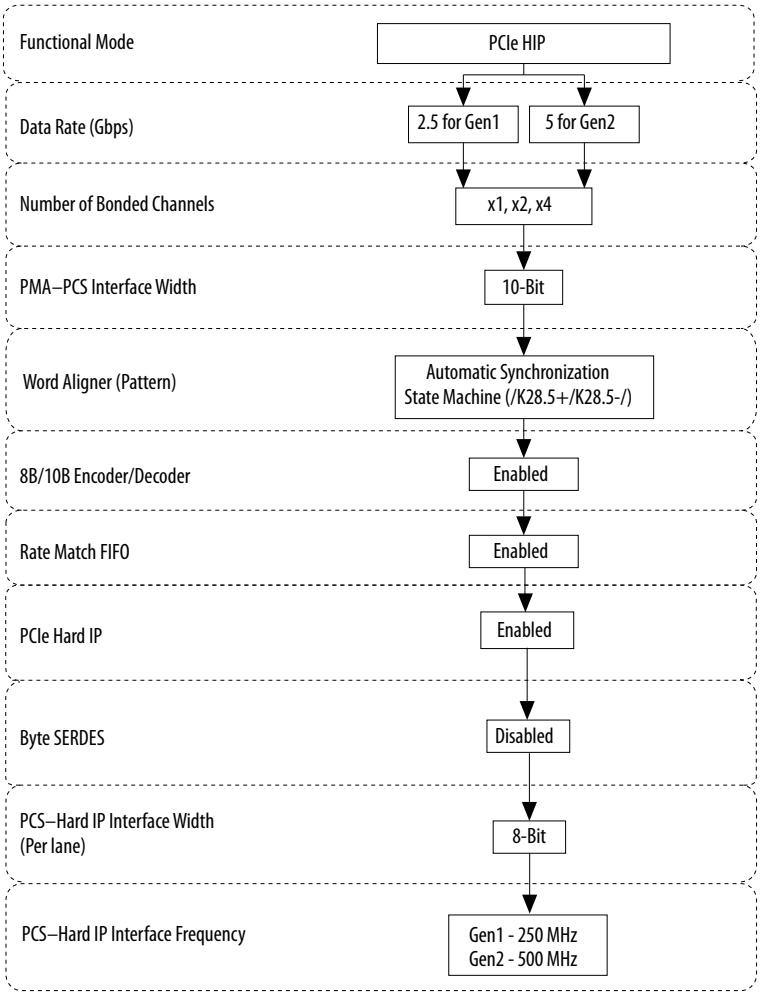


The Cyclone V PCIe Hard IP operates independently from the core logic, which allows the PCIe link to wake up and complete link training in less than 100 ms while the Cyclone V device completes loading the programming file for the rest of the device.

In addition, the Cyclone V device PCIe Hard IP has improved end-to-end datapath protection using error correction code (ECC).

PCIe Transceiver Datapath

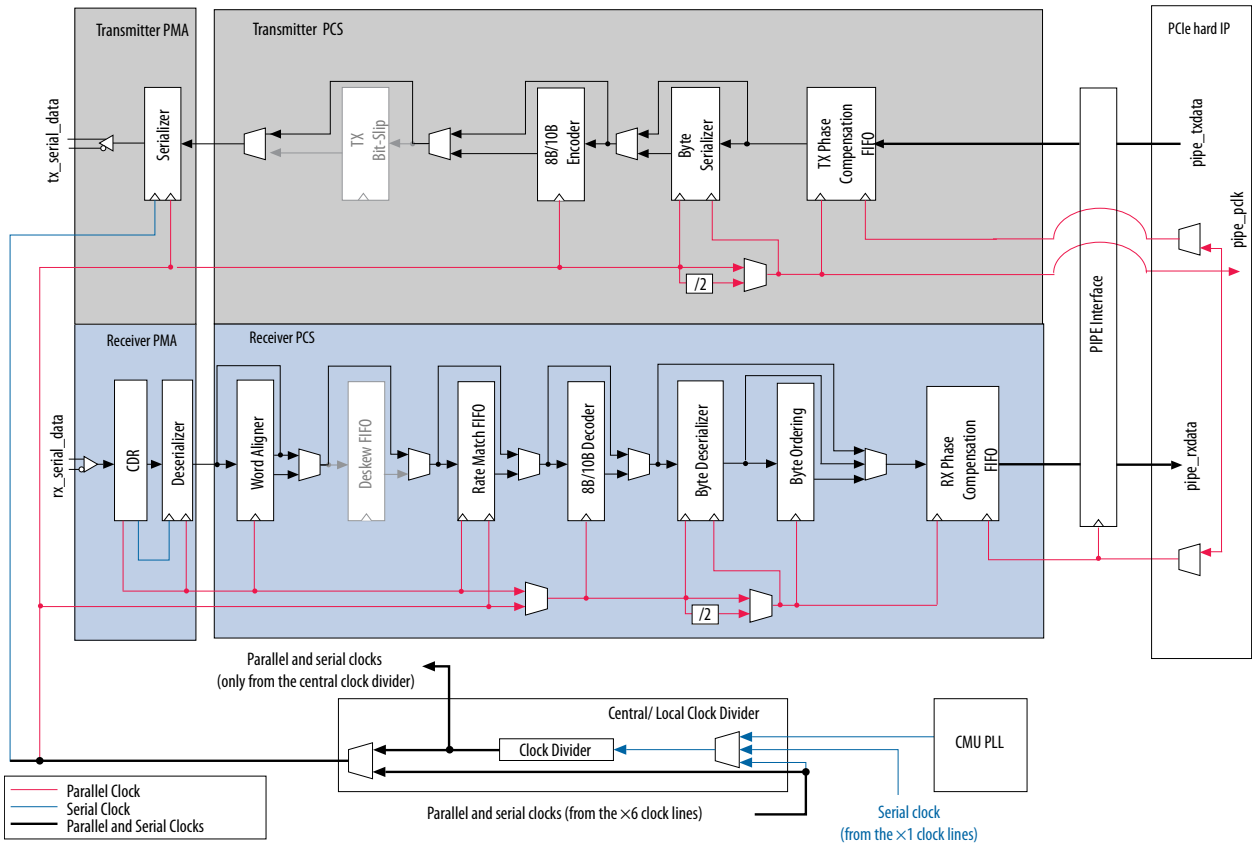
Figure 4-2: Transceivers in a PCIe Hard IP Configuration



Note: Refer to the *Cyclone V Device Datasheet* for the `mgmt_clk_clk` frequency specification when PCIe HIP is used.

Transceiver Channel Datapath

Figure 4-3: Transceiver Channel Datapath in a PIPE Configuration



Related Information

- **Transceiver Architecture in Cyclone V Devices**
- **Cyclone V Device Datasheet**

PCIe Supported Features

The PIPE configuration for the 2.5 Gbps (Gen1) and 5 Gbps (Gen2) data rates supports these features:

- PCIe-compliant synchronization state machine
- x1 and x4 link configurations
- ±300 parts per million (ppm)—total 600 ppm—clock rate compensation
- 8-bit FPGA fabric–transceiver interface
- 16-bit FPGA fabric–transceiver interface
- Transmitter buffer electrical idle
- Receiver detection
- 8B/10B encoder disparity control when transmitting compliance pattern
- Power state management (Electrical Idle only)
- Receiver status encoding

PIPE Interface

In a PIPE configuration, each channel has a PIPE interface block that transfers data, control, and status signals between the PHY-MAC layer and the transceiver channel PCS and PMA blocks.

Note: The PIPE interface block is used in a PIPE configuration and cannot be bypassed.

In addition to transferring data, control, and status signals between the PHY-MAC layer and the transceiver, the PIPE interface block implements the following functions that are required in a PCIe-compliant physical layer device:

- Forces the transmitter buffer into an electrical idle state
- Initiates the receiver detect sequence
- Controls the 8B/10B encoder disparity when transmitting a compliance pattern
- Manages the PCIe power states (Electrical Idle only)
- Indicates the completion of various PHY functions, such as receiver detection and power state transitions on the `pipe_phystatus` signal
- Encodes the receiver status and error conditions on the `pipe_rxstatus[2:0]` signal, as specified in the PCIe specification

Transmitter Electrical Idle Generation

The PIPE interface block places the channel transmitter buffer in an electrical idle state when the electrical idle input signal is asserted.

During electrical idle, the transmitter buffer differential and common configuration output voltage levels are compliant to the PCIe Base Specification 2.1 for the PCIe Gen2 data rate.

The PCIe specification requires that the transmitter buffer be placed in electrical idle in certain power states.

Power State Management

The PCIe specification defines four power states: P0, P0s, P1, and P2.

The physical layer device must support these power states to minimize power consumption:

- P0 is the normal operating state during which packet data is transferred on the PCIe link.
- P0s, P1, and P2 are low-power states into which the physical layer must transition as directed by the PHY-MAC layer to minimize power consumption.

The PIPE interface in the transceivers provides an input port for each transceiver channel configured in a PIPE configuration.

Note: When transitioning from the P0 power state to lower power states (P0s, P1, and P2), the PCIe specification requires that the physical layer device implements power saving measures. The transceivers do not implement these power saving measures except to place the transmitter buffer in electrical idle in the lower power states.

8B/10B Encoder Usage for Compliance Pattern Transmission Support

The PCIe transmitter transmits a compliance pattern when the Link Training and Status State Machine (LTSSM) enters a polling compliance substate. The polling compliance substate assesses if the transmitter is electrically compliant with the PCIe voltage and timing specifications.

Receiver Status

The PCIe specification requires that the PHY encode the receiver status on a 3-bit status signal (`pipe_rxstatus[2:0]`).

This status signal is used by the PHY-MAC layer for its operation. The PIPE interface block receives the status signals from the transceiver channel PCS and PMA blocks, and encodes the status on the `pipe_rxstatus[2:0]` signal to the FPGA fabric. The encoding of the status signals on the `pipe_rxstatus[2:0]` signal is compliant with the PCIe specification.

Receiver Detection

The PIPE interface block in Cyclone V transceivers provides an input signal (`pipe_txdetectrx_loopback`) for the receiver detect operation that is required by the PCIe protocol during the detect substate of the LTSSM.

When the `pipe_txdetectrx_loopback` signal is asserted in the P1 power state, the PCIe interface block sends a command signal to the transmitter buffer in that channel to initiate a receiver detect sequence. In the P1 power state, the transmitter buffer must always be in the electrical idle state.

After receiving this command signal, the receiver detect circuitry creates a step voltage at the output of the transmitter buffer. If an active receiver that complies with the PCIe input impedance requirements is present at the far end, the time constant of the step voltage on the trace is higher than if the receiver is not present. The receiver detect circuitry monitors the time constant of the step signal that is seen on the trace to determine if a receiver was detected. The receiver detect circuitry monitor requires a 125-MHz clock for operation that you must drive on the `fixedclk` port.

Note: For the receiver detect circuitry to function reliably, the AC-coupling capacitor on the serial link and the receiver termination values used in your system must be compliant with the PCIe Base Specification 2.1.

The PCI Express PHY (PIPE) IP core provides a 1-bit PHY status (`pipe_phystatus`) and a 3-bit receiver status signal (`pipe_rxstatus[2:0]`) to indicate whether a receiver was detected or not, in accordance to the PIPE specifications.

Clock Rate Compensation Up to ±300 ppm

In compliance with the PCIe protocol, the receiver channels are equipped with a rate match FIFO to compensate for the small clock frequency differences of up to ± 300 ppm between the upstream transmitter and local receiver clocks.

Related Information

Transceiver Architecture in Cyclone V Devices

PCIe Reverse Parallel Loopback

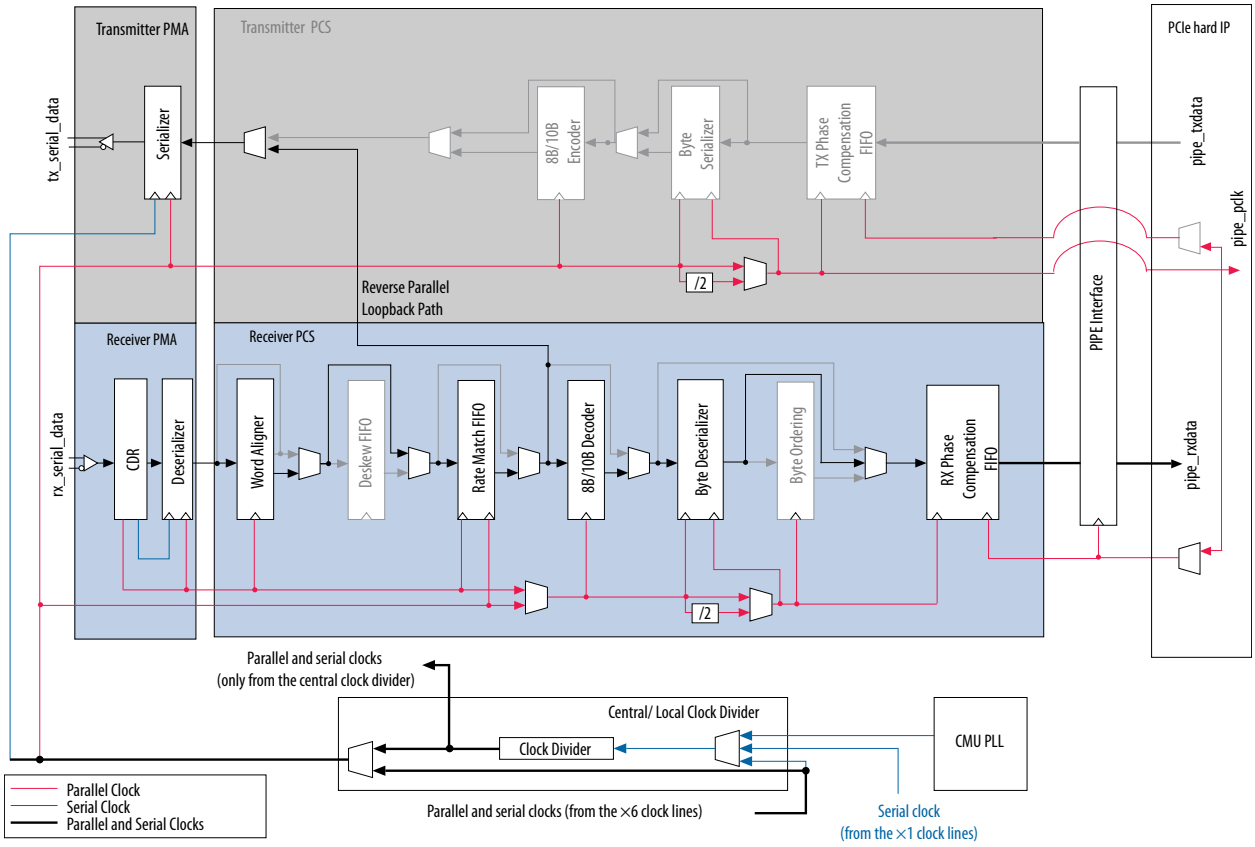
The PCIe reverse parallel loopback is only available in the PCIe functional configuration for the Gen1 data rate. The received serial data passes through the receiver CDR, deserializer, word aligner, and rate matching FIFO buffer. It is then looped back to the transmitter serializer and transmitted out through the transmitter buffer. The received data is also available to the FPGA fabric through the port.

PCIe reverse parallel loopback mode is compliant with PCIe specification 2.1.

Cyclone V devices provide the `pipe_txdetectrx_loopback` input signal to enable this loopback mode. If the `pipe_txdetectrx_loopback` signal is asserted in the P1 power state, receiver detection is performed. If the signal is asserted in the P0 power state, reverse parallel loopback is performed.

Note: The PCIe reverse parallel loopback is the only loopback option that is supported in PIPE configurations.

Figure 4-4: PIPE Reverse Parallel Loopback Mode Datapath



PCIe Supported Configurations and Placement Guidelines

Placement by the Quartus II software may vary with design and device. The following figures show examples of transceiver channel and PCIe Hard IP block locations, supported x1, x2, and x4 bonding configurations, and channel placement guidelines. The Quartus II software automatically places the CMU PLL in a channel different from that of the data channels.

Note: This section shows the supported PCIe channel placement if you use both the top and bottom PCIe Hard IP blocks in the device separately.

The following guidelines apply to all channel placements:

- The CMU PLL requires its own channel and must be placed on channel 1 or channel 4
- The PCIe channels must be contiguous within the transceiver bank
- Lane 0 of the PCIe must be placed on channel 0 or channel 5

In the following figures, channels shaded in blue provide the high-speed serial clock. Channels shaded in gray are data channels.

Figure 4-5: 12 Transceiver Channels and 2 PCIe HIP Blocks with PCIe x2 and x4 Channel Placement

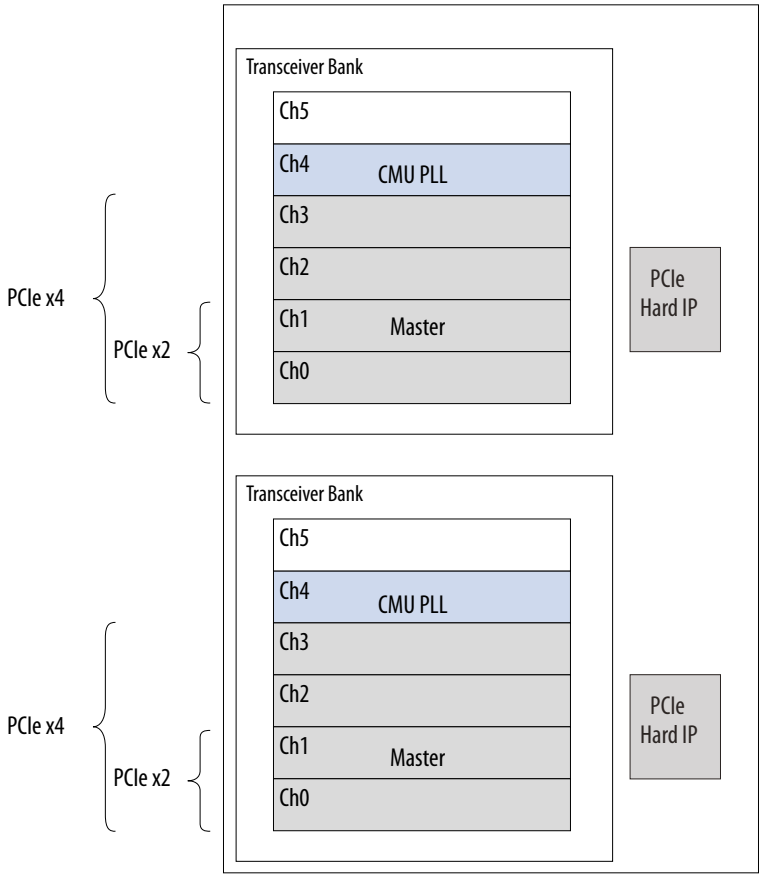


Figure 4-6: 12 Transceiver Channels and 2 PCIe HIP Blocks with PCIe x1 Channel Placement

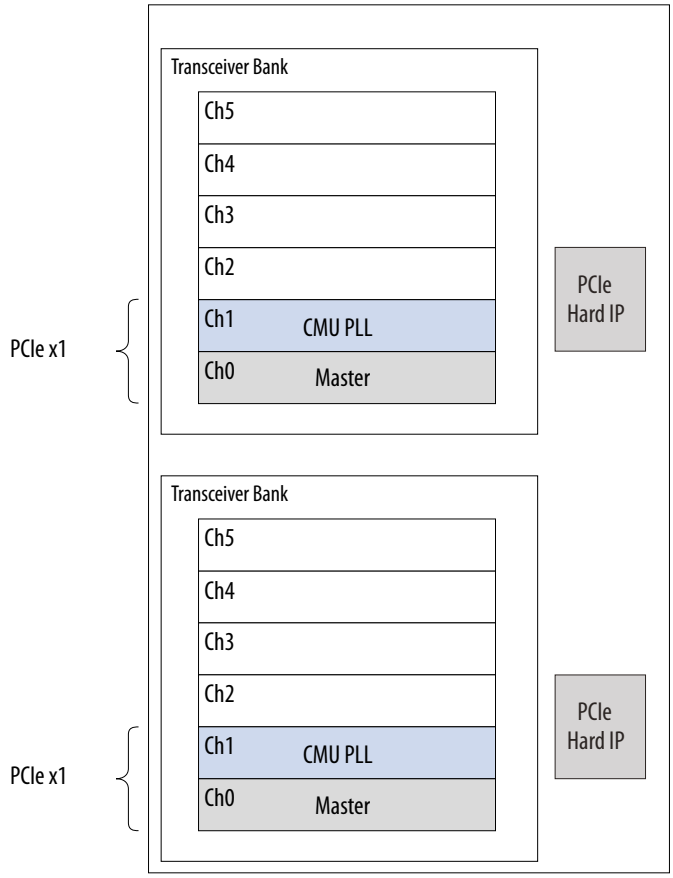


Figure 4-7: 9 Transceiver Channels and 2 PCIe HIP Blocks with PCIe x2 and x4 Channel Placement

