THE DATASHEET OF

5SGXMA9N2F45C2N

Stratix V Device Handbook

Volume 1: Device Interfaces and Integration

SV5V1
2015.01.23

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

Stratix V Device Handbook

Volume 2: Transceivers

Subscribe

Send Feedback

SV5V2
2014.09.30

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

Contents

Transceiver Architecture in Stratix V Devices

1

2014.09.30

SV52002 Subscribe Send Feedback

For a complete understanding of Stratix® V transceivers, first review the transceiver architecture chapter, then refer to the subsequent chapters in this volume.

You can implement Stratix V transceivers using Altera's transceiver intellectual property (IP) which are part of the Quartus® II software.

Stratix V devices provide up to 66 backplane-capable full-duplex clock data recovery (CDR)–based transceivers.

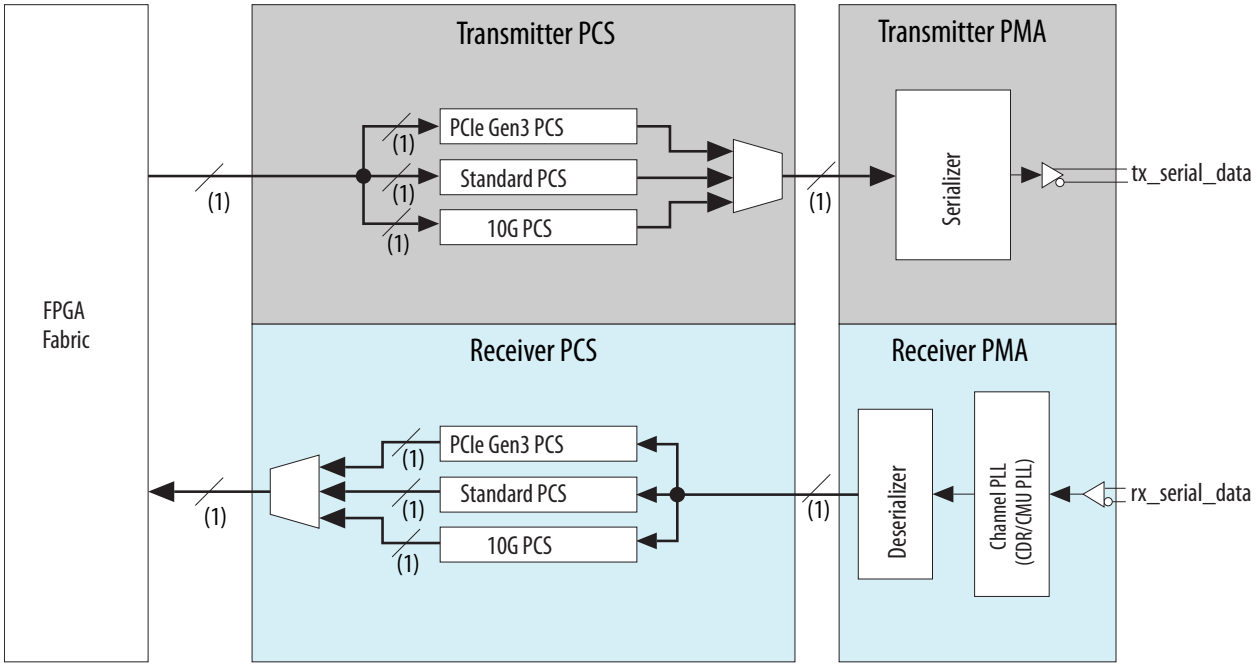
Table 1-1: Device Variants

Stratix Device	Channel Type	
	GX	GT
GS	600 Mbps to 14.1 Gbps	Not supported
GX	600 Mbps to 14.1 Gbps	Not supported
GT	600 Mbps to 12.5 Gbps	19.6 Gbps to 28.05 Gbps

Stratix V transceivers are divided into two blocks: physical medium attachment (PMA) and physical coding sublayer (PCS). The PMA block connects the FPGA to the channel, generates the required clocks, and converts the data from parallel to serial or serial to parallel. The PCS block performs digital processing logic between the PMA and the FPGA core. The PCS block contains the digital processing interface between the PMA and FPGA core. There are three types of PCS blocks in Stratix V devices: a standard PCS block, a 10G PCS, and a PCIe Gen3 PCS that supports the PCIe Gen3 Base specification.

Stratix V transceivers are structured into full-duplex (transmitter and receiver) six-channel groups called transceiver blocks.

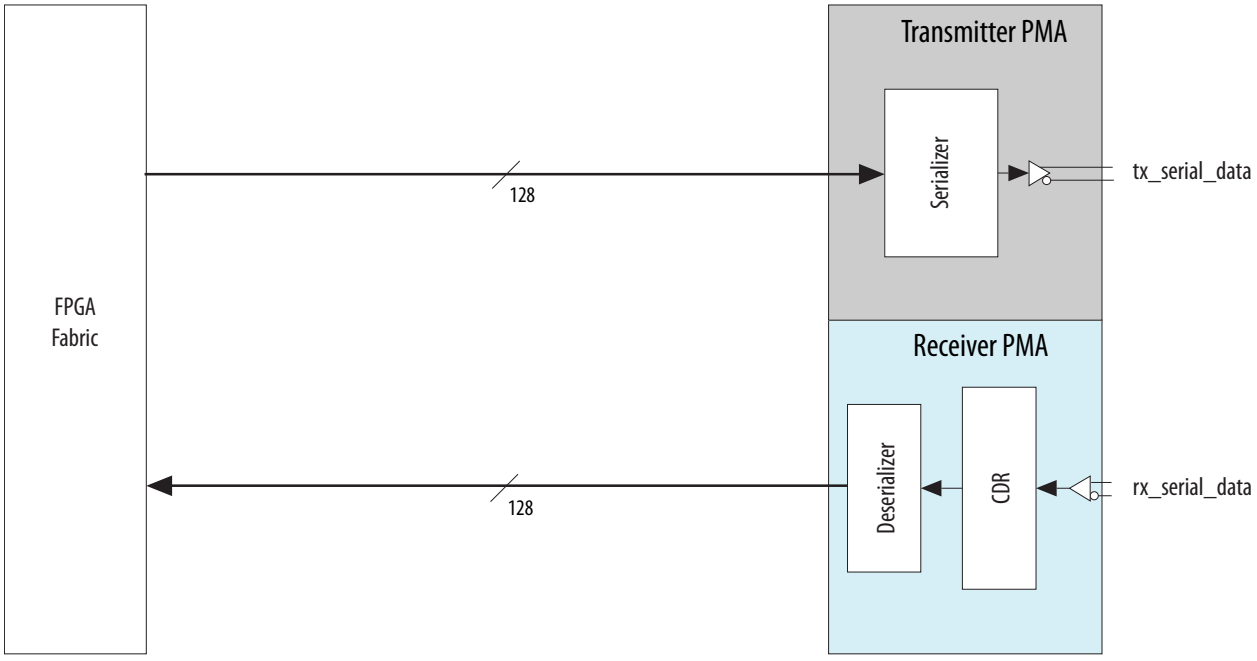
Figure 1-1: Single Full-Duplex GX Channel



Note:
1. The PMA and PCS widths are configurable.

Figure 1-2: Single Full-Duplex GT Channel

The GT channels do not have PCS.



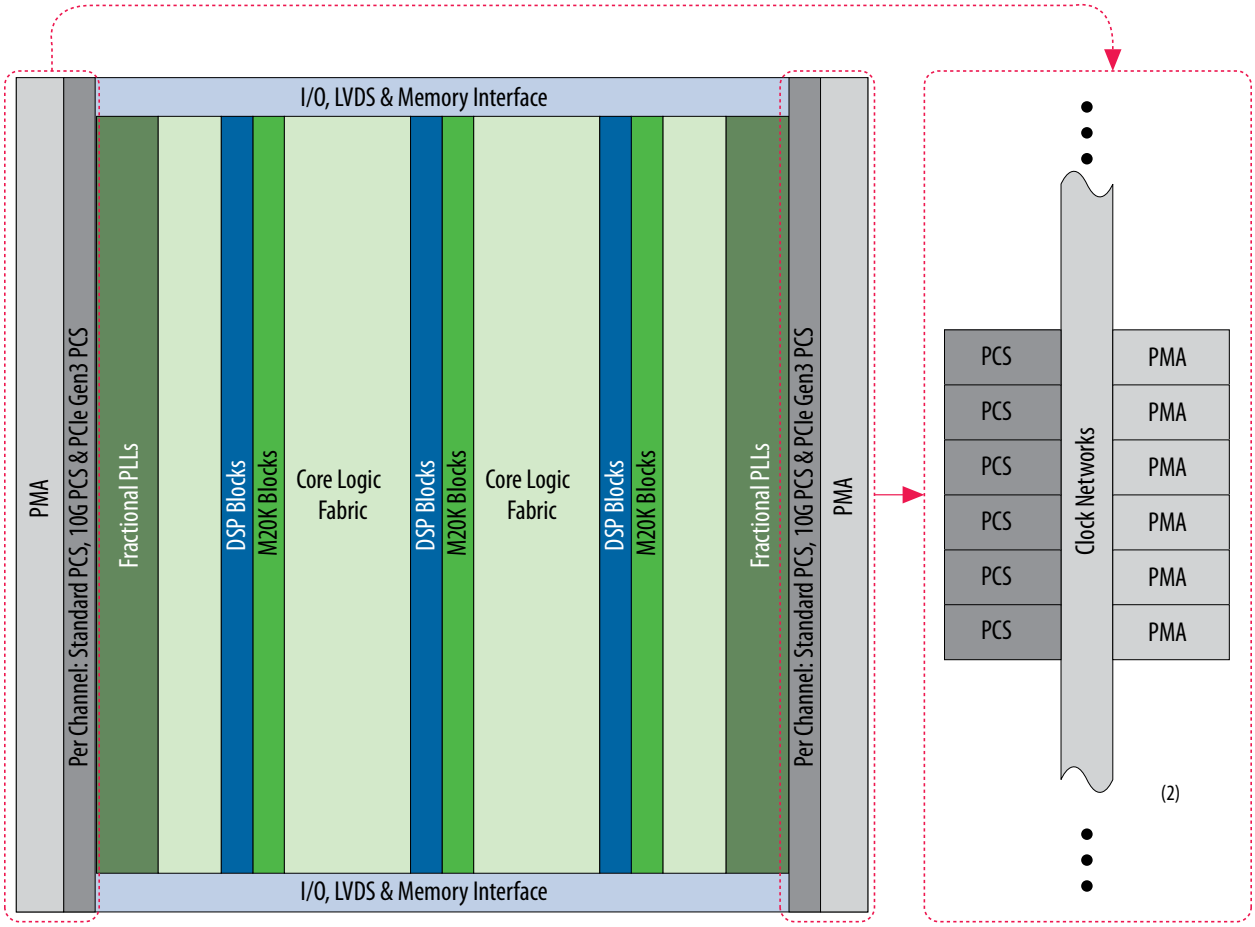
Related Information

- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the *Stratix V Device Handbook* chapters.
- **For details about using transceiver IPs, refer to the Altera Transceiver PHY IP Core User Guide.**
- **For transceiver performance specifications, refer to the Stratix V Device Datasheet.**
- **For information about features that will be supported in a future release of the Quartus® II software, refer to the Upcoming Stratix V Device Features document.**

Device Layout

Stratix V devices have columns of transceivers on the left and right sides of the devices. However, some Stratix V devices have columns of transceivers on the left side of the devices only.

Figure 1-3: Basic Layout of Transceivers



Notes:
1. This figure represents a given variant of a Stratix V device with transceivers. Other variants may have a dif erent floor plan than the one shown here.
2. You can use the unused transceiver channels as additional transceiver transmitter PLLs.

The location of the transceiver bank boundaries are important for clocking resources, bonding channels, and fitting. The transceivers are grouped in transceiver banks of three and six channels.

Within a transceiver bank the lowest numbered pin name is channel 0 of that bank and the highest numbered pin name of the bank is channel 5.

Stratix V GX/GT Channel and PCIe Hard IP Layout

Stratix V devices have many different channel and PCIe Hard IP variants.

Figure 1-4: General Transceiver Bank Locations for GX Devices

This figure shows the layout for a 66 channel device. Some devices have fewer channels.

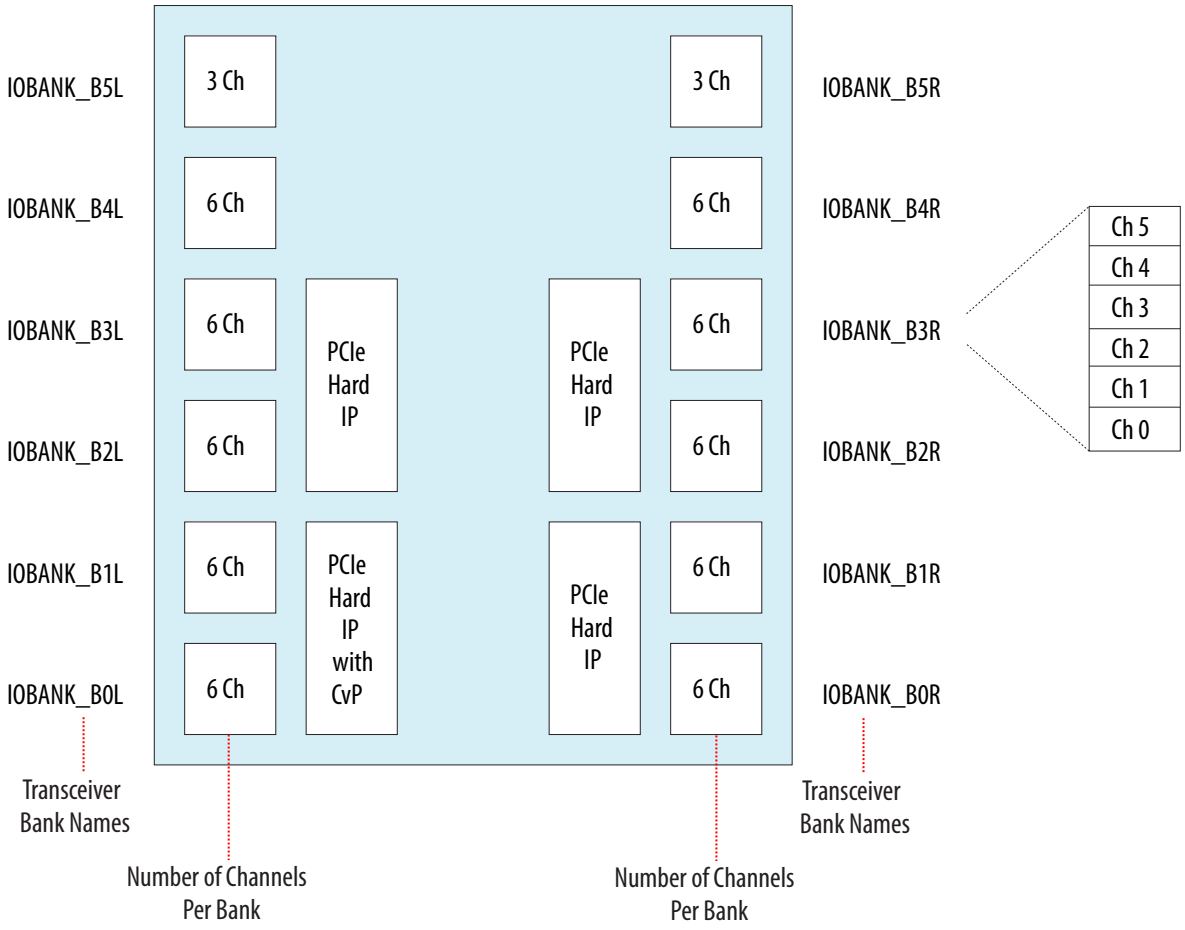
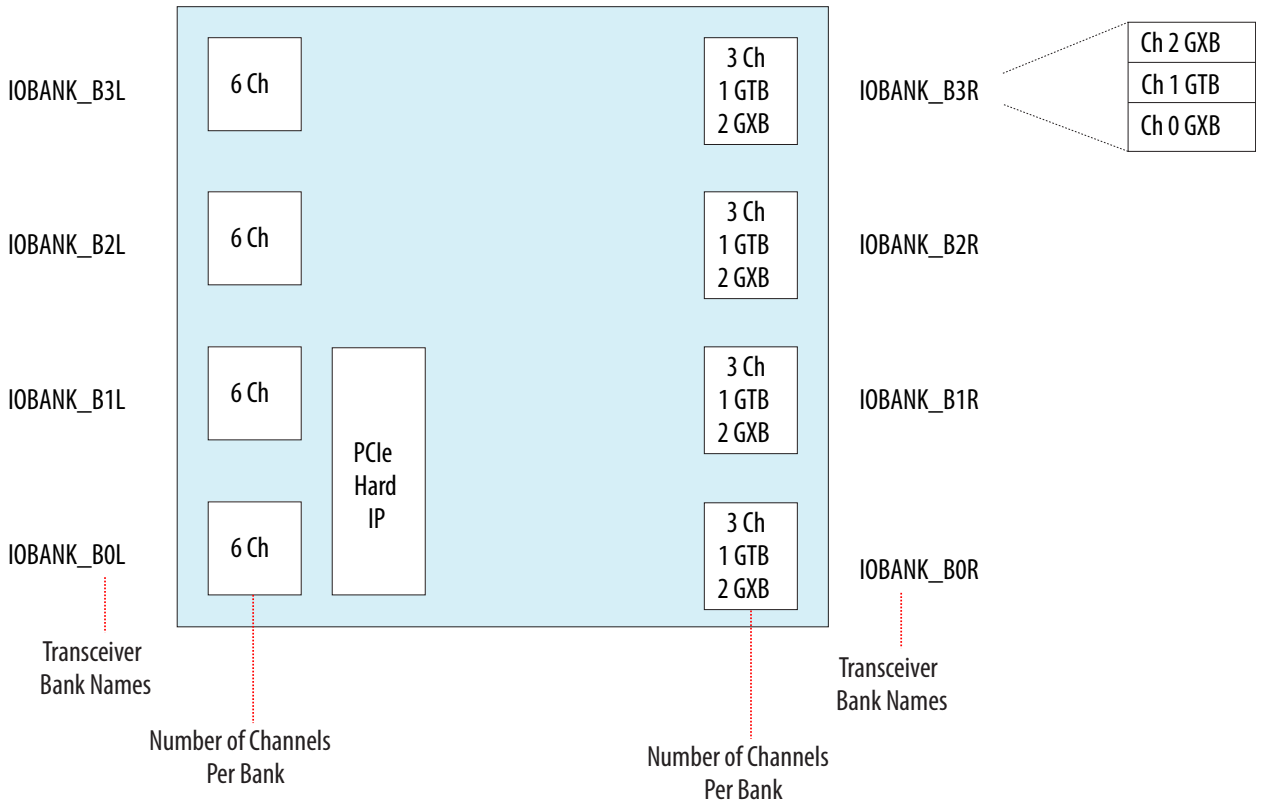


Figure 1-5: General Transceiver Bank Locations for GT Devices



Notes:
1. GT transceiver banks are made up of 1 GT channel and 2 GX channels. The GT channel is the middle channel in the bank.
2. GT devices only come with one PCIe Hard IP block located across GX banks L0 and L1.

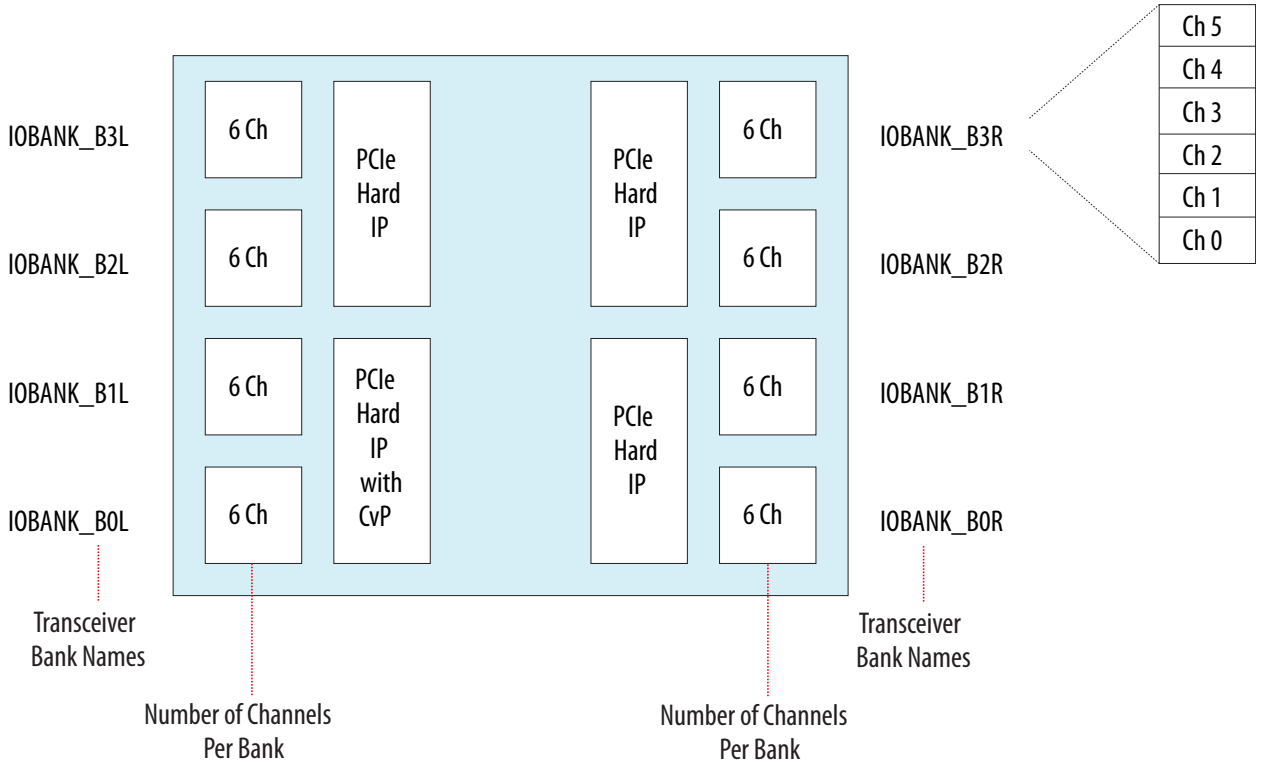
Related Information

- **For more information about device pin-outs, refer to Pin-Out Files for Altera Devices.**
- **For more information about device options, refer to the Stratix V Device Overview.**

Stratix V GS Channel and PCIe Hard IP Layout

Stratix V devices have many different channel and PCIe Hard IP variants.

Figure 1-6: General Transceiver Bank Locations for GS Devices



Channel Variants

GS/GX Devices

- 12-channel devices use banks B0L and B1L
- 24-channel devices use banks B0L, B1L, B0R and B1R
- 36-channel devices use banks B0L, B1L, B2L, B0R, B1R, and B2R
- 48-channel devices use banks B0L, B1L, B2L, B3L, B0R, B1R, B2R, and B3R
- 66-channel devices use all banks (GX only)

GT Devices

These 36-channel devices use banks B0L, B1L, B2L, B3L, B0R, B1R, B2R, and B3R.

GS/GT/GX Device Variants and Packages

Table 1-2: Stratix V GS, GT, and GX Device Variant Packages

Device Variant	PCIe Hard IP Blocks	# of Transceivers	Package	Side of Device with Transceivers
5SGSD3	1	12	EH29	Left
	1	24	HF35	Left and Right

Device Variant	PCIe Hard IP Blocks	# of Transceivers	Package	Side of Device with Transceivers
5SGSD4	1	12	EH29	Left
	1	24	HF35	Left and Right
	1	36	KF40	
5SGSD5	1	24	HF35	
	1	36	KF40	
5SGSD6	1 or 2	36	KF40	
	1 or 4	48	NF45	
5SGSD8	1 or 2	36	KF40	
	1 or 4	48	NF45	
5SGTC5	1	36	KF40	
5SGTC7				

Device Variant	PCIe Hard IP Blocks	# of Transceivers	Package	Side of Device with Transceivers
5SGXA3	1	12	EH29	Left
	1 or 2	24	HF35	Left and Right
		36	KF35	
			KF40	
5SGXA4	1 or 2	24	HF35	
		36	KF35	
			KF40	
5SGXA5	1 or 2	24	HF35	
		36	KF35	
			KF40	
	1 or 4	48	NF40	
			NF45	
5SGXA7	1 or 2	24	HF35	
		36	KF35	
			KF40	
	1 or 4	48	NF40	
			NF45	
5SGXA9	1 or 2	36	KH40	
	1 or 4	48	NF45	
5SGXAB	1 or 2	36	KF40	
	1 or 4	48	NF45	
5SGXB5	1 or 4	66	RF40	
			RF43	
5SGXB6	1 or 4	66	RF40	
			RF43	
5SGXB9	1 or 4	66	RH43	
5SGXBB	1 or 4	66	RH43	

PCIe Hard IP Variants

- 1 PCIe Hard IP variant has a Hard IP block located across GX banks L0 and L1
- 2 PCIe Hard IP variant has Hard IP blocks located across GX banks L0 and L1 and GX banks R0 and R1
- 4 PCIe Hard IP variant has Hard IP blocks located across GX banks L0 and L1, GX banks L2 and L3, GX banks R0 and R1 and GX banks R2 and R3

Related Information

- **For more information about PCIe Hard IP connections for GS and GX devices, refer to Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
- **For more information about PCIe Hard IP connections for GT devices, refer to Stratix V GT Device Family Pin Connection Guidelines**

PMA Architecture

The PMA receives and transmits off-chip high-speed serial data streams.

Figure 1-7: Receiver PMA Block Diagram

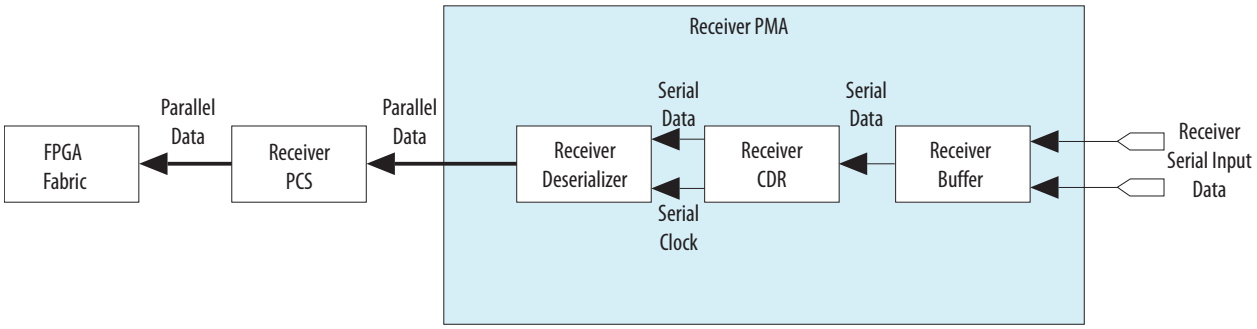
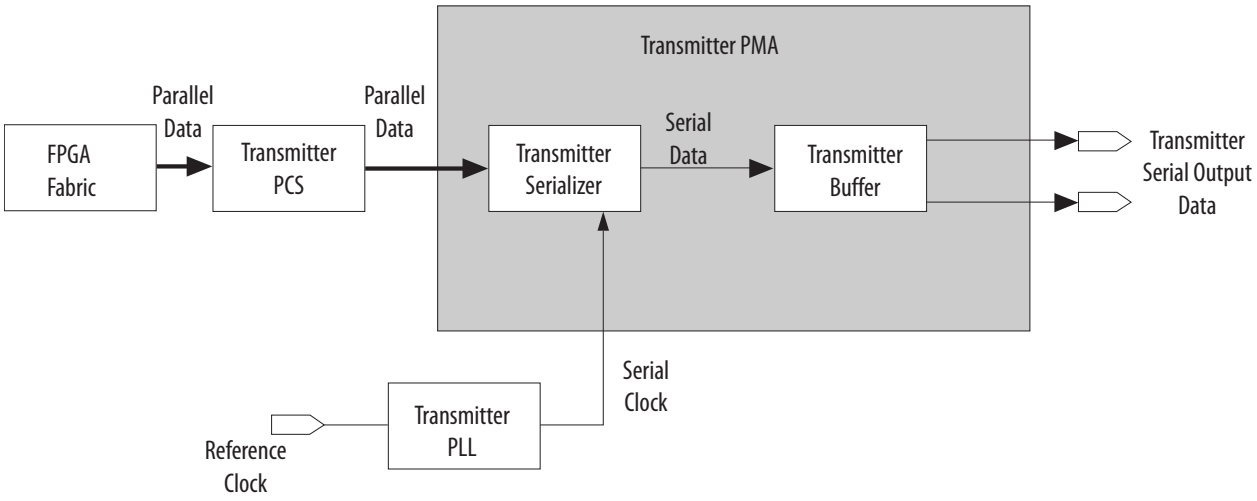


Figure 1-8: Transmitter PMA Block Diagram



Related Information

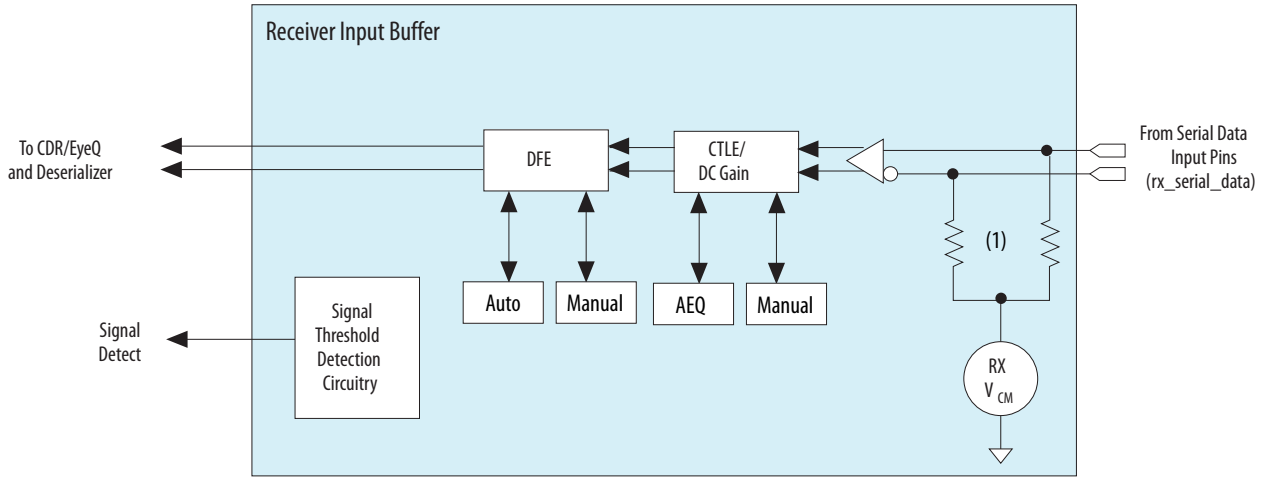
For additional details about clocking, refer to the Transceiver Clocking in Stratix V Devices chapter.

Receiver Buffer

The receiver input buffer receives serial data from the `rx_serial_data` port and feeds the serial data to the CDR and deserializer.

Figure 1-9: Receiver Input Buffer

Stratix V GT channels do not support AEQ and DFE.



Note:
1. For Stratix V GX channels, the available options are OFF, 85, 100, 120, and 150 Ω . Stratix V GT channels only support 100 Ω .

The receiver buffer supports the following features:

- Programmable Equalizer Bandwidth
- Programmable differential On-Chip Termination (OCT)
- Programmable V_{CM}
- AC and DC coupling
- Signal Threshold Detection Circuitry
- Continuous Time Linear Equalization (CTLE)
- DC Gain
- Decision Feedback Equalization (DFE)
- EyeQ

You can program these features using the assignment editor and a reconfiguration controller.

Related Information

Refer to the "Transceiver Performance Specifications" section of the Stratix V Device Datasheet for receiver buffer electrical specifications.

Receiver Equalizer Gain Bandwidth

Stratix V GX channels have two receiver equalizer gain bandwidth modes: half-bandwidth and full-bandwidth. When the data rate is ≤ 6.25 Gbps, Altera recommends selecting half bandwidth setting. When the data rate is between 6.25 Gbps and 14.1 Gbps, Altera recommends the full bandwidth setting. You can select the mode in the Assignment Editor of the Quartus II software (Receiver Equalizer Gain Bandwidth Select).

Stratix V GT channels support full-bandwidth mode only.

Related Information

For more information about the receiver equalizer gain bandwidth modes, refer to the Stratix V Device Datasheet

Programmable Differential On-Chip Termination (OCT)

The receiver buffer supports differential OCT resistances of 85, 100, 120, 150 Ω and OFF for GX channels. The receiver buffer of GT channels supports 100 Ω differential termination only.

Note: The receiver OCT resistors have calibration support to compensate for process, voltage, and temperature (PVT) variations. This does not apply to GT devices.

Programmable V_{CM}

The receiver buffer has on-chip biasing circuitry to establish the required V_{CM} at the receiver input.

The Quartus II software automatically chooses the best V_{CM} setting.

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

Related Information

For more information, refer to the Stratix V Device Datasheet.

Signal Threshold Detection Circuitry

You can enable the optional signal threshold detection circuitry. If enabled, this option senses whether the signal level present at the receiver input buffer is above the signal detect threshold voltage that you specified. This option is not available for all transceiver PHYs or for GT channels.

Related Information

- **For more information about transceiver configurations, refer to Transceiver Configurations in Stratix V Devices.**
- **For more information about the signal threshold detect signal, refer to the Altera Transceiver PHY IP Core User Guide.**

DC Gain

DC gain circuitry provides an equal boost to the incoming signal across the frequency spectrum. The receiver buffer supports DC gain settings of up to 8 dB.

Related Information

For more information about the DC gain settings for the receiver buffer, refer to the Stratix V Device Datasheet.

Continuous Time Linear Equalization (CTLE)

Each receiver buffer has five independently programmable equalization circuits that boost the high-frequency gain of the incoming signal, thereby compensating for the low-pass characteristics of the physical medium. For Stratix V GX channels, the equalization circuitry provides up to 16 dB of high-frequency boost. For Stratix V GT channels, the equalization circuitry provides up to 15 dB of high-frequency boost. The CTLE operates in two modes: manual mode and adaptive equalization (AEQ) mode.

You can dynamically switch between these modes.

Note: The CTLE cannot be bypassed.

Manual Mode

Manual mode allows you to manually adjust the continuous time linear equalization to improve signal integrity. You can statically set the equalizer settings in the IP or you can dynamically change the equalizer settings with the reconfiguration controller IP.

Adaptive Equalization Mode

AEQ mode eliminates the need for manual tuning by enabling the Stratix V device to automatically tune the receiver equalization settings based on the frequency content of the incoming signal and comparing that with internally generated reference signals. The AEQ block resides within the PMA of the receiver channel and is available on all GX channels.

Note: AEQ mode is not supported in GT channels.

There are two AEQ modes: one-time and powerdown:

- One-time mode—The AEQ finds a stable setting of the receiver equalizer and locks to that value. After the stable setting is locked, the equalizer values do not change.
- Powerdown mode—The AEQ of the specific channel is placed in standby mode and the CTLE uses the manually set value.

Related Information

- **For more information about the CTLE specifications, refer to the Stratix V Device Datasheet.**
- **For more information about enabling different options and using them to control the AEQ hardware, refer to the "Transceiver Reconfiguration Controller IP Core" chapter of the Altera Transceiver PHY IP Core User Guide.**

Decision Feedback Equalization

The decision feedback equalization (DFE) feature consists of a 5-tap equalizer, which boosts the high frequency components of a signal without noise amplification by compensating for inter-symbol interference (ISI). There are three DFE modes: manual, auto-adaptation, and triggered.

Note: The DFE is not supported in GT channels.

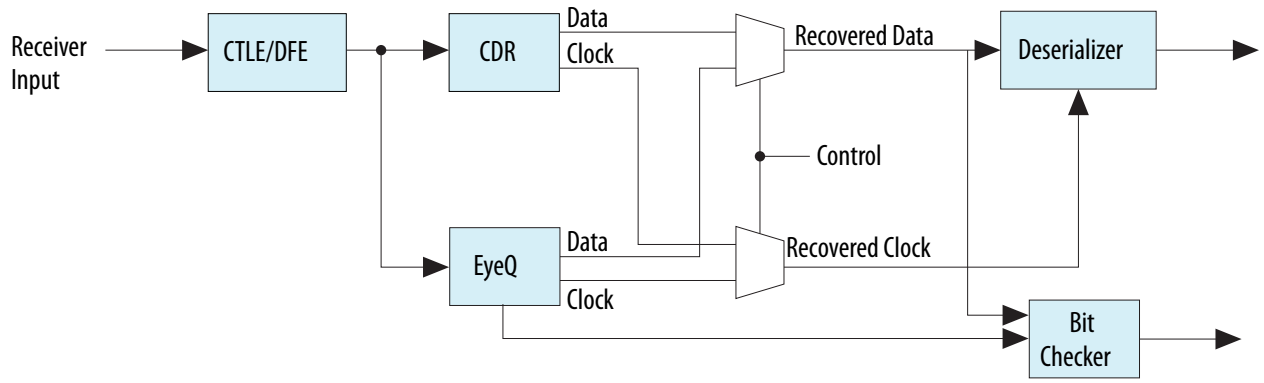
Related Information

For more information, refer to the "Transceiver Reconfiguration Controller IP Core" chapter of the Altera Transceiver PHY IP Core User Guide.

EyeQ

The EyeQ feature is a debug and diagnosis tool that helps you analyze the received data by measuring the horizontal and vertical eye margin. There are two multiplexers, each of which selects one path for the clock and data to feed to the deserializer, respectively.

Figure 1-10: Receiver and EyeQ Architecture



Serial Bit Checker

The serial bit checker is available when you enable EyeQ in the transceiver reconfiguration controller. It provides a means of estimating the number of errors that occur between the captured CDR signal and the sampled EyeQ signal. The advantage of enabling this block, is that it allows bit error register (BER) monitoring over live traffic.

This feature is also available for PCIe configurations.

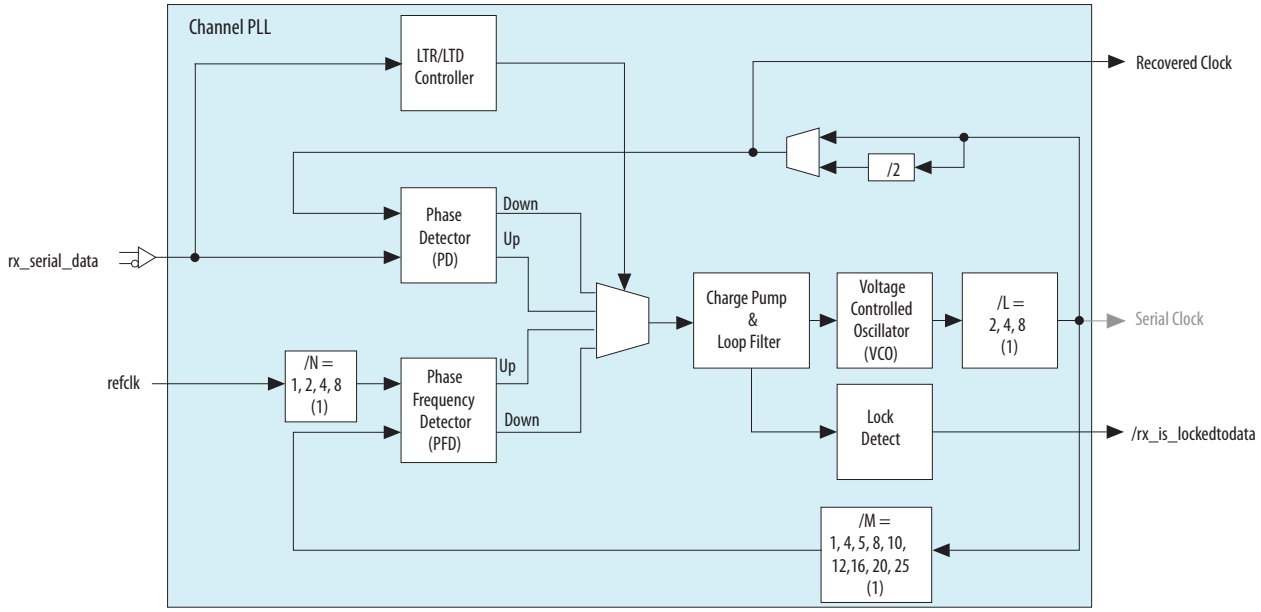
Related Information

For more information about implementing the serial bit checker, refer to the Transceiver Reconfiguration Controller EyeQ Registers section of the Altera Transceiver PHY IP Core User Guide

Receiver Clock Data Recovery Unit

The PMA of each channel includes a channel PLL that you can configure as a receiver clock data recovery (CDR) for the receiver, or a clock multiplier unit (CMU) PLL for the transmitter. For more information about using the channel PLL as a CMU PLL, refer to **Transmitter PLLs** on page 1-17.

Figure 1-11: Channel PLL Configured as CDR



Note:
1. Not all combinations of /N, /M, and/L values are valid. The Quartus II software automatically chooses the optimal values.

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. The `pma_rx_is_lockedtoref` status signal is asserted active high to indicate that the CDR has locked to the phase and frequency of the receiver input reference clock.

Note: The phase detector (PD) is inactive in LTR mode.

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 PD in the CDR tracks the incoming serial data at the receiver input. Depending on the phase difference between the incoming data and the CDR output clock, the PD controls the CDR charge pump that tunes the VCO.

Note: The PFD is inactive in LTD mode. The `rx_is_lockedtoref` status signal toggles randomly and is not significant in LTD mode.

After switching to LTD mode, the `rx_is_lockedtodata` status signal is asserted. 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 `rx_is_lockedtodata` signal toggles until the CDR sees valid data; therefore, you should hold receiver PCS logic in reset (`rx_digitalreset`) for a minimum of 4 µs after `rx_is_lockedtodata` remains continuously asserted.

CDR Lock Modes

You can configure the CDR in either automatic lock mode or manual lock mode. By default, the Quartus II software configures the CDR in automatic lock mode.

Automatic Lock Mode

In automatic lock mode, the CDR initially locks to the input reference clock (LTR mode). After the CDR locks to the input reference clock, the CDR locks to the incoming serial data (LTD mode) when the following conditions are met:

- The signal threshold detection circuitry indicates the presence of valid signal levels at the receiver input buffer.
- The CDR output clock is within the configured ppm frequency threshold setting with respect to the input reference clock (frequency locked).
- The CDR output clock and the input reference clock are phase matched within approximately 0.08 unit interval (UI) (phase locked).

If the CDR does not stay locked to data because of frequency drift or severe amplitude attenuation, the CDR switches back to LTR mode.

Manual Lock Mode

The PPM detector and phase relationship detector reaction times can be too long for some applications that require faster CDR lock time. You can manually control the CDR to reduce its lock time using two optional input ports (`rx_set_locktoref` and `rx_set_locktodata`).

Table 1-3: Relationship Between Optional Input Ports and the CDR 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

Related Information

For more information about manual lock mode, refer to the Transceiver Reset Control in Stratix V Devices chapter.

Receiver 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 or FPGA core.

The deserializer in the RX PMA also provides a clock slip feature. The word aligner block in the PCS can contribute up to one parallel clock cycle of latency uncertainty. You can use the clock slip/bit slip feature to control the word alignment instead, so that you can reduce the latency uncertainty, ensuring deterministic latency. The deterministic latency state machine in the word aligner (within the PCS) automatically controls the clock slip/bit slip operation. The de-serializer first performs clock slip/bit slip, after which the

parallel data is word-aligned in the RX PCS. These features are for protocols like CPRI that require deterministic latency through the PHY layer.

The GX channel deserializer supports 8- and 10-bit, 16- and 20-bit, 32- and 40-bit factors. 64-bit factors are also supported depending on the transceiver configuration. Unlike the GX channel deserializer, the GT channel deserializer does not support programmable data widths and is fixed at 128 bits.

Receiver PMA Bit-Slip

The deserializer has a bit slip feature to enable the high speed serial bit slipping to minimize uncertainty in serialization process per common public radio interface (CPRI) requirement. The bit slip feature is useful for other protocols as well. You can enable this feature through the Custom or Native or Deterministic Latency PHY IP. When you enable this feature, the period of a receiver side parallel clock could be extended by 1 unit interval (UI).

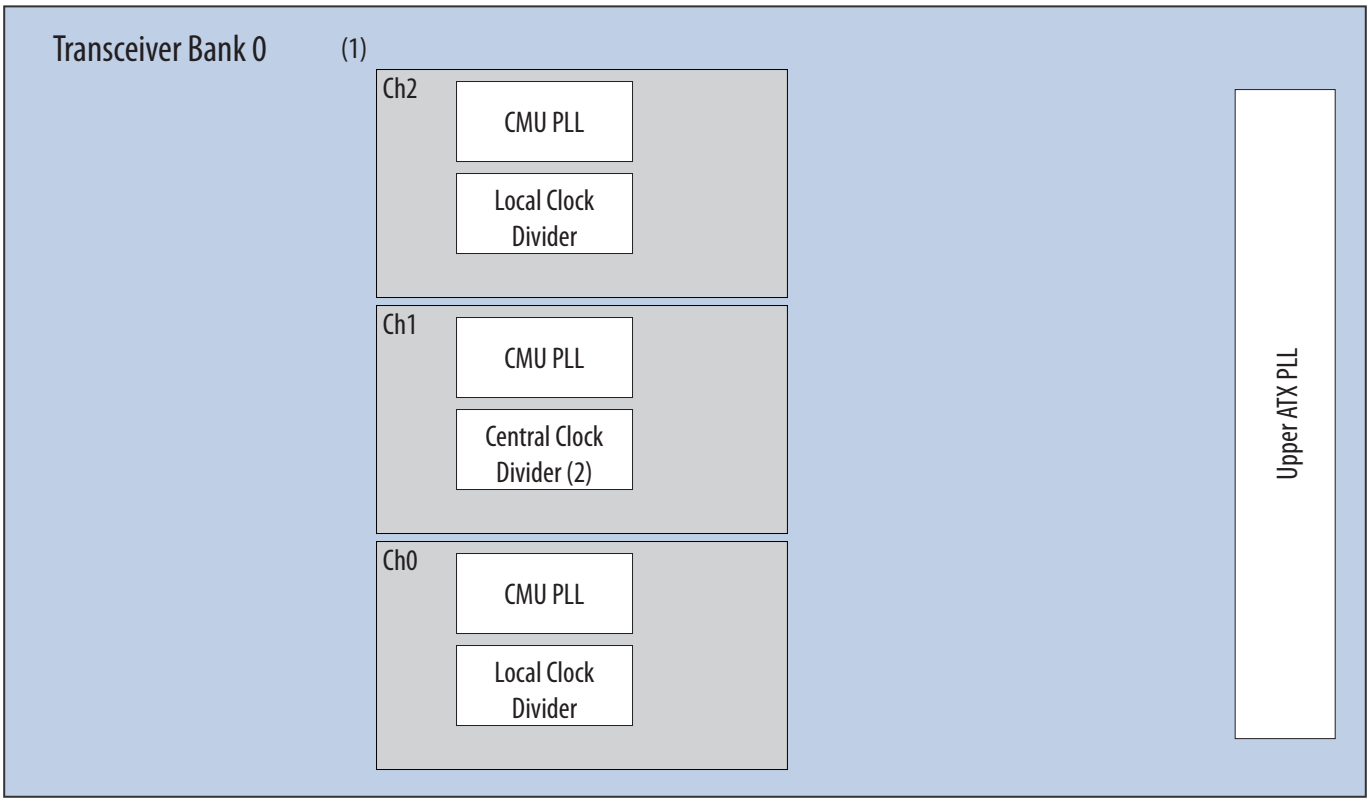
Note: When you enable the bit slip feature and do not use CPRI or deterministic latency state machine, the clock name will be different.

Related Information

For more information, refer to the Altera Transceiver PHY IP Core User Guide.

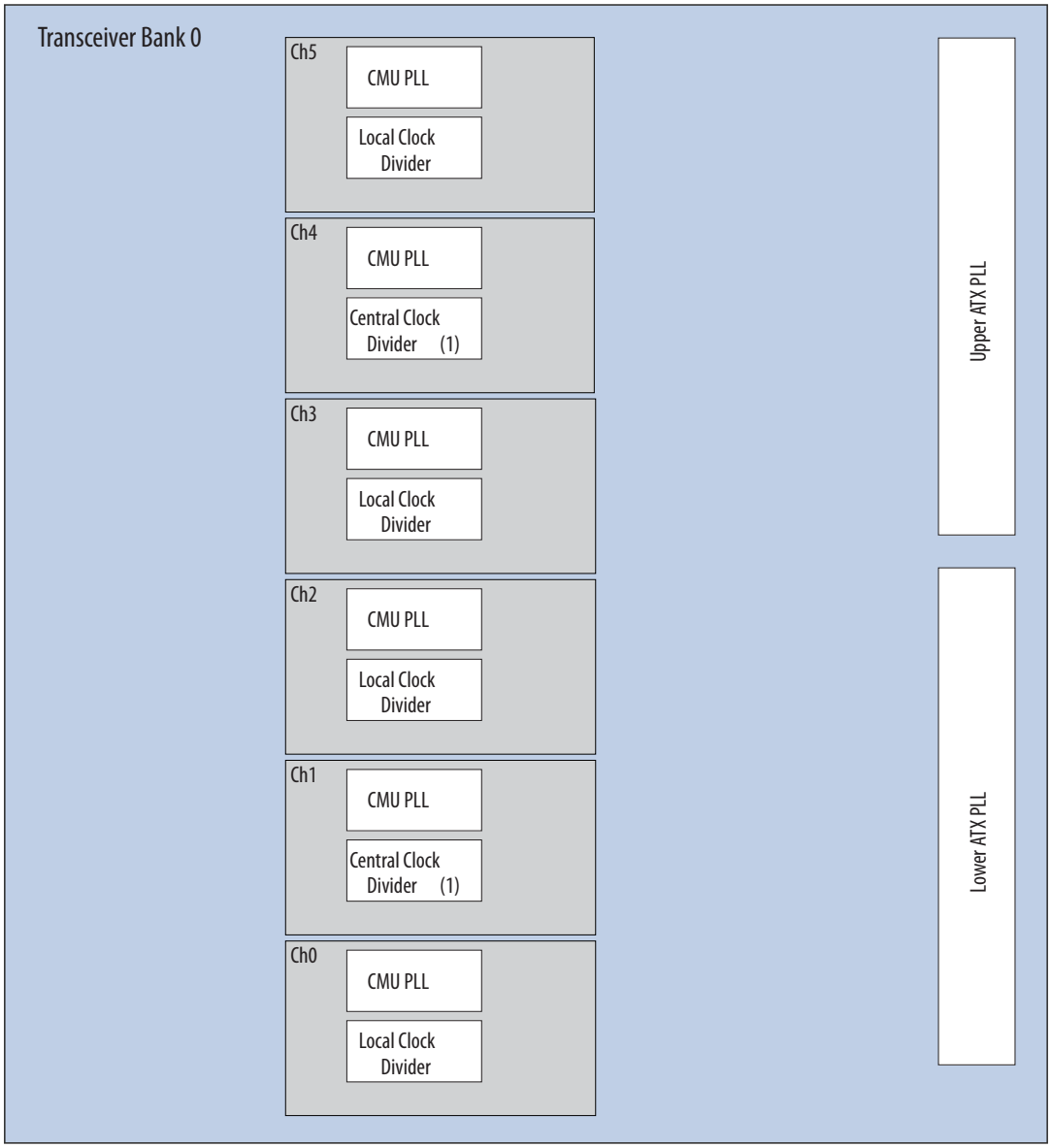
Transmitter PLLs

Figure 1-12: Transmitter PLL Locations in GX Devices (3 Channels)



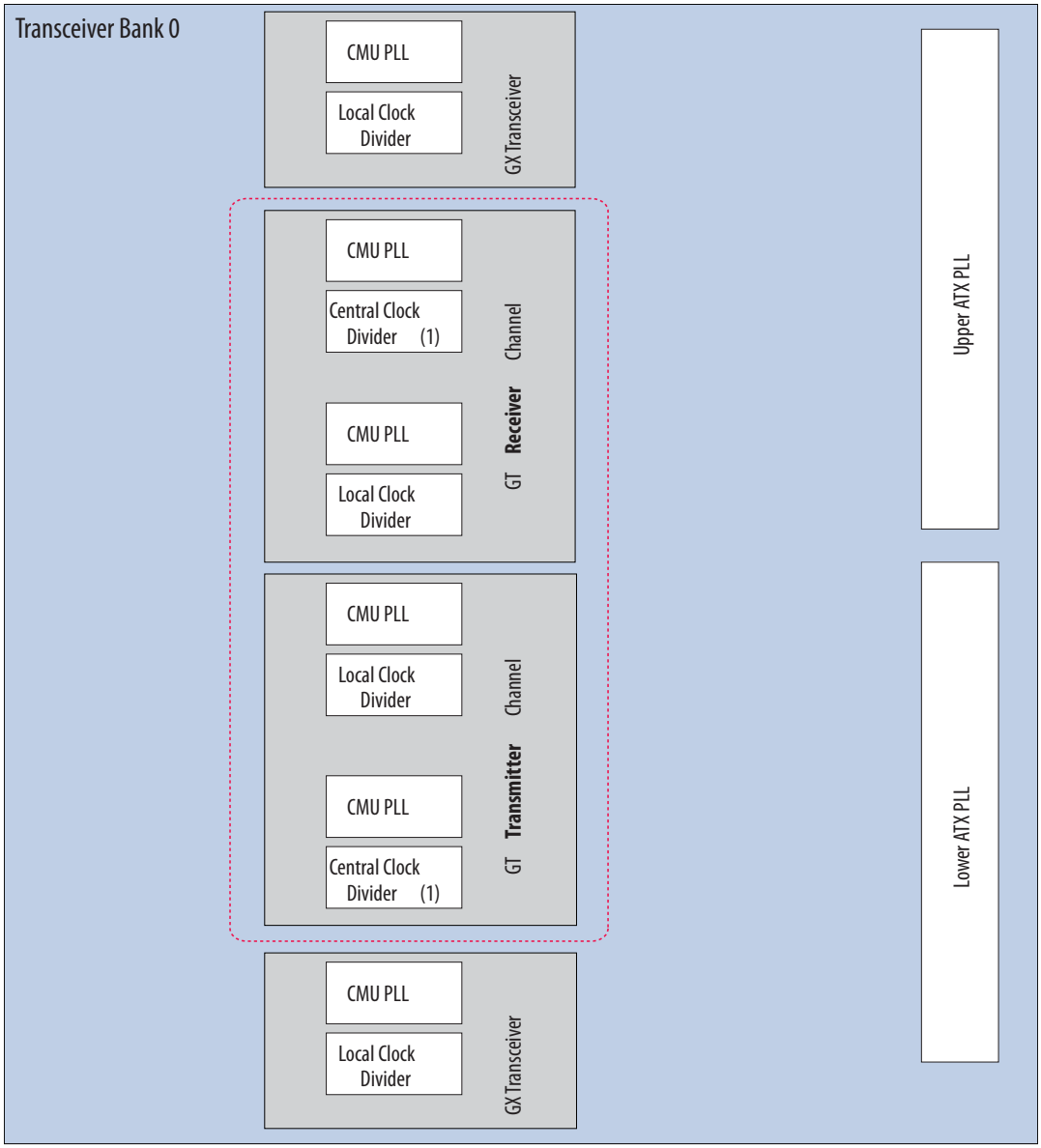
Notes:
1. Stratix V devices 5SGXB5, 5SGXB6, 5SGSB9 and 5SGSBB have one transceiver bank on each side with only three transceiver channels.
2. You can use the central clock divider as a local clock divider.

Figure 1-13: Transmitter PLL Locations in GX Devices (6 channels)



Note:
1. You can use the central clock divider as a local clock divider.

Figure 1-14: Transmitter PLL Locations in GT Devices



Notes:
1. You can use the central clock divider as a local clock divider.

Note: Transmitter PLLs within the upper-half or lower-half of a transceiver bank must be connected to the same Reconfiguration Controller.

Each transmitter channel has a clock divider called a local clock divider. The clock dividers generate the parallel and serial clock sources for the transmitter and optionally for the receiver PCS. The clock dividers on channels 1 and 4 are called central clock dividers because they can drive the x6 and xN clock lines. The central clock dividers can feed the clock lines used to bond channels.

Bonded Configurations

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. In bonded channel configurations the parallel clock is generated by a central clock divider for all channels, rather than using a local clock divider for each transmitter channel. Also, the transmitter phase compensation FIFO in all bonded channels shares 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.

Related Information

For more information about clocking or bonding, refer to the Transceiver Clocking in Stratix V Devices chapter.

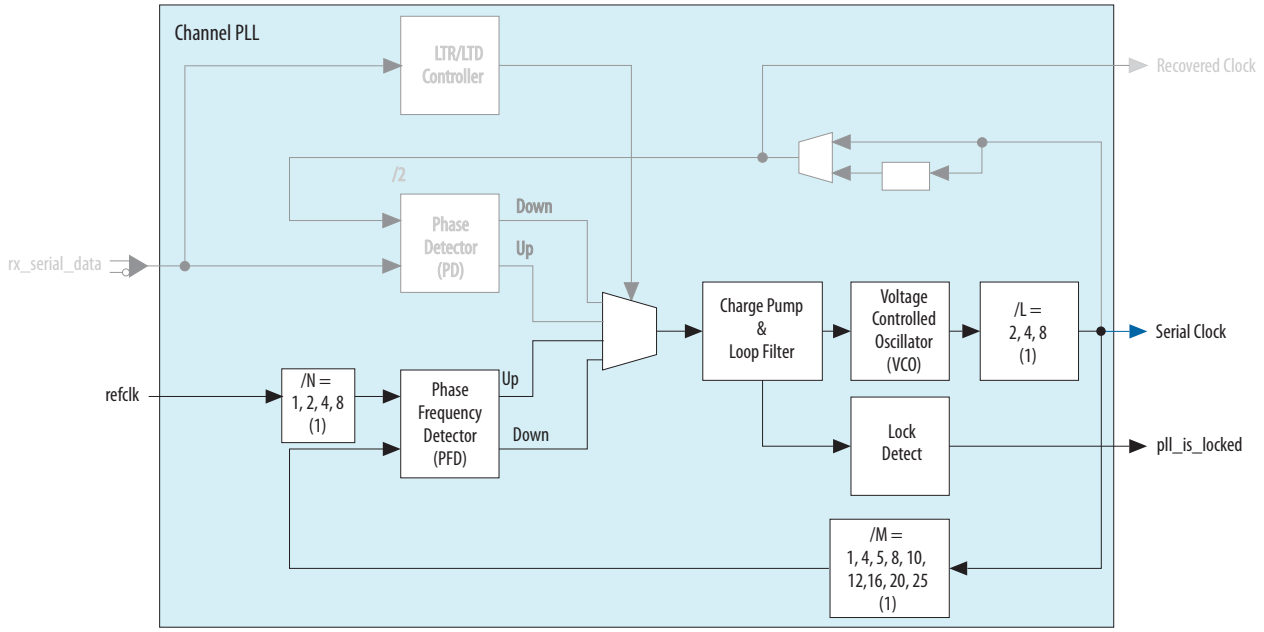
Channel PLL Used as a CMU PLL (Transmitter PLL)

The channel PLL available in the PMA can be used as a CMU PLL. The CMU PLLs in channels 1 and 4 in a transceiver block can also provide a clock to the other transceivers within the same block.

When you use the channel PLL as a CMU PLL, that particular channel cannot be used as a receiver; however, that channel can be used as a transmitter in conjunction with a different transmitter PLL. If all transmitters and receivers within the transceiver block are required, you must use an ATX PLL or a clock from another transceiver block.

For the best performance based on the data rate and input clock frequency, all settings for the CMU PLL and clock dividers are automatically chosen by the Quartus II software.

Figure 1-15: Channel PLL Configured as CMU PLL



Note:
1. Not all combinations of /N, /M, and /L values are valid. The Quartus II software automatically chooses the optimal values.

Note: Transmitter PLLs within the upper-half or lower-half of a transceiver bank must be connected to the same Reconfiguration Controller.

Related Information

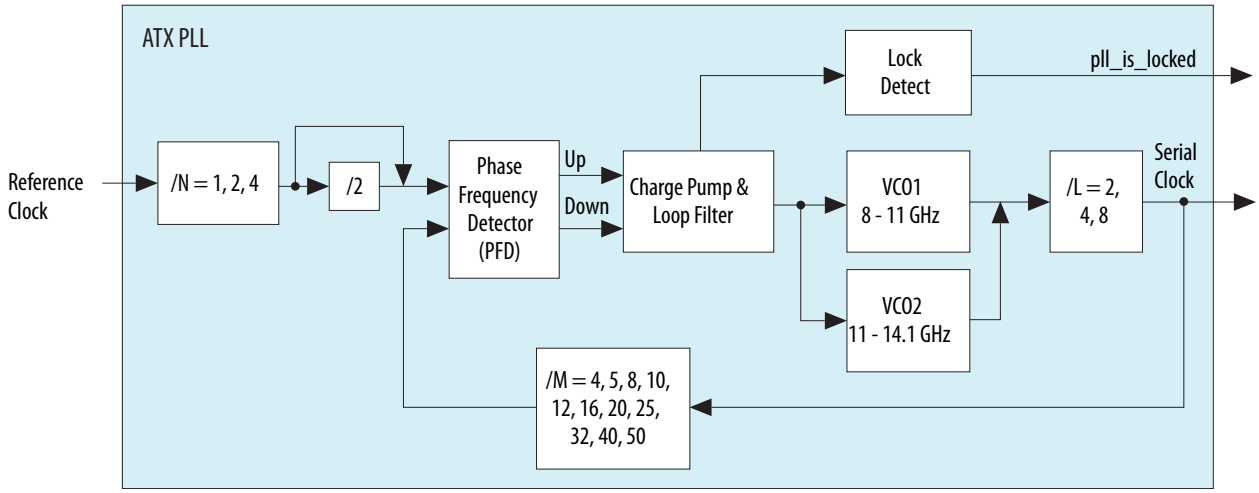
For CMU PLL specifications such as input clock frequency or supported output data ranges, refer to the Stratix V Device Datasheet.

Auxiliary Transmit (ATX) PLL Architecture

Most Stratix V GT, GX, and GS devices contain two ATX PLLs per transceiver bank that can generate the high-speed clocks for the transmitter channels; the 66-channel device is an exception with only one ATX PLL in the top bank. Compared with CMU PLLs, ATX PLLs have lower jitter and do not consume a transceiver channel; however, an ATX PLL's frequency range is more limited.

The serial clock from the ATX PLL is routed to the transmitter clock dividers and can be further divided down to half the data rate of the individual channels. For best performance you should use the reference clock input pins that reside in the same transceiver block as your channel. However, you can use any dedicated reference clock input pins along the same side of the device to clock the ATX PLL.

Figure 1-16: ATX PLL Architecture



Related Information

For ATX PLL specifications such as input clock frequency or supported output data ranges, refer to the Stratix V Device Datasheet.

Transmitter Serializer

The serializer converts the incoming low-speed parallel data from the transceiver PCS or FPGA fabric to high-speed serial data and sends the data to the transmitter buffer. The Stratix V GX channel serializer supports an 8- and 10-bit, 16- and 20-bit, 32- and 40-bit, and 64-bit serialization factor. By default, the serializer block sends out the LSB of the input data first. For Stratix V GT channels, the serializer only supports a 128-bit serialization factor.

PCIe Receiver Detect

For a PCIe configuration for Gen1, Gen2, and Gen3 data rates, the transmitter buffers have a built-in receiver detection circuit. This receiver detection circuit detects if 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

For more information about the PCIe protocol, refer to the Transceiver Configurations in Stratix V Devices chapter.

Transmitter Buffer

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

You can program these features using the assignment editor and a reconfiguration controller.

Note: The Stratix V GT transmitter buffer has only two taps for the pre-emphasis.

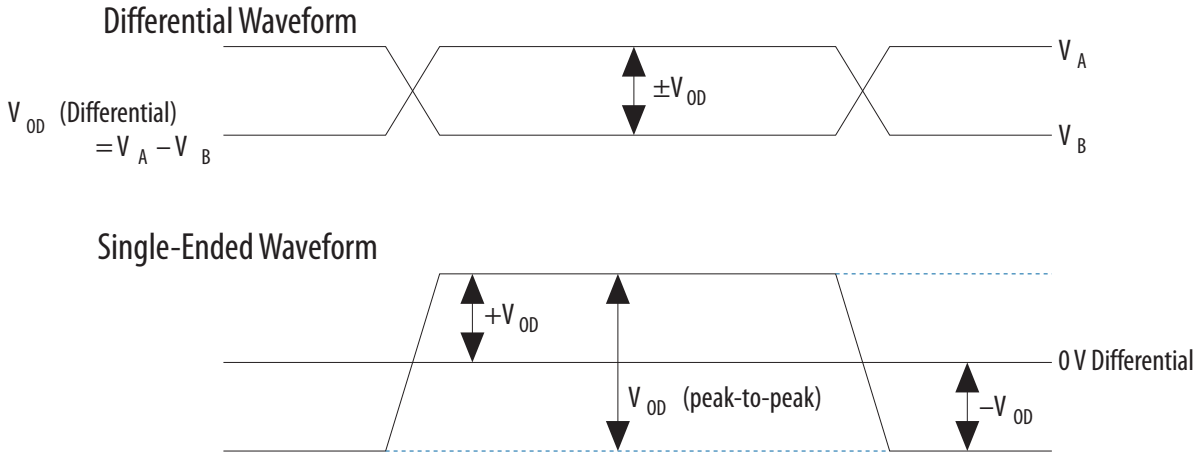
Transmitter Analog Settings

The transmitter analog setting capability improves signal integrity. These analog settings can be dynamically reconfigured without powering down the device using the PMA analog registers within the reconfiguration controller on GX devices only. Dynamic reconfiguration of the PMA settings can be done concurrently and independently per channel

Programmable Output Differential Voltage

You can program the differential output voltage to handle different trace lengths, various backplanes, and receiver requirements.

Figure 1-17: VOD (Differential) Signal Level



Programmable Pre-Emphasis

Pre-emphasis can maximize the data eye margin at the far-end receiver. The programmable pre-emphasis module in each transmit buffer boosts high frequencies in the transmit data signal, to compensate for attenuation in the transmission media.

Stratix V GX channels provide three pre-emphasis taps: pre-tap (16 settings), first post-tap (32 settings), and second post-tap (16 settings). The pre-tap sets the pre-emphasis on the data bit before the transition. The first post-tap and second post-tap set the pre-emphasis on the transition bit and the following bit, respectively. The pre-tap and second post-tap also provide inversion control, shown by negative values.

The Stratix V GT channels have two pre-emphasis taps: pre-tap and post-tap. Each tap has 32 settings.

Related Information

- **For more information about the dynamic reconfiguration feature, refer to the Dynamic Reconfiguration in Stratix V Devices chapter.**
- **For details about the transceiver reconfiguration controller IP, refer to the Altera Transceiver PHY IP Core User Guide.**

Programmable Transmitter On-Chip Termination (OCT)

Transmitter buffers include programmable on-chip differential termination of 85, 100, 120, 150 Ω or OFF, for GX channels The transmitter buffer of GT channels only supports 100 Ω differential termination, but allows for finer adjustment. 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.

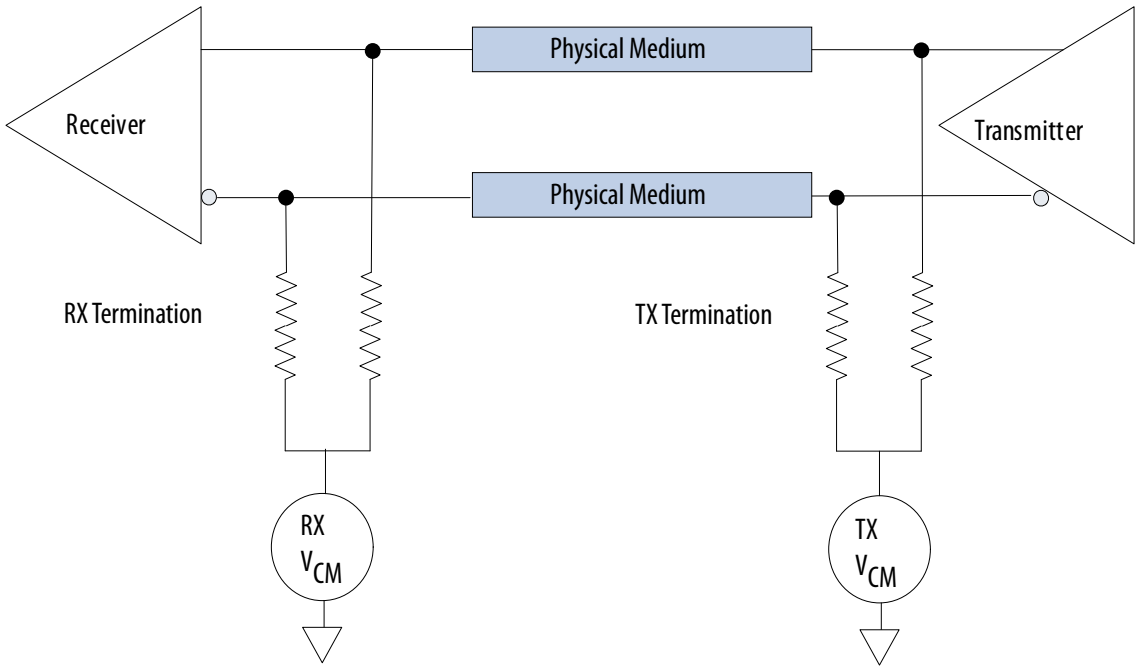
You can disable OCT and use external termination. If you select external termination, the transmitter common mode is tri-stated. Common mode is based on the external termination connection.

Link Coupling

A high-speed serial link can be AC-coupled or DC-coupled, depending on the serial protocol you are implementing.

In a DC-coupled link, the transmitter V_{CM} is seen unblocked at the receiver buffer. The link V_{CM} depends on the transmitter V_{CM} and the receiver V_{CM}. The on-chip or off-chip receiver termination and biasing circuitry must ensure compatibility between the transmitter and receiver V_{CM}.

Figure 1-18: DC-Coupled Link



You can DC-couple a GX/GS channel transmitter to a GX/GS channel receiver for the entire data rate range from 600 Mbps to 14.1 Gbps.

Note: The GT channels only support AC coupling.

Related Information

For DC-coupling other Altera families, and non-Altera devices, refer to the GX/GS transmitter and receiver common-mode requirements listed in theStratix V Device Datasheet.

Transceiver Calibration Blocks

The Stratix V transceiver calibration blocks include OCT calibration, offset cancellation in the receiver buffer and receiver CDR, and ATX PLL calibration.

OCT Calibration

Stratix V GX and GS devices contain calibration circuits that calibrate the OCT resistors and analog portions of the transceiver blocks to ensure that the functionality is independent of process, voltage, and temperature (PVT) variation. GT devices do not have calibration support.

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

Note: You must connect a separate 1.8 kΩ (maximum tolerance ±1%) external resistor on each RREF pin to ground. To ensure the calibration block operates properly, the RREF resistor connection on the board must be free from external noise.

Related Information

For more information about connections, refer to the Pin Connection Guidelines.

Offset Cancellation in the Receiver Buffer and Receiver CDR

Process variation can lead to a V_{CM} offset between the p and n signals within the differential buffers. Stratix V devices have an automatic calibration in their receiver buffers to remove this V_{CM} offset.

You must use the reconfiguration controller IP for offset cancellation to take place. Calibration does not occur during transceiver reset, only during device configuration. Any signals that may appear on the receiver pin do not affect calibration because the receiver buffers are disconnected during calibration.

Note: A maximum of one reconfiguration controller is allowed per transceiver bank upper-half or lower-half triplet.

ATX PLL Calibration

ATX PLL calibration optimizes the ATX PLL VCO settings for the desired output frequency. The reconfiguration controller IP must be instantiated for this calibration to run. The calibration occurs once automatically after device initialization. The calibration can be restarted via register accesses to the Reconfiguration controller if desired.

Both the ATX PLL and reconfiguration reference clocks must be valid and stable upon device power-up to ensure a successful power-up ATX PLL calibration and for its subsequent calibrations. The reference clock must be stable and the ATX PLL cannot be in reset (`pll_powerdown`) when ATX PLL calibration is performed.

Related Information

- **For more information, refer to the Dynamic Reconfiguration in Stratix V Devices chapter.**
- **For more information about ATX PLL calibration, refer to the Auxiliary Transmit (ATX) PLL Calibration section of the Altera Transceiver PHY IP Core User Guide**

Calibration Block Boundary

There is one calibration block in each quadrant of the device.

The calibration block also uses the reconfiguration controller clock (`mgmt_clk_clk`). This puts a restriction on the number of different reconfiguration clock sources that can be used in the design. All the transceiver channels controlled by a single calibration block must be connected to the same reconfiguration clock source.

Note: You can connect multiple reconfiguration controllers to the same clock source.

Table 1-4: Transceiver Calibration Block Boundary for Stratix V Devices

Stratix V Device	Package	Total Number of Transceiver channels in device	Total Number of Transceiver Channels per Side	Number of Contiguous Transceiver Channels Controlled by the Top Calibration Block (counting from top to bottom)	Number of Contiguous Transceiver Channels Controlled by the Bottom Calibration Block (counting from bottom to top)
5SGTC5	KF40	36	24 (Left) / 12 (Right)	12	12
5SGTC7	KF40	36	24 (Left) / 12 (Right)	12	12
5SGXA3	EH29	12	12 (Left only)	6	6
	HF35	24	12	6	6
	KF35	36	18	9	9
	KF40/ KH40	36	18	9	9
5SGXA4	HF35	24	12	6	6
	KF35	36	18	9	9
	KF40/ KH40	36	18	9	9
5SGXA5	HF35	24	12	6	6
	KF35	36	18	9	9
	KF40/ KH40	36	18	9	9
	NF40	48	24	12	12
5SGXA7	HF35	24	12	6	6
	KF35	36	18	9	9
	KF40/ KH40	36	18	9	9
	NF40	48	24	12	12
5SGXA9	KF40/ KH40	36	18	9	9
	NF45	48	24	12	12
5SGXAB	KF40/ KH40	36	18	9	9
	NF45	48	24	12	12

Stratix V Device	Package	Total Number of Transceiver channels in device	Total Number of Transceiver Channels per Side	Number of Contiguous Transceiver Channels Controlled by the Top Calibration Block (counting from top to bottom)	Number of Contiguous Transceiver Channels Controlled by the Bottom Calibration Block (counting from bottom to top)
5SGXB5	RF40	66	33	15	18
	RF43	66	33	15	18
5SGXB6	RF40	66	33	15	18
	RF43	66	33	15	18
5SGXB9	RH43	66	33	15	18
5SGXBB	RH43	66	33	15	18
5SGSD3	EH29	12	12 (Left only)	6	6
	HF35	24	12	6	6
5SGSD4	EH29	12	12 (Left only)	6	6
	HF35	24	12	6	6
	KF40	36	18	9	9
5SGSD5	HF35	24	12	6	6
	KF40	36	18	9	9
5SGSD6	KF40	36	18	9	9
	NF45	48	24	12	12
5SGSD8	KF40	36	18	9	9
	NF45	48	24	12	12

Related Information

Refer to the Transceiver Reconfiguration Controller PMA Analog Control Registers section of the Altera Transceiver PHY IP Core User Guide

PMA Reconfiguration

Modifying programmable values within transceiver buffers can be performed by a single reconfiguration controller for the entire FPGA, or multiple reconfiguration controllers if desired. Within each transceiver bank a maximum of two reconfiguration controllers is allowed; one for the three channels (triplet) in the upper-half of a bank, and one for the lower-half. This is due to a single Avalon-Memory Mapped (AVMM) slave interface per triplet Therefore, many triplets can be connected to a single reconfiguration controller, but only one reconfiguration controller can be connected to the three transceivers within any triplet.

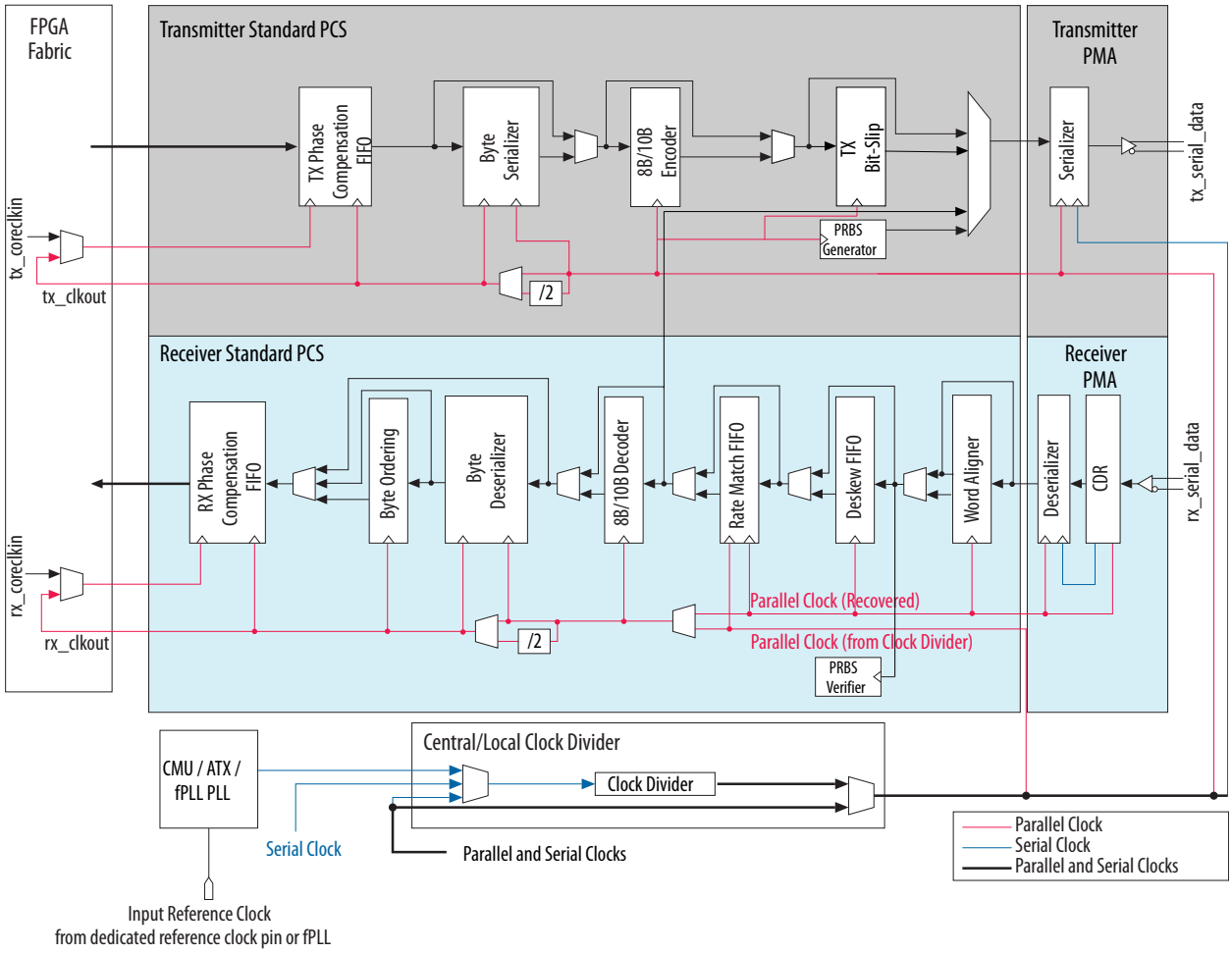
Related Information

Refer to the "Transceiver Reconfiguration Controller IP Core" chapter of the Altera Transceiver PHY IP Core User Guide.

Standard PCS Architecture

The transceiver Standard PCS circuit blocks support data rates up to 12.2 Gbps depending on the transceiver speed grade.

Figure 1-19: Standard PCS Datapath in GX Channels



Note: The PCS can be completely bypassed using the PMA direct mode. This is an option for GX channels. The GT channels do not have a PCS, and always operate in PMA direct mode.

Some transceiver channels interface to the PCIe hard IP block, the PIPE interface for soft IP implementations of PCIe, or directly to the FPGA fabric (FPGA fabric-transceiver interface). The transceiver channel interfaces to the PCIe hard IP block if you use the hard IP block to implement the PCIe PHY MAC, data link layer, and transaction layer. Otherwise, the transceiver channel interfaces directly to the FPGA fabric.

The PCIe hard IP-transceiver interface is outside the scope of this chapter. This chapter describes the FPGA fabric-transceiver interface only.

You can divide the standard transceiver channel datapath into two configurations based on the FPGA fabric-transceiver interface width (channel width) and the transceiver channel PMA-PCS width (serialization factor).

Table 1-5: Transceiver Interface Widths

Name	8- and 10-Bit PMA-PCS Widths	16- and 20-Bit PMA-PCS Widths
FPGA fabric-transceiver interface widths	8 and 10 bit 16 and 20 bit	16 and 20 bit 32 and 40 bit
Supported configurations	PCIe Gen1 and Gen2 XAUI Custom configuration (Custom or Native or Low Latency PHY IPs)	Custom configuration (Custom or Native or Low Latency PHY IPs)
Data rate range in a custom configuration	0.6 to 3.75 Gbps	1.0 to 12.2 Gbps

The standard PCS can be configured for various protocols by selecting different PCS blocks in the receiver and transmitter datapath.

Related Information

- **For more information about the PMA direct mode, refer to the "Stratix V GX Native PHY IP" section of the Transceiver Configurations in Stratix V Devices chapter.**
- **For details about the PMA direct mode, refer also to the "Stratix V Transceiver Native PHY IP Core" section of the Altera Transceiver PHY IP Core User Guide.**
- **For details about selecting different PCS blocks in the receiver and transmitter datapath and about the data rates supported in other configurations, refer to the Transceiver Configurations in Stratix V Devices chapter.**
- **For details about implementing the different PCS functional blocks, refer to the Altera Transceiver PHY IP Core User Guide.**

Receiver Standard PCS Datapath

The functional blocks in the receiver datapath are described in order from the word aligner to the receiver phase compensation FIFO buffer at the FPGA fabric-transceiver interface.

The receiver datapath is flexible and allows multiple modes, depending on the selected configuration.

Note: The Standard PCS is not supported in the GT channels.

Word Aligner

Because the data is serialized before transmission and then deserialized at the receiver, the data loses the word boundary of the upstream transmitter after deserialization. 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.

Serial protocols such as PCIe specify a standard word alignment pattern. For proprietary protocols, the transceiver architecture allows you to select a custom word alignment pattern specific to your implementation.

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

- Synchronization state machine in configurations such as PCIe
- Programmable run length violation detection in all configurations
- Receiver polarity inversion in all configurations except PCIe
- Receiver bit reversal in custom configurations
- Receiver byte reversal in custom 16- and 20-bit width configurations

Depending on the configuration, the word aligner operates in one of the following modes:

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

Table 1-6: Word Aligner Options

Configuration	PMA-PCS Interface Width (Bits)	Word Alignment Mode	Word Alignment Pattern Length	Word Alignment Behavior
Custom 8- and 10-Bit Width	8	Manual Alignment	16 bits	User-controlled signal starts the alignment process. Alignment happens once unless the signal is reasserted.
		Bit-Slip	N/A	User-controlled signal shifts data one bit at a time.
	10	Manual Alignment	7 and 10 bits	User-controlled signal starts the alignment process. Alignment happens once unless the signal is reasserted.
		Bit-Slip	N/A	User-controlled signal shifts data one bit at a time.
		Automatic Synchronized State Machine	7 and 10 bits	Data is required to be 8B/10B encoded. Aligns to the selected word aligner pattern.
		Deterministic Latency State Machine	10 bits	Data is required to be 8B/10B encoded. Aligns to the selected word aligner pattern.
Custom 16- and 20-Bit Width	16	Manual Alignment	8, 16, and 32 bits	User-controlled signal starts the alignment process. Alignment happens once unless the signal is reasserted.
		Bit-Slip	N/A	User-controlled signal shifts data one bit at a time.
	20	Manual Alignment	7, 10, and 20 bits	User-controlled signal starts the alignment process. Alignment happens once unless the signal is reasserted.
		Bit-Slip	N/A	User-controlled signal shifts data one bit at a time.
		Automatic Synchronized State Machine	7, 10, and 20 bits	Data is required to be 8B/10B encoded. Aligns to the selected word aligner pattern.
		Deterministic Latency State Machine	10 and 20 bits	Data is required to be 8B/10B encoded. Aligns to the selected word aligner pattern.