The grayed out PCIe Hard IP block is not used in this example

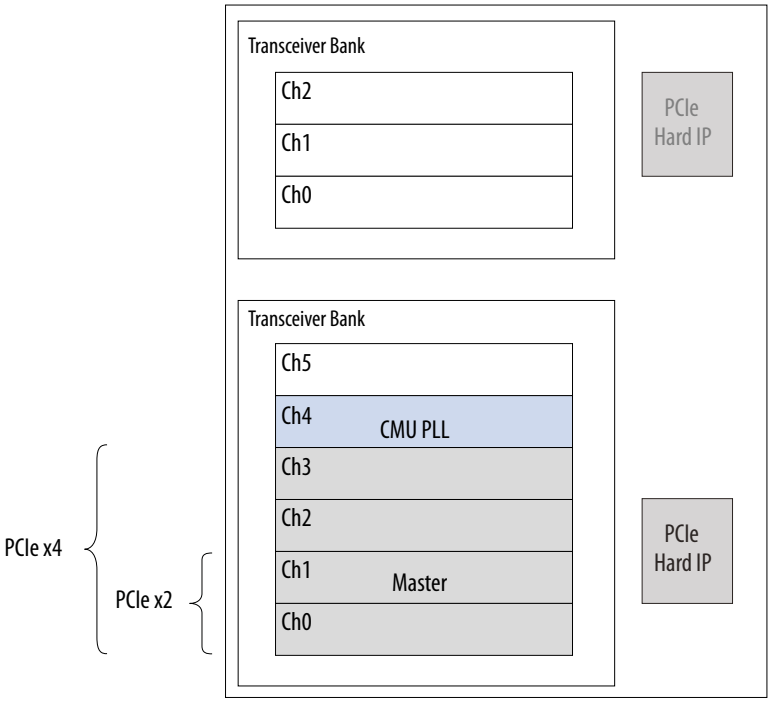


Figure 4-8: 9 Transceiver Channels and 2 PCIe HIP Blocks with PCIe x1 Channel Placement

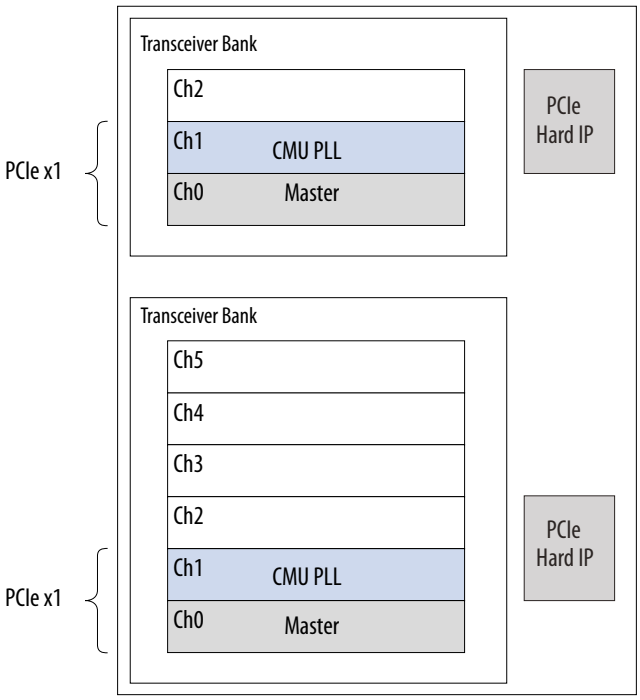


Figure 4-9: 6 Transceiver Channels and 1 PCIe HIP Blocks with PCIe x2 and x4 Channel Placement

The grayed out PCIe Hard IP block is not used in this example.

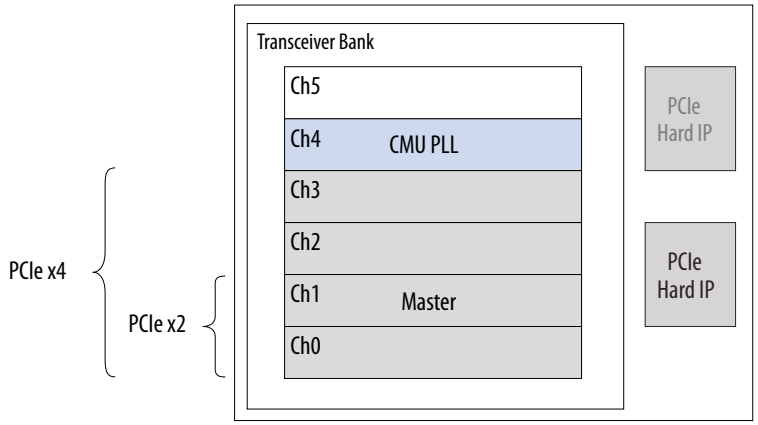


Figure 4-10: 6 Transceiver Channels and 2 PCIe HIP with PCIe x1 Channel Placement

The grayed out PCIe Hard IP block is not used in this example.

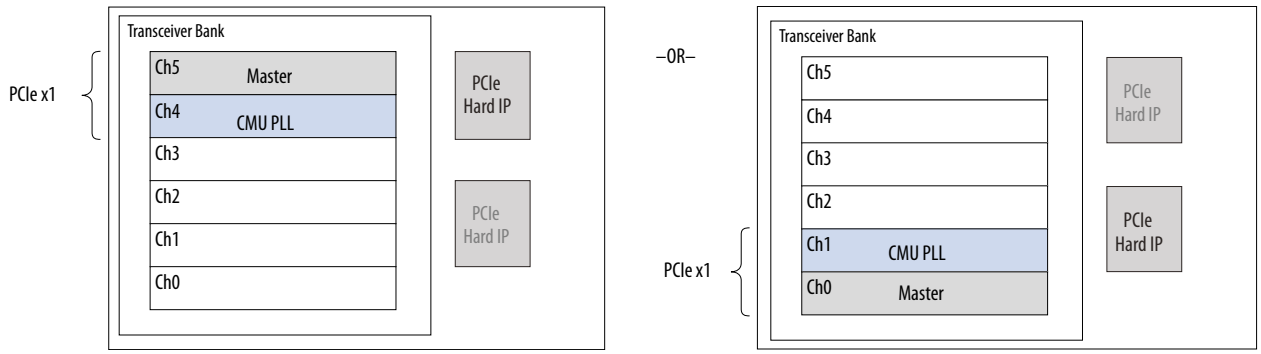
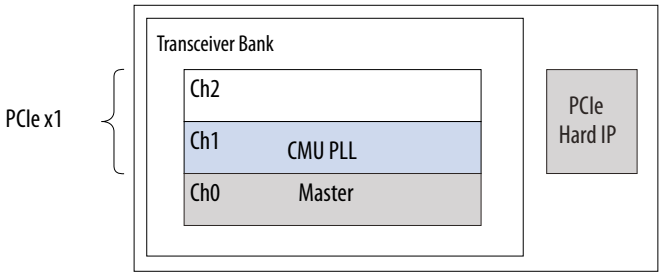


Figure 4-11: 3 Transceiver Channels and 1 PCIe HIP Blocks with PCIe x1 Channel Placement



For PCIe Gen1 and Gen2, there are restrictions on the achievable x1 and x4 bonding configurations if you intend to use both top and bottom Hard IP blocks in the device.

Table 4-2: Hard IP Configurations for PCIe Gen1 and Gen2

The following table lists the configurations allowed for each Cyclone V device when you use both PCIe Hard IP blocks on the top and bottom transceiver banks. Support will vary by the number of transceiver channels in a device.

Top PCIe Hard IP	Bottom PCIe Hard IP	5CGXC4, 5CGXC5, 5CGTD5, 5CSXC5, 5CSTD5	5CGXC7, 5CGTD7, 5CSXC6, 5CSTD6	5CGXC9, 5CGTD9
x1	x1	Yes	Yes	Yes
	x2	No	Yes	Yes
	x4	No	Yes	Yes
x2	x1	No	No	Yes
	x2	No	No	Yes
	x4	No	No	Yes
x4	x1	No	No	Yes
	x2	No	No	Yes
	x4	No	No	Yes

Note: Not all devices listed in the above table have two Hard IP blocks. Refer to the Altera Product Selector for more details.

For full duplex transceiver channels, the following table lists the maximum number of data channels that can be enabled to ensure the channels meet the PCIe Gen2 Transmit Jitter Specification. Follow this recommendation when planning channel placement for PCIe Gen2 using Cyclone V GT or Cyclone V ST device variants.

Table 4-3: Recommended Channel Placement for Full Duplex Transceiver Channels for PCIe Gen2

CMU channels are not counted as data channels.

Device	Maximum Channels Utilization
5CGTD7F672, 5CGTD7F896, 5CGTD9F672, 5CSTD5F896, 5CSTD6F896	6
5CGTD9F896, 5CGTD9F1152	8

Related Information

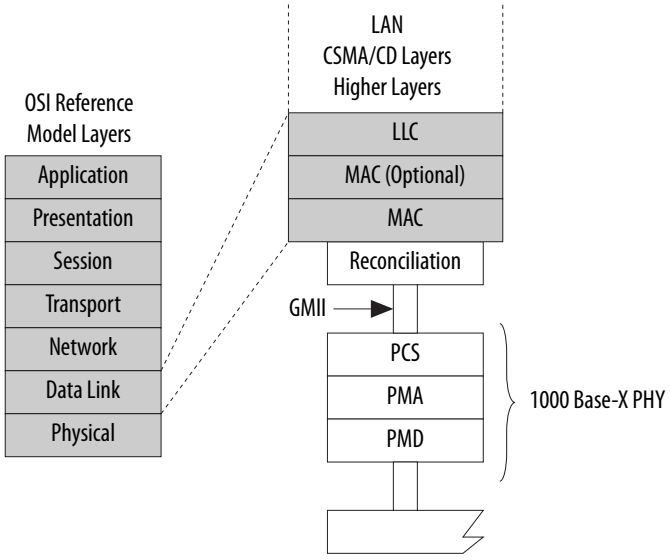
- **Transceiver Architecture in Cyclone V Devices**
- **Altera Product Selector**
  Provides the latest information about Altera products.

Gigabit Ethernet

The IEEE 802.3 specification defines the 1000BASE-X PHY as an intermediate, or transition layer that interfaces various physical media with the MAC in a gigabit ethernet (GbE) system, shielding the MAC layer from the specific nature of the underlying medium. The 1000BASE-X PHY is divided into the PCS, PMA, and PMD sublayers.

The PCS sublayer interfaces with the MAC through the gigabit media independent interface (GMII). The 1000BASE-X PHY defines a physical interface data rate of 1 Gbps and 2.5 Gbps.

Figure 4-12: 1000BASE-X PHY in a GbE OSI Reference Model



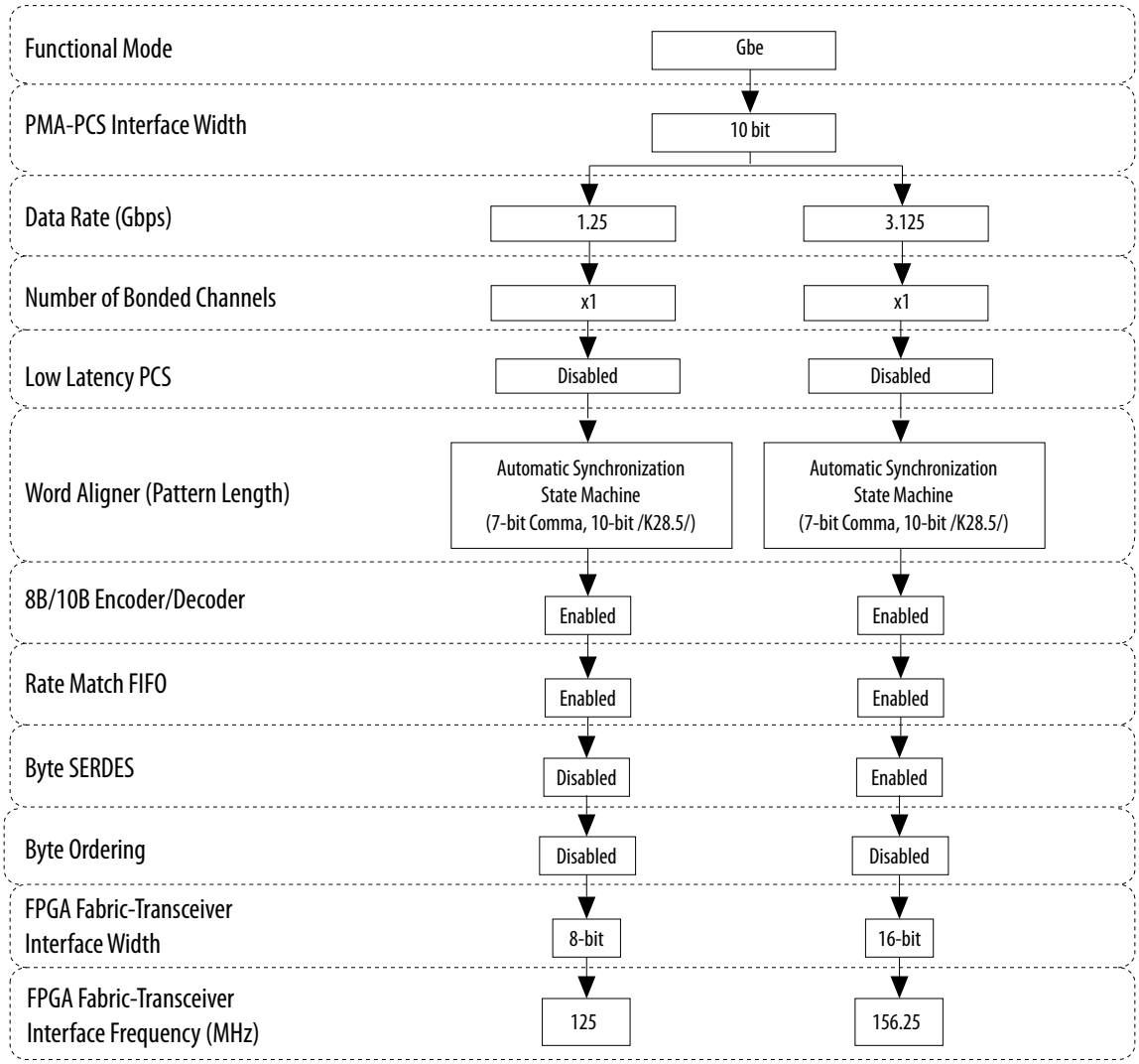
The transceivers, when configured in GbE functional mode, have built-in circuitry to support the following PCS and PMA functions, as defined in the IEEE 802.3 specification:

- 8B/10B encoding and decoding
- Synchronization
- Clock recovery from the encoded data forwarded by the receiver PMD
- Serialization and deserialization

Note: If you enabled the autonegotiation state machine in the FPGA core with the rate match FIFO, refer to the "Rate Match FIFO" section in the "Gigabit Ethernet Transceiver Datapath" section.

Note: The transceivers do not have built-in support for other PCS functions, such as the autonegotiation state machine, collision-detect, and carrier-sense functions. If you require these functions, implement them in the FPGA fabric or in external circuits.

Figure 4-13: Transceiver Blocks in a GbE Configuration



Gigabit Ethernet Transceiver Datapath

Figure 4-14: Transceiver Datapath in GbE-1.25 Gbps Configuration

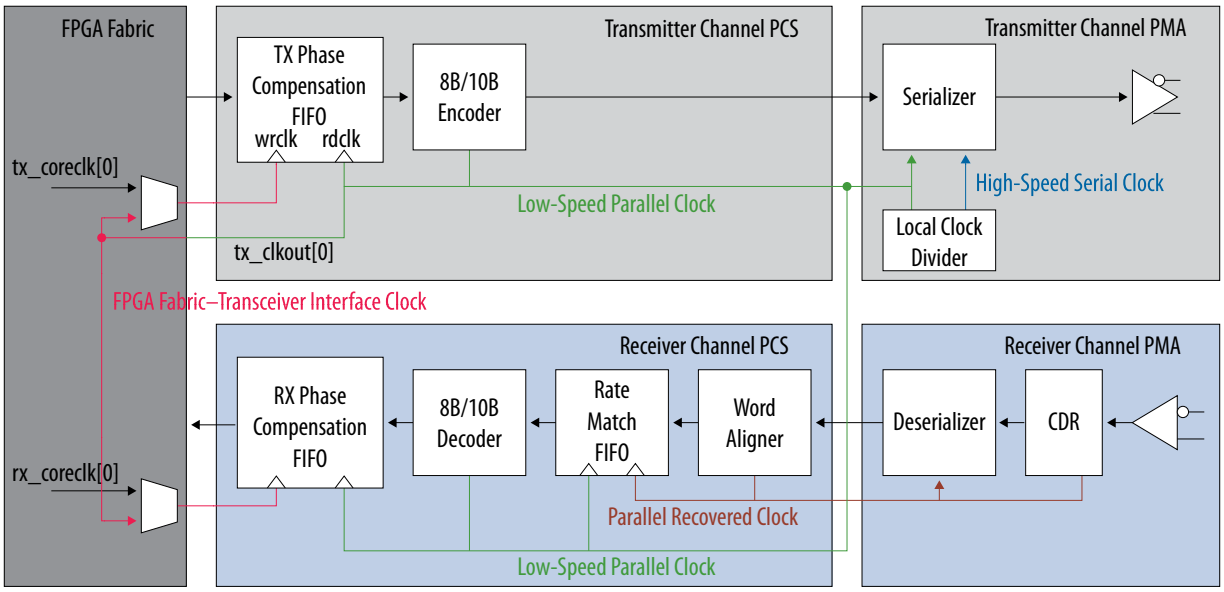


Figure 4-15: Transceiver Datapath in GbE-3.125 Gbps Configuration

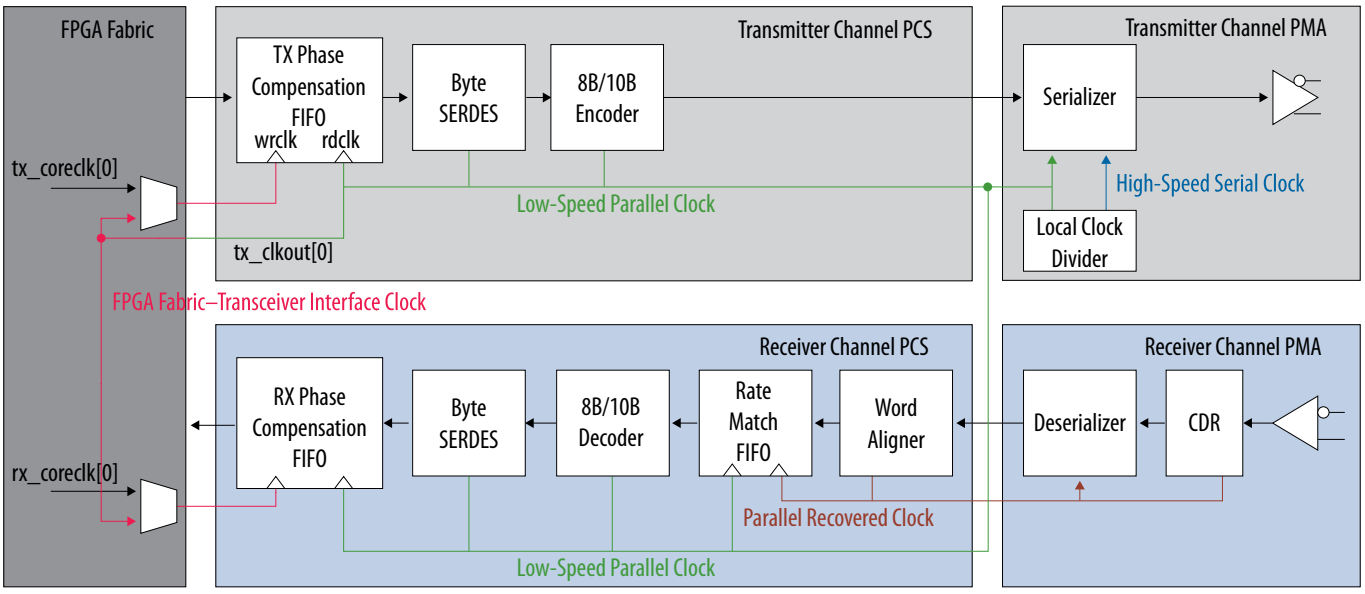


Table 4-4: Transceiver Datapath Clock Frequencies in GbE Configuration

Functional Mode	Data Rate	High-Speed Serial Clock Frequency	Parallel Recovered Clock and Low-Speed Parallel Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency
GbE-1.25 Gbps	1.25 Gbps	625 MHz	125 MHz	125 MHz
GbE-3.125 Gbps	3.125 Gbps	1562.5 MHz	312.5 MHz	156.25 MHz

8B/10B Encoder

In GbE configuration, the 8B/10B encoder clocks in 8-bit data and 1-bit control identifiers from the transmitter phase compensation FIFO and generates 10-bit encoded data. The 10-bit encoded data is fed to the serializer.

For more information about the 8B/10B encoder functionality, refer to the **Transceiver Architecture for Cyclone V Devices** chapter.

Rate Match FIFO

In GbE configuration, the rate match FIFO is capable of compensating for up to ±100 ppm (200 ppm total) difference between the upstream transmitter and the local receiver reference clock. The GbE protocol requires that the transmitter send idle ordered sets /I1/ (/K28.5/D5.6/) and /I2/ (/K28.5/D16.2/) during interpacket gaps, adhering to the rules listed in the IEEE 802.3 specification.