Configuration	PMA-PCS Interface Width (Bits)	Word Alignment Mode	Word Alignment Pattern Length	Word Alignment Behavior
PCIe	10	Automatic Synchronized State Machine	10 bits	Automatically selected word aligner pattern length and pattern.

Example of Manual Alignment Mode Word Aligner with a 10-Bit PMA-PCS Interface Configuration

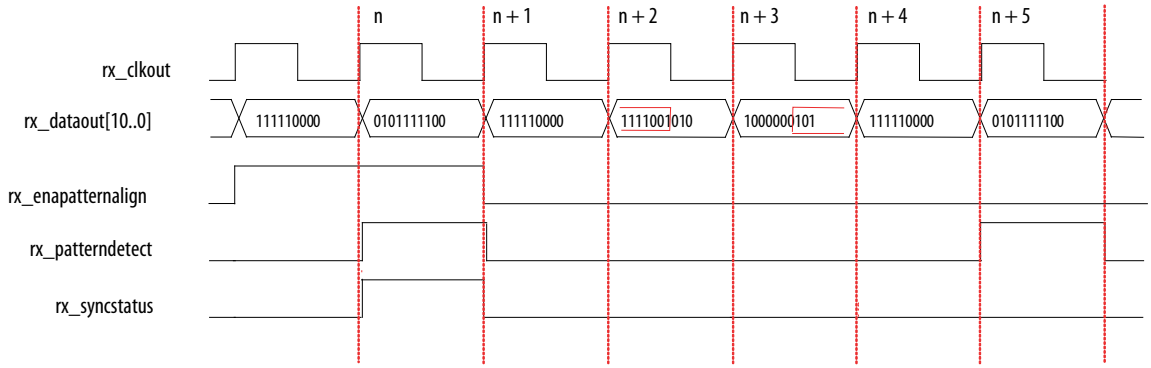
In basic single-width mode with a 10-bit PMA-PCS interface, you can configure the word aligner in manual alignment mode by selecting the **Use manual word alignment mode** option in the IP Catalog.

In manual alignment mode, the word alignment operation is manually controlled with the `rx_std_wa_patternalign` input signal or the `rx_enapatternalign` register. The word aligner operation is level-sensitive to `rx_enapatternalign`. If `rx_enapatternalign` is held high, the word aligner looks for the programmed 7-bit or 10-bit word alignment pattern in the received data stream. It updates the word boundary if it finds the word alignment pattern in a new word boundary. If `rx_enapatternalign` is de-asserted low, the word aligner maintains the current word boundary even when it sees the word alignment pattern in a new word boundary.

Two status signals, `rx_syncstatus` and `rx_patterndetect`, with the same latency as the datapath, are forwarded to the FPGA fabric to indicate the word aligner status. After receiving the first word alignment pattern after the `rx_enapatternalign` signal is asserted high, both the`rx_syncstatus` and `rx_patterndetect` signals are driven high for one parallel clock cycle. Any word alignment pattern received thereafter in the same word boundary causes only the `rx_patterndetect` signal to go high for one clock cycle. Any word alignment pattern received thereafter in a different word boundary causes the word aligner to re-align to the new word boundary only if the `rx_enapatternalign` signal is held high. The word aligner asserts the `rx_syncstatus` signal for one parallel clock cycle whenever it re-aligns to the new word boundary.

In this example, a /K28.5/ (10'b0101111100) is specified as the word alignment pattern. The word aligner aligns to the /K28.5/ alignment pattern in cycle n because the `rx_enapatternalign` signal is asserted high. The `rx_syncstatus` signal goes high for one clock cycle, indicating alignment to a new word boundary. The `rx_patterndetect` signal also goes high for one clock cycle to indicate initial word alignment. At time n + 1, the `rx_enapatternalign` signal is de-asserted to instruct the word aligner to lock the current word boundary. The alignment pattern is detected again in a new word boundary across cycles n + 2 and n + 3. The word aligner does not align to this new word boundary because the `rx_enapatternalign` signal is held low. The /K28.5/ word alignment pattern is detected again in the current word boundary during cycle n + 5, causing the `rx_patterndetect` signal to go high for one parallel clock cycle.

Figure 1-20: Word Aligner with 10-Bit PMA-PCS Manual Alignment Mode



Note: If the word alignment pattern is known to be unique and does not appear between word boundaries, you can constantly hold `rx_enapatternalign` high because there is no possibility of false word alignment. If there is a possibility of the word alignment pattern occurring across word boundaries, you must control `rx_enapatternalign` to lock the word boundary after the desired word alignment is achieved to avoid re-alignment to an incorrect word boundary.

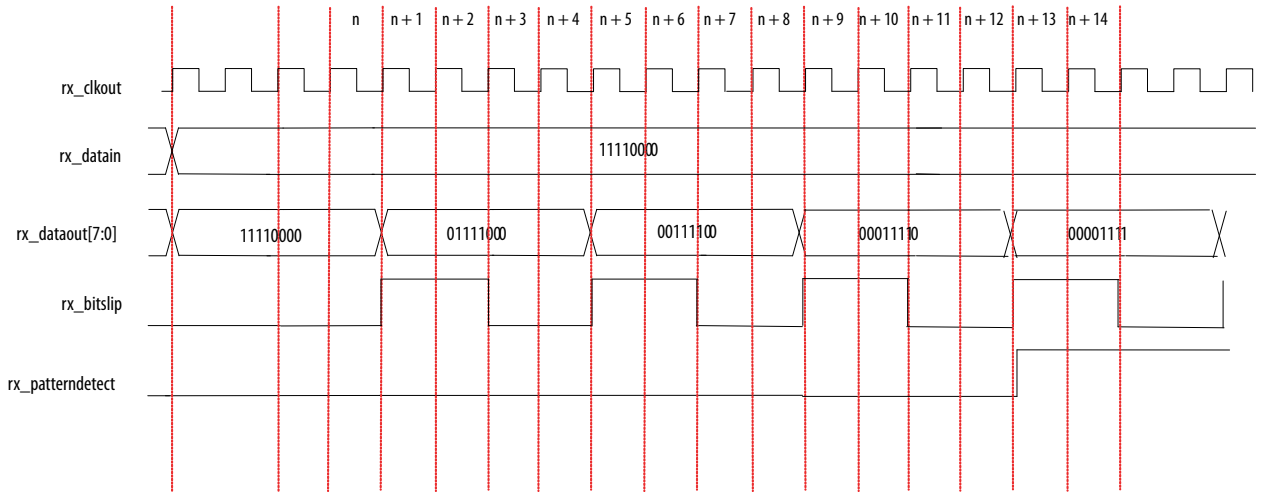
Example of Bit-Slip Mode Word Aligner with an 8-Bit PMA-PCS Interface Configuration

In a custom width configuration with an 8-bit PMA-PCS interface width, you can configure the word aligner in bit-slip mode. In bit-slip mode, the word aligner operation 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_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.

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 the `rx_bitslip` signal to achieve word alignment.

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

Figure 1-21: Word Aligner Configured in Bit-Slip Mode



Example of Automatic Synchronization State Machine Mode Word Aligner 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 the protocol must receive to fall out of synchronization.

In PCIe configurations, the word aligner is in automatic synchronization state machine mode. The word aligner 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 1-7: Synchronization State Machine Modes for a PCI-Express (PIPE) 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 `rx_digitalreset` 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 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 1-8: Word Alignment 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 circuit resides in the word aligner block and detects consecutive 1s or 0s in the data. 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 1-9: Detection Capabilities of the Run Length Violation Circuit

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

Receiver Polarity Inversion

The positive and negative signals of a serial differential link may be erroneously swapped during board layout. Solutions such as board re-spin or major updates to the PLD logic are expensive. The receiver polarity inversion feature is provided to correct this situation.

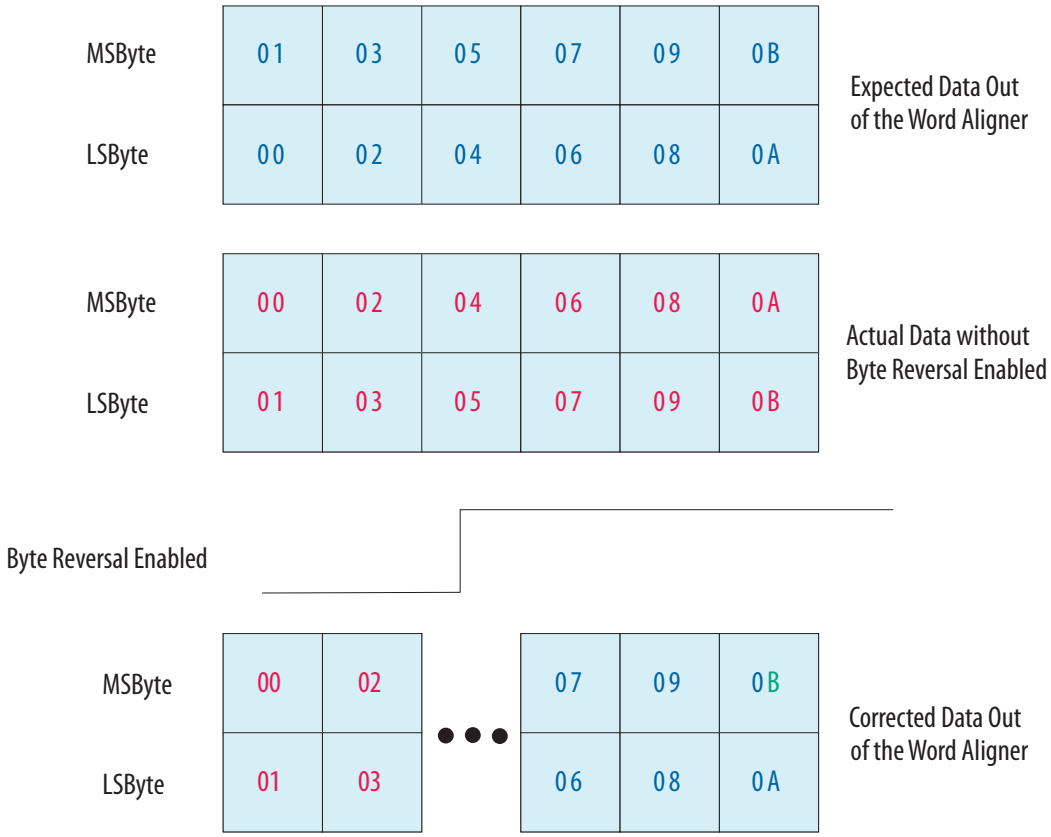
Receiver Bit Reversal

By default, the receiver assumes an 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`. For example, if in 8 bit width mode, `D[7:0]` is rewired to `D[0:7]`.

Receiver Byte Reversal in Custom 16- and 20-Bit Width Configurations

The MSByte and LSByte of the input data to the transmitter may be erroneously swapped. The receiver byte reversal feature corrects this situation.

Figure 1-22: Receiver Byte Reversal Feature



Related Information

For more information on the receiver polarity, bit reversal, and byte reversal features, refer to the "Bit Reversal and Polarity Inversion" section of the Altera Transceiver PHY IP Core User Guide.

PRBS Verifier

The pseudo-random bit stream (PRBS) verifier block verifies the pattern generated by the PRBS generator.

The PRBS verifier:

- Supports 32-bit and 40-bit PMA interfaces
- Supports the following modes and patterns:
  - PRBS31: $x^{31} + x^{28} + 1$
  - PRBS23: $x^{23} + x^{18} + 1$
  - PRBS15: $x^{15} + x^{14} + 1$
  - PRBS9: $x^{9} + x^{5} + 1$
  - PRBS7: $x^{7} + x^{6} + 1$

Note: You can enable either the PRBS verifier or the PRP verifier, but you cannot enable both at the same time.

Related Information

For PRBS verifier implementation information, refer to the Altera Transceiver PHY IP Core User Guide.

Deskew FIFO

The deskew FIFO in each channel receives data from its word aligner. The deskew FIFO circuitry aligns the data across multiple channels.

Note: The deskew circuitry is available only in XAUI mode.

In the XAUI protocol, the code groups received across four lanes of a XAUI link can be misaligned with respect to one another because of skew in the physical medium or differences between the independent clock recoveries per lane.

The XAUI protocol requires the physical layer device to implement a deskew circuit to align all four channels. To enable the deskew circuitry at the receiver, the transmitter sends a /A/ (/K28.3/) code group simultaneously on all four channels during inter-packet gap (IPG). The skew introduced in the physical medium and the receiver channels can cause the /A/ code groups to be misaligned when received.

The deskew operation begins only after link synchronization is achieved on all four channels from the word aligner of each channel. When an aligned set of /A/ code groups is observed at the output of the deskew FIFOs of the four channels, the `rx_channelaligned` signal is asserted high, indicating channel alignment is acquired.

Rate Match (Clock Rate Compensation) FIFO

The rate match (clock rate compensation) FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip (SKP) symbols or ordered sets from the interpacket gap (IPG) or idle streams. 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. The rate match FIFO 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 have the auto-negotiation state machine in the FPGA core with rate match FIFO enabled, refer to the "Rate Match FIFO" section in the "Gigabit Ethernet" section in the *Transceiver Configurations in Stratix V Devices* chapter.

Related Information

- **For more information about the skip pattern and control pattern, refer to the "Rate Match FIFO Parameters" section of the Altera Transceiver PHY IP Core User Guide.**
- **For more information about how to use the rate match FIFO with PCIe, XAUI, and Custom protocols, refer to the Transceiver Configurations in Stratix V Devices chapter.**

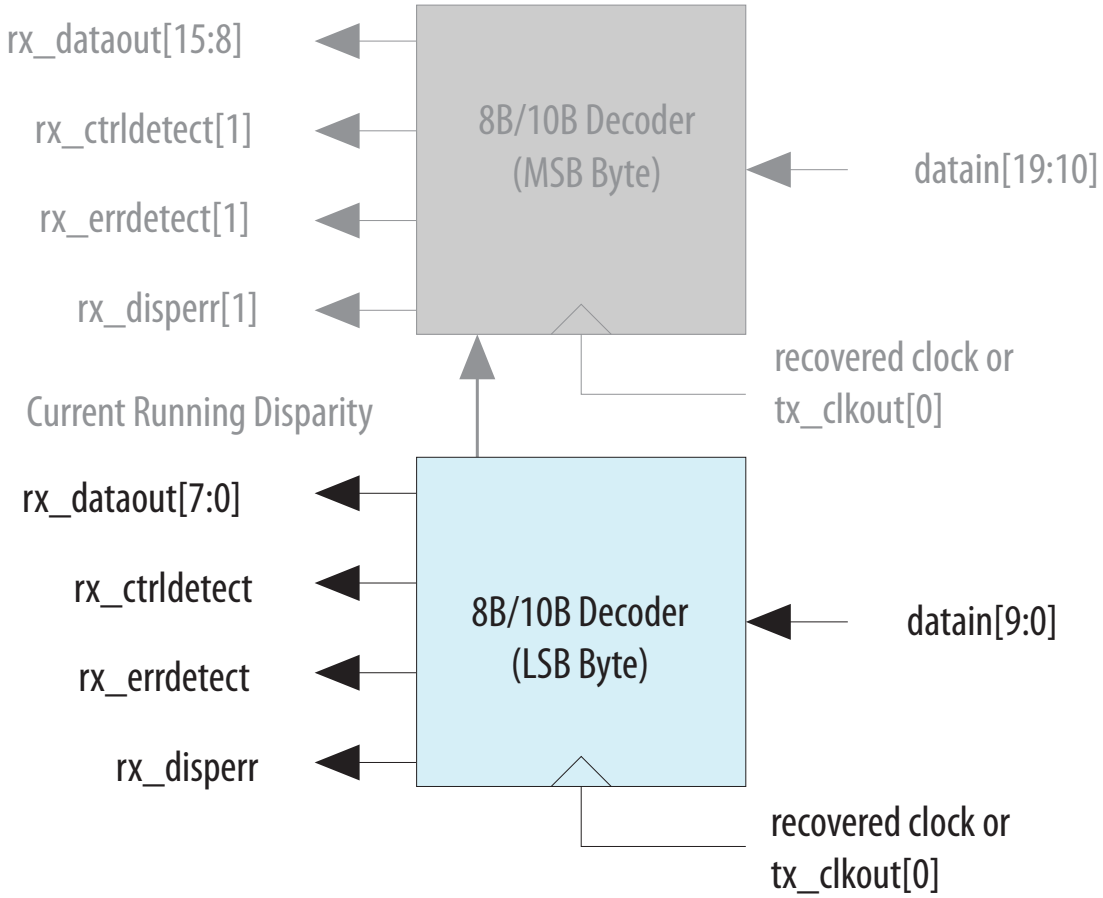
8B/10B Decoder

Many protocols require the serial data sent over the link to be 8B/10B encoded to maintain the DC balance in the transmitted serial data. The PCIe protocol requires the receiver PCS logic to implement an 8B/10B decoder to decode the data before forwarding the data to the upper layers for packet processing.

The receiver channel PCS has an 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.

In 10-bit mode, the 8B/10B decoder receives 10-bit data from the rate match FIFO or word aligner (when the rate match FIFO is disabled) and decodes the data into an 8-bit data +1-bit control identifier. The decoded data is fed to the byte deserializer or the receiver phase compensation FIFO (if the byte deserializer is disabled).

Figure 1-23: 8B/10B Decoder in 10-bit Mode



Note: The 8B/10B decoder is described in IEEE 802.3-2008 clause-49.

In PCIe configuration, the 8B/10B decoder operates only in 10-bit width mode. A PCIe configuration forces selection of the 8B/10B decoder in the receiver datapath.

Control Code Group Detection

The 8B/10B decoder indicates whether the decoded 8-bit code group is a data or control code group on the `rx_datak` signal (not shown in the figure above). If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in the IEEE 802.3 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.

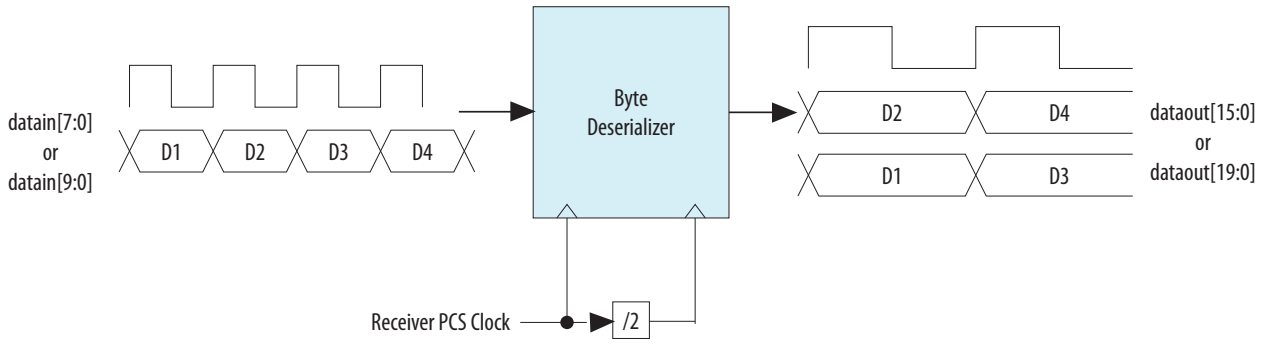
Byte Deserializer

The FPGA fabric-transceiver interface frequency has an upper limit. In designs where the receiver PCS frequency exceeds the upper limit, the byte deserializer is required. The byte deserializer reduces the interface frequency to half while doubling the parallel data width. The byte deserializer is optional in designs that do not exceed the interface frequency upper limit.

Byte Deserializer in 8- and 10-Bit Width Mode

In 8-bit 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 the data into 16- or 20-bit wide data at half the speed.

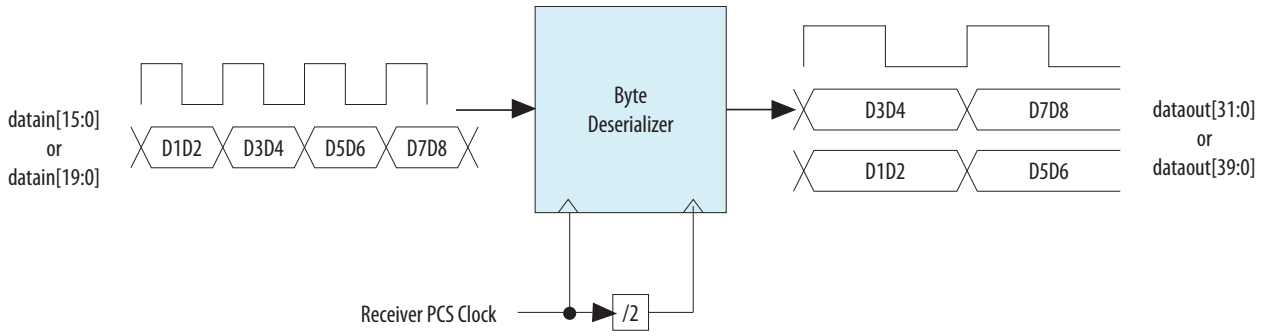
Figure 1-24: Byte Deserializer in 8- or 10-Bit Width Mode



Byte Deserializer in 16- or 20-Bit Width Mode

In 16-bit 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 the data into 32- or 40-bit wide data at half the speed.

Figure 1-25: Byte Deserializer in 16- and 20-Bit Width Mode



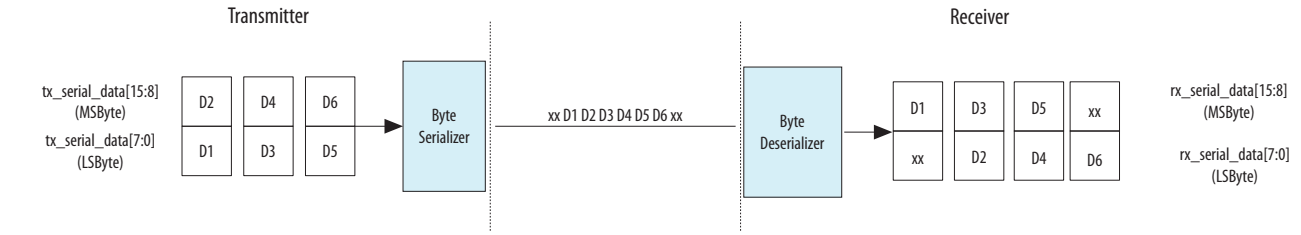
Byte Ordering Block

In 8- or 10-bit width mode with the 16- or 20-bit FPGA fabric-transceiver interface, the byte deserializer receives one data byte (8 or 10 bits) and deserializes the data into two data bytes (16 or 20 bits). Depending on when the receiver PCS logic comes out of reset, the byte ordering at the output of the byte deserializer may or may not match the original byte ordering of the transmitted data. The byte misalign-

ment resulting from byte deserialization is unpredictable because the byte misalignment depends on which byte is being received by the byte deserializer when the byte comes out of reset.

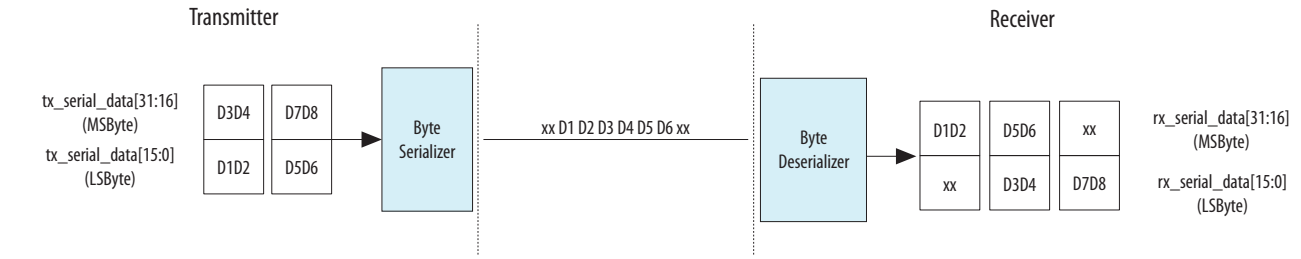
The byte ordering block looks for the user-programmed byte ordering pattern in the byte-deserialized data. You must select a byte ordering pattern that you know is at the LSBytes position of the parallel transmitter data. If the byte ordering block finds the programmed byte ordering pattern in the MSBytes position of the byte-deserialized data, the byte ordering block inserts the appropriate number of user-programmed pad bytes to push the byte ordering pattern to the LSByte(s) position, thereby restoring proper byte ordering.

Figure 1-26: MSByte and LSByte of the Two-Bit Transmitter Data Straddled Across Two Word Boundaries



In 16-bit width mode with a 32-bit FPGA fabric-transceiver interface, the byte deserializer receives two data bytes (16 bits) and deserializes the two data bytes into four data bytes (32 bits).

Figure 1-27: MSByte and LSByte of the Four-Bit Transmitter Data Straddled Across Two Word Boundaries



The byte ordering pattern length and the byte ordering pad pattern length vary depending on the PCS-PMA interface width (8-bit/10-bit/16-bit/20-bit).

Related Information

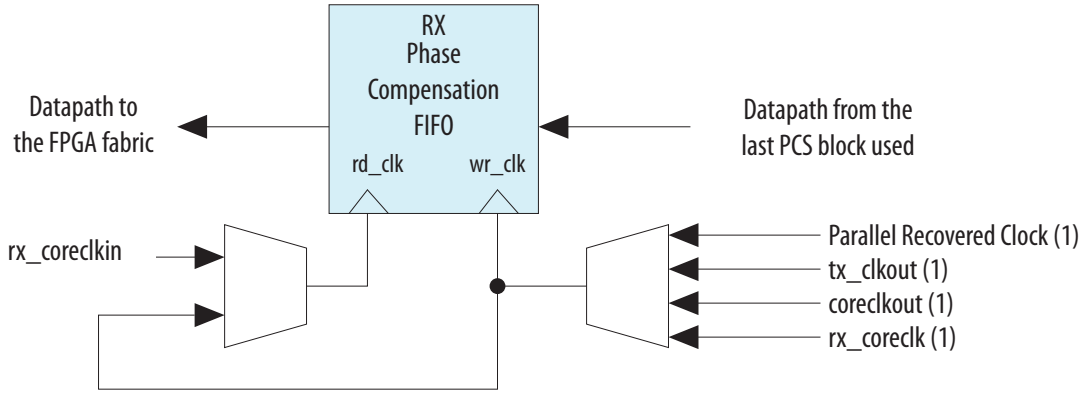
For details about byte ordering and byte ordering pad pattern lengths, refer to the "Byte Ordering Parameters" section of the Altera Transceiver PHY IP Core User Guide.

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 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 1-28: Receiver Phase Compensation FIFO



Note:
1. If you use the byte deserializer, these clocks are divided by two.

Phase Compensation Mode

The receiver phase compensation FIFO compensates for any phase difference between the read and write clocks for the receiver status and data signals.

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

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

Related Information

For a detailed description of the receiver datapath interface clocking modes when using the receiver phase compensation FIFO, seeTransceiver Clocking in Stratix 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 receiver channel to the FPGA fabric. Configure the FIFO to registered mode when interfacing the receiver channel to the PCIe hard IP block to reduce datapath latency. In registered mode, the low-speed parallel clock that is used in the receiver PCS clocks the FIFO.

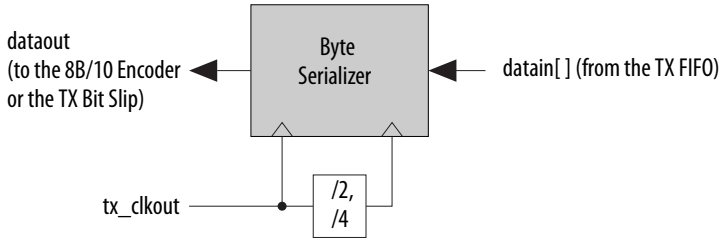
Transmitter Standard PCS Datapath

Note: The Standard PCS is not supported in the GT channels.

Transmitter Phase Compensation FIFO

The transmitter phase compensation FIFO interfaces with the FPGA fabric. The transmitter phase compensation FIFO compensates for the phase difference between the low-speed parallel clock and the FPGA fabric interface clock.

Figure 1-29: Transmitter Phase Compensation FIFO



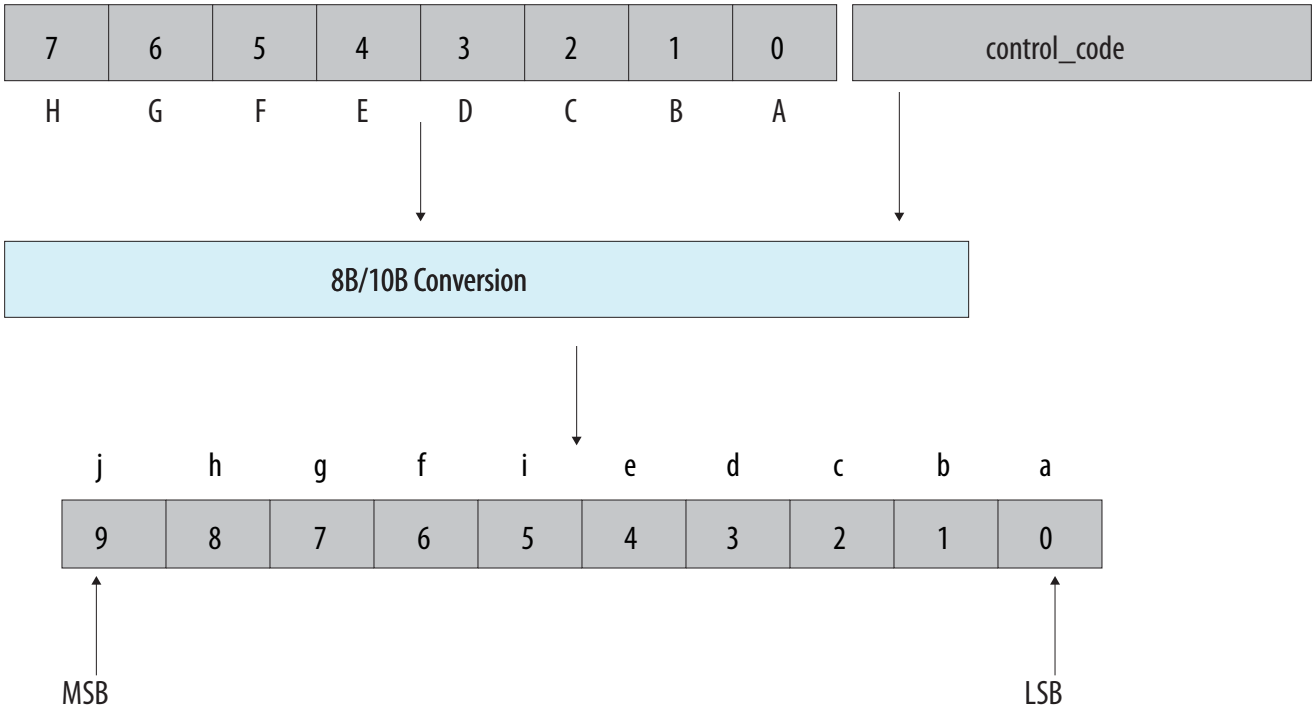
Byte Serializer

The byte serializer allows you to run the transceiver channel at higher data rates while keeping the FPGA fabric interface frequency below the upper limit, by halving the width of the data bus and doubling the data rate. The Byte Serializer forwards the least significant word first followed by the most significant word. For example, assuming a channel width of 32, the byte serializer forwards `tx_parallel_data[15:0]` first, followed by `tx_parallel_data[31:16]`.

8B/10B Encoder

The 8B/10B encoder generates 10-bit code groups from the 8-bit data and 1-bit control identifier. In 8-bit 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.

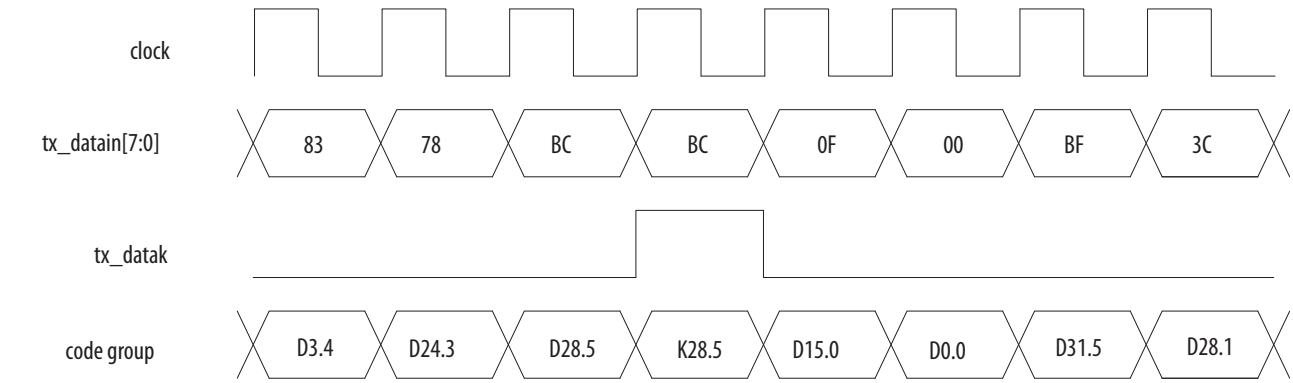
Figure 1-30: 8B/10B Conversion Format



Control Code Encoding

The IEEE 802.3 8B/10B encoder specification identifies only a set of 8-bit characters for which `tx_datak` must be asserted. 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.

Figure 1-31: Control Word and Data Word Transmission



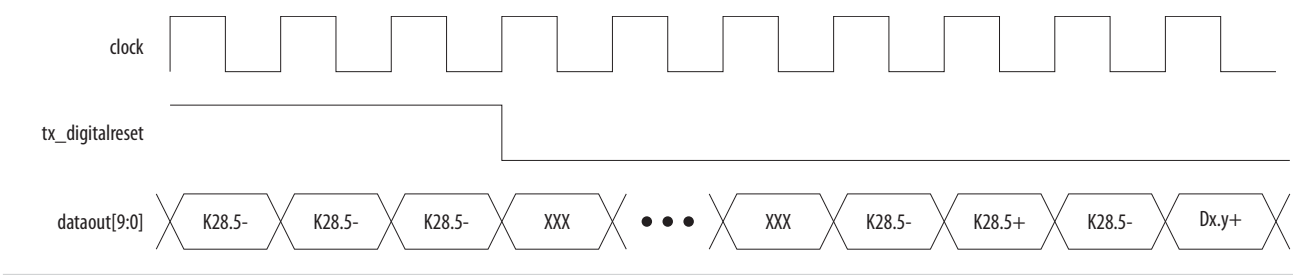
Reset Condition

The `tx_digitalreset` signal resets the 8B/10B encoder. During reset, running disparity (RD) and data registers are cleared. Also, the 8B/10B encoder outputs a K28.5 pattern from the RD- column continuously until `tx_digitalreset` 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 an RD being negative (RD-) and transmits three K28.5 code groups for synchronization before it starts encoding and transmitting the data on its output.

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

When in reset (`tx_digitalreset` is high), a K28.5- (K28.5 10-bit code group from the RD-column) is sent continuously until `tx_digitalreset` 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 1-32: 8B/10B Encoder Output During `tx_digitalreset` Deassertion



Transmitter Polarity Inversion

The positive and negative signals of a serial differential link may be erroneously swapped during board layout. Solutions such as board re-spin or major updates to the PLD logic are expensive. The transmitter polarity inversion feature of the 8B/10B encoder is provided to correct this situation.

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 physical medium attachment (PMA).

Related Information

- **For more information about the 8B/10B code, refer to the "8B/10B Code" section of the Stratix II GX Specifications and Additional Information chapter.**
- **For more information about enabling the polarity inversion and bit-slip features, refer to the "Bit Reversal and Polarity Inversion" section of the Altera Transceiver PHY IP Core User Guide.**

PRBS Generator

The PRBS generator block generates PRBS patterns and square wave patterns.

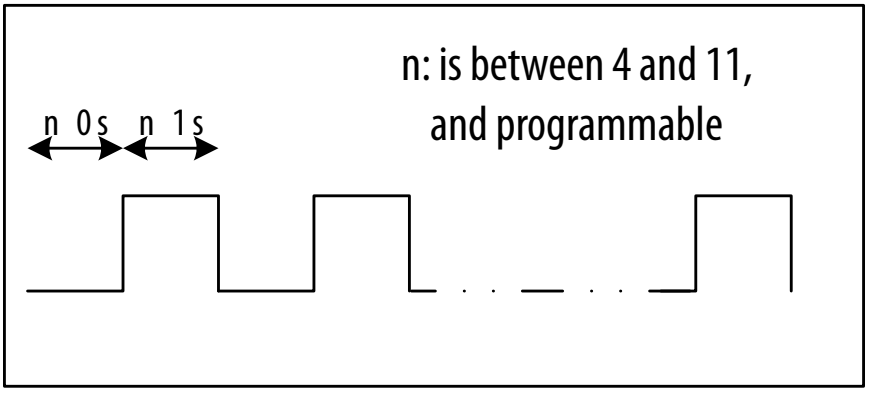
The PRBS generator:

- Supports 32-bit and 40-bit PMA interfaces
- Supports the following modes and patterns:
  - PRBS31: $x^{31} + x^{28} + 1$
  - PRBS9: $x^{9} + x^{5} + 1$
  - PRBS23: $x^{23} + x^{18} + 1$
  - PRBS7: $x^{7} + x^{6} + 1$
  - PRBS15: $x^{15} + x^{14} + 1$

The square-wave generator:

- Has programmable n-number of consecutive serial bit 1s and 0s (where $4 \leq n \leq 11$)
- Supports both 32-bit and 40-bit PMA widths

Figure 1-33: Square Wave Generator



Note: You can enable either the PRBS generator or the PRP generator, but you cannot enable both at the same time.

Related Information

For PRBS generator implementation information, refer to the Altera Transceiver PHY IP Core User Guide

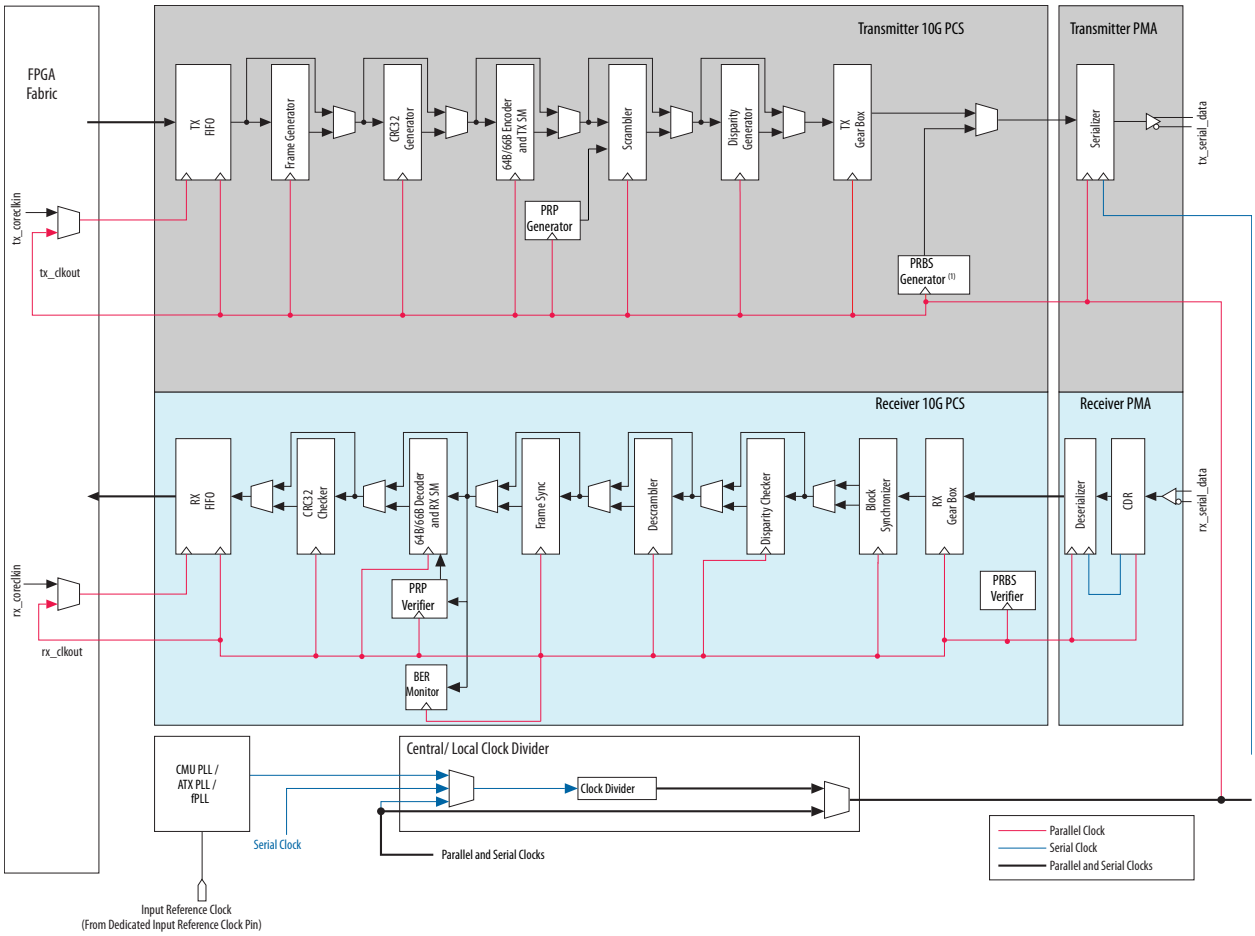
10G PCS Architecture

The 10G PCS architecture offers a full duplex (transmitter and receiver) transceiver channel that supports serial data rates up to 14.1 Gbps for Stratix V GX and GS devices and up to 12.5 Gbps for Stratix V GT devices.

Several functional blocks are customized for various protocols. The different datapath configurations for these protocols are available through the different PHY IPs instantiated through the IP catalog.

Figure 1-34: 10G PCS Datapath in Stratix V GX Channels

Not all the blocks shown in the 10G PCS datapath are available in every configuration.



Note:
1. The PRBS pattern generator can dynamically invert the data pattern that leaves the PCS block.

Related Information

- **For more information about 10G PCS configurations, and clocking schemes and placement restrictions for the different datapath configurations supported in Stratix V devices, refer to the Transceiver Configurations in Stratix V Devices chapter.**
- **For details about implementing the different PCS functional blocks, refer to the Altera Transceiver PHY IP Core User Guide**

Receiver 10G PCS Datapath

The sub-blocks in the receiver 10G PCS datapath are described in order from the receiver gearbox to the receiver FIFO.

Receiver Gearbox

The PMA bus width is smaller than the physical coding sublayer (PCS) bus width; therefore, the receiver gearbox expands the data bus width from the PMA to the PCS. Because bus width adaptation is transparent, you can continuously feed data to the receiver gearbox. In addition to providing bus width adaptation, the receiver gearbox provides the receiver polarity inversion and receiver bit reversal features.

Receiver Polarity Inversion

The receiver gearbox can invert the polarity of the incoming data. This is useful if the receive signals are reversed on the board or backplane layout.

Receiver Bit Reversal

The receiver gearbox allows bit reversal of the received data. Some protocols, such as Interlaken, require the bit reversal feature.

Related Information

- **For more information about enabling the polarity inversion and bit reversal, refer to the "Bit Reversal and Polarity Inversion" section of the Altera Transceiver PHY IP Core User Guide.**
- **For more information about calculating expected latency as a function of the PCS or fabric clocks, refer to the "10G RX FIFO" secttion of the Altera Transceiver PHY IP Core User Guide.**

PRBS Verifier

The pseudo-random bit stream (PRBS) verifier block verifies the pattern generated by the PRBS generator.

The PRBS verifier:

- Supports 32-bit and 40-bit PMA interfaces
- Supports the following modes and patterns:
  - PRBS31: $x^{31} + x^{28} + 1$
  - PRBS23: $x^{23} + x^{18} + 1$
  - PRBS15: $x^{15} + x^{14} + 1$
  - PRBS9: $x^{9} + x^{5} + 1$
  - PRBS7: $x^{7} + x^{6} + 1$

Note: You can enable either the PRBS verifier or the PRP verifier, but you cannot enable both at the same time.

Related Information

For PRBS verifier implementation information, refer to the Altera Transceiver PHY IP Core User Guide.

Receiver Inversion

The PRBS pattern verifier can dynamically invert the data pattern that enters the PCS block.

Table 1-10: PRBS Verifier Inversion Offset

Offset	Bits	R/W	Name	Description
0x16D	[2]	R/W	RX Inversion	Set to 1'b1 to invert the data entering the PCS block.

To invert the PRBS pattern entering the PRBS verifier:

1. Select the logical channel. In this case, logical channel 0.
2. Set the MIF streaming mode to 1.
3. Perform a read-modify-write to bit [2] of offset 0x16D.
4. Assert the channel reset to begin testing on the new PRBS pattern.

The inversion bit on the RX should be disabled to prevent normal data traffic from being inverted while entering the PCS.

Block Synchronizer

The block synchronizer determines the block boundary of a 66-bit word in the case of the 10GBASE-R protocol or a 67-bit word in the case of the Interlaken protocol. The incoming data stream is slipped one bit at a time until a valid synchronization header (bits 65 and 66) is detected in the received data stream. After the predefined number of synchronization headers (as required by the protocol specification) is detected, the block synchronizer asserts the status signal to other receiver PCS blocks down the receiver datapath and to the FPGA fabric.

The block synchronizer is designed in accordance with both the Interlaken protocol specification and the 10GBASE-R protocol specification as described in IEEE 802.3-2008 clause-49.

Disparity Checker

Note: The disparity checker is only used in Interlaken configurations.

The design of the disparity checker is based on the Interlaken protocol specifications. After word synchronization is achieved, the disparity checker monitors the status of the 67th bit of the incoming word and determines whether or not to invert bits [63:0] of the received word.

Table 1-11: Interpretation of the MSB in the 67-Bit Payload for Stratix V Devices

MSB	Interpretation
0	Bits [63:0] are not inverted; the disparity checker processes the word without modification

MSB	Interpretation
1	Bits [63:0] are inverted; the disparity checker inverts the word to achieve the original word before processing it

Descrambler

This function descrambles data per the protocol specifications supported by the 10G PCS. The descrambler operates either in frame synchronous or self synchronous mode.

Frame Synchronous Mode

Frame synchronous mode is used in Interlaken configurations only. When block synchronization is achieved, the descrambler uses the scrambler seed from the received scrambler state word. This block also forwards the current descrambler state to the frame synchronizer.

Self Synchronous Mode

Self synchronous mode is used in 10GBASE-R configurations only.

Frame Synchronizer

Note: The frame synchronizer is only used in Interlaken configurations.

The frame synchronizer block achieves lock by looking for four synchronization words in consecutive metaframes. After synchronization, the frame synchronizer monitors the scrambler word in the metaframe. After three consecutive mismatches, the frame synchronizer deasserts the lock signal and starts the synchronization process again. Lock status is available to the FPGA fabric.

Bit-Error Rate (BER) Monitor

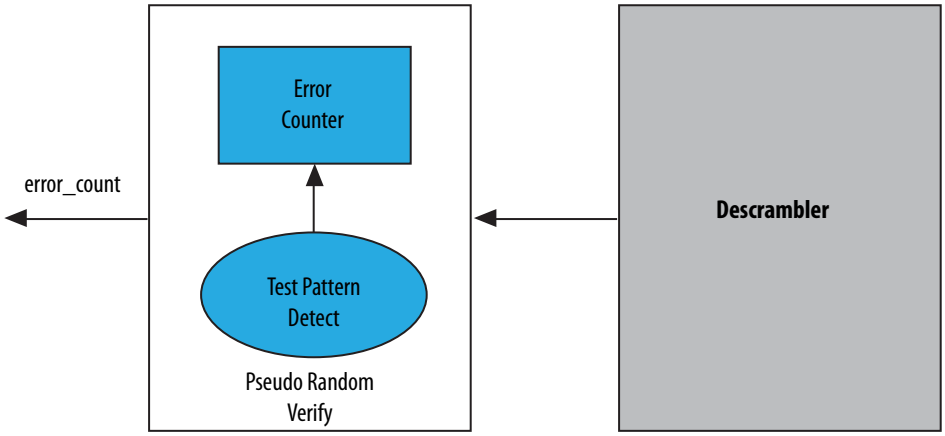
The BER monitor block conforms to the 10GBASE-R protocol specification as described in IEEE 802.3-2008 clause-49. After block lock is achieved, the BER monitor starts to count the number of invalid synchronization headers within a 125-µs period. If more than 16 invalid synchronization headers are observed in a 125-µs period, the BER monitor provides the status signal to the FPGA fabric, indicating a high bit error rate condition.

PRP Verifier

The PRP verifier is available in Stratix V devices for the 10GBASE-R protocol mode. The PRP verifier monitors the output of the descrambler when block synchronization is achieved. You can enable this block for custom 10GBASE-R configurations. The PRP verifier:

- Searches for a test pattern (two local faults, or all 0's) or its inverse
- Tracks the number of mismatches with a 16-bit error counter

Figure 1-35: PRP Verifier Block Diagram



Note: You can enable either the PRP verifier or the PRBS verifier, but you cannot enable both at the same time.

Related Information

For implementation details, refer to the Altera Transceiver PHY IP User Guide.

64B/66B Decoder

Note: The 64B/66B encoder is used only in 10GBASE-R configurations.

The 64B/66B decoder block contains a 64B/66B decoder sub-block and a receiver state machine sub-block. The 64B/66B decoder sub-block converts the received data from the descrambler into 64-bit data and 8-bit control characters. The receiver state machine sub-block monitors the status signal from the BER monitor. If the status signal is asserted, the receiver state machine sends local fault ordered sets to the FPGA interface.

The 64B/66B decoder block is designed in accordance with the 10GBASE-R protocol specification as described in IEEE 802.3-2008 clause-49.

CRC-32 Checker

The CRC-32 checker block supports the Interlaken protocol. The CRC-32 checker calculates the CRC from the incoming data and compares the result to the CRC value sent in the diagnostic word. The CRC error signal is sent to the FPGA fabric.

Receiver FIFO

The receiver FIFO block operates in different modes based on the transceiver datapath configuration.

The Custom and Low Latency PHY IPs automatically select an appropriate receiver FIFO mode for the configuration you use; however, you should select the receiver FIFO mode when using the Native PHY IP.

Clock Compensation Mode

The receiver FIFO is configured in clock compensation mode for the 10GBASE-R configuration. In clock compensation mode, the FIFO deletes idles or ordered sets and inserts only idles to compensate up to a ±100 ppm clock difference between the remote transmitter and the local receiver.

Generic Mode

The receiver FIFO is configured in generic mode for the Interlaken configuration. In generic mode, the receiver FIFO provides the FIFO partially empty and FIFO full status signals to the FPGA fabric to control the read side of the FIFO.

Phase Compensation Mode

The receiver FIFO is configured in phase compensation mode for the 10G custom configuration. In phase compensation mode, the FIFO compensates for the phase difference between the FIFO write clock and the read clock.

Related Information

For more information about the different receiver FIFO operating modes, refer to the Transceiver Configurations in Stratix V Devices chapter.

Transmitter 10G PCS Datapath

The sub-blocks in the transmitter 10G PCS datapath are described in order from the transmitter FIFO to the transmitter gearbox.

Transmitter FIFO

The transmitter FIFO provides an interface between the transmitter channel PCS and the FPGA fabric.

In 10GBASE-R configurations, the transmitter FIFO receives data from the FPGA fabric. The data output from the transmitter FIFO block goes to the 64B/66B encoder.

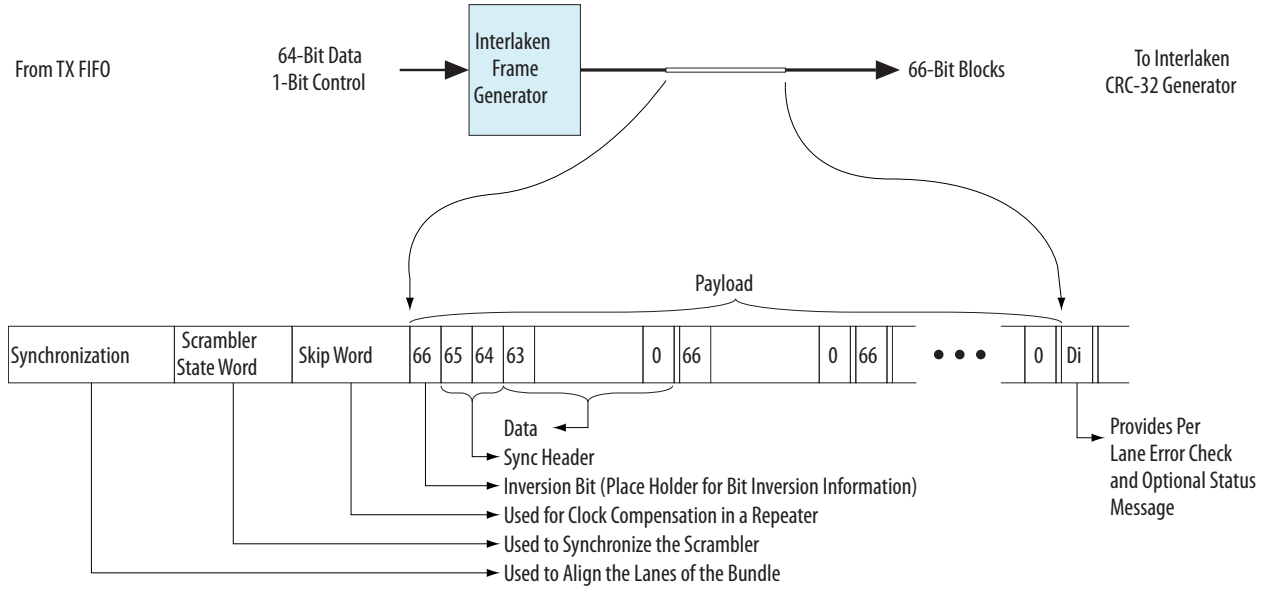
In Interlaken configurations, the transmitter FIFO sends a control signal to indicate whether it is ready to receive data from the FPGA fabric. The user logic sends the data to the transmitter FIFO only if this control signal is asserted. In this configuration, data output from the transmitter FIFO block goes to the frame generator.

Frame Generator

Note: The frame generator is used only in Interlaken configurations.

The frame generator block takes the data from the transmitter FIFO and encapsulates the payload and burst/idle control words from the FPGA fabric with the framing layer's control words, such as the synchronization word, scrambler state word, skip word, and diagnostic word, to form a metaframe. The Interlaken PHY IP Parameter Editor allows you to set the metaframe length.

Figure 1-36: Frame Generator



CRC-32 Generator

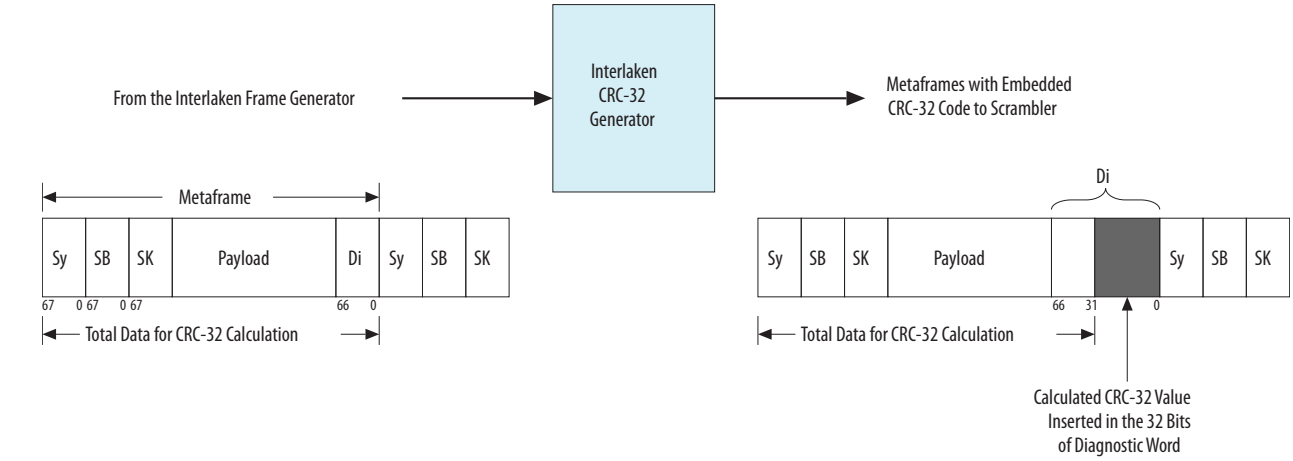
Note: The CRC-32 generator is used only in Interlaken configurations.

The CRC-32 generator block receives data from the frame generator and calculates the cyclic redundancy check (CRC) code for each block of data. This CRC code value is stored in the CRC32 field of the diagnostic word.

The CRC-32 calculation covers most of the metaframe, including the diagnostic word, except the following:

- bits `[66:64]` of each word
- 58-bit scrambler state within the scrambler state word
- 32-bit CRC-32 field within the diagnostic word

Figure 1-37: CRC-32 Generator



64B/66B Encoder

Note: The 64B/66B encoder is used only in 10GBASE-R configurations.

The 64B/66B encoder conforms to the 10GBASE-R protocol specification as described in IEEE 802.3-2008 clause-49.

This block contains the 64B/66B encoder sub-block and the transmitter state machine sub-block. The 64B/66B encoder sub-block receives data from the transmitter FIFO and encodes the 64-bit data and 8-bit control characters to the 66-bit data block required by the 10GBASE-R configuration. The transmit state machine in the 64B/66B encoder sub-block checks the validity of the 64-bit data from the MAC layer and ensures proper block sequencing.

Scrambler

The scrambler operates in frame synchronous mode and self synchronous mode. Frame synchronous mode is used in Interlaken configurations. Self synchronous mode operates in 10GBASE-R configurations, as specified in IEEE 802.3-2008 clause-49.

PRP Generator

The pseudo-random pattern (PRP) generator block operates in conjunction with the scrambler to generate pseudo-random patterns for the RX and TX tests in 10G Ethernet mode. It generates various test patterns from various seeds loaded to the scrambler and select data patterns. You can enable this block for custom 10GBASE-R configurations.

Note: You can enable either the PRP generator or the PRBS generator, but you cannot enable both at the same time.

Related Information

For details about seed and data pattern selection, refer to the Altera Transceiver PHY IP Core User Guide.

Disparity Generator

Note: The disparity generator is used only in Interlaken configurations.

The disparity generator block conforms to the Interlaken protocol specification and provides a DC-balanced data output. The disparity generator receives data from the scrambler and inverts the running disparity to stay within the ±96-bit boundary. To ensure this running disparity requirement, the disparity generator inverts bits `[63:0]` and sets bit `[66]` to indicate the inversion.

Table 1-12: Interpretation of the MSB in the 67-Bit Payload for Stratix V Devices

MSB	Interpretation
0	Bits `[63:0]` are not inverted; the disparity generator processes the word without modification
1	Bits `[63:0]` are inverted; the disparity generator inverts the word before processing it

PRBS Generator

The PRBS generator block generates PRBS patterns and square wave patterns.

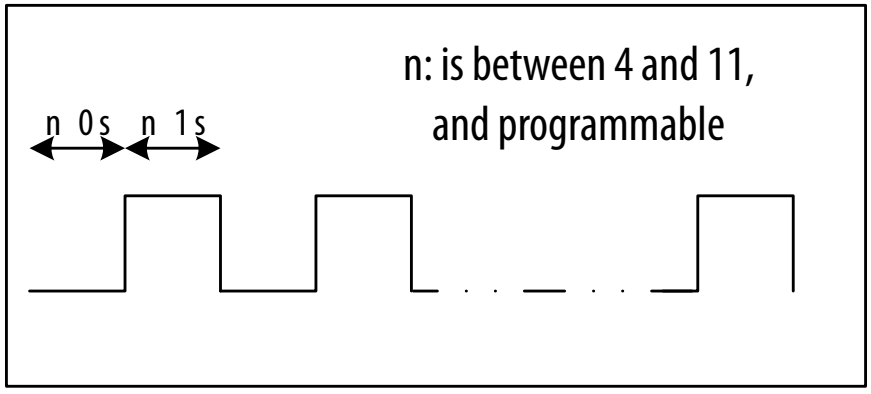
The PRBS generator:

- Supports 32-bit and 40-bit PMA interfaces
- Supports the following modes and patterns:
  - PRBS31: $x^{31} + x^{28} + 1$
  - PRBS9: $x^{9} + x^{5} + 1$
  - PRBS23: $x^{23} + x^{18} + 1$
  - PRBS7: $x^{7} + x^{6} + 1$
  - PRBS15: $x^{15} + x^{14} + 1$

The square-wave generator:

- Has programmable n-number of consecutive serial bit 1s and 0s (where $4 \leq n \leq 11$)
- Supports both 32-bit and 40-bit PMA widths

Figure 1-38: Square Wave Generator



Note: You can enable either the PRBS generator or the PRP generator, but you cannot enable both at the same time.

Related Information

For PRBS generator implementation information, refer to the Altera Transceiver PHY IP Core User Guide

Transmitter Inversion

The PRBS pattern generator can dynamically invert the data pattern that leaves the PCS block.

Table 1-13: PRBS Generator Inversion Offset

Offset	Bits	R/W	Name	Description
0x141	[0]	R/W	TX Inversion	Set to 1'b1 to invert the data leaving the PCS block.

To invert the PRBS pattern leaving the PRBS generator:

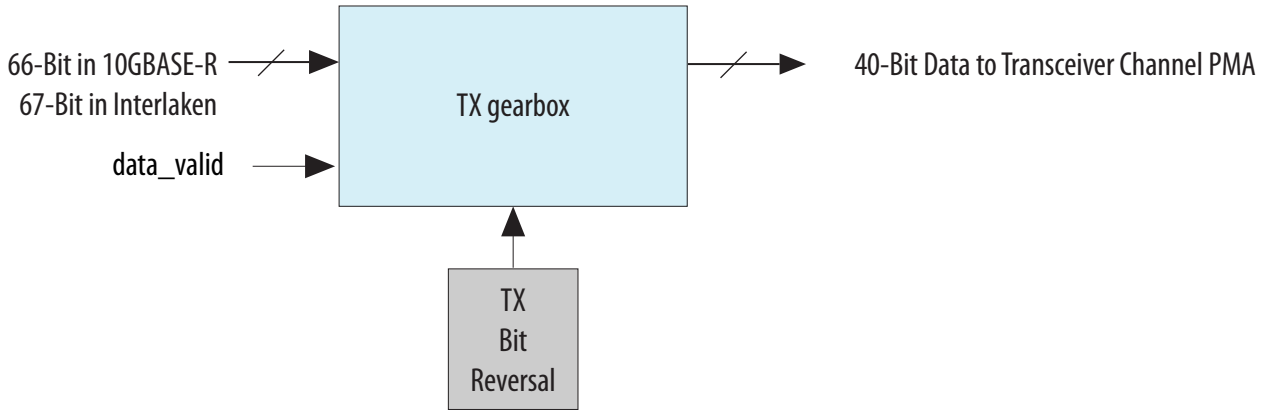
1. Select the logical channel. In this case, logical channel 0.
2. Set the MIF streaming mode to 1.
3. Perform a read-modify-write to bit [0] of offset 0x141.
4. Assert the channel reset to begin testing on the new PRBS pattern.

The inversion bit on the TX should be disabled to prevent normal data traffic from being inverted while leaving the PCS.

Transmitter Gearbox

The transmitter gearbox adapts the PCS data width to a smaller bus width for interfacing with the PMA. Because of the transmitter gearbox, the difference in the bus widths between the PCS and the PMA is transparent to the logic in the FPGA fabric.

Figure 1-39: Transmitter Gearbox



In addition to providing bus width adaptation, the transmitter gearbox provides the transmitter polarity inversion, bit reversal, and bit-slip features.

Transmitter Polarity Inversion

Transmitter polarity can be used to reverse the positive and negative differential buffer signals. This is useful if these signals are reversed on the board or backplane layout.

A high value on the `tx_invpolarity` register, which is accessed via the Avalon-MM PHY management interface, 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. Dynamically changing the `tx_invpolarity` register value 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.

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

Transmitter Bit Reversal

The transmitter gearbox can reverse the order of transmitted bits. By default, the transmitter sends out the LSB of a word first. Some protocols, such as Interlaken, require that the MSB of a word (bit 66 in a word `[66:0]`) is transmitted first. When you enable the transmitter bit reversal, the parallel input to the gearbox is swapped and the MSB is sent out first. The Quartus II software automatically sets the bit reversal for Interlaken configurations.

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 interface, minus one.

The transmitter bit-slip is not supported for all PHYs. Low latency PHY does not allow this feature.

Related Information

- **For the supported PMA-PCS widths and for custom configurations, refer to the Transceiver Configurations in Stratix V Devices chapter.**
- **For more information about enabling the polarity inversion and bit reversal, refer to the "Bit Reversal and Polarity Inversion" section of the Altera Transceiver PHY IP Core User Guide.**
- **For more information about calculating expected latency as a function of the PCS or fabric clocks, refer to the "10G TX FIFO" section of the Altera Transceiver PHY IP Core User Guide.**

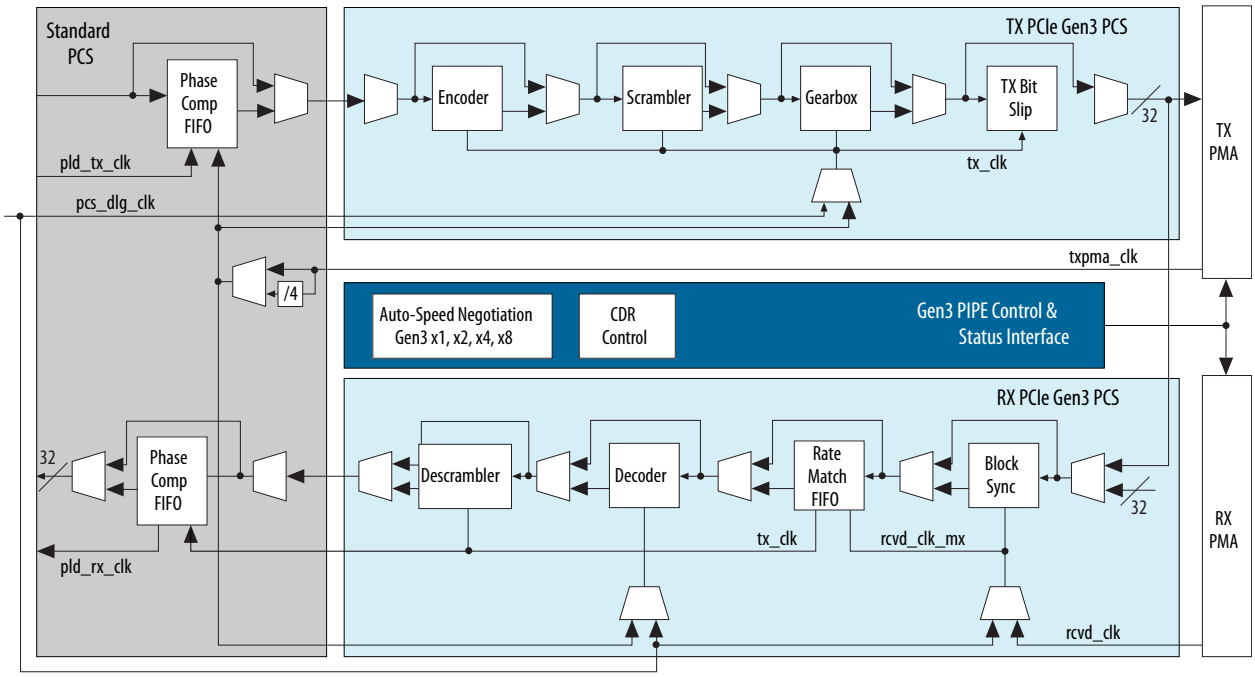
PCIe Gen3 PCS Architecture

Stratix V architecture supports the PCIe Gen3 specification. The PCIe Gen3 uses a 128/130 bit block encoding/decoding scheme which is different from the 8B/10B scheme used in Gen1 and Gen2. The 130-bit block contains a 2-bit sync header and 128-bit data payload. For this reason, Stratix V devices include a separate Gen3 PCS that supports functionality at Gen3 speeds. You can use Altera hard IP and interface to the transceivers, or you can implement your MAC and connect it to the transceiver through the PIPE interface.

This PIPE interface supports the seamless switching of Data and Clock between the Gen1, Gen2, and Gen3 PCS, and provides support for PIPE 3.0 features.

Figure 1-40: PCIe Gen3 PCS Top Level Block Diagram

The RX/TX Phase Comp FIFOs are physically placed in, and shared with, the Standard PCS.



Receiver PCIe Gen3 PCS Datapath

The receiver channel PCIe Gen3 PCS datapath architecture is described from block sync to phase compensation FIFO.

Block Synchronizer

PMA parallelization occurs at arbitrary word boundaries. Consequently, the parallel data from the RX PMA CDR needs to be realigned to meaningful character boundaries. The block sync module searches for the Electrical Idle Exit Sequence Ordered Set (or the last number of fast training sequences (NFTS) Ordered Set) and skip (SKP) Ordered Set to identify the correct boundary for the incoming stream and achieve the block alignment. The block is realigned to the new block boundary following the receipt of a SKP Ordered Set, as it can be of variable length.

Rate Match FIFO

The Rate Match FIFO (or clock compensation FIFO) compensates for minute frequency differences between the local clock (sometimes referred to as the FPGA soft IP clock or FPGA system clock) and the recovered clock. This is achieved by inserting and deleting SKP characters in the data stream to keep the FIFO from going empty or full, respectively.

The Rate Match FIFO is fully compliant with the GigE and PCI-Express (Gen1 and Gen2) protocols. For protocol configurations, the 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

Decoder

The Decoder checks for decode errors in the data stream. It also enables or disables the Descrambler based on the Data and Ordered Set received.

Descrambler

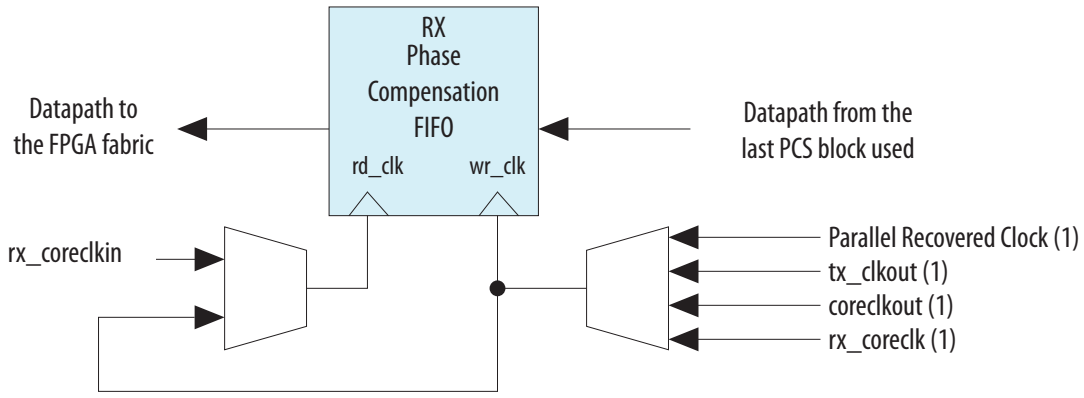
In a multi-lane link environment, each of the receiver lanes may implement a separate linear feedback shift register (LFSR) for de-scrambling. The LFSR uses the following polynomial: $G(X) = X^{23} + X^{21} + X^{16} + X^8 + X^5 + X^2 + 1$. It is a standard PRBS23 polynomial.

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 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 1-41: Receiver Phase Compensation FIFO



Note:
1. If you use the byte deserializer, these clocks are divided by two.

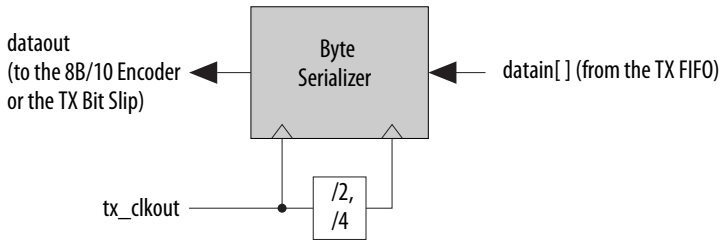
Transmitter PCIe Gen3 PCS Datapath

The transmitter channel PCIe Gen3 PCS datapath is described from phase compensation FIFO to gearbox.

Transmitter Phase Compensation FIFO

The transmitter phase compensation FIFO interfaces with the FPGA fabric. The transmitter phase compensation FIFO compensates for the phase difference between the low-speed parallel clock and the FPGA fabric interface clock.

Figure 1-42: Transmitter Phase Compensation FIFO



Scrambler

In a multi-lane link environment, each of the transmitter lanes may implement a LFSR for scrambling. The LFSR uses the following polynomial: $G(X) = X^{23} + X^{21} + X^{16} + X^8 + X^5 + X^2 + 1$. It is a standard PRBS23 polynomial. The scrambler is used to provide enough edge density, since there is no 8B/10B encoding in PCIe Gen3, so that the RX PMA CDR can lock to the incoming data stream and generate the recovered clock.

Encoder

The PCIe Gen3 base specification defines that the data packets have to be scrambled and descrambled, whereas the Ordered Set packets (except the first symbol of TS1 and TS2 Ordered Set) do not have to be scrambled or descrambled. The Encoder/Decoder continuously checks the header and payload of the packet and generates a signal to enable the scrambler/descrambler based upon whether the payload is an ordered set or a data packet. It also generates a signal to reset the scrambler/descrambler to the initial seed value if an Electrical Idle Exit Ordered Set or a Fast Training Sequence Ordered Set is received or transmitted. In addition, the encoder/decoder logic monitors the Ordered Set and the header for invalid values, and generates an error flag if they do.

Gearbox

The PCIe 3.0 base specification specifies a block size of 130 bits, with the exception of SKP Ordered Sets which can be variable length. An implementation of a 130-bit data path takes significant resources, so the PCIe Gen3 PCS data path is implemented as 32 bits wide. As the TX PMA data width is fixed to 32 bits, and the block size is 130 bits with variations, a gearbox is needed to convert the 130 bits to 32 bits. This gearbox has a transmitter bit-slip feature.

PIPE Interface

The PIPE Data interface on the hard transceiver is compatible with the PIPE 3.0 specification. Advanced equalization signals are not provided in Altera's PIPE interface.

Auto Speed Negotiation

Auto Speed Negotiation controls the operating speed of the transceiver when operating under PIPE 3.0 modes. By monitoring the rate control signal from the physical MAC (PHY MAC) layer, this feature changes the transceiver from PCIe Gen1 operation mode to Gen2 operation mode, or from PCIe Gen1 operation mode to Gen2 operation mode to Gen3 operation mode, or vice versa, with all the appropriate settings.

Electrical Idle Inference

In conjunction with side band signals from the FPGA side, the Electrical Idle Inference feature infers Electrical Idle assuming that the signal detect is not reliable. This is based on the PCIe Base Specification Revision 2.0/3.0.

Clock Data Recovery (CDR) Control

The CDR control feature is used for Rx.L0s fast exit when operating in PIPE/PCIe Gen3 mode. After detecting an Electrical Idle Ordered Set (EIOS), it takes manual control of the CDR by forcing it into a lock-to-reference (LTR) mode. When an exit from electrical idle is detected, this feature moves the CDR into lock-to-data (LTD) mode to achieve fast data lock.

Document Revision History

The revision history for this chapter.

Table 1-14: Document Revision History

Date	Version	Changes

September 2014	2014.09.30	• Added a link to Altera mySupport in the *Link Coupling* section.
		• Added a note to the *10G PCS Datapath in Stratix V GX Channels* figure.
		• Added a note to the *Rate Match (Clock Compensation) FIFO* section.
		• Added the "Word Aligner in Deterministic Latency State Machine Mode" section and updated the Manual Mode description in the *Word Aligner* section.
		• Removed the values for Word Alignment Pattern Length in Bit-Slip mode in the *Word Aligner Options* table.
		• Removed XAUI Mode and PCIe Mode from the *Receiver Phase Compensation FIFO* section.
		• Added the following columns to the Transceiver Calibration Block Boundary for Stratix V Devices table:
		  • Package
		  • Total Number of Transceiver channels in device
		  • Total Number of Transceiver Channels per Side
		• Changed the description of the receiver phase compensation FIFO.
		• Added the *Phase Compensation Mode* section.
		• Added the *Registered Mode* section.
		• Added the *Receiver Inversion* section.
		• Added the *Transmitter Inversion* section.
		• Changed "MegaWizard Plug-in Manager" to "IP catalog" in the *10G PCS Architecture* section.
		• Updated the *PCIe Gen3 PCS Top Level Block Diagram* to show the `pld_rx_clk` as an input to the Phase Compensation FIFO block.

January 2014	2014.01.07	• Updated the *Stratix V GX/GT Channel and PCIe Hard IP Layout* section. • Updated the *Stratix V GS Channel and PCIe Hard IP Layout* section. • Updated the *Channel Variants* section. • Updated the *GS/GT/GX Device Variants and Packages* section. • Updated the *Receiver Equalizer Gain Bandwidth* section.

Date	Version	Changes
October 2013	2013.10.11	• Updated the *Word Aligner* section. • Updated the *Lock-to-Reference Mode* section.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base • Updated Figure 1-10. • Updated Figure 1-15. • Updated the *Receiver Deserializer* section. • Updated the *Continuous Time Linear Equalization* section. • Added the *GS/GT/GX Device Variants and Packages* section. • Added the *Stratix V GS Channel and PCIe Hard IP Layout* section. • Updated Figure 1-17. • Updated Figure 1-18. • Updated Figure 1-30. • Added the *PRBS Verifier* section. • Added the *PRBS Generator* section. • Added the *PRP Verifier* section. • Added the *Serial Bit Checker* section. • Updated the *Decision Feedback Equalization* section. • Updated the *Transmitter Analog Settings* section. • Updated the *Receiver PMA Bit-Slip* section. • Updated the *ATX PLL Calibration* section. • Updated the *Calibration Block Boundary* section. • Updated Figure 1-20. • Updated the *8B/10B Decoder* section. • Updated the *Transmitter Phase Compensation FIFO* section.
December 2012	2012.12.17	Reorganized content and updated template .

Date	Version	Changes
June 2012	2.3	• Updated Figure 1–6, Figure 1–10, and Figure 1–11. • Updated Table 1–3 • Updated "Stratix V Device Layout", "PMA Architecture", "Standard PCS Architecture" and "10G PCS Architecture" sections. • Updated Table 1–2, Table 1–4, Table 1–1, and Table 1–5. • Updated Figure 1–1, Figure 1–3, Figure 1–4, Figure 1–8, and Figure 1–21. • Updated "Transmitter Polarity Inversion" section. • Added "PCIe Gen3 PCS Architecture" section.
February 2012	2.2	• Updated Figure 1–1. • Updated "Transmitter Polarity Inversion" section.

Transceiver Clocking in Stratix V Devices

2

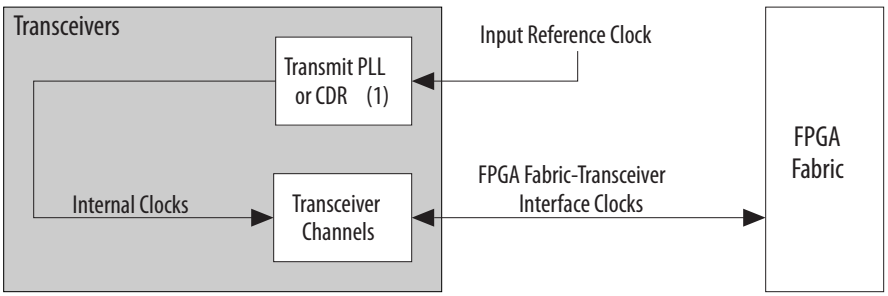
2014.09.30

SV52003

Subscribe Send Feedback

This chapter provides information about the Stratix 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



Note: (1) The transmit phase-locked loop (PLL) can be a CMU PLL (channel PLL), fPLL (fractional PLL Clock), or an ATX PLL (Auxiliary Transmit PLL).

Related Information

Stratix V Device Handbook: Known Issues

Lists the planned updates to the *Stratix V Device Handbook* chapters.

Input Reference Clocking

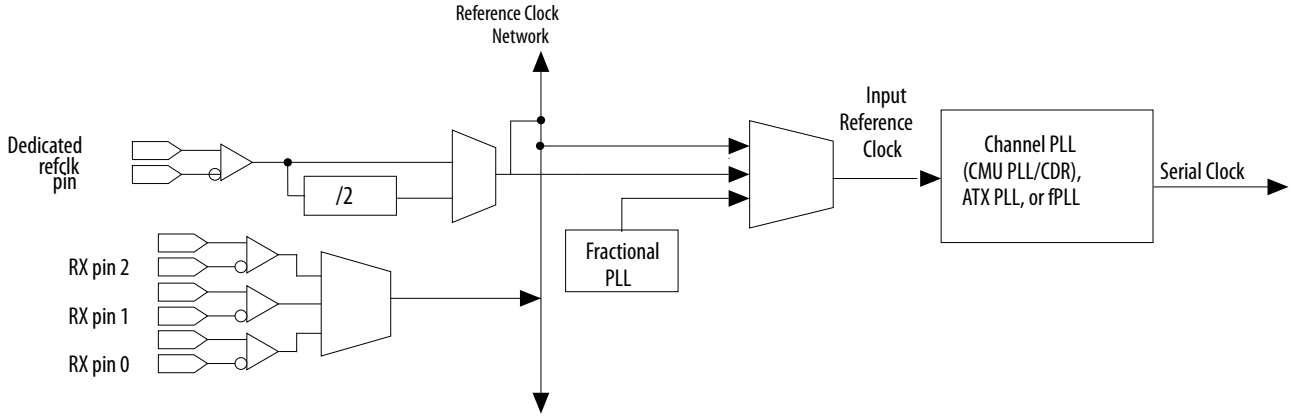
The reference clock for the transmitter PLL and CDR generates the clocks required for transceiver operation.

Each transceiver channel has a channel PLL that can be configured as a transmitter clock multiplier unit (CMU) PLL or a receiver CDR PLL. In the CMU PLL configuration, the channel PLL uses the input reference clock to generate a serial clock. In the receiver CDR PLL configuration, the channel PLL locks to the input reference clock in lock-to-reference (LTR) mode. The auxiliary transmit (ATX) PLL and the fractional PLL use the input reference clock to synthesize a serial clock.

Input Reference Clock Sources

The channel PLL, ATX PLL, and fractional PLL can derive the input clock from a dedicated `refclk` pin, another fractional PLL, or through the reference clock network.