The rate match operation begins after the synchronization state machine in the word aligner indicates that the synchronization is acquired-by driving the `rx_syncstatus` signal high. The rate matcher always deletes or inserts both symbols (/K28.5/ and /D16.2/) of the /I2/ ordered sets, even if only one symbol needs to be deleted to prevent the rate match FIFO from overflowing or underrunning. The rate matcher can insert or delete as many /I2/ ordered sets as necessary to perform the rate match operation.

Two flags are forwarded to the FPGA fabric:

- `rx_rmfifodatadeleted`—Asserted for two clock cycles for each deleted /I2/ ordered set to indicate the rate match FIFO deletion event
- `rx_rmfifodatainserted`—Asserted for two clock cycles for each inserted /I2/ ordered set to indicate the rate match FIFO insertion event

Note: If you have the autonegotiation state machine in the FPGA, note that the rate match FIFO is capable of inserting or deleting the first two bytes (/K28.5//D2.2/) of /C2/ ordered sets during autonegotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the autonegotiation link to fail. For more information, refer to the **Altera Knowledge Base Support Solution**.

For more information about the rate match FIFO, refer to the **Transceiver Architecture for Cyclone V Devices** chapter.

GbE Protocol-Ordered Sets and Special Code Groups

Table 4-5: GIGE Ordered Sets

The following ordered sets and special code groups are specified in the IEEE 802.3-2008 specification.

Code	Ordered Set	Number of Code Groups	Encoding
/C/	**Configuration**	—	Alternating /C1/ and /C2/
/C1/	Configuration 1	4	/K28.5/D21.5/ `Config_Reg` (12)
/C2/	Configuration 2	4	/K28.5/D2.2/ `Config_Reg` (12)
/I/	**IDLE**	—	Correcting /I1/, Preserving /I2/
/I1/	IDLE 1	2	/K28.5/D5.6/
/I2/	IDLE 2	2	/K28.5/D16.2/
-	**Encapsulation**	—	—
/R/	`Carrier_Extend`	1	/K23.7/
/S/	`Start_of_Packet`	1	/K27.7/
/T/	`End_of_Packet`	1	/K29.7/
/V/	`Error_Propagation`	1	/K30.7/

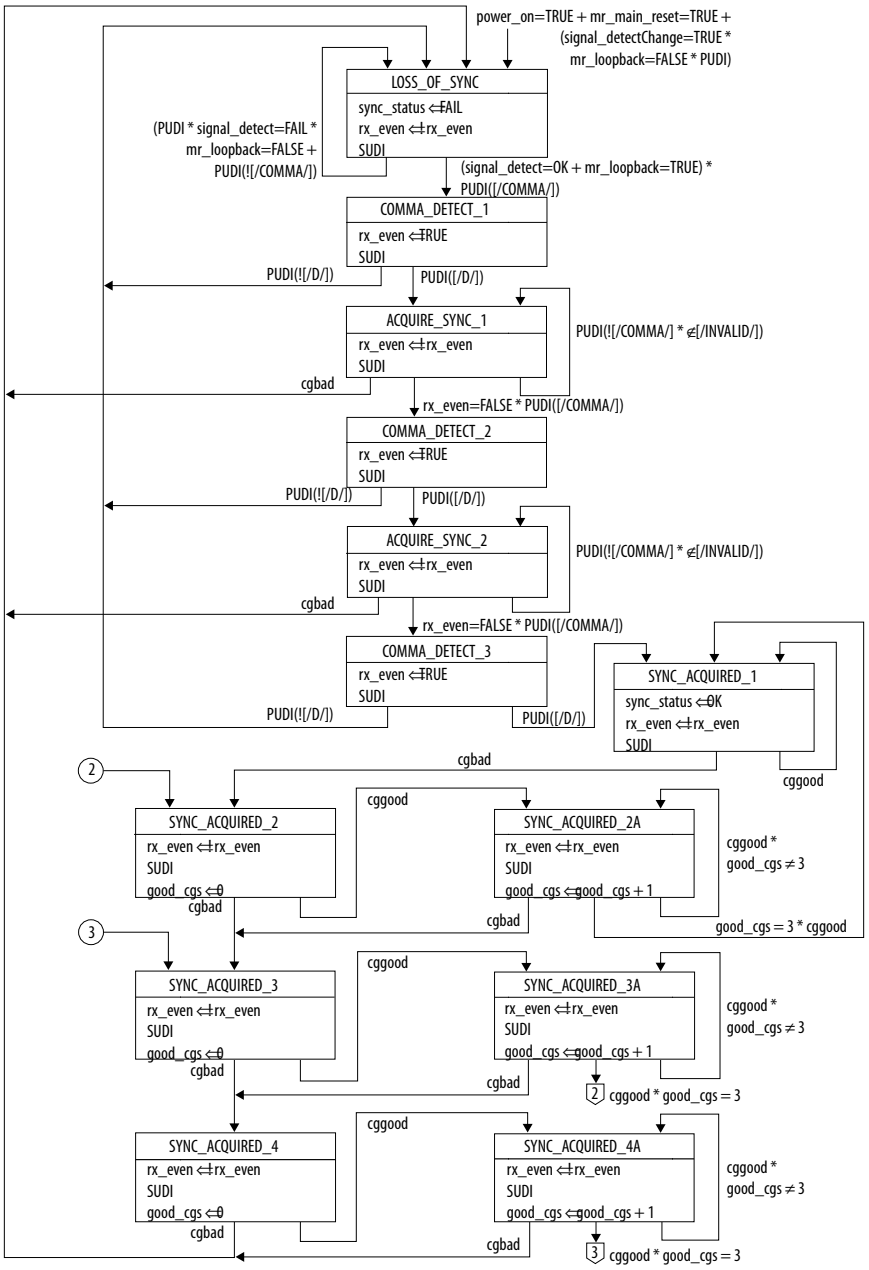
Table 4-6: Synchronization State Machine Parameters in GbE Mode

Synchronization State Machine Parameters	Setting
Number of valid {/K28.5/, /Dx,y/} ordered sets received to achieve synchronization	3
Number of errors received to lose synchronization	4
Number of continuous good code groups received to reduce the error count by 1	4

(12) Two data code groups represent the `Config_Reg` value.

Figure 4-16: Synchronization State Machine in GbE Mode

This figure is from "Figure 36–9" in the IEEE 802.3-2008 specification. For more details about the 1000BASE-X implementation, refer to Clause 36 of the IEEE 802.3-2008 specification.



Related Information

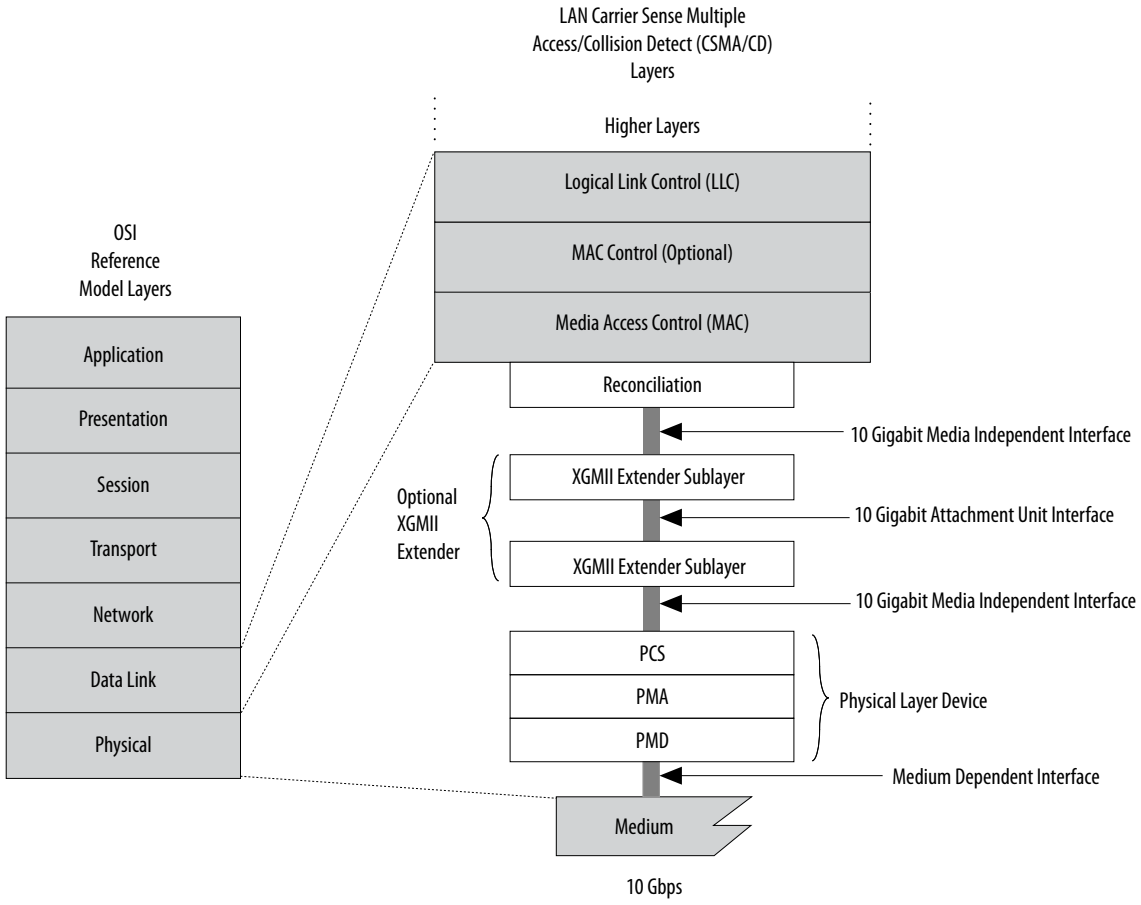
Refer to the "Custom PHY IP Core" and "Native PHY IP Core" chapters in the Altera Transceiver PHY IP Core User Guide

XAUI

In a XAUI configuration, the transceiver channel data path is configured using soft PCS. It provides the transceiver channel datapath description, clocking, and channel placement guidelines. To implement a XAUI link, instantiate the XAUI PHY IP core in the IP Catalog, which is under Ethernet in the Interfaces menu. The XAUI PHY IP core implements the XAUI PCS in soft logic.

XAUI is a specific physical layer implementation of the 10 Gigabit Ethernet link defined in the IEEE 802.3ae-2002 specification. The XAUI PHY uses the XGMII interface to connect to the IEEE802.3 MAC and Reconciliation Sublayer (RS). The IEEE 802.3ae-2002 specification requires the XAUI PHY link to support a 10 Gbps data rate at the XGMII interface and four lanes each at 3.125 Gbps at the PMD interface.

Figure 4-17: XAUI and XGMII Layers



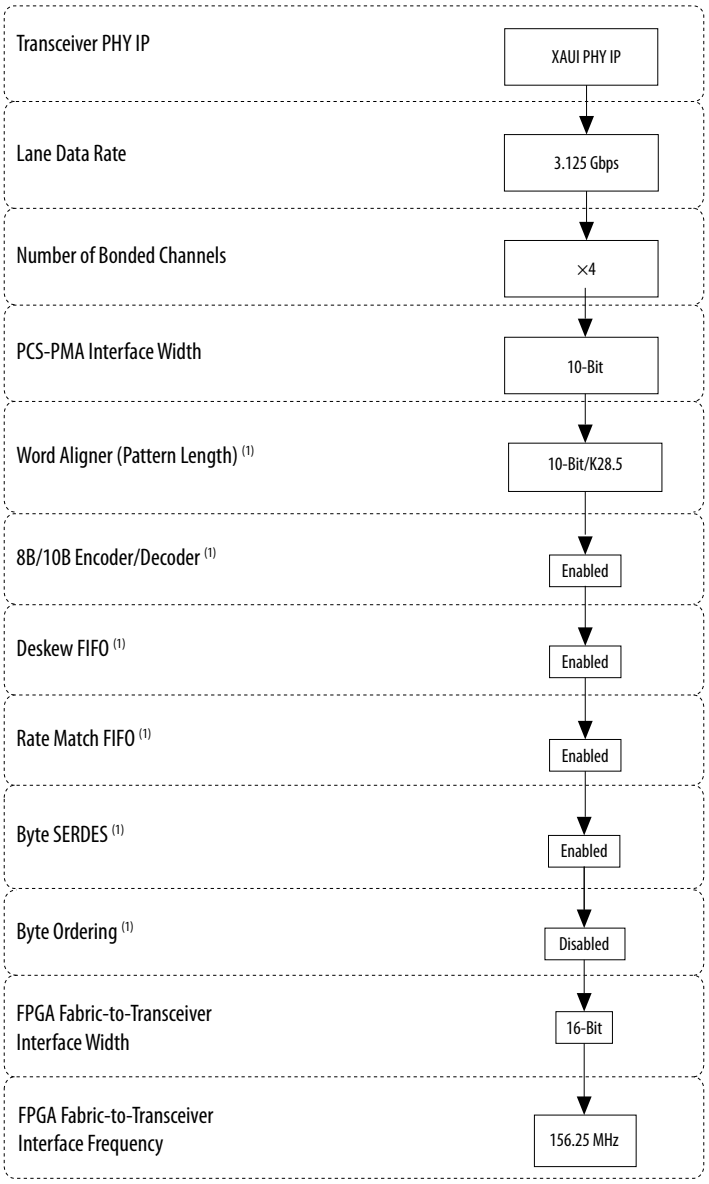
Related Information

Refer to the "XAUI PHY IP Core" chapter in the Altera Transceiver PHY IP Core User Guide.

Transceiver Datapath in a XAUI Configuration

The XAUI PCS is implemented in soft logic inside the FPGA core when using the XAUI PHY IP core. You must ensure that your channel placement is compatible with the soft PCS implementation.

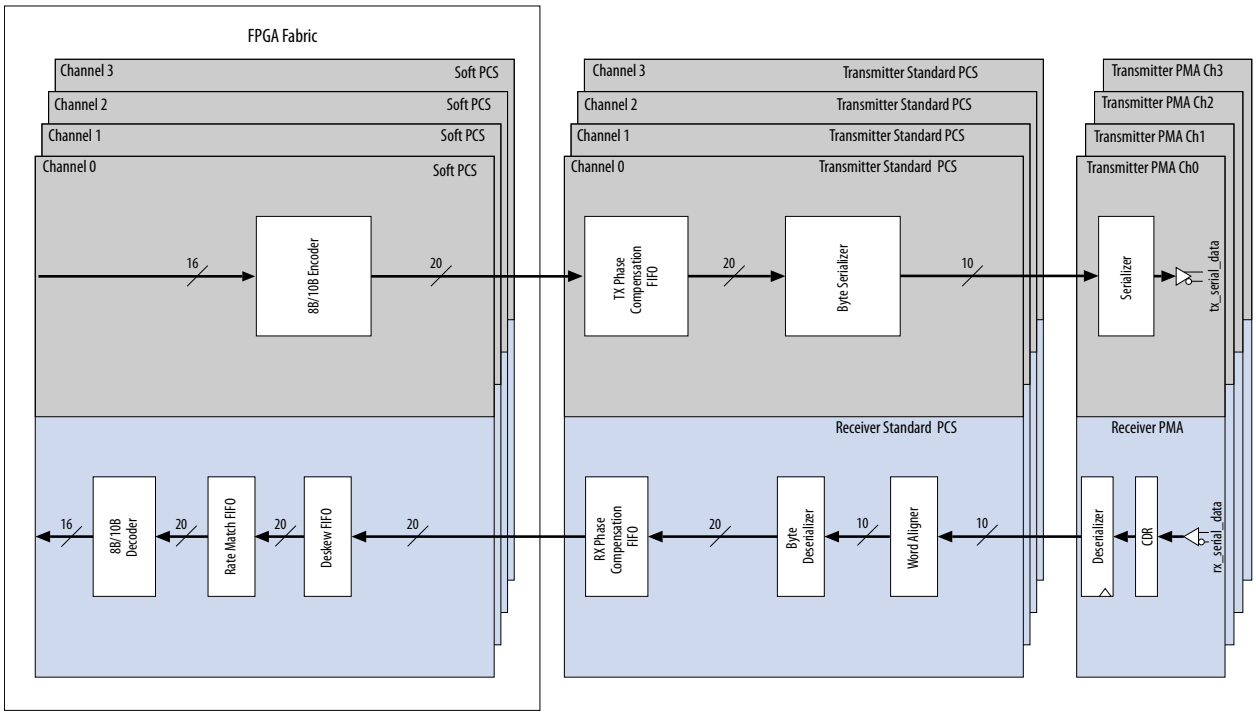
Figure 4-18: XAUI Configuration Datapath



Note:
1. Implemented in soft logic.

Figure 4-19: Transceiver Channel Datapath for XAUI Configuration

Standard PCS in a low latency configuration is used in this configuration. Additionally, a portion of the PCS is implemented in soft logic.



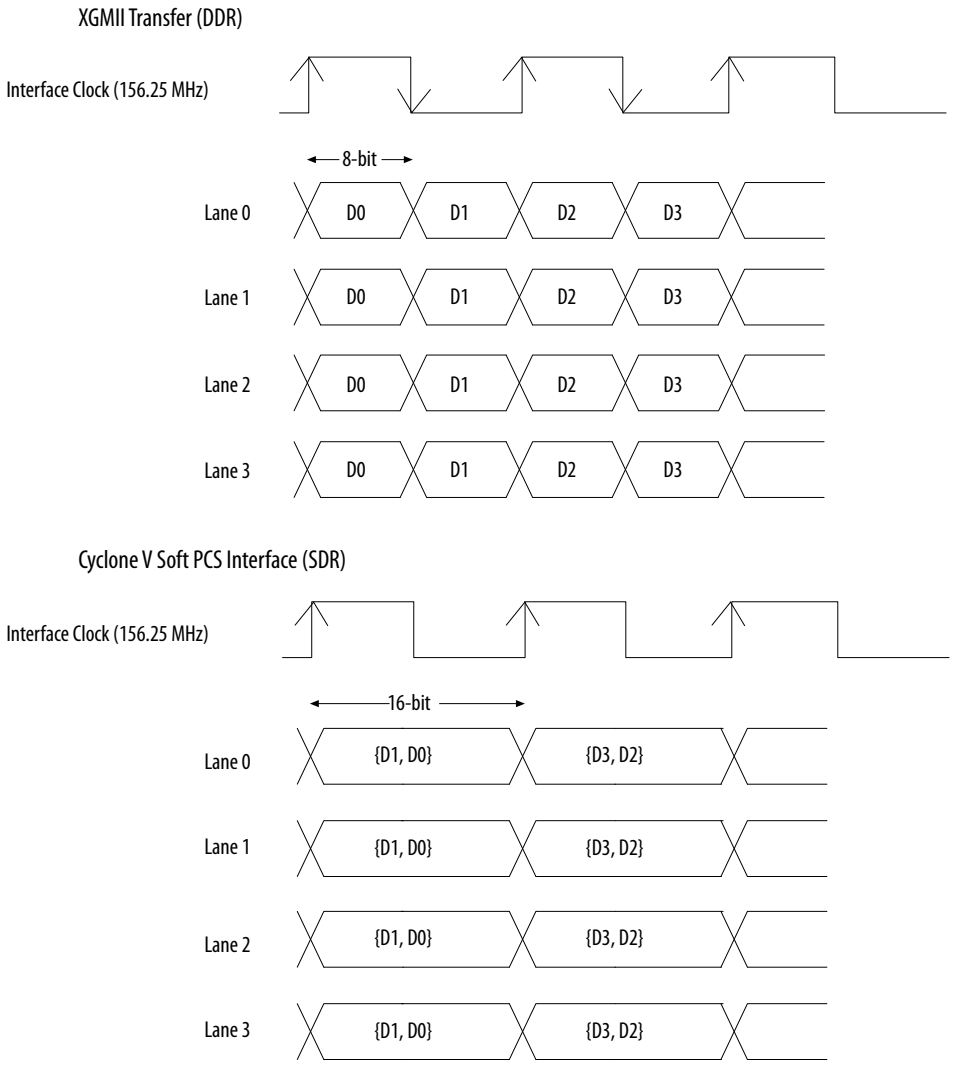
XAUI Supported Features

64-Bit SDR Interface to the MAC/RS

Clause 46 of the IEEE 802.3-2008 specification defines the XGMII interface between the XAUI PCS and the Ethernet MAC/RS. The specification requires each of the four XAUI lanes to transfer 8-bit data and 1-bit wide control code at both the positive and negative edge (DDR) of the 156.25 MHz interface clock.

Cyclone V transceivers and soft PCS solution in a XAUI configuration do not support the XGMII interface to the MAC/RS as defined in IEEE 802.3-2008 specification. Instead, they allow the transferring of 16-bit data and 2-bit control code on each of the four XAUI lanes, only at the positive edge (SDR) of the 156.25 MHz interface clock

Figure 4-20: Implementation of the XGMII Specification in Cyclone V Devices Configuration



8B/10B Encoding/Decoding

Each of the four lanes in a XAUI configuration support an independent 8B/10B encoder/decoder as specified in Clause 48 of the IEEE802.3-2008 specification. 8B/10B encoding limits the maximum number of consecutive 1s and 0s in the serial data stream to five, thereby ensuring DC balance as well as enough transitions for the receiver CDR to maintain a lock to the incoming data.