Figure 2-2: Input Reference Clock Sources to Transmit PLLs and CDR

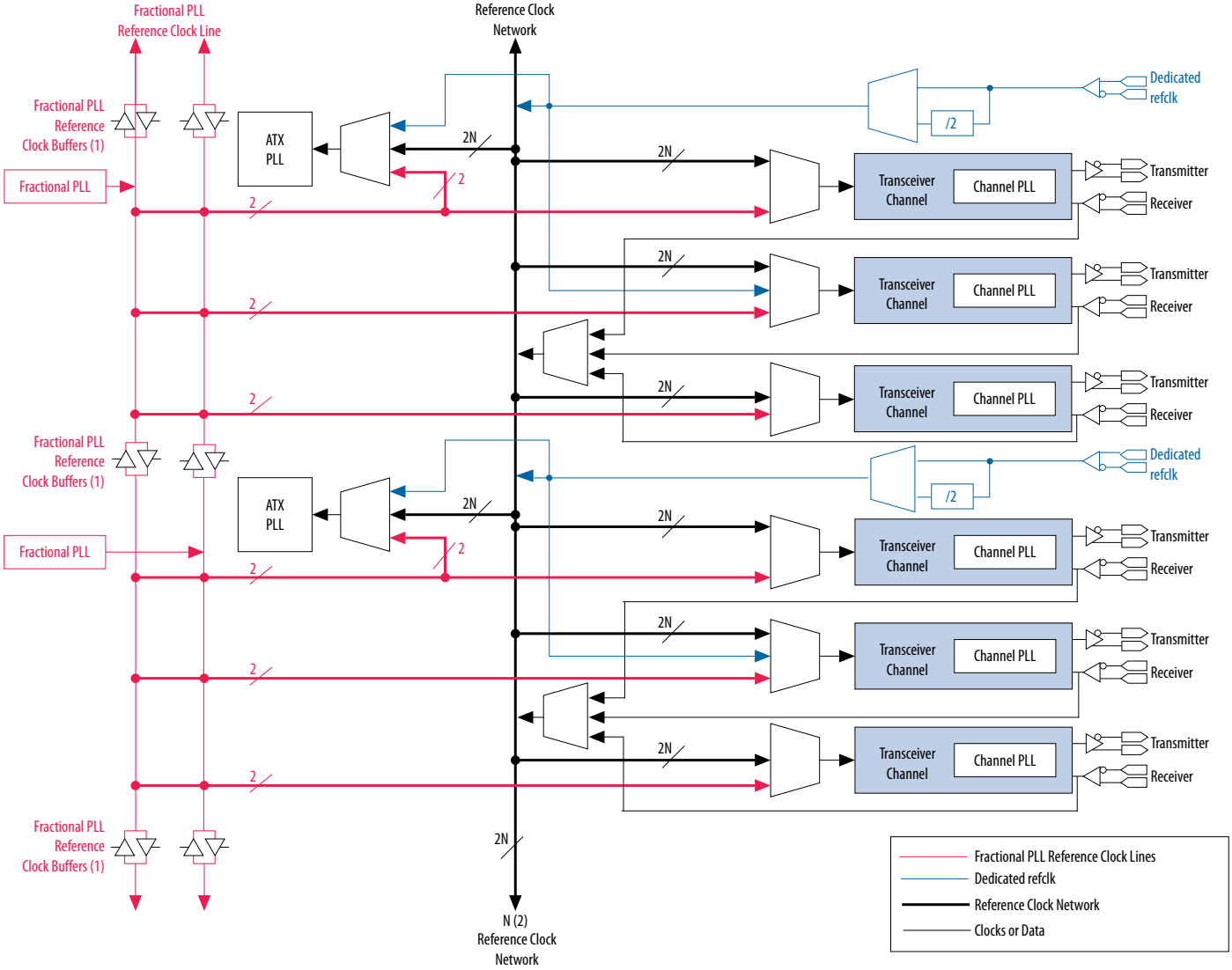


Note : You can choose only one of the three RX pins to be used as a reference clock source.

Note: For optimal performance, use the `refclk` source that is closest to the transmit PLL in the same transceiver bank.

Figure 2-3: Input Reference Clock Sources for GX Transceiver Channels

For more information about the fractional PLL input clock sources shown in the following figure, refer to **Figure 2-6**.

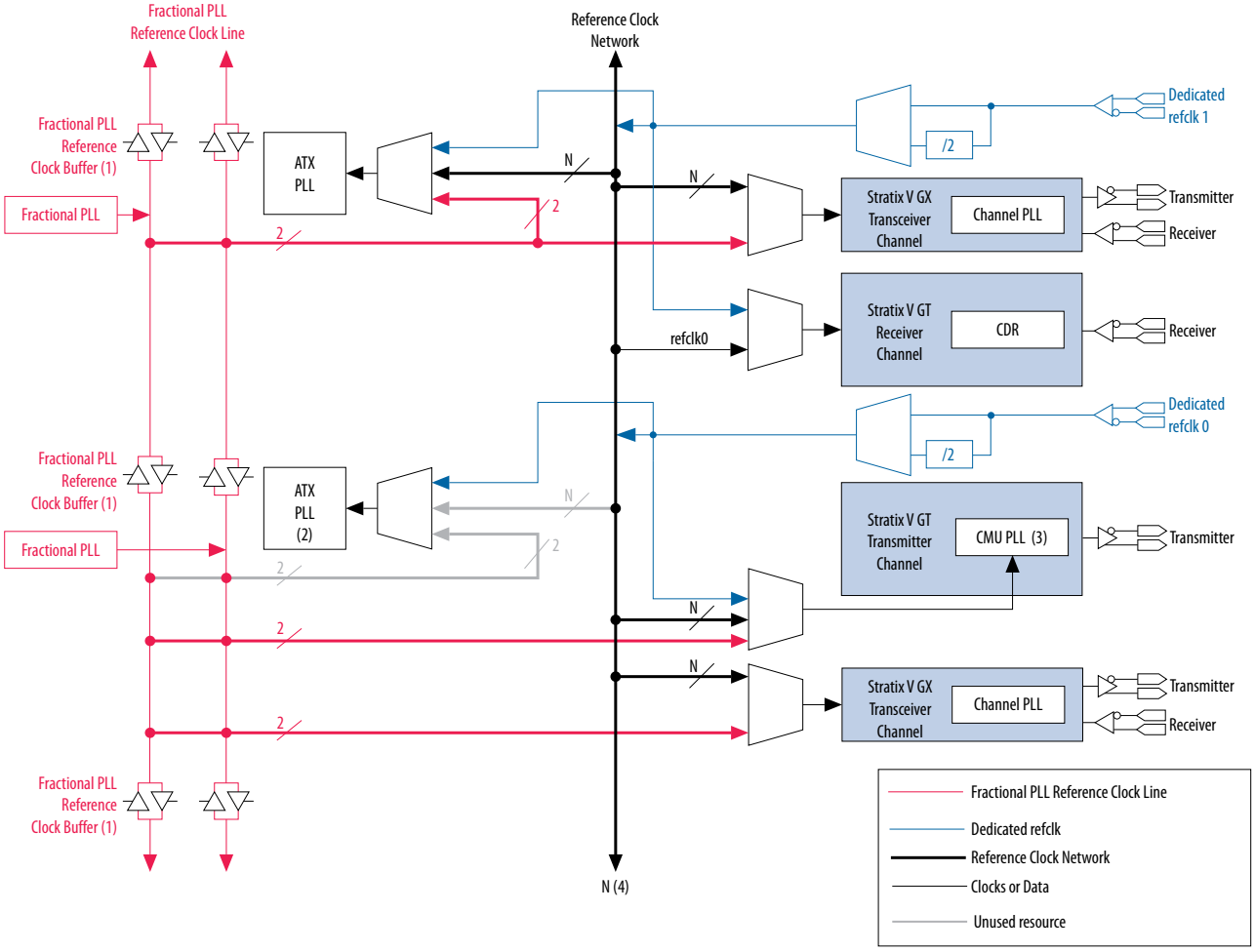


Notes: (1) The fractional PLL refclk buffers allow you to segment the reference clock line into multiple segments, such that fractional PLLs in dif erent transceiver banks can drive the same fractional PLL reference clock line.
(2) N equals the number of transceiver channels on a side divided by 3, which is equal to the number of dedicated refclk pins.

The following figure shows the input reference clock sources for a GT transceiver channel and two GX transceiver channels in a GT transceiver bank.

Figure 2-4: Input Reference Clock Sources for GT and GX Transceiver Channels in Stratix V GT Devices

For more information about the fractional PLL input clock sources shown in the following figure, refer to **Figure 2-6**.



Notes:
(1) The fractional PLL refclk buffers allow you to segment the reference clock line into multiple segments, such that fractional PLLs in dif erent transceiver banks can drive the same fractional PLL reference clock line.
(2) The bottom ATX PLL of a GT transceiver bank provides the serial clock to the GT transmitter channel.
(3) The CMU PLL of the GT transmitter channel drives an x1 clock line that can be used by the top and bottom GX transceiver channels in the GT transceiver bank.
(4) N equals twice the number of GT channels.

Note: Altera recommends using a dedicated clock `refclk0` for the bottom ATX PLL that provides the serial clock to the GT transmitter channel.

Dedicated refclk Pins

GX transceiver banks have one dedicated `refclk` pin for each group of three transceiver channels. The dedicated `refclk0/refclk1` pins can drive reference clock network or ch1/ch4 channel PLLs respectively in a transceiver bank.

There are two dedicated `refclk` pins available in each GT transceiver bank. The two `refclk` pins can also provide the reference clocks to the GX channels in a GT transceiver bank through the reference clock network.

The following table lists the electrical specifications for the input reference clock signal driven on the refclk pins.

Table 2-1: Electrical Specifications for the Input Reference Clock

Protocol	I/O Standard	Coupling	Termination
PCI Express (PCIe)	• 1.2V PCML, 1.4 PCML • 1.4V PCML • 1.5V PCML • 2.5V PCML • Differential LVPECL • LVDS	AC	On - Chip (1)
	• HCSL (2)	DC	Off - Chip(3)
All other protocols	• 1.2V PCML, 1.4 PCML • 1.4V PCML • 1.5V PCML • 2.5V PCML • Differential LVPECL • LVDS	AC	On - Chip (1)

Note: If you select the HCSL I/O standard for the PCIe reference clock, add the following assignment to your project's quartus settings file (.qsf):

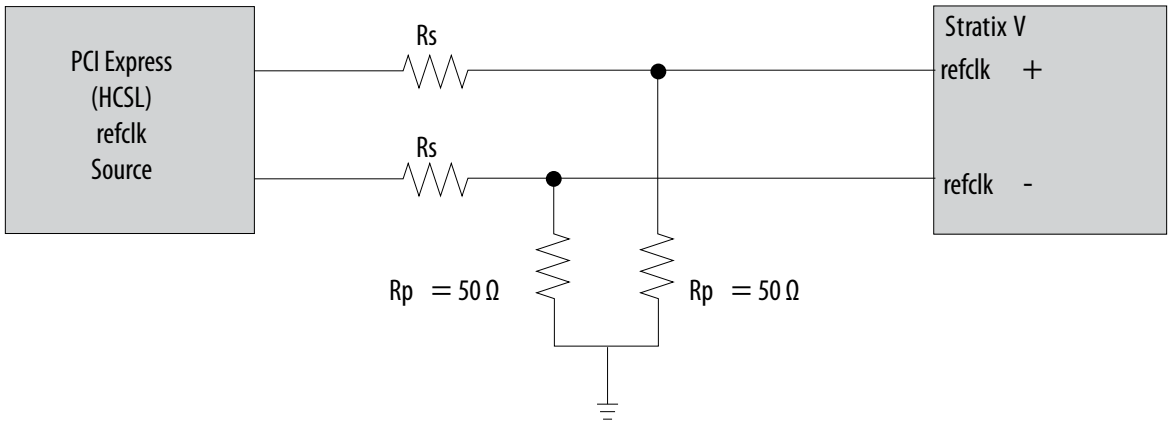
```
set_instance_assignment -name XCVR_REFCLK_PIN_TERMINA-
TION_DC_COUPLING_EXTERNAL_RESISTOR -to <refclk_pin_name>
```

(1) For more information about termination values supported, refer to the *DC Characteristics* section in *Stratix V Device Datasheet.*

(2) In PCIe mode, you have the option of selecting the HCSL standard for the reference clock if compliance to the PCIe protocol is required. You can select this I/O standard option only if you have configured the transceiver in PCIe mode.

(3) For an example termination scheme, refer to **Figure 2-5**

Figure 2-5: Termination Scheme for a Reference Clock Signal When Configured as HCSL



Note: 1. No biasing is required if the reference clock signals are generated from a clock source that conforms to the PCIe specification
2. Select Rs and / or Rp resistor values as recommended by the PCIe clock source vendor.

Related Information

Stratix V Device Datasheet.

Dedicated refclk Pins Using the Reference Clock Network

Each dedicated `refclk` pin can drive any transmitter PLL on the same side of the device through the reference clock network. Designs using multiple transmitter PLLs that require the same reference clock frequency and are located along the same side of the device can share the same dedicated `refclk` pin.

RX Pins Using the Reference Clock Network

The RX pins can be used as `refclk` pins. The RX pins can drive any transmitter PLL on the same side of the device through the reference clock network. Only one RX differential pin pair per three channels can be used as a reference clock and there is no /2 factor available, unlike the dedicated reference clock pin, as shown in the **Figure 2-3**.

Note: For more information about the QSF assignments, refer to the *Altera Transceiver PHY IP Core User Guide* and the *Stratix V Device Datasheet* for the supported I/O standards.

Related Information

- **Altera Transceiver PHY IP Core User Guide**
- **Stratix V Device Datasheet**

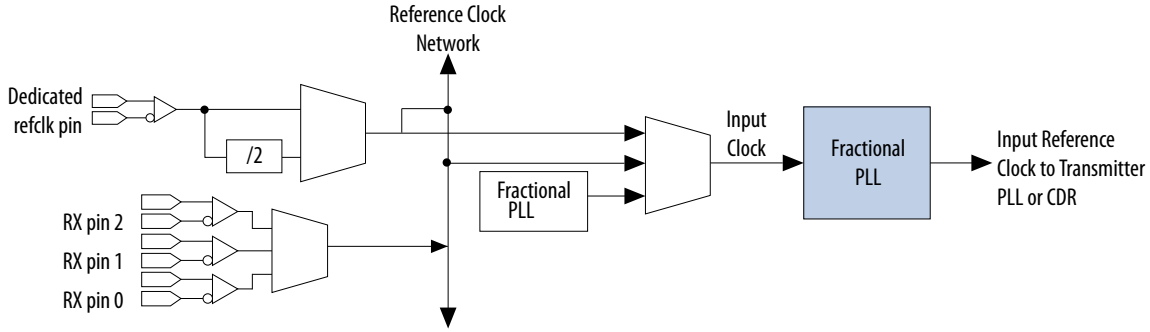
Fractional PLLs

Stratix V devices provide a fractional PLL for each group of three transceiver channels.

Each fractional PLL drives one of two clock lines spanning the side of the device that can provide an input reference clock to any transmitter PLL or CDR on the same side of the device. A fractional PLL enables you to use an input reference clock in your system that is not supported by the transmitter PLL or CDR to synthesize a supported input reference clock.

Figure 2-6: Fractional PLL Input Clock Sources

The following figure shows the input clock sources for the fractional PLLs located within the transceiver banks.



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

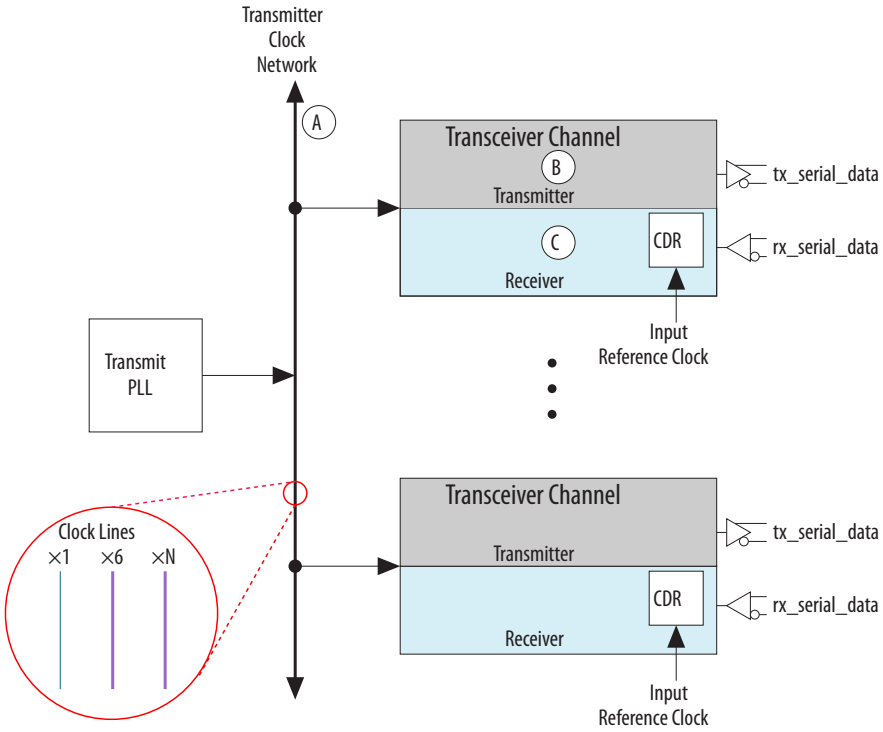
In the internal clocking architecture, different physical coding sublayer (PCS) configurations and channel bonding options result in various transceiver clock paths.

Table 2-2: Internal Clocking Subsections

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

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 2-7: Internal Clocking



The reference clock from one input source is fed to a transmitter PLL. The transmitter PLL could be either a channel PLL configured as a CMU PLL, or an ATX PLL, or a fractional PLL. The transmitter PLL generates a serial clock that is distributed using a transmitter clock network to the transceiver channels.

Note: The clocking described in this section is internal to the transceiver, and the clock routing is primarily performed by the Quartus® II software, based on the transceiver configuration selected.

Transmitter Clock Network

The transmitter clock network routes the clock from the transmitter PLL to the transmitter channel.

The transmitter clock network provides two clocks to the transmitter channel:

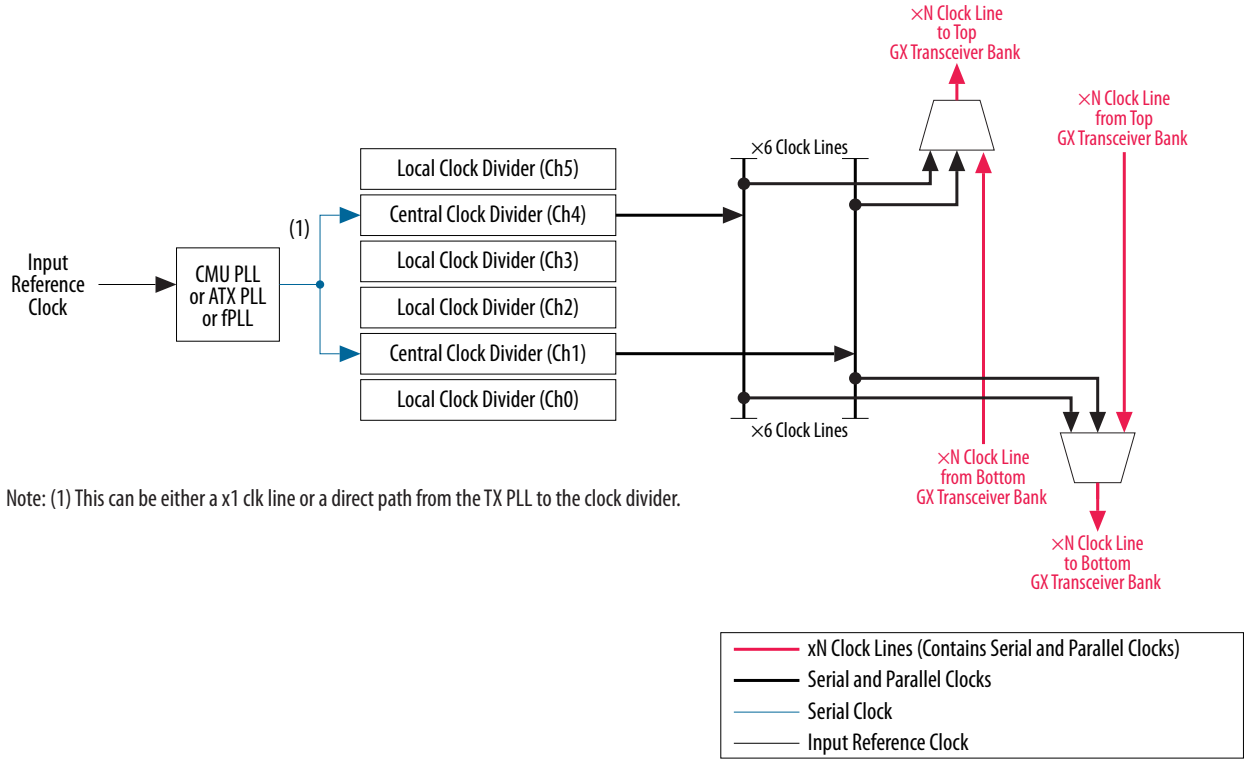
- Serial clock—high-speed clock for the serializer
- Parallel clock—low-speed clock for the serializer and the PCS

Stratix V transceivers support various non-bonded and bonded transceiver clocking configurations. If you use a bonded configuration, both the serial clock and the PCS internal parallel clock are routed from the transmitter PLL to the transmitter channel. If you use a non-bonded configuration, then only the serial clock is routed from the transmitter PLL to the transmitter channel and the PCS internal parallel clock is generated by the clock divider of each channel.

Note: The reference clock and the PCS internal parallel clock are not the same. The reference clock as described in **Input Reference Clocking** on page 2-1 is used to drive the transmit PLL, which generates the serial clock. The PCS internal parallel clock is derived from the serial clock, and equals the serial clock divided by the serialization factor of the serializer.

Figure 2-8: Transmitter Clock Network

The following figure shows the transceiver clock network, beginning with the input reference clock, followed by the transmitter PLL, clock dividers, and ending with the x6 and xN clock lines.



Note: (1) This can be either a x1 clk line or a direct path from the TX PLL to the clock divider.

Note: For more information about bonding, refer to the "Bonded Configurations" section of the *Transceiver Architecture in Stratix V Devices* chapter.

Related Information

Transceiver Architecture in Stratix V Devices

Transmitter Clock Lines

The transmitter clock network consists of two types of dedicated clocking resources.

The dedicated clocking resources are:

- Non-bonded configurations

  x1 clock lines
  xN clock lines (in non-bonded configurations available only for Native PHY)

- Bonded configurations (not available for GT transceiver channels)

  x6 clock lines
  x6 PLL Feedback Compensation
  xN clock lines (available only for PCIe and Native PHY)

Note: The Quartus II software performs the clock routing related to the transmitter clock network based on the transceiver configuration selected.

Table 2-3: Data Rates and Spans Supported Using Stratix V Clock Sources and Clock Networks

Clock Network	Transceiver Channel	Clock Source	Max Data Rate	Bonding	Span
x1	GX	ATX PLLs in a transceiver bank	14.1 Gbps(4)	No	Transceiver bank
		CMU PLLs in a transceiver bank	12.5 Gbps (4)		Transceiver bank
		Fractional PLLs in a transceiver bank	3.125 Gbps		fPLLs can only span upper or lower 3 channels in a transceiver bank.
xN (Native PHY)	GX	ATX PLLs in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive only the serial clock from the x6 clock lines.	8 Gbps	No	xN lines span a side of the device. Specified datarate can drive up to 13 data channels above and up to 13 data channels below TX PLL.
		Channel PLLs in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive only the serial clock from the x6 clock lines.	7.99 Gbps		
		Fractional PLLs in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive only the serial clock from the x6 clock lines.	3.125 Gbps		
x1	GT	Bottom ATX PLL in a GT transceiver bank	28 Gbps (4)	No	Transceiver bank

(4) For the fastest speed grade only. For the remaining speed grades, refer to the *Stratix V Device Datasheet.*

Clock Network	Transceiver Channel	Clock Source	Max Data Rate	Bonding	Span
x6	GX	ATX PLLs in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The x6 clock lines receive both the serial and parallel clock from the central clock dividers.	14.1 Gbps (4)	Yes	Transceiver bank
		The channel (CMU) PLLs provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The x6 clock lines receive both the serial and parallel clock from the central clock dividers.	12.5 Gbps (4)		
		Fractional PLLs provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The x6 clock lines receive both the serial and parallel clock from the central clock dividers.	3.125 Gbps		
x6 PLL Feedback Compensation (5)		One ATX PLL per bonded transceiver bank provides a serial clock to the central clock dividers of Ch 1 and Ch 4. The central clock dividers in the transceiver bank drive the x6 clock lines and provide feedback path to the ATX PLL. The x6 clock lines receive both the serial and parallel clocks from the central clock dividers.	14.1 Gbps (4)	Yes	x6 lines span a transceiver bank. The x6 lines across multiple transceiver banks can be bonded together through PLL feedback compensation path to span the entire side of the device.
		One CMU PLL per bonded transceiver bank provides a serial clock to the central clock dividers of Ch 1 and Ch 4. The central clock dividers in the transceiver bank drive the x6 clock lines and provide feedback path to the CMU PLL. The x6 clock lines receive both the serial and parallel clocks from the central clock dividers.	12.5 Gbps (4)		

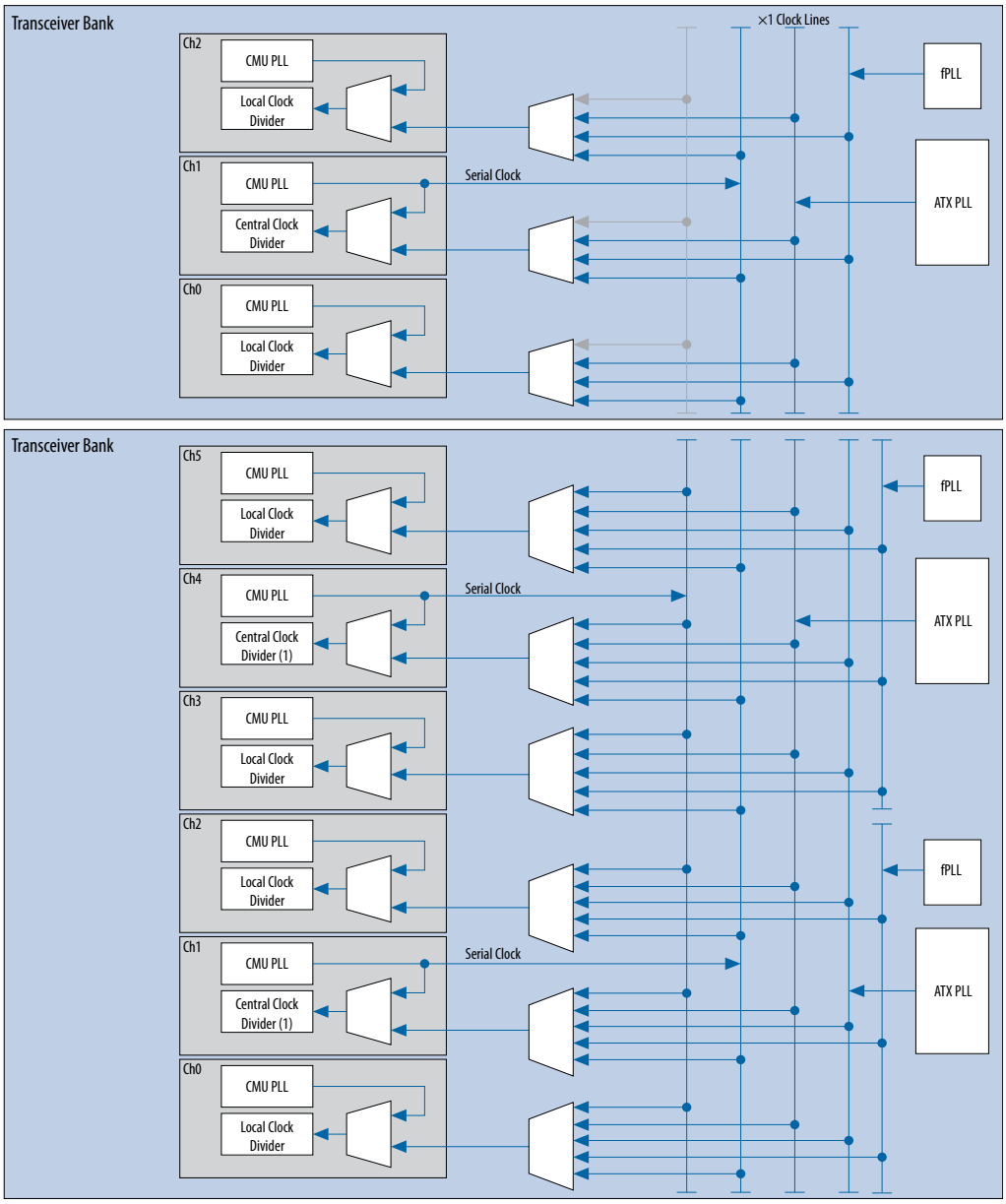
(5) The input reference clock frequency of the transmit PLL must be the same as the parallel clock frequency which clock the PCS bonded channels.

Clock Network	Transceiver Channel	Clock Source	Max Data Rate	Bonding	Span
xN (PCIe)(6)	GX	The ATX or channel (CMU) PLL provides a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive the serial and parallel clocks from the x6 clock lines.	8 Gbps	Yes	xN lines span a side of the device, but can bond only up to eight contiguous data channels.
xN (Native PHY)	GX	ATX PLLs in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive the serial and parallel clocks from the x6 clock lines.	9.8304 Gbps (4)	Yes	xN lines span a side of the device. Specified datarate can bond up to 7 contiguous data channels above and up to 7 contiguous data channels below TX PLL.
			8 Gbps	Yes	xN lines span a side of the device. Specified datarate can bond up to 13 contiguous data channels above and up to 13 contiguous data channels below TX PL
		Channel (CMU) PLLs in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive the serial and parallel clocks from the x6 clock lines.	7.99 Gbps	Yes	xN lines span a side of the device. Specified datarate can bond up to 13 contiguous data channels above and up to 13 contiguous data channels below TX PL
		Fractional PLLs (fPLLs) in a transceiver bank provide a serial clock to the central clock dividers of Ch1 and Ch4. The central clock dividers in the transceiver bank drive the x6 clock lines. The xN clock lines receive the serial and parallel clocks from the x6 clock lines.	3.125 Gbps		

Note: Stratix V devices 5SGXB5, 5SGXB6, 5SGSB, and 5SGSB have one transceiver bank on each side with only three transceiver channels. For more information, refer to the *Transceiver Architecture in Stratix V Devices* chapter.

(6) For more information about PCIe x8 configurations, refer to the section titled "Hard IP x8 Configuration" in the *Transceiver Configurations in Stratix V Devices* chapter.

Figure 2-9: x1 Clock Lines Used by GX Transmitter Channels for Non-Bonded Configuration



Note: (1) You can use the central clock divider as a local clock divider.

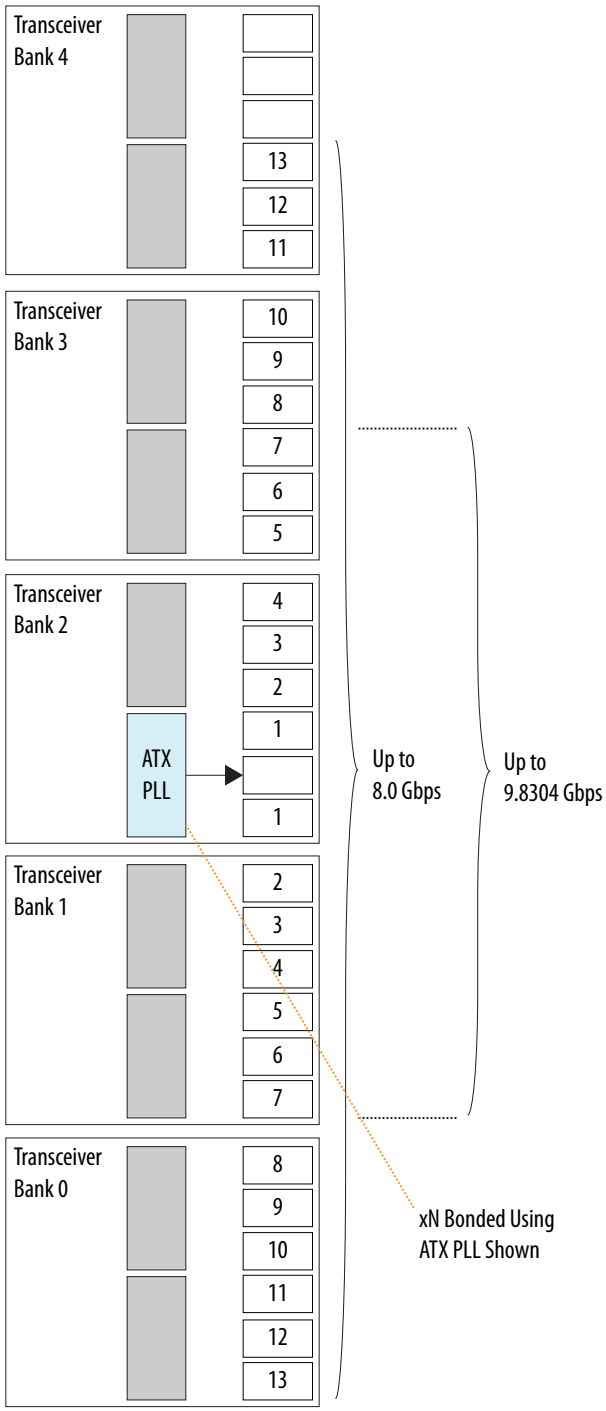
The x1 clock lines route the serial clock to the clock dividers of any channel within a transceiver bank. Refer to **Table 2-3** for details. The channel PLL, if configured as a CMU PLL, can drive the clock divider of its own channel, but you will not be able to use the channel PLL as a CDR. Without a CDR, you can use the channel only as a transmitter channel.

The x6 clock lines are used for bonded configurations within transceiver banks and PLL Feedback Compensation when bonding across multiple transceiver banks. The x6 clock lines are also used to route both the serial clock and parallel clock from the central clock dividers to the transceiver channels. When spanning across multiple transceiver banks, the xN clock lines can be used for both non-bonded configuration and bonded configurations.

The central clock dividers of channel 1 and channel 4 in a transceiver bank drive the x6 clock lines. The x6 clock lines then drive the xN clock lines. The xN clock lines used for both non-bonded and bonded configurations, span the entire side of the device and can provide the serial and parallel clock to contiguous channels within or outside a transceiver bank.

For both xN bonded and xN non-bonded configurations, the xN clock lines can support up to 13 contiguous channels above and up to 13 contiguous channels below the selected transmitter PLL which drives the central clock dividers of channel 1 or channel 4 of the same transceiver bank.

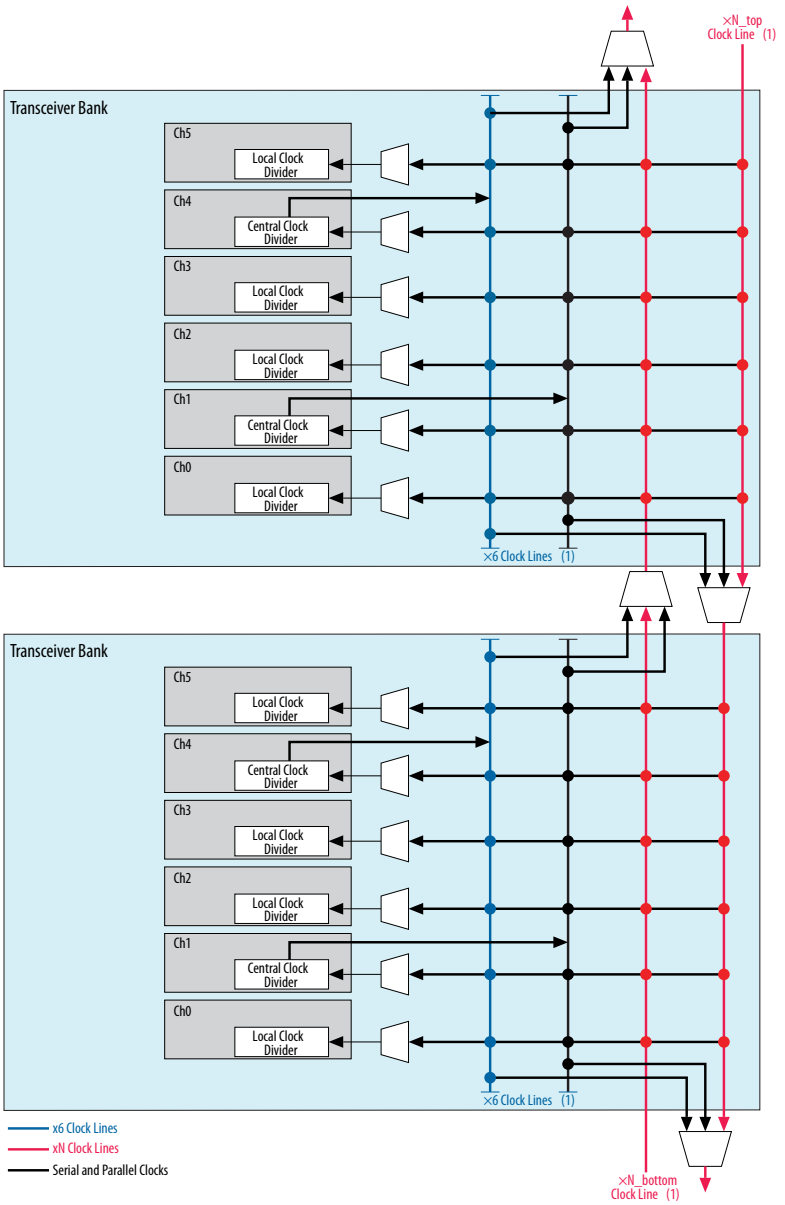
Figure 2-10: Channel Span for xN Bonded and Non Bonded Configurations



For xN bonded configurations, the channel where the central clock divider resides (channel 1 or 4) can be used as a data channel. Hence, a total of up to 27 contiguous data channels can be supported in the bonded configuration with the xN clock lines. However, for xN non-bonded configurations, the channel 1 or channel 4 of the transceiver bank where the central clock divider resides cannot be used as a data

channel since the parallel clock cannot be generated in this channel. Hence, a total of up to 26 contiguous data channels can be supported in the non-bonded configuration with the xN clock lines.

Figure 2-11: x6 and xN Clock Lines Used for Bonded Configurations



Note: (1) The clock lines carry both serial and parallel clocks.

Related Information

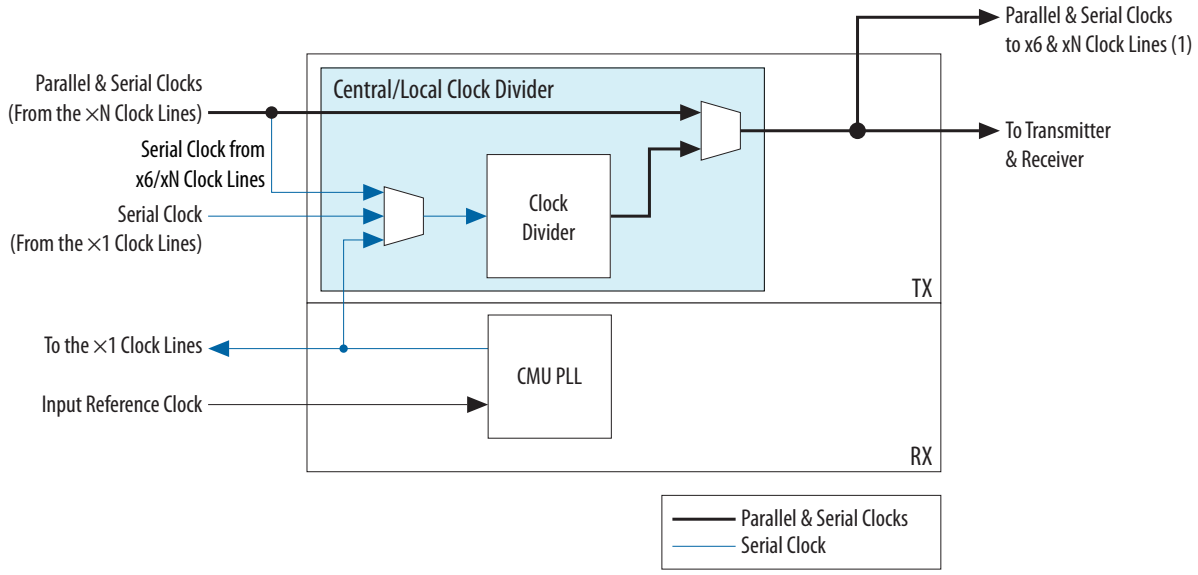
- Stratix V Device Datasheet
- Transceiver Configurations in Stratix V Devices
- Transceiver Architecture in Stratix V Devices

Clock Dividers

Each transmitter channel has a local clock divider. Some of the clock dividers that drive the x6 and xN clock lines are called central clock dividers.

Central clock dividers are located in channels 1 and 4 of the GX transceiver bank. The clock dividers generate the parallel and serial clock sources for the transmitter and optionally for the receiver PCS. The central clock dividers can feed the clock lines used to bond channels.

Figure 2-12: Clock Dividers



Note: (1) This is available only for the central clock divider in channels 1 and 4.

Note: For more information about clock dividers and the division factors supported, refer to the *Transceiver Architecture in Stratix V Devices* chapter.

Related Information

Transceiver Architecture in Stratix V Devices

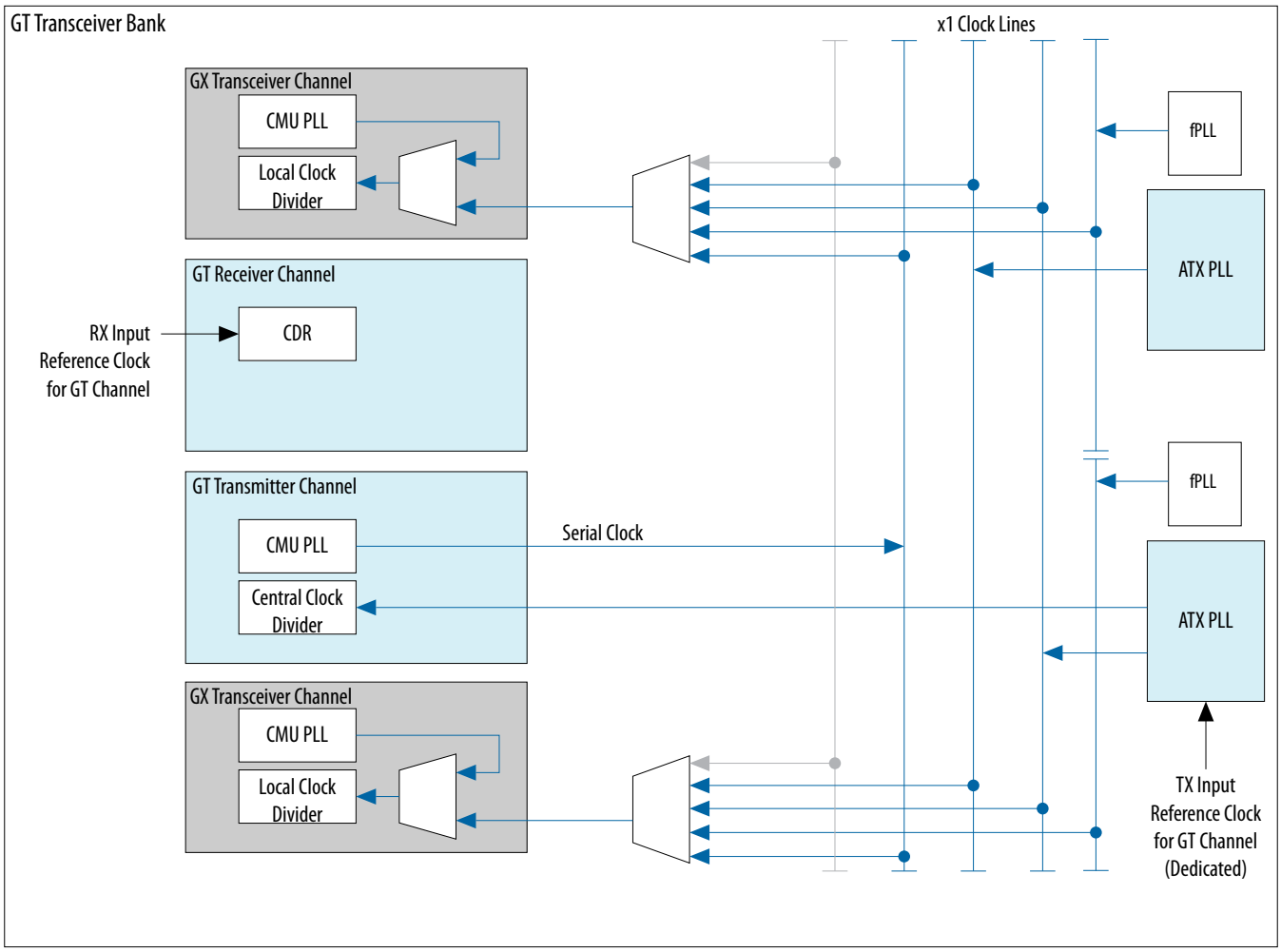
Transmitter Clock Network in Stratix V GT Transceiver Channels

The x1 clock lines in GT transceiver banks route the serial clock from the ATX PLL to the central clock divider of the GT transmitter channel. The x1 clock lines can also route the serial clock from the GT transmitter channel CMU PLL and the ATX PLLs to the local clock dividers of the GX transceiver channels.

Figure 2-13: x1 Clock Lines Used by Stratix V GT Transmitter Channels

The following figure shows the x1 clock lines used by a GT transceiver channel and two GX transceiver channels in a GT transceiver bank. There is one GT transmitter channel per GT transceiver bank. The transmitter channel must receive the clock from the bottom ATX PLL of a GT transceiver bank.

Note: The channel PLL in the GT receiver channel is always used as a CDR.

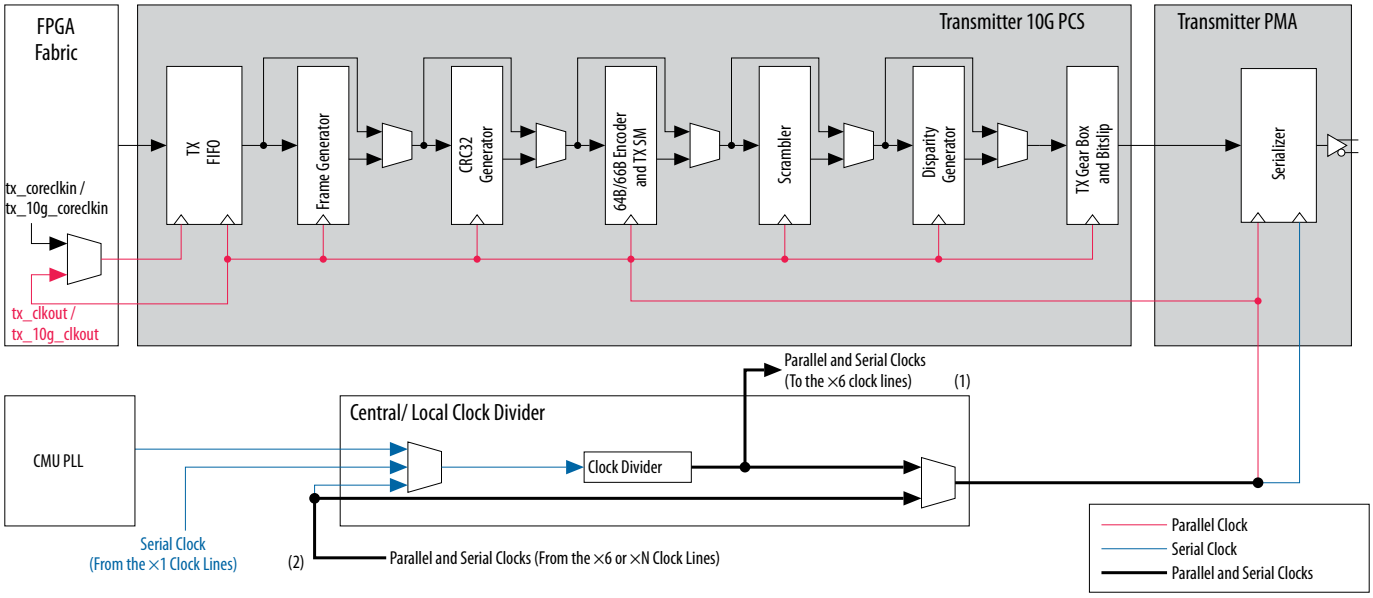


Transmitter Clocking

Transmitter clocking refers to the clocking architecture internal to the transmitter channel of a transceiver.

Transmitter 10G PCS Clocking

Figure 2-14: Transmitter 10G PCS and PMA Clocking

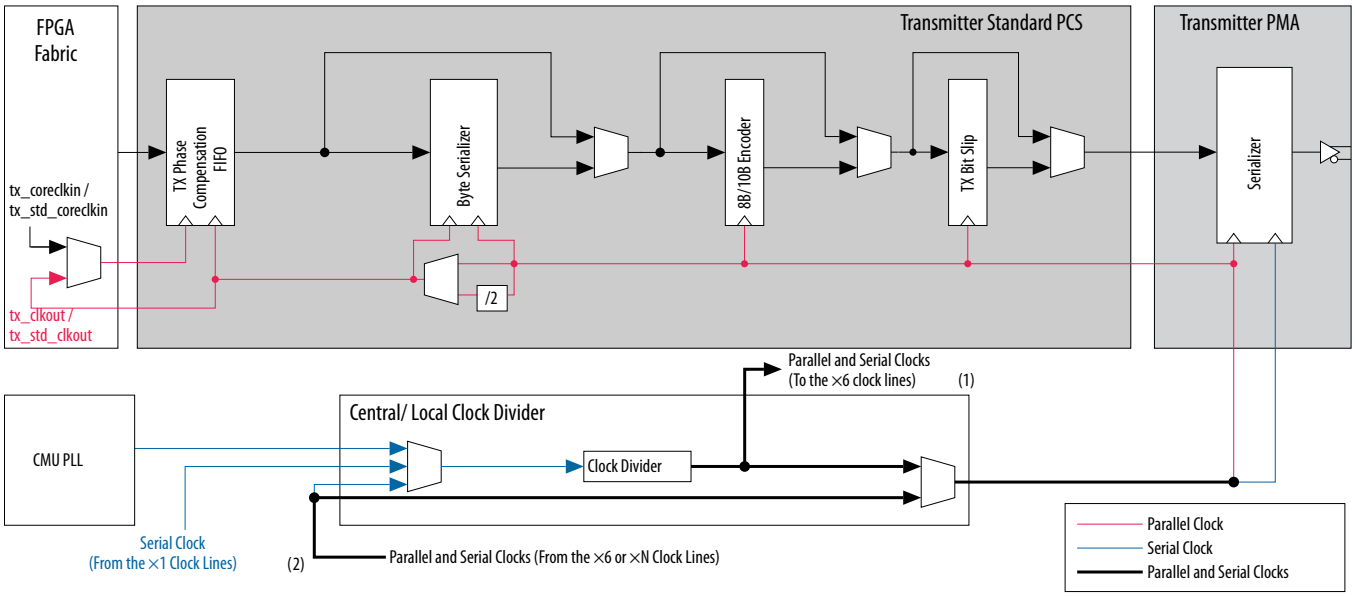


Notes: (1) Available only in the central clock dividers of channel 1 and channel 4 in a transceiver bank.
(2) x1 clock lines can be driven by a CMU PLL, ATX PLL, or a fractional PLL.

Transmitter Standard PCS Clocking

Figure 2-15: Transmitter Standard PCS and PMA Clocking

The clock divider block provides the serial clock to the serializer of the transmitter PMA and the parallel clock to the transmitter PCS.



Notes: (1) Available only in the central clock dividers of channel 1 and channel 4 in a transceiver bank.
(2) x1 clock lines can be driven by a CMU PLL, ATX PLL, or a fractional PLL.

In the 10G PCS channel, the parallel clock is used by all the blocks up to the read side of the transmitter (TX) FIFO.

In the standard PCS channel, the parallel clock is used by all the blocks up to the read side of the TX phase compensation FIFO in all configurations that do not use the byte serializer block. For configurations that use the byte serializer block, the clock is divided by a factor of two for the byte serializer and the read side of the TX phase compensation FIFO. The clock used to clock the read side of the TX phase compensation FIFO is also forwarded to the FPGA fabric to provide an interface between the FPGA fabric and the transceiver.

Note: For more information about clocking schemes used in different configurations, refer to the *Transceiver Configurations in Stratix V Devices* chapter.

Related Information

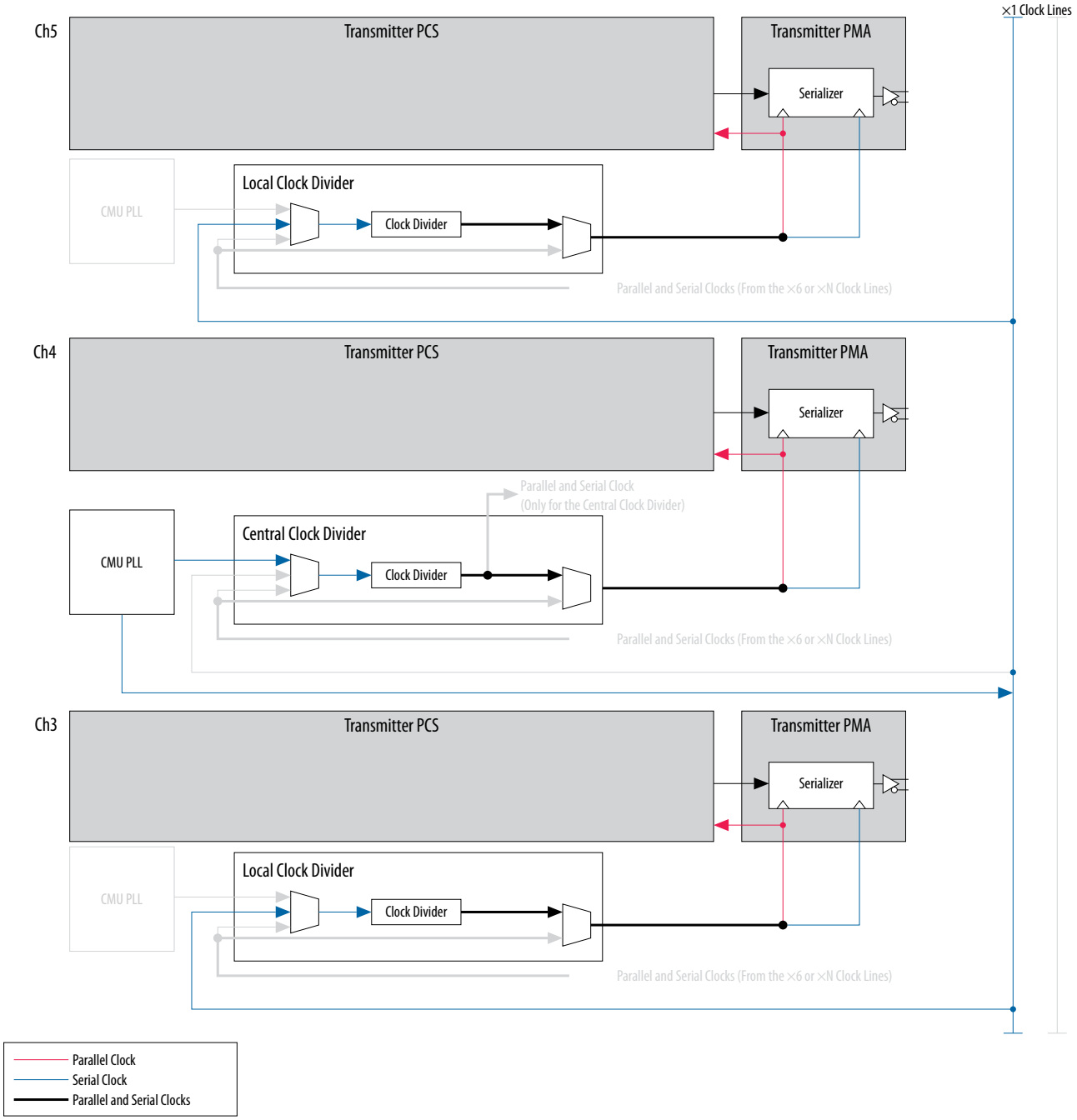
Transceiver Configurations in Stratix V Devices

Non-Bonded Channel Configurations Using the x1 Clock Network

In non-bonded channel configurations using the x1 clock networks, the parallel clock is generated by the clock divider of individual channels.

Figure 2-16: Three Transmitter-Only Channels Configured in x1 Non-Bonded Configuration

The figure shows three transmitter-only channels in non-bonded configuration driven by the channel PLL of channel 4 configured as a CMU PLL driving the x1 clock line. The clock divider block of each channel generates its own parallel clock by dividing the serial clock from the x1 clock line.

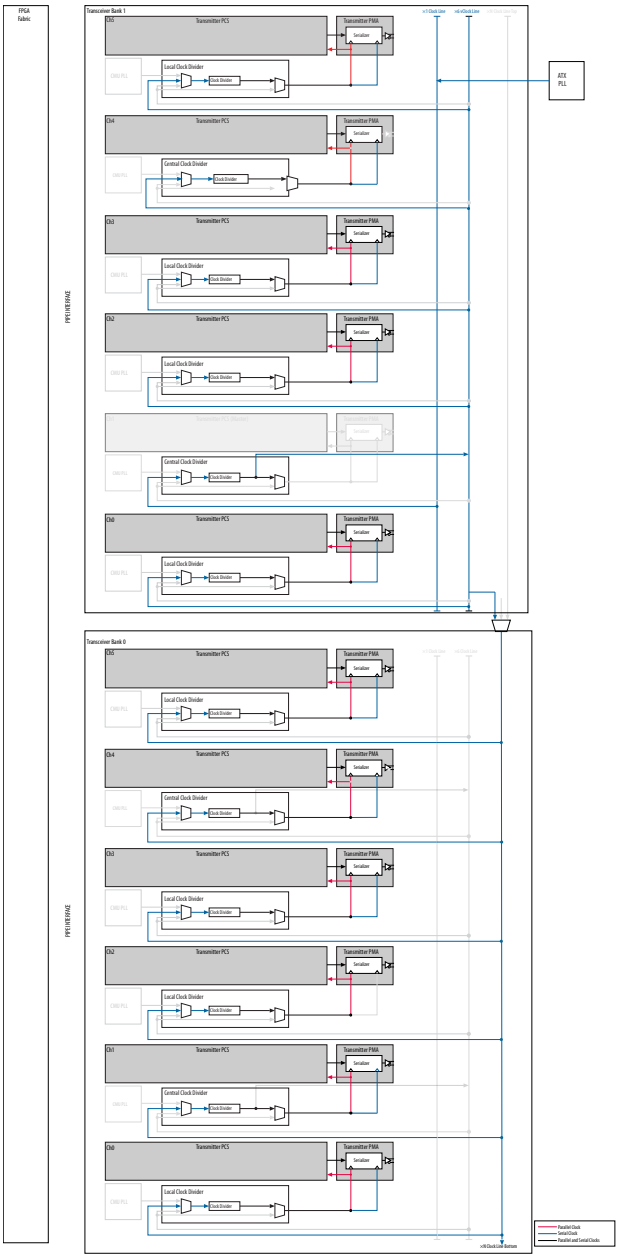


Non-Bonded Channel Configurations Using the xN Clock Network

In non-bonded channel configurations using the xN clock network, the parallel clock is generated by the clock divider of the individual channels.

Figure 2-17: Transmitter Channels in xN Non-Bonded Configuration

The figure shows 11 transmitter channels in non-bonded configuration. These channels are driven by the ATX PLL of the transceiver bank 1 which drives the x6 clock line through the central clock divider of the transceiver channel 1 in bank 1. The local clock divider block of each channel generates its own parallel clock by dividing the serial clock from the xN clock line. The channel where the central clock divider resides cannot generate the parallel clock and therefore it cannot be used as a data channel.



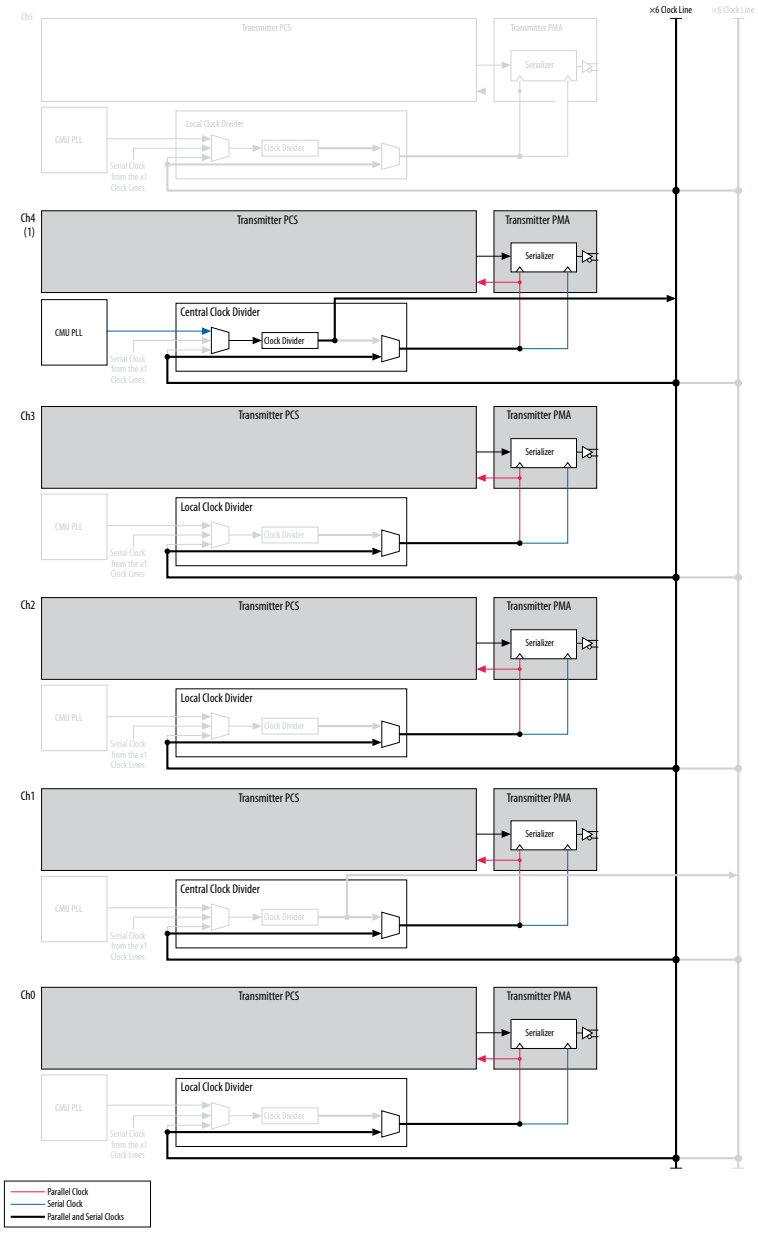
Bonded Channel Configurations

In bonded configurations, both the parallel clock and serial clock are sourced from either the x6 or xN clock line.

The central clock dividers source the serial clock from a transmitter PLL from the same transceiver bank using the x1 clock line. The central clock divider generates the parallel clock and drives both the serial clock and parallel clock on the x6 clock line, which can drive the xN clock line.

Figure 2-18: Five Transmitter-Only Channels Configured in Bonded Configuration

The figure shows five transmitter-only channels configured in a bonded configuration and driven by the channel PLL of channel 4 configured as a CMU PLL. The central clock divider of channel 4 generates a parallel clock and drives both the serial clock and parallel clock on the x6 clock line. All bonded channels source both serial and parallel clocks from the x6 clock line.



Note: (1) Channel 4 cannot be used as a receiver channel because the CMU PLL is used to generate clocks for other channels.

Related Information

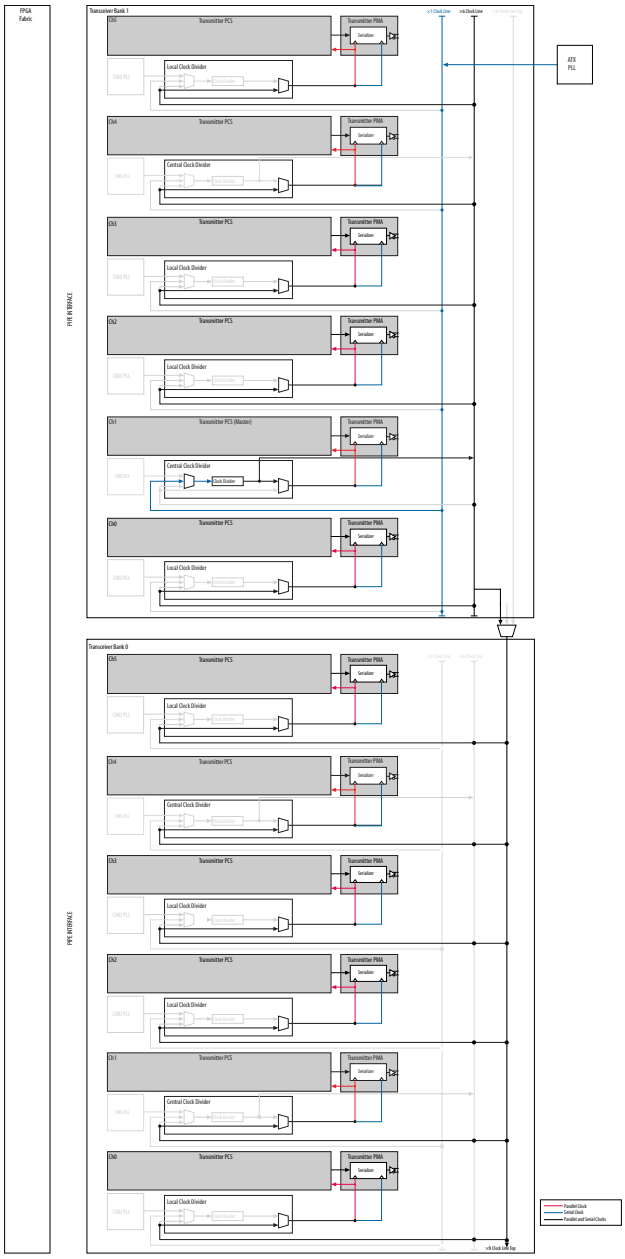
Transceiver Configurations in Stratix V Devices

For an example of using the xN clock lines, refer to the PCIe x8 configuration in the PCI Express (PCIe)—Gen1, Gen2, and Gen3 section of the *Transceiver Configurations in Stratix V Devices* chapter.

Bonded Channel Configurations Using the xN Clock Network

Figure 2-19: Transmitter Channels in xN Bonded Configuration

The following figure shows 12 transmitter channels configured in a bonded configuration and driven by the ATX PLL of the transceiver bank 1. The ATX PLL drives the central clock divider of channel 1 in transceiver bank 1. The central clock divider of channel 1 generates a parallel clock and drives both the serial and parallel clocks on the x6 clock line. All bonded channels in transceiver bank 1 source both the serial and parallel clocks from the x6 clock line. The x6 clock line in transceiver bank 1 also drives the xN clock lines in transceiver bank 0. All bonded channels in transceiver bank 0 source both the serial and parallel clocks from the xN clock line.



Bonded Channel Configurations Using the PLL Feedback Compensation Path

You can bond channels across multiple banks by using the PLL feedback compensation path.

The PLL feedback compensation path loops the parallel clock, which is used by the PCS blocks, back to the transmitter PLL. The PLL feedback compensation path synchronizes the parallel clock used to clock the PCS blocks in all transceiver banks with the `refclk`. You can use the PLL feedback compensation path to reduce channel-to-channel skew, which is introduced by the clock divider in each transceiver bank.

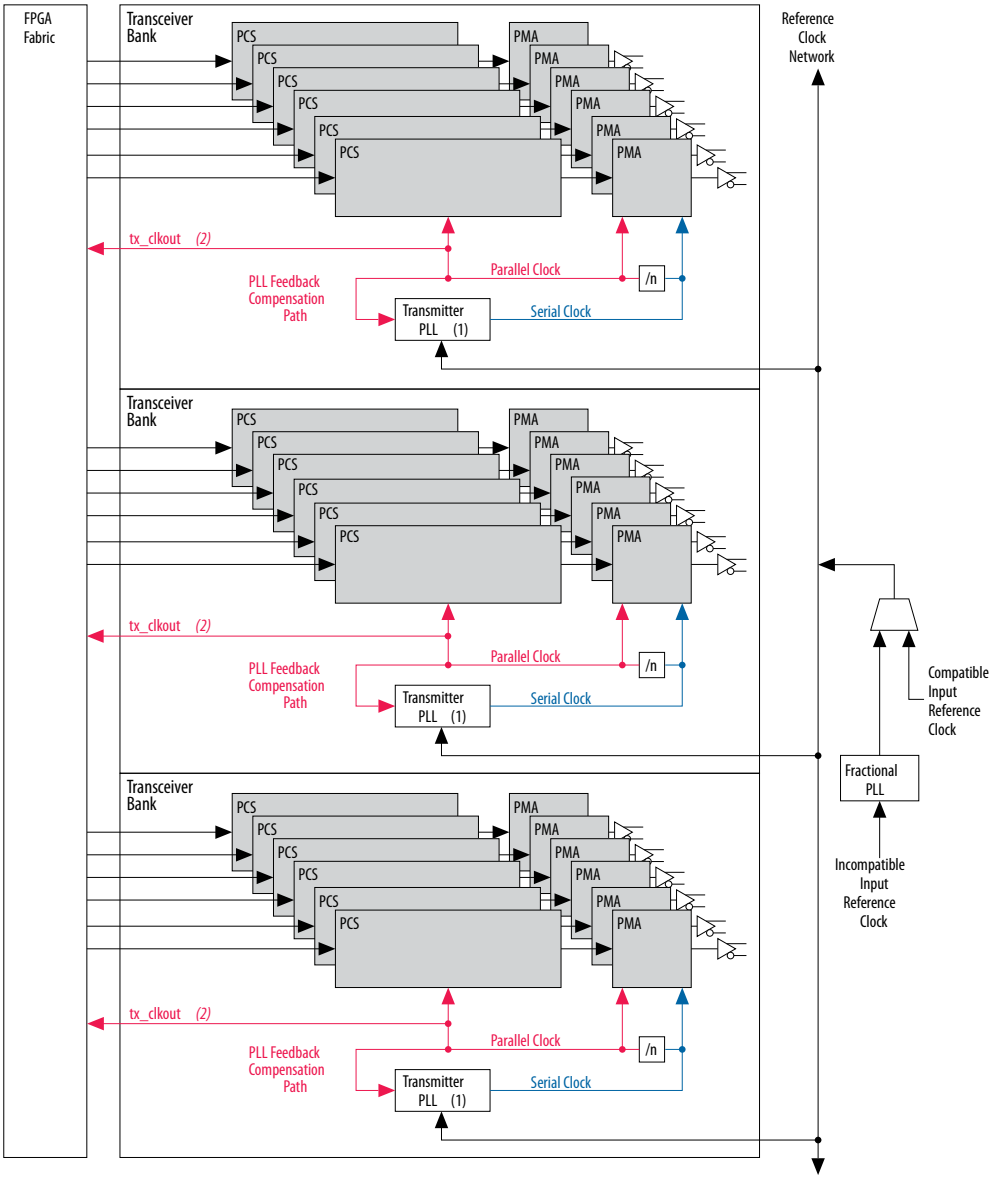
To bond channels using the PLL feedback compensation path, the input reference clock frequency used by the transmitter PLL must be the same as the parallel clock that clocks the PCS of the same channel. If the input reference clock frequency is not equal to the parallel clock frequency, use a fractional PLL to synthesize an input reference clock with the same frequency as the parallel clock.

Notes:

- fPLL does not support PLL feedback compensation when used as a TX PLL.
- Every transceiver bank with a bonded channel configured using the PLL feedback compensation path consumes a transmit PLL.

Figure 2-20: Three Transceiver Bank Channels Bonded Using the PLL Feedback Compensation Path

The figure shows 18 transmitter channels within three transceiver banks that are bonded using the PLL feedback compensation path. A fractional PLL is used to synthesize a clock that has the same frequency as the parallel clocks, which are looped back to the transmitter PLL.

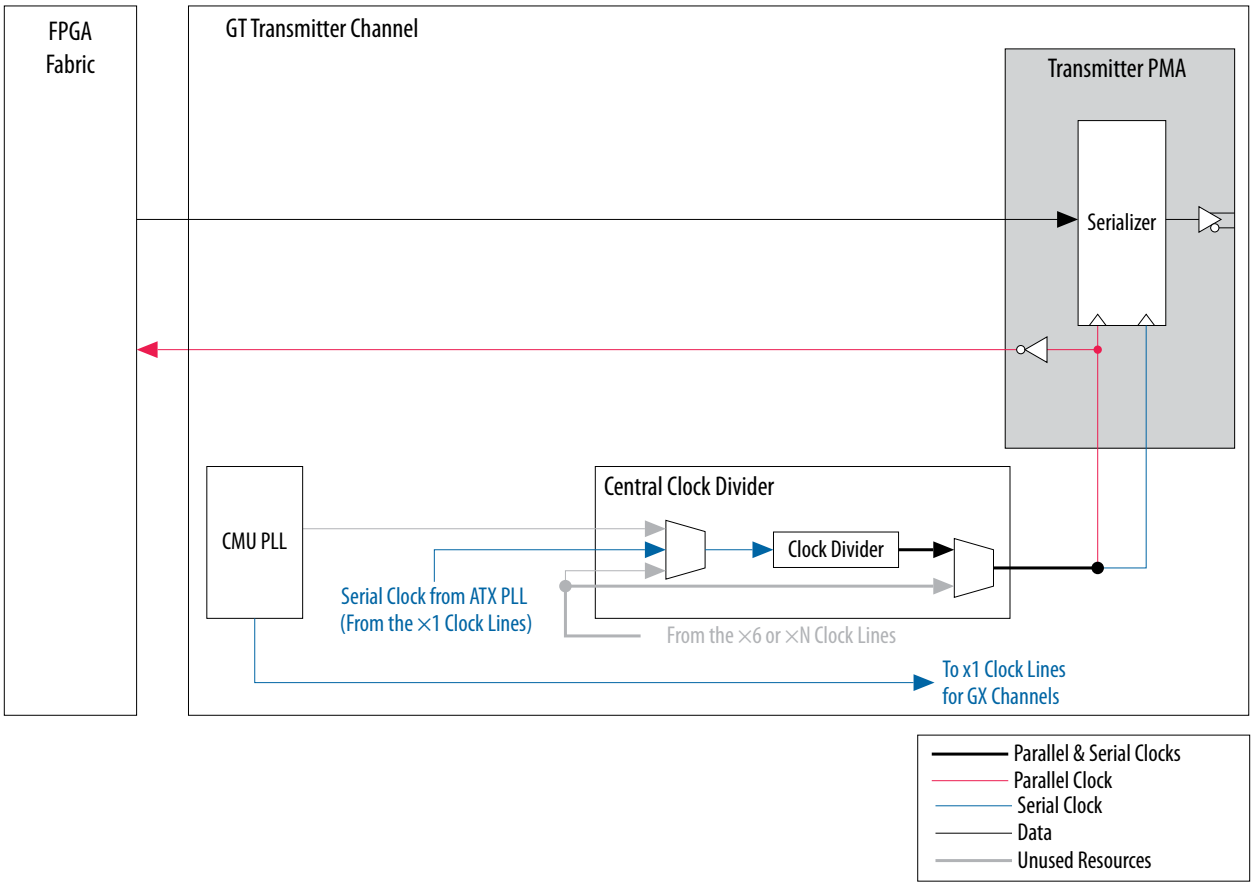


Notes: (1) The transmitter PLL can be an ATX PLL or a CMU PLL. You can have up to six channels per bank with an ATX PLL and five channels per bank with a CMU PLL.
(2) tx_clkout from any of the banks can be used with the FPGA fabric-transceiver interface for all the bonded channels.

Transmitter GT Channel Clocking

In a Stratix V GT transmitter channel the central clock divider block provides the serial and the parallel clocks to the serializer.

Figure 2-21: GT Channel Transmitter Clocking



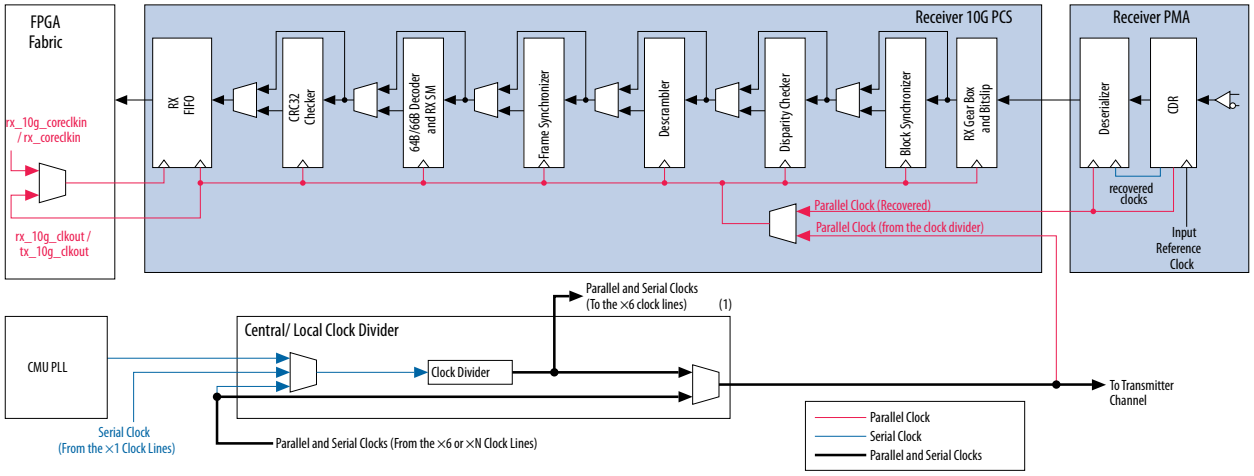
In a Stratix V GT transmitter channel, the parallel clock is forwarded to the FPGA fabric to provide an interface between the FPGA fabric and the transceiver. All PCS functions, such as encoding and bit slipping, must be implemented in the FPGA core.

Receiver Clocking

Receiver clocking refers to the internal clocking architecture of the receiver channel of a transceiver.

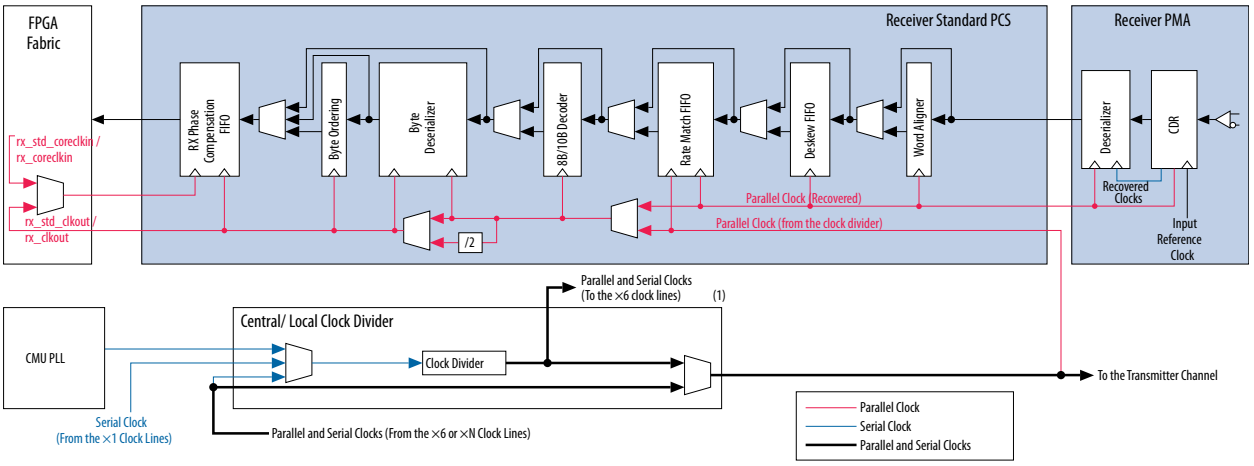
GX Channel Receiver Clocking

Figure 2-22: Receiver 10G PCS and PMA Clocking



Note: (1) Available only in the central clock dividers of channel 1 and channel 4 in a transceiver bank.

Figure 2-23: Receiver Standard PCS and PMA Clocking



Note: (1) Available only in the central clock dividers of channel 1 and channel 4 in a transceiver bank.

The CDR in the PMA of each channel recovers the serial clock from the incoming data. The CDR also divides the serial clock (recovered) to generate the parallel clock (recovered). Both clocks are used by the deserializer. 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 used by the transmitter PCS for that channel

Table 2-4: Clock Sources for All Receiver PCS Blocks

PCS	Block	Clock Source
Standard	Word aligner	Parallel clock (recovered)
	Rate match FIFO	Write side: parallel clock (recovered) Read side: parallel clock from the clock divider
	8B/10B decoder	If rate matcher is not used: parallel clock (recovered) If rate matcher is used: parallel clock from the clock divider
	Byte deserializer	Write side: • If rate matcher is not used: parallel clock (recovered) • If rate matcher is used: parallel clock from the clock divider Read side: 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 Side: Parallel clock (divided). This clock is also forwarded to the FPGA fabric Read Side: Clock sourced from the FPGA fabric
10G	All PCS blocks	Regular mode: parallel clock (recovered) Loopback mode: parallel clock from the clock divider[7]

Related Information

Transceiver Loopback Support in Stratix V Devices

Non-Bonded Channel Configurations

In non-bonded configurations, the receiver standard PCS requires both the parallel clock (recovered) and parallel clock from the clock divider.

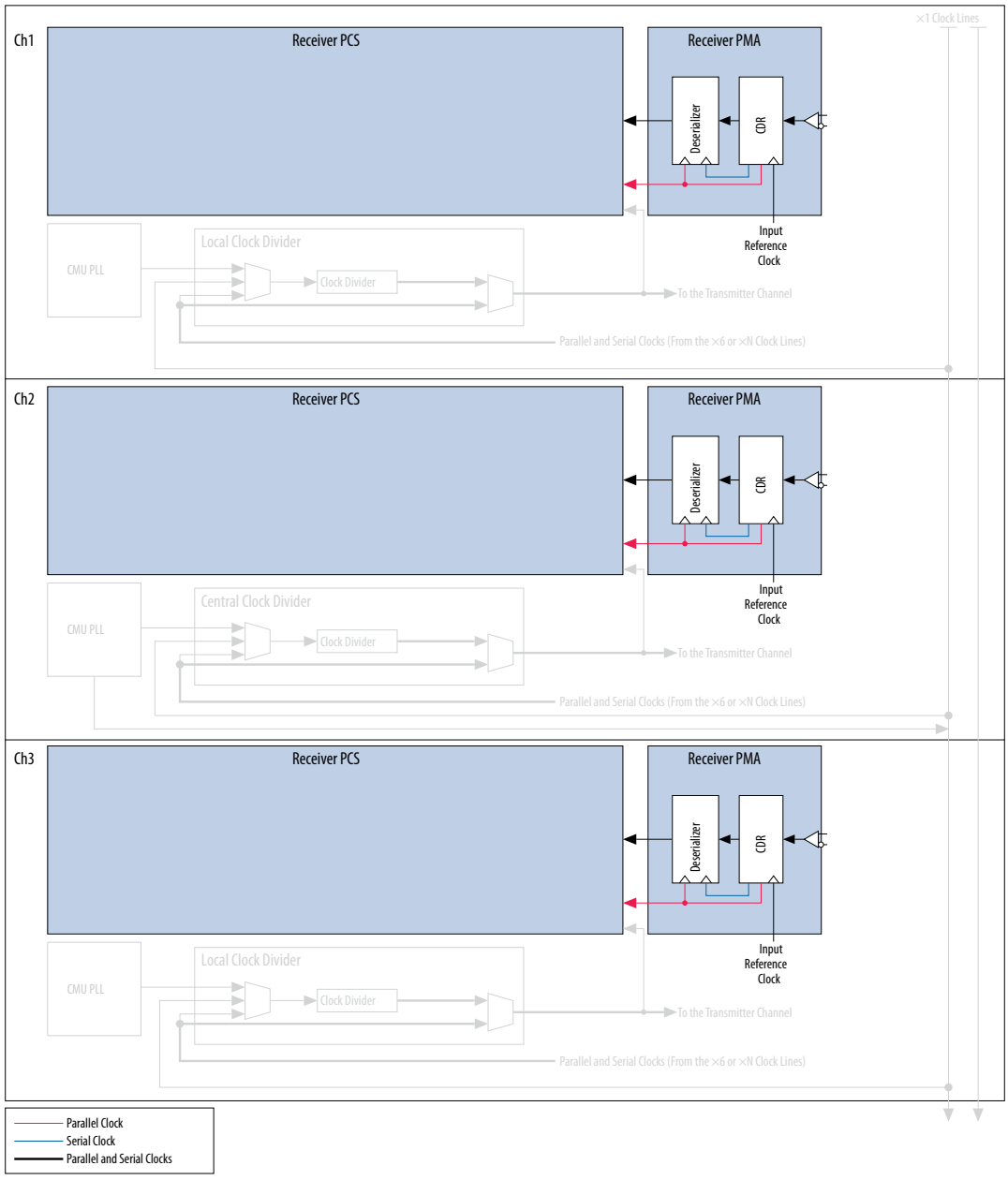
Depending on the configuration, the receiver PCS may require the parallel clock from the clock divider that is used for the transmitter PCS.

Note: In non-bonded configurations, the receiver 10G PCS uses only the parallel clock (recovered) for all its blocks.

[7] For more information about loopback mode, refer to the *Transceiver Loopback Support in Stratix V Devices* chapter.

Figure 2-24: Three Channels Configured in Non-Bonded Configuration

The figure shows three channels configured in non-bonded configuration. They use the receiver standard PCS that does not use a rate match FIFO. The CDR of each channel recovers the serial clock (recovered) from the incoming data and generates the parallel clock (recovered) by dividing the serial clock (recovered). Depending on the configuration, the receiver PCS may also use the parallel clock from the clock divider that is generated by the local clock divider for the transmitter.