The XAUI PHY IP core provides status signals to indicate running disparity as well as the 8B/10B code group error.

Transmitter and Receiver State Machines

In a XAUI configuration, the Cyclone V soft PCS implements the transmitter and receiver state diagrams shown in Figure 48-6 and Figure 48-9 of the IEEE802.3-2008 specification.

In addition to encoding the XGMII data to PCS code groups, in conformance with the 10GBASE-X PCS, the transmitter state diagram performs functions such as converting Idle ||I|| ordered sets into Sync ||K||, Align ||A||, and Skip ||R|| ordered sets.

In addition to decoding the PCS code groups to XGMII data, in conformance with the 10GBASE-X PCS, the receiver state diagram performs functions such as converting Sync ||K||, Align ||A||, and Skip ||R|| ordered sets to Idle ||I|| ordered sets.

Synchronization

The word aligner block in the receiver PCS of each of the four XAUI lanes implements the receiver synchronization state diagram shown in Figure 48-7 of the IEEE802.3-2008 specification.

The XAUI PHY IP core provides a status signal per lane to indicate if the word aligner is synchronized to a valid word boundary.

Deskew

The lane aligner block in the receiver PCS implements the receiver deskew state diagram shown in Figure 48-8 of the IEEE 802.3-2008 specification.

The lane aligner starts the deskew process only after the word aligner block in each of the four XAUI lanes indicates successful synchronization to a valid word boundary.

The XAUI PHY IP core provides a status signal to indicate successful lane deskew in the receiver PCS.

Clock Compensation

The rate match FIFO in the receiver PCS datapath compensates up to ±100 ppm difference between the remote transmitter and the local receiver. It does so by inserting and deleting Skip ||R|| columns, depending on the ppm difference.

The clock compensation operation begins after:

- The word aligner in all four XAUI lanes indicates successful synchronization to a valid word boundary.
- The lane aligner indicates a successful lane deskew.

The rate match FIFO provides status signals to indicate the insertion and deletion of the Skip ||R|| column for clock rate compensation.

Transceiver Clocking and Channel Placement Guidelines in XAUI Configuration

Transceiver Clocking

Figure 4-21: Transceiver Clocking for XAUI Configuration

One of the two channel PLLs configured as a CMU PLL in a transceiver bank generates the transmitter serial and parallel clocks for the four XAUI channels. The x6 clock line carries the transmitter clocks to the PMA and PCS of each of the four channels.

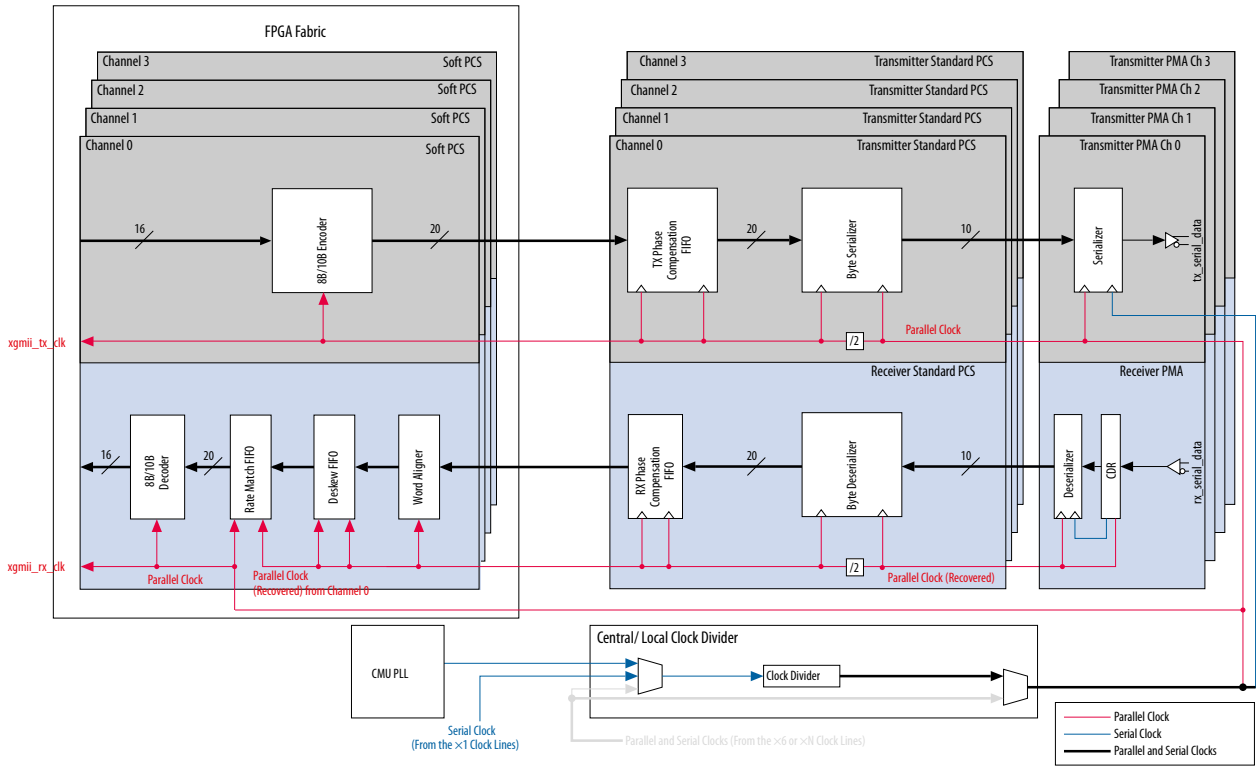


Table 4-7: Input Reference Clock Frequency and Interface Speed Specifications for XAUI Configurations

Input Reference Clock Frequency (MHz)	FPGA Fabric-Transceiver Interface Width	FPGA Fabric-Transceiver Interface Frequency (MHz)
156.25	16-bit data, 2-bit control	156.25

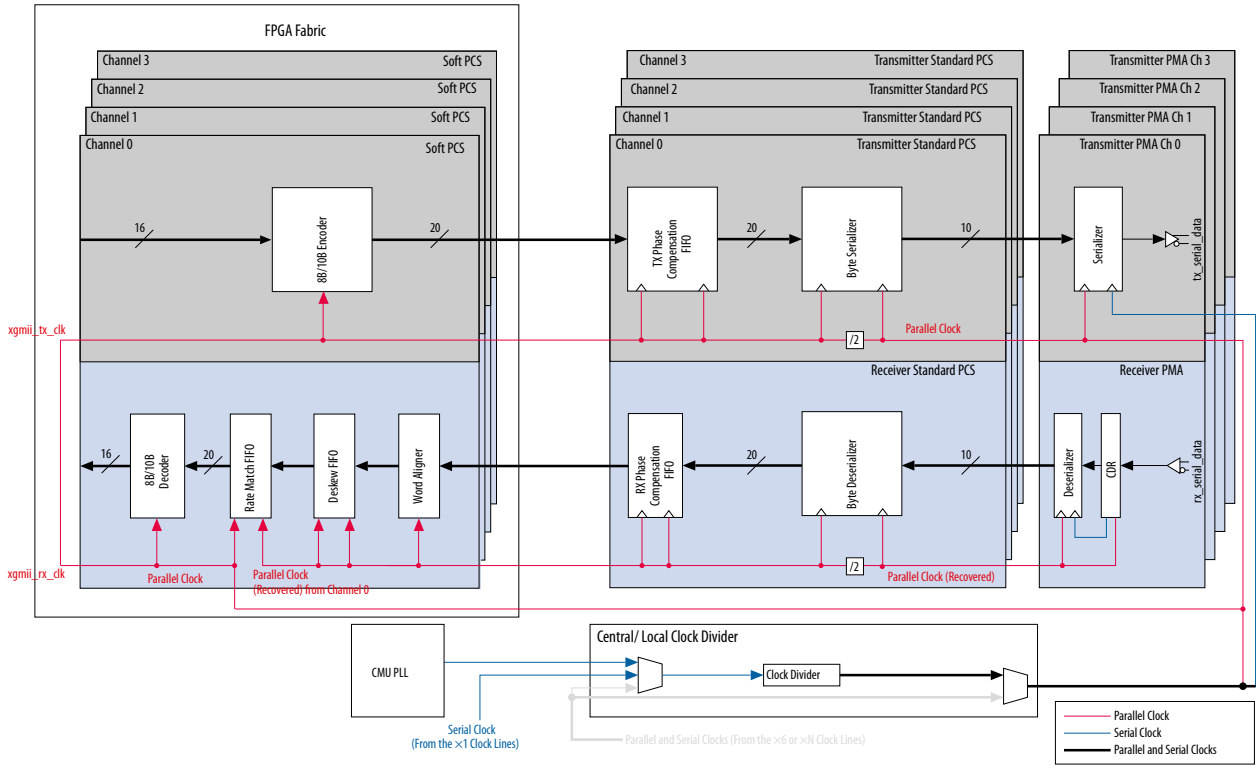
Transceiver Clocking Guidelines for Soft PCS Implementation

In the soft PCS implementation in the XAUI configuration, you must route `xgmii_rx_clk` to `xgmii_tx_clk` as shown in the following figure.

This method uses `xgmii_rx_clk` to compensate for the phase difference on the TX side.

Without this method, the `tx_digitalreset` signal may experience intermittent failure.

Figure 4-22: Transceiver Clocking for XAUI Soft PCS Implementation

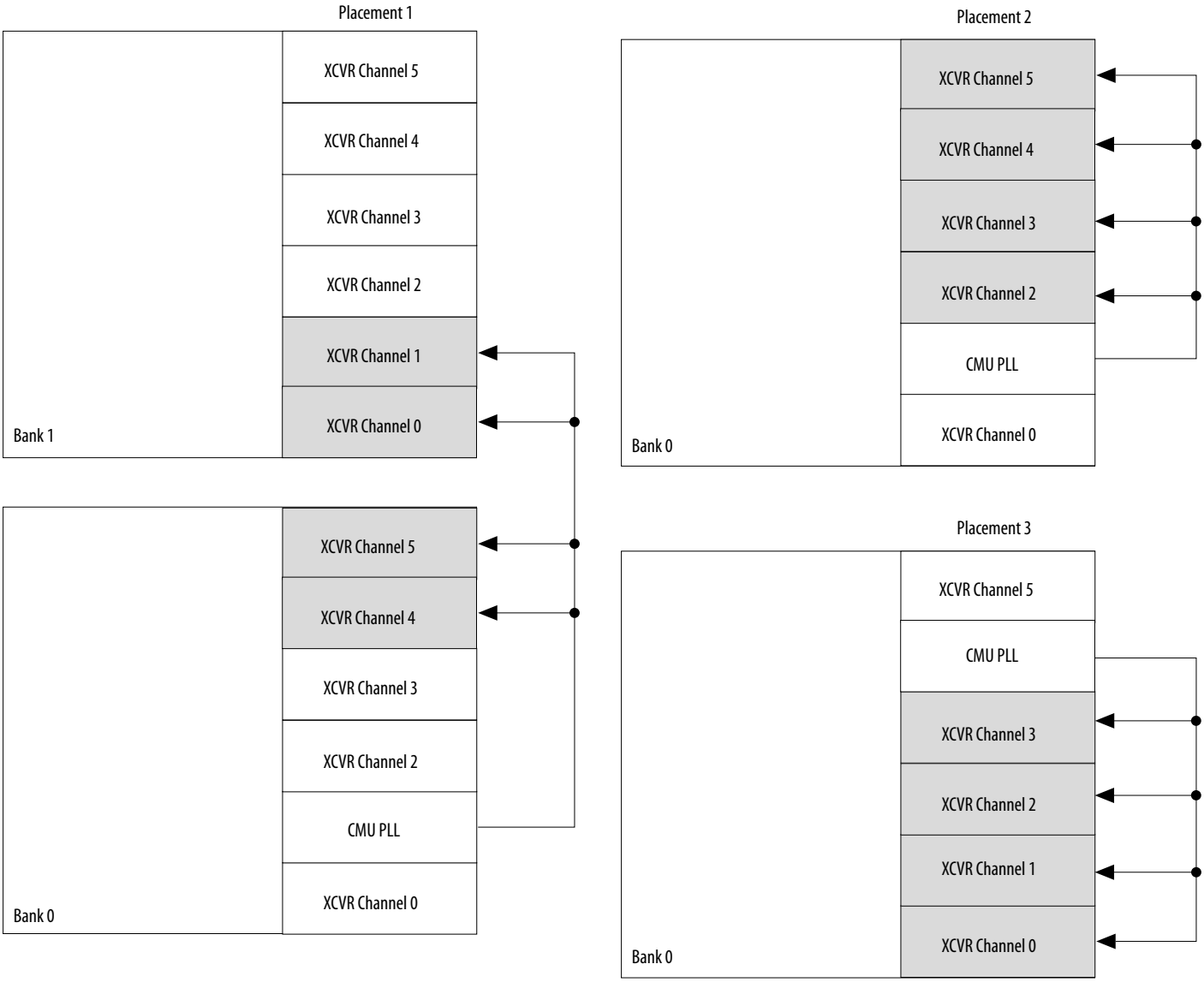


Transceiver Channel Placement Guidelines

In the soft PCS implementation of the XAUI configuration, you can construct the four XAUI lanes at any channels within the two transceiver banks. However, Altera recommends you place the four channels contiguously to close timing more easily The channels may all be placed in one bank or they may span two banks. The following figure shows several possible channel placements when using the CMU PLL to drive the XAUI link.

Figure 4-23: Transceiver Channel Placement Guidelines in a XAUI Configuration

The Quartus II software implements the XAUI PCS in soft logic. Each XAUI link requires a dedicated CMU PLL. A single CMU PLL cannot be shared among different XAUI links.



Related Information

To implement the QSF assignment workaround using the Assignment Editor, refer to the "XAUI PHY IP Core" chapter in the Altera Transceiver PHY IP Core User Guide.

Serial Digital Interface

The Society of Motion Picture and Television Engineers (SMPTE) defines various Serial Digital Interface (SDI) standards for transmission of uncompressed video.

The following SMPTE standards are popular in video broadcasting applications:

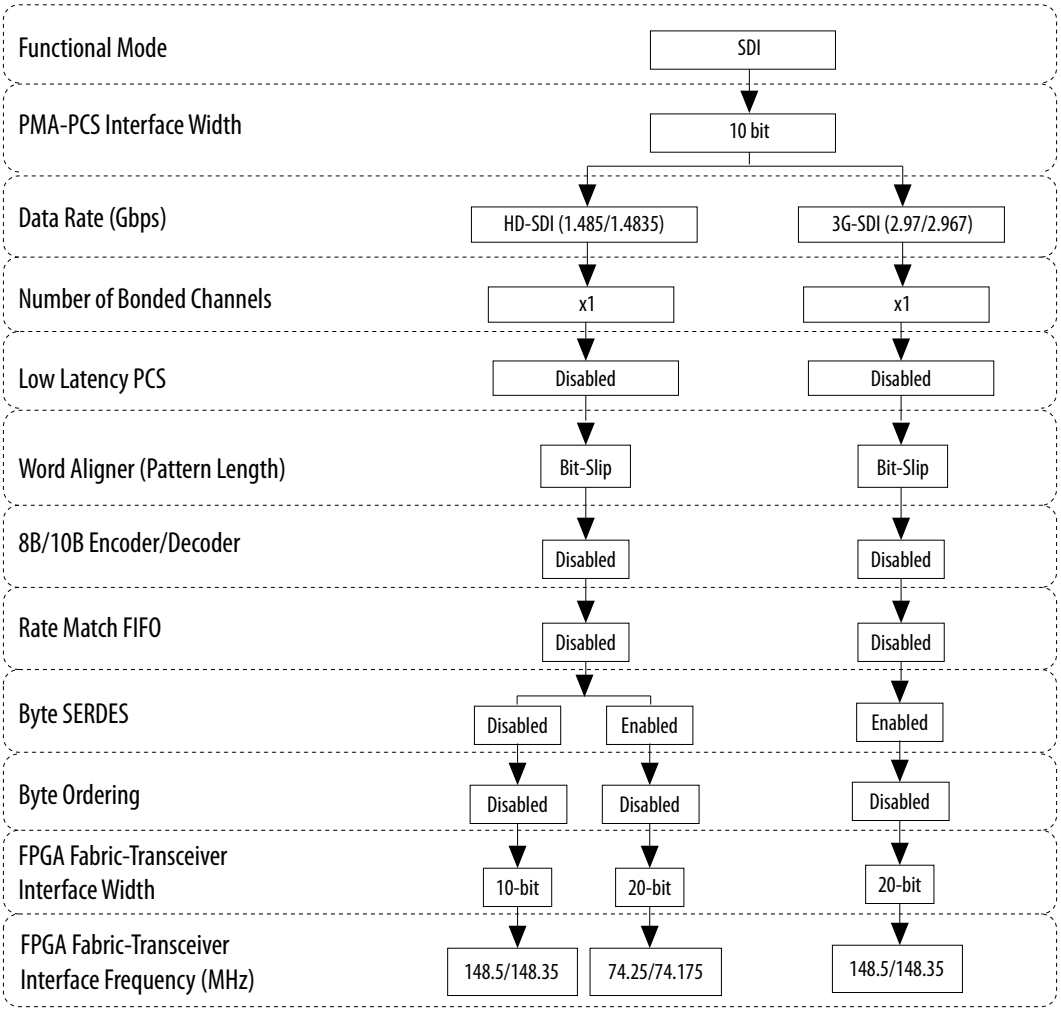
- SMPTE 259M standard - more popularly known as the standard-definition (SD) SDI; defined to carry video data at 270 Mbps
- SMPTE 292M standard - more popularly known as the high-definition (HD) SDI; defined to carry video data at either 1485 Mbps or 1483.5 Mbps
- SMPTE 424M standard - more popularly known as the third-generation (3G) SDI; defined to carry video data at either 2970 Mbps or 2967 Mbps

Configurations Supported in SDI Mode

Table 4-8: Configurations Supported in SDI Mode

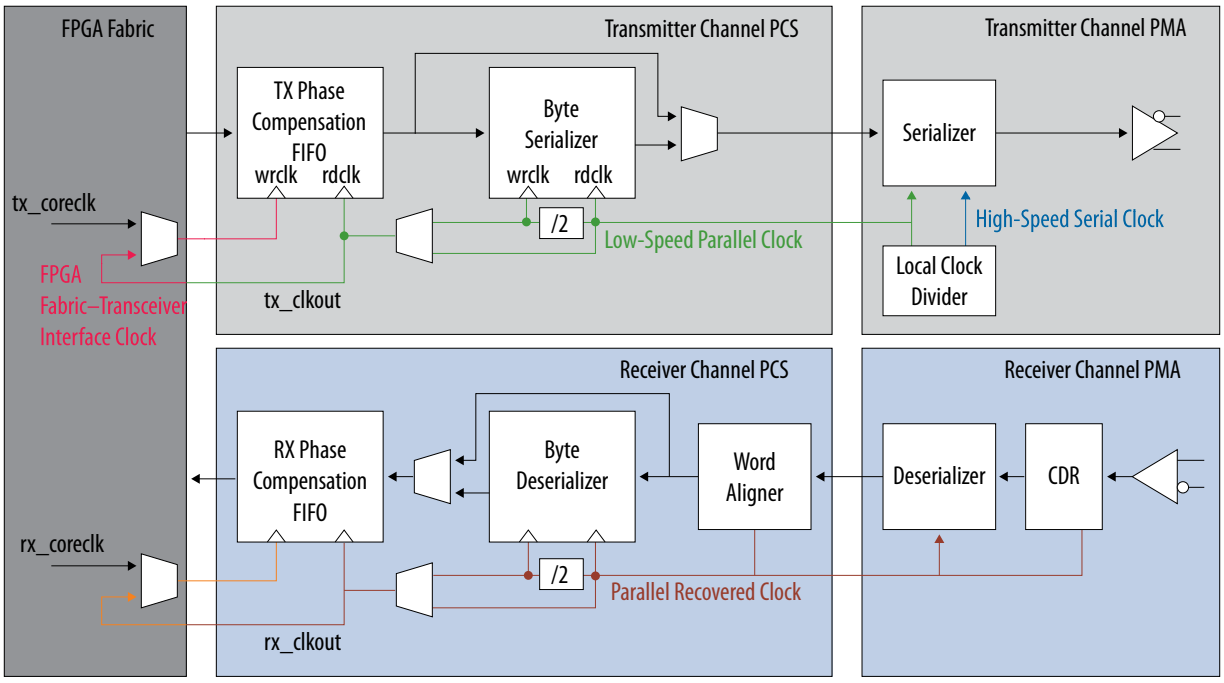
Configuration	Data Rate (Mbps)	REFCLK Frequencies (MHz)	FPGA Fabric-Transceiver Interface Width
HD	1,485	74.25, 148.5	10 bit and 20 bit
	1,483.5	74.175, 148.35	10 bit and 20 bit
3G	2,970	148.5, 297	Only 20-bit interfaces allowed in 3G
	2,967	148.35, 296.7	Only 20-bit interfaces allowed in 3G

Figure 4-24: SDI Mode



Serial Digital Interface Transceiver Datapath

Figure 4-25: SDI Mode Transceiver Datapath



Transmitter Datapath

The transmitter datapath in the HD-SDI configuration with a 10-bit wide FPGA fabric-transceiver interface consists of the transmitter phase compensation FIFO and the 10:1 serializer. In HD-SDI and 3G-SDI configurations with 20-bit wide FPGA fabric-transceiver interface, the transmitter datapath also includes the byte serializer.