Note: For more information about the clocking scheme used in different configurations, refer to the Transceiver Configurations in Stratix V Devices chapter.

Related Information

Transceiver Configurations in Stratix V Devices

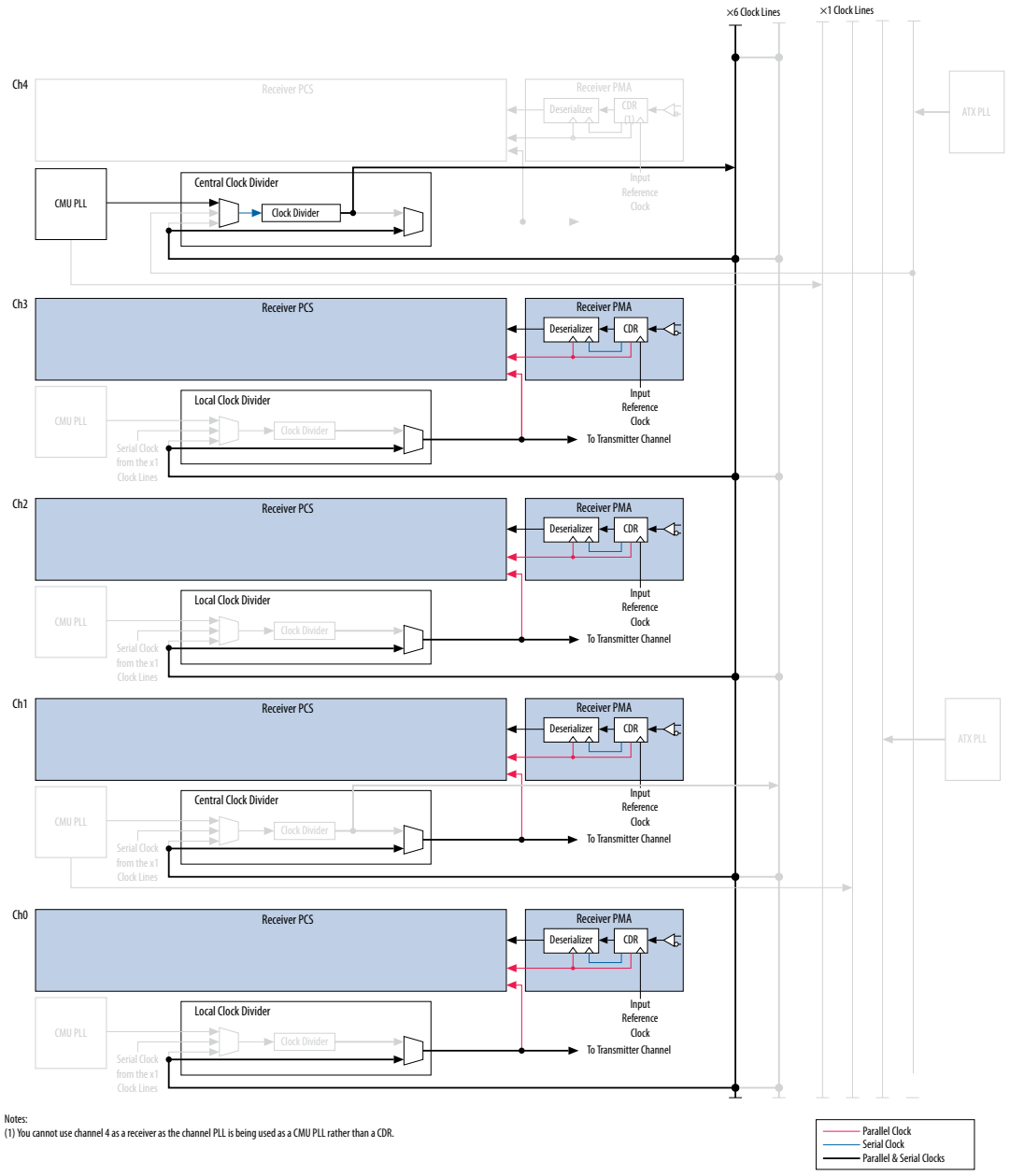
Bonded Channel Configurations

In bonded configurations, the receiver standard PCS requires both the parallel clock (recovered) and parallel clock from the clock divider.

In bonded configurations, the receiver 10G PCS uses only the parallel clock (recovered) for all its blocks.

Figure 2-25: Four Receiver Channels Configured in Bonded Duplex Configuration

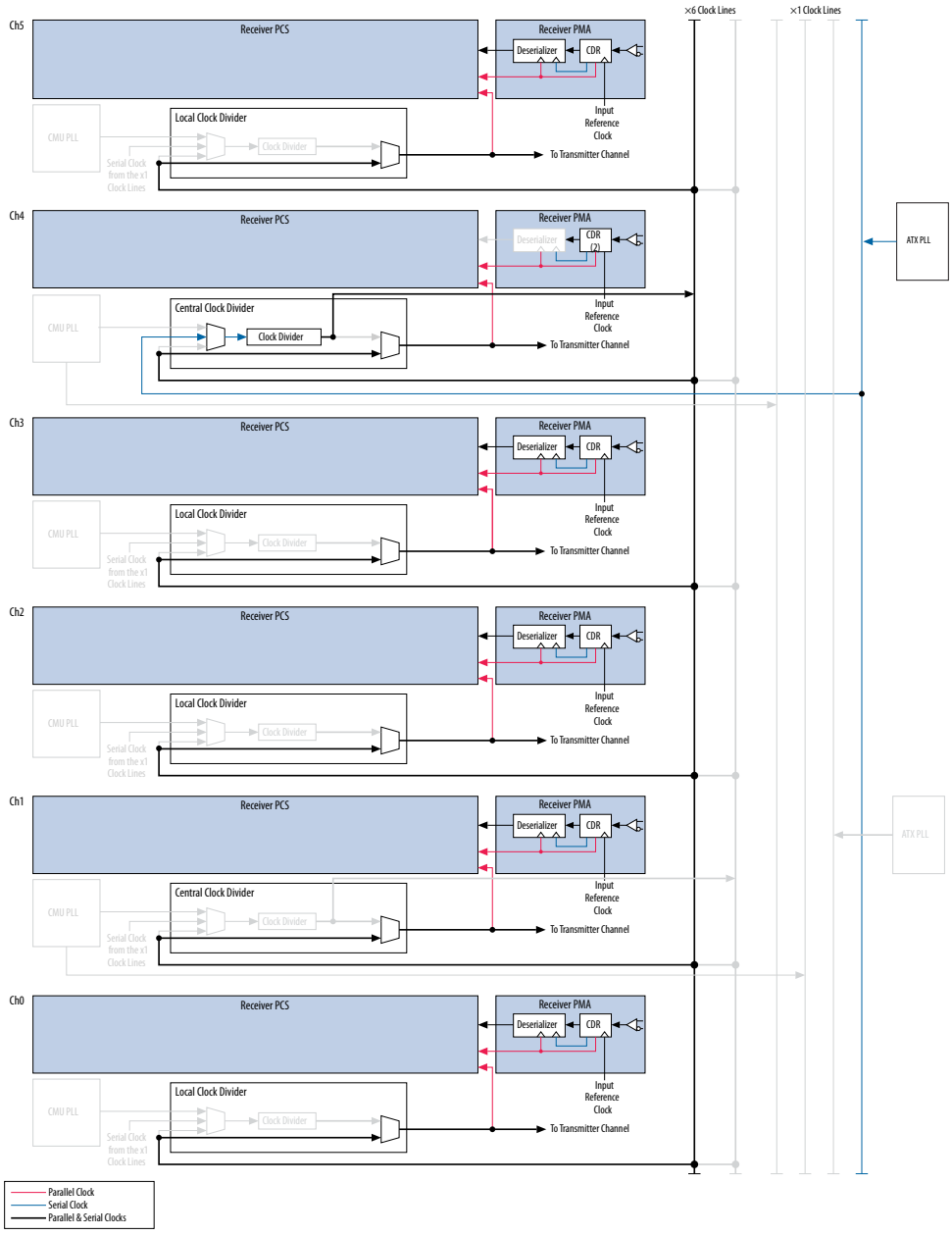
The figure shows four channels in a transceiver bank configured in bonded configuration, using the receiver standard PCS. The receiver PCS uses both the parallel clock (recovered) and parallel clock from the clock divider. The parallel clock from the clock divider is generated by the central clock divider for the transmitter PCS. It also drives some blocks in the receiver PCS, depending on the configuration you use.



The following shows all six channels in the transceiver bank in bonded configuration, as opposed to a maximum of four, shown in the previous figure. Six channel bonding is possible because the ATX PLL is used as a transmitter PLL instead of a channel PLL in the transceiver bank. Using the ATX PLL or fractional PLL allows you to use the channel PLLs of both channels 1 and 4 as CDRs to perform receiver operations.

Note: For more information about the clocking scheme used in different configurations, refer to the Transceiver Configurations in Stratix V Devices chapter.

Figure 2-26: Six Channels Configured in Bonded Configuration Using ATX PLL



Related Information

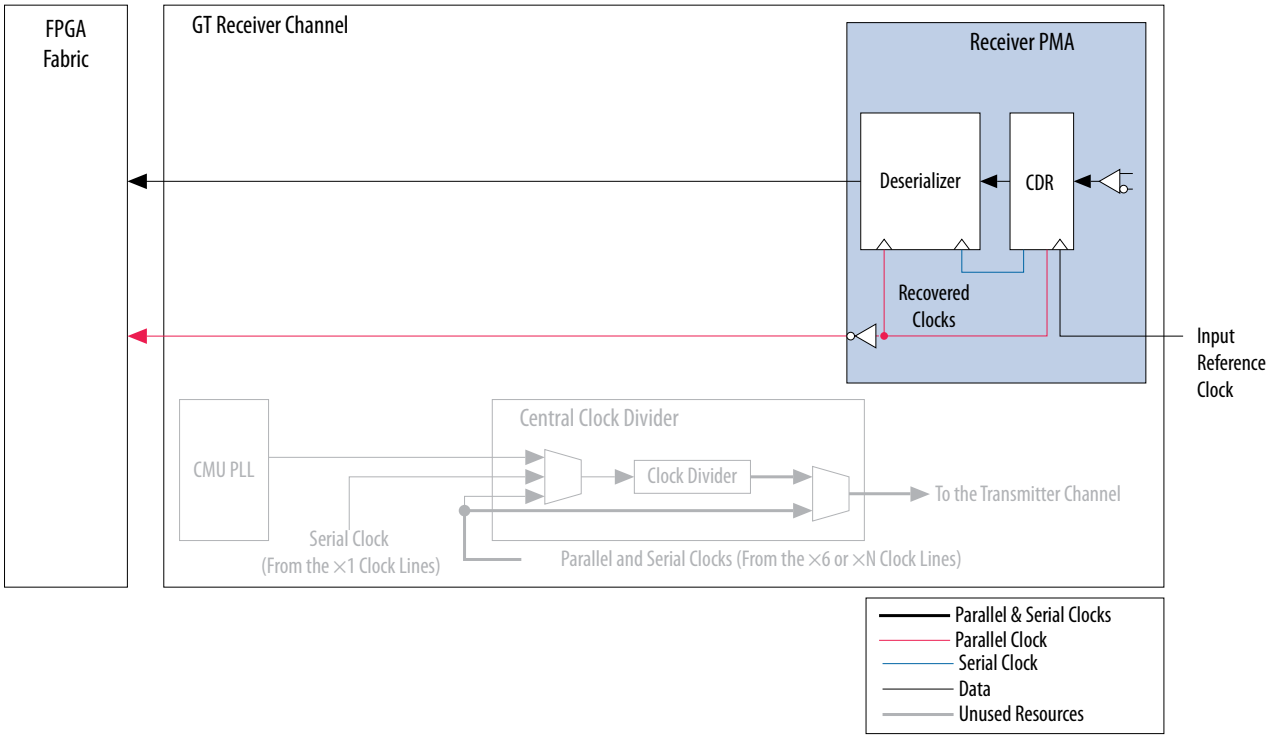
Transceiver Configurations in Stratix V Devices

GT Channel Receiver Clocking

The CDR in the PMA of the GT receiver channel recovers the serial clock from the incoming data and is driven by an input reference clock or clock from the reference clock network in the same GT transceiver bank.

The CDR also divides the serial clock (recovered) to generate the parallel clock (recovered). Both clocks are used by the deserializer. The parallel clock (recovered) is forwarded to the FPGA fabric to interface the FPGA fabric with the transceiver. All PCS functions, such as word alignment, rate matching, decoding, and byte ordering, must be implemented in the FPGA core because the PCS is unavailable in the GT receiver channel.

Figure 2-27: GT Channel Receiver Clocking



FPGA Fabric-Transceiver Interface Clocking

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—Refer to **Input Reference Clock Sources** on page 2-1. The input reference clock can be an FPGA fabric-transceiver interface clock when it is also forwarded to the FPGA fabric to clock the logic in the FPGA fabric.
- 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—Used to form a part of the FPGA fabric-transceiver interface clocks as follows:

  `phy_mgmt_clk`—Avalon®-MM interface clock used for controlling the transceivers, dynamic reconfiguration, and calibration

  `fixed_clk`—125 MHz fixed-rate clock used in the PCIe (PIPE) receiver detect circuitry

Table 2-5: FPGA Fabric–Transceiver Interface Clocks

Clock Name	Clock Description	Interface Direction	FPGA Fabric Clock Resource Utilization
pll_refclk, rx_cdr_refclk	A transceiver PMA TX PLL and CDR reference clock, sourced by dedicated differential pins of the device.	Input	GCLK, RCLK, PCLK
tx_clkout, tx_pma_clkout	Clock forwarded by the transceiver for clocking the transceiver datapath interface. The value of tx_clkout / tx_pma_clkout is derived by dividing the data rate by the serialization factor. For example, a 3 Gbps link with a serialization factor of 20 will result in a tx_clkout of 150 MHz.	Transceiver-to-FPGA fabric	
rx_clkout, rx_pma_clkout	Clock forwarded by the receiver for clocking the receiver datapath interface. The value of rx_clkout / rx_pma_clkout is derived by dividing the data rate by the deserialization factor. For example, a 10 Gbps link with a deserialization factor of 40 will result in a rx_clkout of 250 MHz.		
tx_10g_coreclkin/tx_std_coreclkin	User-selected clock for clocking the transmitter datapath interface	FPGA fabric-to-transceiver	
rx_10g_coreclkin / rx_std_coreclkin	User-selected clock for clocking the receiver datapath interface		
fixed_clk	PCIe receiver detect clock		
phy_mgmt_clk[8]	Avalon-MM interface management clock		

Note: You can forward the `pll_ref_clk`, `tx_clkout`, and `rx_clkout` clocks to a fractional PLL so that the fractional PLL can synthesize a clock for the FPGA logic. A second fractional PLL can be reached by periphery clocks, depending on your device and channel placement, and may require using a RGCLK or GCLK.

Table 2-6: Configuration Specific Port Names for tx_clkout and rx_clkout

Configuration	Port Name for tx_clkout	Port Name for rx_clkout
Custom	`tx_clkout`	`rx_clkout`
Native - 10G PCS	`tx_10g_clkout`	`rx_10g_clkout`
Native - Standard PCS	`tx_std_clkout`	`rx_std_clkout`
Native - PMA Direct	`tx_pma_clkout`	`rx_pma_clkout`
Interlaken	`tx_clkout`	`rx_clkout`
Low Latency	`tx_clkout`	`rx_clkout`
PCIe	`pipe_pclk`	`pipe_pclk`
XAUI	`xgmii_tx_clk`	`xgmii_rx_clk`

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

Related Information

Clock Networks and PLLs in Stratix V Devices

Transmitter Datapath Interface Clocking

The transmitter datapath interface consists of the following:

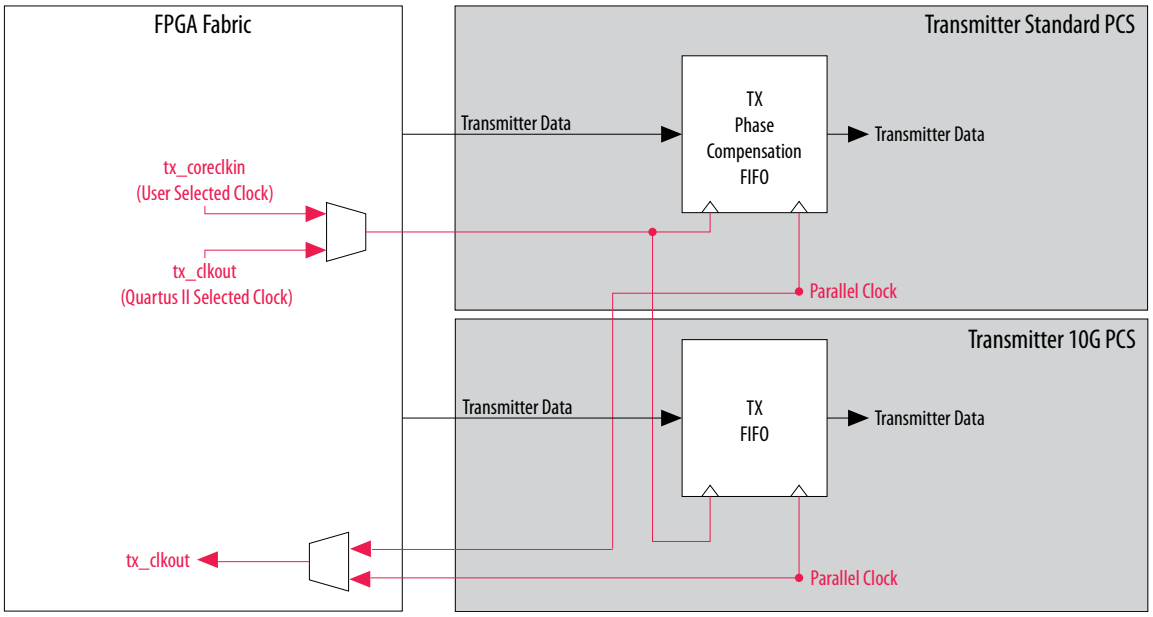
- Write side of the TX phase compensation FIFO—for configurations that use the standard PCS channel
- Write side of the TX FIFO—for configurations that use the 10G PCS channel

This interface is clocked by the transmitter datapath interface clock. The transmitter PCS forwards the following clocks to the FPGA fabric:

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

(8) The phy_mgmt_clk is a free-running clock that is not derived from the transceiver blocks, except if phy_mgmt_clk is derived from the dedicated refclk pin.

Figure 2-28: Transmitter Datapath Interface Clocking



All configurations using the standard PCS channel must have a 0 parts per million (ppm) difference between the transmitter datapath interface clock and the read side clock of the TX phase compensation FIFO.

Note: For more information about interface clocking for each configuration, refer to the Transceiver Configurations in Stratix V Devices chapter.

You can clock the transmitter datapath interface by using one of the following:

- Quartus II-selected transmitter datapath interface clock
- User-selected transmitter datapath interface clock

Note: User selection allows you to share the transceiver datapath interface clocks to reduce GCLK, RCLK, and PCLK resource utilization in your design.

Related Information

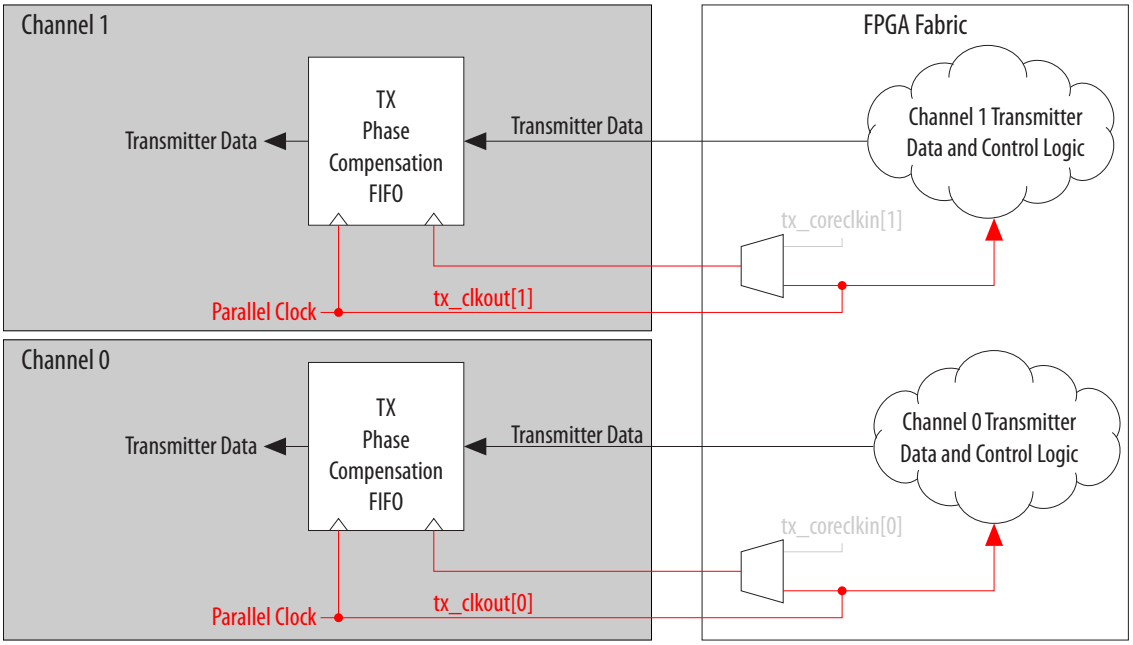
Transceiver Configurations in Stratix V Devices

Quartus II-Selected Transmitter Datapath Interface Clock

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

Figure 2-29: Transmitter Datapath Interface Clocking for Non-Bonded Channels

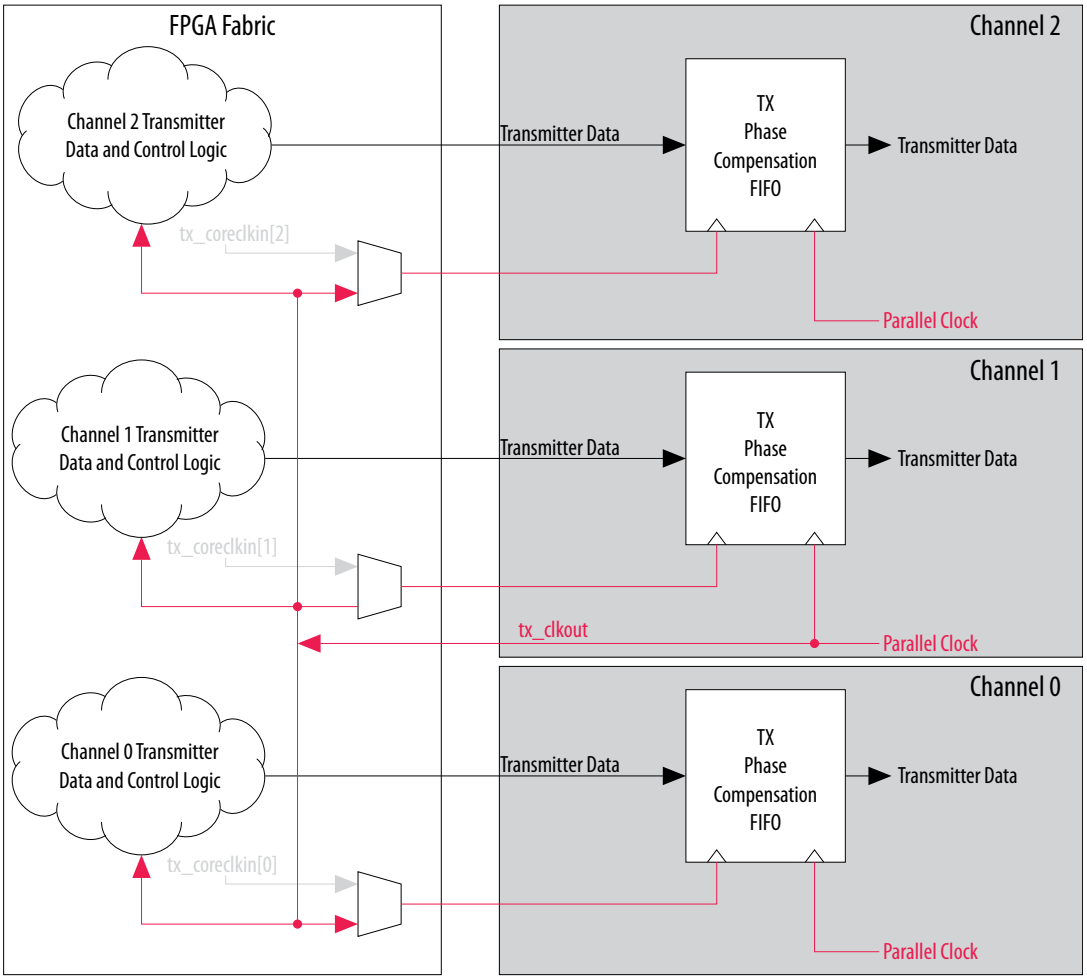
The figure shows the transmitter datapath interface of two non-bonded channels clocked by their respective transmitter PCS clocks that are forwarded to the FPGA fabric.



Note: The FPGA fabric-transceiver interface clocking for GT transmitter channels is similar to the non-bonded GX transmitter channel FPGA fabric-transceiver interface clocking.

Figure 2-30: Transmitter Datapath Interface Clocking for Three Bonded Channels

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



Selecting a Transmitter Datapath Interface Clock

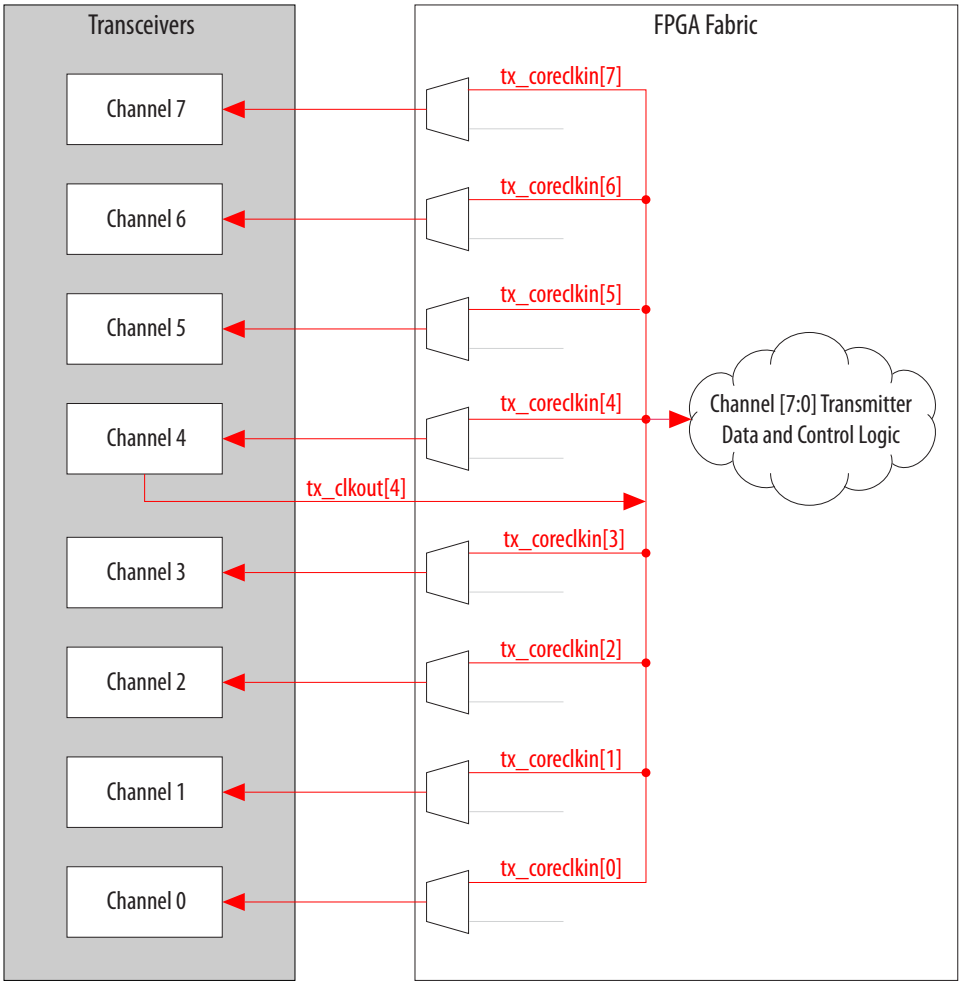
Multiple non-bonded transmitter channels use a large portion of GCLK, RCLK, and PCLK resources. Selecting a common clock driver for the transmitter datapath interface of all identical transmitter channels saves clock resources.

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 (VOD), transmitter common-mode voltage (VCM), or pre-emphasis.

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 eight identical channels clocked by a single clock (`tx_clkout` of channel 4).

Figure 2-31: Eight Identical Channels with a Single User-Selected Transmitter Interface Clock



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

1. Instantiate the `tx_coreclkin` port for all the identical transmitter channels (`tx_coreclkin[7:0]`).
2. Connect `tx_clkout[4]` to the `tx_coreclkin[7:0]` ports.
3. Connect `tx_clkout[4]` to the transmitter data and control logic for all eight channels.

Note: Resetting or powering down channel 4 causes a loss of the clock for all eight 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
- When there is 0 PPM between `refclk` and `tx_clkout`

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

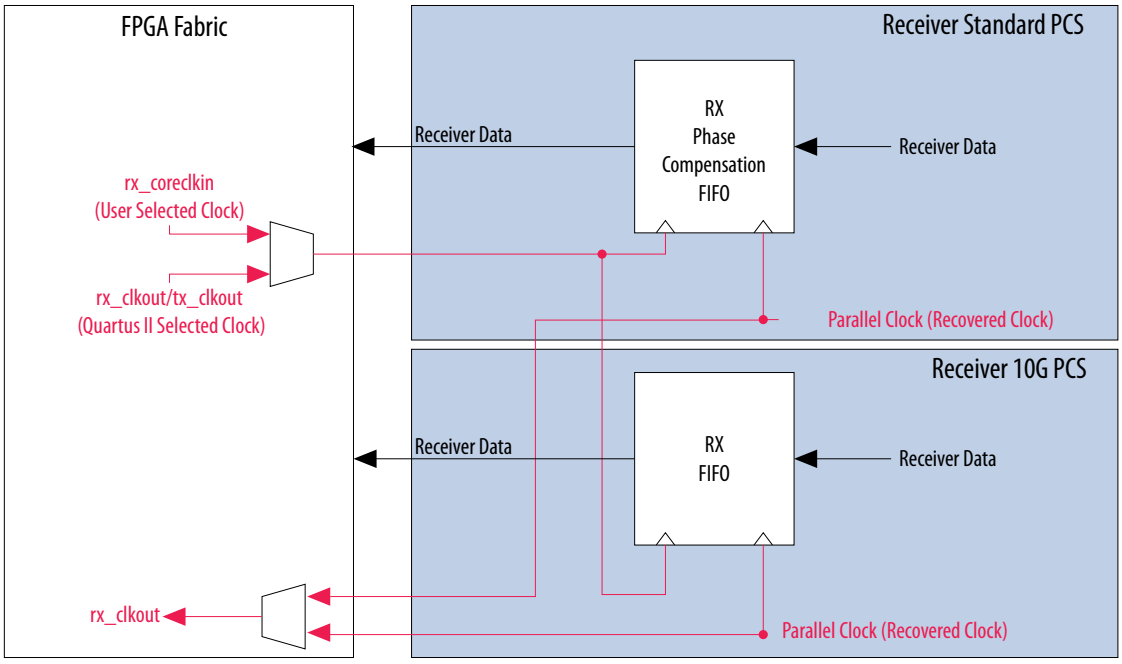
The receiver datapath interface consists of the following:

- Read side of the RX phase compensation FIFO—for configurations that use the standard PCS channel
- Read side of the RX FIFO—for configurations that use the 10G PCS channel

This interface is clocked by the receiver datapath interface clock. 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 2-32: Receiver Datapath Interface Clocking



All configurations that use the standard 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 clocking sections for each configuration in the Transceiver Configurations in Stratix V Devices chapter.

You can clock the receiver datapath interface by using one of the following:

- Quartus II-selected receiver datapath interface clock
- User-selected receiver datapath interface clock

Note: User-selection is provided to share the transceiver datapath interface clocks to reduce GCLK, RCLK, and PCLK resource utilization in your design.

Related Information

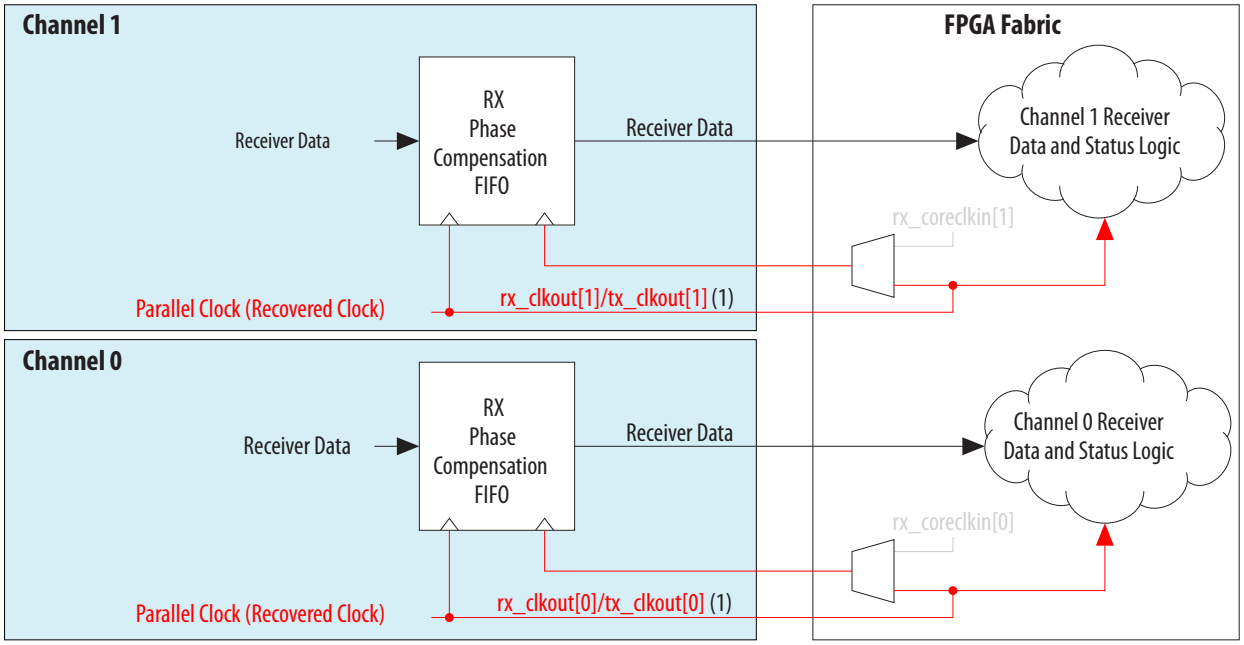
Transceiver Configurations in Stratix V Devices

Quartus II Software-Selected Receiver Datapath Interface Clock

The Quartus II software automatically picks the appropriate clock from the FPGA fabric to clock the receiver datapath interface.

Figure 2-33: Receiver Datapath Interface Clocking for Non-Bonded Channels

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

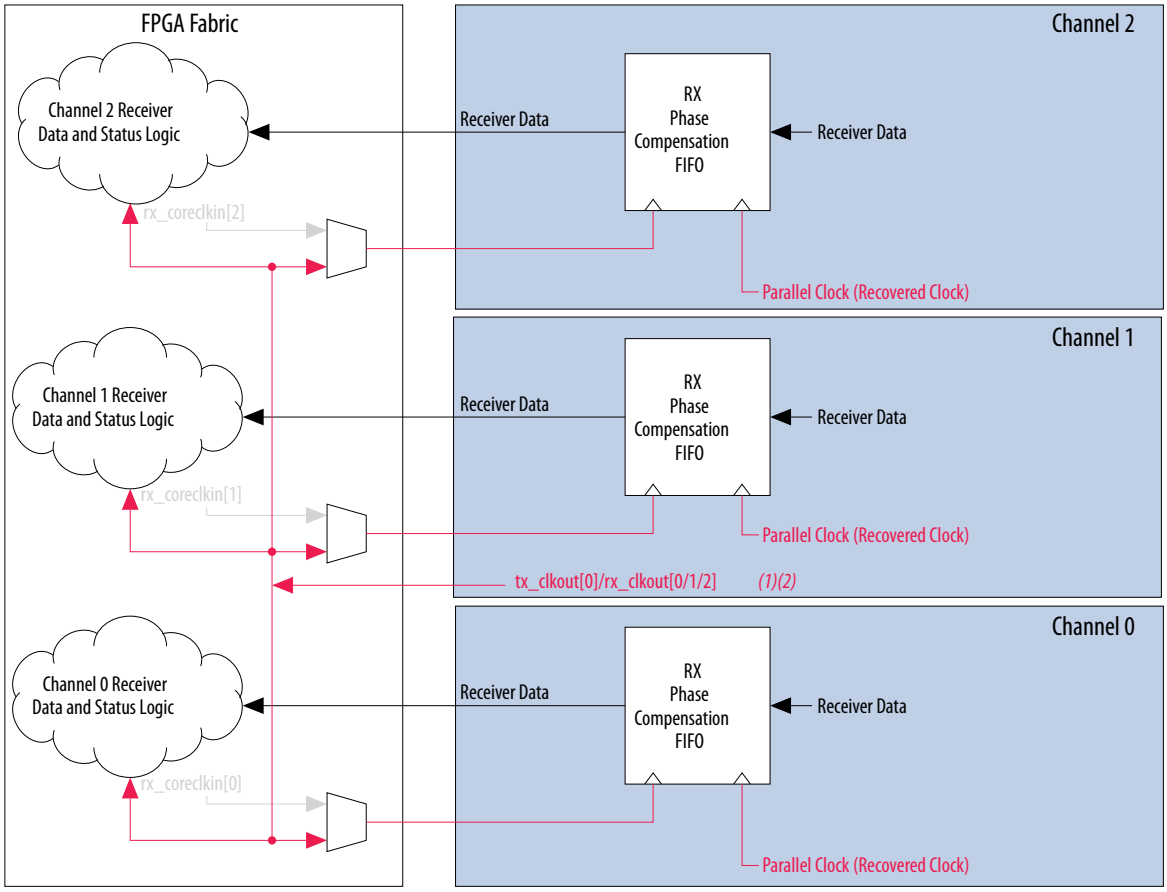


Note : (1) If you use a rate matcher, the tx_clkout clock is used.

Note: The FPGA fabric-transceiver interface clocking for GT receiver channels is similar to the non-bonded GX receiver channel FPGA fabric-transceiver interface clocking.

Figure 2-34: Receiver Datapath Interface Clocking for Three 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 in a transceiver bank.



Notes:
(1) tx_clkout[0] can only be used if a rate matcher is used.
(2) rx_clkout[0], rx_clkout[1] or rx_clkout[2] can be used to clock the read side of the rx phase compensation FIFO of all channels only if there is 0 ppm difference between the rx_clkout[0], rx_clkout[1], and rx_clkout[2].

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.

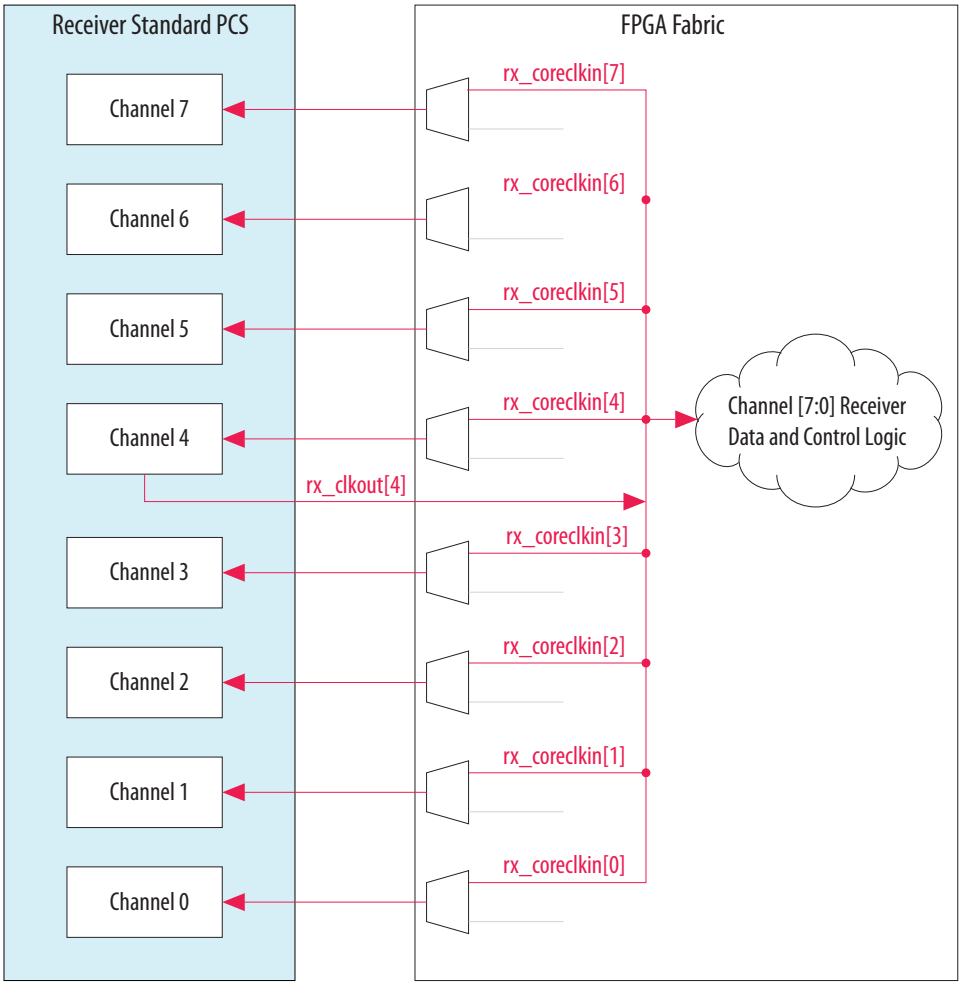
Note: Identical receiver channels are defined as channels that have the same input reference clock source for the CDR and the same receiver PMA and PCS configuration. Identical receiver channels need to be PPM aligned with regards to their remote transmitters. These channels may have different analog 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 eight identical channels that are clocked by a single clock (`rx_clkout` of channel 4).

Figure 2-35: Eight Identical Channels with a Single User-Selected Receiver Interface Clock



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

- Instantiate the `rx_coreclkin` port for all the identical receiver channels (`rx_coreclkin[7:0]`).
- Connect `rx_clkout[4]` to the `rx_coreclkin[7:0]` ports.
- Connect `rx_clkout[4]` to the receiver data and control logic for all eight channels.

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

GXB 0 PPM Core Clock Assignment

The common clock should have a 0 PPM difference with respect to the read side of the TX FIFO (in the 10G PCS channel) or TX phase compensation FIFO (in the Standard PCS channel) 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.

The 0 PPM common clock driver can be driven by one of the following sources:

- `tx_clkout` in non-bonded channel configurations
- `tx_clkout[0]` in bonded channel configurations
- `rx_clkout` in non-bonded channel configurations
- `refclk` when there is 0 PPM difference between `refclk` and `tx_clkout`

Table 2-7: 0 PPM Core Clock Settings

The following table lists the 0 PPM core clock settings that you make in the Quartus II Assignment Editor.

Assignments[9]	Description
To	`tx_dataout/rx_datain pins` of all channels whose `tx/rx_coreclk` ports are connected together and driven by the 0 PPM clock driver.
Assignment Name	0 PPM coreclk setting
Value	ON

Note: For more information about QSF assignments and how 0 PPM is used with various transceiver PHYs, refer to the *Altera Transceiver PHY IP Core User Guide*.

Related Information

Altera Transceiver PHY IP Core User Guide

Document Revision History

The table below lists the revision history for this chapter.

[9] You can find the full hierarchy name of the 0 PPM clock driver using the Node Finder feature in the Quartus II Assignment Editor.

Table 2-8: Document Revision History

Date	Version	Changes
September 2014	2014.09.30	• Modifed *Figure: Four Receiver Channels Configured in Bonded Duplex Configuration* in " Bonded Channel Configurations" section to indicate that CMU PLL can support only four channels in a bonded configuration. • Modified *Figure: Input Reference Clock Sources to Transmit PLLs and CDR* and *Figure: Fractional PLL Input Clock Sources*. The reference clock network can be used as an input reference clock source and the dedicated reference clock pins feed the reference clock network. • Modified *Figure: Transmitter Datapath Interface Clocking*. The tx_clkout multiplexer is implemented in the FPGA fabric and is not present in the PCS. • Modified *Figure: Receiver Datapath Interface Clocking*. The rx_clkout multiplexer is implemented in the FPGA fabric and is not present in the PCS. • 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. • Modified the definition of identical receiver channels in *Selecting a Receiver Datapath Interface Clock* section.
October 2013	2013.10.11	• Updated "Dedicated refclk Pins" section.
May 2013	2013.05.06	• Updated for Quartus II software version 13.0 feature support. • Added table "Electrical Specifications for the Input Reference Clock". • Added figure " Termination Scheme for a Reference Clock Signal When Configured as HCSL". • Updated table " Data Rates and Spans Supported Using Stratix V Clock Sources and Clock Networks". • Added information and figures for bonded and non-bonded channel configurations using the xN clock network. • Added link to the known document issues in the Knowledge Base.

Date	Version	Changes
December 2012	2012.12.17	• Reorganized content and updated template. • Updated for the Quartus II software version 12.1. • Updated Figures 2-2, 2-3, 2-4, 2-5, 2-7, 2-8, 2-11, 2-21, and 2-26. • Added the “RX Pins Using the Reference Clock Network” section. • Updated table "Data Rates and Spans Supported Using Stratix V Clock Sources and Clock Networks." • Updated table "FPGA Fabric-Transceiver Interface Clocks" to address FB #60881. • Updated additional info in table "FPGA Fabric-Transceiver Interface Clocks" to address FB #65061. • Updated table "Configuration Specific Port Names for tx_clkout and rx_clkout."
June 2012	1.6	• Updated for the Quartus II software version 12.0 and reordered paragraphs. • Added Clock Divider section previously located in Architecture chapter. • Added information about GXB 0 PPM core clock assignment. • Updated Figures 2-4, 2-7, 2-17, 2-21, 2-23, and 2-30.
February 2012	1.5	• Updated document and figures for clarity. • Edited Figures 2-2, 2-3, and 2-4.
December 2011	1.4	• Updated document and figures for clarity. • Changed path for serial and parallel clocks in channel 4 in Figure 2–14, Figure 2–20, and Figure 2–21.
November 2011	1.3	• Added information about GT transceivers. • Added information about bonding channels across transceiver banks by using the PLL feedback compensation path. • x8 bonding using the xN clock lines is now available for PCIe Gen3. • Added information about the transceiver clocks used in the FPGA fabric. • Added information about fractional mode when using fractional PLLs. • Added information about using the FPGA fabric clocks as a reference clock to GX transceiver channels. • Added information about forwarding transceiver clocks to a fractional PLL so that the fractional PLL can synthesize a clock for the FPGA logic.

Date	Version	Changes
May 2011	1.2	• Added information about fractional PLLs as they provide an input reference clock in "Input Reference Clocking." • Chapter moved to Volume 3.
December 2010	1.1	• Updated clock names. • Updated figures for more accurate depiction of transceiver clocking. • Added information about ATX PLLs.
July 2010	1.0	Initial release.

Transceiver Reset Control in Stratix V Devices

3

2014.09.30

SV52004 Subscribe Send Feedback

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.

The Stratix V transceiver reset sequence is mandatory to initialize the physical coding sublayer (PCS) and physical medium attachment (PMA) blocks. Multiple reset options are available to reset the analog and digital portions of the transmitter and receiver.

Altera provides an embedded reset controller, but you can also provide your own user-coded reset controller.

Table 3-1: Stratix V Reset Control Options

Transceiver PHY IP Core	Embedded Reset Controller	User-Coded Reset Controller	Transceiver PHY Reset Controller IP	Avalon Memory-Mapped Reset Registers
XAUI	Yes			Yes
PCI Express	Yes			Yes
10GBASE-R	Yes	Yes	Yes	Yes
Interlaken	Yes			Yes
Custom Configuration	Yes	Yes	Yes	Yes
Low Latency	Yes	Yes	Yes	Yes
Deterministic Latency	Yes	Yes	Yes	Yes
Native PHY		Yes	Yes	

Related Information

Stratix V Device Handbook: Known Issues
Lists the planned updates to the *Stratix 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: You must have a valid and stable ATX PLL reference clock before deasserting the `phy_mgmt_clk_reset` and `mgmt_rst_reset` signals for successful ATX PLL 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

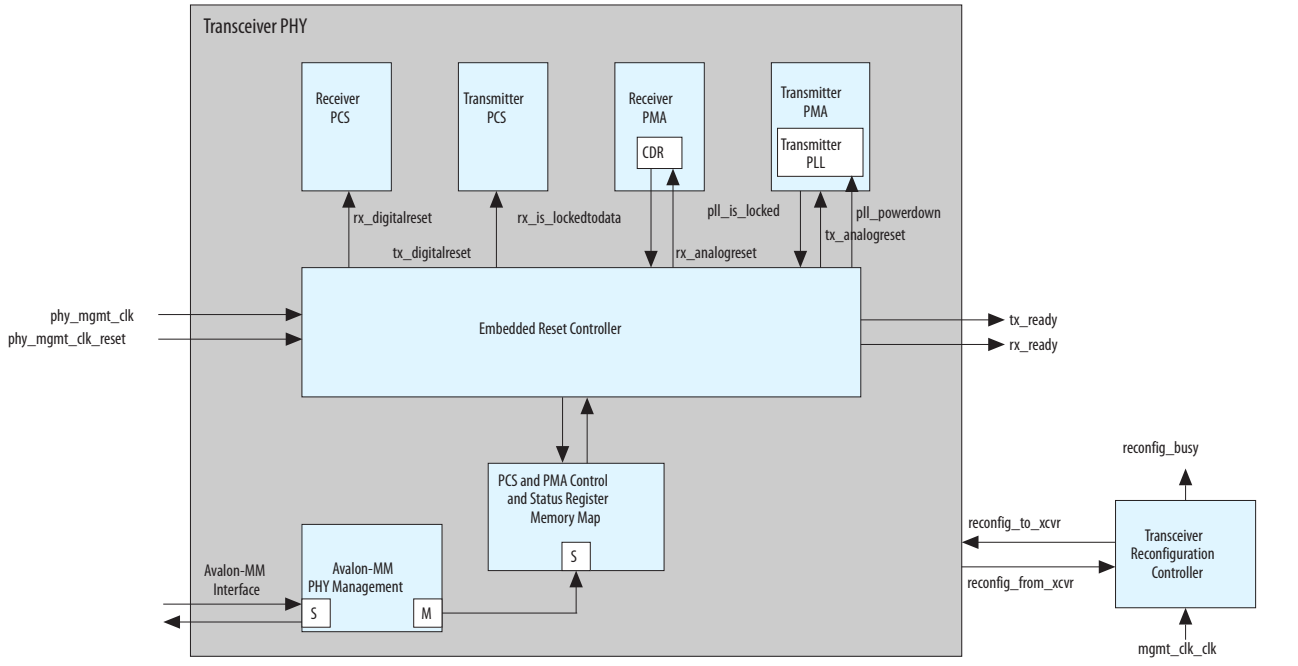


Table 3-2: Embedded Reset Controller Reset Control and Status Signals

Signal Name	Signal	Description
`phy_mgmt_clk`	Control Input	Clock for the embedded reset controller.

Signal Name	Signal	Description
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.

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

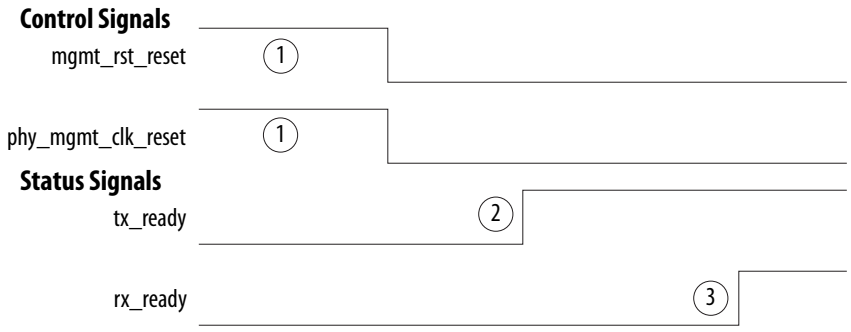
Follow this reset sequence to ensure a reliable link initialization after the initial power-up.

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

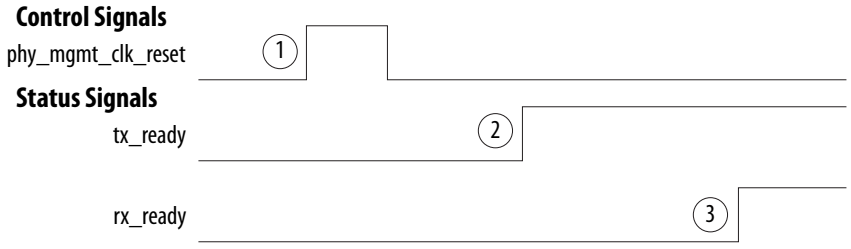
Follow this reset sequence to reset the entire transceiver at any point during the device operation, to re-establishing a link, or after certain dynamic reconfigurations.

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.

Related Information

- **Transceiver Reset Using Avalon Memory Map Registers** on page 3-11
- **Transceiver Reset Control Signals Using Avalon Memory Map Registers** on page 3-11

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

Note: You must have a valid and stable ATX PLL reference clock before deasserting the `pll_powerdown` and `mgmt_rst_reset` signals for successful ATX PLL calibration.

This reset controller comes with a clear text Verilog file that you modify based on your requirements.

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

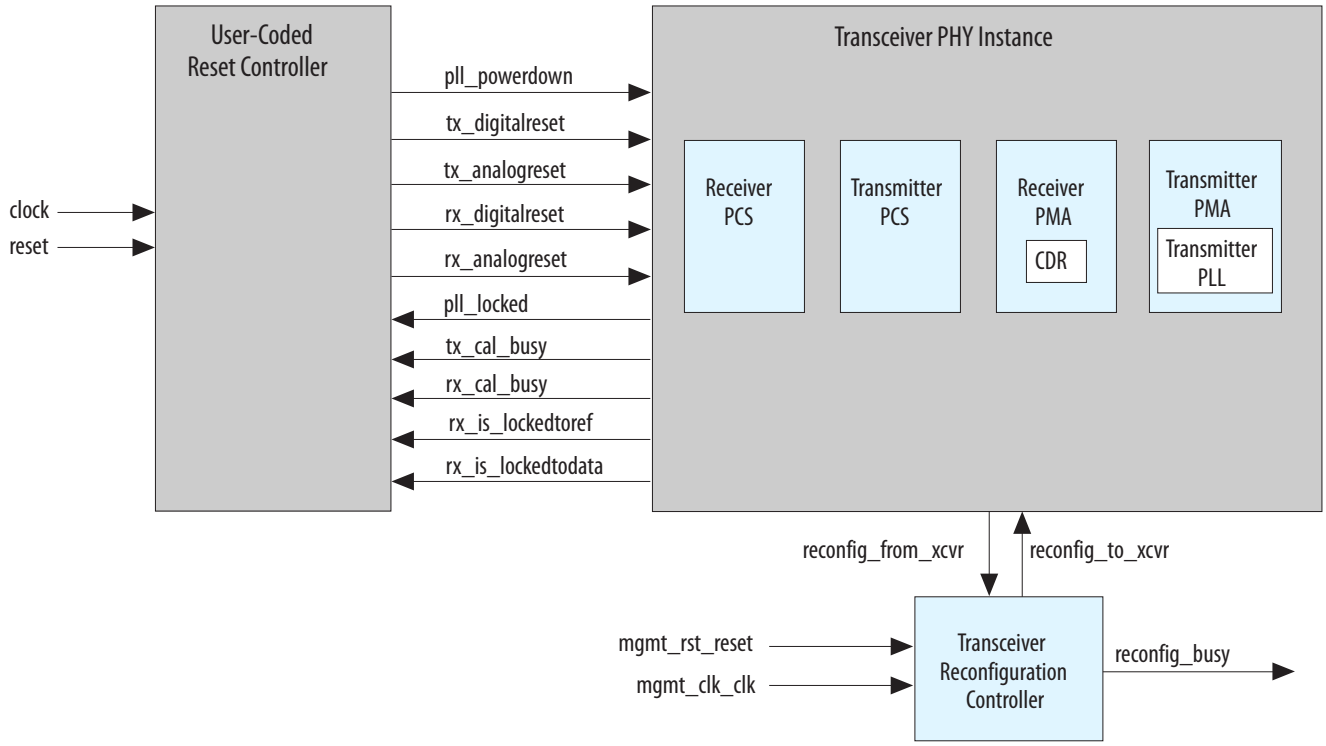


Table 3-3: Signals Used by the Transceiver PHY instance, Transceiver Reconfiguration Controller, and User-Coded Reset Controller

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.
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

Signal Name	Signal Type	Description
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`.
   - Assert `pll_powerdown` for a minimum duration of 1 µs ($t_{pll\_powerdown}$). If you use ATX PLL calibration, deassert `pll_powerdown` before `mgmt_rst_reset` so that the ATX PLL is not powered down during calibration. Otherwise, `pll_powerdown` can be deasserted anytime after `mgmt_rst_reset` is deasserted.
   - Make sure there is a stable reference clock to the PLL before deasserting `pll_powerdown` and `mgmt_rst_reset`.
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

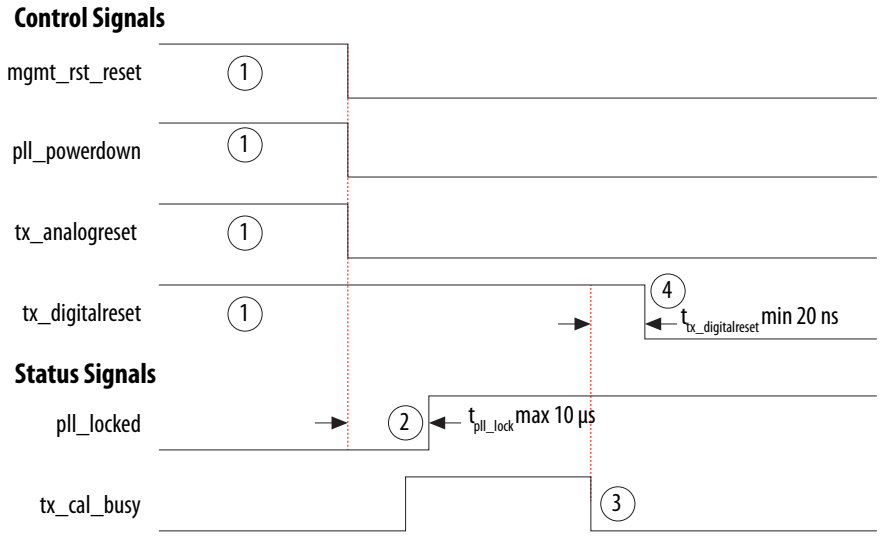


Table 3-4: Guidelines for Resetting the PLL, TX PMA, and TX PCS

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.

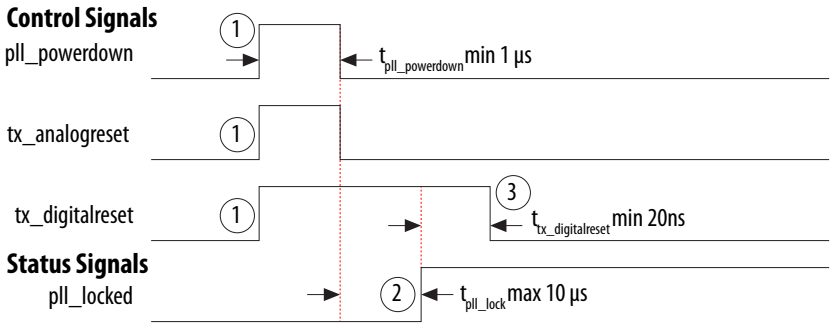
1. To reset the transmitter:

- 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`.

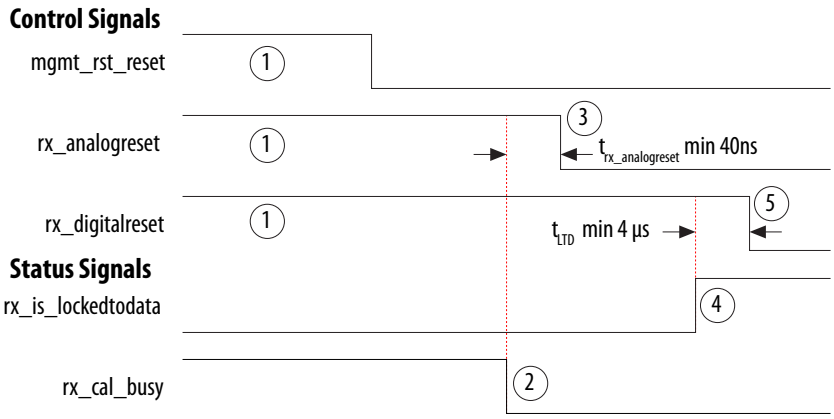
The receiver is now out of reset and ready for operation.

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 Stratix V Devices
For information about CDR lock modes.

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

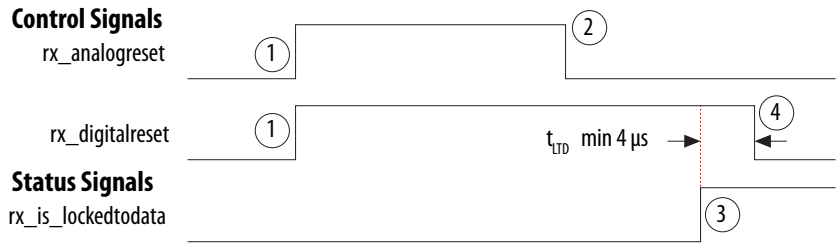
Follow this reset sequence to reset the analog or digital blocks of the receiver at any point during the device operation. This might be necessary for re-establishing a link or after certain dynamic reconfigurations.

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 Stratix 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 3-5: 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.

Register Name	Description
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.

The two control signals to enable and control the CDR in manual lock mode are rx_set_locktoref and rx_set_locktodata.

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 3-6: Control Settings for the CDR in Manual Lock Mode

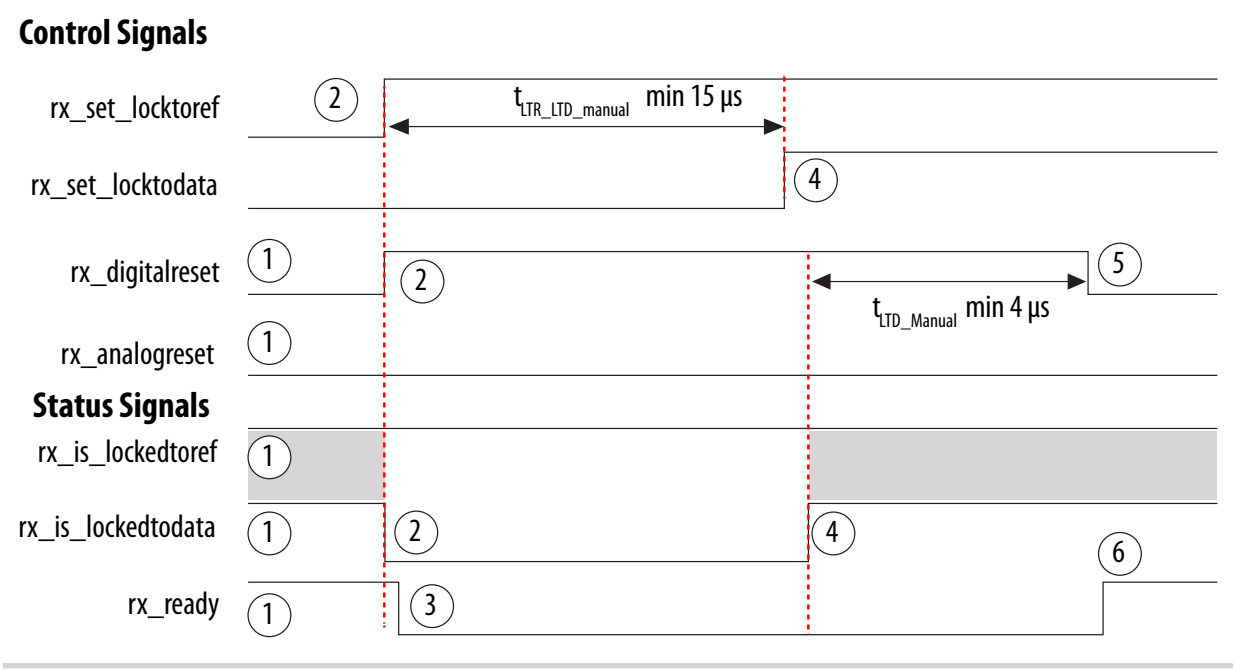
rx_set_locktoref	rx_set_locktodata	CDR Lock Mode
0	0	Automatic
1	0	Manual-RX CDR LTR
X	1	Manual-RX CDR LTD

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 after $t_{LTR\_LTD\_manual}$ to switch the CDR to the lock-to-data mode. 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.
5. Deassert the `rx_digitalreset` signal after t_{LTD_Manual}.
6. After the `rx_digitalreset` signal is deasserted, the `rx_ready` status signal gets asserted if you are using the Transceiver PHY Reset Controller, indicating that the receiver is now ready to receive data with the CDR in manual mode.

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



Transceiver Blocks Affected by the Reset and Powerdown Signals

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

Table 3-7: Transceiver Blocks Affected

Transceiver Block	pll_ powerdown	rx_digital-reset	rx_analogr-eset	tx_digitalreset	tx_analogreset
PLL					
CMU PLL	Yes	—	—	—	—
ATX 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	—	—	—

Transceiver Block	pll_ powerdown	rx_digital-reset	rx_analogr-eset	tx_digitalreset	tx_analogreset
Receiver Phase Compensation FIFO	—	Yes	—	—	—
Receiver 10G PCS					
Receiver Gear Box	—	Yes	—	—	—
Receiver Block Synchronizer	—	Yes	—	—	—
Receiver Disparity Checker	—	Yes	—	—	—
Receiver Descrambler	—	Yes	—	—	—
Receiver Frame Sync	—	Yes	—	—	—
Receiver 64B/66B Decoder	—	Yes	—	—	—
Receiver CRC32 Checker	—	Yes	—	—	—
Receiver 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	—
Transmitter Bit-Slip	—	—	—	Yes	—
Transmitter 10G PCS					
Transmitter FIFO	—	—	—	Yes	—
Transmitter Frame Generator	—	—	—	Yes	—
Transmitter CRC32 Generator	—	—	—	Yes	—
Transmitter 64B/66B Encoder	—	—	—	Yes	—
Transmitter Scrambler	—	—	—	Yes	—
Transmitter Disparity Generator	—	—	—	Yes	—

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

Document Revision History

Date	Version	Changes
September 2014	2014.09.30	• Added statement at the top of the chapter that an embedded reset controller is provided, but you can provide your own reset controller instead. • Added information about deasserting signals to the "Resetting the Transceiver with the PHY IP Embedded Reset Controller during Device Power-Up" section. • Added "Control Signals" and "Status Signals" labels to the timing diagrams. • Added a link to the Related Links in the "Clock Data Recovery in Manual Lock Mode" section.
October 2013	2013.10.11	• Changed term of User-Controlled Reset Controller to User-Coded Reset Controller.
May 2013	2013.05.06	• Added information about ATX PLL calibration. • Changes to figures to reflect correct values for t_{pll_lock}. • Added link to the known document issues in the Knowledge Base.
February 2013	2013.02.21	• Added information about reset options. • Clarified content about resetting the transmitter with the user-controlled reset controller during power-up. • Updated incorrect description of `pll_powerdown` when using Memory Map registers.
December 2012	2012.12.17	• Added information about resetting the transceiver during power-up and device operation • Restructured document

Date	Version	Changes
June 2012	3.3	• Updated for the Quartus II software version 12.0. • Revised Figure 3-2 , Figure 3-4 , and Figure 3-6 . • Added new Table 3-1. • Added new steps describing wave forms. • Deleted power-up Figures 3-2 and 3-5 and changed text accordingly.

Transceiver Configurations in Stratix V Devices

4

2014.09.30

SV52005 Subscribe Send Feedback

Stratix® V devices have a dedicated transceiver physical coding sublayer (PCS) and physical medium attachment (PMA) circuitry.

To implement a protocol, use a PHY IP listed in **Table 4-1**.

Stratix V devices support the following communication protocols:

- 10GBASE-R and 10GBASE-KR
- Interlaken
- PCI Express® (PCIe®)—Gen1, Gen2, and Gen3
- CPRI and OBSAI—Deterministic Latency Protocols
- XAUI

Support for other communication protocols or user-defined protocols can be enabled with the following PHY IPs:

- Native PHY IP using standard PCS and 10G PCS hardware options including reconfigurability between different PCS options
- Custom PHY IP using the standard PCS in a custom datapath
- Low Latency PHY IP using the standard or 10G PCS in a low latency datapath configuration

Related Information

- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the *Stratix V Device Handbook* chapters.
- **Upcoming Stratix V Device Features**
- **Altera Transceiver PHY IP Core User Guide**

Protocols and Transceiver PHY IP Support

Table 4-1: Protocols and PHY IP Features Support

Protocol Standard	Transceiver IP	PCS Type	Avalon-MM Register Interface	Reset Controller
PCIe Gen3 x1, x2, x4, x8	PHY IP Core for PCIe (PIPE) (10)	Standard and Gen3	Yes	Embedded

Protocol Standard	Transceiver IP	PCS Type	Avalon-MM Register Interface	Reset Controller
PCIe Gen2 x1, x2, x4, x8	PHY IP Core for PCIe (PIPE) (10)	Standard	Yes	Embedded
PCIe Gen1 x1, x2, x4, x8	PHY IP Core for PCIe (PIPE) (10)	Standard	Yes	Embedded
10GBASE-R	10GBASE-R	10G	Yes	Embedded
	Native PHY	10G	No	External Reset IP
10G/40/100G Ethernet	Native PHY	10G	No	External Reset IP
1G/10Gb Ethernet	1G/10GbE and 10GBASE-KR	Standard and 10G	Yes	Embedded
1G/10Gb Ethernet with 1588	1G/10GbE and 10GBASE-KR	Standard and 10G	Yes	Embedded
10G Ethernet with 1588	Native PHY	10G	No	External Reset IP
10GBASE-KR and 1000BASE-X	1G/10GbE and 10GBASE-KR	Standard and 10G	Yes	Embedded
1000BASE-X and SGMII Gigabit Ethernet	Custom PHY Standard	Standard	Yes	Embedded or External Reset IP
XAUI	XAUI PHY IP	Standard Soft-PCS	Yes	Embedded
SPAUI	Low Latency PHY	Standard and 10G	Yes	Embedded or External Reset IP
	Native PHY	Standard and 10G	No	External Reset IP
DDR XAUI	Low Latency PHY	Standard and 10G	Yes	Embedded or External Reset IP
	Native PHY	Standard and 10G	No	External Reset IP
Interlaken (CEI-6G/11G)	Interlaken PHY	10G	Yes	Embedded
	Native PHY (11)	10G	No	External Reset IP
OTU-4 (100G) via OIF SFI-S	Low Latency PHY	10G	Yes	External Reset IP
	Native PHY	10G	No	External Reset IP

(10) Hard IP for PCI Express is also available as a MegaCore function.
(11) A Soft-PCS bonding IP is required.

Protocol Standard	Transceiver IP	PCS Type	Avalon-MM Register Interface	Reset Controller
OTU-3 (40G) via OIF SFI-5.2/SFI-5.1	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Native PHY	10G	No	External Reset IP
OTU-2 (10G) via OIF SFI-5.1s	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
OTU-1 (2.7G)	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
SONET/SDH STS-768/ STM-256 (40G) via OIF SFI-5.2	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
SONET/SDH STS-768/ STM-256 (40G) via OIF SFI-5.2/SFI-5.1	Native PHY	Standard and 10G	No	External Reset IP
SONET/SDH STS-192/ STM-64 (10G) via SFP +/SFF-8431/ CEI-11G	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Native PHY	10G	No	External Reset IP
SONET/SDH STS-192/ STM-64 (10G) via OIF SFI-5.1s/SxI-5/ SFI-4.2	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
SONET STS-96 (5G) via OIF SFI-5.1s	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
SONET/SDH STS-48/ STM-16 (2.5G) via SFP/TFI-5.1	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
SONET/SDH STS-12/ STM-4 (0.622G) via SFP/TFI-5.1	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
Intel QPI	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	PMA-Direct	No	External Reset IP

Protocol Standard	Transceiver IP	PCS Type	Avalon-MM Register Interface	Reset Controller
10G SDI	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Native PHY	10G	No	External Reset IP
SD-SDI/HD-SDI/ 3G-SDI	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
10G GPON/EPON	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Native PHY	10G	No	External Reset IP
GPON/EPON	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
16/10G Fibre Channel	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Native PHY	10G	No	External Reset IP
8G/4G Fibre Channel	Low Latency PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
FDR/FDR-10 Infiniband x1, x4, x12	Low Latency PHY	10G	Yes	Embedded or External Reset IP
	Native PHY	10G	No	External Reset IP
SDR/DDR/QDR Infiniband x1, x4, x12	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
CPRI 4.2/OBSAI RP3 v4.2	Deterministic PHY	Standard	Yes	Embedded
	Native PHY	Standard	No	External Reset IP
SRIO 2.2/1.3 (12)	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
SATA 3.0/2.0/1.0 and SAS 2.0/1.0	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP

(12) Nx Multi-Alignment Deskew State Machine must be implemented in the core.

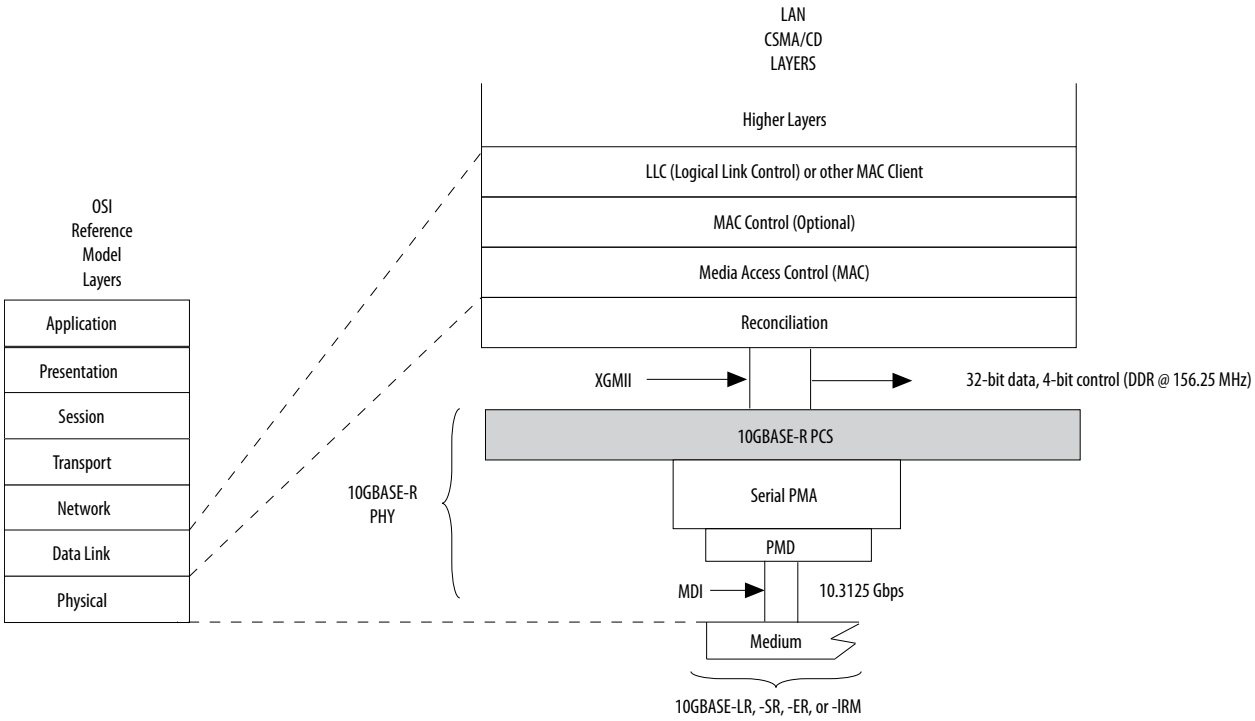
Protocol Standard	Transceiver IP	PCS Type	Avalon-MM Register Interface	Reset Controller
HiGig+/2+	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
JESD204A	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
ASI	Custom PHY	Standard	Yes	Embedded or External Reset IP
SPI 5 (50G)	Custom PHY	Standard	Yes	Embedded or External Reset IP
	Native PHY	Standard	No	External Reset IP
Custom and other protocols	Native PHY	Standard, 10G, and PMA-Direct	No	External Reset IP

10GBASE-R and 10GBASE-KR

10GBASE-R is used in optical module LAN applications such as optical routers, servers, and switches, and 10GBASE-KR is used in electrical backplane applications such as blade servers using Stratix V transceivers.

10GBASE-R is a specific physical layer implementation of the 10 Gigabit Ethernet link defined in clause 49 of the IEEE 802.3-2008 specification. The 10GBASE-R PHY uses the XGMII interface to connect to the IEEE802.3 media access control (MAC) and reconciliation sublayer (RS). The IEEE 802.3-2008 specification requires each 10GBASE-R link to support a 10 Gbps data rate at the XGMII interface and a 10.3125 Gbps serial line rate with 64B/66B encoding.

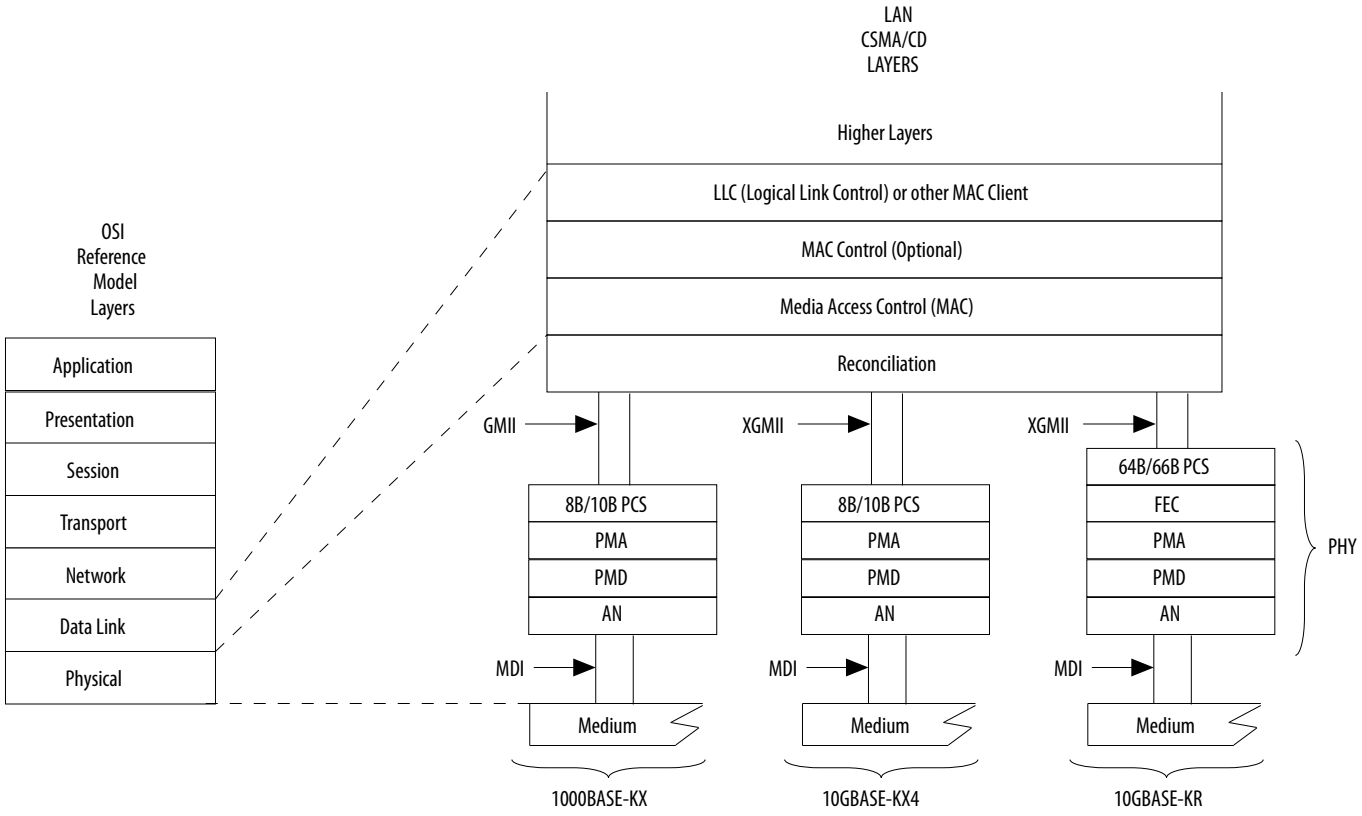
Figure 4-1: 10GBASE-R PHY Connection to IEEE802.3 MAC and RS



Note: To implement a 10GBASE-R link, instantiate the **10GBASE-R PHY IP** core in the IP Catalog, under **Ethernet** in the Interfaces menu.

The IEEE 802.3ap-2007 specification also requires each backplane link to support multi-data rates of 1Gbps and 10 Gbps speeds. 10GBASE-KR and 1000BASE-KX is the electrical backplane physical layer implementation for the 10 Gigabit and 1 Gigabit Ethernet link defined in clause 72 and clause 70 respectively of the IEEE 802.3ap-2007 specification. The 10 Gbps backplane ethernet 10GBASE-KR implementation uses the XGMII interface to connect to the reconciliation sublayer (RS) with 64B/66B PCS encoding, the optional Forward Error Correction (FEC), and Auto-Negotiation (AN) support to the Highest Common Denominator (HCD) technology with the partner link. The optional FEC, LT, and AN logic is implemented in the core fabric. The 1Gbps backplane ethernet 1000BASE-KX implementation uses the GMII interface to connect to the reconciliation sublayer (RS) with 8B/10B PCS encoding and Auto-Negotiation support to the HCD technology with the partner link.

Figure 4-2: 10GBASE-KR PHY Connection to IEEE802.3 MAC and RS



Note: To implement a 10GBASE-KR link with 1000BASE-KX support, instantiate the **1G/10GbE PHY IP** and **10GBASE-KR PHY IP** cores in the IP Catalog, under **Ethernet** in the Interfaces menu.

An additional license is required in order to use the 1G/10GbE and 10GBASE-KR PHY IP Core which also supports 10GBASE-R and 1000BASE-X links and auto-negotiation between the 10 Gigabit and 1 Gigabit Ethernet data rates.

Related Information

- **Altera Transceiver PHY IP Core User Guide**
- **10-Gbps Ethernet MAC MegaCore Function User Guide**

10GBASE-R and 10GBASE-KR Transceiver Datapath Configuration

The following figures show the transceiver blocks and settings enabled in 10GBASE-R and 10GBASE-KR configurations.

10GBASE-R

Figure 4-3: 10GBASE-R Datapath Configuration

The blocks shown as "Disabled" are not used, but incur latency. The blocks shown as "Bypassed" are not used and do not incur latency.

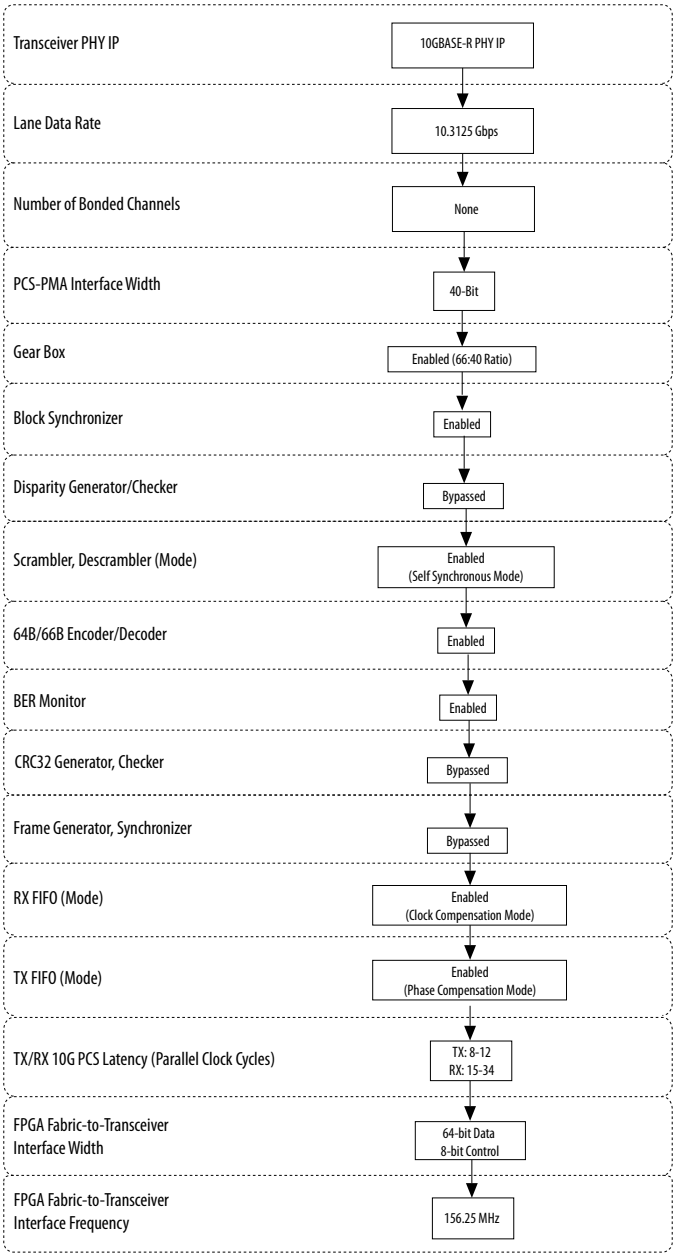
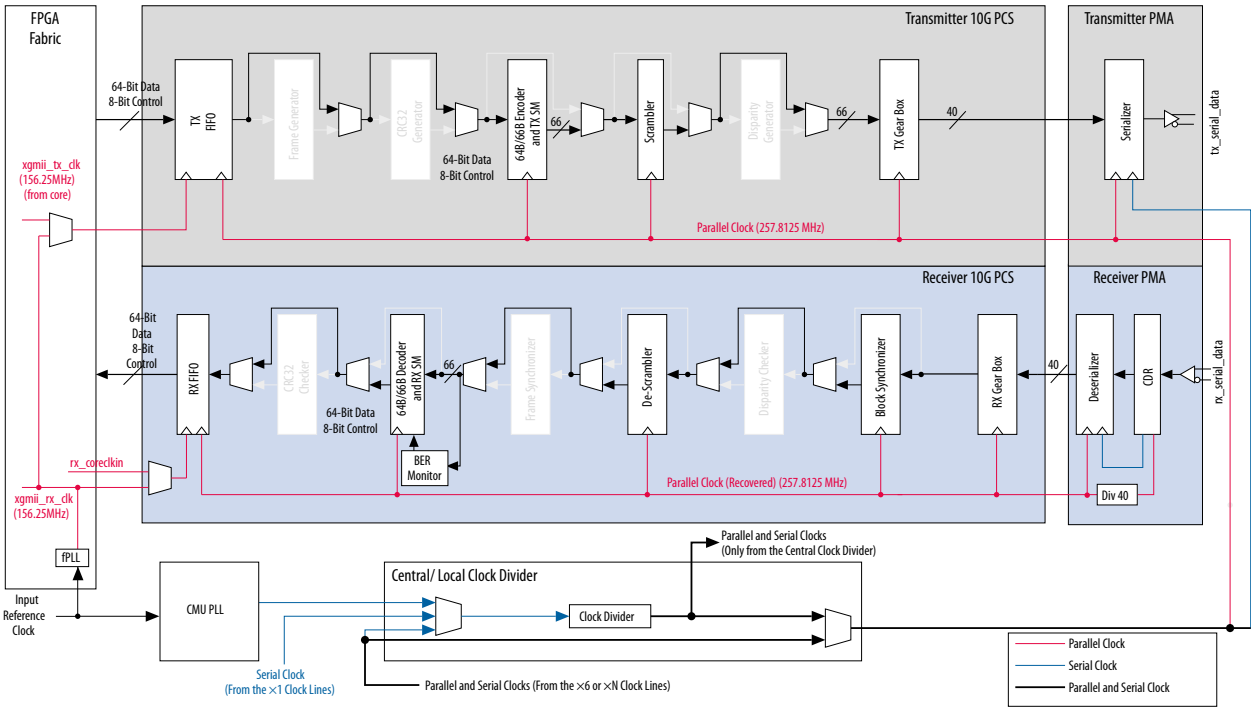


Figure 4-4: Transceiver Channel Datapath for a 10GBASE-R Configuration



10GBASE-KR

Figure 4-5: 10GBASE-R/KR and 1000Base-X/KX Datapath Configuration

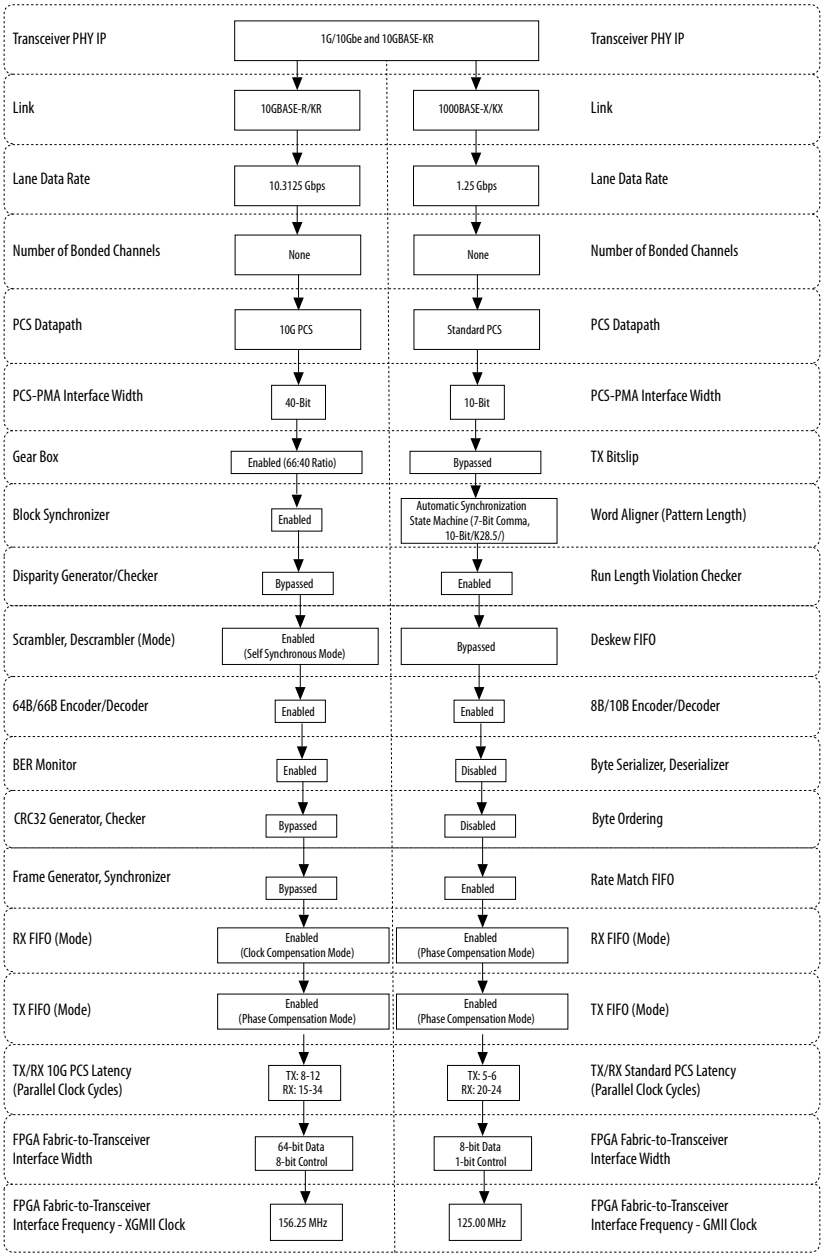
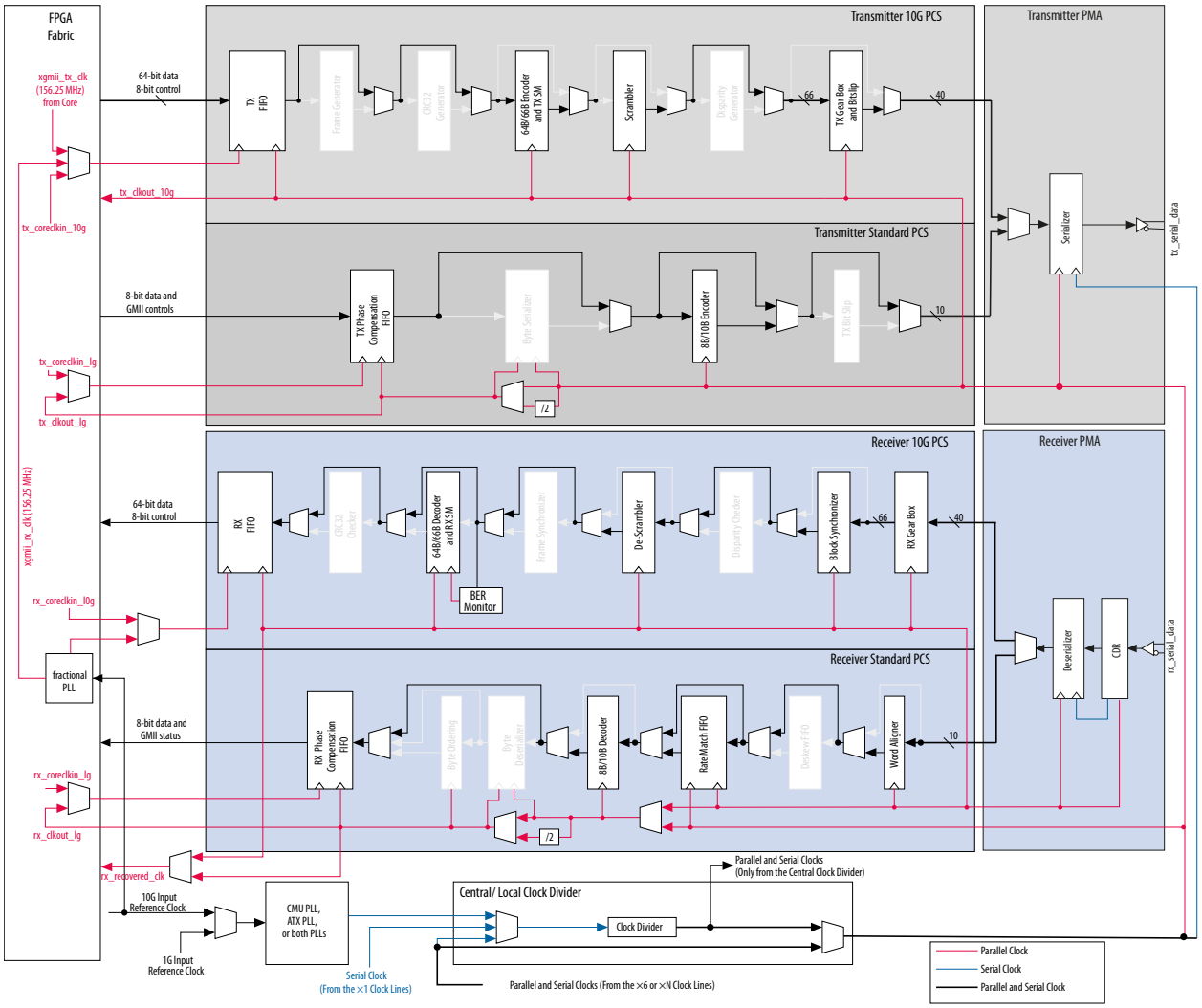


Figure 4-6: Transceiver Channel Datapath for 10GBASE-R/KR and 1000BASE-X/KX Configuration



10GBASE-R and 10GBASE-KR Supported Features

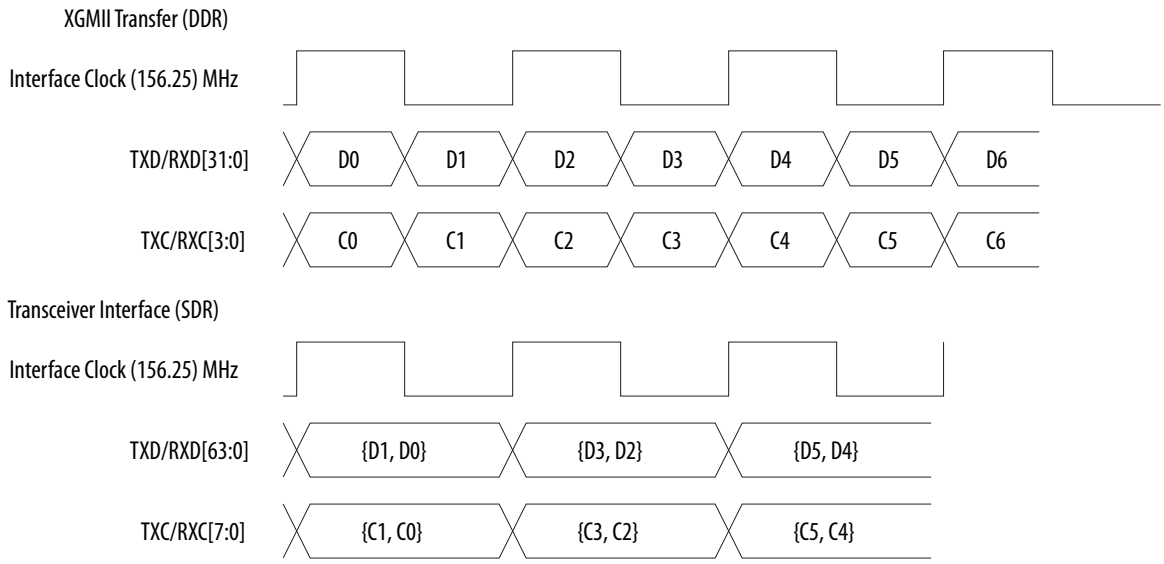
The following features are supported by the transceivers in 10GBASE-R and 10GBASE-KR configurations.

64-Bit Single Data Rate (SDR) Interface to the MAC/RS in 10GBASE-R and 10GBASE-KR Configurations

Clause 46 of the IEEE 802.3-2008 specification defines the XGMII interface between the 10GBASE-R and 10GBASE-KR PCS and the Ethernet MAC/RS. The XGMII interface defines the 32-bit data and 4-bit wide control character clocked between the MAC/RS and the PCS at both the positive and negative edge (double data rate – DDR) of the 156.25 MHz interface clock.

The transceivers do not support the XGMII interface to the MAC/RS as defined in the IEEE 802.3-2008 specification. Instead, they support a 64-bit data and 8-bit control SDR interface between the MAC/RS and the PCS.

Figure 4-7: XGMII Interface (DDR) versus Stratix V Transceiver Interface (SDR) for 10GBASE-R and 10GBASE-KR Configurations



64B/66B Encoding/Decoding in 10GBASE-R and 10GBASE-KR Configurations

The transceivers in 10GBASE-R and 10GBASE-KR configurations support 64B/66B encoding and decoding as specified in Clause 49 of the IEEE802.3-2008 specification. The 64B/66B encoder receives 64-bit data and 8-bit control code from the transmitter FIFO and converts it into 66-bit encoded data. The 66-bit encoded data contains two overhead sync header bits that the receiver PCS uses for block synchronization and bit-error rate (BER) monitoring.

The 64B/66B encoding also ensures enough transitions on the serial data stream for the receiver clock data recovery (CDR) to maintain its lock on the incoming data.

Transmitter and Receiver State Machines in 10GBASE-R and 10GBASE-KR Configurations

The transceivers in 10GBASE-R and 10GBASE-KR configurations implement the transmitter and receiver state diagrams shown in Figure 49-14 and Figure 49-15 of the IEEE802.3-2008 specification.

Besides encoding the raw data specified in the 10GBASE-R and 10GBASE-KR PCS, the transmitter state diagram performs functions such as transmitting local faults (LBLOCK_T) under reset, as well as transmitting error codes (EBLOCK_T) when the 10GBASE-R PCS rules are violated.

Besides decoding the incoming data specified in the 10GBASE-R and 10GBASE-KR PCS, the receiver state diagram performs functions such as sending local faults (LBLOCK_R) to the MAC/RS under reset and substituting error codes (EBLOCK_R) when the 10GBASE-R and 10GBASE-KR PCS rules are violated.

Block Synchronizer in 10GBASE-R and 10GBASE-KR Configurations

The block synchronizer in the receiver PCS determines when the receiver has obtained lock to the received data stream. It implements the lock state diagram shown in Figure 49-12 of the IEEE 802.3-2008 specification.

The block synchronizer provides a status signal to indicate whether it has achieved block synchronization or not.

Self-Synchronous Scrambling/Descrambling in 10GBASE-R and 10GBASE-KR Configurations

The scrambler/descrambler blocks in the transmitter/receiver PCS implements the self-synchronizing scrambler/descrambler polynomial 1 + x39 + x58, as described in clause 49 of the IEEE 802.3-2008 specification. The scrambler/descrambler blocks are self-synchronizing and do not require an initialization seed. Barring the two sync header bits in each 66-bit data block, the entire payload is scrambled or descrambled.

BER Monitor in 10GBASE-R and 10GBASE-KR Configurations

The BER monitor block in the receiver PCS implements the BER monitor state diagram shown in Figure 49-13 of the IEEE 802.3-2008 specification. The BER monitor provides a status signal to the MAC whenever the link BER threshold is violated.

The 10GBASE-R core and the 1G/10GbE and 10GBASE-KR PHY IP core (10GBASE-KR mode) provide a status flag to indicate a high BER whenever 16 synchronization header errors are received within a 125 µs window.

Clock Compensation in 10GBASE-R and 10GBASE-KR Configurations

The receiver FIFO in the receiver PCS datapath compensates up to ±100 ppm difference between the remote transmitter and the local receiver. The receiver FIFO does so by inserting Idles (/I/) and deleting Idles (/I/) or Ordered Sets (/O/), depending on the ppm difference.

- **Idle Insertion** — The receiver FIFO inserts eight /I/ codes following an /I/ or /O/ to compensate for clock rate disparity.
- **Idle (/I/) or Sequence Ordered Set (/O/) Deletion** — The receiver FIFO deletes either four /I/ codes or ordered sets (/O/) to compensate for the clock rate disparity. The receiver FIFO implements the following IEEE802.3-2008 deletion rules:
  - Deletes the lower four /I/ codes of the current word when the upper four bytes of the current word do not contain a Terminate /T/ control character.
  - Deletes one /O/ ordered set only when the receiver FIFO receives two consecutive /O/ ordered sets.

10GBASE-KR and 1000BASE-KX Link Training

The Link Training function defined in clause 72 of IEEE 802.3ap-2007 specification is implemented in the core fabric. The 1G/10GbE and 10GBASE-KR PHY IP Link Training logic includes the Training Frame Generator, Training Frame Synchronizer, PRBS11 generator, control channel codec, Local Device (LD) transceiver transmit PMA pre-emphasis coefficient status reporting, the Link Partner (LP) transmit PMA pre-emphasis coefficient update request, and the receiver link training status.

Stratix V GX channels employ three PMA transmit driver pre-emphasis taps: pre-tap, main tap, and first post-tap as required and defined by clause 72, Section 72.7.1.10 Transmitter output waveform for 10GBASE-KR PHY operation. The pre-emphasis coefficients is dynamically adjusted by the PHY IP during the Link Training process.

10GBASE-KR and 1000BASE-KX Auto-Negotiation

The Auto-Negotiation function defined in clause 73 of IEEE 802.3ap-2007 specification must be implemented in the core fabric. The 1G/10GbE and 10GBASE-KR PHY IP Auto-Negotiation logic includes the Differential Manchester Encoding (DME) page codec, AN page lock and synchronizer, and the Transmit, Receive, and Arbitration logic state machines.

10GBASE-KR Forward Error Correction

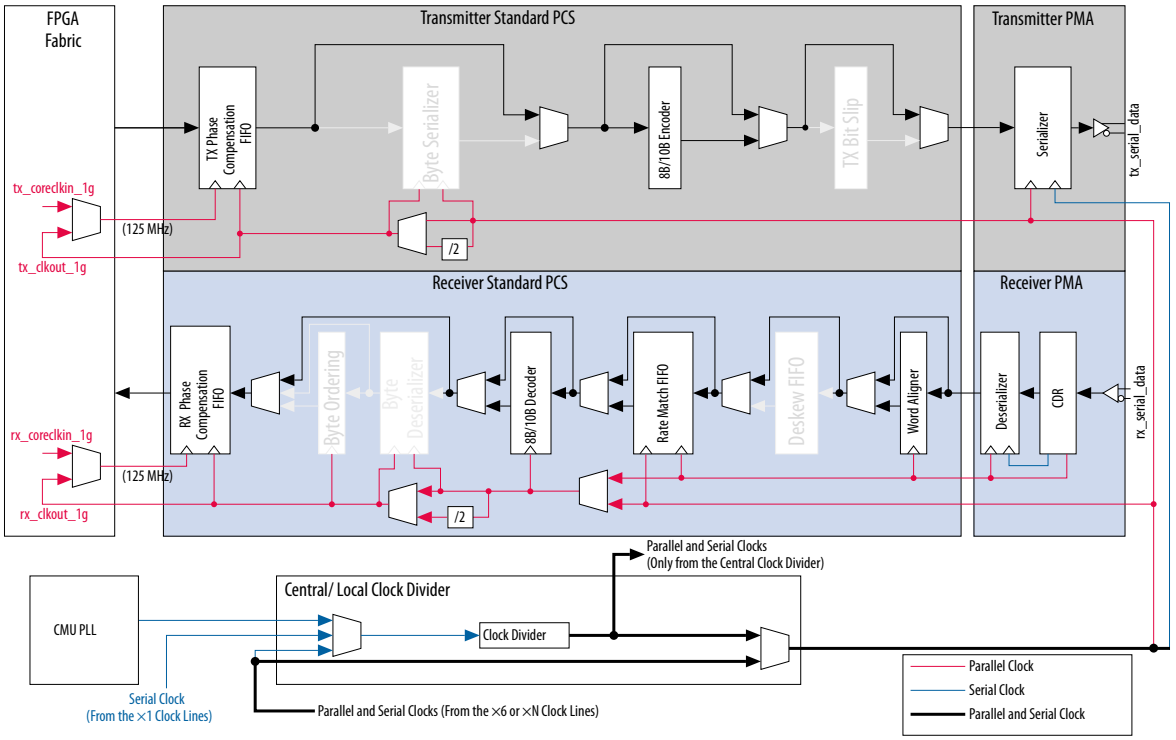
The FEC function defined in clause 74 of IEEE 802.3ap-2007 specification must be implemented in the core fabric. In Stratix V devices, the hard PCS does not support applications that require FEC function-

ality. To implement a 10GBASE-KR link with FEC support, the entire PCS functionality and the FEC logic must be implemented in the core fabric and the transceiver configured in Low Latency Configuration using the Native PHY IP.

1000BASE-X and 1000BASE-KX Transceiver Datapath

The following figure shows the transceiver datapath and clock frequencies in 1000BASE-X and 1000BASE-KX configurations.

Figure 4-8: 1000BASE-X and 1000BASE-KX Datapath Configurations



1000BASE-X and 1000BASE-KX Supported Features

The following features are supported by the transceivers in 1000BASE-X and 1000BASE-KX configurations.

8B/10B Encoder in 1000BASE-X and 1000BASE-KX Configurations

In 1000BASE-X and 1000BASE-KX modes, 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.

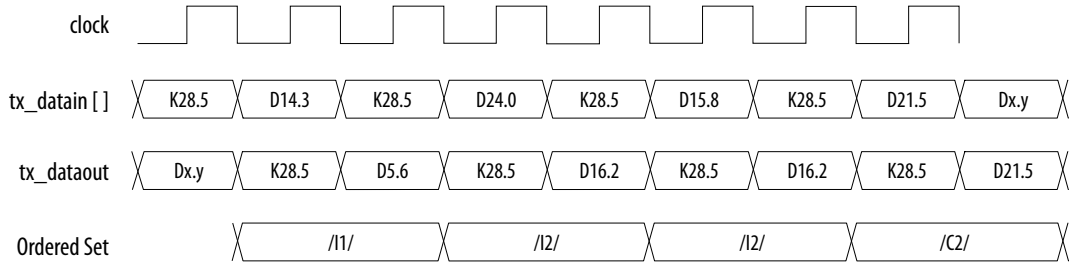
Idle Ordered-Set Generation in 1000BASE-X and 1000BASE-KX Configurations

The IEEE 802.3 specification requires the 1000BASE-X and 1000BASE-KX PHY to transmit idle ordered sets (/I/) continuously and repetitively whenever the GMII is idle. This ensures that the receiver maintains bit and word synchronization whenever there is no active data to be transmitted.

In 1000BASE-X and 1000BASE-KX functional modes, any /Dx.y/ following a /K28.5/ comma is replaced by the transmitter with either a /D5.6/ (/I1/ ordered set) or a /D16.2/ (/I2/ ordered set), depending on the current running disparity. The exception is when the data following the /K28.5/ is /D21.5/ (/C1/ ordered set) or /D2.2/ (/C2/) ordered set. If the running disparity before the /K28.5/ is positive, an /I1/ ordered set is generated. If the running disparity is negative, a /I2/ ordered set is generated. The disparity at the end of a /I1/ is the opposite of that at the beginning of the /I1/. The disparity at the end of a /I2/ is the same as the beginning running disparity (right before the idle code). This ensures a negative running disparity at the end of an idle ordered set. A /Kx.y/ following a /K28.5/ is not replaced.

Note: /D14.3/, /D24.0/, and /D15.8/ are replaced by /D5.6/ or /D16.2/ (for /I1/, /I2/ ordered sets). /D21.5/ (part of the /C1/ order set) is not replaced.

Figure 4-9: Example of Automatic Ordered Set Generation



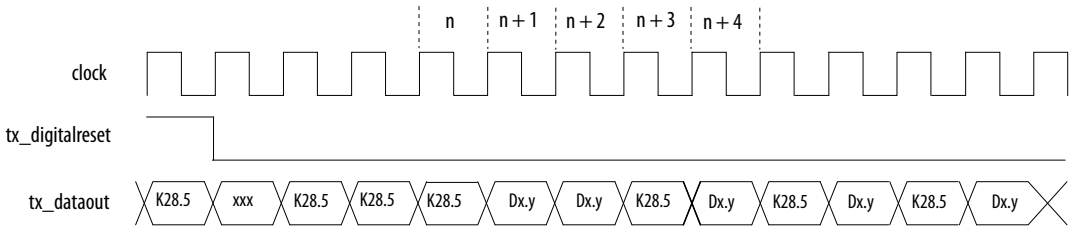
Reset Condition in 1000BASE-X and 1000BASE-KX Configurations

After deassertion of `tx_digitalreset`, the 1000BASE-X and 1000BASE-KX transmitters automatically transmit three /K28.5/ comma code groups before transmitting user data on the `tx_datain` port. This could affect the synchronization state machine behavior at the receiver.

Depending on when you start transmitting the synchronization sequence, there could be an even or odd number of /Dx.y/ code groups transmitted between the last of the three automatically sent /K28.5/ code groups and the first /K28.5/ code group of the synchronization sequence. If there is an even number of /Dx.y/ code groups received between these two /K28.5/ code groups, the first /K28.5/ code group of the synchronization sequence begins at an odd code group boundary (`rx_even` = FALSE). An IEEE802.3-compliant 1000BASE-X or 1000BASE-KX synchronization state machine treats this as an error condition and goes into the loss of sync state.

The following figure shows an example of even numbers of /Dx.y/ between the last automatically sent /K28.5/ and the first user-sent /K28.5/. The first user-sent /K28.5/ code group received at an odd code group boundary in cycle n + 3 takes the receiver synchronization state machine in the loss of sync state. The first synchronization ordered set /K28.5/Dx.y/ in cycles n + 3 and n + 4 is discounted and three additional ordered sets are required for successful synchronization.

Figure 4-10: Example of Reset Condition in 1000BASE-X and 1000BASE-KX Configurations



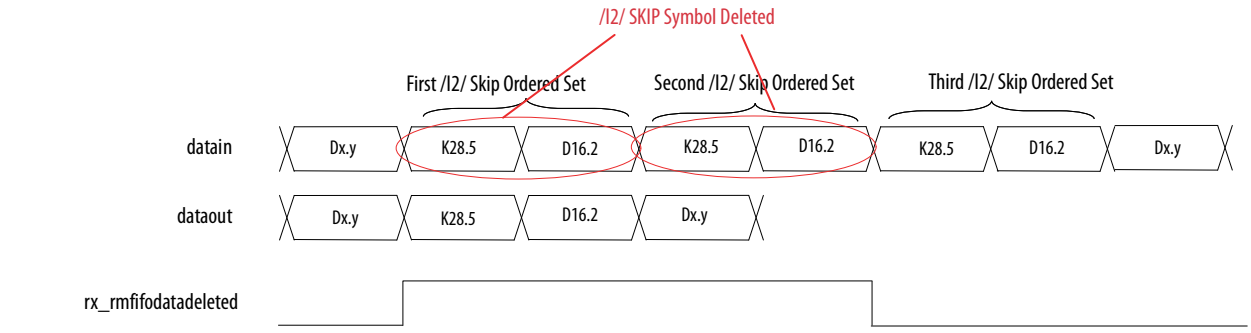
Rate Match FIFO in 1000BASE-X and 1000BASE-KX Configurations

In 1000BASE-X and 1000BASE-KX modes, 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 1000BASE-X and 1000BASE-KX protocols require the transmitter to send idle ordered sets /I1/ (/K28.5/D5.6/) and /I2/ (/K28.5/D16.2/) during inter-packet 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 synchronization is acquired by driving the `rx_syncstatus` signal high. The rate matcher deletes or inserts both symbols (/K28.5/ and /D16.2/) of the /I2/ ordered sets even if it requires deleting only one symbol to prevent the rate match FIFO from overflowing or under-running. It can insert or delete as many /I2/ ordered sets as necessary to perform the rate match operation.

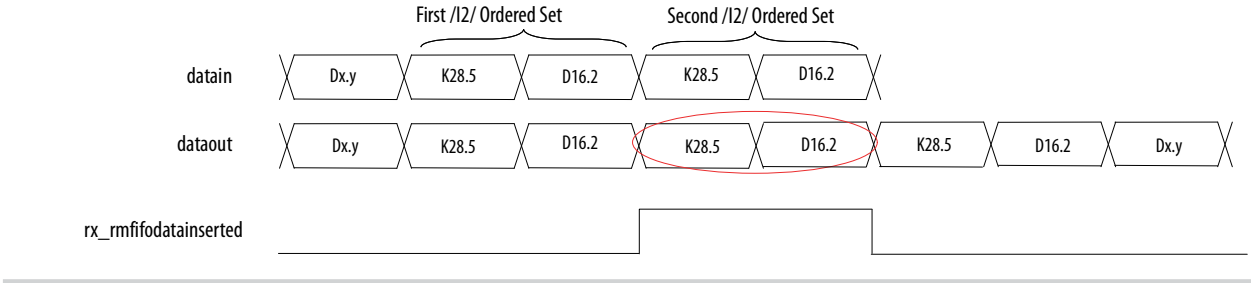
The following figure shows an example of rate match FIFO deletion where three symbols are required to be deleted. Because the rate match FIFO can only delete /I2/ ordered set, it deletes two /I2/ ordered sets (four symbols deleted).

Figure 4-11: Example of Rate Match Deletion in 1000BASE-X and 1000BASE-KX Configurations



The following figure shows an example of rate match FIFO insertion in the case where one symbol is required to be inserted. Because the rate match FIFO can only delete /I2/ ordered set, it inserts one /I2/ ordered set (two symbols inserted).

Figure 4-12: Example Rate Match Insertion in 1000BASE-X and 1000BASE-KX Configurations



Two register bits, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, indicate rate match FIFO deletion and insertion events. Both the `rx_rmfifodatadeleted` and `rx_rmfifodatainserted` status flags are latched High during deleted and inserted /I2/ ordered sets.

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**.

Word Aligner in 1000BASE-X and 1000BASE-KX Configurations

The word aligner in 1000BASE-X and 1000BASE-KX functional modes is configured in automatic synchronization state machine mode. The Quartus II software automatically configures the synchronization state machine to indicate synchronization when the receiver receives three consecutive synchronization ordered sets. A synchronization ordered set is a /K28.5/ code group followed by an odd number of valid /Dx.y/ code groups. The fastest way for the receiver to achieve synchronization is to receive three continuous {/K28.5/, /Dx.y/} ordered sets.

Receiver synchronization is indicated on the `rx_syncstatus` port of each channel. A high on the `rx_syncstatus` port indicates that the lane is synchronized; a low on the `rx_syncstatus` port indicates that the lane has fallen out of synchronization. The receiver loses synchronization when it detects four invalid code groups separated by less than three valid code groups or when it is reset.

Synchronization State Machine Parameters in 1000BASE-X and 1000BASE-KX Configurations

Table 4-2: Synchronization State Machine Parameters in 1000BASE-X or 1000BASE-KX Mode

Synchronization State Machine Parameters	Settings
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

Transceiver Clocking in 10GBASE-R, 10GBASE-KR, 1000BASE-X, and 1000BASE-KX Configurations

The CMU PLL or the auxiliary transmit (ATX) PLLs in a transceiver bank generate the transmitter serial and the fractional PLL for the parallel clocks for the 10GBASE-R, 10GBASE-KR, 1000BASE-X, and 1000BASE-KX channels. The following table lists the configuration details.

Table 4-3: Input Reference Clock Frequency and Interface Speed Specifications for 10GBASE-R, 10GBASE-KR, and 1000BASE-KX Configurations

PHY IP Type	PHY Type	Input Reference Clock Frequency (MHz)	FPGA Fabric-Transceiver Interface Width	FPGA Fabric-Transceiver Interface Frequency (MHz)
10GBASE-R PHY IP	10GBASE-R	644.53125, 322.265625	64-bit data, 8-bit control	156.25
1G/10GbE and 10GBASE-KR PHY IP	10GBASE-R and 10GBASE-KR	644.53125, 322.265625	64-bit data, 8-bit control	156.25

PHY IP Type	PHY Type	Input Reference Clock Frequency (MHz)	FPGA Fabric-Transceiver Interface Width	FPGA Fabric-Transceiver Interface Frequency (MHz)
1G/10GbE and 10GBASE-KR PHY IP	1000BASE-X and 1000BASE-KX	125, 62.5	8-bit data, `gmii_tx_en` and `gmii_tx_err` control	125

Interlaken

Interlaken is a scalable, chip-to-chip interconnect protocol that enables transmission speeds from 10 to more than 100 Gbps.

Stratix V devices support a transmission speed of up to 14.1 Gbps per lane in an Interlaken configuration. All the PCS blocks in the Interlaken configuration conform to the Interlaken Protocol Definition, Rev 1.2.

To implement an Interlaken link, instantiate the **Interlaken PHY IP** core in the IP Catalog, under **Interlaken** in the Interfaces menu.

Related Information

Refer to the Interlaken PHY IP Core chapter in the Altera Transceiver PHY IP Core User Guide

Transceiver Datapath Configuration

Figure 4-13: Interlaken Datapath Configuration

Blocks shown as “Disabled” are not used but incur latency. Blocks shown as “Bypassed” are not used and do not incur any latency. The maximum data rates and frequencies are for the fastest speed grade devices.

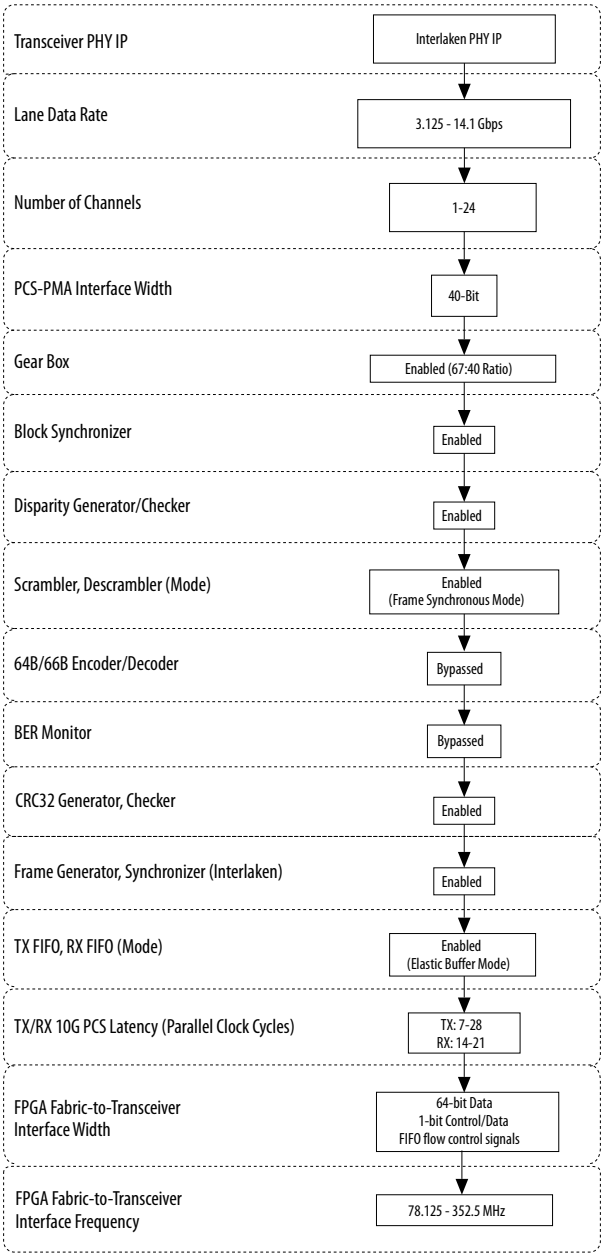
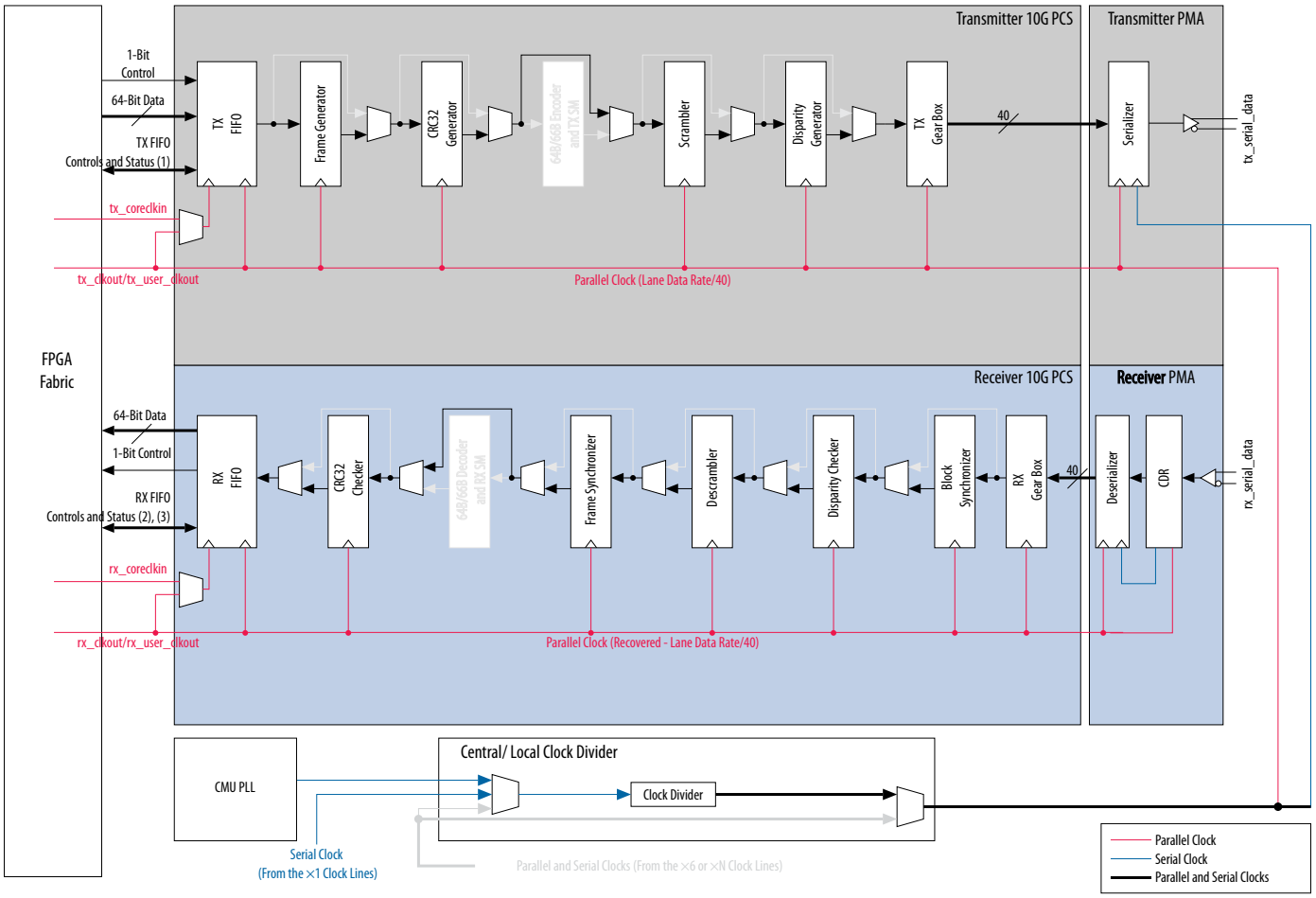


Figure 4-14: Transceiver Channel Datapath for Interlaken Configuration



Notes:
(1) TX FIFO Control and Status (transmit backpressure and datavalid, synchronization done)
(2) RX FIFO Control (receive FIFO read enable and datavalid)
(3) RX FIFO Status (receive FIFO overflow and partially empty)

Supported Features

The Interlaken protocol supports a number of framing layer functions. The functions are defined in the Interlaken Protocol Definition, Rev 1.2.

Table 4-4: Supported Features in Interlaken Configuration

Feature	Supported
Metaframe generation and payload insertion	Yes
Block synchronization (word alignment) and metaframe synchronization (frame synchronization)	Yes
64B/67B framing	Yes
±96 bits disparity maintenance	Yes
Frame synchronous scrambling and descrambling	Yes

Feature	Supported
Diagnostic word generation	Yes
Framing Layer Control Word Forwarding	Yes
CRC-32 generation and checking of lane data integrity	Yes
Multi-lane deskew alignment	No
Transmit and receive FIFO backpressure control and handshake	Yes

Block Synchronizer

The block synchronizer in the receiver PCS achieves and maintains a 64B/67B word boundary lock. This block searches for valid synchronization header bits within the data stream and achieves lock after 64 consecutive legal synchronization patterns are found. After a 64B/67B word boundary lock is achieved, the block synchronizer continuously monitors and flags for invalid synchronization header bits. If 16 or more invalid synchronization header bits are found within 64 consecutive word boundaries, the block synchronizer deasserts the lock state and searches again for valid synchronization header bits.

The block synchronizer implements the flow diagram shown in Figure 13 of Interlaken Protocol Definition v1.2 and provides the word lock status to the FPGA fabric.

64B/67B Frame Generator

The transmit frame generator implements 64B/67B encoding, as explained in Interlaken Protocol Definition v1.2. The Interlaken metaframe generator synchronously generates the framing layer control words, frame synchronizer, scrambler state, skip words, and diagnostic word, and maps the transmitter data into the payload of the metaframes. The metaframe length is programmable from 5 to a maximum value of 8191, 8-byte words.

Note: Ensure that the metaframe length is programmed to the same value for both the transmitter and receiver.

Frame Synchronizer

The receive frame synchronizer delineates the metaframe boundaries and searches for each of the framing layer control words: Synchronization, Scrambler State, Skip, and Diagnostic. When four consecutive synchronization words have been identified, the frame synchronizer achieves the frame locked state. Subsequent metaframes are then checked for valid synchronization and scrambler state words. If four consecutive invalid synchronization words or three consecutive mismatched scrambler state words are received, the frame synchronizer loses frame lock. In addition, the frame synchronizer provides a receiver metaframe lock status to the FPGA fabric.

Running Disparity

The disparity generator inverts the sense of bits in each transmitted word to maintain a running disparity of ± 96 bit boundary. It supplies a framing bit in bit position 66 as explained in Table 4 of Interlaken Protocol Definition Revision 1.2. The framing bit enables the disparity checker to identify whether bits[63:0] for that word are inverted.

Frame Synchronous Scrambling/Descrambling

The scrambler/descrambler block in the transmitter/receiver PCS implements the scrambler/descrambler polynomial x58 + x39 + 1 per Interlaken Protocol Definition Revision 1.2. Synchronization and Scrambler State Words, as well as the 64B/67B framing bits are not scrambled/descrambled. The Interlaken PHY IP

core automatically programs random linear feedback shift register (LFSR) initialization seed values per lane.

The receiver PCS synchronizes the scrambler with the metaframe as described in the state flow shown in Figure 1 of Interlaken Protocol Definition Revision 1.2.

The frame synchronizer features a whole set of error and performance monitoring ports to the FPGA fabric interface and register status bits when using the Avalon® Memory-Mapped Management Interface. A receiver ready port, frame lock status, and cyclic redundancy check (CRC)-32 error detection port is available to the FPGA fabric. The Avalon Memory-Mapped Management Interface provides additional functionality with word boundary lock, frame lock status, synchronization word error detection, scrambler mismatch error, and CRC-32 error detection status register bits.

Skip Word Insertion

The frame generator generates the mandatory fixed location skip words with every metaframe following the scrambler state word and generates additional skip words based on the transmitter FIFO capacity state.

Skip Word Deletion

The frame synchronizer does not delete skip words. Instead, the frame synchronizer forwards the skip words it receives to the MAC layer so the MAC can maintain and perform deskew alignment.

Diagnostic Word Generation and Checking of Lane Data Integrity (CRC-32)

The CRC-32 generator calculates the CRC for each metaframe and appends it to the diagnostic word of the metaframe. An optional CRC-32 error flag is also provided to the FPGA fabric.

Framing Layer Control Word Forwarding

The four metaframe framing layer control words-Synchronization, Scrambler State, Skip, and Diagnostic Words-are not deleted but forwarded to the MAC layer. This action enables the MAC layer to employ multi-lane deskew alignment within the FPGA fabric.

Multi-Lane Deskew Alignment

The Interlaken PHY IP does not support multi-lane deskew alignment. You must implement the multi-lane deskew alignment state machine in the core fabric or the Altera Interlaken MegaCore® function within the FPGA fabric.

Transmit and Receive FIFO Control and Status

The Interlaken PCS configures the transmit and receive FIFOs in elastic buffer mode. In this mode of operation, a lane synchronization, backpressure and FIFO control, and status port signals are provided to the MAC layer for handshaking.

Transceiver Multi-Lane Bonding and Transmit Skew

A soft-bonding IP is used for Interlaken bonding in the transceivers. The transceiver clocking in each lane is configured as non-bonded. For multi-lane designs, a dedicated PLL reference clock pin that is equidistant from the transmit PLLs in each bank must be selected. You must tightly match lane board traces to minimize lane-to-lane skew.

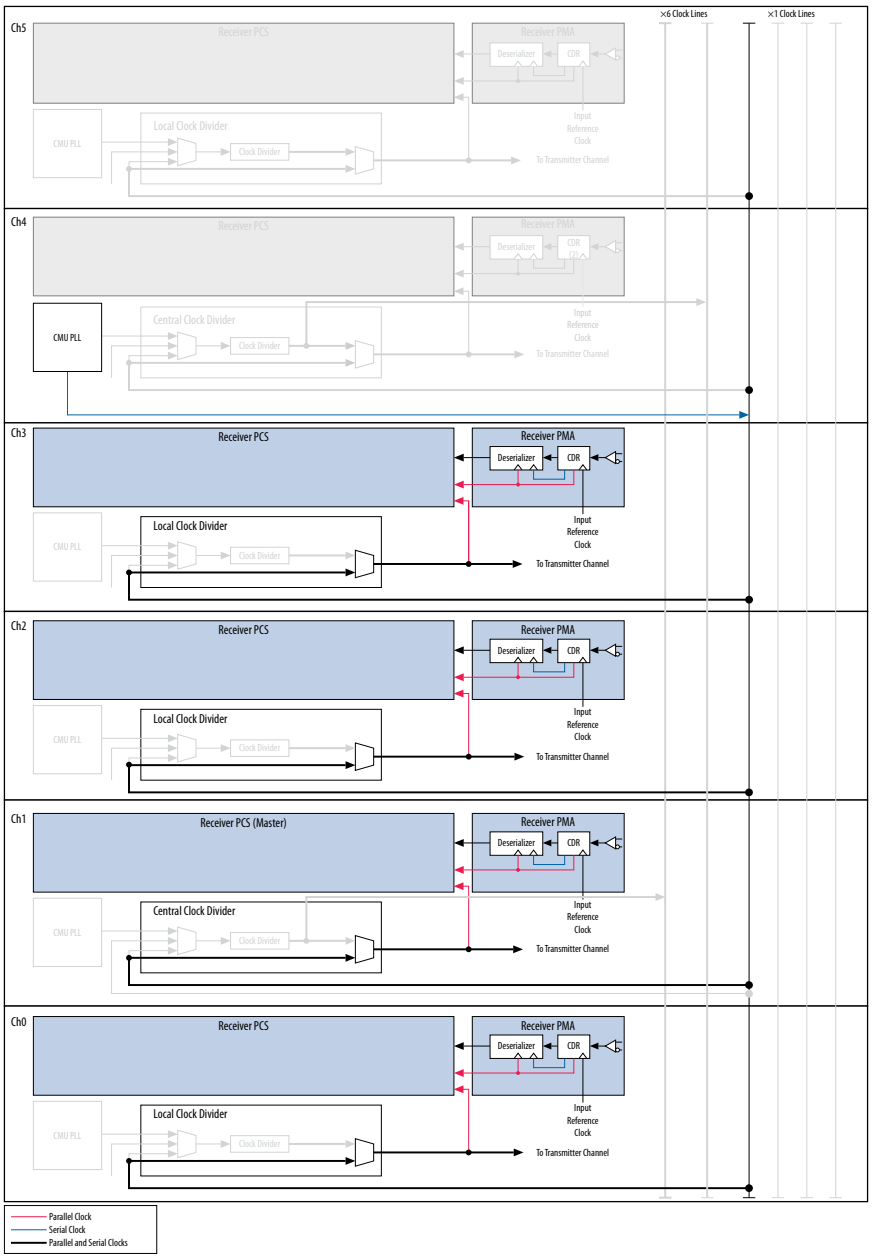
Related Information

- **For more information about Interlaken PHY IP control and status signals associated with each feature, refer to the Interlaken PHY IP Core chapter in the Altera Transceiver PHY IP Core User Guide**
- **Interlaken MegaCore Function User Guide**

Transceiver Clocking

Describes the transceiver clocking for the Interlaken protocol.

Figure 4-15: Clocking Resources Available in a Four-Lane Interlaken Configuration



A CMU PLL may provide a clock for up to five Interlaken lanes within a transceiver bank. If an ATX PLL is used, the PLL can clock up to six Interlaken lanes in a transceiver bank.

Note: To enable the ATX PLL, you must select **ATX PLL** for the **PLL type** parameter in the Interlaken PHY IP.

PCI Express (PCIe)—Gen1, Gen2, and Gen3

The PCIe specification (version 3.0) provides implementation details for a PCIe-compliant physical layer device at Gen1 (2.5 Gbps), Gen2 (5 Gbps), and Gen3 (8 Gbps) signaling rates.

The devices have built-in PCIe hard IP blocks to implement the PHY MAC layer, data link layer, and transaction layer of the PCIe protocol stack. Up to four PCIe hard IP block reside within a Stratix V device. If you enable the PCIe hard IP block, the transceiver interfaces with the hard IP block. Otherwise, the transceiver interfaces directly through the PIPE interface. You must then implement a Soft-IP MAC layer, data link layer, and transaction layer to the PIPE interface from the core fabric.

You can configure the transceivers in a PCIe functional configuration using one of the following methods:

- Stratix V Hard IP for PCI Express
- PHY IP core for PCI Express (PIPE)

The following table shows the two methods supported by transceivers in a PCIe functional configuration.

Table 4-5: Support for Transceivers

Support	Stratix V Hard IP for PCI Express	PHY IP Core for PCI Express (PIPE)
Gen1, Gen2, and Gen3 data rates	Yes	Yes
MAC, data link, and transaction layer	Yes	—
Transceiver interface	Hard IP through PIPE 3.0-like	PIPE 2.0 for Gen1 and Gen2 PIPE 3.0-like for Gen3 with Gen1/Gen2 support

To implement the PHY IP Core for PCI Express (PIPE) configuration, instantiate the **PHY IP Core for PCI Express (PIPE)** in the IP Catalog, under **PCI Express** in the Interfaces menu.

Stratix V transceivers support x1, x2, x4, and x8 lane configurations. In a PCIe x1 configuration, the PCS and PMA blocks of each channel are clocked and reset independently. PCIe x2, x4, and x8 configurations support channel bonding for two-lane, four-lane, and eight-lane PCIe links. In these bonded channel configurations, the PCS and PMA blocks of all bonded channels share common clock and reset signals.

Related Information

- **Stratix V Hard IP for PCI Express User Guide**
- **Refer to the PHY IP Core for PCI Express (PIPE) chapter in the Altera Transceiver PHY IP Core User Guide**

Transceiver Datapath Configuration

The transceiver datapaths for PCI Express are different depending on whether or not Gen3 is enabled.

Figure 4-16: PCIe Gen1 and Gen2 PIPE Datapath Configuration

This transceiver datapath configuration is for a configuration without Gen3 enabled.

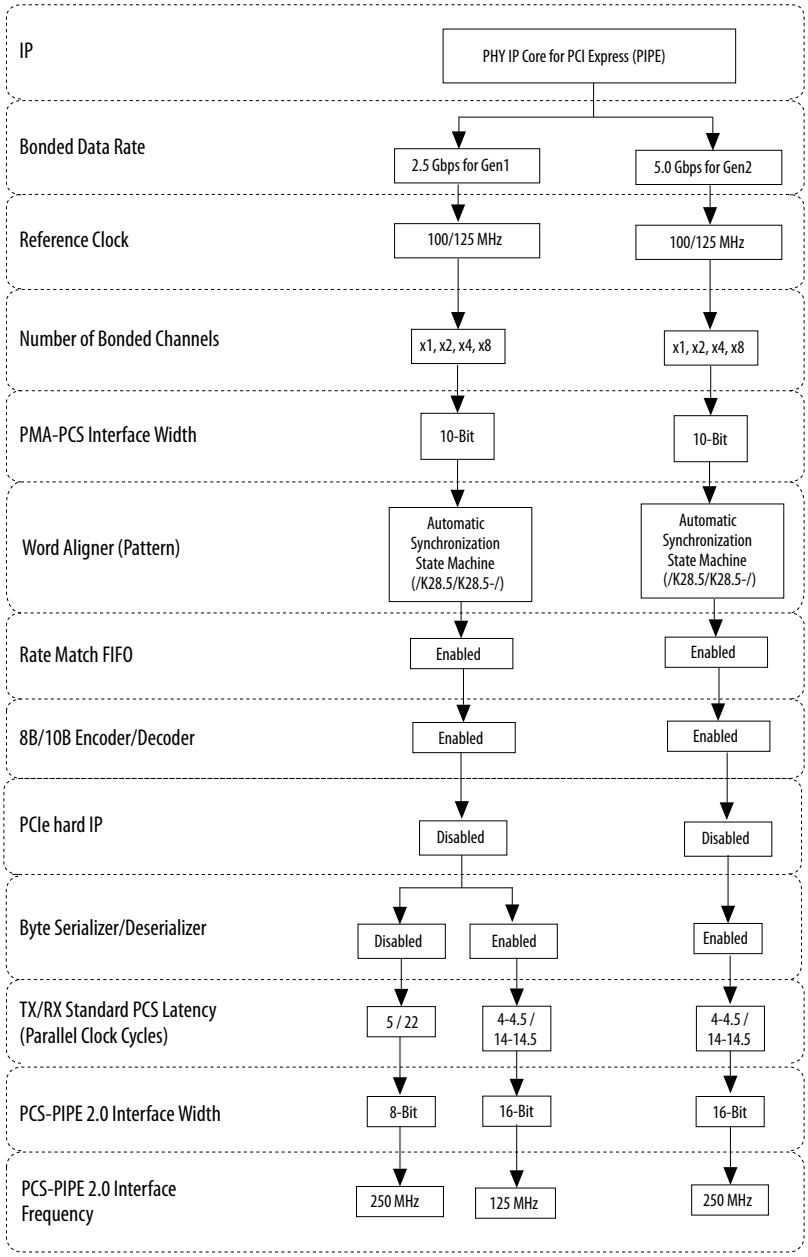
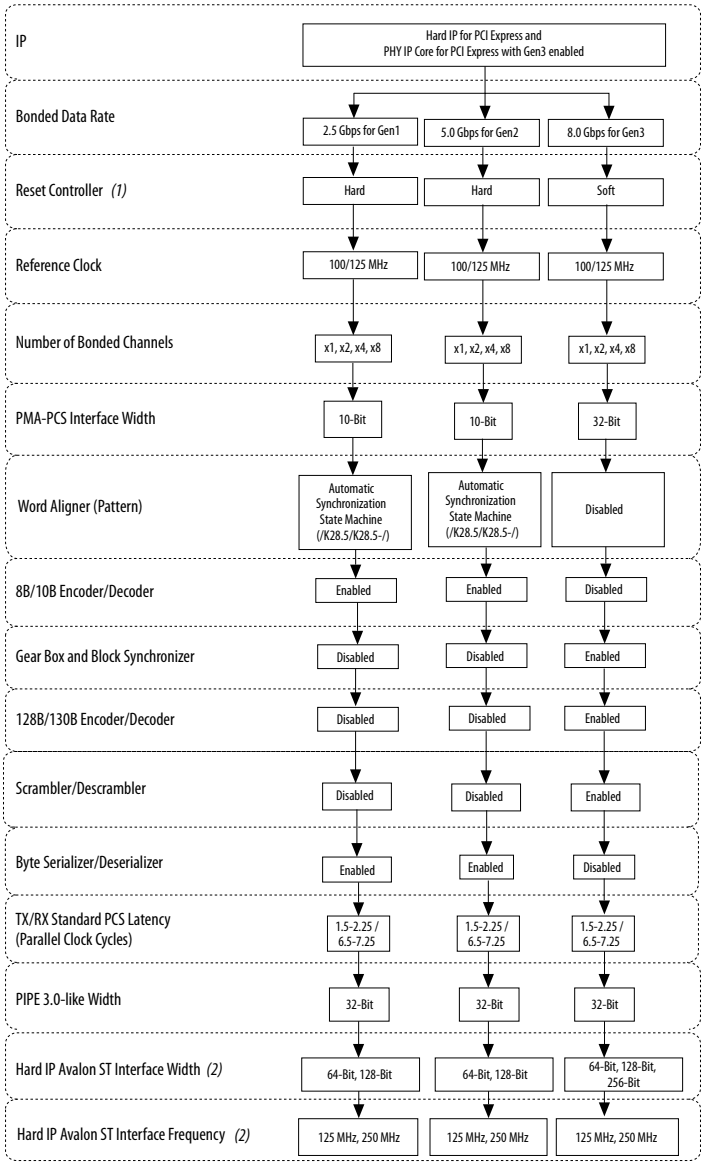


Figure 4-17: PCIe Gen1, Gen2, and Gen3 Hard IP and PHY IP Core for PCI Express Datapath Configuration

This transceiver datapath configuration is for a configuration with Gen3 enabled.

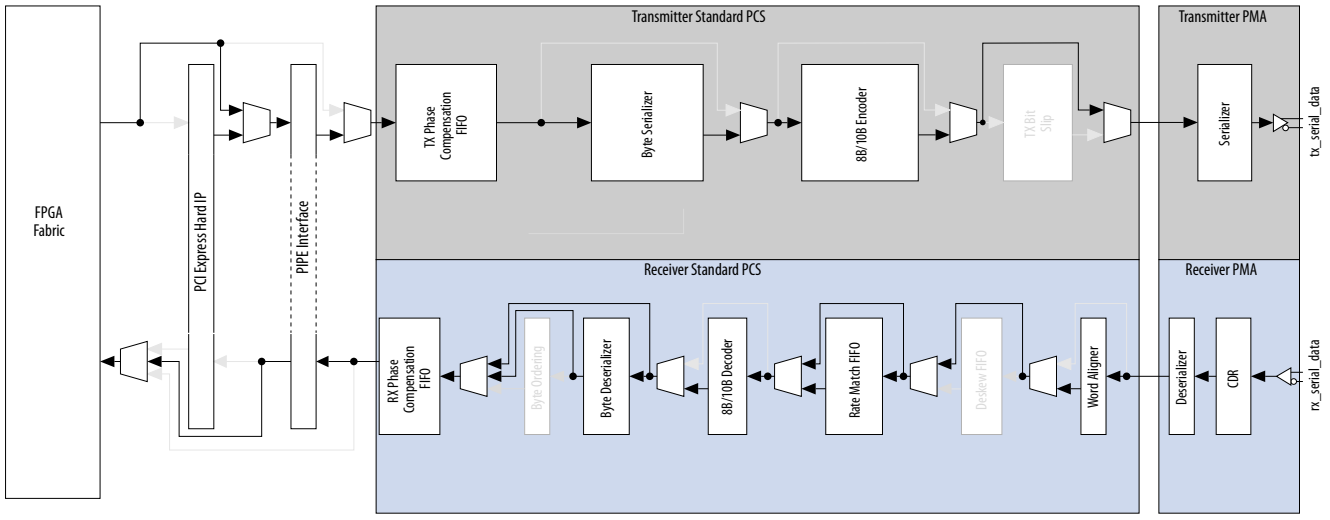


Notes:
(1) The PHY IP Core for PCI Express (PIPE configuration) employs the Embedded Reset Controller IP. It does not use the Hard or Soft Reset Controller employed in the Hard IP for PCI Express (HIP configuration).
(2) Does not apply to PHY IP Core for PCI Express configuration. Applies only to Hard IP for PCI Express configuration.

Transceiver Channel Datapath

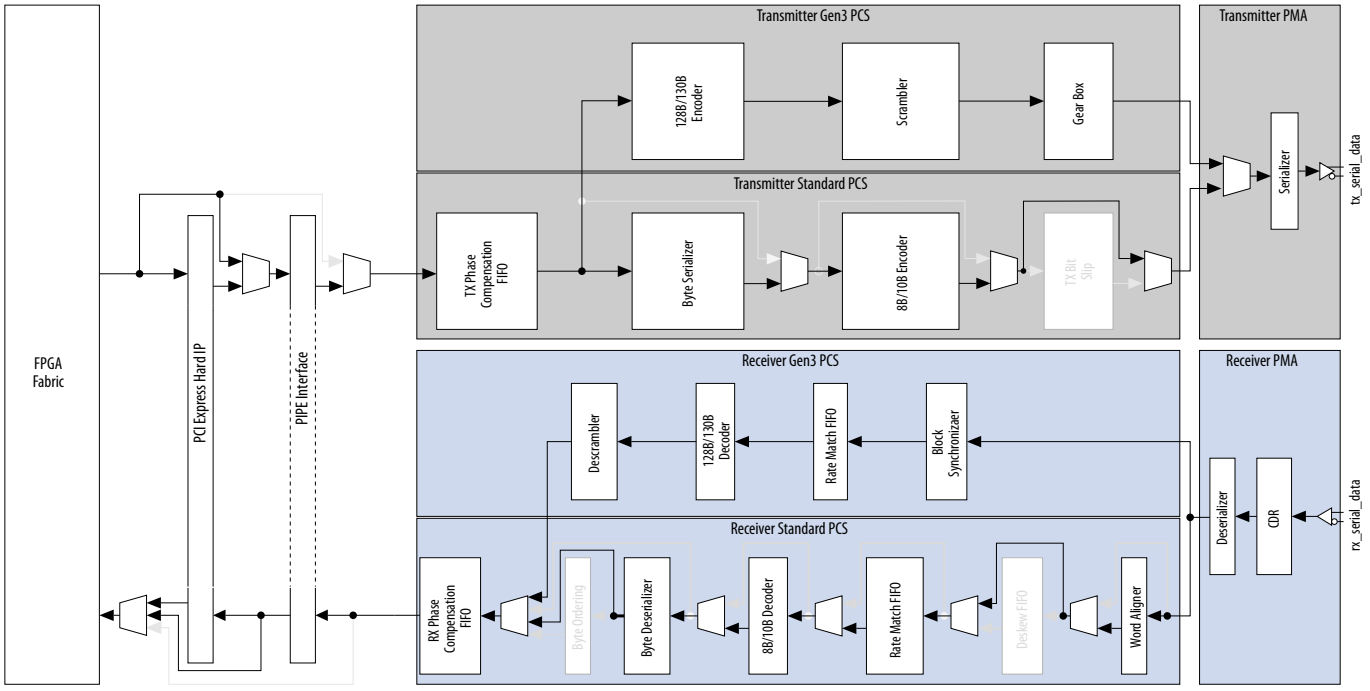
The following figure shows the Stratix V transmitter and receiver channel datapath for PCIe Gen1/Gen2 configurations when using PIPE configuration with Gen3 disabled. In this configuration, the transceiver connects to a PIPE 2.0 compliant interface.

Figure 4-18: Transceiver Channel Datapath for PCIe Gen1/Gen2 in PIPE Configuration with Gen3 Disabled



The following figure shows the Stratix V transmitter and receiver channel datapath for PCIe Gen1/Gen2/Gen3 configurations with a 32-bit PIPE 3.0-like interface and PCI Express Base Specification Version 3.0 is enabled.

Figure 4-19: Transceiver Channel Datapath for PCIe Gen1/Gen2/Gen3 Configurations



Related Information

Transceiver Architecture in Stratix V Devices

Supported Features for PCIe Configurations

The features supported for a PCIe configuration are different for the 2.5 Gbps, 5 Gbps, and 8 Gbps data rate configurations.

Table 4-6: Supported Features for PCIe Configurations

Feature	Gen1 (2.5 Gbps)	Gen2 (5 Gbps)	Gen3 (8 Gbps)
x1, x2, x4, x8 link configurations	Yes	Yes	Yes
PCIe-compliant synchronization state machine	Yes	Yes	Yes
±300 ppm (total 600 ppm) clock rate compensation	Yes	Yes	Yes
8-bit FPGA fabric-transceiver interface (PIPE 2.0)	Yes	—	—
16-bit FPGA fabric-transceiver interface (PIPE 2.0)	Yes	Yes	—
32-bit FPGA fabric-transceiver interface (PIPE 3.0-like)	—	—	Yes
64-bit Hard IP Avalon-ST interface width (Hard IP only)	Yes	Yes	Yes
128-bit Hard IP Avalon-ST interface width (Hard IP only)	Yes	Yes	Yes
256-bit Hard IP Avalon-ST interface width (Hard IP only)	—	Yes	Yes
Transmitter driver electrical idle	Yes	Yes	Yes
Receiver Detection	Yes	Yes	Yes
8B/10B encoder/decoder disparity control	Yes	Yes	—
128B/130B encoder/decoder	—	—	Yes
Power state management	Yes	Yes	Yes
Receiver PIPE status encoding (pipe_rxstatus[2:0])	Yes	Yes	Yes
Dynamic switching between 2.5 Gbps and 5 Gbps signaling rate	—	Yes	—
Dynamic switching between 2.5 Gbps, 5 Gbps, and 8 Gbps signaling rate	—	—	Yes
Dynamic transmitter margining for differential output voltage control	—	Yes	Yes
Dynamic transmitter buffer de-emphasis of -3.5 dB and -6 dB	—	Yes	Yes
Dynamic Gen3 transceiver pre-emphasis, de-emphasis, and equalization	—	—	Yes

PIPE 2.0 Interface

In a PCIe 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. The PIPE

configuration complies with the PIPE 2.0 specification. If you use a PIPE configuration, you must implement the PHY-MAC layer using soft IP in the FPGA fabric.

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

- Forcing the transmitter driver into the electrical idle state
- Initiating the receiver detect sequence
- Controlling the 8B/10B encoder/decoder
- Controlling the 128B/130B encoder/decoder
- Managing the PCIe power states
- Indicating the completion of various PHY functions
- Encoding the receiver status and error conditions on the `pipe_rxstatus[2:0]` signal, conforming to the PCIe PIPE 3.0 specification

Transceiver datapath clocking varies between non-bonded (x1) and bonded (x2, x4, and x8) configurations.

Dynamic Switching Between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signal Rates

In a PIPE configuration, the PIPE Parameter Editor provides an input signal (`pipe_rate`) that is functionally equivalent to the RATE signal specified in the PCIe specification. A low-to-high transition on this input signal (`pipe_rate`) initiates a data rate switch from Gen1 to Gen2. A high-to-low transition on the input signal initiates a data rate switch from Gen2 to Gen1. The signaling rate switch between Gen1 and Gen2 is achieved by changing the transceiver datapath clock frequency between 250 MHz and 500 MHz, while maintaining a constant, 16-bit width transceiver interface.

Transmitter Electrical Idle Generation

The PIPE interface block in Stratix V devices puts the transmitter buffer in the channel 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 2.0 for both PCIe Gen1 and Gen2 data rates.

The PCIe specification requires the transmitter driver to be in electrical idle in certain power states. For more information about input signal levels required in different power states, refer to “Power State Management”.

Power State Management

The PCIe specification defines four power states—P0, P0s, P1, and P2—that the physical layer device must support 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 Stratix V 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 the physical layer device to implement power saving measures. Stratix V transceivers do not implement these power saving measures except for putting 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 the Polling.Compliance substate. The Polling.Compliance substate is used to assess if the transmitter is electrically compliant with the PCIe voltage and timing specifications.

Receiver Electrical Idle Inference

The PCIe protocol allows inferring the electrical idle condition at the receiver instead of detecting the electrical idle condition with analog circuitry.

In all PIPE configurations, (x1, x2, x4, and x8), each receiver channel PCS has an optional Electrical Idle Inference module that implements the electrical idle inference conditions specified in the PCIe Base Specification 2.0.

Receiver Status

The PCIe specification requires the PHY to 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 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 conforms to the PCIe specification.

Receiver Detection

The PIPE interface block in Stratix V transceivers provides an input signal (`pipe_txdetectrx_loopback`) for the receiver detect operation required by the PCIe protocol during the Detect state 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 driver 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 when compared with the time constant of the step voltage when the receiver is not present. The receiver detect circuitry monitors the time constant of the step signal seen on the trace to determine if a receiver was detected. The receiver detect circuitry 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 transceiver on-chip termination must be used and 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 2.0.

The PIPE 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, as per the PIPE 2.0 specifications.

Gen1 and Gen2 Rate Match FIFO

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

PCIe Reverse Parallel Loopback

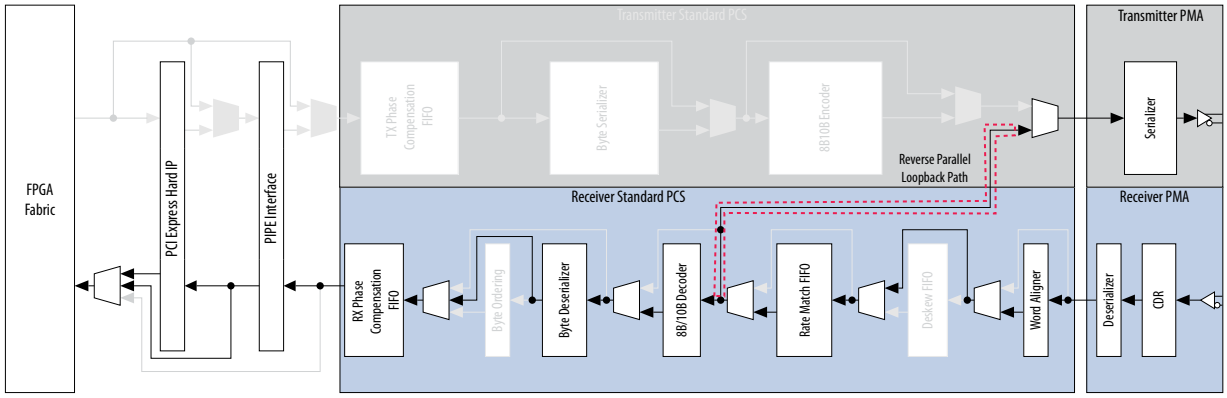
PCIe reverse parallel loopback is only available in a PCIe functional configuration for Gen1, Gen2, and Gen3 data rates. The received serial data passes through the receiver CDR, deserializer, word aligner, and rate matching FIFO buffer. The data is then looped back to the transmitter serializer and transmitted out

through the transmitter buffer. The received data is also available to the FPGA fabric through the port. This loopback mode is compliant with the PCIe specification 2.0. Stratix V devices provide an input signal to enable this loopback mode.

Note: This is the only loopback option supported in PIPE configurations.

Figure 4-20: PCIe Reverse Parallel Loopback Mode Datapath

The grayed-out blocks are Inactive.



Related Information

- **Transceiver Clocking and Channel Placement Guidelines** on page 4-34
- **Refer to the PHY IP Core for PCI Express (PIPE) chapter in the Altera Transceiver PHY IP Core User Guide**
- **Refer to the "Standard PCS Architecture" section in the Transceiver Architecture in Stratix V Devices chapter**
- **For the power state requirements when switching between Gen1 and Gen2 data rates, refer to the PCIe Base Specification 2.0.**

Supported Features for PCIe Gen3

The PCIe Gen3 hard PCS supports the Gen3 base specification. PCIe Gen3 operations can be configured using the Stratix V Hard IP for PCI Express IP or PHY IP Core for PCI Express.

In Stratix V Hard IP for PCI Express, selecting **PCIe Base Specification Version 3.0** or **PCI Express Base Specification Version 2.1** enables a 32-bit wide PIPE 3.0-like interface for Gen1, Gen2, and Gen3 operations.

In PHY IP Core for PCI Express, selecting Gen3 enables the 32-bit wide PIPE 3.0-like interface and selecting Gen1 or Gen2 enables the 16-bit/8-bit wide PIPE 2.0 interface for Gen1 and Gen2 operation.

Block Synchronization (Word Aligner)

The block synchronizer aligns the recovered serial data coming from the CDR to 130-bit word boundaries. The block synchronizer delineates the word boundaries by searching and identifying the Electrical IDLE Exit Sequence Ordered Set (EIEOS) or the Last FTS OS and SKP ordered set to correctly identify the word boundary from the incoming serial data stream. The block synchronizer continues to realign to a new block boundary following the receipt of an SKP ordered set because of varying word lengths.

Gen3 Rate Match FIFO

To accommodate PCIe protocol requirements and to compensate for clock frequency differences of up to ±300 ppm between source and termination equipment, receiver channels have a rate match FIFO. The rate match FIFO adds or deletes four SKP characters (32 bits) to keep the FIFO from becoming empty or full. It monitors the block synchronizer for a `skip_found` signal. If the rate match FIFO is almost full, the FIFO deletes four SKP characters. If the rate match FIFO is nearly empty, the FIFO inserts an SKP character at the start of the next available SKP ordered set.

128B/130B Encoder/Decoder

Unlike PCIe Gen1 and Gen2, the PCIe Gen3 encoder/decoder does not use 8B/10B encoding. The PCIe Gen3 encoder/decoder uses a 2-bit sync header and a 128-bit data word. The PCS encoder appends the two sync header bits to every 128 bits of data and enables scrambling for the data packets except for ordered set packets and the first symbol of a TS1/TS2 ordered set. The encoder/decoder continuously enables or disables scrambling, based on whether the payload being processed is an ordered set or a data packet. If an Electrical IDLE Exit Ordered Set or a Fast Training Sequence Ordered Set is received, the scrambler is reset to the initial seed value. The encoder/decoder also monitors the data stream for ordered set and sync header bit violations.

Gen3 Gear Box

The PCIe 3.0 base specification requires a block size of 130 bits with the exception of SKP ordered sets, which can be 66, 98, 130, 162, or 194 bits in length. The 130-bit block of data generated by the 128B/130B encoder and variable length SKP characters must be reordered in 32-bit parallel data segments that the PMA serializer can accept. The transceivers employ a gear box to accommodate this fractional bit difference between the 130-bit data word and a fixed 32-bit serialization PMA factor for Gen3.

Scrambler/Descrambler

Scrambling and descrambling are used during PCIe Gen3 operation to guarantee adequate transitions for the receiver in order to correctly regenerate the recovered clock. The 2-bit sync header bit, ordered set, and the first symbol of the TS1/TS2 ordered set are never scrambled.

PIPE 3.0-Like Gen3 Interface

PCIe Gen3 is a new feature added to the transceivers. The PCS supports PCI Express 3.0 base specification. The PIPE interface has been expanded to a 32-bit wide PIPE 3.0-like interface. The PIPE interface controls PHY functions such as transmission of electrical idle, receiver detection, and speed negotiation and control. In summary, the Gen3 PIPE 3.0-like interface block performs the following:

- Dynamic clock selection between Gen1, Gen2, and Gen3 speeds
- Gen3 auto speed negotiation (ASN)
- Controlling the 128B/130B encoder/decoder
- Gen3 Electrical Idle Entry and Exit detections/CDR Control Block
- Dynamic Gen3 and Gen2/Gen1 PCS data rate Auto Speed Negotiation
- Dynamic transceiver PMA data rate and PLL switching

Auto-Speed Negotiation Block

PCIe Gen3 mode enables ASN (auto-speed negotiation) between Gen1 (2.5 Gbps), Gen2 (5.0 Gbps), and Gen3 (8.0 Gbps) signaling data rates. The signaling rate switch is accomplished through frequency scaling and configuration of the PMA and PCS blocks using a fixed 32-bit wide PIPE 3.0-like Interface.

The PMA switches clocks between Gen1, Gen2, and Gen3 data rates in a glitch-free manner. For a non-bonded x1 channel, an ASN module facilitates speed negotiation in that channel. For bonded x2, x4, and x8 channels, the ASN module selects the master channel to control the rate switch. The master channel distributes the speed change request to the other PMA and PCS channels.

Table 4-7: PIPE Gen3 32-Bit PCS Clock Rates

PCIe Gen3 Capability Mode Enabled	Gen1	Gen2	Gen3
Lane data rate	2.5G	5G	8G
PCS clock frequency	250 MHz	500 MHz	250 MHz
FPGA Core IP clock frequency	62.5 MHz	125 MHz	250 MHz
PIPE interface width	32-bit	32-bit	32-bit
Rate[1:0]	00	01	10

The PCIe Gen3 speed negotiation process is initiated by writing a 1 to bit 5 of the Link Control register of the root port, causing a PIPE rate signal change from the hard IP. The ASN then places the PCS in reset, dynamically shuts down the clock paths to disengage the current active state PCS (either Standard PCS or Gen3 PCS). If a switch to or from Gen3 is requested, the ASN automatically selects the correct PCS clock paths and datapath selection in the multiplexers. The ASN block then sends a request to the PMA block to switch the data rate change and waits for a rate change done signal for confirmation. When the PMA completes the rate change and sends confirmation to the ASN block, ASN enables the clock paths to engage the new PCS block and releases the PCS reset. Successful completion of this process is indicated by assertion of the `pipe_phystatus` signal by the ASN block to the hard IP block.

Note: In PHY IP Core for PCI Express configuration, the Core IP must set the values to `pipe_rate[1:0]` to initiate the transceiver datarate switch sequence.

Note: When you switch speeds to either Gen2 or Gen3, hold the LTSSM steady for 700 µs in Recovery.RCVRLOCK. The `rx_is_lockedtodata` signal from the CDR must be stable during this time. The PHY MAC interface should not look at `rxvalid` during this time because its contents may be invalid.

Transmitter Electrical IDLE Generation

The PIPE 3.0-like interface under the control of the hard IP block in Hard IP for PCIe or the user Core IP in PHY IP Core for PCIe may place the transmitter in electrical idle during low power states and the ASN process. Before the transmitter enters electrical idle, the HIP sends an electrical idle order set (EIOS) to the PHY. For Gen1 and Gen2, the order set format is COM, IDL, IDL, IDL. For Gen3, the order set format consists of 16 symbols with value 0x66.

During electrical idle, the transmitter differential and common mode voltage levels are compliant to the PCIe Base Specification 3.0.

Receiver Electrical IDLE Inference

If there is no activity on the link for a period of time or during the ASN process, the Inferring Electrical Idle condition is detected by the receiver PHY. These conditions are specified according to Table 4-11 of the PCI Express Base Specification, Rev 3.0.

Gen3 Power State Management

The PCIe base specification defines low power states for PHY layer devices to minimize power consumption. The Gen3 PCS does not implement these power saving measures, except when placing the

transmitter driver in electrical idle state in the low power states. In P2 low power state, the transceivers do not disable the PIPE block clock.

CDR Control Block

The CDR control block controls the PMA CDR to obtain bit and symbol alignment and deskew within the allocated time, and generates status signals for other PCS blocks. The PCIe base specification requires that the receiver L0s power state exit time be a maximum of 4 ms for Gen1, 2 ms for Gen2, and 4 ms for Gen3 signaling rates. The transceivers have an improved CDR control block to accommodate fast lock times when the CDR must relock to the new multiplier/divider settings when entering or exiting Gen3 speeds.

Transceiver Clocking and Channel Placement Guidelines

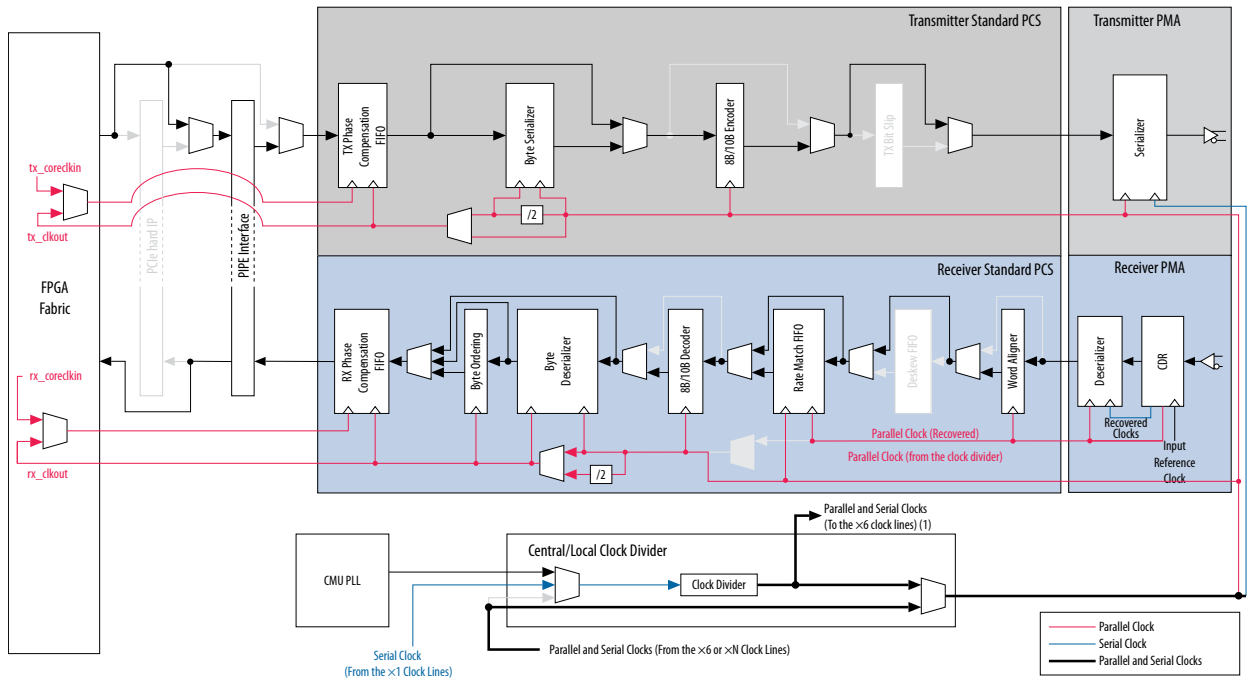
This section describes the transceiver clocking for Gen1 and Gen2 Hard IP and PIPE configurations. The channel placement guidelines are only described for Gen1 and Gen2 PIPE configuration. The channel placement guidelines for Gen1 and Gen2 Hard IP configuration are not included.