Note: In SDI mode, the transmitter is purely a parallel-to-serial converter. You must implement the SDI transmitter functions, such as the scrambling and cyclic redundancy check (CRC) code generation, in the FPGA logic array.

Receiver Datapath

In the 10-bit channel width SDI configuration, the receiver datapath consists of the clock recovery unit (CRU), 1:10 deserializer, word aligner in bit-slip mode, and receiver phase compensation FIFO. In the 20-bit channel width SDI configuration, the receiver datapath also includes the byte deserializer.

Note: You must implement the SDI receiver functions, such as descrambling, framing, and CRC checker, in the FPGA logic array.

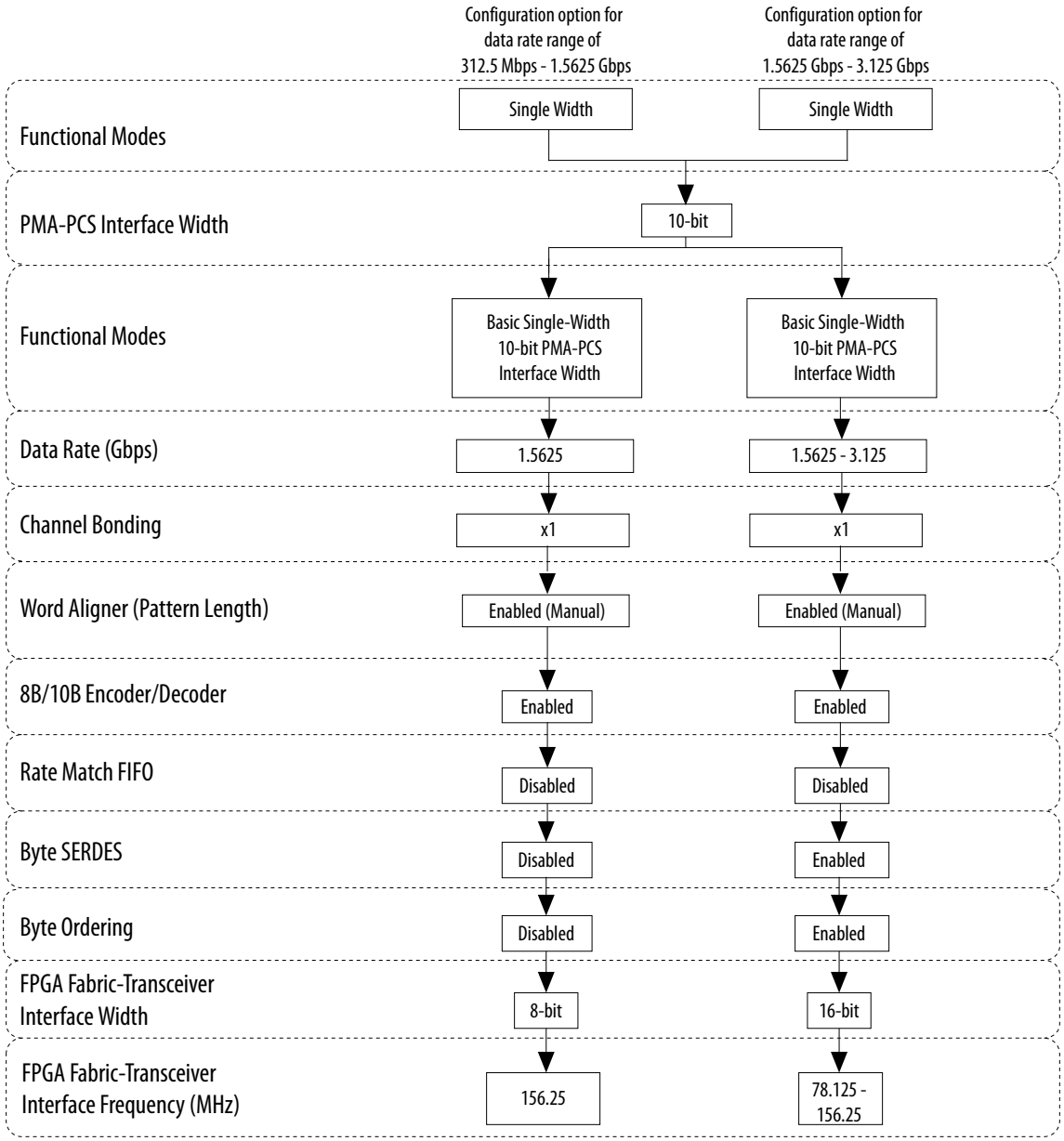
Receiver Word Alignment and Framing

In SDI systems, the word aligner in the receiver datapath is not useful because the word alignment and framing happen after descrambling. Altera recommends that you drive the `rx_bitslip` signal of the PHY IP core low to avoid having the word aligner insert bits in the received data stream.

Serial Data Converter (SDC) JESD204

The SDC (JESD204) protocol conforms to JESD204, a JEDEC standard that enables a high-speed serial connection between analog-to-digital converters and logic devices using only a two-wire high-speed serial interface. SDC (JESD204) data rate ranges of 312.5 Mbps to 3.125 Gbps are supported. The minimum supported data rate is 611 Mbps, so a 5x oversampling factor is used for the SDC (JESD204) data rate of 312.5 Mbps, resulting in a data rate of 1.5625 Gbps.

Figure 4-26: Configurations for the SDC (JESD204) Protocol



SATA and SAS Protocols

Serial ATA (SATA) and Serial Attached SCSI (SAS) are data storage protocol standards that have the primary function of transferring data (directly or otherwise) between the host system and mass storage devices, such as hard disk drives, optical drives, and solid-state disks.

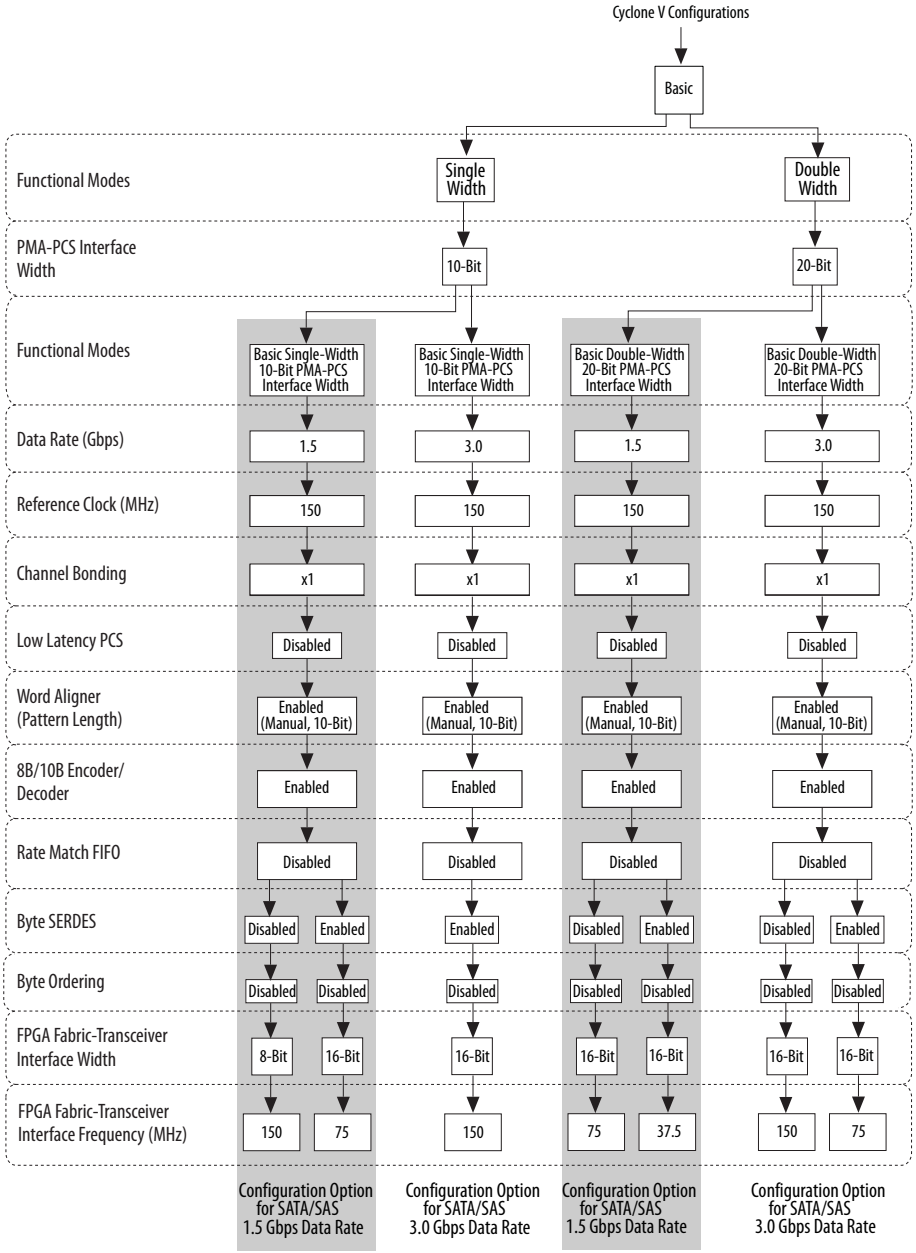
These serial storage protocols offer several advantages over older parallel storage protocol (ATA and SCSI) interfaces:

- Faster data transfer
- Hot swapping (when supported by the operating system)
- Thinner cables for more efficient air cooling
- Increased operation reliability

Table 4-9: Serial Data Rates for SATA and SAS Protocols

Protocol	SATA (Gbps)	SAS (Gbps)
Gen1	1.5	3.0
Gen2	3.0	—

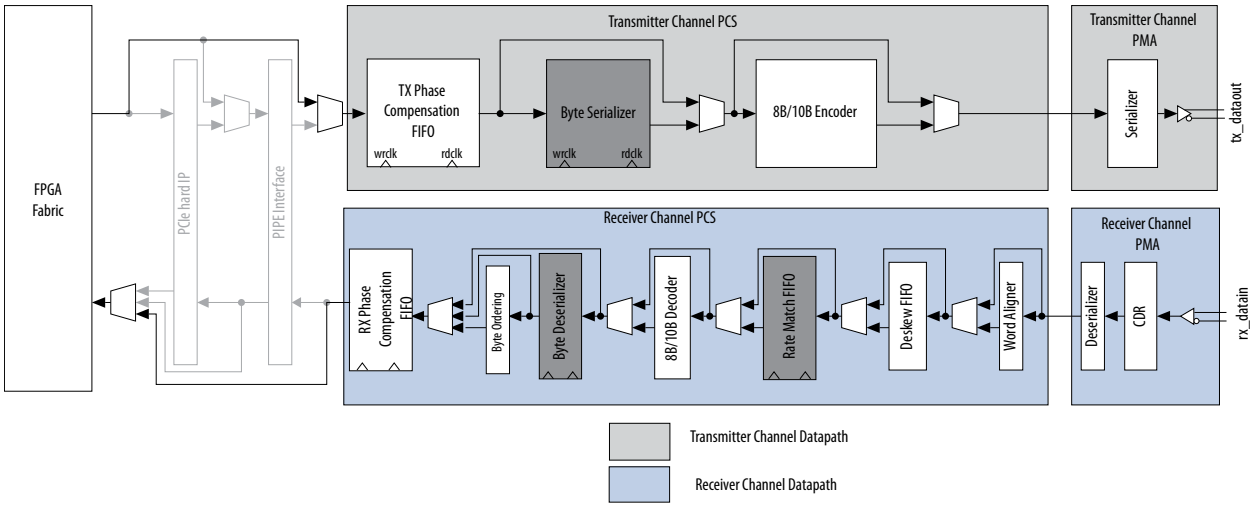
Figure 4-27: Configurations for the SATA and SAS Protocols



Deterministic Latency Protocols—CPRI and OBSAI

A deterministic latency option is available for use in high-speed serial interfaces such as the Common Public Radio Interface (CPRI) and OBSAI Reference Point 3 (OBSAI RP3). Both CPRI and OBSAI RP3 protocols place stringent requirements on the amount of latency variation that is permissible through a link that implements these protocols.

Figure 4-28: Transceiver Datapath in Deterministic Latency Mode



Latency Uncertainty Removal with the Phase Compensation FIFO in Register Mode

To remove the latency uncertainty through the receiver's phase compensation FIFO, the receiver and transmitter phase compensation FIFOs are always set to register mode. In register mode, the phase compensation FIFO acts as a register and thereby removes the uncertainty in latency. The latency through the transmitter and receiver phase compensation FIFO in register mode is one clock cycle.

The following options are available:

- Single-width mode with 8-bit channel width and 8B/10B encoder enabled or 10-bit channel width with 8B/10B disabled
- Double-width mode with 16-bit channel width and 8B/10B encoder enabled or 20-bit channel width with 8B/10B disabled

Channel PLL Feedback for Deterministic Relationship

To implement the deterministic latency functional mode, the phase relationship between the low-speed parallel clock and channel PLL input reference clock must be deterministic. A feedback path is enabled to ensure a deterministic relationship between the low-speed parallel clock and channel PLL input reference clock.

To achieve deterministic latency through the transceiver, the reference clock to the channel PLL must be the same as the low-speed parallel clock. For example, if you need to implement a data rate of 1.2288 Gbps for the CPRI protocol, which places stringent requirements on the amount of latency variation, you must choose a reference clock of 122.88 MHz to allow the usage of a feedback path from the channel PLL. This feedback path reduces the variations in latency.

When you select this option, provide an input reference clock to the channel PLL that has the same frequency as the low-speed parallel clock.

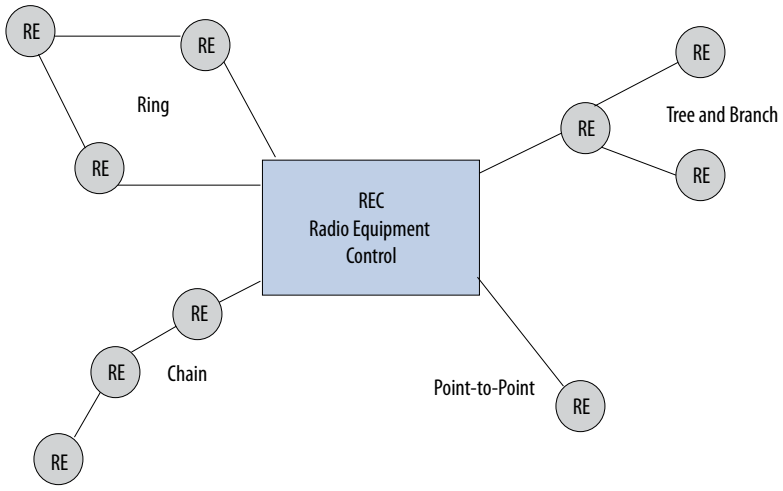
CPRI and OBSAI

Use the deterministic latency functional mode to implement protocols such as CPRI and OBSAI.

The CPRI interface defines a digital point-to-point interface between the Radio Equipment Control (REC) and the Radio Equipment (RE), allowing flexibility in either co-locating the REC and the RE, or a remote location of the RE.

Figure 4-29: CPRI Topologies

In most cases, CPRI links are between REC and RE modules or between two RE modules in a chain configuration.



If the destination for the high-speed serial data that leaves the REC is the first RE, it is a single-hop connection. If the serial data from the REC must traverse through multiple REs before reaching the destination RE, it is a multi-hop connection.

Remotely locating the RF transceiver from the main base station introduces a complexity with overall system delay. The CPRI specification requires that the accuracy of measurement of roundtrip delay on single-hop and multi-hop connections be within ±16.276 ns to properly estimate the cable delay.

For a single-hop system, this allows a variation in roundtrip delay of up to ±16.276 ns. However, for multi-hop systems, the allowed delay variation is divided among the number of hops in the connection—typically, equal to ±16.276 ns/(the number of hops) but not always equally divided among the hops.

Deterministic latency on a CPRI link also enables highly accurate triangulation of the location of the caller.

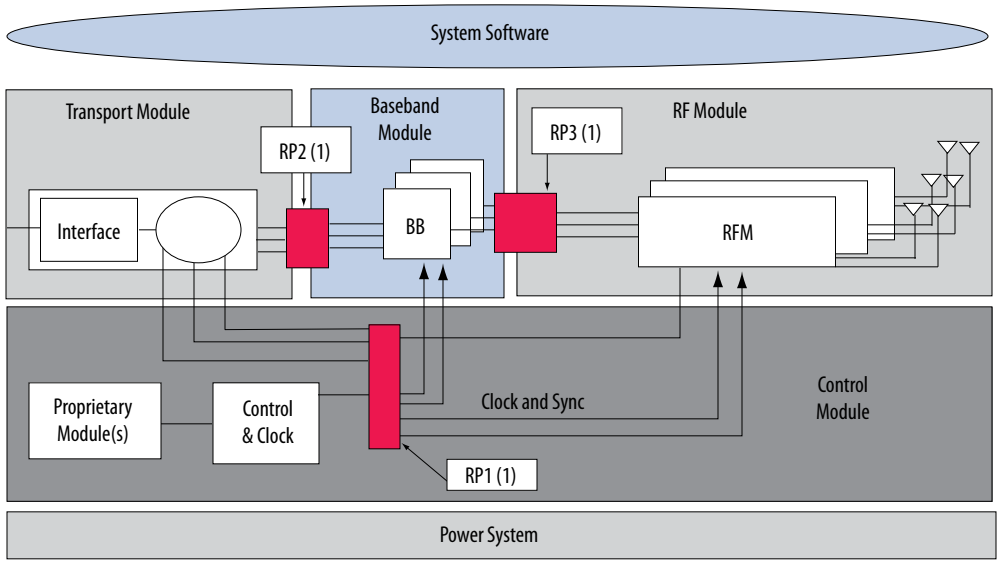
OBSAI was established by several OEMs to develop a set of specifications that can be used for configuring and connecting common modules into base transceiver stations (BTS).

The BTS has four main modules:

- Radio frequency (RF)
- Baseband
- Control
- Transport

In a typical BTS, the radio frequency module (RFM) receives signals using portable devices and converts the signals to digital data. The baseband module processes the encoded signal and brings it back to the baseband before transmitting it to the terrestrial network using the transport module. A control module maintains the coordination between these three functions.

Figure 4-30: Example of the OBSAI BTS Architecture



(1) RP = Reference Point

Using the deterministic latency option, you can implement the CPRI data rates in the following modes:

- Single-width mode—with 8/10-bit channel width
- Double-width mode—with 16/20-bit channel width

Table 4-10: Sample Channel Width Options for Supported Serial Data Rates

Serial Data Rate (Mbps)	Channel Width (FPGA-PCS Fabric)			
	Single-Width		Double-Width	
	8-Bit	16-Bit	16-Bit	32-Bit
614.4	Yes	Yes	No	No
1228.8	Yes	Yes	Yes	Yes
2457.6	No	Yes	Yes	Yes
3072	No	Yes	Yes	Yes
4915.2	No	No	No	Yes
6144 (13)	No	No	No	Yes

(13) 6144 Mbps is supported only for the CPRI protocol in the C5 and I5 speed grades.

6.144-Gbps Support Capability in Cyclone V GT Devices

Cyclone V GT devices support a 6.144 Gbps data rate for the CPRI protocol only. For CPRI 6.144-Gbps transmit jitter compliance, Altera recommends you use only up to three full-duplex channels for every two transceiver banks. The transceivers are grouped in banks of three channels. For transceiver bank information, refer to the *Transceiver Architecture in Cyclone V Devices* chapter.

The maximum number of CPRI channels allowed for 9-channel and 12-channel devices is as follows. The same limitation applies to devices with fewer transceiver channels.

- For a 9-channel device, you can implement a maximum of 4 full duplex 6.144-Gbps CPRI-compliant channels.
- For a 12-channel device, you can implement a maximum of 6 full duplex 6.144-Gbps CPRI-compliant channels.