Transceiver Clocking for PCIe Gen1 and Gen2

PIPE x1 Configuration

The high-speed serial clock is provided by the CMU PLL in a channel different from that of the data channel. The local clock divider block in the data channel generates a parallel clock from this high-speed clock and distributes both clocks to the PMA and PCS of the data channel.

Figure 4-21: Transceiver Clocking in a Gen1/Gen2 PIPE x1 Configuration

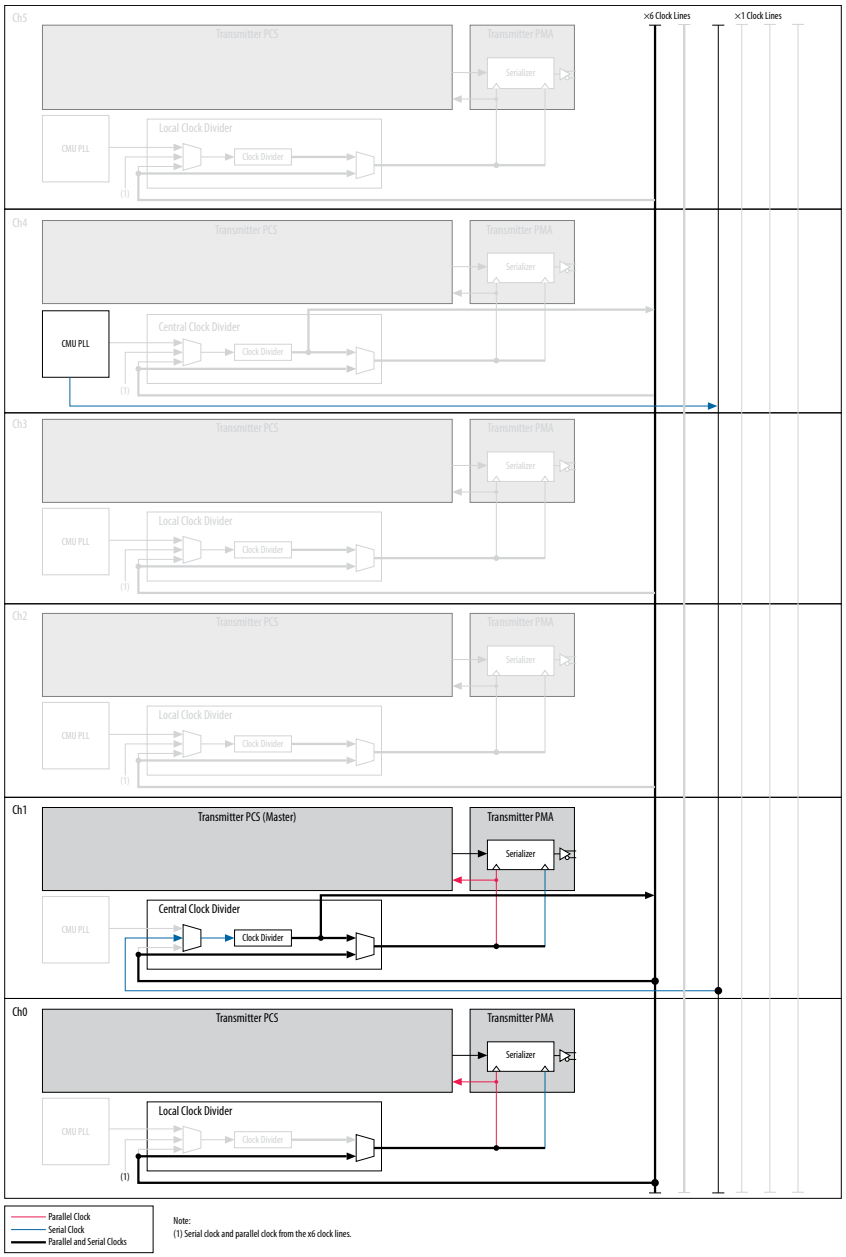


PIPE x2 Configuration

In a PIPE x2 bonded configuration, clocking within the PCS is independent for each receiver channel. Clocking is bonded only for transmitter channels, while the control signals are bonded for both

transmitter and receiver channels. The Quartus II software automatically places the transmit CMU PLL and master channel in either channel 1 or channel 4 within a transceiver bank

Figure 4-22: Transmitter Clocking in a Gen1/Gen2 PIPE x2 Configuration



PIPE x4 Configuration

In a PIPE x4 bonded configuration, clocking within the PCS is independent for each receiver channel. Clocking is bonded only for transmitter channels, while the control signals are bonded for both transmitter and receiver channels. The Quartus II software automatically places the transmit CMU PLL and master channel in either channel 1 or channel 4 within a transceiver bank.

Figure 4-23: Transmitter Clocking in a Gen1/Gen2 PIPE x4 Configuration

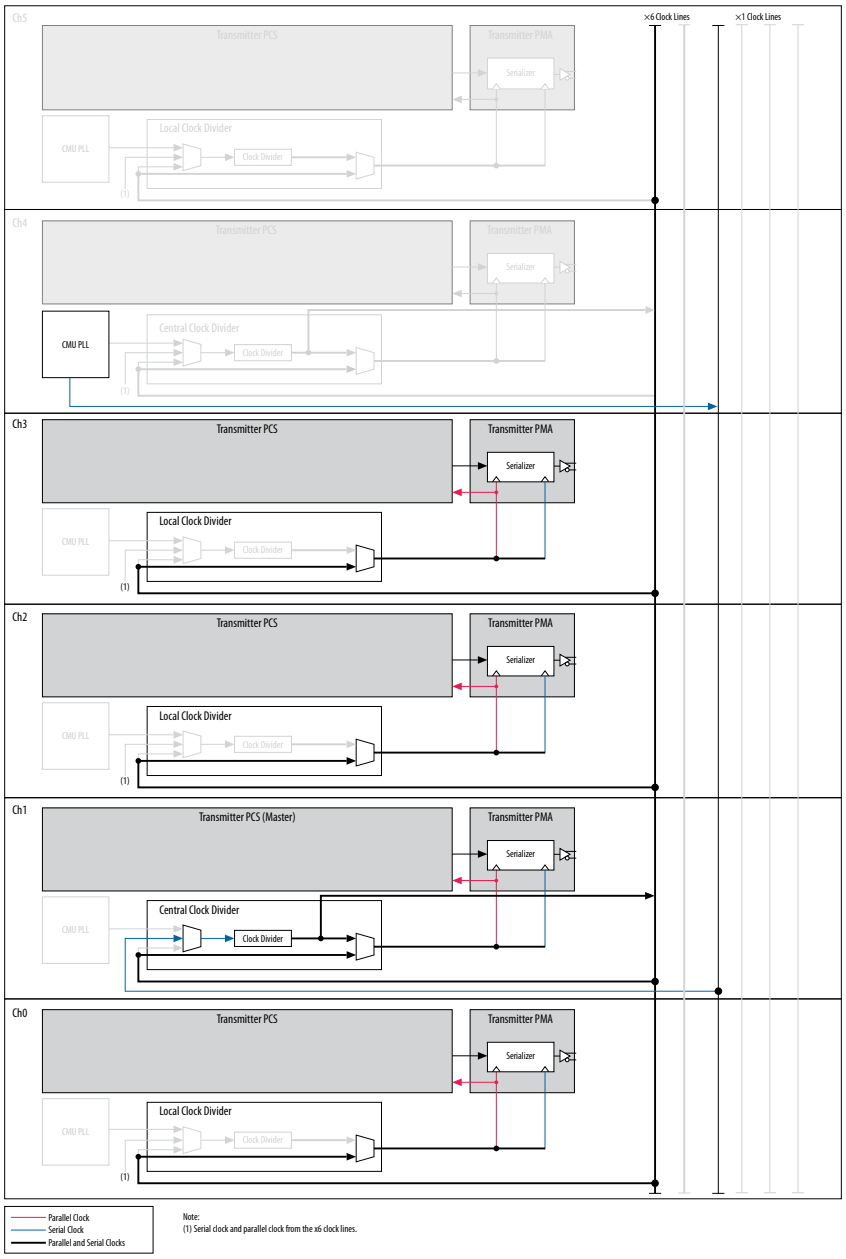
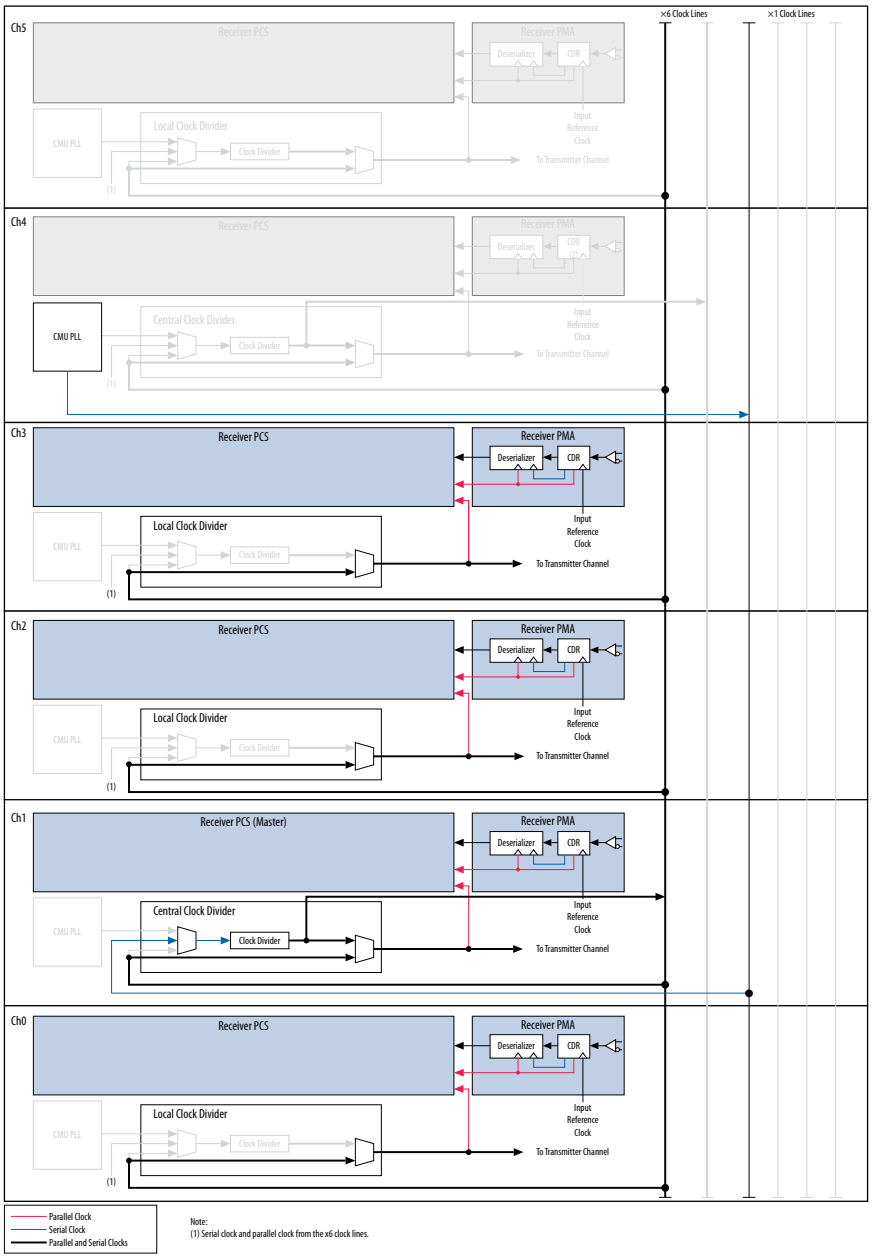


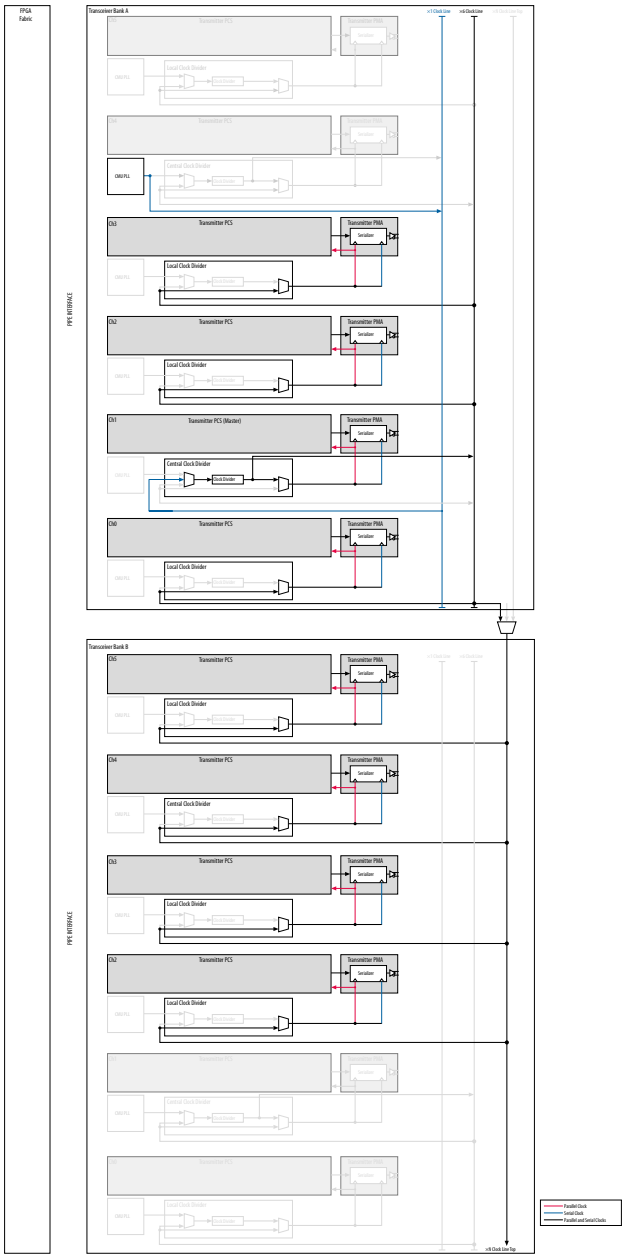
Figure 4-24: Receiver Clocking in a Gen1/Gen2 PIPE x4 Configuration



PIPE x8 Configuration

In the x8 PCIe bonded configuration, clocking is independent for receiver channels. Clocking and control signals are bonded only for transmitter channels.

Figure 4-25: Transceiver Clocking in a Gen1/Gen2 PIPE x8 Configuration



Transceiver Channel Placement Guidelines for Gen1, Gen2, and Gen3 PIPE Configurations

Note: The channel placement guidelines are only described for Gen1, Gen2, and Gen3 x1, x2, x4, and x8 PIPE configurations. The channel placement guidelines for Gen1, Gen2, and Gen3 Hard IP configuration are not included.

The following table lists the physical placement of PIPE channels in x1, x2, x4, and x8 bonding configurations. The Quartus® II software automatically places the CMU PLL in a channel different from that of the data channels.

Table 4-8: PIPE Configuration Channel Placement

Placement by the Quartus II software may vary with design, thus resulting in higher channel usage.

Configuration	Data Channel Placement	Channel Utilization Using CMU PLL in Gen1 and Gen2	Channel Utilization Using ATX PLL in Gen1 and Gen2	Channel Utilization Using CMU and ATX PLL in Gen3
x1	Any channel	2	1	2
x2	Contiguous channels	3	2	3
x4	Contiguous channels	5	4	5
x8	Contiguous channels	9	8	9

Channel Placement for Gen1, Gen2, and Gen3 x1 PIPE Configuration

For PIPE x1 configurations, the channel can be placed anywhere within a transceiver bank that contains the transmitter PLL. In Gen1 and Gen2 configurations, you can select either the ATX PLL or the CMU PLL as the transmitter PLL. In Gen3 configurations, a CMU PLL is used for Gen1 and Gen2 datarates and an ATX PLL is used for Gen3 datarates.

Channel Placement for Gen1, Gen2, and Gen3 x2 and x4 PIPE Configuration

The following two figures show examples of channel placement for PIPE x2 and x4 configurations. In a PIPE x2 or x4 configuration, the two or four channels must be contiguous and within the same transceiver bank, but they can be placed in any order as long as Logical Lane 1 is placed on the master channel. In Gen1 and Gen2 configurations, you can select either the ATX PLL or the CMU PLL as the transmitter PLL. In Gen3 configurations, a CMU PLL is used for Gen1 and Gen2 datarates and an ATX PLL is used for Gen3 datarates. The CMU PLL and/or ATX PLL must be within the same transceiver bank as the master channel.

In the figures, channels shaded in blue provide the transmit CMU PLL generating the high-speed serial clock. Channels shaded in gray are data channels. The Quartus II software automatically selects one of the following within a transceiver bank:

- The CMU PLL in either channel 1 or channel 4.
- The upper or lower ATX PLL if the ATX PLL is selected as the transmitter PLL within the transceiver bank containing the master channel.

Gen3 channel placement requires both a CMU and an ATX PLL in the same transceiver bank as the master channel.

Figure 4-26: Example of PIPE x2 Gen1, Gen2, and Gen3 Channel Placement Using an ATX PLL, a CMU PLL, or Both

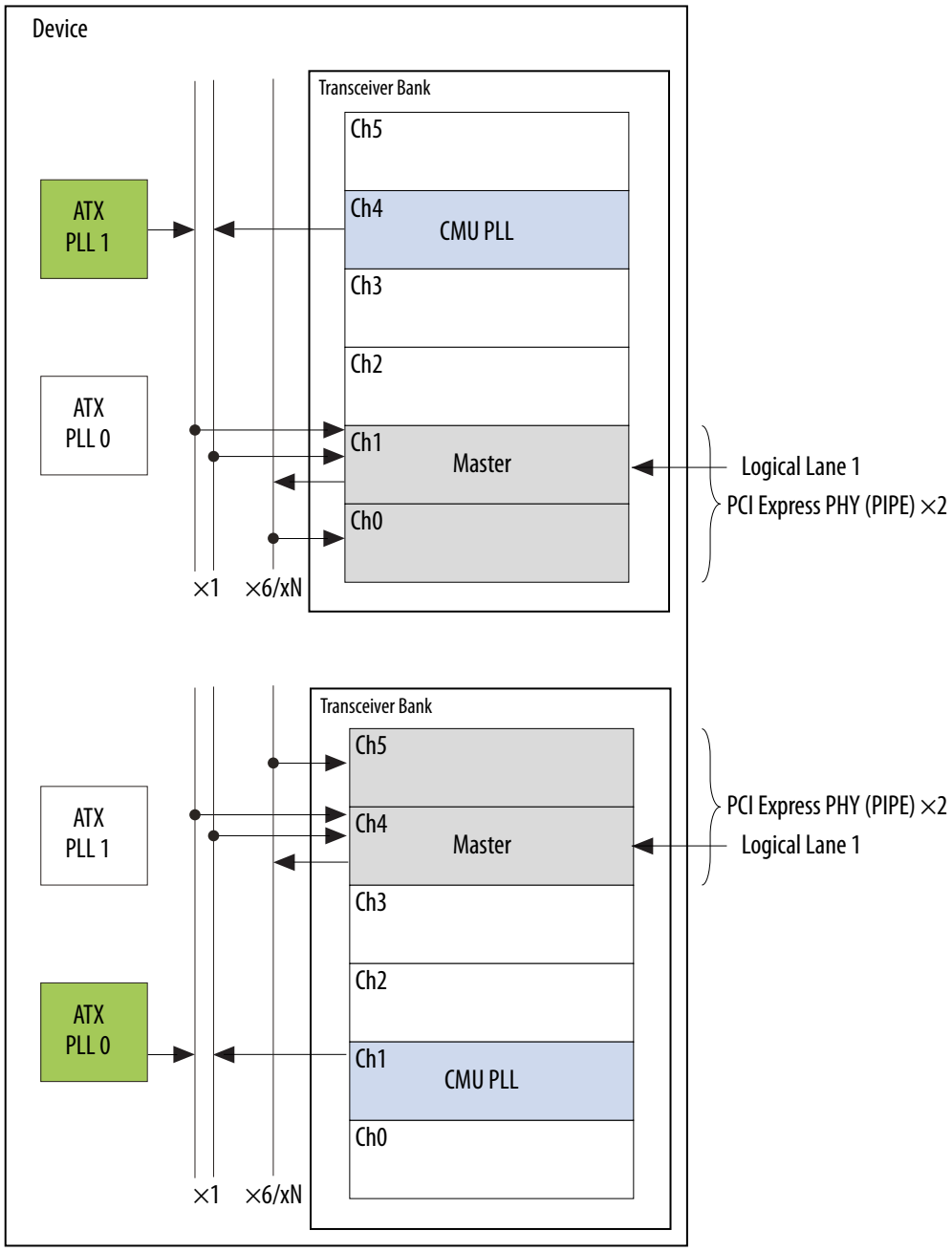
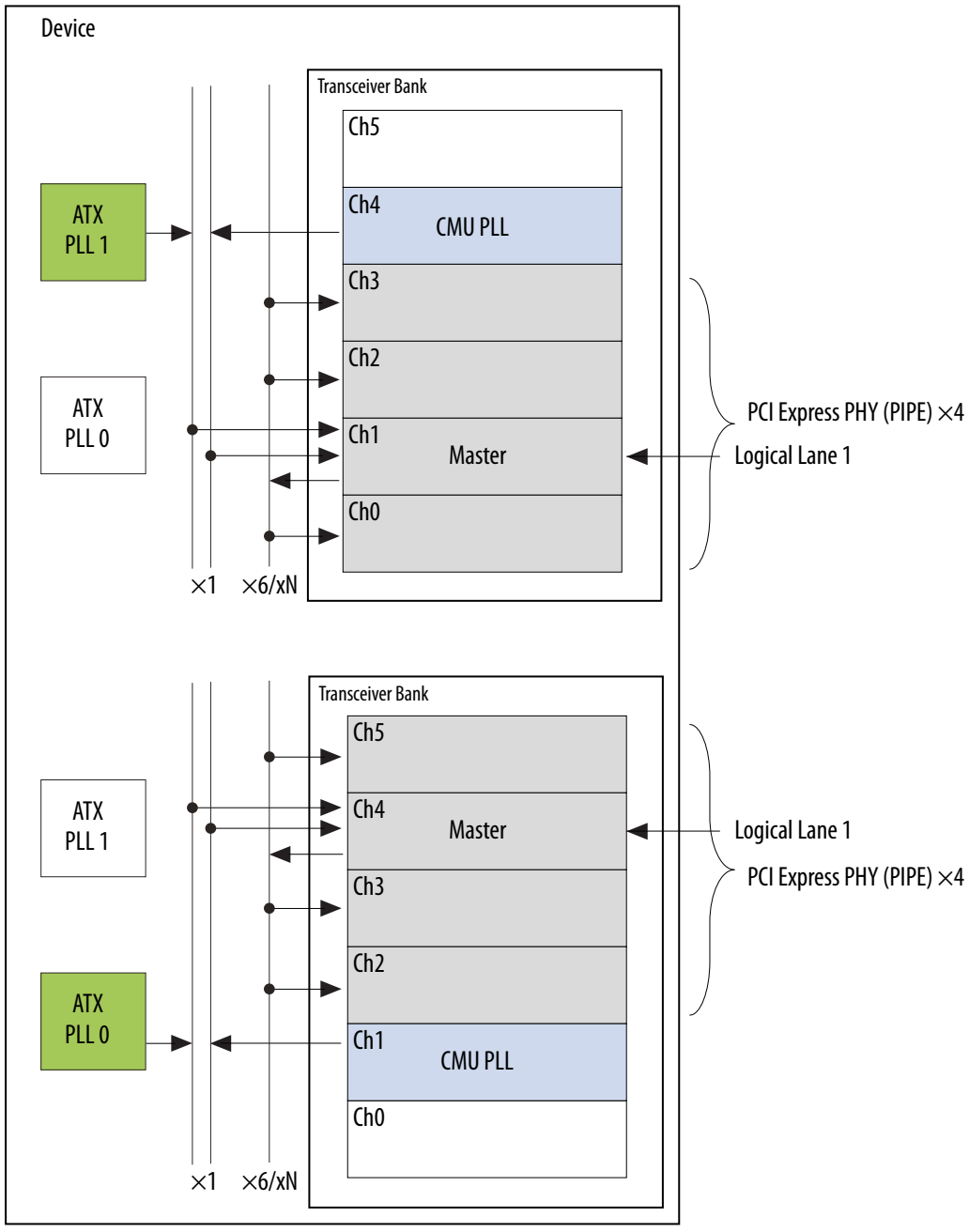


Figure 4-27: Example of PIPE x4 Gen1, Gen2, and Gen3 Channel Placement Using an ATX PLL, a CMU PLL, or Both

Channels shaded in blue provide the transmit CMU PLL generating the high-speed serial clock. Channels shaded in gray are data channels. The Quartus II software automatically selects the CMU PLL in either channel 1 or channel 4 within a transceiver bank. Gen3 channel placement requires an additional ATX PLL in the same transceiver bank as the master channel.



Channel Placement for Gen1, Gen2, and Gen3 x8 PIPE Configuration

In a PIPE x8 configuration, the eight channels must be contiguous, but they can be placed in any order as long as Logical Lane 0 is placed on the master channel.

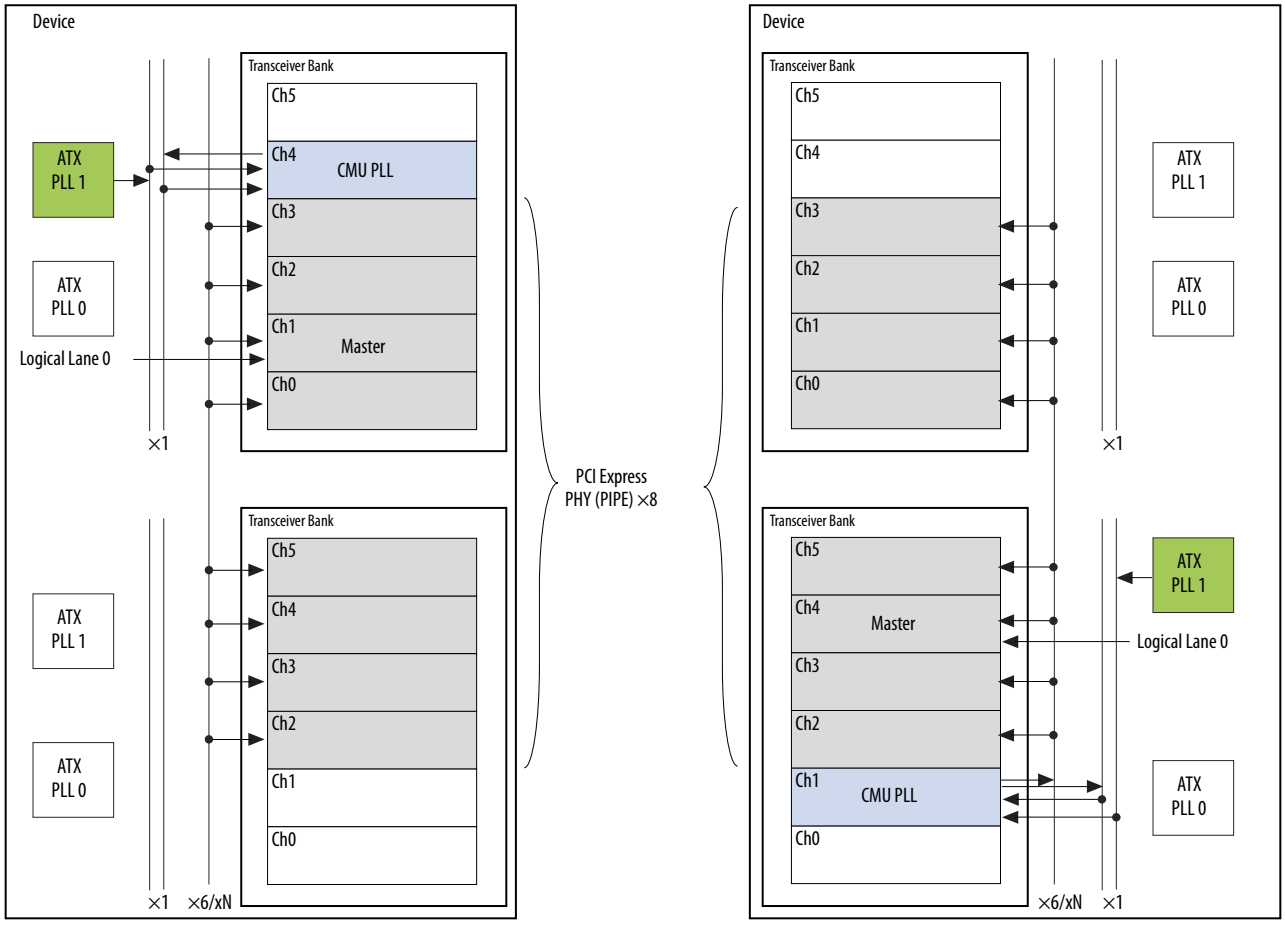
The Quartus II software automatically selects one of the following within a transceiver bank:

- The CMU PLL in either channel 1 or channel 4.
- The upper or lower ATX PLL if the ATX PLL is selected as the transmitter PLL within the transceiver bank containing the master channel.

In Gen1 and Gen2 configurations, you can select either the ATX PLL or the CMU PLL as the transmitter PLL. In Gen3 configurations, a CMU PLL is used for Gen1 and Gen2 datarates and an ATX PLL is used for Gen3 datarates. The CMU PLL and/or ATX PLL must be within the same transceiver bank.

Figure 4-28: Example of PIPE x8 Gen1, Gen2, and Gen3 Channel Placement Using an ATX PLL, a CMU PLL, or Both

Channels shaded in blue provide the transmit CMU PLL generating the high-speed serial clock. Channels shaded in gray are data channels. Gen3 channel placement requires both a CMU and ATX PLL in the same transceiver bank as the master channel.



Related Information

For channel placement guidelines for PCIe hard IP configuration using the Hard IP for PCI Express, refer to the Stratix V Hard IP for PCI Express User Guide.

Advanced Channel Placement Guidelines for PIPE Configurations

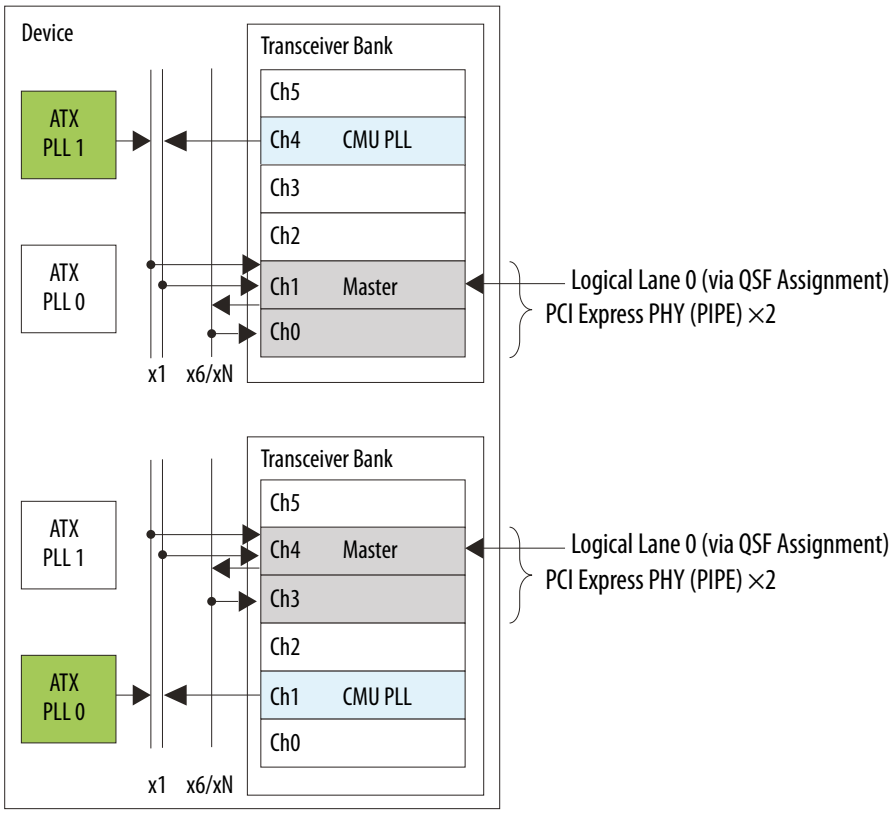
Advanced channel placement options for PIPE configurations are enabled through Quartus Settings File (QSF) assignments. A QSF assignment allows you to override the master channel assignment. By using a

QSF assignment, master channels can be assigned any logical channel number instead of the default Quartus II logical lane assignment. Any PIPE channel placement can also be made compatible with the HIP configuration channel placement.

In the following figures, channels shaded in blue provide the transmit CMU PLL generating the high-speed serial clock. Channels shaded in gray are data channels. An ATX PLL shaded in green can be substituted for the CMU PLL for Gen1 and Gen2 configurations only. Gen3 channel placement requires both the CMU PLL for Gen1/Gen2 datarates and the ATX PLL for Gen3 datarates to be located in the same transceiver bank as the master channel. The Quartus II software automatically selects the CMU PLL in either channel 1 or channel 4 and/or the upper or lower ATX PLL within a transceiver bank.

Advanced Channel Placement for PIPE x2 Gen1, Gen2, and Gen3 Configurations

Figure 4-29: PIPE x2 Gen1, Gen2, and Gen3 Advanced Channel Placement Using CMU and/or ATX PLL



Advanced Channel Placement for PIPE x4 Gen1, Gen2, and Gen3 Configurations

Figure 4-30: PIPE x4 Gen1, Gen2, and Gen3 Advanced Channel Placement Using CMU and/or ATX PLL in the Same Transceiver Bank

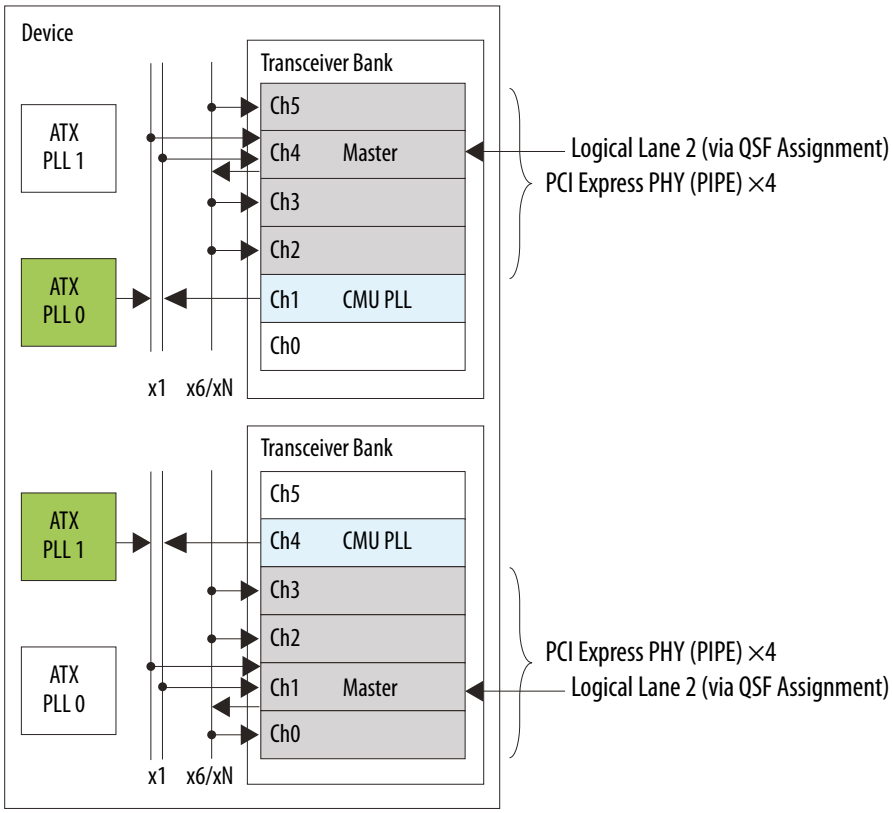


Figure 4-31: PIPE x4 Gen1, Gen2, and Gen3 Advanced Channel Placement Using CMU and/or ATX PLL Across Two Transceiver Banks – example 1

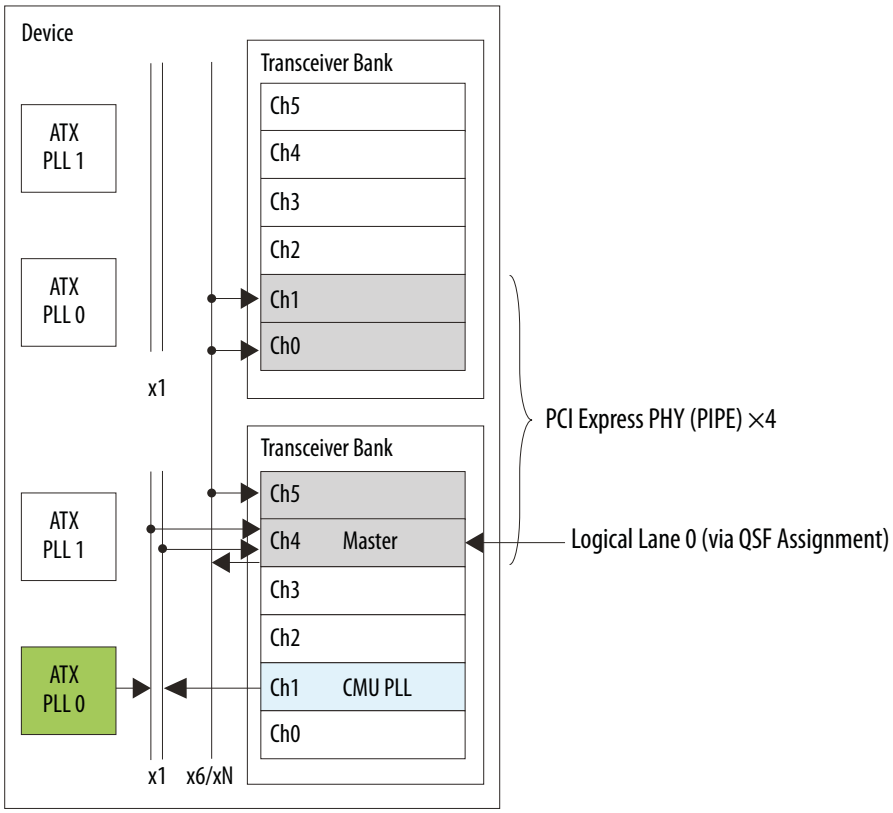
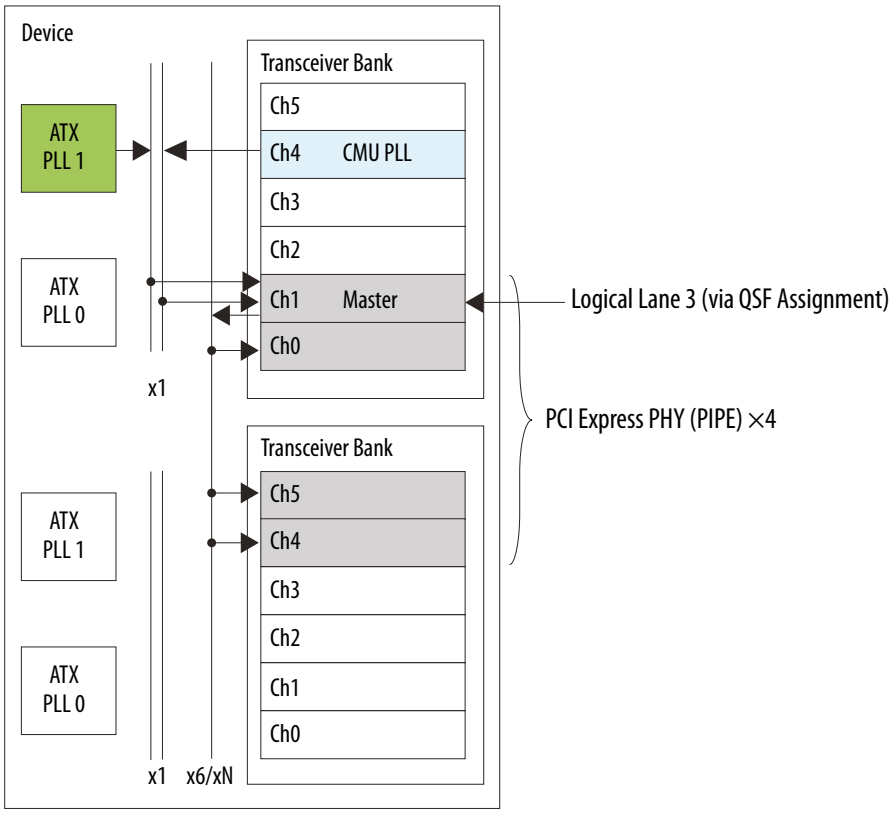


Figure 4-32: PIPE x4 Gen1, Gen2, and Gen3 Advanced Channel Placement Using CMU and/or ATX PLL Across Two Transceiver Banks – example 2



Advanced Channel Placement for PIPE x8 Gen1, Gen2, and Gen3 Configurations

For PCIe x8 advanced channel placement where the master channel resides between the contiguous data channel assignments, a second QSF assignment is required that allows the master channel to be placed between data channels.

For a HIP-compatible PCIe x8 channel placement, the master channel must be assigned logical channel 4 in the lower transceiver bank and the second QSF assignment for the reserve channel that allow master channel placement between contiguous data channel assignments are required.

Figure 4-33: PIPE x8 Gen1, Gen2, and Gen3 Advanced Channel Placement That is Compatible with HIP x8 Channel Placement

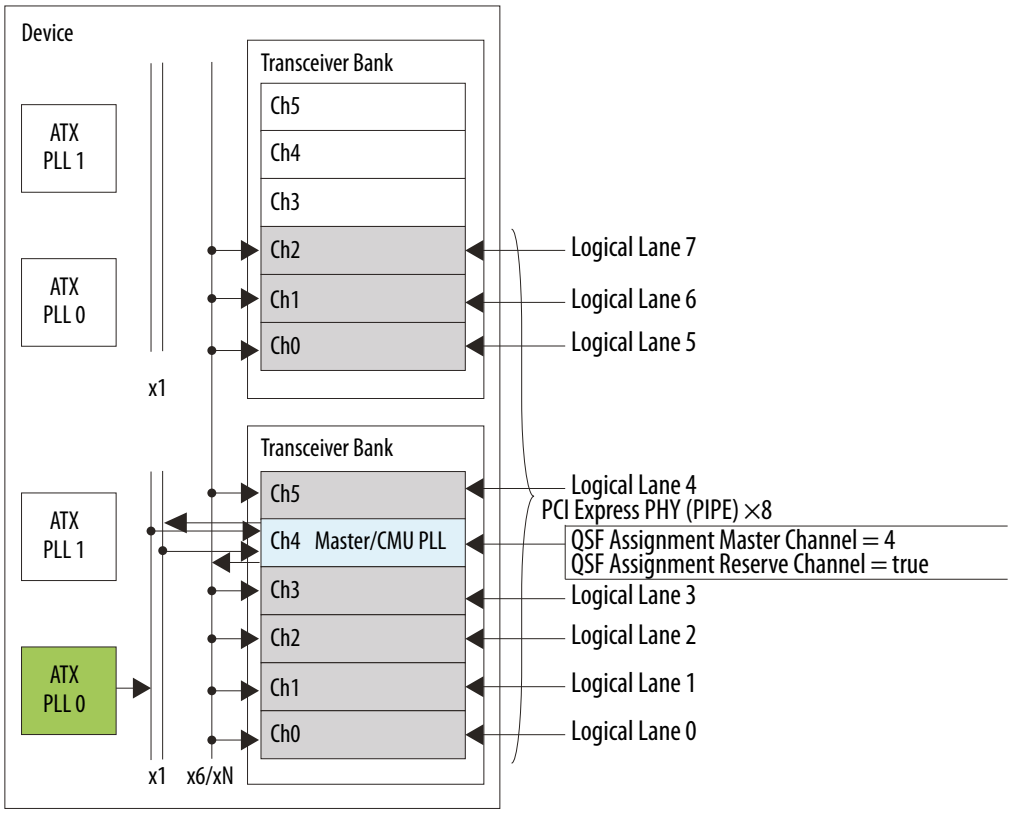
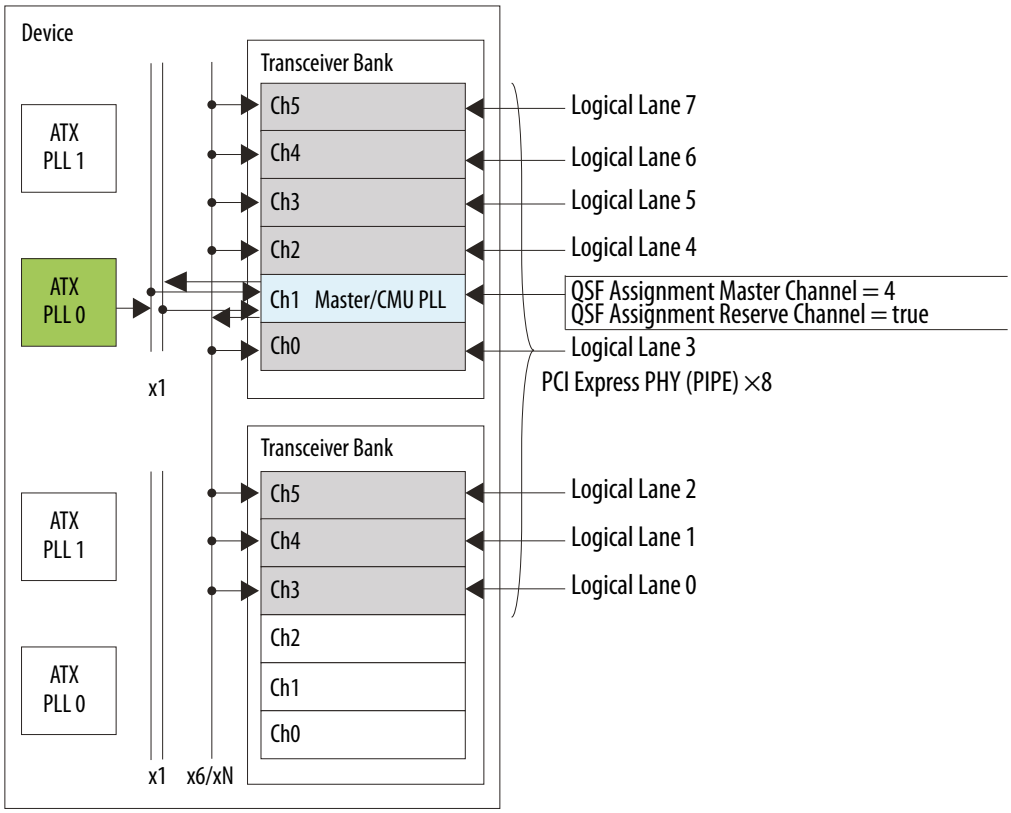


Figure 4-34: PIPE x8 Gen1, Gen2, and Gen3 Advanced Channel Placement That is Not Compatible with HIP x8 Channel Placement



The following figures show PIPE x8 Gen1, Gen2, and Gen3 advanced channel placement that requires only a master channel QSF assignment.

Figure 4-35: PIPE x8 Gen1, Gen2, and Gen3 Advanced Channel Placement – example 1

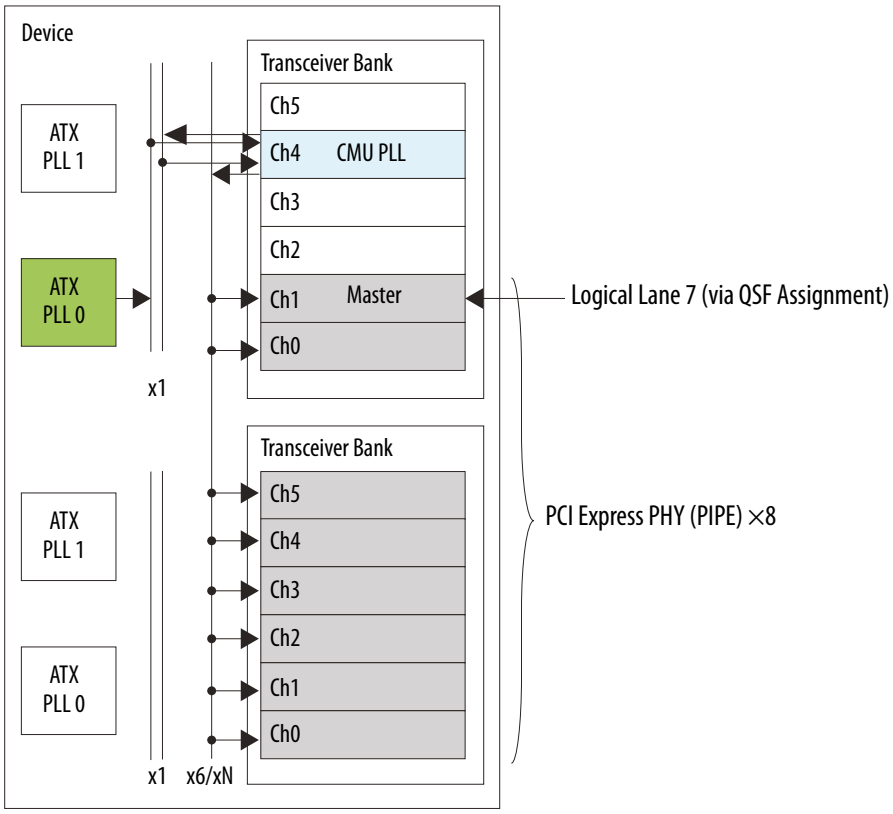


Figure 4-36: PIPE x8 Gen1, Gen2, and Gen3 Advanced Channel Placement – example 2

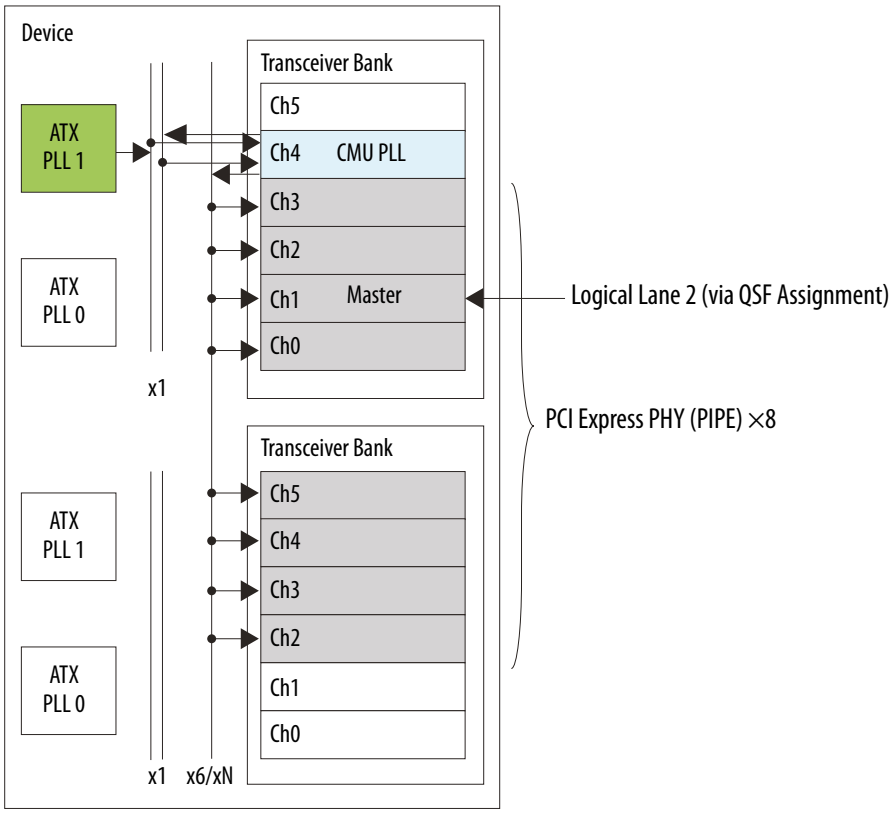
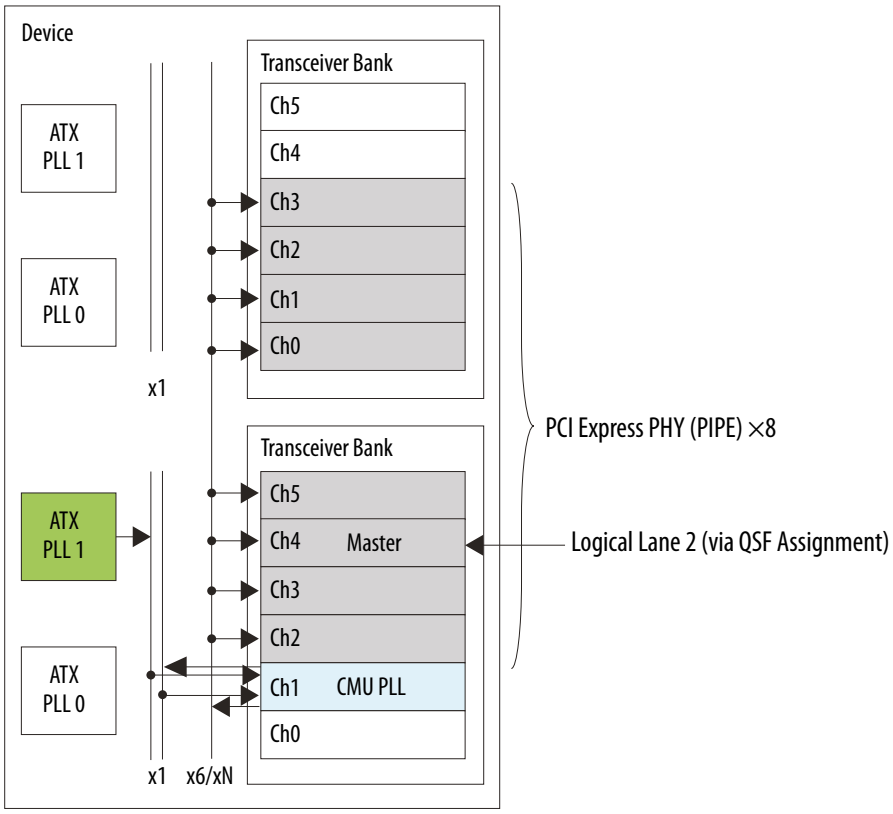


Figure 4-37: PIPE x8 Gen1, Gen2, and Gen3 Advanced Channel Placement – example 3



Transceiver Clocking for PCIe Gen3

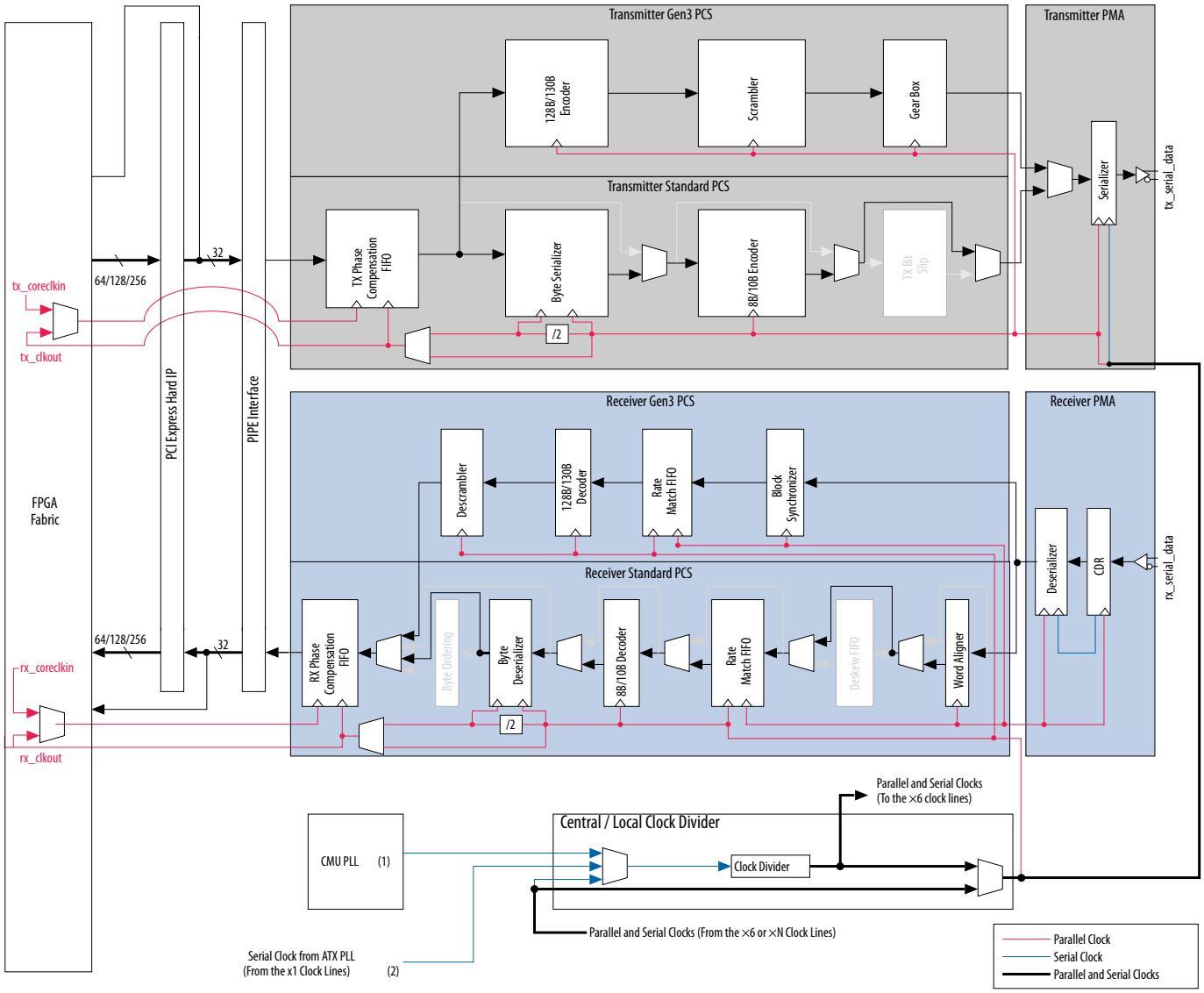
This section describes the transceiver clocking topology for both the PCIe Gen3 Hard IP and PIPE configuration.

In a PCIe x1, x2, x4, and x8 Gen3 Mode, both a channel PLL (CMU PLL) from transceiver physical channel 1 or 4 of the transceiver bank and either the top or bottom ATX PLL are used to generate the high-speed serial clock and support ASN. The CMU PLL supports Gen1 and Gen2 data rates while the ATX PLL supports Gen3 data rates. To enable rapid switching between Gen1, Gen2, and Gen3 data rates, a multiplexer selects either the free running CMU PLL for Gen1 and Gen2 data rates or the free running ATX PLL for Gen3 data rates. PLL reconfiguration is not used to support ASN.

Gen3 x1 Configuration

Figure 4-38: Transceiver Clocking in a Gen1/Gen2/Gen3 PCIe x1 Hard IP and PIPE Configuration

For Gen1 and Gen2, use the CMU PLL. For Gen3, use the ATX PLL.



For **PCIe x1 Gen3** using Hard IP configuration, the CMU PLL (transceiver physical channel 1) and the bottom ATX PLL of the transceiver bank are configured to generate the high-speed serial clock for the transmitter datapath clock and the rate matcher side of the FIFO in the receiver datapath if rate matching is enabled for the data channel. Two transceiver channels are needed to implement PCIe x1 Gen3, one for the data channel and one for the CMU PLL. The local clock divider block in the data channel generates a parallel clock from this high-speed serial clock and distributes both clocks to the PMA and PCS of the data channel.

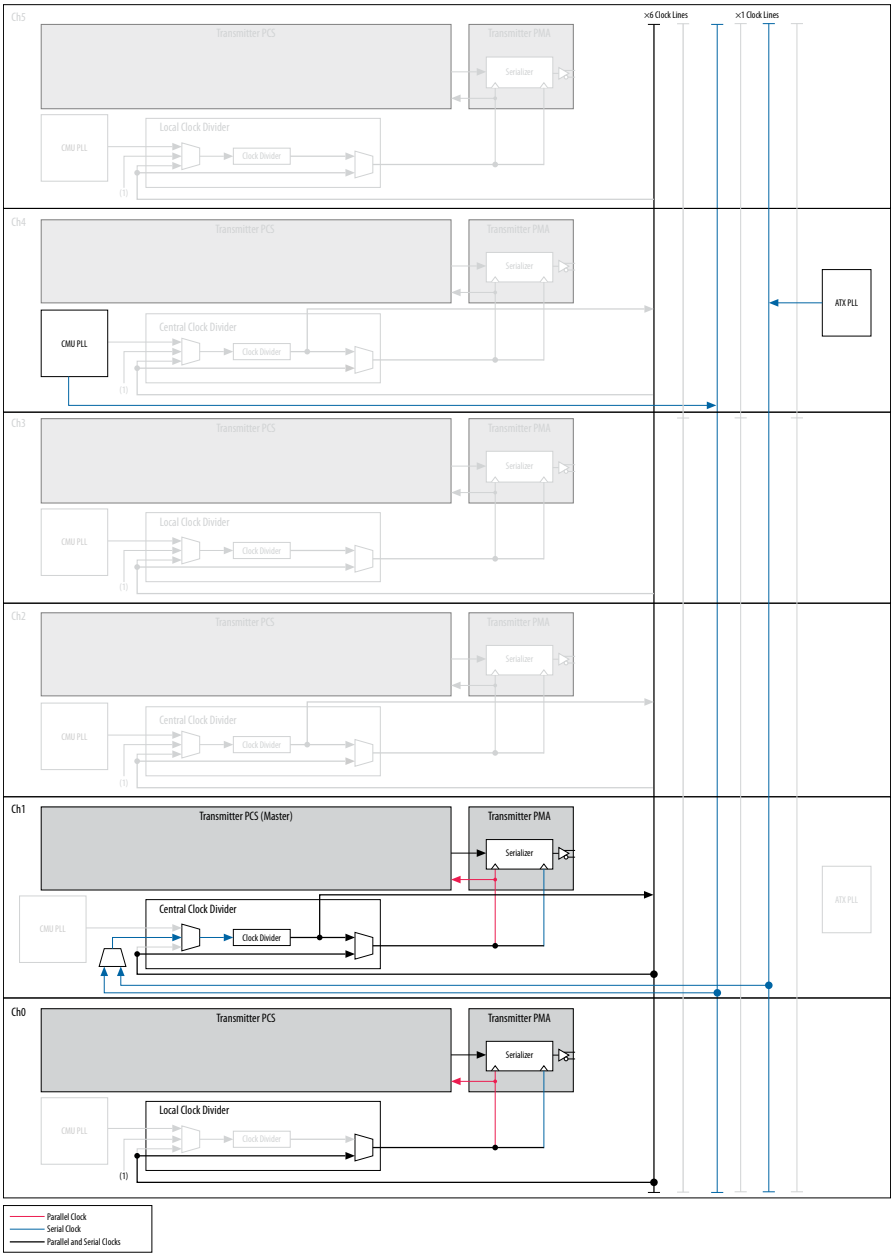
For **PCIe x1 Gen3** using PIPE configuration, the CMU PLL (transceiver physical channel 1 or 4) and the top or bottom ATX PLL of the transceiver bank are configured to generate the high-speed serial clock for the transmitter datapath clock and the rate matcher side of the FIFO in the receiver datapath if rate matching is enabled for the data channel. Two transceiver channels are needed to implement PCIe x1

Gen3, one for the data channel and one for the CMU PLL. The local clock divider block in the data channel generates a parallel clock from this high-speed serial clock and distributes both clocks to the PMA and PCS of the data channel.

Gen3 x2 Configuration

Figure 4-39: Transmitter Clocking in a Gen1/Gen2/Gen3 PCIe x2 Hard IP and PIPE Configuration

Unlike the Hard IP configuration, the PIPE configuration has the additional flexibility of using the top four transceiver channels in a transceiver bank or spanning the four lanes across two banks.



For **PCIe x2 Gen3** using Hard IP configuration, the CMU PLL (transceiver physical channel 4) and the top ATX PLL of the transceiver bank are configured to generate the high-speed serial clock. A total of three transceiver channels are required to implement PCIe x2 Gen3, including two data channels and one

channel for the CMU PLL. The Quartus II software automatically selects channel 1 in the transceiver bank as the master channel. Channel 1 bonds and drives all the transmitter datapath's clocking and the rate matcher side of the FIFO in the receiver datapaths if rate matching is enabled for the two data channels. The local clock divider block in each data channel generates the parallel clock from the high-speed serial clock and distributes both clocks to the PMA and PCS of that data channel.

For **PCIe x2 Gen3** using PIPE configuration, the CMU PLL (transceiver physical channel 1 or 4) and the top or bottom ATX PLL of the transceiver bank are configured to generate the high-speed serial clock. A total of three transceiver channels are required to implement PCIe x2 Gen3, including two data channels and one channel for the CMU PLL. The Quartus II software automatically selects either channel 1 or 4 in the transceiver bank as the master channel. Channel 1 or 4 bonds and drives all the transmitter datapath's clocking and the rate matcher side of the FIFO in the receiver datapaths if rate matching is enabled for the two data channels. The local clock divider block in each data channel generates the parallel clock from the high-speed serial clock and distributes both clocks to the PMA and PCS of that data channel.

Gen3 x4 Configuration

Figure 4-40: Transmitter Clocking in a Gen1/Gen2/Gen3 PCIe x4 Hard IP and PIPE Configuration

Unlike the Hard IP configuration, the PIPE configuration has the additional flexibility of using the top four transceiver channels in a transceiver bank or spanning the four lanes across two banks.

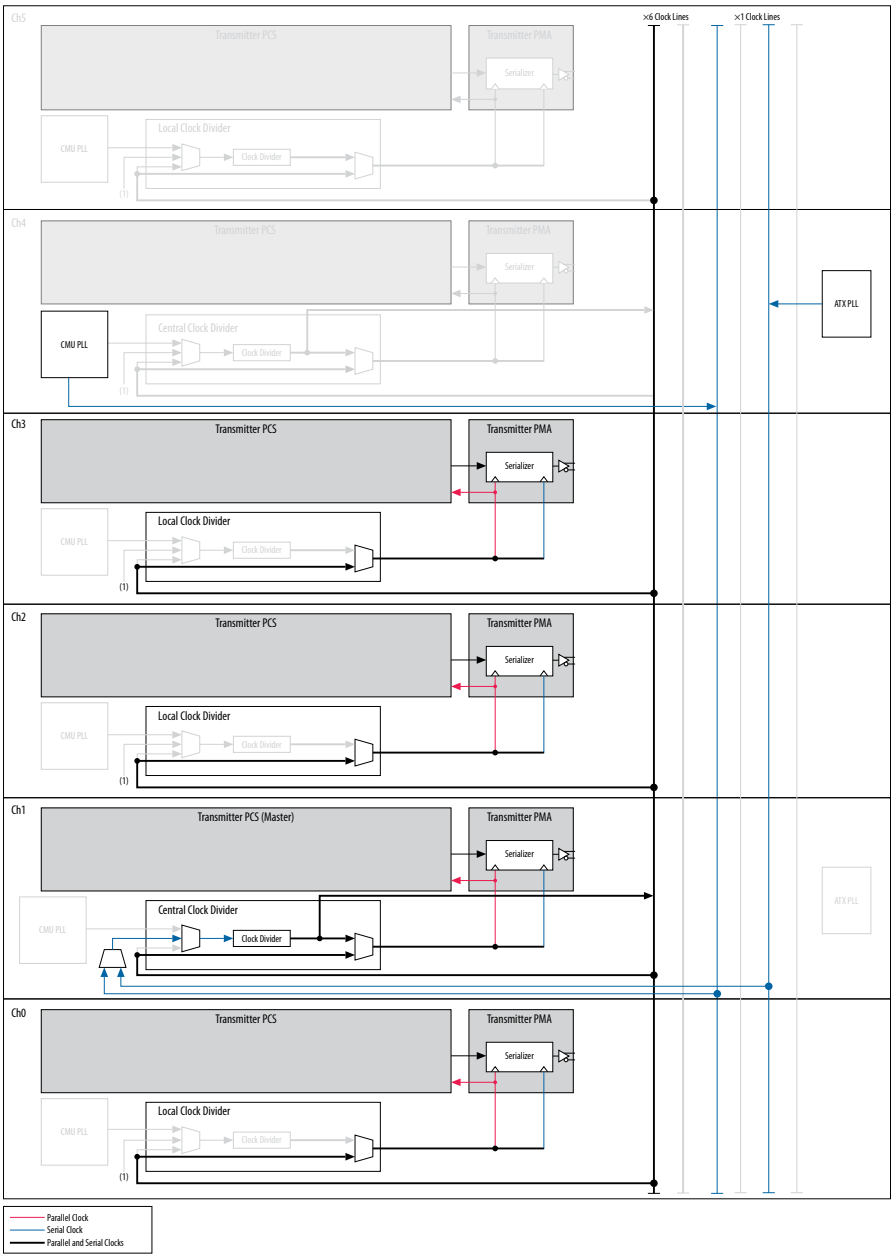
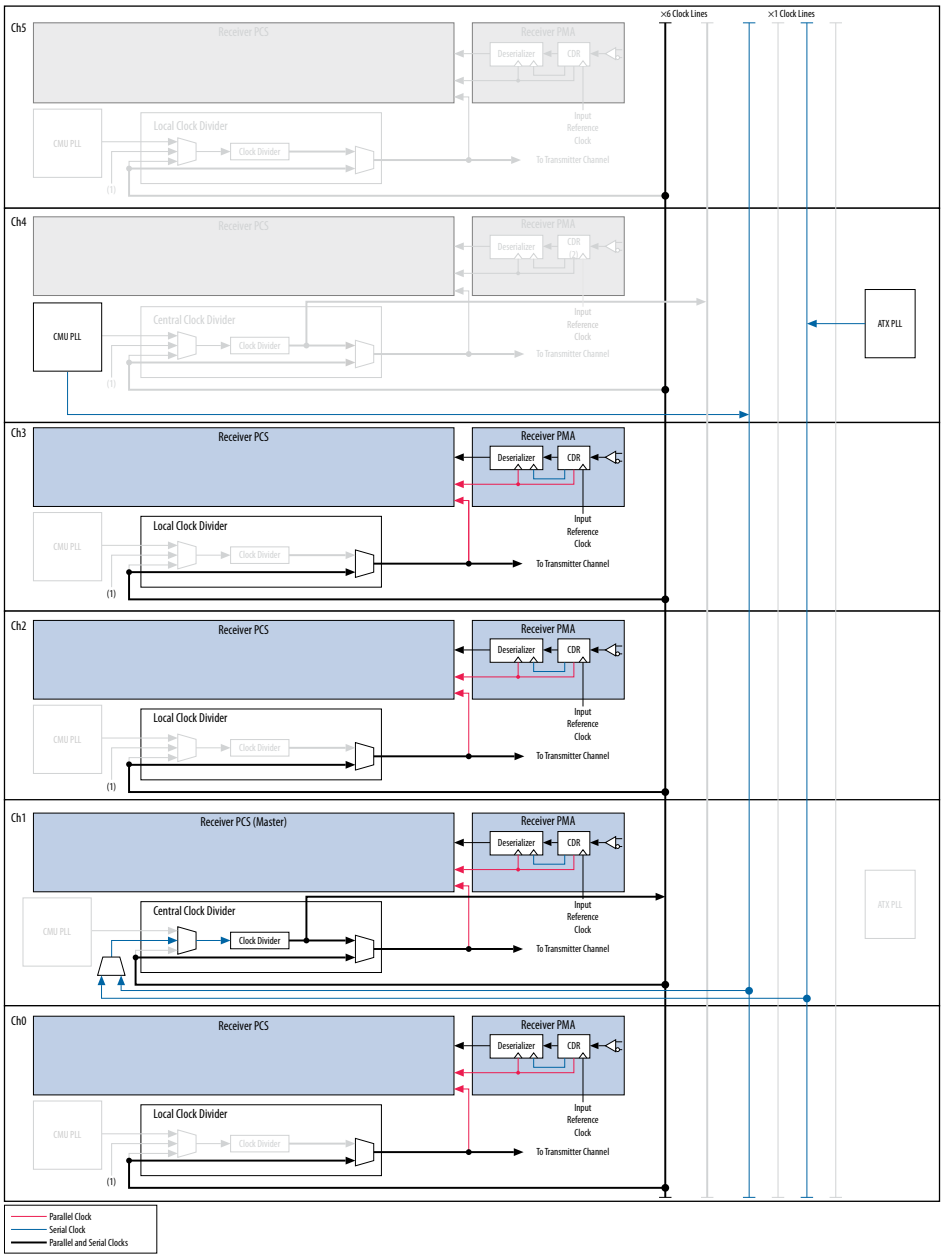


Figure 4-41: Receiver Clocking in a Gen1/Gen2/Gen3 PCIe x4 Hard IP and PIPE Configuration



For **PCIe x4 Gen3** using Hard IP configuration, the CMU PLL (transceiver physical channel 4) and the top ATX PLL of the transceiver bank are configured to generate the high-speed serial clock. A total of five transceiver channels are required to implement PCIe x4 Gen3, including four data channels and one channel for the CMU PLL. The Quartus II software automatically selects channel 1 in the transceiver bank as the master channel. Channel 1 bonds and drives all the transmitter datapath's clocking and the rate matcher side of the FIFO in the receiver datapaths if rate matching is enabled for the four data channels. The local clock divider block in each data channel generates the parallel clock from the high-speed serial clock and distributes both clocks to the PMA and PCS of that data channel.

For **PCIe x4 Gen3** using PIPE configuration, the CMU PLL (transceiver physical channel 1 or 4) and the top or bottom ATX PLL of the transceiver bank are configured to generate the high-speed serial clock. A

total of five transceiver channels are required to implement PCIe x4 Gen3, including four data channels and one channel for the CMU PLL. The Quartus II software automatically selects either channel 1 or 4 in the transceiver bank as the master channel. Channel 1 or 4 bonds and drives all the transmitter datapath's clocking and the rate matcher side of the FIFO in the receiver datapaths if rate matching is enabled for the four data channels. The local clock divider block in each data channel generates the parallel clock from the high-speed serial clock and distributes both clocks to the PMA and PCS of that data channel.

Gen3 x8 Configuration

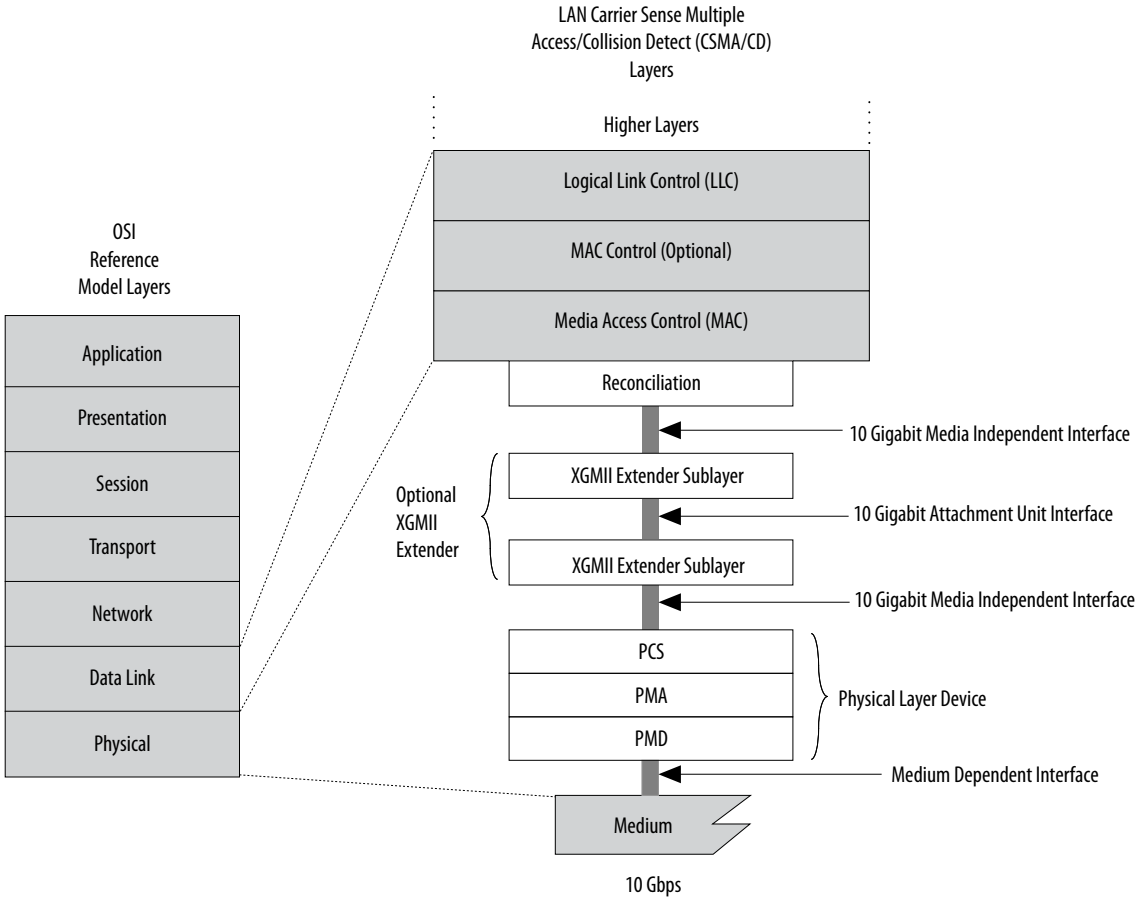
For **PCIe x8 Gen3**, the CMU PLL (transceiver physical channel 4) and the top or bottom ATX PLL of the lower transceiver bank are configured to generate the high-speed serial clock. A total of nine transceiver channels are required to implement PCIe x8 Gen3, including eight data channels and one channel for the CMU PLL. The Quartus II software automatically selects channel 4 in the transceiver bank as the master channel. Channel 4 bonds and drives all the transmitter datapath's clocking and the rate matcher side of the FIFO in the receiver datapaths if rate matching is enabled for the eight data channels. The local clock divider blocks in each data channel generates the parallel clock from this high-speed serial clock and distributes both clocks to the PMA and PCS of that data channel. The master channel in the x8 case is not a data channel.

XAUI

To implement a XAUI link, instantiate the **XAUI PHY IP** core in the IP Catalog, 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 4-42: XAUI and XGMII Layers



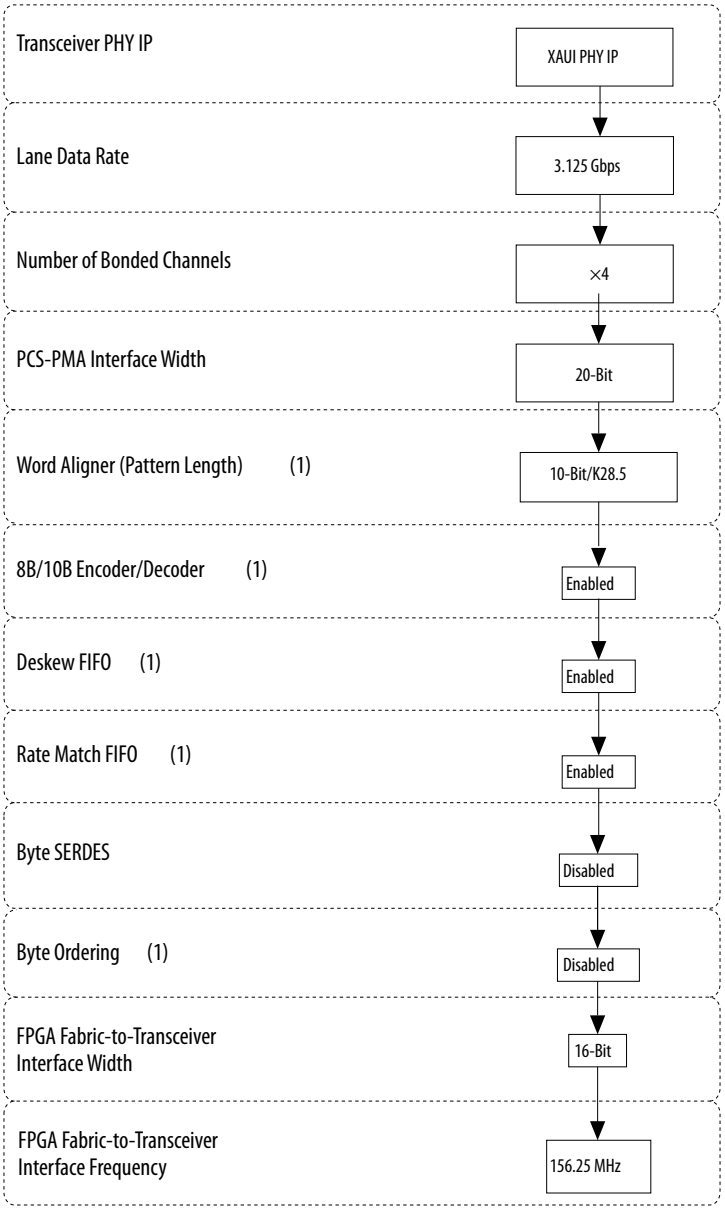
Related Information

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.

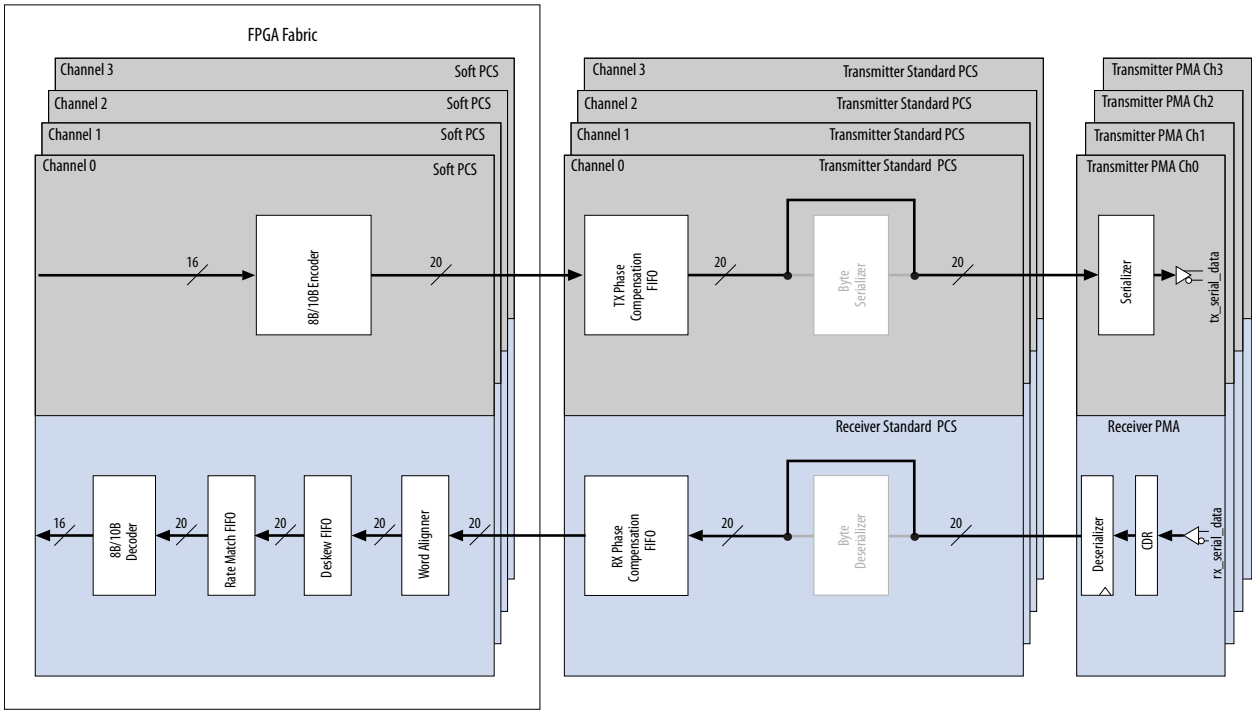
Figure 4-43: XAUI Datapath Configuration



(1) Implemented in soft logic.

Figure 4-44: 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.