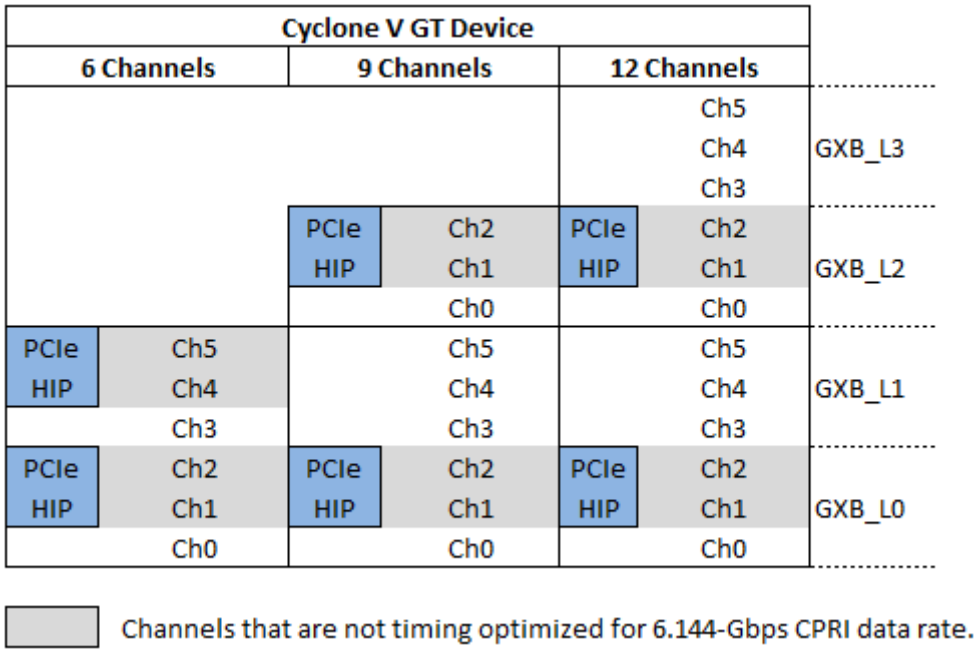
You must increase the voltage on VCCE_GXB and VCCL_GXB to 1.2 V to support the maximum number of channels.

The reference clock frequency for the 6.144 Gbps CPRI channel must be ≥ 307.2 MHz.

The maximum number of transceiver channels in a Cyclone V GT device that can achieve 6.144-Gbps CPRI compliance is based on:

- Transceiver performance in meeting the TX jitter specification for 6.144-Gbps CPRI.
- CPRI channels with an auto-rate negotiation capability from 1228.8 Mbps to 6.144 Gbps.
- 6.144-Gbps CPRI channel restriction based on the following figure.

Figure 4-31: 6.144-Gbps CPRI Channel Placement Restriction



The channels next to a PCIe Hard IP block are not timing optimized for the 6.144-Gbps CPRI data rate. Affected channels are shaded in gray in the above figure. Avoid placing the 6.144-Gbps CPRI channels in the affected channels. The affected channels can still be used as a CMU for the CPRI channels.

Related Information

Transceiver Architecture in Cyclone V Devices

CPRI Enhancements

The deterministic latency state machine in the word aligner reduces the known delay variation from the word alignment process and automatically synchronizes and aligns the word boundary by slipping a clock cycle in the deserializer. Incoming data to the word aligner is aligned to the boundary of the word alignment pattern (K28.5). User logic is not required to manipulate the TX bit slipper for constant round-trip delay. In manual mode, the TX bit slipper is able to compensate one unit interval (UI).

The word alignment pattern (K28.5) position varies in byte deserialized data. Delay variation is up to ½ parallel clock cycle. You must add in extra user logic to manually check the K28.5 position in byte deserialized data for the actual latency.

Figure 4-32: Deterministic Latency State Machine in the Word Aligner

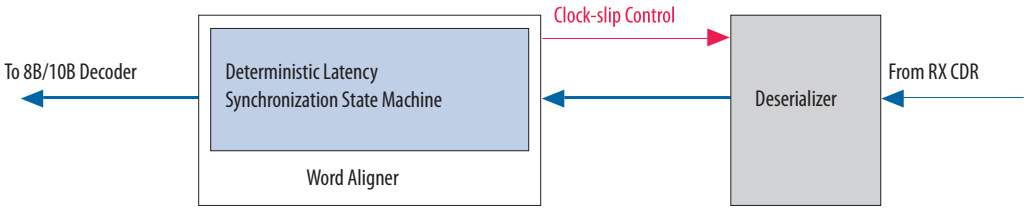


Table 4-11: Methods to Achieve Deterministic Latency Mode in Cyclone V Devices

Existing Feature		Enhanced Feature (14)	
Description	Requirement	Description	Requirement
Manual alignment with bit position indicator provides deterministic latency. Delay variation up to 1 parallel clock cycle	Extra user logic to manipulate the TX bit slipper with a bit position indicator from the word aligner for constant total round-trip delay	Deterministic latency state machine alignment reduces the known delay variation in word alignment operation	None

Related Information

Refer to the "Deterministic Latency PHY IP Core" chapter in the Altera Transceiver PHY IP Core User Guide

(14) Enhanced deterministic latency feature in Cyclone V devices.

Document Revision History

Table 4-12: Document Revision History

Date	Version	Changes
January 2016	2016.01.19	• Moved the Word Aligner block to the Receiver Standard PCS section in the "Transceiver Channel Datapath for XAUI Configuration" figure. • Changed the title of the table "Recommended Channel Placement for PCIe Gen2" to "Recommended Channel Placement for Full Duplex Transceiver Channels for PCIe Gen2".
September 2014	2014.09.30	• Added specific channel placement guidelines to the "PCIe Supported Configurations and Placement Guidelines" section. • Added a note to the "Hard IP Configurations for PCIe Gen1 and Gen2" table. • Added a note to the "Gigabit Ethernet" section. • Added a note to the "Rate Match FIFO" section in the "Gigabit Ethernet Transceiver Datapath" section. • Added channel placement guidelines in the XAUI "Transceiver Channel Placement Guidelines" section. • Added another example to the "Transceiver Channel Placement Guidelines in a XAUI Configuration" figure.
October 2013	2013.10.17	• Added clock frequency information to "6.144-Gbps Support Capability in Cyclone V GT Devices" section. • Removed fPLL information from the "Transceiver Clocking and Channel Placement Guidelines in XAUI Configuration" section. • Added Gen2 information to the "Hard IP Configurations for PCIe Gen1 and Gen2" table.

Date	Version	Changes
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Removed the "Receiver Electrical Idle Inference" section. • Added the "Recommended Channel Placement for PCIe Gen2" table. • Updated the figures in the "PCIe Supported Configurations and Placement Guidelines" section. • Added the "Transceiver Clocking Guidelines for Soft PCS Implementation" section. • Added the "6-Gbps Support Capability in Cyclone V GT Devices" section.
November 2012	2012.11.19	• Reorganized content and updated template. • Added the "XAUI" section. • Added the "PCI Express" section.
June 2012	1.1	• Updated for the Quartus II software version 12.0 release. • Updated Table 4–1. • Updated Figure 4–2. • Updated Figure 4–18. • Added the “Gigabit Ethernet” section. • Added the “Serial Digital Interface” section. • Added the “Serial Data Converter (SDC) JESD204” section. • Added the “SATA and SAS Protocols” section.
October 2011	1.0	Initial release.

Transceiver Custom Configurations in Cyclone V Devices

5

2014.09.30

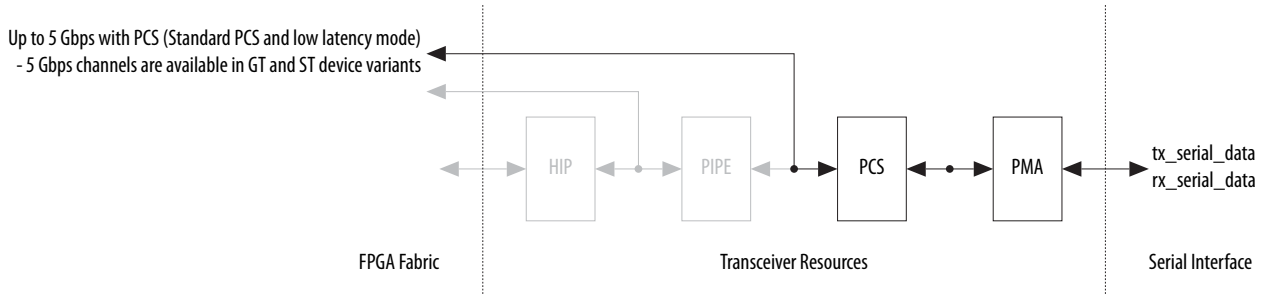
CV-53005 Subscribe Send Feedback

For integration with the FPGA fabric, the full-duplex transceiver channel supports custom configuration with physical medium attachment (PMA) and physical coding sublayer (PCS).

You can customize the transceiver with one of the following configurations:

- Standard PCS— Physical coding sublayer (PCS) and physical medium attachment (PMA)
- Standard PCS in low latency mode— Low latency PCS and PMA

Figure 5-1: Custom Configuration Options



Related Information

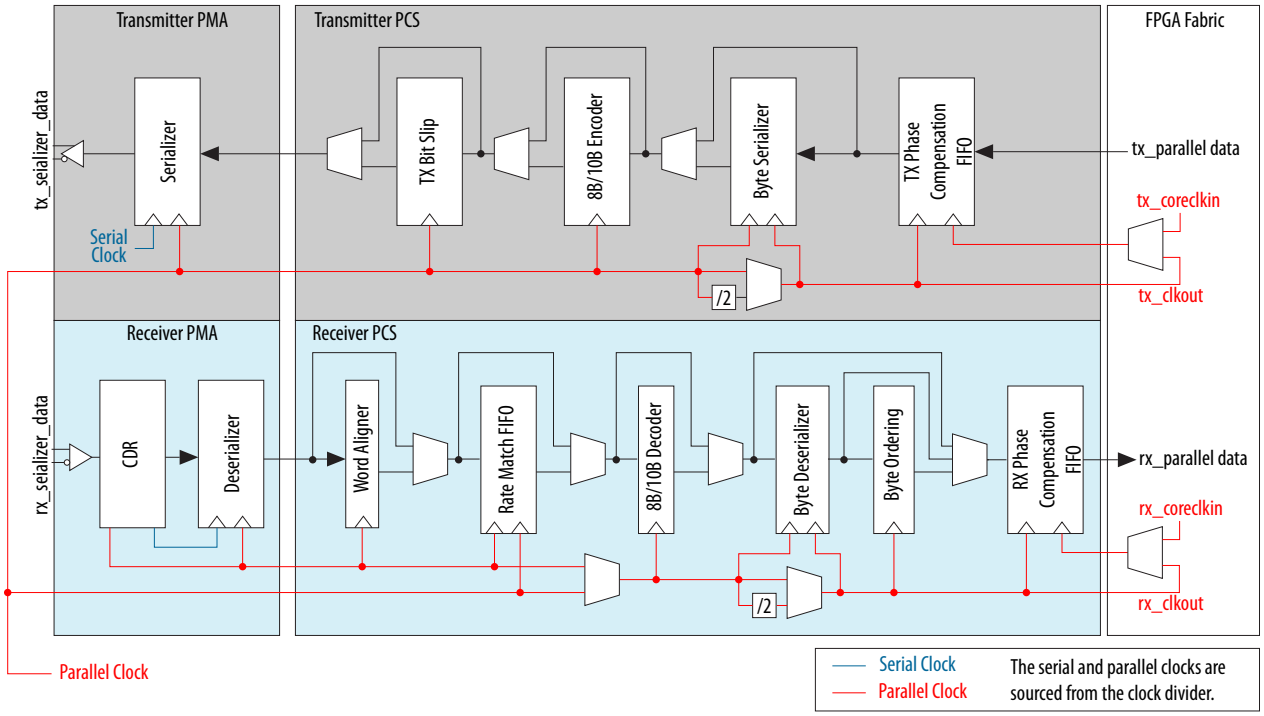
Cyclone V Device Handbook: Known Issues
Lists the planned updates to the *Cyclone V Device Handbook* chapters.

Standard PCS Configuration

In this configuration, you can customize the transceiver channel to include a PMA and PCS with functions that your application requires. The transceiver channel interfaces with the FPGA fabric through the PCS.

Figure 5-2: Complete Datapath in a Custom Configuration

Based on your application requirements, you can enable, modify, or disable the blocks, except the deskew FIFO block, as shown in the following figure.



Custom Configuration Channel Options

There are multiple channel options when you use Custom Configuration.

The supported interface width varies depending on the usage of the byte serializer/deserializer (SERDES), and the 8B/10B encoder or decoder. The byte serializer or deserializer is assumed to be enabled. Otherwise, the maximum data rate supported is half of the specified value.

The maximum supported data rate varies depending on the customization.

Table 5-1: Maximum Supported Data Rate

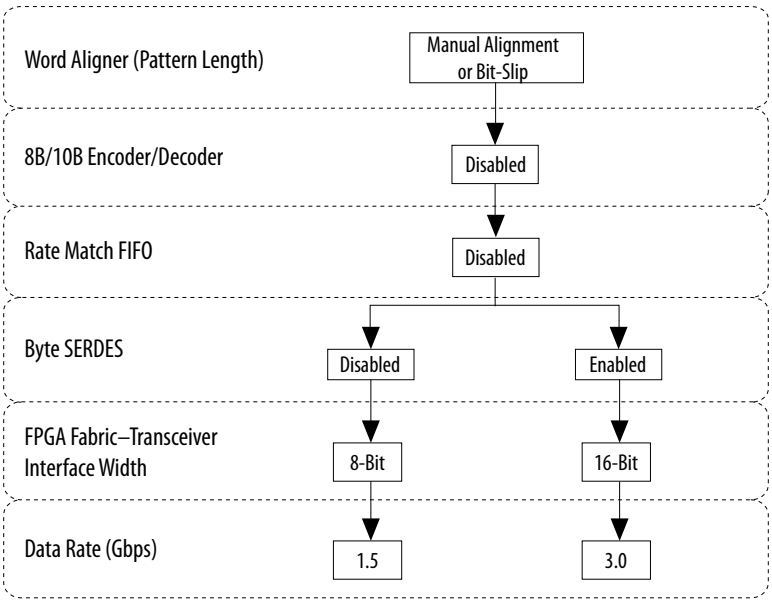
The following table shows the maximum supported data rate for the fastest speed grade in Standard PCS (transceiver speed grade 6) for Cyclone V GX and SX devices, and (transceiver speed grade 5) for Cyclone V GT and ST devices.

Data Configuration	PMA-PCS Interface Width	PCS-FPGA Fabric Interface Width		Maximum Data Rate for GX and SX (Mbps)	Maximum Data Rate for GT and ST (Mbps)
		8B/10B Enabled	8B/10B Disabled		
Single-width	8	—	8	1,500	1,500
			16	3,000	3,000
	10	8	10	1,875	1,875
		16	20	3,125	3,750
Double-width	16	—	16	2,621.44	2,621.44
			32 (15)	3,125	6,144
	20	16	20	3,125	3,276.8
		32 (15)	40 (15)	3,125	6,144

In all the supported configuration options of the channel, the transmitter bit-slip function is optional, where:

- The blocks shown as "Disabled" are not used but incur latency.
- The blocks shown as "Bypassed" are not used and do not incur any latency.
- The transmitter bit-slip is disabled.

Figure 5-3: Configuration Options for Custom Single-Width Mode (8-bit PMA–PCS Interface Width)



(15) The combination of –6 transceiver and –8 device speed grades is not supported when the **Enable Byte Serializer/Deserializer with 16- and 20-Bit PMA-PCS Widths** option is selected.

Figure 5-4: Configuration Options for Custom Single-Width Mode (10-bit PMA–PCS Interface Width)

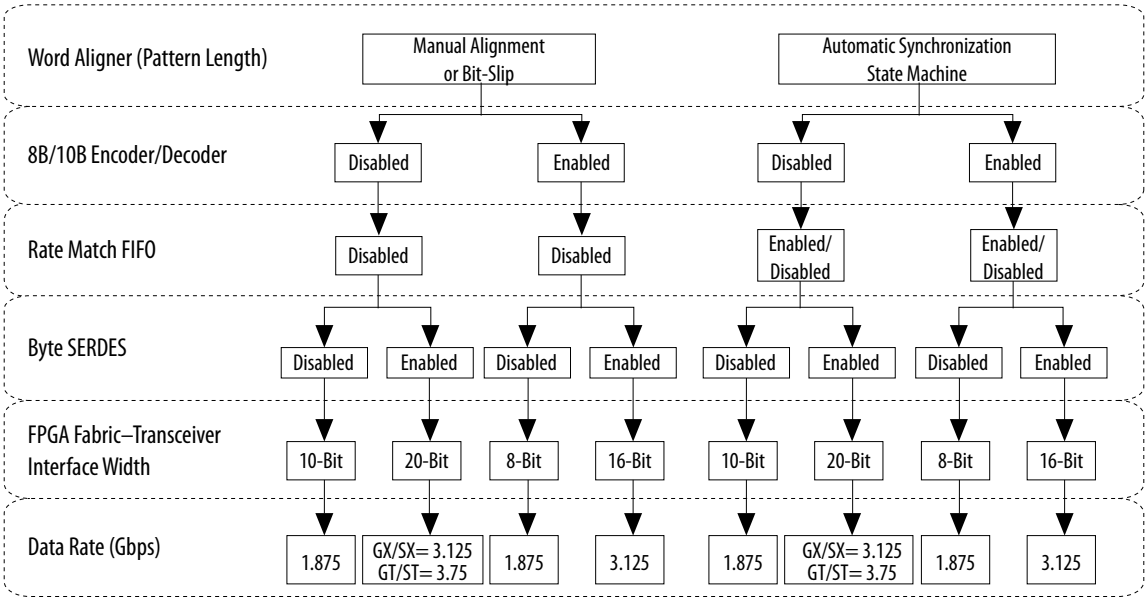


Figure 5-5: Configuration Options for Custom Double-Width Mode (16-bit PMA–PCS Interface Width)

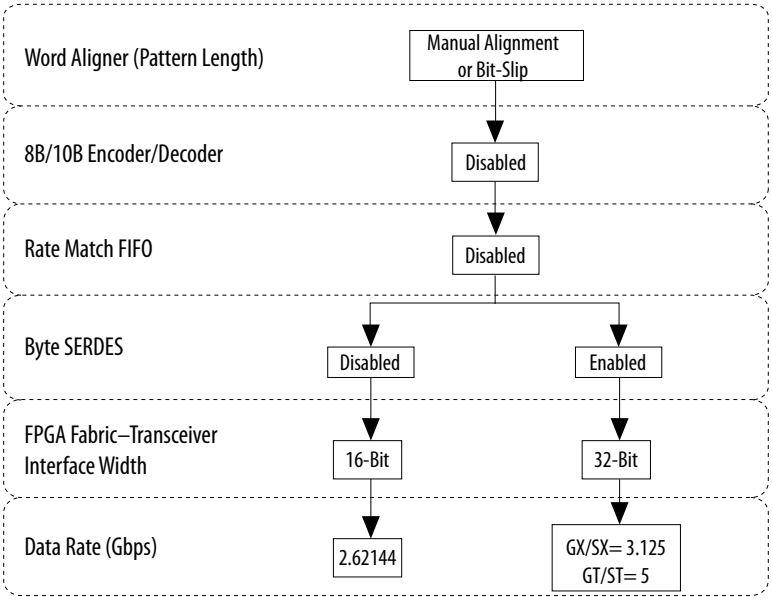
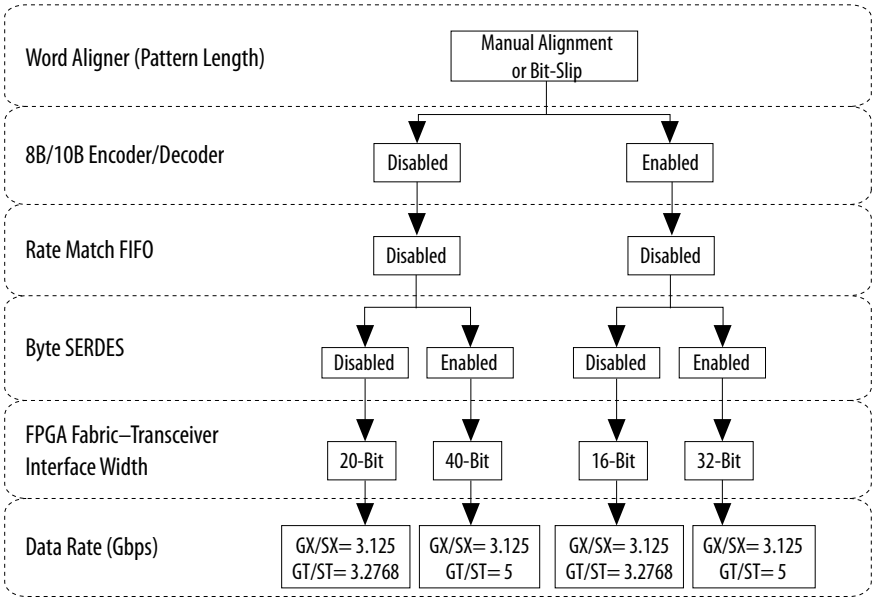


Figure 5-6: Configuration Options for Custom Double-Width Mode (20-bit PMA–PCS Interface Width)



Rate Match FIFO in Custom Configuration

In a custom configuration, the 20-bit pattern for the rate match FIFO is user-defined. The FIFO operates by looking for the 10-bit control pattern followed by the 10-bit skip pattern in the data, after the word aligner restores the word boundary. After finding the pattern, the FIFO performs a skip pattern insertion or deletion to ensure that the FIFO does not underflow or overflow a given parts per million (ppm) difference between the clocks.

The rate match FIFO operation requires 8B/10B-coded data.

Rate Match FIFO Behaviors in Custom Single-Width Mode

The different operations available in custom single-width mode for the rate match FIFO are symbol insertion, symbol deletion, full condition, and empty condition.

Table 5-2: Rate Match FIFO Behaviors in Custom Single-Width Mode (10-bit PMA–PCS Interface Width)

Operation	Behavior
Symbol Insertion	Inserts a maximum of four skip patterns in a cluster, only if there are no more than five skip patterns in the cluster after the symbol insertion.
Symbol Deletion	Deletes a maximum of four skip patterns in a cluster, only if there is one skip pattern left in the cluster after the symbol deletion.
Full Condition	Deletes the data byte that causes the FIFO to go full.
Empty Condition	Inserts a /K30.7/ (9'h1FE) after the data byte that caused the FIFO to go empty.

Rate Match FIFO Behaviors in Custom Double-Width Mode

The different operations available in custom double-width mode for the rate match FIFO are symbol insertion, symbol deletion, full condition, and empty condition.

Table 5-3: Rate Match FIFO Behaviors in Custom Double-Width Mode (20-bit PMA–PCS Interface Width)

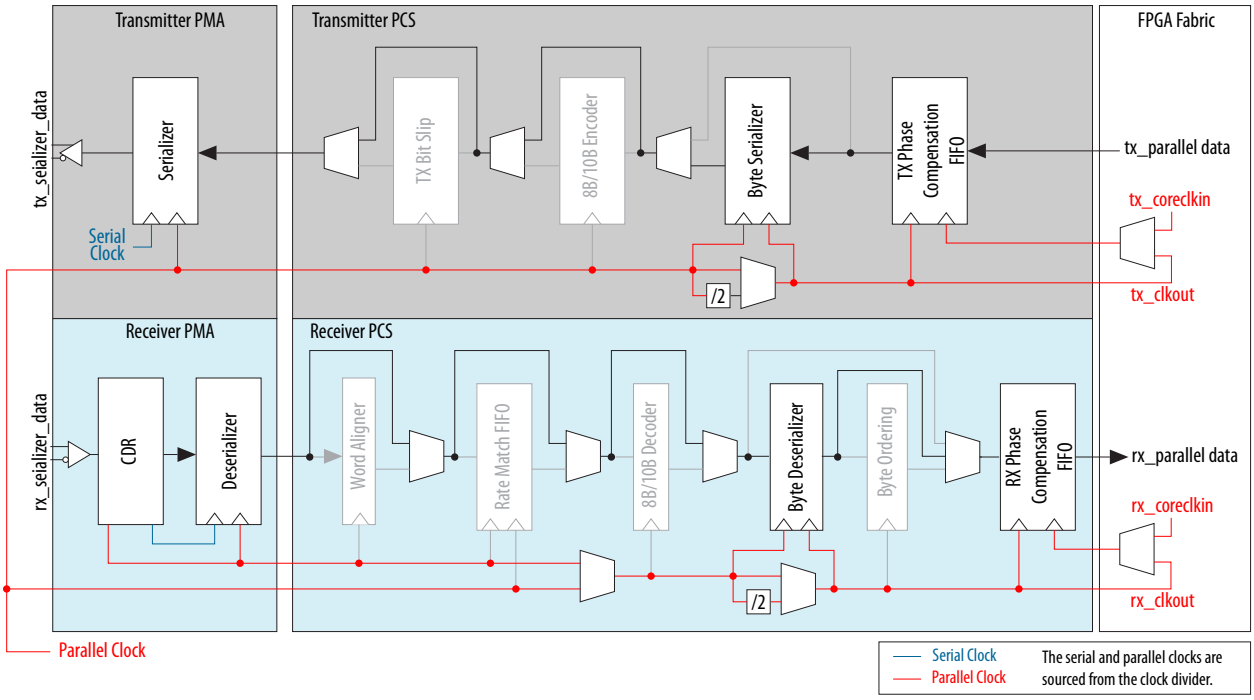
Operation	Behavior
Symbol Insertion	Inserts as many pairs (10-bit skip patterns at the LSByte and MSByte of the 20-bit word at the same clock cycle) of skip patterns as needed.
Symbol Deletion	Deletes as many pairs (10-bit skip patterns at the LSByte and MSByte of the 20-bit word at the same clock cycle) of skip patterns as needed.
Full Condition	Deletes the pair (20-bit word) of data bytes that causes the FIFO to go full.
Empty Condition	Inserts a pair of /K30.7/ ({9'h1FE, 9'h1FE}) after the data byte that causes the FIFO to go empty.

Standard PCS in Low Latency Configuration

In this configuration, you can customize the transceiver channel to include a PMA and PCS that bypasses most of the PCS logical functionality for a low latency datapath.

To provide a low latency datapath, the PCS includes only the phase compensation FIFO in phase compensation mode, and optionally, the byte serializer and byte deserializer blocks, as shown in the following figure. The transceiver channel interfaces with the FPGA fabric through the PCS.

Figure 5-7: Datapath in Low Latency Custom Configuration



The maximum supported data rate varies depending on the customization and is identical to the custom configuration except that the 8B/10B block is disabled

Low Latency Custom Configuration Channel Options

There are multiple channel options when you use Low Latency Custom Configuration.

In the following figures:

- The blocks shown as “Disabled” are not used but incur latency.
- The blocks shown as “Bypassed” are not used and do not incur any latency.
- The transmitter bit-slip is disabled.

Figure 5-8: Configuration Options for Low Latency Custom Single-Width Mode (8-bit PMA–PCS Interface Width)

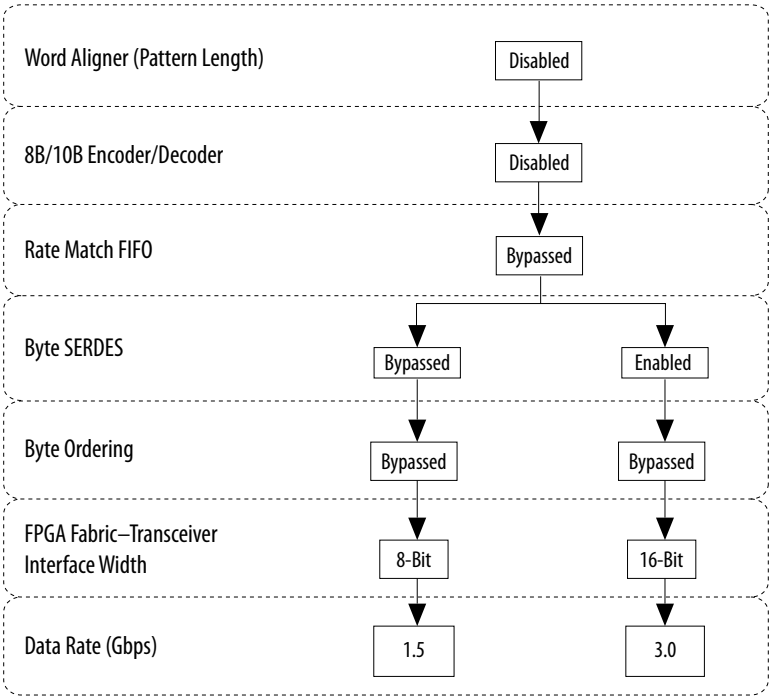


Figure 5-9: Configuration Options for Low Latency Custom Single-Width Mode (10-bit PMA–PCS Interface Width)

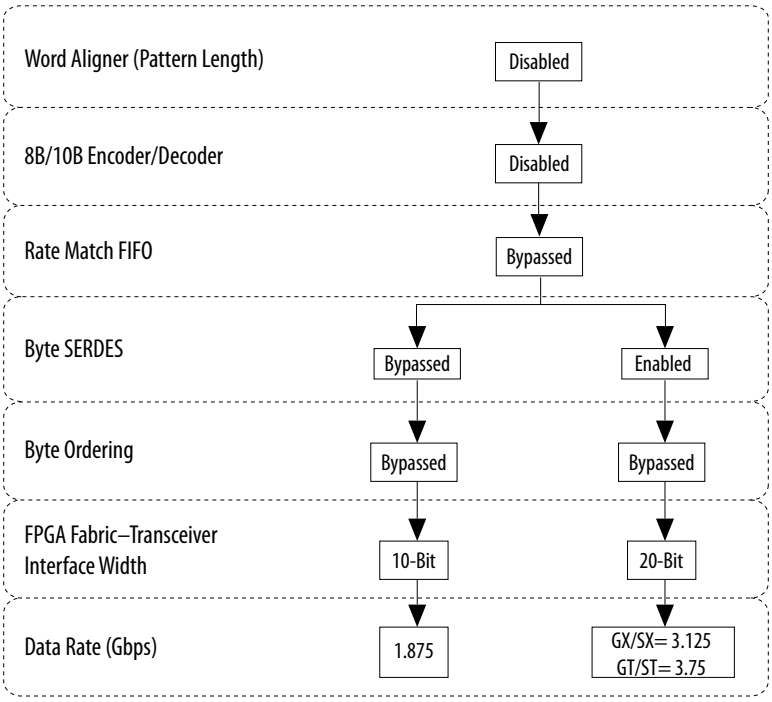


Figure 5-10: Configuration Options for Low Latency Custom Double-Width Mode (16-bit PMA–PCS Interface Width)

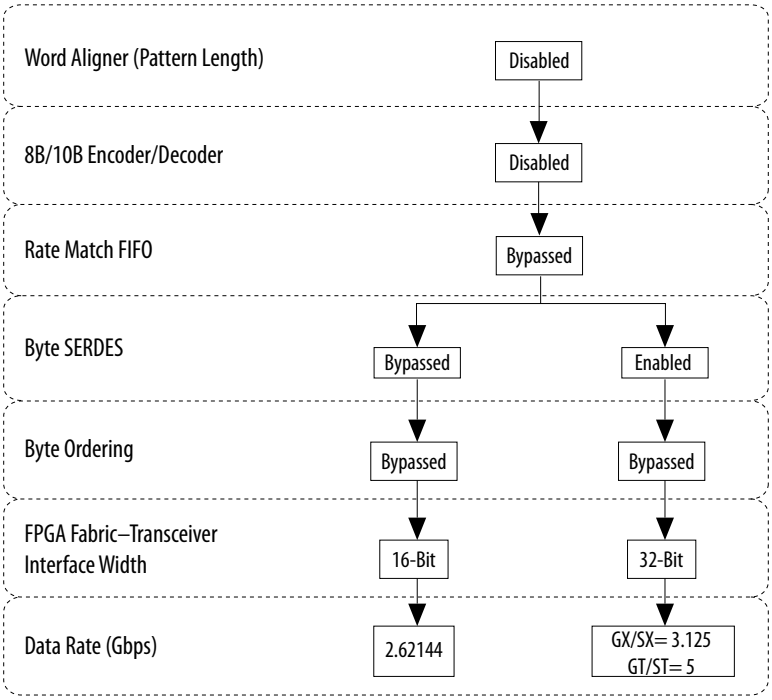
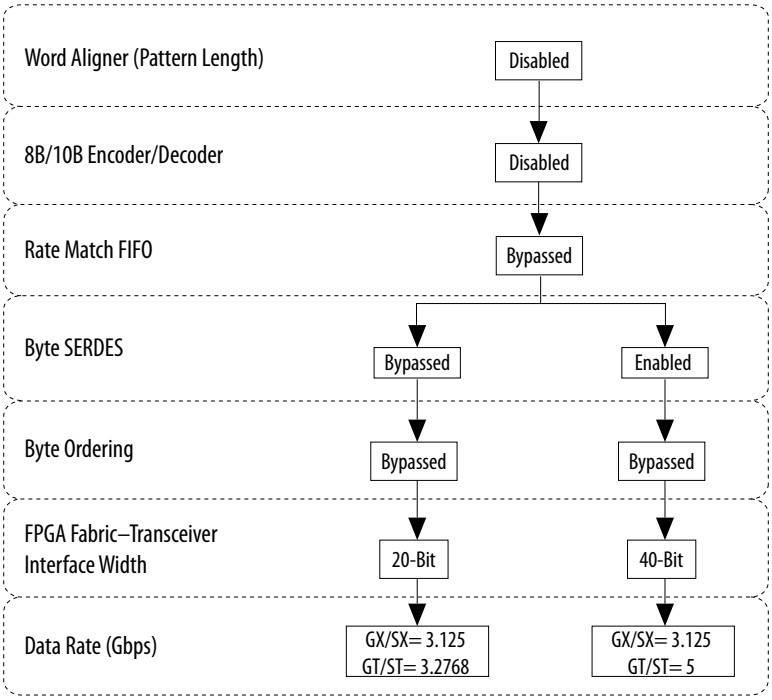


Figure 5-11: Configuration Options for Low Latency Custom Double-Width Mode (20-bit PMA–PCS Interface Width)



Document Revision History

Table 5-4: Document Revision History

Date	Version	Changes
September 2014	2014.09.30	Changed the "Maximum Supported Data Rate" table: • Changed the maximum data rate for double-width GT devices to 6,144 Mbps. • Added a footnote to several cells about a limitation for certain speed grade cases.
May 2013	2013.05.06	Added link to the known document issues in the Knowledge Base.
November 2012	2012.11.19	Reorganized content and updated template.
June 2012	1.1	Updated for the Quartus II software version 12.0 release.
October 2011	1.0	Initial release.

Transceiver Loopback Support

6

2013.05.06

CV-53006

Subscribe Send Feedback

The Cyclone V loopback options allow you to verify how different functional blocks work in the transceiver.

Related Information

Cyclone V Device Handbook: Known Issues
Lists the planned updates to the Cyclone V Device Handbook chapters.

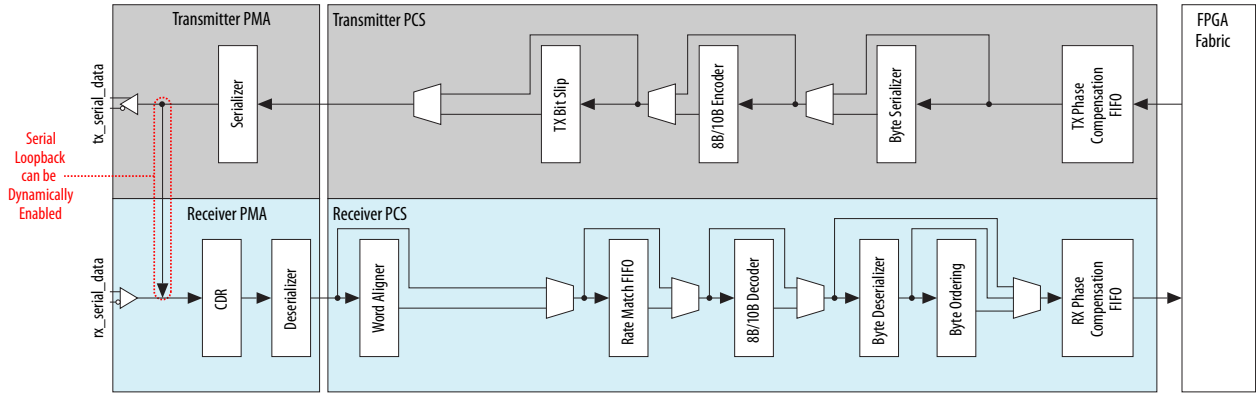
Serial Loopback

This section describes the use of serial loopback as a debugging aid to ensure the enabled PCS and PMA blocks in the transmitter and receiver channels function correctly.

Serial loopback is available for all transceiver configurations except the PIPE mode. You can use serial loopback as a debugging aid to ensure that the enabled physical coding sublayer (PCS) and physical media attachment (PMA) blocks in the transmitter and receiver channels are functioning correctly. Furthermore, you can dynamically enable serial loopback on a channel-by-channel basis.

The data from the FPGA fabric passes through the transmitter channel and is looped back to the receiver channel, bypassing the receiver buffer. The received data is available to the FPGA logic for verification.

Figure 6-1: Serial Loopback Datapath



When you enable serial loopback, the transmitter channel sends data to both the `tx_serial_data` output port and to the receiver channel. The differential output voltage on the `tx_serial_data` port is based on the selected differential output voltage (V_{OD}) settings.

The looped-back data is forwarded to the receiver clock data recovery (CDR). You must provide an alignment pattern for the word aligner to enable the receiver channel to retrieve the byte boundary.

If the device is not in the serial loopback configuration and is receiving data from a remote device, the recovered clock from the receiver CDR is locked to the data from the remote source.

Note: For the `phy_serial_loopback` register access description and addressing, refer to the "Loopback Modes" section of the *Transceiver Reconfiguration Controller* chapter in the *Altera Transceiver PHY IP Core User Guide*.

If the device is placed in the serial loopback configuration, the data source to the receiver changes from the remote device to the local transmitter channel—prompting the receiver CDR to start tracking the phase of the new data source. During this time, the recovered clock from the receiver CDR may be unstable. Because the receiver PCS is running off of this recovered clock, you must place the receiver PCS under reset by asserting the `rx_digitalreset` signal during this period.

Note: When moving into or out of serial loopback, you must assert the `rx_digitalreset` signal for a minimum of two parallel clock cycles.

Related Information

Altera Transceiver PHY IP Core User Guide

Forward Parallel Loopback

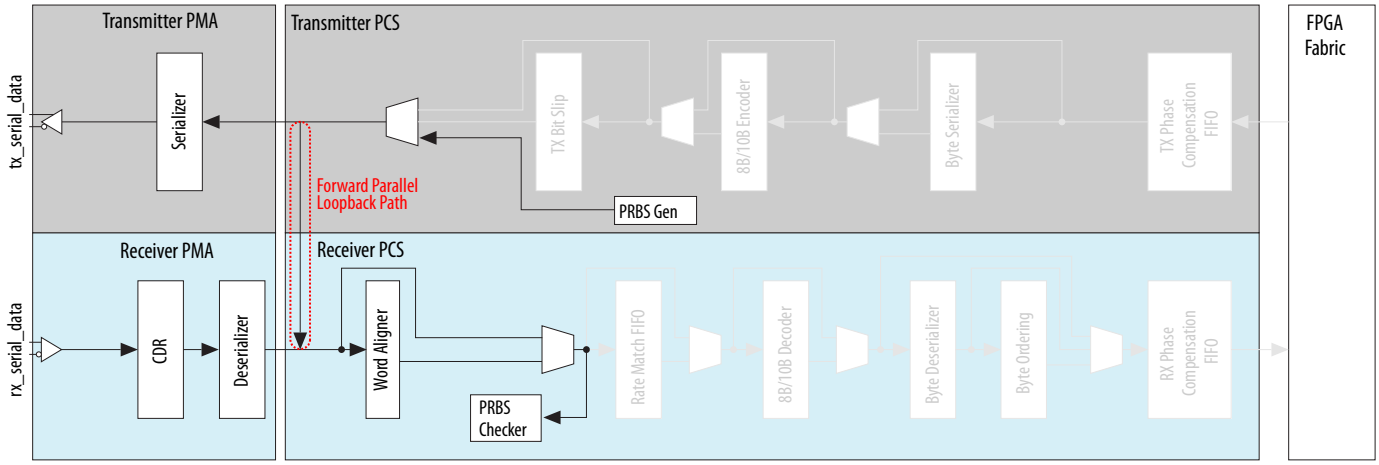
Forward parallel loopback is a debugging aid to ensure the enabled PCS blocks in the transmitter and receiver channel function correctly.

Forward parallel loopback is only available in transceiver Native PHY. You enable forward parallel loopback by enabling the PRBS test mode, through the dynamic reconfiguration controller. You must perform a `rx_digitalreset` after the dynamic reconfiguration operation has completed.

Parallel data travels across the forward parallel loopback path, passing through the RX word aligner, and finally verified inside the RX PCS PRBS verifier block. Check the operations status from the FPGA fabric.

Figure 6-2: Parallel Loopback Datapath

The following figure shows the parallel PRBS data generated by the TX PCS PRBS generator block.



Note: Usage details for the feature are described in the Altera Transceiver PHY IP Core User Guide.

Related Information

Altera Transceiver PHY IP Core User Guide

PIPE Reverse Parallel Loopback

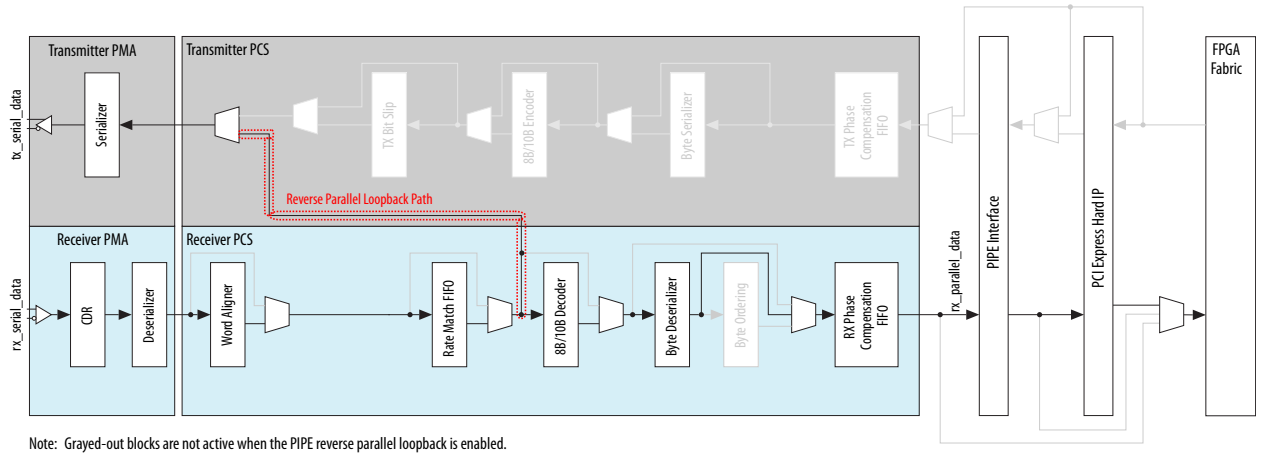
This section describes PIPE Reverse Parallel Loopback debugging option using parallel data through the rate match FIFO, transmitter serializer, and `tx_serial_data` port path.