Supported Features

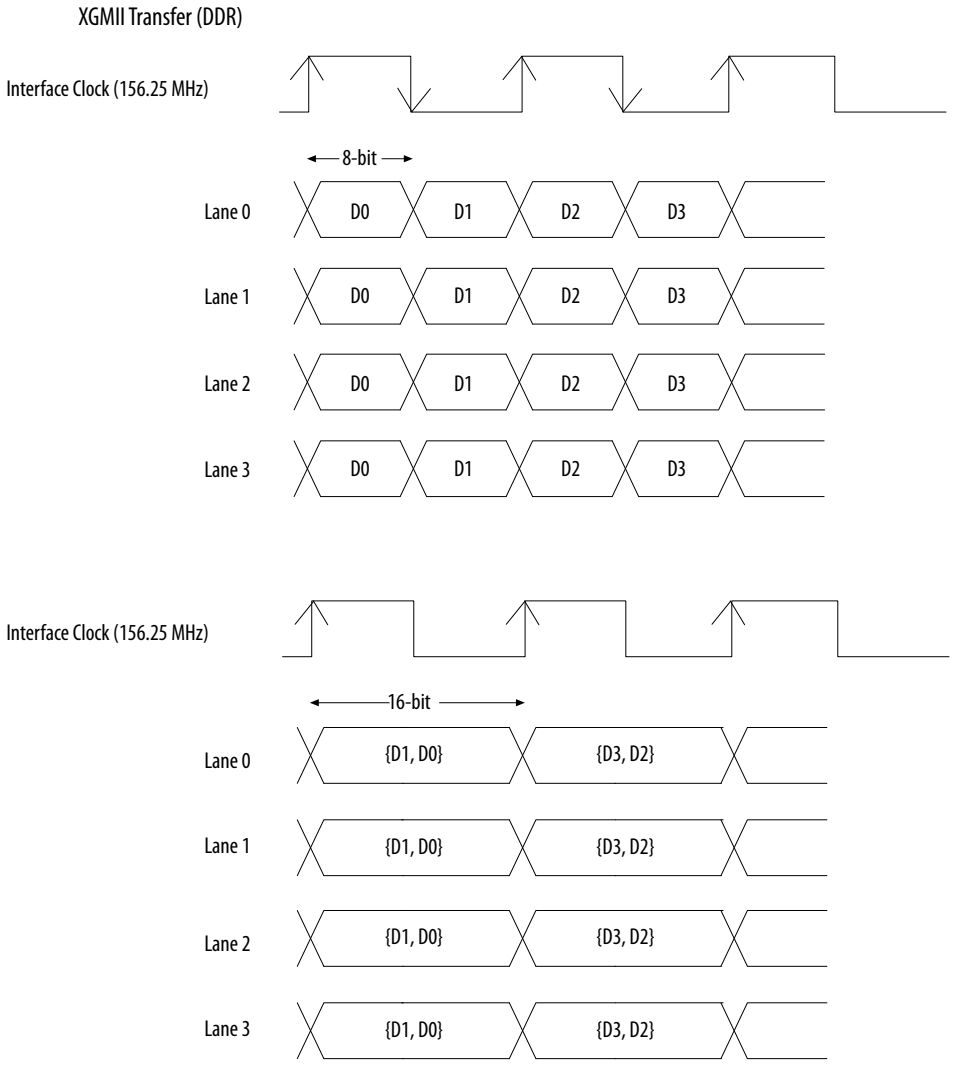
Stratix V transceivers support the following features in a XAUI configuration.

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.

Stratix V transceivers 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 4-45: Implementation of the XGMII Specification in Stratix V Devices



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 Stratix V transceivers implement 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

Transceiver Clocking

Figure 4-46: Transceiver Clocking Diagram 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.

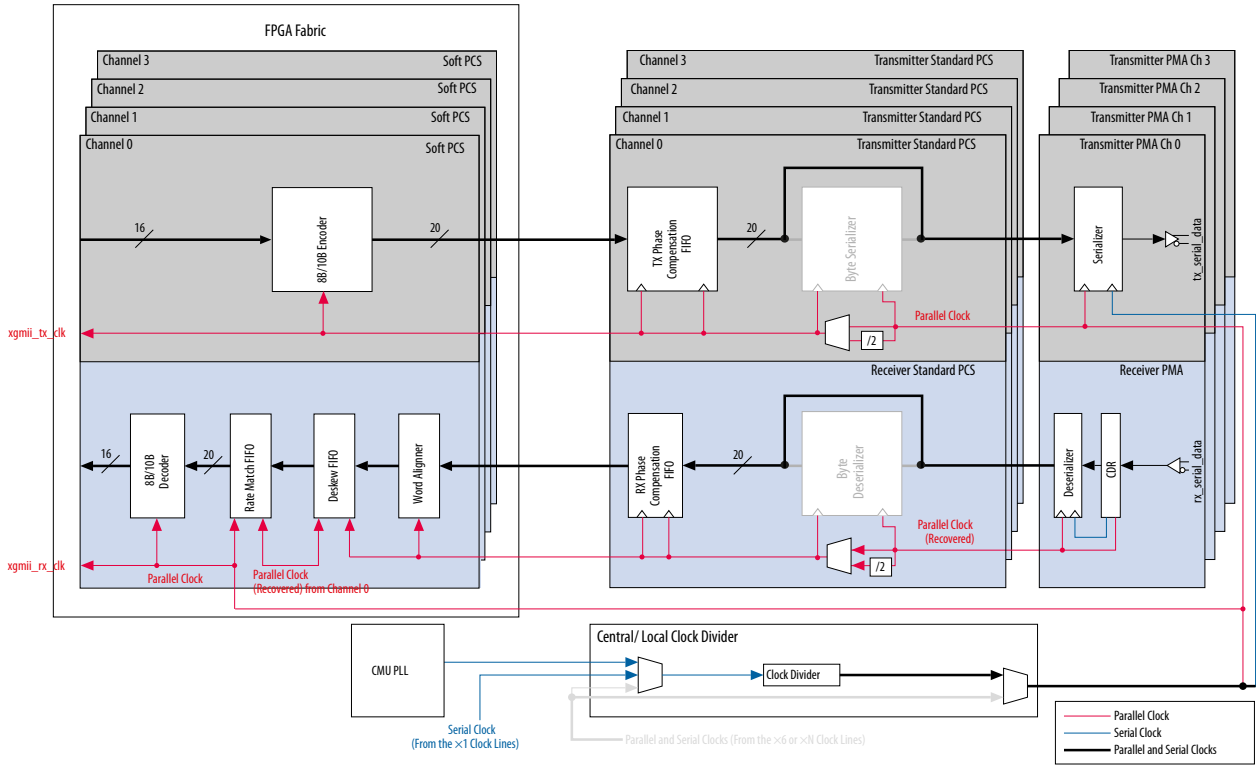


Table 4-9: Input Reference Clock Frequency and Interface Speed Specifications for XAUI Configurations

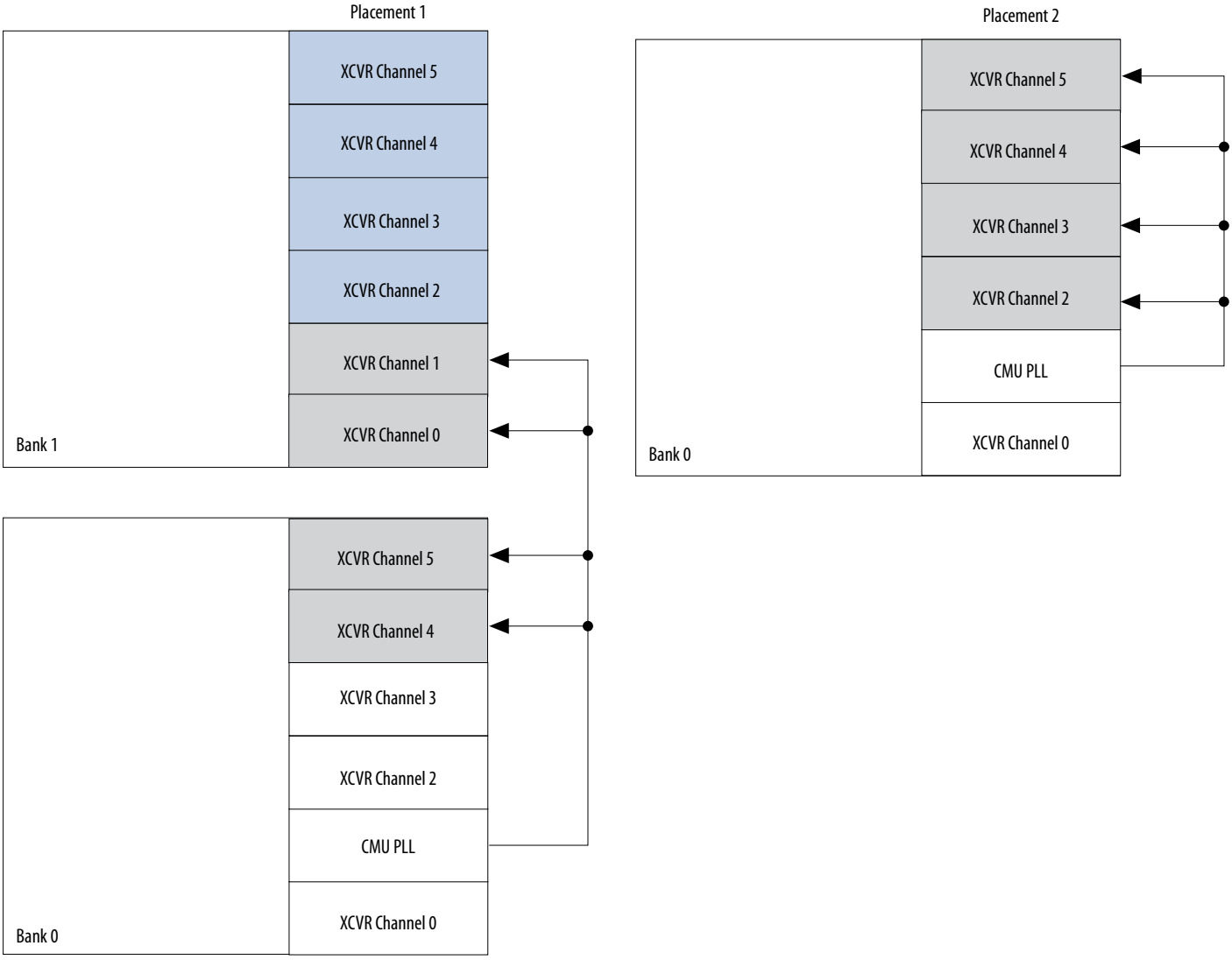
Input Reference Clock Frequency (MHz)	FPGA Fabric-Transceiver Interface Width	FPGA Fabric-Transceiver Interface Frequency (MHz)
156.25	16-bit data, 2-bit control	156.25

Transceiver Channel Placement Guidelines

In the soft PCS implementation of the XAUI configuration, all four channels must be placed continuously. The channels may all be placed in one bank or they may span two banks.

Figure 4-47: Transceiver Channel Placement Guidelines in a XAUI Configuration

Use one of the two allowed channel placements when using either the CMU PLL or the ATX PLL to drive the XAUI link. The Quartus II software implements the XAUI PCS in soft logic.



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.

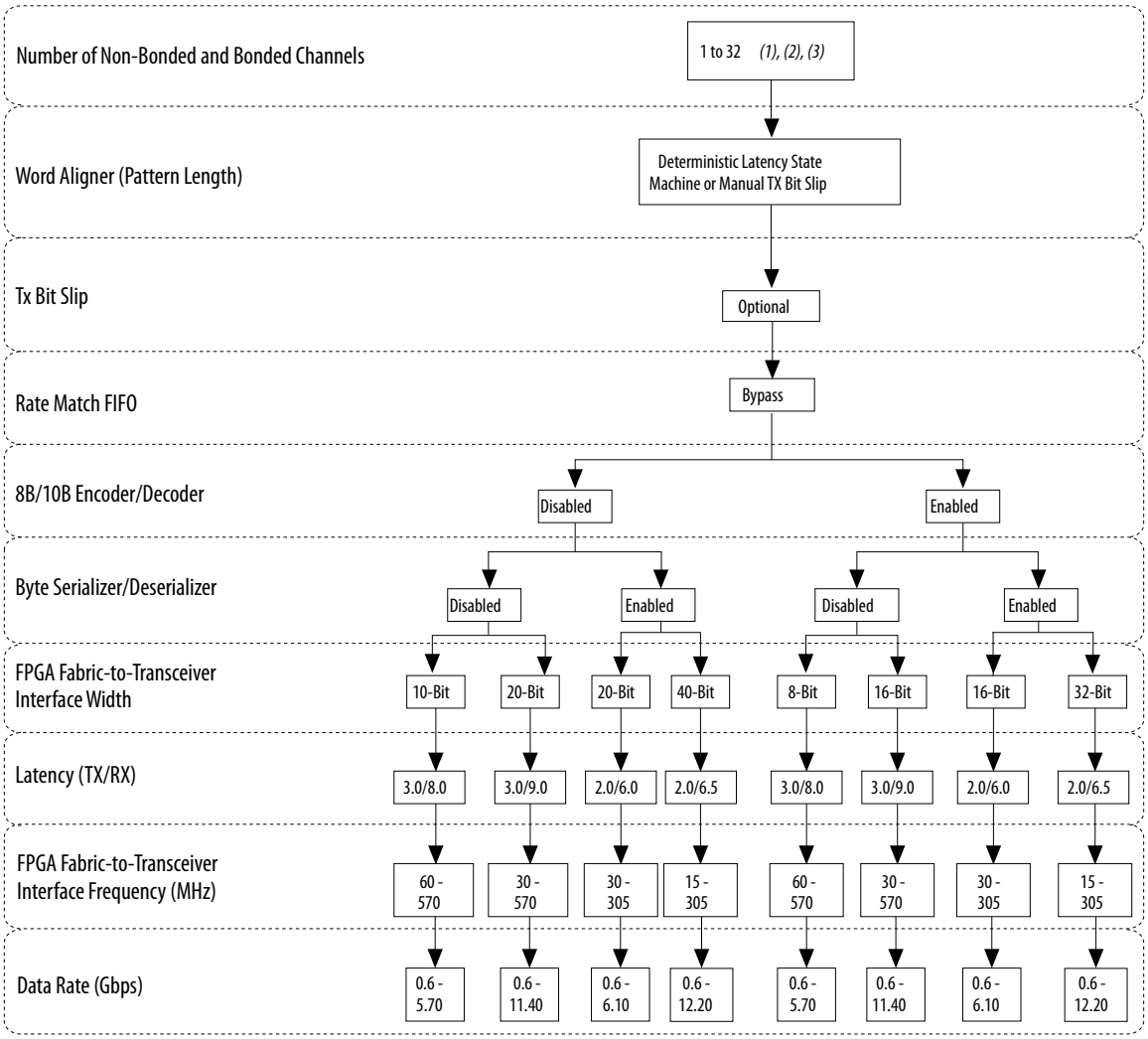
CPRI and OBSAI—Deterministic Latency Protocols

Stratix V devices have a deterministic latency option 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.

Transceiver Datapath Configuration

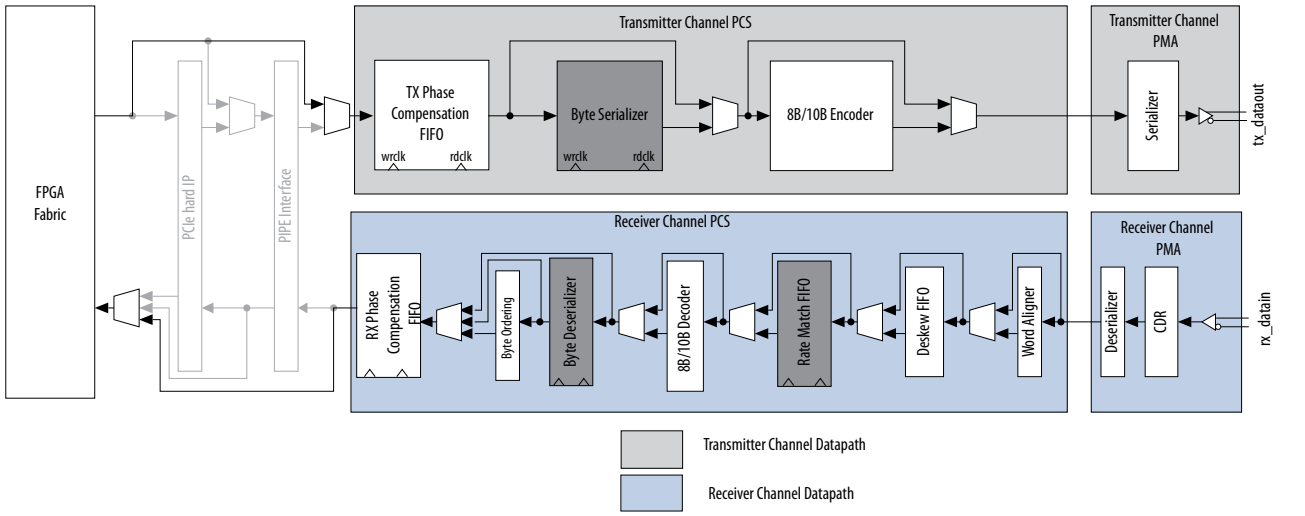
Stratix V devices have a number of options available for the deterministic latency datapath configuration.

Figure 4-48: Deterministic Latency Datapath Configuration



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.
(3) The TX-client feedback path to the transmit PLL is only supported in a non-bonded single lane instance.

Figure 4-49: Transceiver Datapath in Deterministic Latency Mode



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 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

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 is of the same frequency as the low-speed parallel clock.

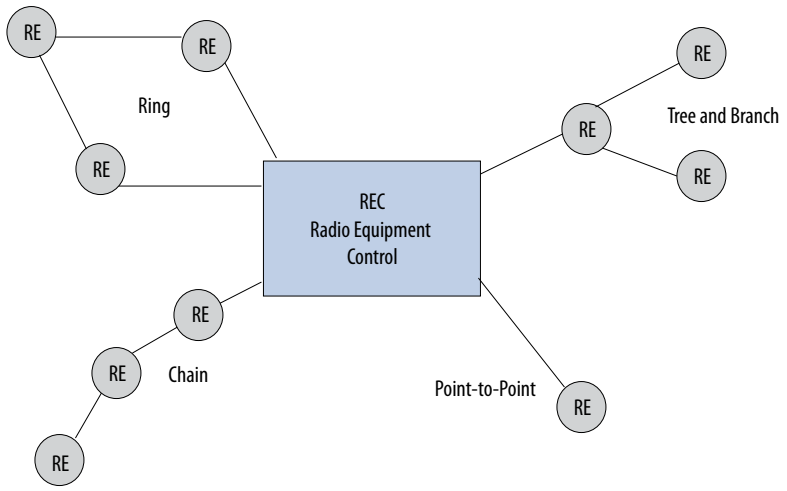
CPRI and OBSAI

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 4-50: 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.

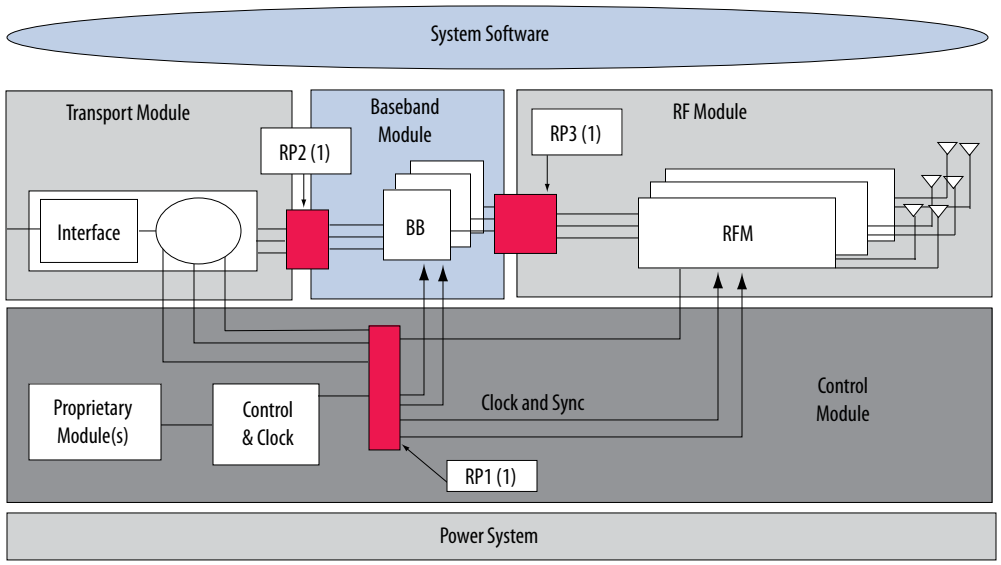
OBSAI was established by several OEMs to develop a set of specifications that can be used for configuring and connecting common modules into base transceiver stations (BTS).

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 4-51: 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	—	—
1228.8	Yes	Yes	Yes	Yes
2457.6	—	Yes	Yes	Yes
3072	—	Yes	Yes	Yes
4915.2	—	—	—	Yes
6144	—	—	—	Yes
9800 (13)	—	—	—	Yes

Related Information

For more information, refer to the Deterministic Latency PHY IP Core chapter in the Altera Transceiver PHY IP Core User Guide.

(13) Applicable to C1, C2, C2L, C3, I2, I2L, and I3L with -1 and -2 transceiver speed grades only.

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 4-52: Deterministic Latency State Machine in the Word Aligner

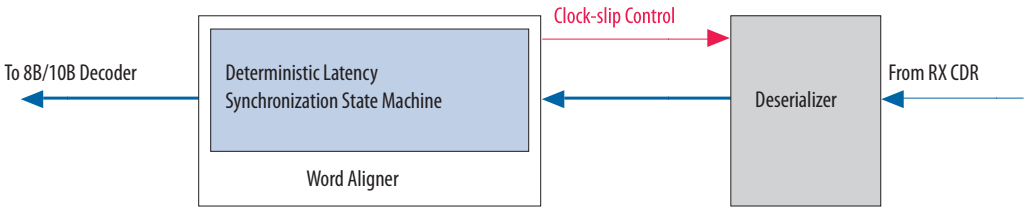


Table 4-11: Methods to Achieve Deterministic Latency Mode in Stratix V Devices

Existing Feature		Enhanced Feature	
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

Transceiver Configurations

Stratix V transceivers offer both standard PCS and 10G PCS configurations. These configurations allow you to modify, enable, or disable blocks based on your protocol requirements. This flexibility allows you to implement various protocols through the Custom, Low Latency, and Native PHY IPs.

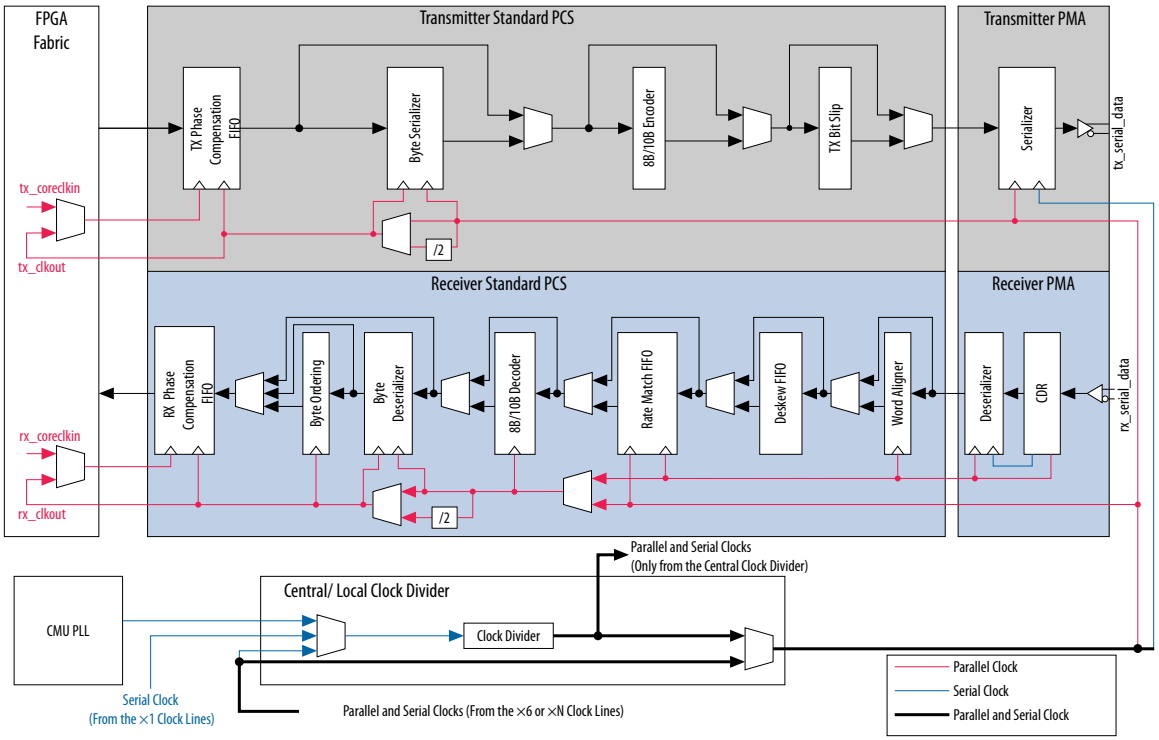
Standard PCS Configurations—Custom Datapath

Use the Custom PHY IP to enable the standard PCS in custom datapath. To implement a Custom PHY link, instantiate the **Custom PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. Define your custom datapath configurations by selecting the blocks to use and the appropriate data width.

The custom datapath consists of the following blocks:

- 8B/10B encoder and decoder
- Word aligner
- Deskew FIFO
- Rate match FIFO (clock rate compensation FIFO)
- Byte ordering block
- Phase compensation FIFO
- Byte serializer and deserializer
- Transmit bit slip

Figure 4-53: Standard PCS Custom Datapath and Clocking



You can divide the custom datapath into two configurations based on the FPGA fabric-transceiver interface width and the PMA-PCS interface width (serialization factor):

- **Custom 8/10-bit-width**—the PCS-PMA interface width is in 8-bit or 10-bit mode for lower data rates.
- **Custom 16/20-bit-width**—the PCS-PMA interface width is in 16-bit or 20-bit mode for higher data rates.

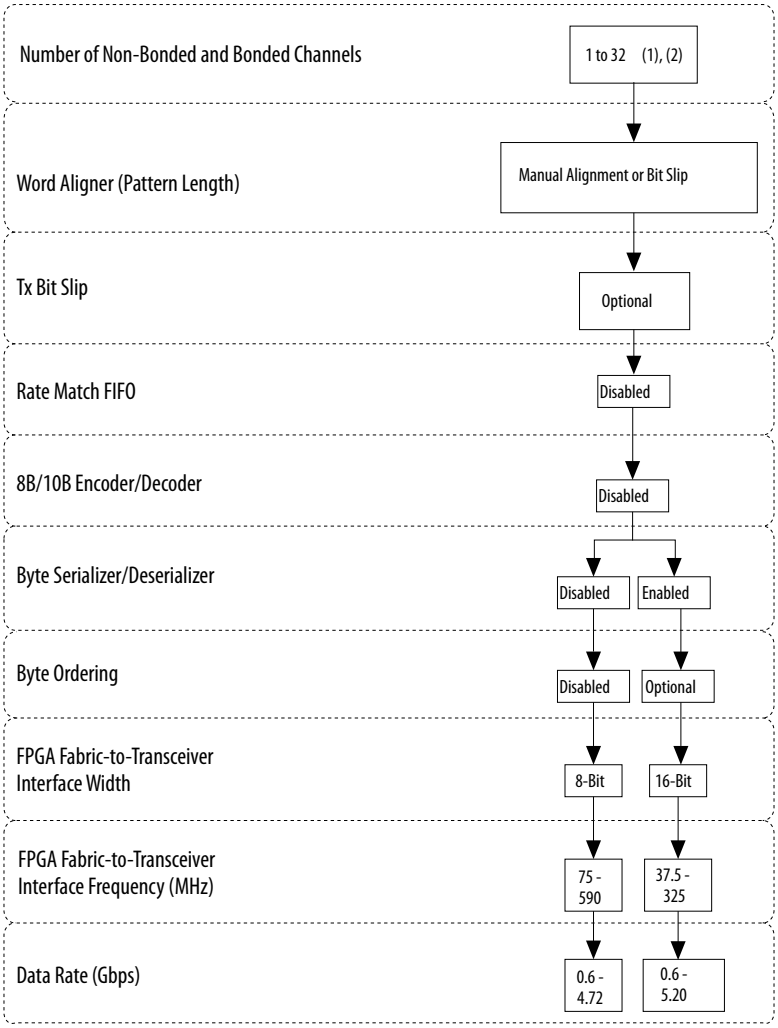
Table 4-12: PCS-PMA Interface Widths and Supported Data Rates

PCS-PMA Interface Width	Supported Data Rate Range PMA
Custom 8-bit width	600 Mbps to 5.20 Gbps
Custom 10-bit width	600 Mbps to 6.50 Gbps
Custom 16-bit width	600 Mbps to 9.76 Gbps

PCS-PMA Interface Width	Supported Data Rate Range PMA
Custom 20-bit width	600 Mbps to 12.20 Gbps

Figure 4-54: Standard PCS Custom 8-Bit PMA-PCS Interface Width

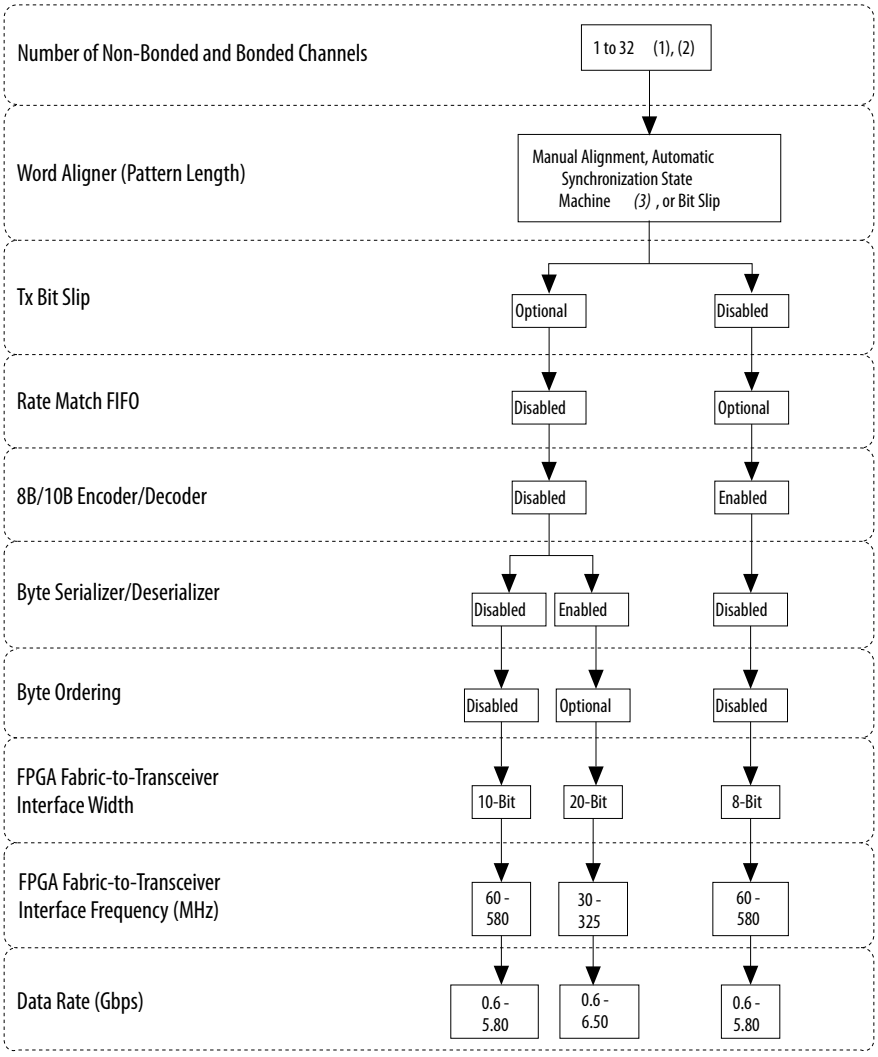
Shows the available options for the standard PCS custom 8-bit PMA-PCS interface width. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.

Figure 4-55: Standard PCS Custom 10-Bit PMA-PCS Interface Width

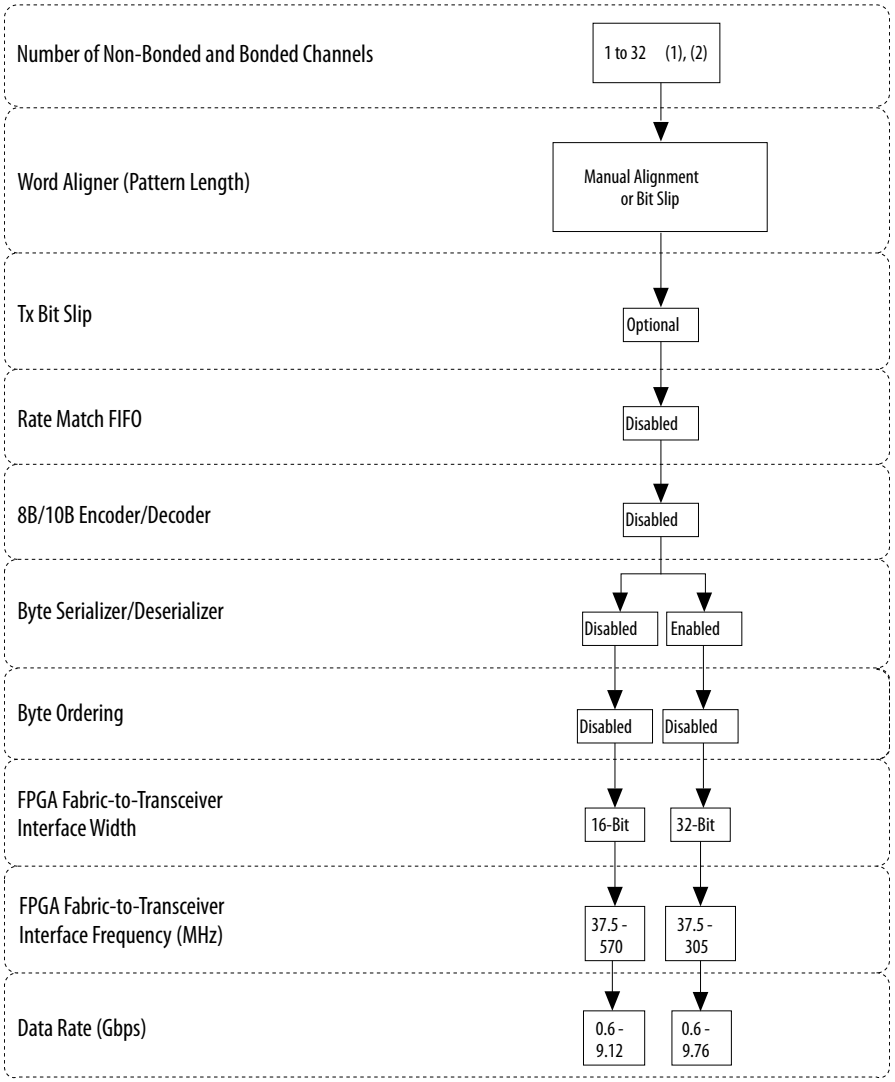
Shows the available options for the standard PCS custom 10-bit PMA-PCS interface width. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.
(3) Automatic Synchronization State Machine requires enabling the 8B/10B Encoder/Decoder.

Figure 4-56: Standard PCS Custom 16-Bit PMA-PCS Interface Width

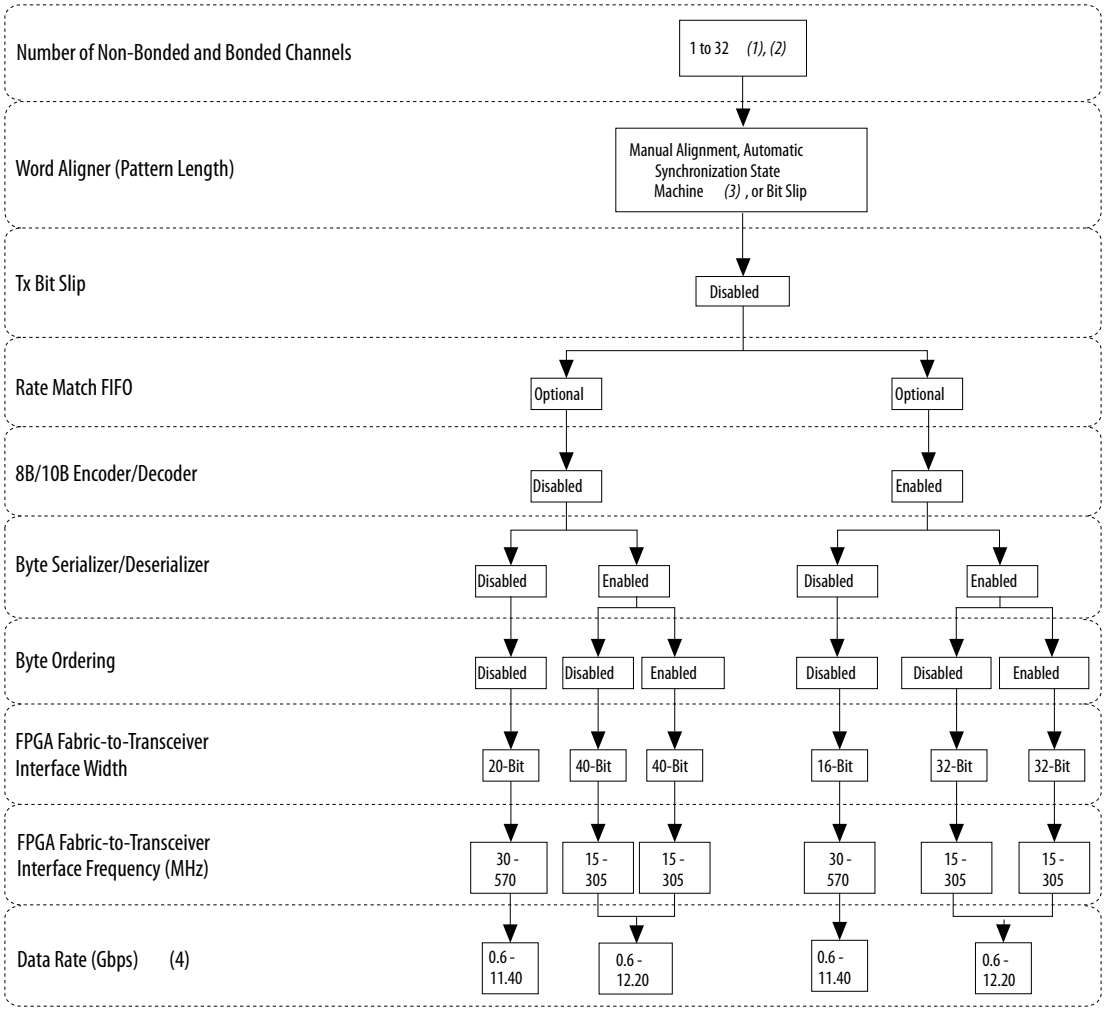
Shows the available options for the standard PCS custom 16-bit PMA-PCS interface width. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.

Figure 4-57: Standard PCS Custom 20-Bit PMA-PCS Interface Width

Shows the available options for the standard PCS custom 20-bit PMA-PCS interface width. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.
(3) Automatic Synchronization State Machine requires enabling the 8B/10B Encoder/Decoder.
(4) The maximum data rate specification is valid only for the -2 (fastest) speed grade devices. For data rate specifications for other speed grades, refer to the device datasheet for that device.

Related Information

- **Refer to the "Standard PCS Architecture" section in the Transceiver Architecture in Stratix V Devices**
- **For information about the maximum data rate for a certain speed grade, refer to the Stratix V Device Datasheet**
- **Refer to the "Custom PHY IP Core" chapter in the Altera Transceiver PHY IP Core User Guide**

Standard PCS Configurations—Low Latency Datapath

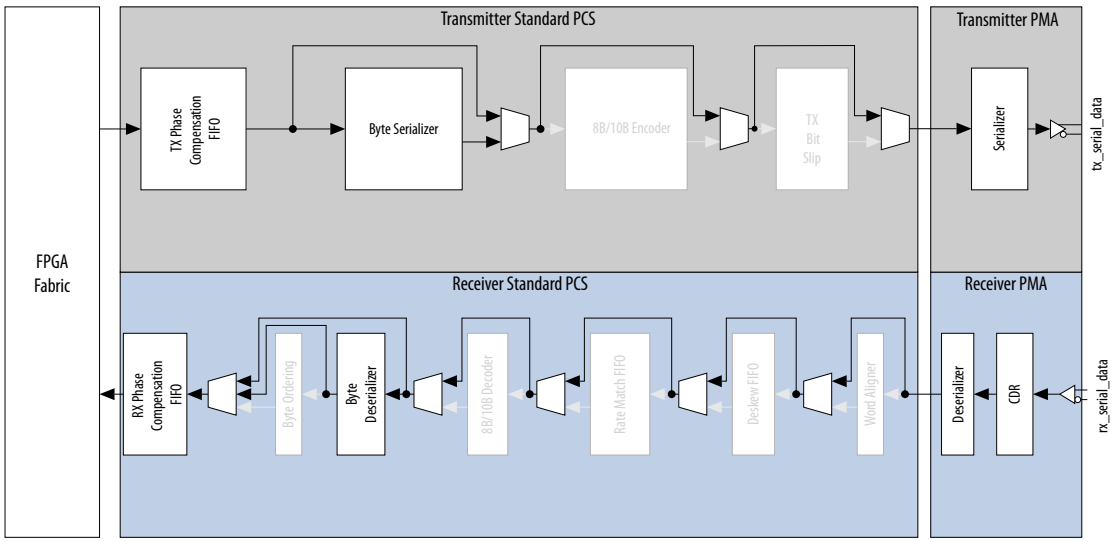
A low latency datapath bypasses much of the standard PCS, allowing more design control in the FPGA fabric. Use the Low Latency PHY IP to enable the standard PCS in a low latency datapath.

To implement a Low Latency PHY link, instantiate the **Low Latency PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. In the Low Latency GUI under the **General** tab, select **Standard** on the **Datapath type** field.

The standard PCS can be used in a low latency datapath that contains only the following blocks:

- Phase compensation FIFO
- Byte serializer and deserializer

Figure 4-58: Standard PCS Low Latency Datapath



You can divide the low latency datapath into two configurations based on the FPGA fabric-transceiver interface width and the PMA-PCS interface width (serialization factor):

- **Low latency 8/10-bit-width**—the PCS-PMA interface width is in 8-bit or 10-bit mode for lower data rates.
- **Low latency 16/20-bit-width**—the PCS-PMA interface width is in 16-bit or 20-bit mode for higher data rates.

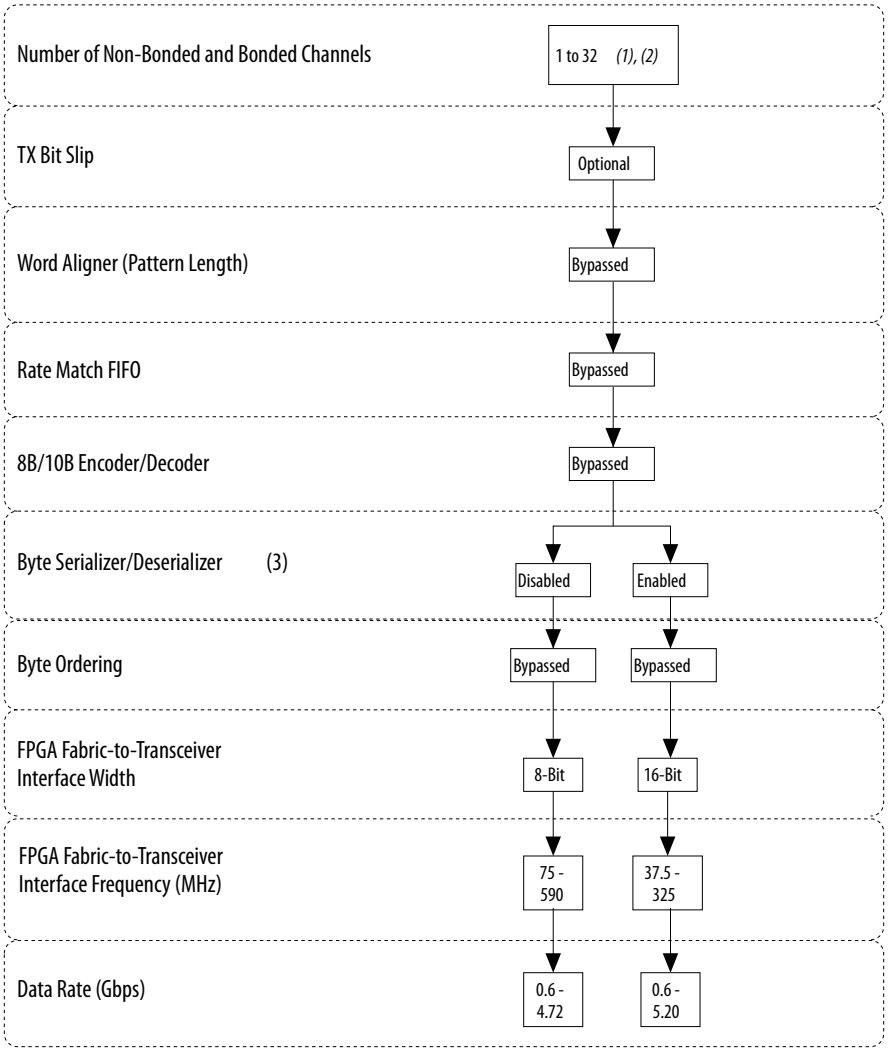
Table 4-13: PCS-PMA Interface Widths and Data Rates

Low Latency PHY IP Core	Supported Data Rate Range PMA
Low Latency 8-bit width	600 Mbps to 5.20 Gbps
Low Latency 10-bit width	600 Mbps to 6.50 Gbps
Low Latency 16-bit width	600 Mbps to 9.76 Gbps
Low Latency 20-bit width	600 Mbps to 12.20 Gbps

In the low latency datapath, the TX and RX phase compensation FIFOs are always enabled. Depending on the targeted data rate, you may bypass the byte serializer and deserializer blocks.

Figure 4-59: Standard PCS Low Latency 8-Bit PMA-PCS Interface Width

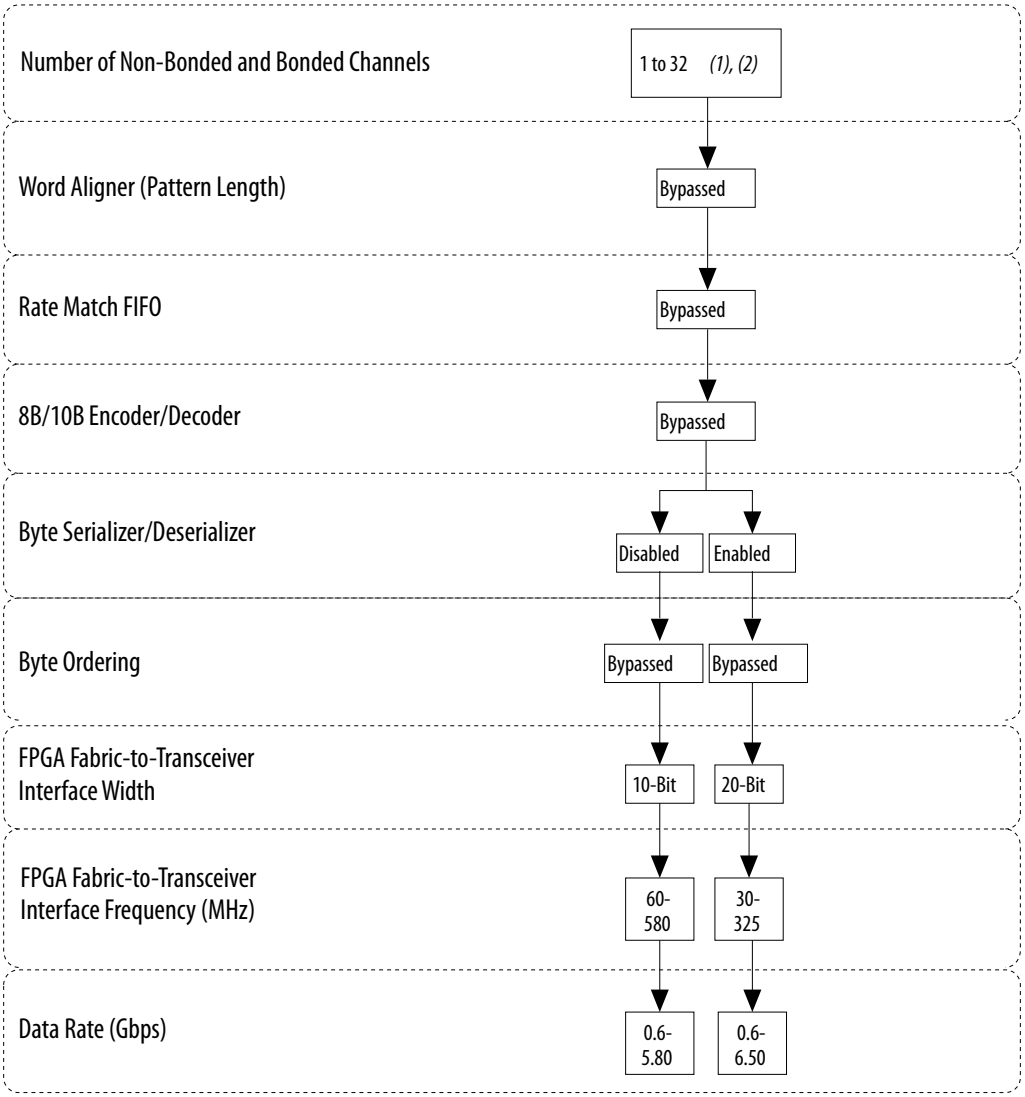
Shows the available options for the standard PCS low latency 8-bit PMA-PCS interface width. The blocks shown as "Disabled" are not used but incur latency. The blocks shown as "Bypassed" are not used and do not incur any latency. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.
(3) The Quartus II software selects whether the byte serializer/deserializer is enabled or disabled based on the datapath width.

Figure 4-60: Standard PCS Low Latency 10-Bit PMA-PCS Interface Width

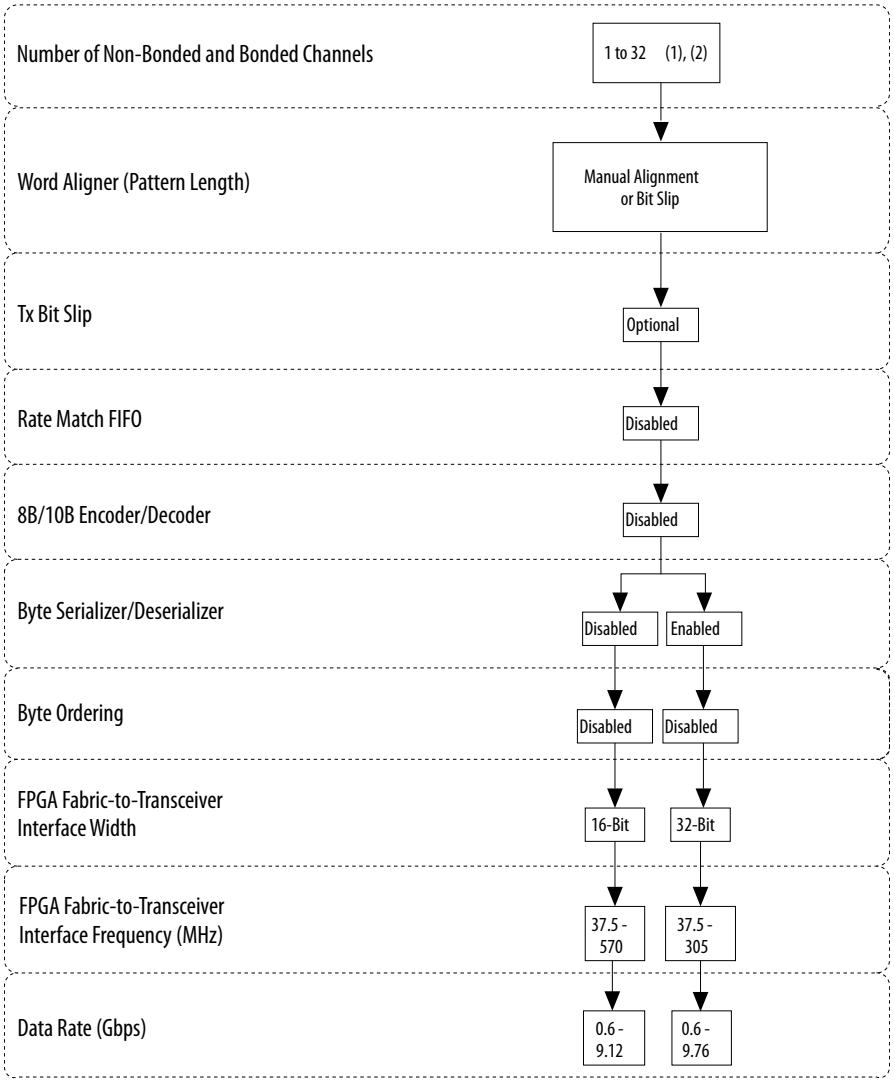
Shows the available options for the standard PCS low latency 10-bit PMA-PCS interface width. The blocks shown as "Disabled" are not used but incur latency. The blocks shown as "Bypassed" are not used and do not incur any latency. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.

Figure 4-61: Standard PCS Low Latency 16-Bit PMA-PCS Interface Width

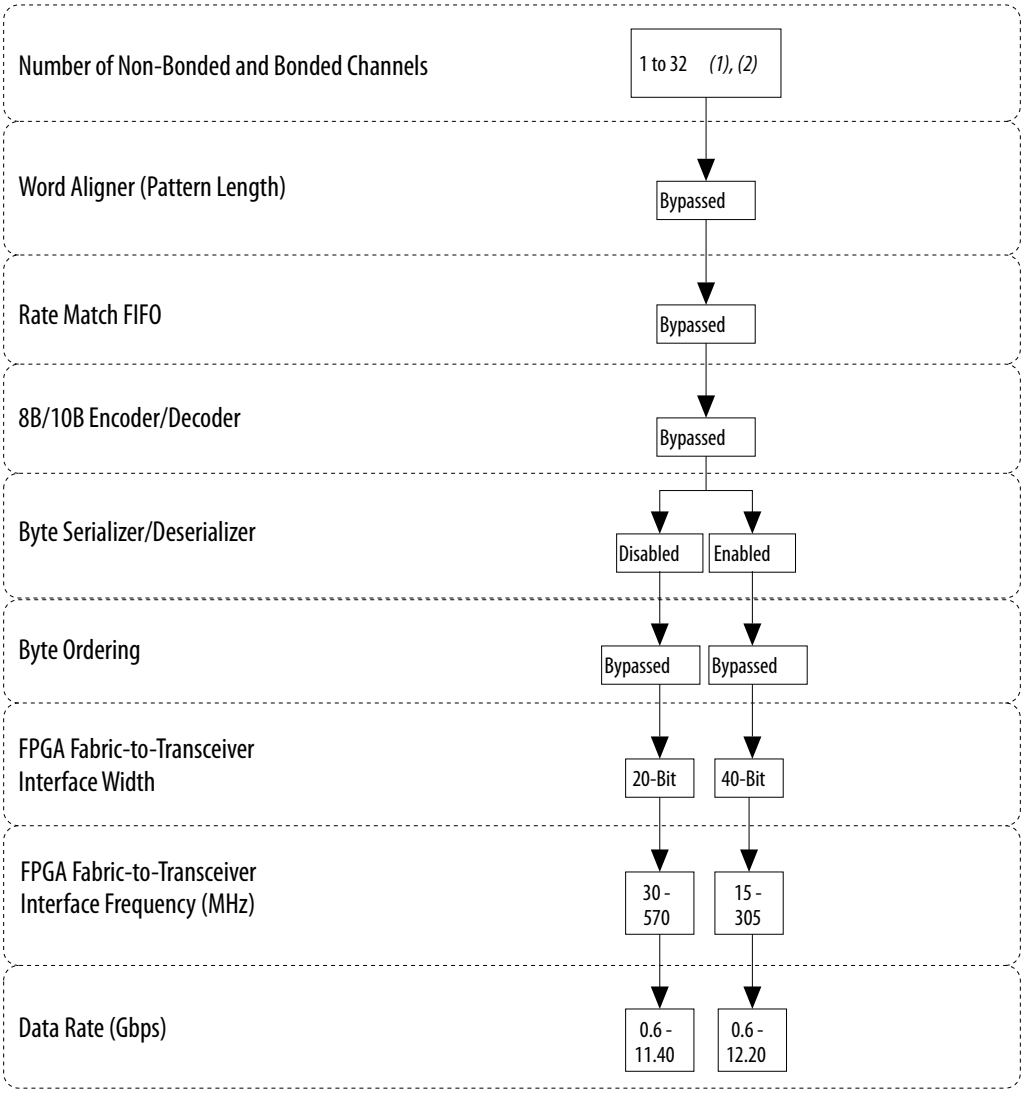
Shows the available options for the standard PCS low latency 16-bit PMA-PCS interface width. The blocks shown as "Disabled" are not used but incur latency. The blocks shown as "Bypassed" are not used and do not incur any latency. The maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.

Figure 4-62: Standard PCS Low Latency 20-Bit PMA-PCS Interface Width

Shows the available options for the standard PCS low latency 20-bit PMA-PCS interface width. 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 maximum frequencies are for the fastest devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.

Related Information

- **Refer to the "Standard PCS Architecture" section in the Transceiver Architecture in Stratix V Devices**
- **For information about the maximum data rate for a certain speed grade, refer to the Stratix V Device Datasheet**

- **Refer to the "Low Latency PHY IP Core" chapter in the Altera Transceiver PHY IP Core User Guide**

Transceiver Channel Placement Guidelines

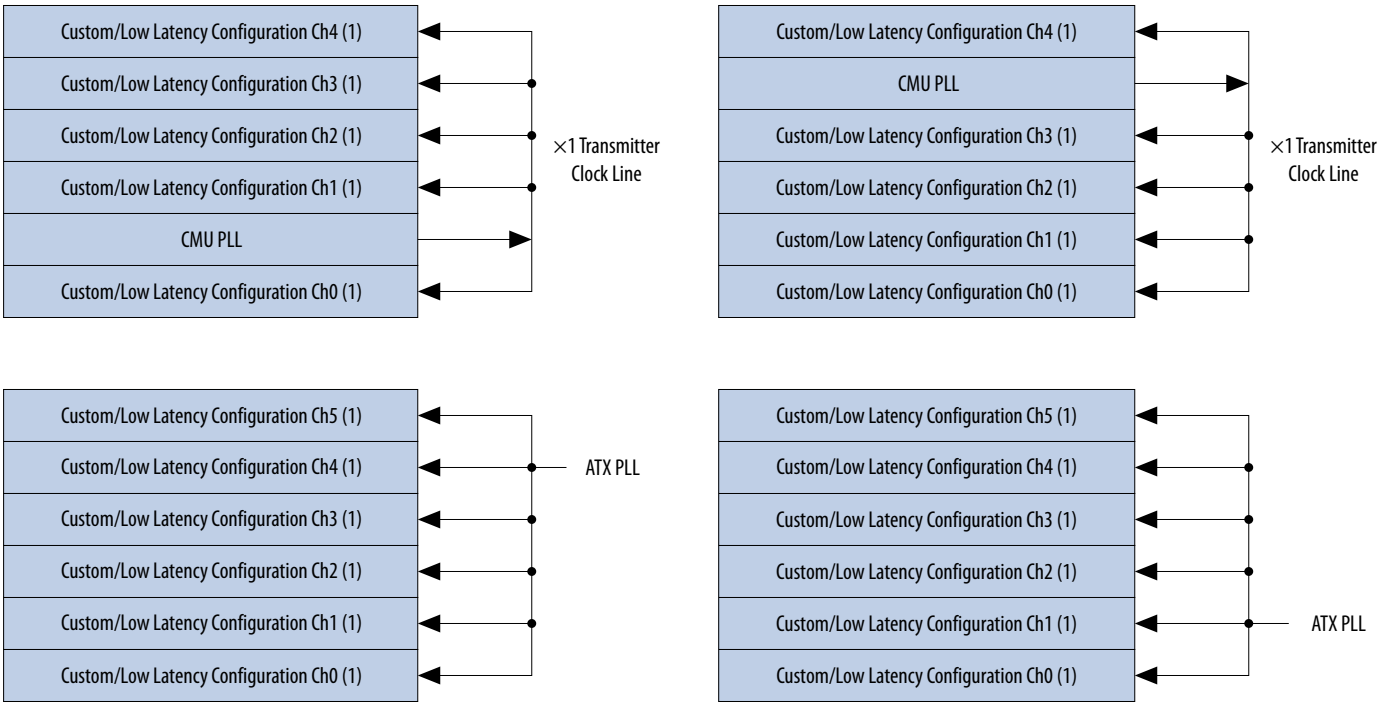
You can use CMU PLLs or ATX PLLs in non-bonded and bonded configurations.

Stratix V devices allow the placement of up to five channels when a CMU PLL is used or up to six channels when an ATX PLL is used in a non-bonded configuration within the same transceiver bank:

- Custom PHY IP with standard PCS datapath configuration
- Low Latency PHY IP with Standard PCS or 10G PCS (same data rate) in low latency datapath configuration

Figure 4-63: Non-Bonded Channel Placement Guidelines with Standard and 10G PCS in Custom and Low Latency Datapath Configurations

All channels are assumed to contain a transmitter and receiver.

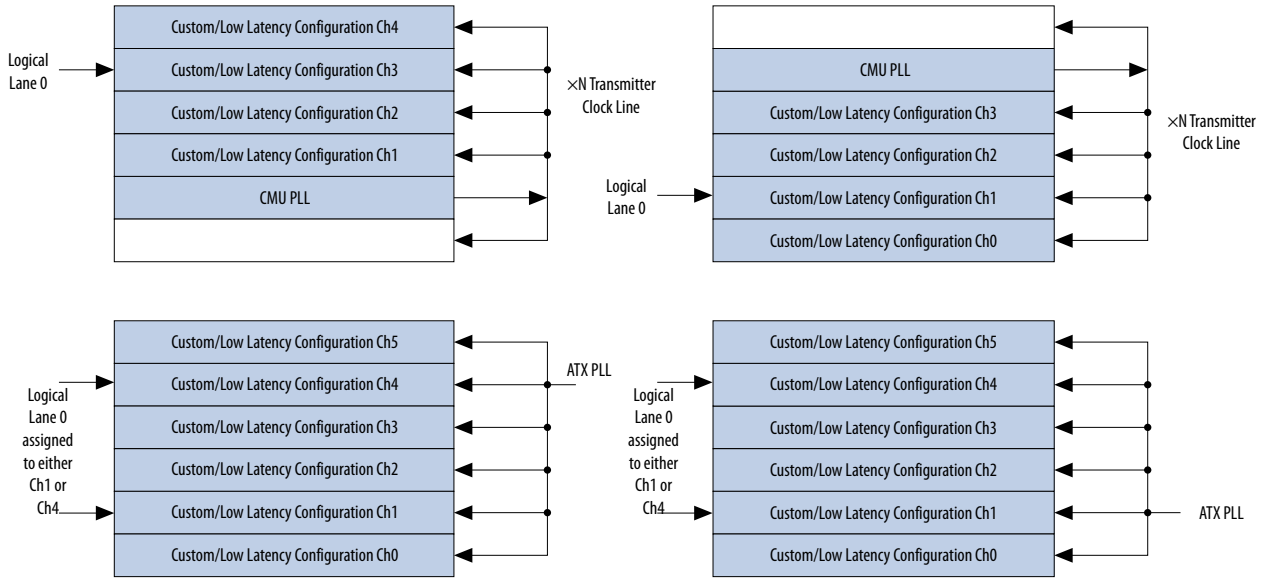


Stratix V devices allow the placement of up to four channels when a CMU PLL is used or up to six channels when an ATX PLL is used in a bonded configuration within the same transceiver bank:

- Custom PHY IP with standard PCS datapath configuration
- Low Latency PHY IP with Standard PCS or 10G PCS (same data rate) in low latency datapath configuration

The xN bonding method requires Logical Lane 0 be placed at either transceiver physical channel 1 or 4 within a transceiver bank. The PLL feedback compensation bonding method does not have a Logical Lane 0 assignment requirement and must be used when more than one transceiver bank is needed. However, PLL feedback compensation bonding requires the use of one PLL per transceiver bank.

Figure 4-64: Bonded Channel Placement Guidelines with Standard and 10G PCS in Custom and Low Latency Datapath Configurations



10G PCS Configurations

The Low Latency PHY IP can also configure 10G PCS in the low latency datapath.

To implement a Low Latency PHY link with the 10G PCS, instantiate the **Low Latency PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. In the Low Latency GUI under the **General** tab, select **10G** on the **Datapath type** field.

A Low Latency PHY IP core with the 10G PCS is available for 32-bit, 40-bit, 50-bit, 64-bit, or 66-bit PCS data width configurations.

Figure 4-65: 10G PCS Low Latency Configuration Datapath

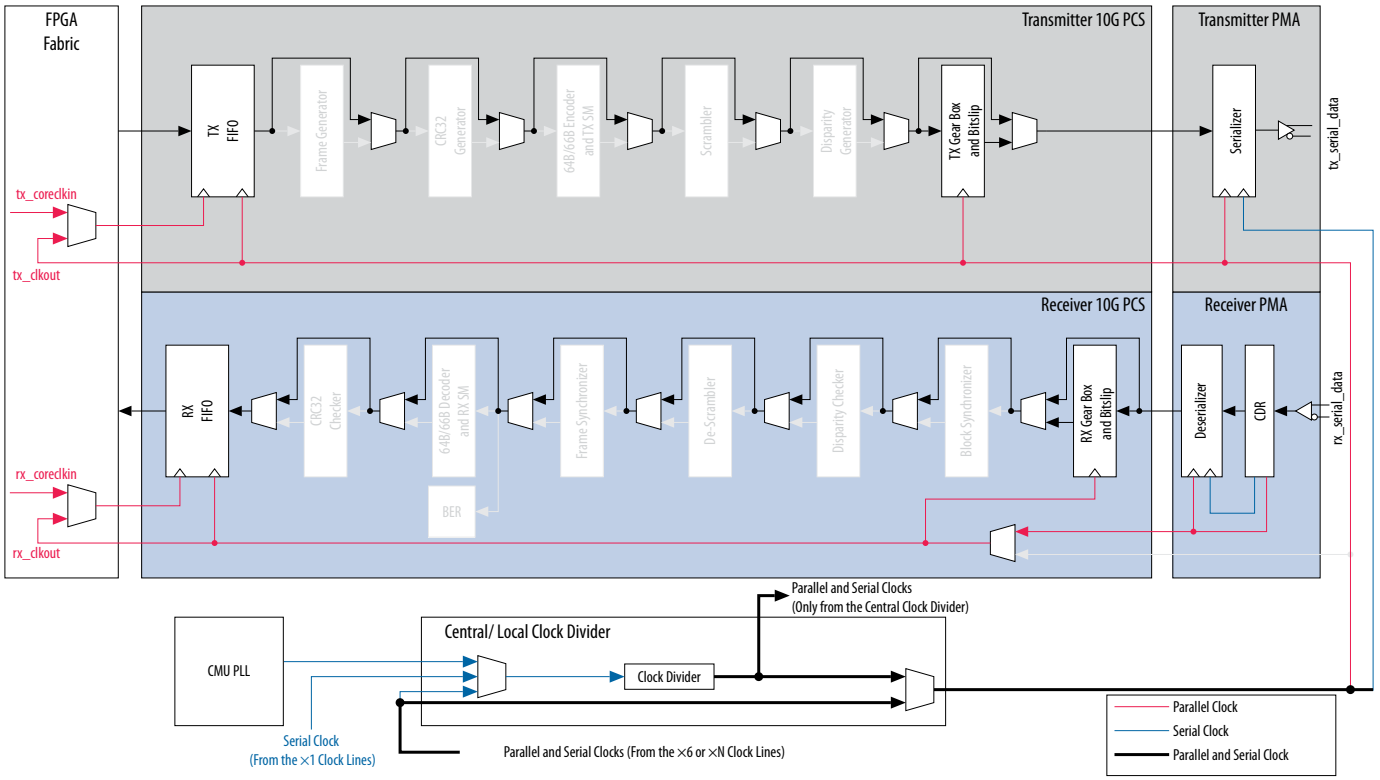
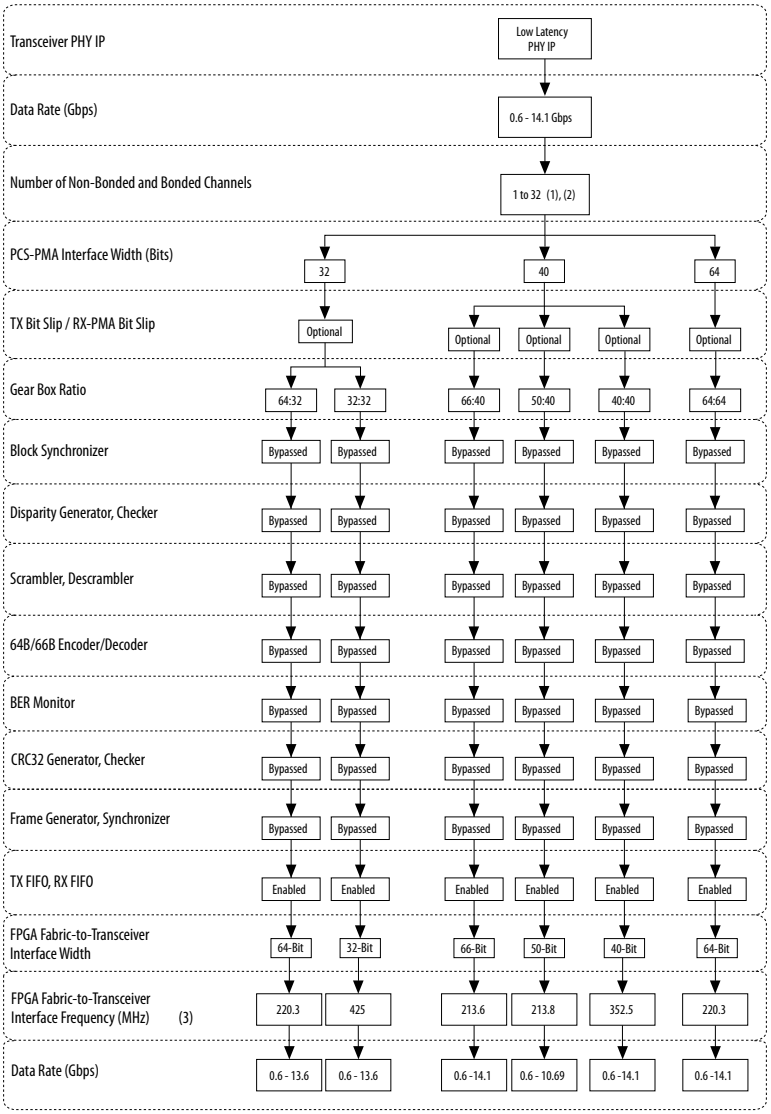


Figure 4-66: Options for 10G PCS Low Latency Configuration

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 FPGA fabric-to-transceiver interface maximum frequency is for the fastest speed grade devices.



Notes:
(1) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(2) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.
(3) You must generate an rx_coreclkin with the specified frequency whenever the gear box is enabled.

The Quartus II software supports both non-bonded configuration and bonded configurations up to 32 lanes in the link when the 10G PCS in low latency datapath configuration is enabled. If you create multiple non-bonded channels with the 10G PCS in low latency mode, a common parallel clock (used in the bonded lane or channel configuration) is not generated by the central clock divider block. Each transmitter channel takes the high-speed clock, generated by the channel PLL, and locally divides it to generate the parallel clock.

Related Information

- **For the limits of all speed grades, refer to the "Transceiver Performance Specifications" section in the Stratix V Device Datasheet**
- **Transceiver Clocking in Stratix V Devices**
- **Refer to the Low Latency PHY IP Core chapter in the Altera Transceiver PHY IP Core User Guide**

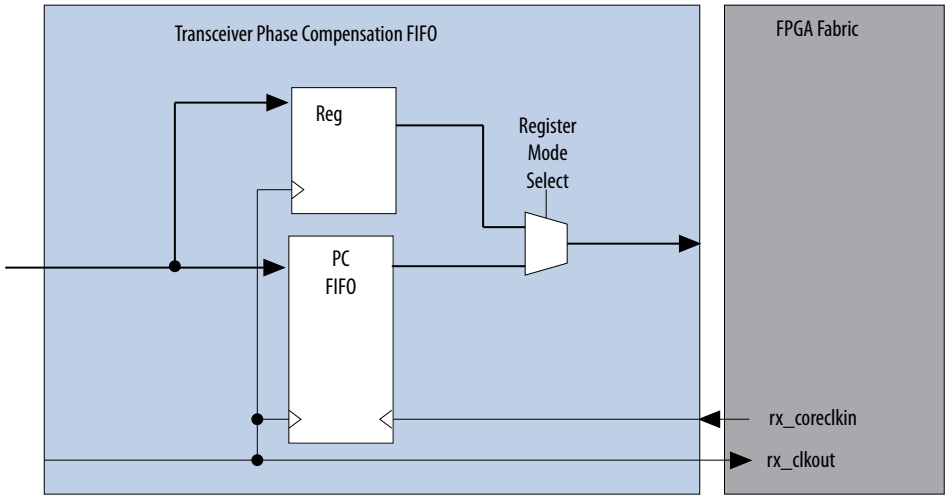
10G PCS Datapath Functionality

Various 10G PCS blocks are available when you implement the 10G PCS in low latency mode.

Transmitter and Receiver FIFO

The FIFOs can be configured in phase compensation or registered mode for the RX path. In phase compensation mode, the FIFO compensates the phase differences in the clock between the read and write side of the FIFO. The clocking scheme for the write side of the transmitter (TX) and receiver (RX) FIFOs depends on whether the gear box is enabled and on its ratio (40:66, 40:50, or 32:64). The clocking scheme is described in **Clocking** on page 1-85.

Figure 4-67: Phase Compensation FIFO in RX Path



Gear Box

The gear box translates the datapath width differences between the PCS and the physical medium attachment (PMA) interfaces. The gear box contains handshake control logic and FIFOs to implement the data-width translation. For the supported gear box ratio, refer to figure "Options for 10G PCS Low Latency Configuration".

TX Bit Slip Feature

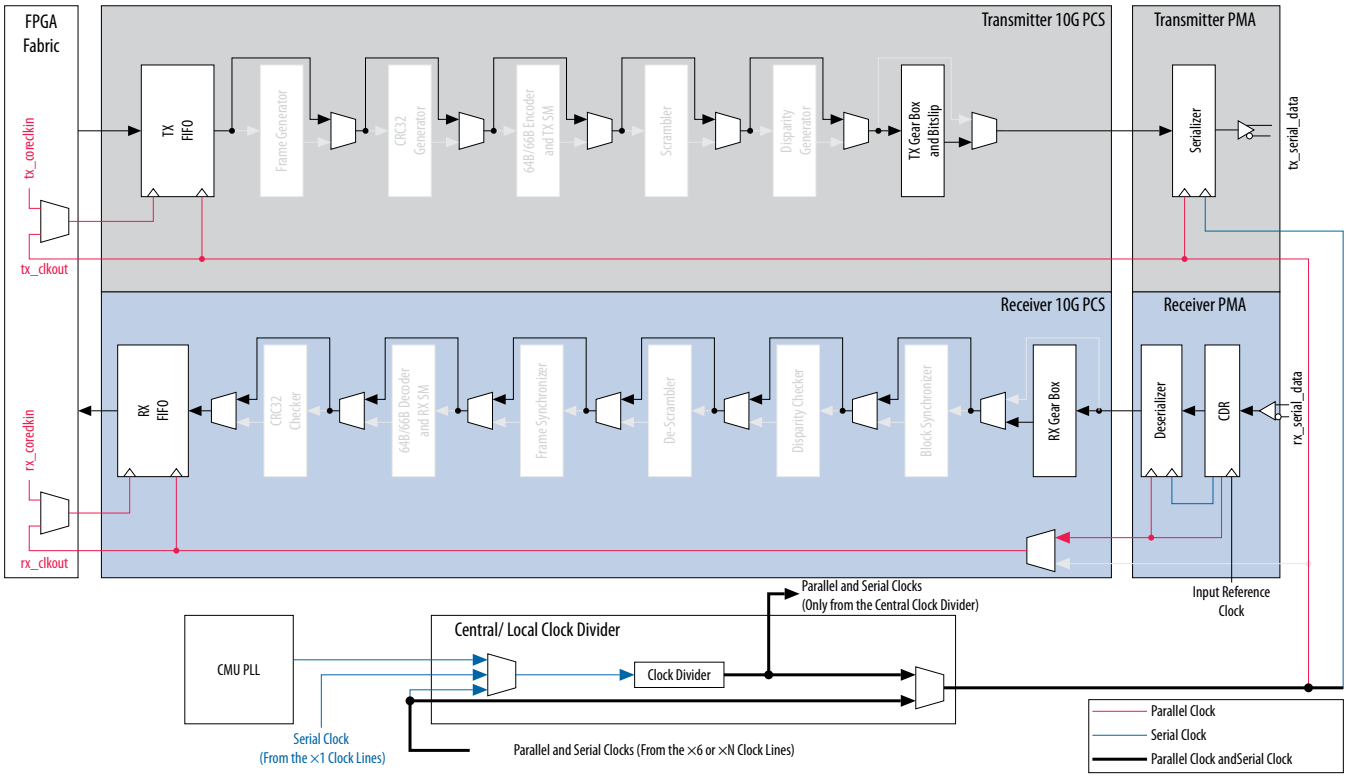
The bit slip feature allows you to slip the transmitter side bits before they are sent to the gear box. The number of bits slipped is equal to the FPGA fabric-to-transceiver interface width minus 1. For example, if the FPGA fabric-to-transceiver interface width is 64 bits, a maximum of 63 bits can be slipped. That is, `bit[63]` from the first word and `bit[62:0]` are concatenated to form a 64 bit word (`bit[62:0]` from the second word, `bit[63]` from the first word LSB). The 7-bit input control signal is available to the FPGA fabric. For a 63-bit shift mentioned above, set the value of the input control to 7'b0011111.

Clocking

The transceiver datapath clocking scheme depends on the gear box ratio.

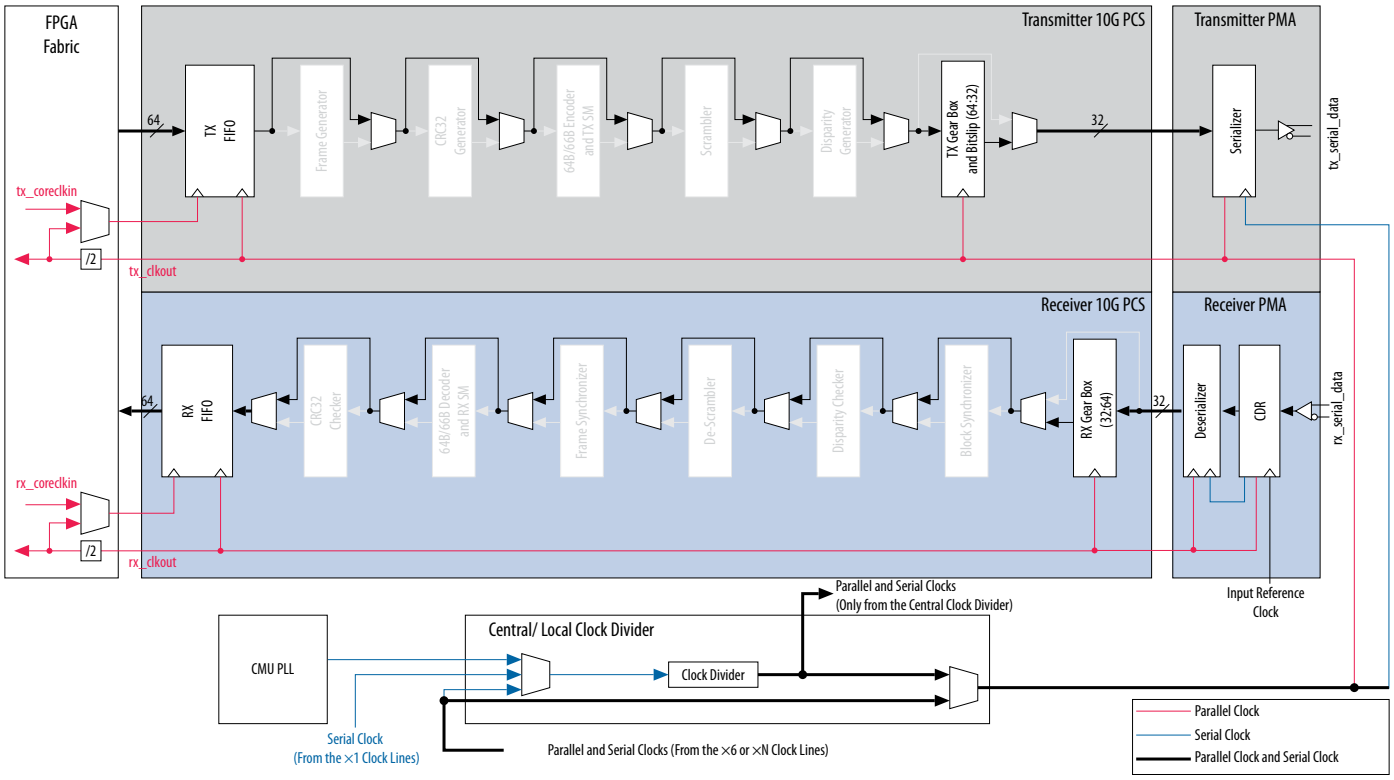
When the gear box ratio is 64:64, 40:40, or 32:32, there is no frequency difference between the read and write side of the TX and RX FIFO clocks because the gear box is the same ratio. The Quartus II software automatically connects the clocks to the read and write side of the TX FIFO and RX FIFO. In this configuration, the data from the TX FIFO is still fed to the gear box before being sent to the serializer. The gear box cannot be bypassed or disabled.

Figure 4-68: 10G PCS Low Latency Datapath with Gear Box in 64:64, 40:40, and 32:32 Ratio



When the gear box ratio is 64:32. The FPGA fabric interface width (64 bits) is exactly twice the internal transceiver datapath width. You can divide the `tx_clkout` and `rx_clkout` in the FPGA fabric by two, and use them to clock the write side of TX FIFO and the read side of RX FIFO, respectively. Select the `tx_coreclkin` and the `rx_coreclkin` ports in the Low Latency PHY IP core and connect the divided clock to these ports.

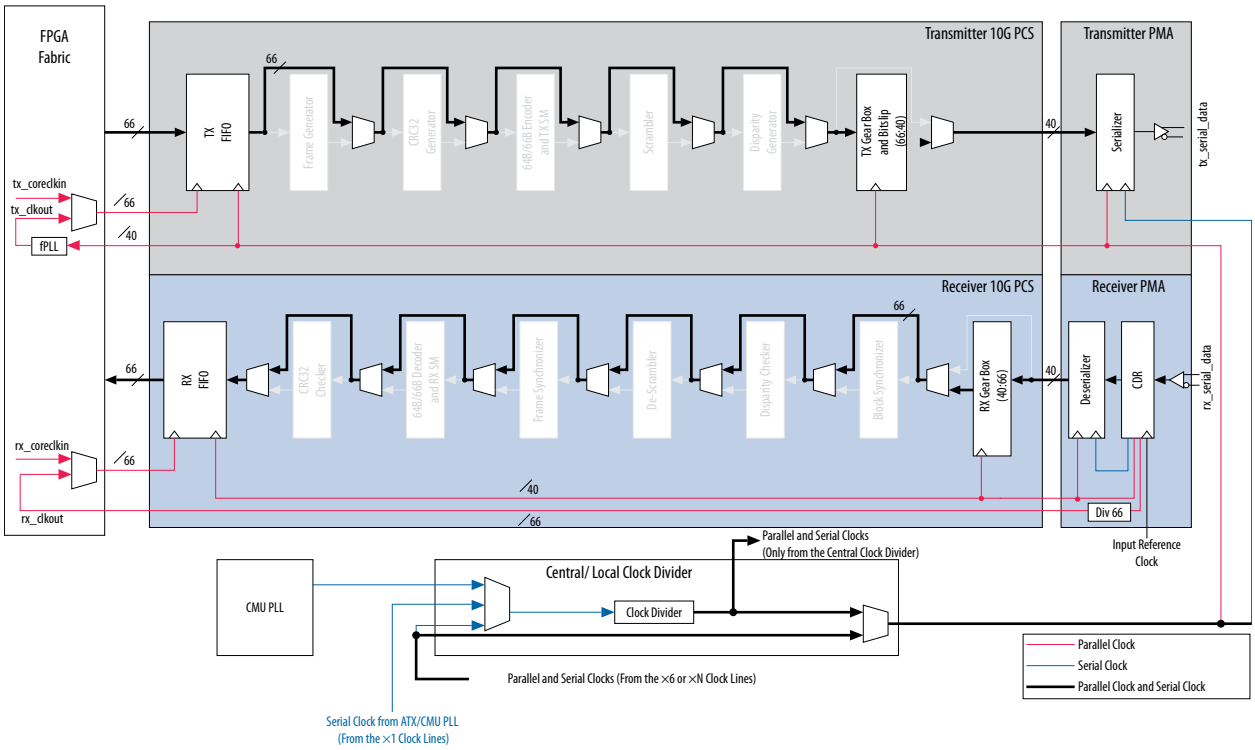
Figure 4-69: 10G PCS Low Latency Datapath with the Gear Box Ratio of 64:32



When the gear box ratio is 66:40, the `rx_clkout` parallel clock provided is a recovered clock coming from the CDR with a divided-by-66 output frequency.