PIPE reverse parallel loopback is only available in the PCIe® configuration for Gen1 and Gen2 data rates. Figure 2 shows the received serial data passing through the receiver CDR, deserializer, word aligner, and rate match FIFO buffer. The parallel data from the rate match FIFO is then looped back to the transmitter serializer and transmitted out through the `tx_serial_data` port. The received data is also available to the FPGA fabric through the `rx_parallel_data` signal.

PIPE reverse parallel loopback is compliant with the PCIe 2.0 specification. To enable this loopback configuration, assert the `tx_detectrx_loopback` signal.

Note: PIPE reverse parallel loopback is the only loopback option supported in the PCIe configuration.

Figure 6-3: PIPE Reverse Parallel Loopback Configuration Datapath



Note: Grayed-out blocks are not active when the PIPE reverse parallel loopback is enabled.

Reverse Serial Loopback

You can use the reverse serial loopback option to debug with data through the `rx_serial_data port`, receiver CDR, and `tx_serial_data` port path.

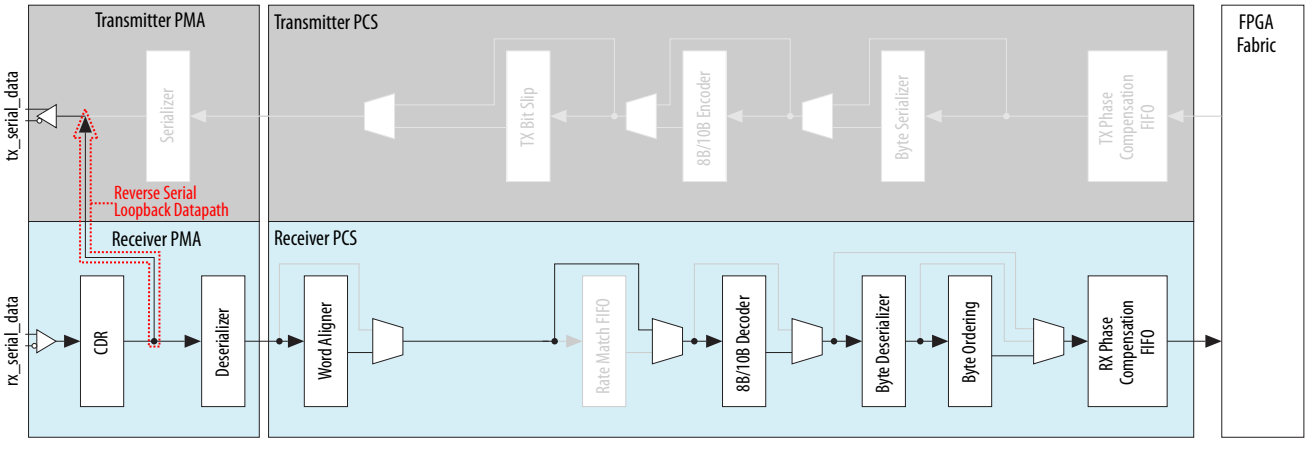
You can enable reverse serial loopback through the reconfiguration controller.

Note: For further details, refer to the Altera Transceiver PHY IP Core User Guide.

In reverse serial loopback, the data is received through the `rx_serial_data` port, re-timed through the receiver CDR, and sent to the `tx_serial_data` port. The received data is also available to the FPGA logic. No dynamic pin control is available to select or deselect reverse serial loopback.

The transmitter buffer is the only active block in the transmitter channel. You can change the V_{OD} and the pre-emphasis first post tap values on the transmitter buffer through the dynamic reconfiguration controller. Reverse serial loopback is often implemented when using a bit error rate tester (BERT) on the upstream transmitter.

Figure 6-4: Reverse Serial Loopback Datapath



Note: Grayed-out blocks are not active when the reverse serial loopback is enabled.

Related Information

Altera Transceiver PHY IP Core User Guide

Reverse Serial Pre-CDR Loopback

This section describes reverse serial pre-CDR loopback debugging with a data path through the `rx_serial_data` port to the `tx_serial_data` port, and before the receiver CDR.

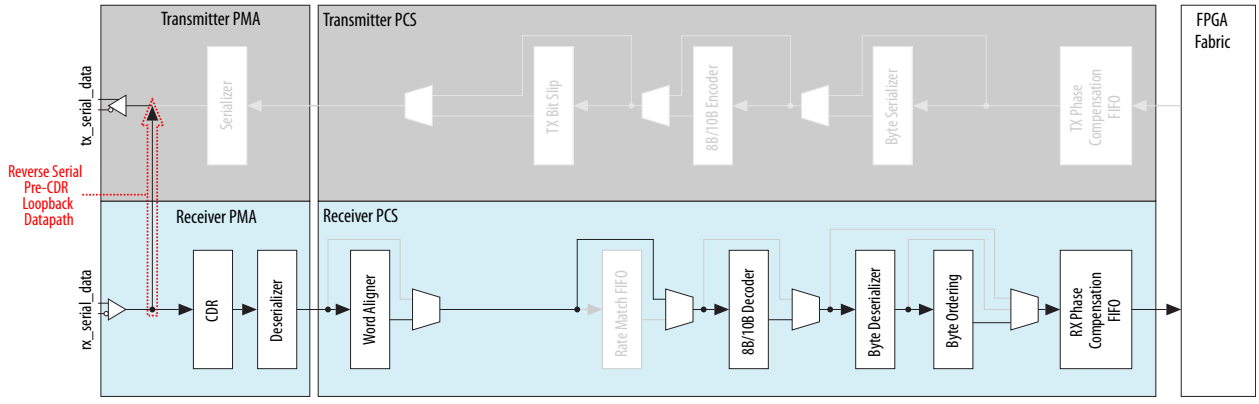
You can enable reverse serial pre-CDR loopback through the reconfiguration controller.

Note: For further details, refer to the Altera Transceiver PHY IP Core User Guide.

In reverse serial pre-CDR loopback, the data received through the `rx_serial_data` port is looped back to the `tx_serial_data` port before the receiver CDR. The received data is also available to the FPGA logic. No dynamic pin control is available to select or deselect reverse serial pre-CDR loopback.

The transmitter buffer is the only active block in the transmitter channel. You can change the VOD on the transmitter buffer through the dynamic reconfiguration controller. The pre-emphasis settings for the transmitter buffer cannot be changed in this configuration.

Figure 6-5: Reverse Serial Pre-CDR Loopback Datapath



Note: Grayed-out blocks are not active when the reverse serial pre-CDR loopback is enabled.

Related Information

Altera Transceiver PHY IP Core User Guide

Document Revision History

The table below lists the revision history for this chapter.

Table 6-1: Document Revision History

Date	Version	Changes
May 2013	2013.05.06	• Added the Forward Parallel Loopback topic. • Updated the Reverse Serial Loopback topic. • Updated the Reverse Serial Pre-CDR Loopback topic. • Added link to the known document issues in the Knowledge Base.
November 2012	2012.11.19	• Reorganized content and updated template. • Minor updates for the Quartus II software version 12.1.
June 2012	1.0	Initial release.

Dynamic Reconfiguration in Cyclone V Devices

7

2014.09.30

CV-53007

Subscribe Send Feedback

The transceiver reconfiguration controller offers several different dynamic reconfiguration modes. You can choose the appropriate reconfiguration mode that best suits your application needs. All the dynamic reconfiguration modes are implemented through the transceiver Reconfiguration Controller PHY IP.

Related Information

Cyclone V Device Handbook: Known Issues
Lists the planned updates to the *Cyclone V Device Handbook* chapters.

Dynamic Reconfiguration Features

The following table lists the available dynamic reconfiguration features.

Table 7-1: Reconfiguration Features

Reconfiguration Feature	Description	Affected Blocks
Offset Cancellation	Counter offset variations due to process operation for the analog circuit. This feature is mandatory if you use receivers.	CDR
DCD Calibration	Compensates for the duty cycle distortion caused by clock network skew.	TX buffer and clock network skew
Analog Controls Reconfiguration	Fine-tune signal integrity by adjusting the transmitter (TX) and receiver (RX) analog settings while bringing up a link.	Analog circuit of TX and RX buffer
Loopback Modes	Enable or disable Pre- and Post-CDR Reverse Serial Loopback dynamically.	PMA

Reconfiguration Feature	Description	Affected Blocks
Data Rate Change	Increase or decrease the data rate (/1, /2, /4, /8) for autonegotiation purposes such as CPRI and SATA/SAS applications	TX Local clock dividers
	Reconfigure the TX PLL settings for protocols with multi-data rate support such as CPRI	TX PLL
	Switch between multiple TX PLLs for multi-data rate support	• TX PLL • Fractional PLL (Reconfigure the fPLL data rate with the ALTERA_PLL_RECONFIG megafunction.)
	Channel reconfiguration—Reconfigure the RX CDR from one data rate to another data rate	CDR
	FPGA fabric - transceiver channel data width reconfiguration	FPGA fabric - transceiver channel interface.

Related Information

- **Backplane Applications with 28 nm FPGAs**
- **AN661: Implementing Fractional PLL Reconfiguration with ALTERA_PLL and ALTERA_PLL_RECONFIG Megafunctions**
  For information about reconfiguration of the fPLL data rate.

Offset Cancellation

Offset cancellation adjusts the offsets within the CDR parameters for process variations.

Every transceiver channel has offset cancellation circuitry to compensate for the offset variations that are caused by process operations. The offset cancellation circuitry is controlled by the offset cancellation control logic IP within the Transceiver Reconfiguration Controller. Resetting the Transceiver Reconfiguration Controller during user mode does not trigger the offset cancellation process.

When offset cancellation calibration is complete, the `reconfig_busy` status signal is deasserted to indicate the completion of the process.

The clock (`mgmt_clk_clk`) used by the Transceiver Reconfiguration Controller is also used for transceiver calibration and must be 75-125 MHz if the Hard IP for PCIe Express IP core is not enabled. When the Hard IP for PCIe Express is enabled, the frequency range is 75-100 MHz. If the clock (`mgmt_clk_clk`) is not free-running, hold the reconfiguration controller reset (`mgmt_rst_reset`) until the clock is stable.

Transmitter Duty Cycle Distortion Calibration

The duty cycle calibration function tunes the transmitter to minimize duty cycle distortion.

The transmitter clocks generated by the CMU that travel across the clock network may introduce duty cycle distortions (DCD). Reduce DCD with the DCD calibration IP that is integrated in the transceiver reconfiguration controller.

Enabled the DCD calibration IP in Cyclone GT devices for better TX jitter performance, if either of the following conditions are met:

- Data rate is ≥ 4915.2 Mbps
- Clock network switching (TX PLL switching) and the data rate is ≥ 4915.2 Mbps

The following DCD calibration modes are supported:

- DCD calibration at power-up mode
- Manual DCD calibration during user mode

DCD calibration is performed automatically after device configuration and before entering user mode if the transceiver channels connected have the **Calibrate duty cycle during power up** option enabled. You can optionally trigger DCD calibration manually during user mode if:

- You reconfigure the transceiver from lower data rates to higher data rates (≥ 4.9152 Gbps)
- You perform clock network switching (TX PLL switching) and switch to data rates of ≥ 4915.2 Mbps

You do not have to enable DCD calibration if the transceivers are operating below 4.9152 Gbps.

When DCD calibration and offset cancellation are enabled, the `reconfig_busy` status signal from the reconfiguration controller is deasserted to indicate the completion of both processes. If DCD calibration is not enabled, the deassertion of `reconfig_busy` signal indicates the completion of the offset cancellation process.

Related Information

- **AN 676: Using the Transceiver Reconfiguration Controller for Dynamic Reconfiguration in Arria V and Cyclone V Devices**
- **Altera Transceiver PHY IP Core User Guide**

PMA Analog Controls Reconfiguration

You can dynamically reconfigure the analog controls setting after offset cancellation is complete and the reset sequence is performed. You can continue with the subsequent reconfigurations of the analog controls when the `reconfig_busy` status signal is low. A high on the `reconfig_busy` signal indicates that the reconfiguration operation is in progress.

You can reconfigure the following transceiver analog controls:

- Transmitter pre-emphasis
- Differential output voltage (V_{OD})
- Receiver equalizer control
- Direct-current (DC) gain settings

To reconfigure the analog control settings, perform read and write operations to the PMA analog settings reconfiguration control IP within the reconfiguration controller.

Related Information

Altera Transceiver PHY IP Core User Guide
For information about the read and write operations with the reconfiguration controller

Dynamic Reconfiguration of Loopback Modes

You can enable the pre- and post-CDR reverse serial loopback modes by writing the appropriate bits of the Transceiver Reconfiguration Controller.

The following loopback paths are available:

- **Post-CDR reverse serial loopback path**— The RX captures the input data and feeds it into the CDR. The recovered data from the CDR output feeds into the TX driver and sends to the TX pins through the TX driver. For this path, the RX and CDR can be tested. For this path, the TX driver can be programmed to use either the main tap only or the main tap and the pre-emphasis first post-tap. Enabling or disabling the post-CDR reverse serial loopback modes is done through the PMA Analog Reconfiguration IP in the Transceiver Reconfiguration PHY IP.
- **Pre-CDR reverse serial loopback path**— The RX captures the input data and feeds it back to the TX driver through a buffer. With this path, you can perform a quick check for the quality of the RX and TX buffers. Enabling or disabling the pre-CDR reverse serial loopback mode.

Note: Serial loopback can be implemented with the transceiver PHY IP directly using the Avalon interface or a control port.

Related Information

- **Transceiver Reconfiguration Controller chapter of the Altera Transceiver PHY IP Core User Guide**
- **Transceiver Loopback Support in Cyclone V Devices**

Transceiver PLL Reconfiguration

You can use the PLL reconfiguration registers to switch the reference clock input to the TX PLL or the clock data recovery (CDR) circuitry.

For example, you can switch the reference clock from 100 MHz to 125 MHz. You can also change the data rate from 2.5 Gbps to 5 Gbps by reconfiguring the transmitter PLL connected to the transceiver channel.

Note: Reference clock switching is only supported on the dedicated REFCLK pin.

The Transceiver Reconfiguration PHY IP provides an Avalon ® -MM user interface to perform PLL reconfiguration.

Related Information

"PLL Reconfiguration" section in the Transceiver Reconfiguration Controller chapter of the Altera Transceiver PHY IP Core User Guide
For information about performing PLL reconfiguration.

Transceiver Channel Reconfiguration

You can use channel reconfiguration to dynamically reconfigure the channel in a transceiver PHY IP core. Among the settings that you can change dynamically are the data rate and interface width.

You can reconfigure the channels in the following ways:

- Reconfigure the CDR of the receiver channel.
- Enable and disable all static PCS sub-blocks.
- Select an alternate PLL within the transceiver block to supply a different clock to the transceiver clock generation block.
- Reconfigure the TX local clock divider with a 1, 2, 4, or 8 division factor.

Every transmitter channel has a clock divider. When you reconfigure these clock dividers, ensure that the functional mode of the transceiver channel supports the reconfigured data rate.

Transceiver Interface Reconfiguration

You can reconfigure the transceiver interfaces by reconfiguring the FPGA fabric transceiver channel data width that includes PCS-PLD and PMA-PCS interfaces.

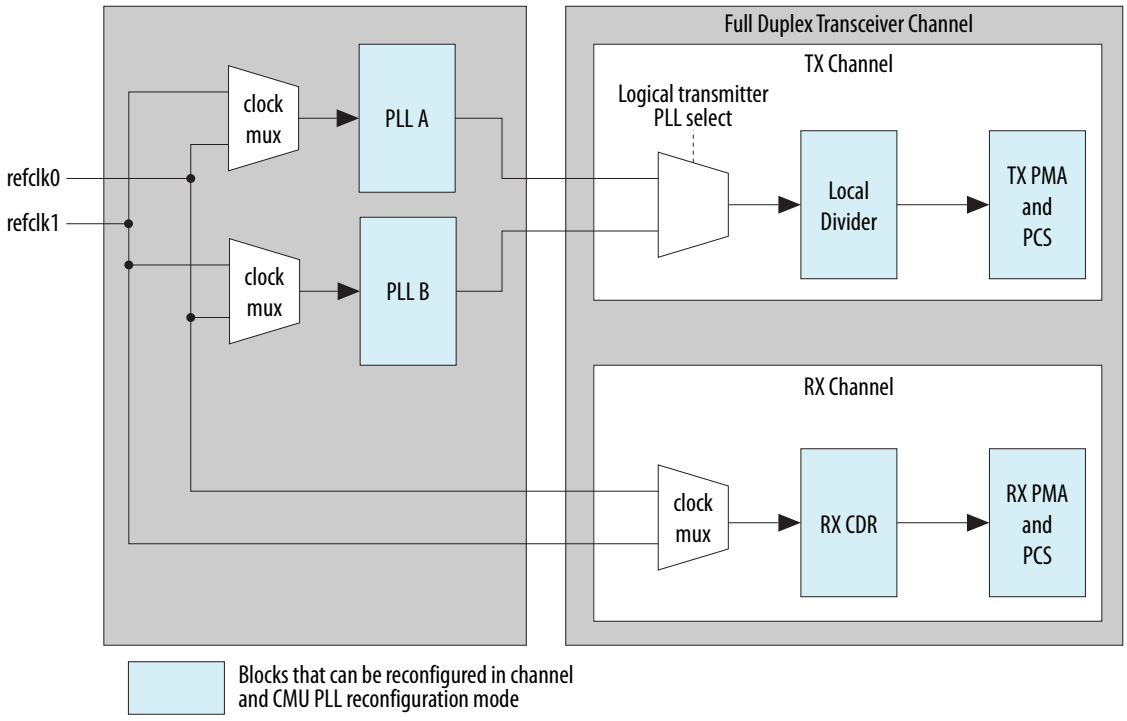
For example, you can reconfigure the custom PHY IP to enable or disable the 8B/10B encoder/decoder. There is no limit to the number of functional modes you can reconfigure the transceiver channel to if the various clocks involved support the transition. When you switch the custom PHY IP from one function mode to a different function mode, you may need to reconfigure the FPGA fabric-transceiver channel data width, enable or disable PCS sub-blocks, or both, to comply with the protocol requirements.

Channel reconfiguration only affects the channel involved in the reconfiguration (the transceiver channel specified by the unique logical channel address), without affecting the remaining transceiver channels controlled by the same Transceiver Reconfiguration Controller. PLL reconfiguration affects all channels that are currently using that PLL for transmission.

Channel reconfiguration from either a transmitter-only configuration to a receiver-only configuration or vice versa is not allowed.

Figure 7-1: Transceiver Channel and PLL Reconfiguration in a Transceiver Block

The following figure shows the functional blocks you dynamically reconfigure using transceiver channel and PLL reconfiguration mode.



Related Information

"Channel and PLL Reconfiguration" section in the Transceiver Reconfiguration Controller chapter of the Altera Transceiver PHY IP Core User Guide

For information about transceiver channel and PLL reconfiguration.

Reduced .mif Reconfiguration

Reduce reconfiguration time by reconfiguring only the affected blocks in the transceiver channels.

This reconfiguration mode affects only the modified settings of the channel to reduce reconfiguration time significantly. For example, in SATA/SAS applications, auto-rate negotiation must be completed within a short period of time to meet the protocol specification. The reduced **.mif** method helps to reconfigure the channel to meet these specifications. You can generate the reduced **.mif** files manually or by using the **xcvr_diffmifgen.exe** utility.

Related Information

Altera Transceiver PHY IP Core User Guide

For information about reduced **.mif** creation.

Unsupported Reconfiguration Modes

The following reconfiguration modes are not supported:

- Switching between a receiver-only channel and a transmitter-only channel
- Switching between bonded to non-bonded mode or bonded mode with different xN lanes count (for example, switching from bonded x2 to bonded x4)
- Switching between one PHY IP to another PHY IP (for example Deterministic Latency PHY IP to Custom PHY IP)
- TX PLL reconfiguration is not supported if the TX PLL is connected to bonded channels

You can achieve PHY to PHY IP reconfiguration only if you use the Native PHY IP to configure your transceiver. For example, if you use the Native PHY IP to configure the SDI mode and a custom proprietary IP mode (also configured using Native PHY IP), you can reconfigure these two modes within the Native PHY IP. The switching refers to enabling and disabling the PCS sub-blocks for both SDI and the custom proprietary IP mode.

Document Revision History

Date	Version	Changes
September 2014	2014.09.30	Added FPGA fabric to transceiver channel interface width reconfiguration feature in *Table: Dynamic Reconfiguration Features.*
May 2013	2013.05.06	• Updated TX DCD calibration information • Included link AN 661 for fPLL reconfiguration • Added link to the known document issues in the Knowledge Base.
November 2012	2012.11.19	• Rewritten and reorganized content, and updated template • Added TX DCD • Added Transceiver PLL Reconfiguration • Transceiver Channel Reconfiguration • Listed unsupported features