The `tx_clkout` parallel clock is generated from the transmit PLL feeding a fractional PLL that is automatically instantiated from the FPGA core with a divided-by-66 output frequency.

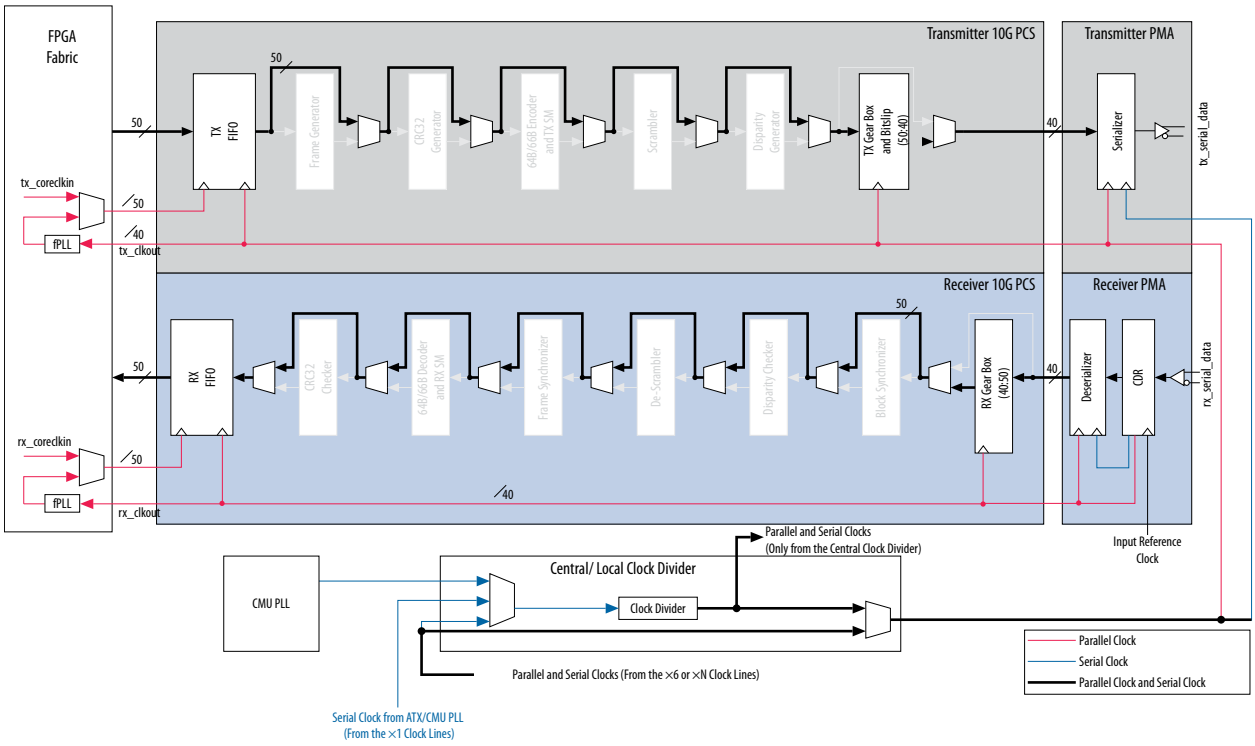
Figure 4-70: 10G PCS Low Latency Datapath with the Gear Box Ratio of 66:40



When the gear box ratio is not an integral multiple of the FPGA fabric interface width (for example, 50:40), you must instantiate a fractional PLL to provide the appropriate clock frequency to the write side of the TX FIFO. Set the division factor in the fractional PLL so that its output frequency is equal to the transmitter or lane data rate divided by 50 for the 50:40 gear box ratio. The clock source that provides the input reference clock to the fractional PLL and the CMU or ATX transmit PLL must be the same because the TX FIFO operates as a phase compensation FIFO, unlike a clock compensation or rate match FIFO. Therefore, the clock requires a zero ppm between the read and write operations.

For the receiver side, enable the `rx_coreclkin` port and connect a second fractional PLL output to the `rx_coreclkin` port. The RX FIFO operates as a phase compensation FIFO. Therefore, the read and write side of the RX FIFO must have a zero ppm difference.

Figure 4-71: 10G PCS Low Latency Datapath with the Gear Box Ratio of 50:40



Note:
(1) The clock source that provides the input reference clock to the fractional PLL (fPLL) and the CMU or ATX PLL (the CMU or ATX PLL generates the high-speed clock for the serializer) must be the same. The transmitter and the receiver FIFOs compensates only for phase differences. Therefore, the same clock source ensures zero ppm between the read and write clocks of the FIFOs.

Using the coreclkin Ports

The `tx_coreclkin` and `rx_coreclkin` ports offer the flexibility to use the `tx_clkout` and `rx_clkout` from one channel to clock the TX and RX FIFOs multiple channels for source synchronous links or if the upstream transmitters are all clocked by the same clock source. The `tx_coreclkin` and `rx_coreclkin` ports require a zero ppm difference between the `tx_clkout` and `rx_clkout` ports, respectively, with a divided-by-50 input frequency.

Related Information

For more information, refer to the "User-Selected Transmitter Datapath Interface Clock" and "User-Selected Receiver Datapath Interface Clock" sections in the Transceiver Clocking in Stratix V Devices chapter

Merging Instances

You can merge transmitter and receiver instances with the different 10G PCS datapath configurations in the same 10 Gbps physical channel.

For example, the Quartus II software allows you to create the two following instances and place them in the same physical transceiver channel:

- Transmitter only instance with a 40-bit FPGA fabric interface
- Receiver only instance with a 64-bit FPGA fabric interface

However, you cannot merge a transmitter instance and receiver instance (1 channel instance) using different PCS blocks (10G PCS and standard PCS) within the same physical transceiver channel.

Transceiver Channel Placement Guidelines

Stratix V devices allow the placement of up to four or five channels when a CMU PLL is used or up to six channels when an ATX PLL is used with Custom and Low Latency datapath configurations with Standard PCS and 10G PCS (same data rate) within the same transceiver bank.

Related Information

Transceiver Channel Placement Guidelines on page 4-80
You can use CMU PLLs or ATX PLLs in non-bonded and bonded configurations.

Native PHY IP Configuration

The Native PHY IP is a full exposure of the transceiver hardware features with little abstraction of the physical hardware layer.

The Stratix V GT transceiver channel is not supported in the Native PHY IP.

Access to both the Standard PCS and 10G PCS hardware, as well as PMA Direct modes can be enabled with full user control over the transceiver interfaces, parameters, and ports. Enable the Standard PCS and 10G PCS or PMA Direct mode to design for multi-datarate protocols, speed negotiation, and support multiple PCS datapath natively on the transceiver link.

The Transceiver Reconfiguration Controller is used to dynamically switch between the Standard PCS and 10G PCS datapaths. In addition, the Reconfiguration Controller is required for calibration, remote loopback enablement, PLL reference clock switching, channel PCS and PLL reconfiguration and switching, and to dynamically adjust PMA transmit pre-emphasis, receiver CDR, CTLE, and DFE advance settings.

Dynamic switching to and from PMA Direct mode is not supported.

Not all hardware combinations are legal or supported, so the user must have sufficient prior knowledge of the transceiver hardware, PLLs, and clocking architecture to determine valid PCS hardware setting, parameters, and combinations. All serial transceiver protocols can be supported by the Native PHY IP.

Note: Altera recommends all new serial protocol designs use the Native PHY IP with the exception of XAUI and PCI Express. A default preset is provided for ASI, SDI, SRIO, CPRI, GIGE, Interlaken, SAS, SATA, and other protocol configurations as well as Low Latency configurations for the Standard PCS and 10G PCS similar to the Low Latency PHY IP implementation. Users can also select the default preset for guidance and then modify the configurations for custom applications and have the ability to save the modified preset.

The transmit CMU or ATX Phase-Locked Loop (PLL) selection is embedded in the PHY IP. In addition, the fractional PLL (fPLL) can now be used as a transmit PLL for lane datarates up to 3.125Gbps. User must select the appropriate PLL for balancing datarate and jitter performance trade-off requirements. Unlike the other PHY IPs, the Native PHY IP does not have an Avalon Memory-Mapped (Avalon-MM) interface as the intent is to have direct access to the port interfaces. As a result, there are no embedded registers. In addition, the reset controller is also not embedded in the Native PHY IP. Altera recommends that the Transceiver PHY Reset Controller IP is used to implement the reset sequence and to make PLL sharing and merging effortless.

To implement a Native PHY link, instantiate the **Stratix V Transceiver Native PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. Select options to generate valid custom transceiver configurations or select the default preset for by double-clicking in the window menu.

Native PHY Transceiver Datapath Configuration

The following figure shows the transceiver Standard PCS blocks, 10G PCS blocks, and their settings in addition to PMA Direct Mode available in a Native PHY IP configuration.

Figure 4-72: Transceiver Blocks in a Native PHY IP Configuration

The Optional PCS blocks that are "Disabled" are not used, but incur latency. The Optional PCS blocks selected as "Bypassed" are not used and do not incur latency.

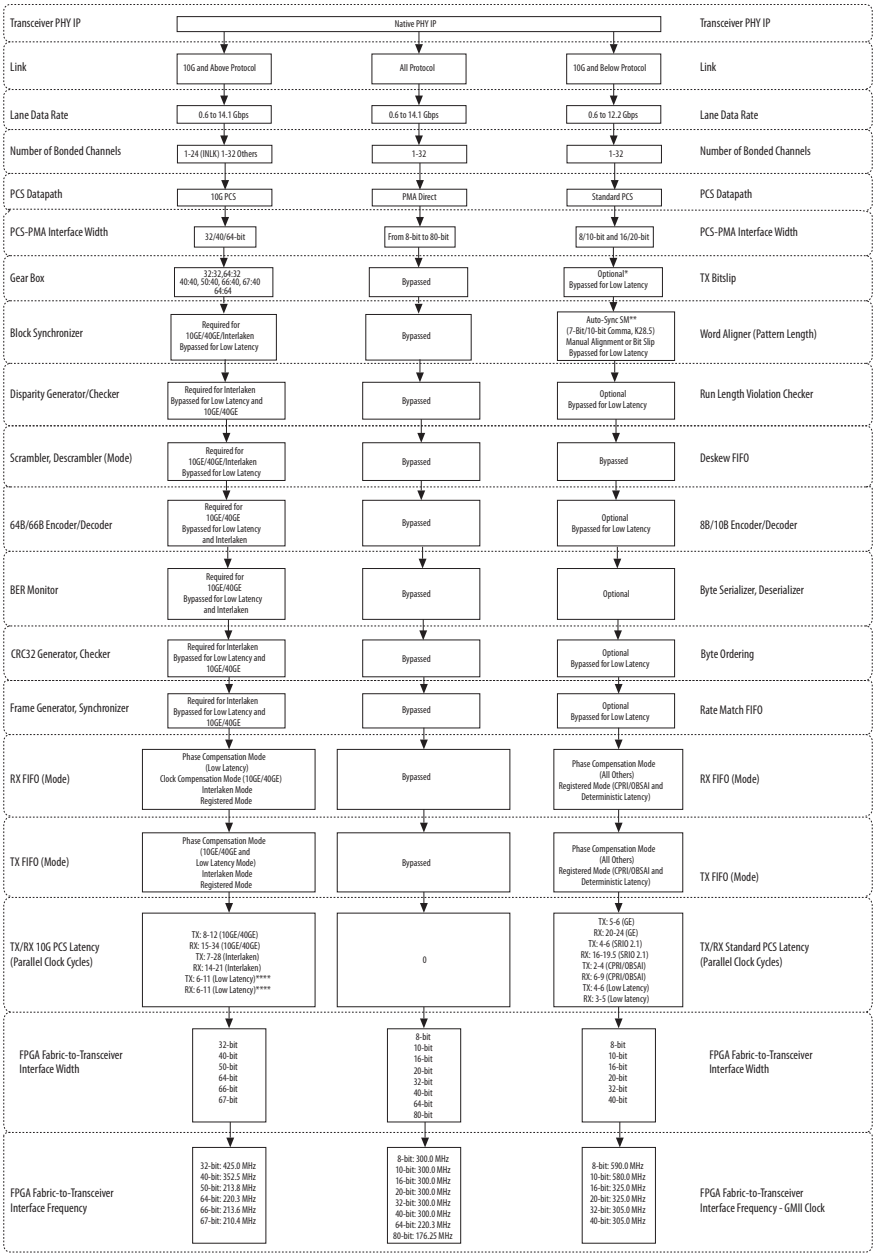
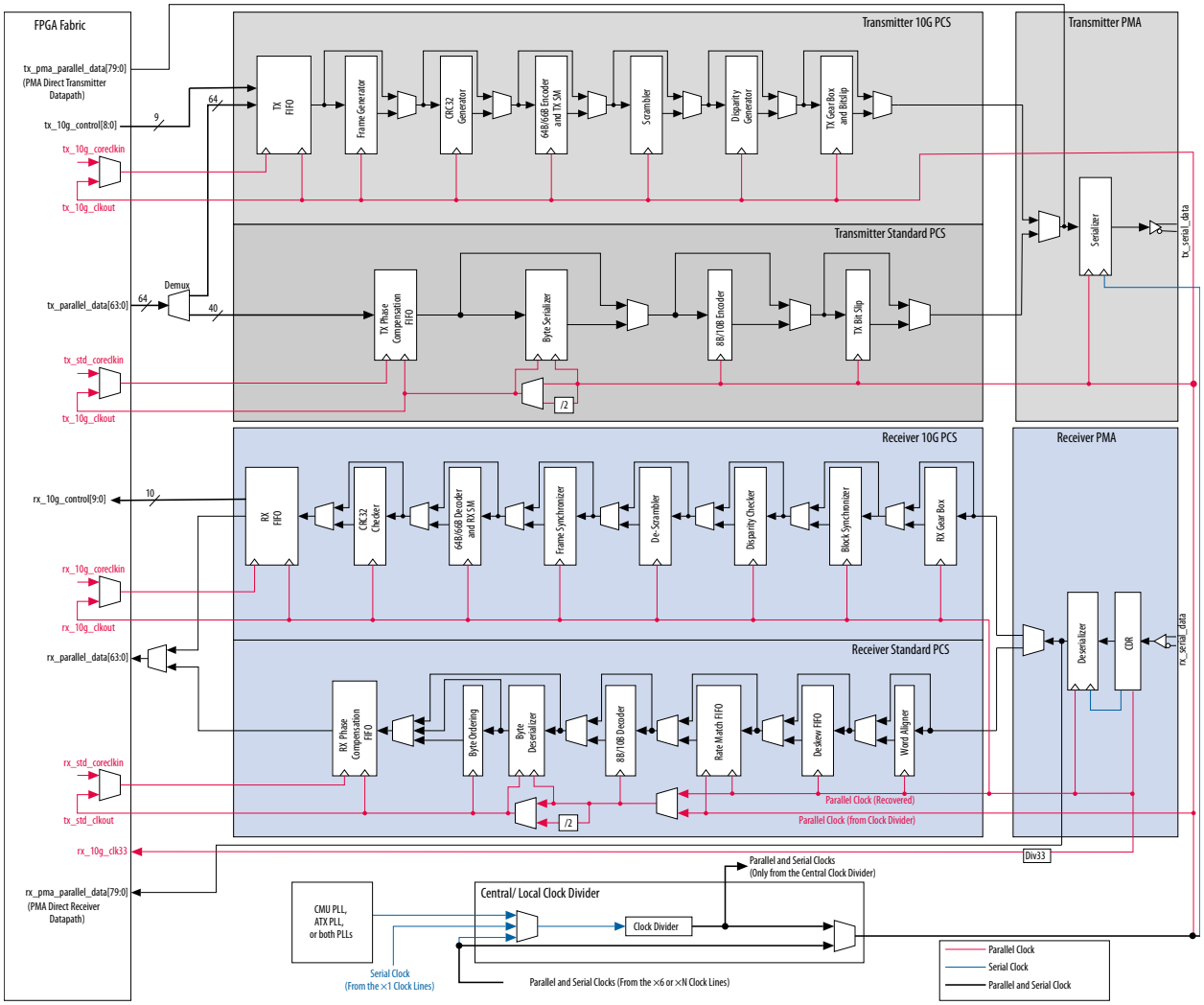


Figure 4-73: Native PHY IP Datapath Configuration

The following figure shows the Standard PCS and 10G PCS blocks, their associated datapaths, and the PMA Direct datapath available for implementation with the Native PHY IP.



Standard PCS Features

The Standard PCS can reach lane datarates up to 12.2 Gbps with the widest PCS-PMA width and FPGA fabric-to-transceiver interface width configuration. The Standard PCS is used when supporting protocols with lane datarates below 10 Gbps such as Gigabit Ethernet, CPRI/OBSAI, SD/HD/3G-SDI, HiGig, Hypertransport, SRIO, JESD204A, SATA and SAS, 1G/2G/4G/8G Fibre Channel, GPON/EPON, SFI-4.2/SFI-5.1, TFI, SPI-4.2/SPI-5.1, STS-12/12c, STS-48/48c, OTU-0.

Standard PCS Receiver and Transmitter Blocks

To implement a Native PHY link with the Standard PCS datapath, instantiate the **Stratix V Transceiver Native PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. Select option to enable the Standard PCS by checking the box. A Standard PCS tab appears with the parameters and configuration options for each block.

The following blocks can be enabled or disabled and configured in the Standard PCS.

- Word Aligner
- Deskew FIFO
- Rate Match FIFO
- 8B/10B Encoder/Decoder
- Byte Serializer/De-Serializer
- Byte Ordering
- Receive Phase Compensation FIFO (Can also be configured as registered mode)
- Transmit Phase Compensation FIFO (Can also be configured as registered mode)
- TX Bitslipper

Related Information

- **Transceiver Architecture in Stratix V Devices**
- **Altera Transceiver PHY IP Core User Guide**

10G PCS Supported Features

The 10G PCS supports protocols with lane datarates that are 10Gbps and above, such as 10/40/100 Gigabit Ethernet, Interlaken, SPAUI, 10G SDI, 10G Fibre Channel, Infiniband, 10G GPON/EPON, SFI-5.2, STS-192/192c, STS-768/768c, OTU-2/3. The 10G PCS can reach lane datarates up to 14.1 Gbps with the widest FPGA fabric-to-transceiver interface width configuration.

10G PCS Receiver and Transmitter Blocks

To implement a Native PHY link with the 10G PCS datapath, instantiate the **Transceiver Native PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. When you select the 10G PCS option, a 10G PCS tab appears with the parameters and configuration options for each block.

The following blocks below can be enabled and disabled and configured in the 10G PCS.

- Receive and Transmit FIFO
- CRC32 Generator/Checker
- Metaframe Generator/Synchronizer
- 64B/66B Encoder/Decoder
- Scrambler/Descrambler
- Disparity Generator/Checker
- Block Synchronizer
- Multi-Gearbox

The hard PCS blocks natively support 10/40/100 Gigabit Ethernet and Interlaken. The remaining protocols are supported via 10G PCS Low Latency datapath configuration with the appropriate gearbox ratios.

10/40/100 Gigabit Ethernet Blocks Supported Configuration:

- Receiver FIFO in Clock Compensation Mode and Transmit FIFO in Phase Compensation Mode
- 64B/66B Encoder/Decoder
- Scrambler/Descrambler
- Block Synchronizer
- 66:40 Gearbox Ratio

10/40/100 Gigabit Ethernet Blocks with 1588 Supported Configuration:

- Receiver and Transmit FIFO in Registered Mode
- 64B/66B Encoder/Decoder
- Scrambler/Descrambler
- Block Synchronizer
- 66:40 Gearbox Ratio

Interlaken Blocks Supported Configuration:

- Receiver and Transmit FIFO in Interlaken Elastic Buffer (Generic) Mode
- CRC32 Generator/Checker
- Metaframe Generator/Synchronizer
- Scrambler/Descrambler
- Disparity Generator/Checker
- Block Synchronizer
- 67:40 Gearbox Ratio

SFI-5.2 Blocks Supported Configuration:

- Receiver and Transmit FIFO in Phase Compensation Mode
- 64:64, 40:40, 64:32, and 32:32 Gearbox Ratios

10G SDI Blocks Supported Configuration:

- Receiver and Transmit FIFO in Phase Compensation Mode
- 50:40 Gearbox Ratio

Other Protocol Blocks Supported Configuration in Basic Mode:

- Receiver and Transmit FIFO in Phase Compensation Mode
- 64:64, 66:40, 40:40, 64:32, and 32:32 Gearbox Ratios

Related Information

- **Transceiver Architecture in Stratix V Devices**
- **Altera Transceiver PHY IP Core User Guide**

Receiver and Transmit Gearbox in Native PHY IP

The Native PHY IP supports many 10G PCS:PMA gearbox ratios.

Users have the freedom to choose the best gearbox ratio that matches their core IP. The 67:40 is mainly used for Interlaken configurations and the 66:40 ratio is mainly used in 10, 40, and 100 Gigabit Ethernet configurations and the 50:40 is used in 10 Gigabit SDI applications. The other ratios can support additional standard communication and transport protocols such as GPON, EPON, SFI-5.2 and OTN.

10G PCS Supported Gearbox Ratios:

- 64:64 PCS:PMA Width
- 67:40 PCS:PMA Width
- 66:40 PCS:PMA Width
- 50:40 PCS:PMA Width
- 40:40 PCS:PMA Width
- 64:32 PCS:PMA Width
- 32:32 PCS:PMA Width

10G Datapath Configurations with Native PHY IP

Table 4-14: 10G PCS Datapath Configurations

The table lists the 10G PCS datapath configuration for 10/40/100 Gigabit Ethernet, 10/40/100 Gigabit Ethernet with 1588, Interlaken, 10G SDI, and other 10G protocols.

Transceiver PHY IP	Native PHY IP					
Link	10/40/ 100GBASE-R/KR	10/40/ 100GBASE-R with 1588	Interlaken	SFI-5.2	10G SDI	Other 10G Protocols (Basic Mode)
Lane Datarate	10.3125Gbps	10.3125Gbps	3.125 - 14.1 Gbps	0.6 - 14.1 Gbps(14)	10.692Gbps	0.6 - 14.1 Gbps (14)
PMA Channel Bonding Option(15) (16)	Non-bonded, xN,feedback compensation	Non-bonded, xN, feedback compensation	Non-bonded	Non-bonded, xN, feedback compensation	Non-bonded, xN, feedback compensation	Non-bonded, xN, feedback compensation
PCS Datapath	10G PCS	10G PCS	10G PCS	10G PCS	10G PCS	10G PCS
PCS-PMA Interface Width (Serialization Factor)	40-bit	40-bit	40-bit	32/40/64-bit	40-bit	32/40/64-bit
Gearbox Ratios	66:40 (17)	66:40 (17)	67:40	32:32, 64:32(17), 40:40, 64:64	50:40 (17)	32:32, 64:32(17), 40:40, 66:40(17), 64:64

(14) Gearbox ratios of 64:32 and 32:32 have a maximum supported datarate of 13.6Gbps.
(15) For xN bonding, the number of bonded channels is up to four using CMU PLL and up to six using ATX PLL, provided the data rate is supported by the CMU PLL and ATX PLL.
(16) Bonding more than six channels requires PLL feedback compensation bonding. PLL feedback compensation bonding requires one PLL per transceiver bank and the PLL reference clock frequency must have the same value as the lane data rate divided by the serialization factor.
(17) May require the use of an internal fractional PLL (fPLL) for selected Gearbox ratio.

Transceiver PHY IP	Native PHY IP					
Link	10/40/ 100GBASE-R/KR	10/40/ 100GBASE-R with 1588	Interlaken	SFI-5.2	10G SDI	Other 10G Protocols (Basic Mode)
Block Synchronizer	Enabled	Enabled	Enabled	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
Disparity Generator, Checker	Bypassed	Bypassed	Enabled	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
Scrambler, Descrambler	Enabled	Enabled	Enabled	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
64B/66B Encoder, Decoder	Enabled	Enabled	Bypassed	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
BER Monitor	Enabled	Enabled	Bypassed	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
CRC32 Generator, Checker	Bypassed	Bypassed	Enabled	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
Frame Generator, Synchronizer	Bypassed	Bypassed	Enabled	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)	Bypassed (Low Latency Mode)
RX FIFO (Mode)	Clock Compensation Mode	Registered Mode	Interlaken Mode	Phase Compensation Mode	Phase Compensation Mode	Phase Compensation Mode (Low Latency Mode)
TX FIFO (Mode)	Phase Compensation Mode	Registered Mode	Interlaken Mode	Phase Compensation Mode	Phase Compensation Mode	Phase Compensation Mode (Low Latency Mode)

Transceiver PHY IP	Native PHY IP					
Link	10/40/ 100GBASE-R/KR	10/40/ 100GBASE-R with 1588	Interlaken	SFI-5.2	10G SDI	Other 10G Protocols (Basic Mode)
TX/RX 10G PCS Latency (Parallel Clock Cycles) (18)	TX: 8-12 RX: 15-34	TX: 1-4 RX: 2-5	TX: 7-28 RX: 14-21	TX: 6-10 (64:32) TX: 7-10 (64:64, 40:40, 32:32) RX: 6-10 (64:32) RX: 7-10 (64:64, 40:40, 32:32)	TX: 7-11 RX: 6-12	TX: 6-10 (64:32) TX: 6-11 (66:40) TX: 7-10 (64:64, 40:40, 32:32) RX: 6-10 (64:32) RX: 6-11 (66:40) RX: 7-10 (64:64, 40:40, 32:32)
FPGA Fabric-to- Transceiver Interface Widths	66-bit	66-bit	67-bit	32-bit 40-bit 64-bit	50-bit	32-bit 40-bit 64-bit 66-bit
FPGA Fabric-to- Transceiver Interface Width Maximum Frequencies	66-bit: 156.25 MHz	66-bit: 156.25 MHz	67-bit: 78.125-312.5 MHz (19)	32-bit (32:32) : 340.0 MHz 40-bit (40:40) : 312.5 MHz 64-bit (64:32) : 170.0 MHz (20) 64-bit (64:64) : 195.4 MHz	50-bit: 213.8 MHz (19)	32-bit (32:32): 340.0 MHz 40-bit (40:40): 312.5 MHz 64-bit (64:32): 170.0 MHz (20) 64-bit (64:64): 195.4 MHz 66-bit (66:40): 189.4 MHz (19)

(18) PCS Latency values are with default recommended FIFO partially full and partially empty values. Disabled if Standard PCS 8B/10 Encoder/Decoder is used.
(19) PCS `tx_clkout` frequency output is lane datarate/40 for 10G-SDI, Interlaken, and Basic Mode.
(20) PCS `tx_clkout` frequency output is lane datarate/32 for SFI-S and Basic Mode.

PMA Direct Supported Features

The PMA Direct is used to support protocols that require extremely low or zero transceiver PCS latency such as QPI. In PMA Direct mode, the transceiver can reach lane data rates up to 14.1Gbps with the widest FPGA fabric-to-transceiver interface width configuration.

There are no PCS blocks in the PMA Direct configuration, so clock phase compensation must be designed in the fabric core. Data and clock signals are interfaced directly to the transceiver PMA. Consequently, you must also compensate for the timing and clock phase differences from the core fabric interface of the FPGA to the transceiver PMA. The PMA interface width has a wide range of selections from 8-bit, 10-bit, 16-bit, 20-bit, 32-bit, 40-bit, 64-bit, and 80-bit. The FPGA fabric interface width is fixed at 80-bit and you must select the correct ports for their PMA interface width configurations.

To implement a Native PHY link with the PMA Direct datapath, instantiate the **Transceiver Native PHY IP** in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. Do not select the options to enable the Standard or 10G PCS. The Standard and 10G PCS tabs do not appear, indicating that the PMA Direct datapath configuration has been selected.

Figure 4-73 shows the transceiver PMA Direct datapath and clocking in the device channels.

Channel and PCS Datapath Dynamic Switching Reconfiguration

The Native PHY IP is the only PHY IP that can support transceiver channel dynamic switching between Standard PCS and 10G PCS. Dynamic switching to and from PMA Direct mode is not supported. The dynamic switching mechanism via streamer-based reconfiguration as well as the associated transceiver PLL, standard PMA, and advance transceiver PMA features reconfiguration is employed with the Reconfiguration Controller IP.

Related Information

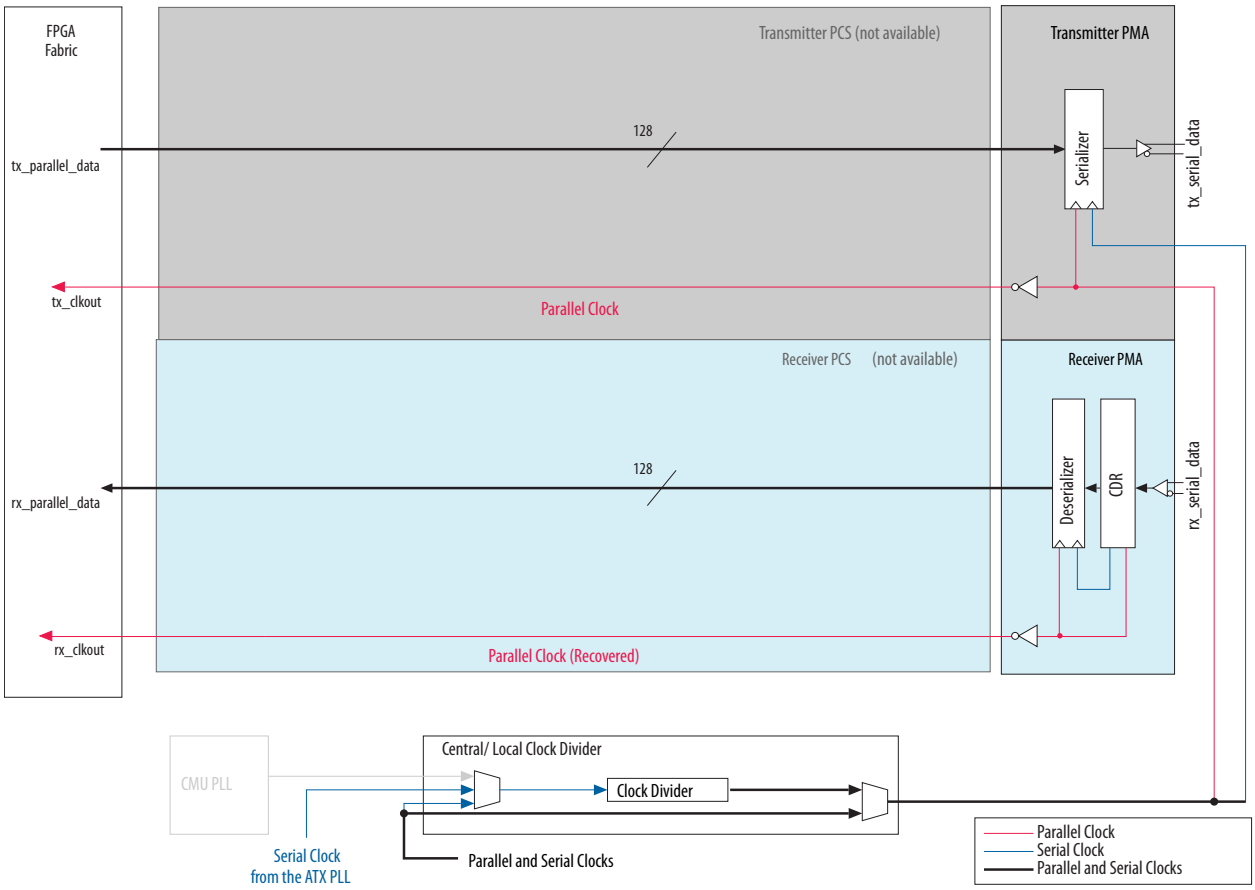
- **Dynamic Reconfiguration in Stratix V Devices**
- **Altera Transceiver PHY IP Core User Guide**

Stratix V GT Device Configurations

Stratix V GT devices contain both 28.05 Gbps GT transceivers and 12.5 Gbps GX transceivers. The GT transceivers can be configured only in PMA-Direct configuration using the Low Latency PHY IP. The GT channels have serial data rates that range from 19.6 Gbps to 28.05 Gbps.

To implement a Low Latency PHY link with the GT channel, instantiate the **Low Latency PHY IP** core in the IP Catalog, under **Transceiver PHY** in the Interfaces menu. In the **Low Latency GUI** under the **General** tab, select **GT** on the **Datapath type** selection.

Figure 4-74: Transceiver Datapath and Clocking in Stratix V GT Channels



Related Information

- **Transceiver Configurations** on page 4-69
  Refer to this section for information about configuring GX transceivers.
- **For the clocking details of Stratix V GT channels, refer to Transceiver Clocking in Stratix V Devices**

Document Revision History

Table 4-15: Document Revision History

Date	Version	Changes
September 2014	2014.09.30	• Added a note about autonegotiation to the "Rate Match FIFO in 1000BASE-X and 1000BASE-KX Configurations" portion of the "1000BASE-X and 1000BASE-KX Supported Features" section. • Added clock labels to the "Native PHY IP Datapath Configuration" figure.
January 2014	2014.01.07	• Added a note to the "Auto-Speed Negotiation Block" section. • Updated "Sample Channel Width Options for Supported Serial Data Rates" table in the "CPRI and OBSAI" section.
October 2013	2013.10.11	• Updated "Advanced Channel Placement Guidelines for PIPE Configurations" section. • Updated "Transceiver Clocking for PCIe Gen3" section.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Added second figure to the "10GBASE-R and 10GBASE-KR" section. • Added the "10GBASE-KR Forward Error Correction" section. • Updated the "Transceiver Channel Placement Guidelines for Gen1, Gen2, and Gen3 PIPE Configurations" section. • Added the "Advance Channel Placement Guidelines for PIPE Configurations" section.

Date	Version	Changes
February 2013	2013.02.21	• Added "Stratix V GT Device Configurations" section. • Updated "Transceiver Clocking and Channel Placement Guidelines" section for PCI Express. • Updated "Transceiver Channel Datapath for XAUI Configuration" and "Transceiver Clocking Diagram for XAUI Configuration" figures.
December 2012	2012.12.17	• Added 10GBASE-KR information. • Added Native PHY information.
June 2012	2.3	• Added “CPRI and OBSAI—Deterministic Latency Protocols” section. • Added “Multi-Lane Deskew Alignment”, “Transmit and Receive FIFO Control and Status”, and “Transceiver Multi-Lane Bonding and Transmit Skew” sections to the Interlaken section. • Updated the “Transceiver Channel Placement Guidelines” section in the XAUI section. • Updated Figure 4–6, Figure 4–12, Figure 4–13, Figure 4–19, Figure 4–20, Figure 4–26, Figure 4–36, Figure 4–38, Figure 4–40, Figure 4–41, Figure 4–42, Figure 4–43, Figure 4–45, Figure 4–46, Figure 4–47, Figure 4–48, Figure 4–50, Figure 4–52. • Updated Table 4–1, Table 4–6, Table 4–7, Table 4–8, Table 4–13, Table 4–14, Table 4–15.

Date	Version	Changes
February 2012	2.2	• Added Figure 4–38 and Figure 4–39. • Updated Figure 4–3, Figure 4–5, Figure 4–6, Figure 4–7, Figure 4–8, Figure 4–9, Figure 4–11, Figure 4–13, Figure 4–14, Figure 4–26, Figure 4–27, Figure 4–38, Figure 4–39, Figure 4–42, and Figure 4–45. • Added Table 4–4. • Updated Table 4–5. • Removed "Transceiver Channel Placement Guidelines" in Interlaken section. • Removed "Transceiver Channel Placement Guidelines" in 10GBASE-R section.
December 2011	2.1	• Updated Figure 4–3, Figure 4–5, Figure 4–7, Figure 4–8, Figure 4–9, Figure 4–11, Figure 4–12, Figure 4–14, Figure 4–15, Figure 4–16, Figure 4–17, Figure 4–19, Figure 4–20, Figure 4–21, Figure 4–22, Figure 4–23, Figure 4–30, Figure 4–31, Figure 4–32, Figure 4–33, Figure 4–34, Figure 4–35, Figure 4–36, Figure 4–43, Figure 4–45, and Figure 4–48. • Updated Table 4–3 and Table 4–9. • Removed "Clock Compensation for Repeater Applications" section. • Minor text edits.
November 2011	2.0	• Merged the 11.0 Transceiver Custom Configurations chapter into this chapter and reorganized this chapter. • Added PCI Gen3 information. • Added Stratix V GT device information. • Removed "GIGE" section.

Date	Version	Changes
May 2011	1.2	• Updated Figure 4–8 and Figure 4–9. • Updated "Supported Features" on page 4–10. • Updated Table 4–5. • Updated Figure 4–16, Figure 4–18, and Figure 4–19. • Added "GIGE" section. • Updated "XAUI" on page 4–38. • Updated "Transceiver Datapath in a XAUI Configuration" on page 4–39. • Updated "Transceiver Channel Placement Guidelines" on page 4–44. • Updated Figure 4–33. • Chapter moved to volume 3 for the 11.0 release.
December 2010	1.1	• Updated "PCI Express (PIPE) 2.0 Interface", "Dynamic Switching Between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signal Rates", "Receiver Status", and "Receiver Detection" sections. • Updated Figure 4–32.
April 2010	1.0	Initial release.

Transceiver Loopback Support in Stratix V Devices

5

2014.09.30

SV52007 Subscribe Send Feedback

The Stratix V loopback options allow you to verify how different functional blocks work in the transceiver.

Related Information

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

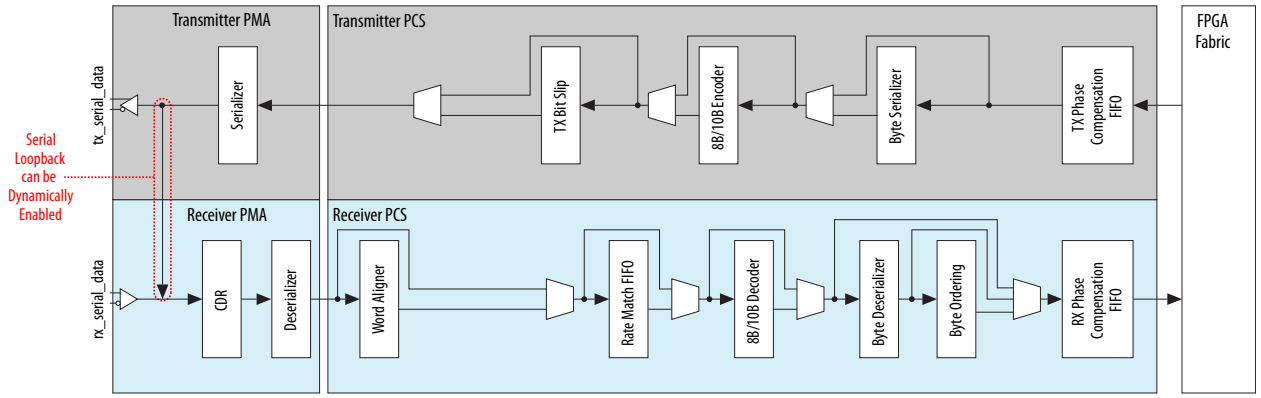
Serial Loopback

Serial loopback is a debugging aid to ensure that 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 5-1: Serial Loopback Datapath



You can enable serial loopback using the PHY IP Parameter Editor or the reconfiguration controller, depending on which PHY IP mode you select. 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.

Note: For more information about the PHY IP core registers, refer to the Altera Transceiver PHY IP User Guide.

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.

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

PIPE Reverse Parallel Loopback

For debugging, the PIPE Reverse Parallel Loopback option uses parallel data through the rate match FIFO, transmitter serializer, and the tx_serial_data port path.

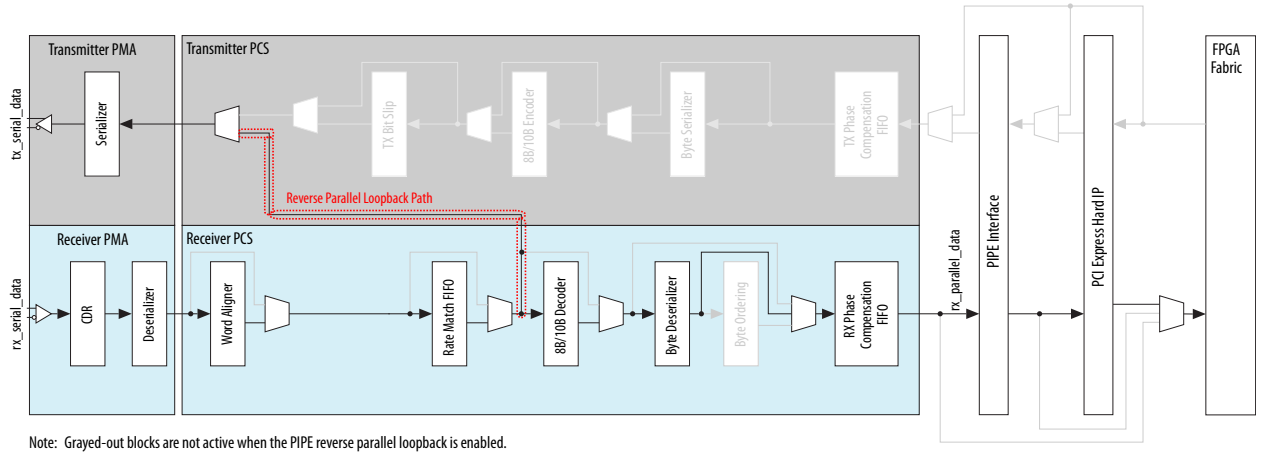
PIPE reverse parallel loopback is only available in the PCIe® configuration for Gen1 and Gen2 data rates. The following figure shows the received serial data passing through the receiver CDR, deserializer, word aligner, and rate match FIFO buffer. The parallel data from the rate match FIFO is then looped 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_detectrxloopback signal.

Note: PIPE reverse parallel loopback is the only loopback option supported in the PCIe configuration. PIPE reverse parallel loopback is not supported in the GT channels of Stratix V GT devices.

Note: For more information, refer to the "PCI Express Reverse Parallel Loopback" section in the Transceiver Configurations in Stratix V Devices chapter.

Figure 5-2: PIPE Reverse Parallel Loopback Configuration Datapath



Note: Grayed-out blocks are not active when the PIPE reverse parallel loopback is enabled.

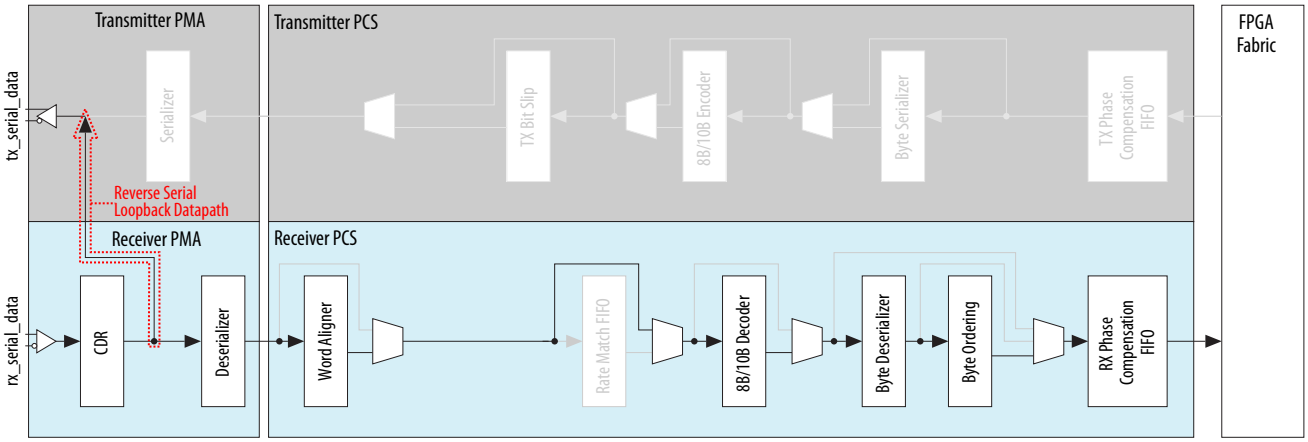
Related Information

Transceiver Configurations in Stratix V Devices

Reverse Serial Loopback

The Reverse Serial Loopback option debugs with data through the `rx_serial_data` port, receiver CDR, and `tx_serial_data` port path.

Figure 5-3: Reverse Serial Loopback Datapath



Note: Grayed-out blocks are not active when the reverse serial loopback is enabled.

Enable reverse serial loopback by accessing the register space within the reconfiguration controller through the Avalon-MM interface.

Note: For the register definitions needed to enable this functionality, 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 out to the `tx_serial_data` port. The received data is also available to the FPGA

fabric through the rx_parallel_data signal. No dynamic pin control is available to select or deselect reverse serial loopback.

You set the reverse serial loopback with the PMA analog registers in the reconfiguration controller.

The only transmitter channel resource used when implementing reverse serial loopback is the transmitter buffer. You can define the V_{OD} and first post tap values on the transmitter buffer using assignment statements in the project .qsf or in the Quartus II Assignment Editor. You can also change these values dynamically with the reconfiguration controller.

Note: For more information about how to dynamically change these analog settings, refer to the Altera Transceiver PHY IP Core User Guide.

Reverse serial loopback is often implemented when using an external bit error rate tester (BERT) on the upstream transmitter.

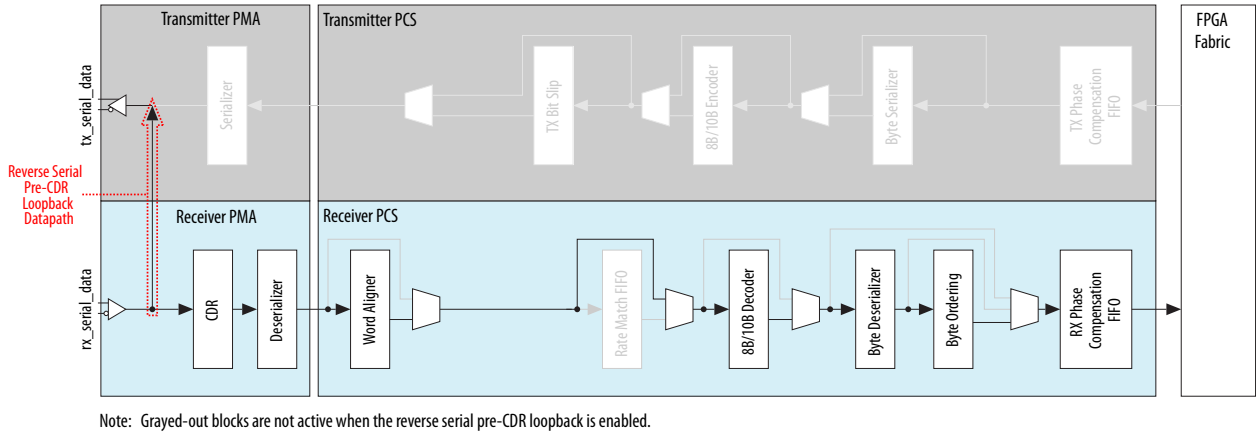
Related Information

Altera Transceiver PHY IP Core User Guide

Reverse Serial Pre-CDR Loopback

The reverse serial pre-CDR loopback option debugs with a data path through the rx_serial_data port to the tx_serial_data port, and before the receiver CDR.

Figure 5-4: Reverse Serial Pre-CDR Loopback Datapath



Note: Grayed-out blocks are not active when the reverse serial pre-CDR loopback is enabled.

Enable the reverse serial pre-CDR loopback by accessing the register space within the reconfiguration controller through the Avalon-MM interface.

Note: For the register definitions needed to enable this functionality, 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 fabric through the rx_parallel_data signal. In pre-CDR reverse loopback, RX input main data passes through the RX buffer, then loops back to the TX directly. There is no clock in this path. No dynamic pin control is available to select or deselect reverse serial pre-CDR loopback.

Set the reverse serial pre-CDR loopback with the PMA analog registers in the reconfiguration controller.

The only transmitter channel resource used when implementing reverse serial pre-CDR loopback is the transmitter buffer. You can change the V_{OD} on the transmitter buffer in the available Parameter Editor of the available PHY IP or using the reconfiguration controller. The receiver data characteristics that are looped back in reverse serial pre-CDR loopback are preserved by the transmitter buffer. The pre-emphasis settings for the transmitter buffer cannot be changed in this configuration.

In post-CDR reverse loopback, the CDR clock can generate pre-emphasis data from the main data and loopback to the TX. However, only 1st post-tap data is generated and sent with the main data to the TX. This is for design and layout cost considerations. The 1st post-tap is the most used tap, and is covered in the test mode.

Related Information

Altera Transceiver PHY IP Core User Guide

Document Revision History

Table 5-1: Document Revision History

Date	Version	Changes
September 2014	2014.09.30	• Changed Reverse Serial Pre-CDR Loopback section to indicate that VOD transmitter buffer settings can be modified through the Parameter Editor and the reconfiguration controller IP. • Changed MegaWizard Plug-in Manager reference to Parameter Editor.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated the Reverse Serial Pre-CDR Loopback topic.
December 2012	2012.12.17	Reorganized content and updated template.
June 2012	2.4	Update for the Quartus II software version 12.0.
February 2012	2.3	Minor edits for clarity.
December 2011	2.2	Updated the document to clarify information.
November 2011	2.1	• This chapter was formerly chapter 6. • There are no content changes for this version of the chapter.
May 2011	2.0	• Added the “Reverse Serial Loopback” and “Reverse Serial Pre-CDR Loopback” sections. • Updated Figure 5–2. • Updated the chapter title. • Chapter moved to Volume 3. • Minor text edits.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

Dynamic Reconfiguration in Stratix V Devices

6

2014.09.30

SV52008 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

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

Dynamic Reconfiguration Features

The following table lists the available dynamic reconfiguration features.

Table 6-1: 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
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
	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.

Offset Cancellation

Offset cancellation adjusts the offsets within the RX PMA and 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`) to the Transceiver Reconfiguration Controller is also used for transceiver calibration and must be within the range of 100-125 MHz. If the clock (`mgmt_clk_clk`) is not free-running, the reconfiguration controller reset (`mgmt_rst_reset`) must be held in reset until the clock is stable.

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

The dynamic reconfiguration controller provides an Avalon ® Memory-Mapped (Avalon-MM) user interface to dynamically reconfigure individual PMA settings on a per channel basis.

Related Information

- **Altera Transceiver PHY IP Core User Guide**
  For information about the read and write operations with the reconfiguration controller
- **AN 645: Dynamic Reconfiguration of PMA Controls in Stratix V Devices**

On-Chip Signal Quality Monitoring (EyeQ)

The bit error rate (BER) eye contour can be used to measure the quality of the received data. EyeQ is a debug and diagnostic tool that analyzes the received data recovery path, including the receiver's gain, noise level, and recovery clock jitter. EyeQ can also measure vertical eye height, effectively allowing a BER eye contour to be plotted.

EyeQ uses a phase interpolator (PI) and sampler (SMP) to estimate the horizontal eye opening. Controlled by a logic generator, the PI generates a sampling clock and the SMP samples the data from the receiver output. The SMP outputs parallel data that is monitored for CRC or BER errors. When the PI output clock phase is shifted by small increments, the data error rate goes from high to low to high if the receiver is good. The number of steps of valid data is defined as the width of the eye. If none of the steps yield valid data, the width of the eye is equal to 0, which means the eye is closed.

The Transceiver Reconfiguration Controller provides an Avalon-MM user interface to enable the EyeQ feature.

Related Information

Transceiver Reconfiguration Controller chapter of the Altera Transceiver PHY IP Core User Guide.
For information about enabling the EyeQ feature.

Decision Feedback Equalization

Decision feedback equalization (DFE) helps compensate for backplane attenuation because of insufficient bandwidth.

DFE works by estimating the intersymbol interference (ISI) that is imposed by the channel on an incoming bit and canceling out the ISI as that bit is sampled by the CDR circuitry. The advantage of DFE is that it boosts the power of the highest frequency component of the received data without increasing its noise power. Use DFE in conjunction with the transmitter pre-emphasis and receiver linear equalization.

The Transceiver Reconfiguration Controller provides an Avalon-MM user interface to step through the DFE tap settings.

Note: In Stratix V GT devices, the GT channels do not support DFE. This feature is supported in the GX channels.

Related Information

"DFE" section in the Transceiver Reconfiguration Controller chapter of the Altera Transceiver PHY IP Core User Guide
For more information about DFE.

Adaptive Equalization

Adaptive equalization (AEQ) solves issues related to changing data rates and backplane losses.

High-speed interface systems require different equalization settings to compensate for changing data rates and backplane losses. Manual tuning of the receiver channel equalization stages involves finding the optimal settings through trial and error, and then locking in those values during compilation. This manual static method is cumbersome and inefficient when system characteristics vary. The AEQ automatically tunes an active receiver channel equalization filter based on a frequency content comparison between the incoming signals and the internally generated reference signals.

In Stratix V GT devices, the GT channels do not support AEQ. This feature is supported in the GX channels.

The Transceiver Reconfiguration Controller provides an Avalon-MM user interface to enable the AEQ feature.

Related Information

"AEQ" section in the Transceiver Reconfiguration Controller chapter of the Altera Transceiver PHY IP Core User Guide
For information about enabling different options and using them to control the AEQ hardware.

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:

- **Serial loopback path**— The output from the serializer is fed back to the CDR. While in this mode, the serializer also feeds the data to the TX output port. Enabling or disabling serial loopback mode is done through the PHY management interface.
- **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 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.

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.

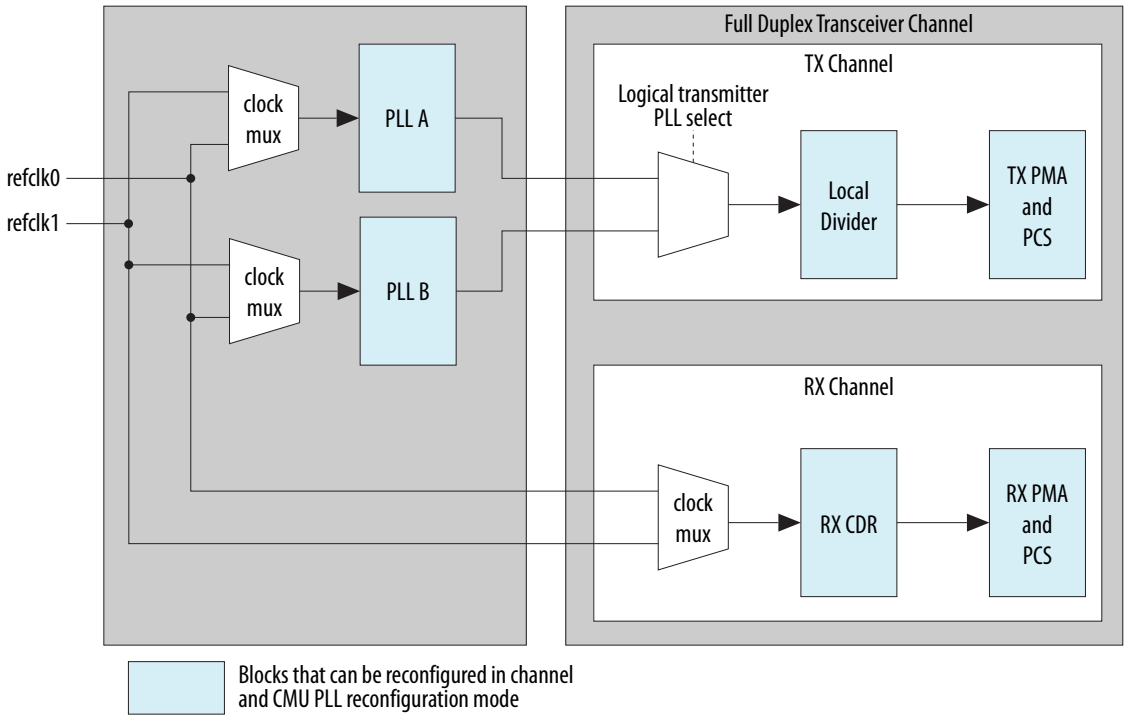
Channel reconfiguration only affects the channel involved in the reconfiguration (the transceiver channel specified by the unique logical channel address), without affecting the remaining transceiver channels

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 6-1: 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.

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	Added link to the known document issues in the Knowledge Base.

Date	Version	Changes
December 2012	2012.12.17	• Rewritten and reorganized content, and updated template
February 2012	2.2	• Updated the "Decision Feedback Equalization" section. • Minor text edits
December 2011	2.1	Minor text edits.

Contents

Logic Array Blocks and Adaptive Logic Modules in Stratix V Devices

1

2015.01.23

SV51002

Subscribe Send Feedback

This chapter describes the features of the logic array block (LAB) in the Stratix® V core fabric.

The LAB is composed of basic building blocks known as adaptive logic modules (ALMs) that you can configure to implement logic functions, arithmetic functions, and register functions.

You can use half of the available LABs in the Stratix V devices as a memory LAB (MLAB).

The Quartus® II software and other supported third-party synthesis tools, in conjunction with parameterized functions such as the library of parameterized modules (LPM), automatically choose the appropriate mode for common functions such as counters, adders, subtractors, and arithmetic functions.

This chapter contains the following sections:

- LAB
- ALM Operating Modes

Related Information

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

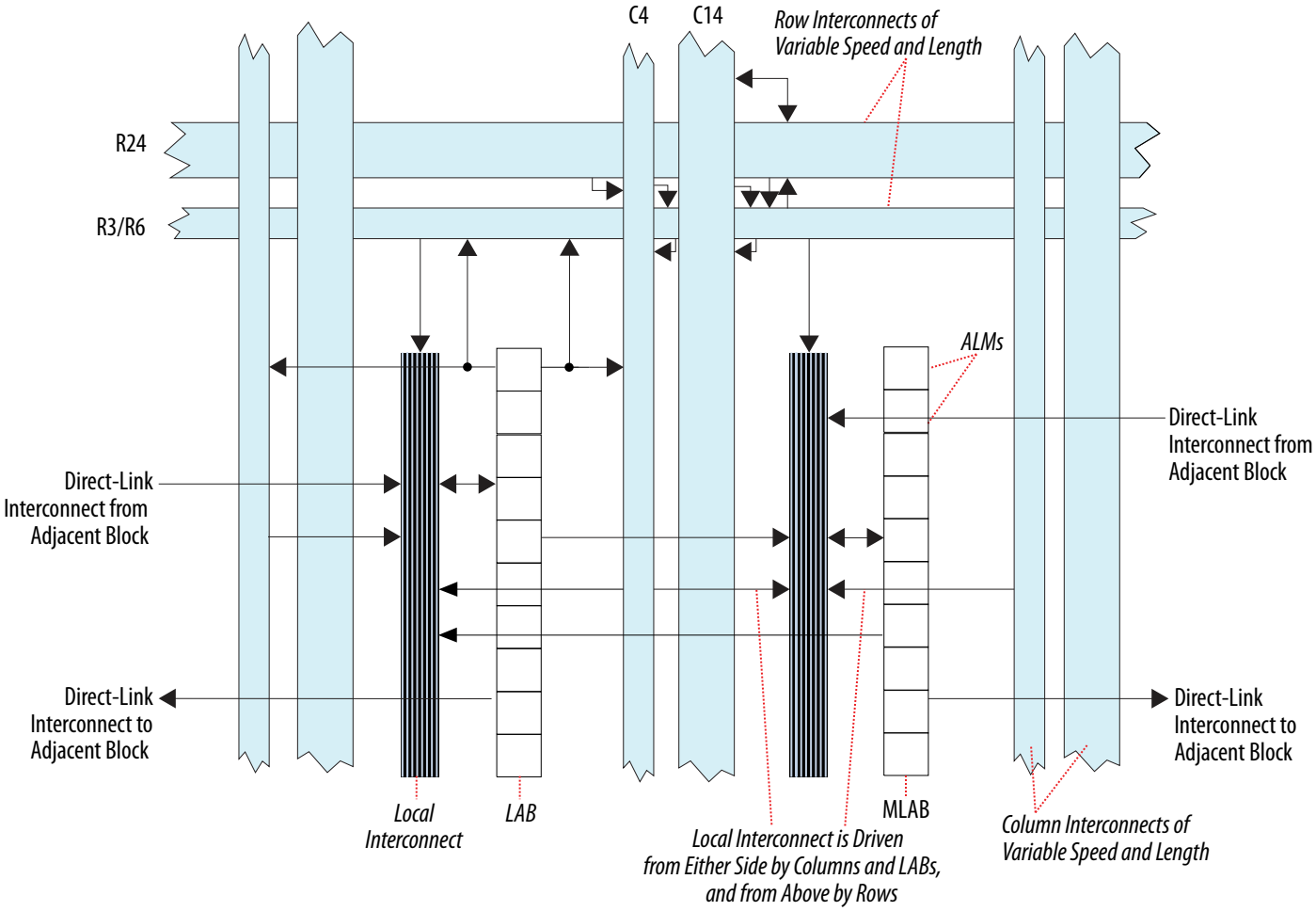
LAB

The LABs are configurable logic blocks that consist of a group of logic resources. Each LAB contains dedicated logic for driving control signals to its ALMs.

MLAB is a superset of the LAB and includes all the LAB features.

Figure 1-1: LAB Structure and Interconnects Overview in Stratix V Devices

This figure shows an overview of the Stratix V LAB and MLAB structure with the LAB interconnects.

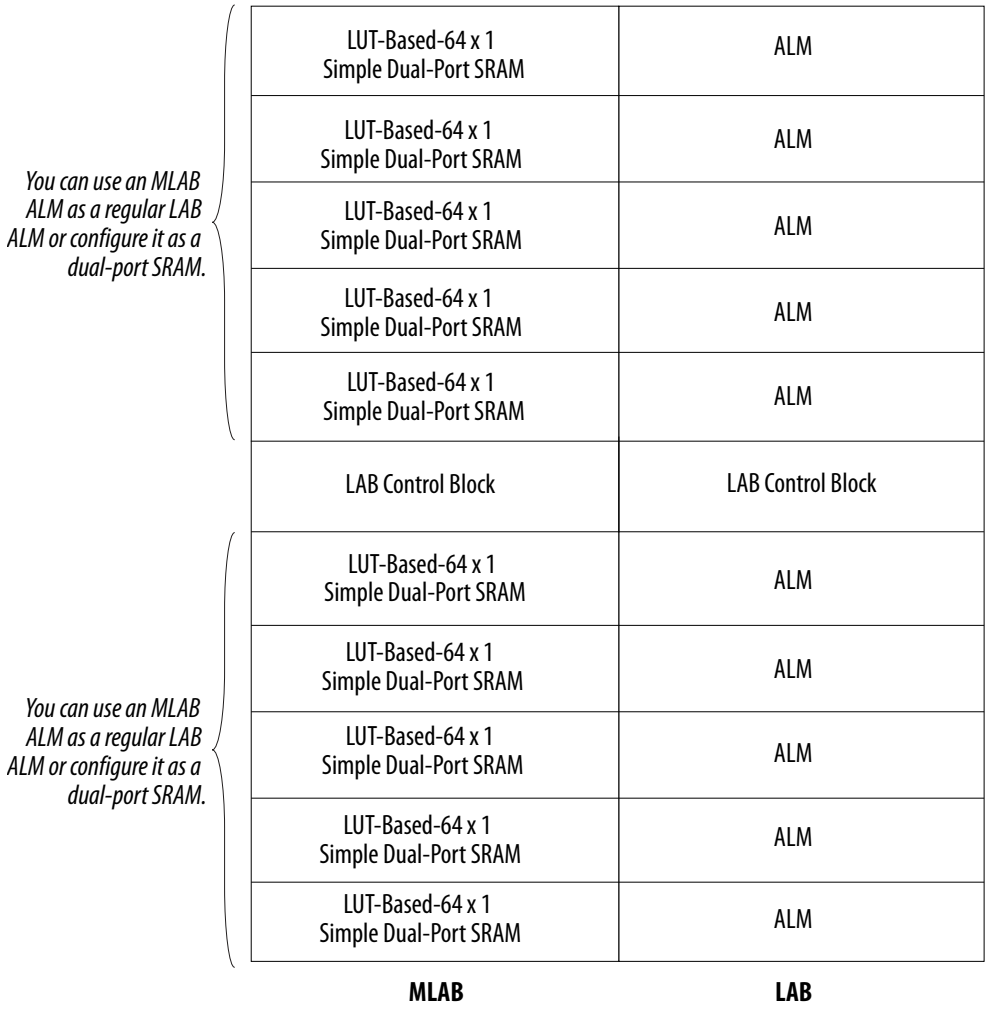


MLAB

Each MLAB supports a maximum of 640 bits of simple dual-port SRAM.

You can configure each ALM in an MLAB as either a 64 × 1 or a 32 × 2 block, resulting in a configuration of either a 64 × 10 or a 32 × 20 simple dual-port SRAM block.

Figure 1-2: LAB and MLAB Structure for Stratix V Devices



Local and Direct Link Interconnects

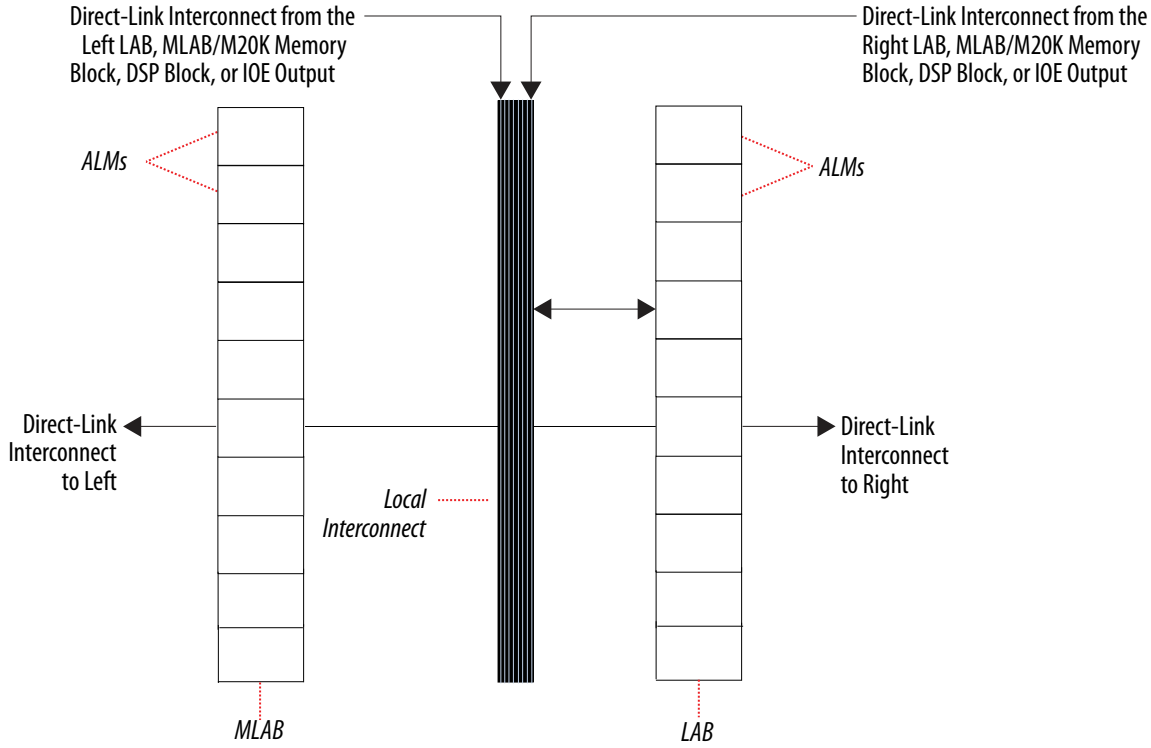
Each LAB can drive 30 ALMs through fast-local and direct-link interconnects. Ten ALMs are in any given LAB and ten ALMs are in each of the adjacent LABs.

The local interconnect can drive ALMs in the same LAB using column and row interconnects and ALM outputs in the same LAB.

Neighboring LABs, MLABs, M20K blocks, or digital signal processing (DSP) blocks from the left or right can also drive the LAB's local interconnect using the direct link connection.

The direct link connection feature minimizes the use of row and column interconnects, providing higher performance and flexibility.

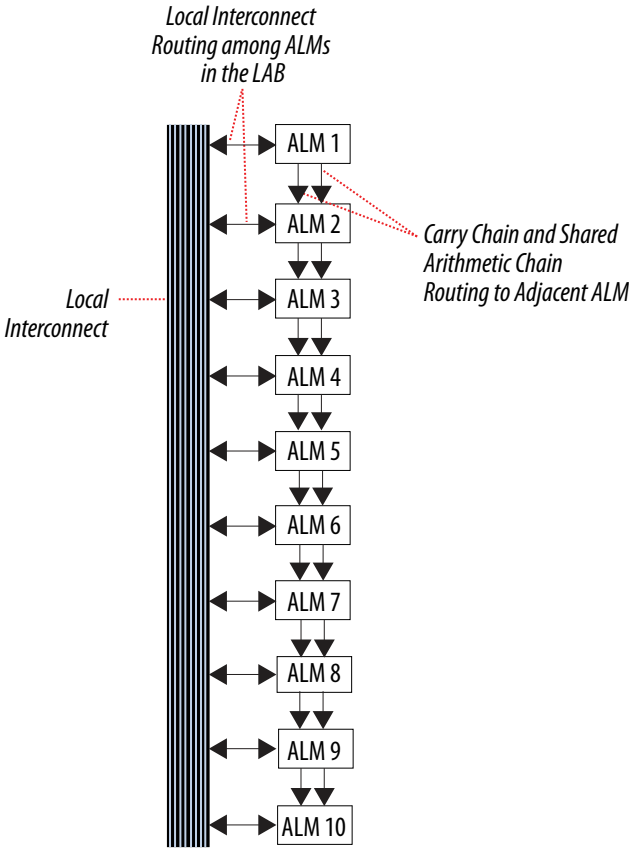
Figure 1-3: LAB Fast Local and Direct Link Interconnects for Stratix V Devices



Shared Arithmetic Chain and Carry Chain Interconnects

There are two dedicated paths between ALMs—carry chain and shared arithmetic chain. Stratix V devices include an enhanced interconnect structure in LABs for routing shared arithmetic chains and carry chains for efficient arithmetic functions. These ALM-to-ALM connections bypass the local interconnect. The Quartus II Compiler automatically takes advantage of these resources to improve utilization and performance.

Figure 1-4: Shared Arithmetic Chain and Carry Chain Interconnects



LAB Control Signals

Each LAB contains dedicated logic for driving the control signals to its ALMs, and has two unique clock sources and three clock enable signals.

The LAB control block generates up to three clocks using the two clock sources and three clock enable signals. Each clock and the clock enable signals are linked.

De-asserting the clock enable signal turns off the corresponding LAB-wide clock.

The LAB row clocks [5..0] and LAB local interconnects generate the LAB-wide control signals. The MultiTrack interconnect's inherent low skew allows clock and control signal distribution in addition to data. The MultiTrack interconnect consists of continuous, performance-optimized routing lines of different lengths and speeds used for inter- and intra-design block connectivity.

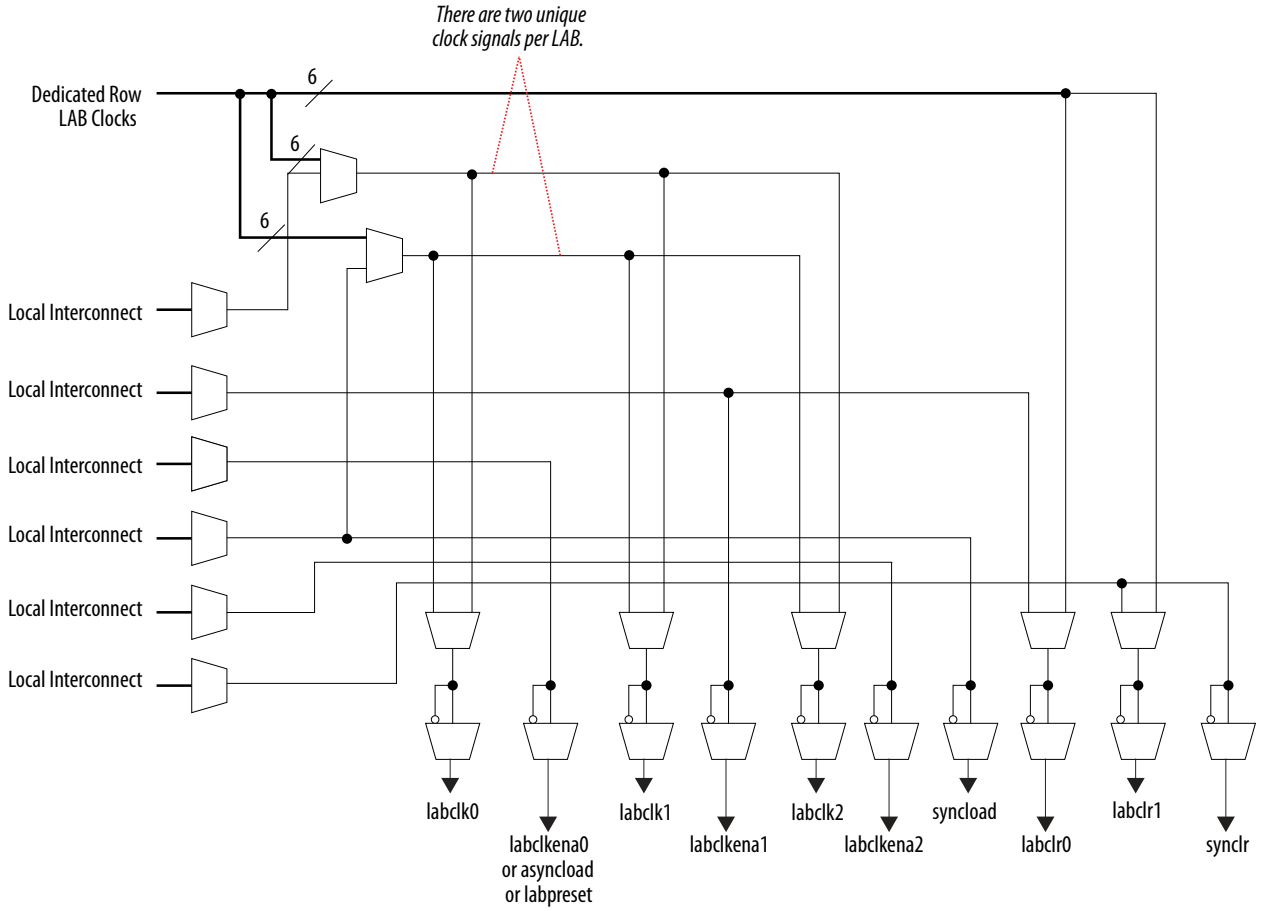
Clear and Preset Logic Control

LAB-wide signals control the logic for the register's clear signal. The ALM directly supports an asynchronous clear function. You can achieve the register preset through the **NOT-gate push-back logic** option in the Quartus II software. Each LAB supports up to two clears.

Stratix V devices provide a device-wide reset pin (`DEV_CLRn`) that resets all the registers in the device. An option set before compilation in the Quartus II software controls this pin. This device-wide reset overrides all other control signals.

Figure 1-5: LAB-Wide Control Signals for Stratix V Devices

This figure shows the clock sources and clock enable signals in a LAB.



ALM Resources

Each ALM contains a variety of LUT-based resources that can be divided between two combinational adaptive LUTs (ALUTs) and four registers.

With up to eight inputs for the two combinational ALUTs, one ALM can implement various combinations of two functions. This adaptability allows an ALM to be completely backward-compatible with four-input LUT architectures. One ALM can also implement any function with up to six inputs and certain seven-input functions.

One ALM contains four programmable registers. Each register has the following ports:

- Data
- Clock
- Synchronous and asynchronous clear
- Synchronous load

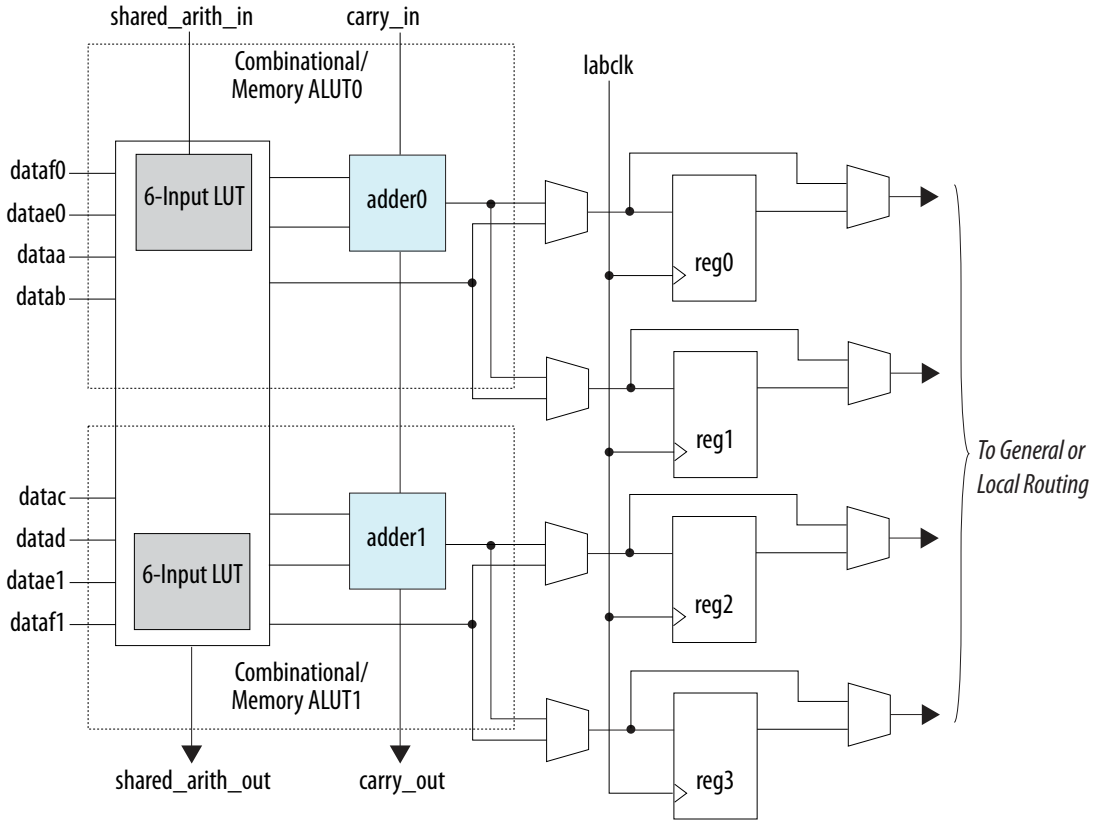
Global signals, general-purpose I/O (GPIO) pins, or any internal logic can drive the clock and clear control signals of an ALM register.

GPIO pins or internal logic drives the clock enable signal.

For combinational functions, the registers are bypassed and the output of the look-up table (LUT) drives directly to the outputs of an ALM.

Note: The Quartus II software automatically configures the ALMs for optimized performance.

Figure 1-6: ALM High-Level Block Diagram for Stratix V Devices



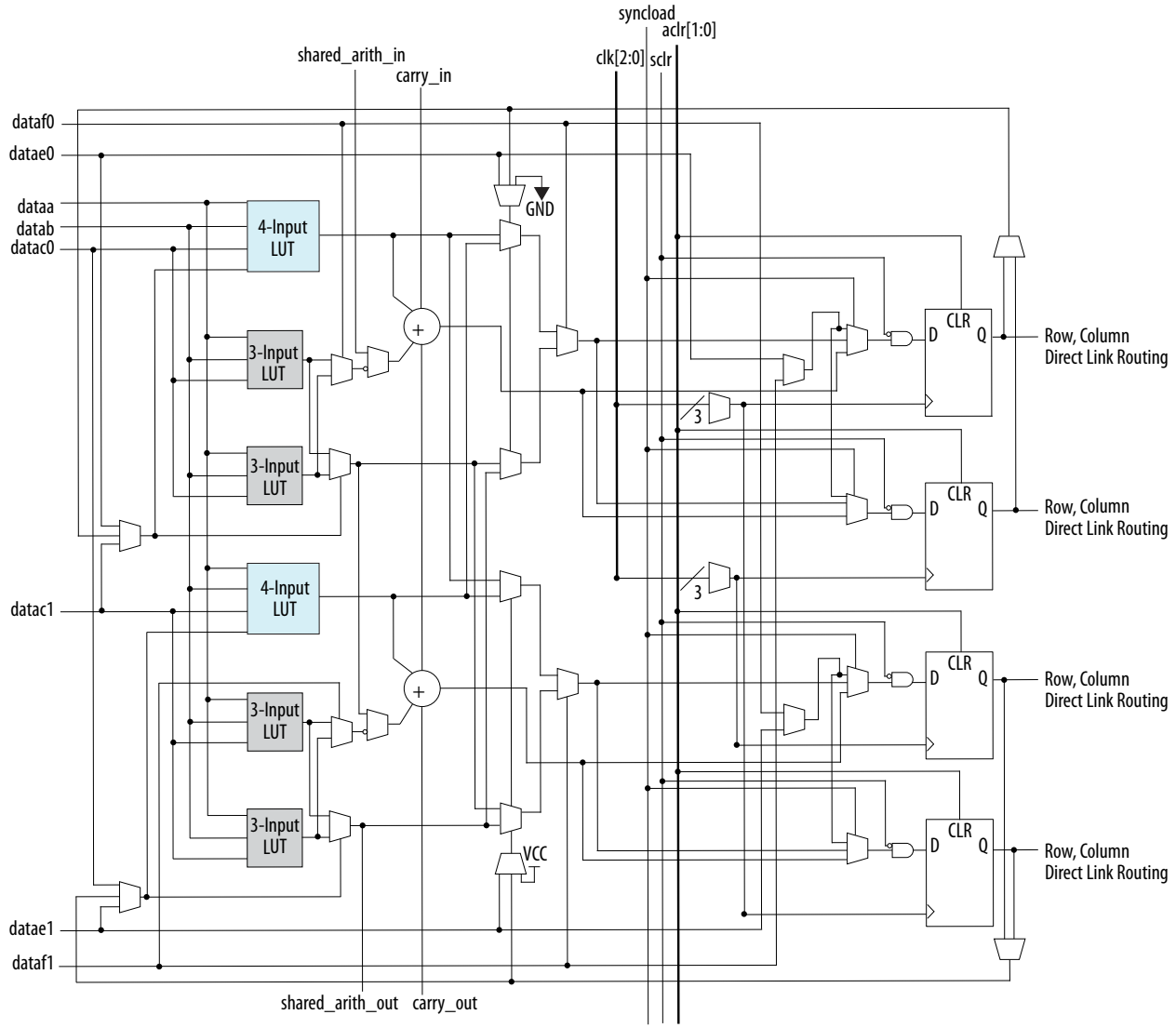
ALM Output

The general routing outputs in each ALM drive the local, row, and column routing resources. Two ALM outputs can drive column, row, or direct link routing connections, and one of these ALM outputs can also drive local interconnect resources.

The LUT, adder, or register output can drive the ALM outputs. The LUT or adder can drive one output while the register drives another output.

Register packing improves device utilization by allowing unrelated register and combinational logic to be packed into a single ALM. Another mechanism to improve fitting is to allow the register output to feed back into the look-up table (LUT) of the same ALM so that the register is packed with its own fan-out LUT. The ALM can also drive out registered and unregistered versions of the LUT or adder output.

Figure 1-7: ALM Connection Details for Stratix V Devices



ALM Operating Modes

The Stratix V ALM operates in any of the following modes:

- Normal mode
- Extended LUT mode
- Arithmetic mode
- Shared arithmetic mode

Normal Mode

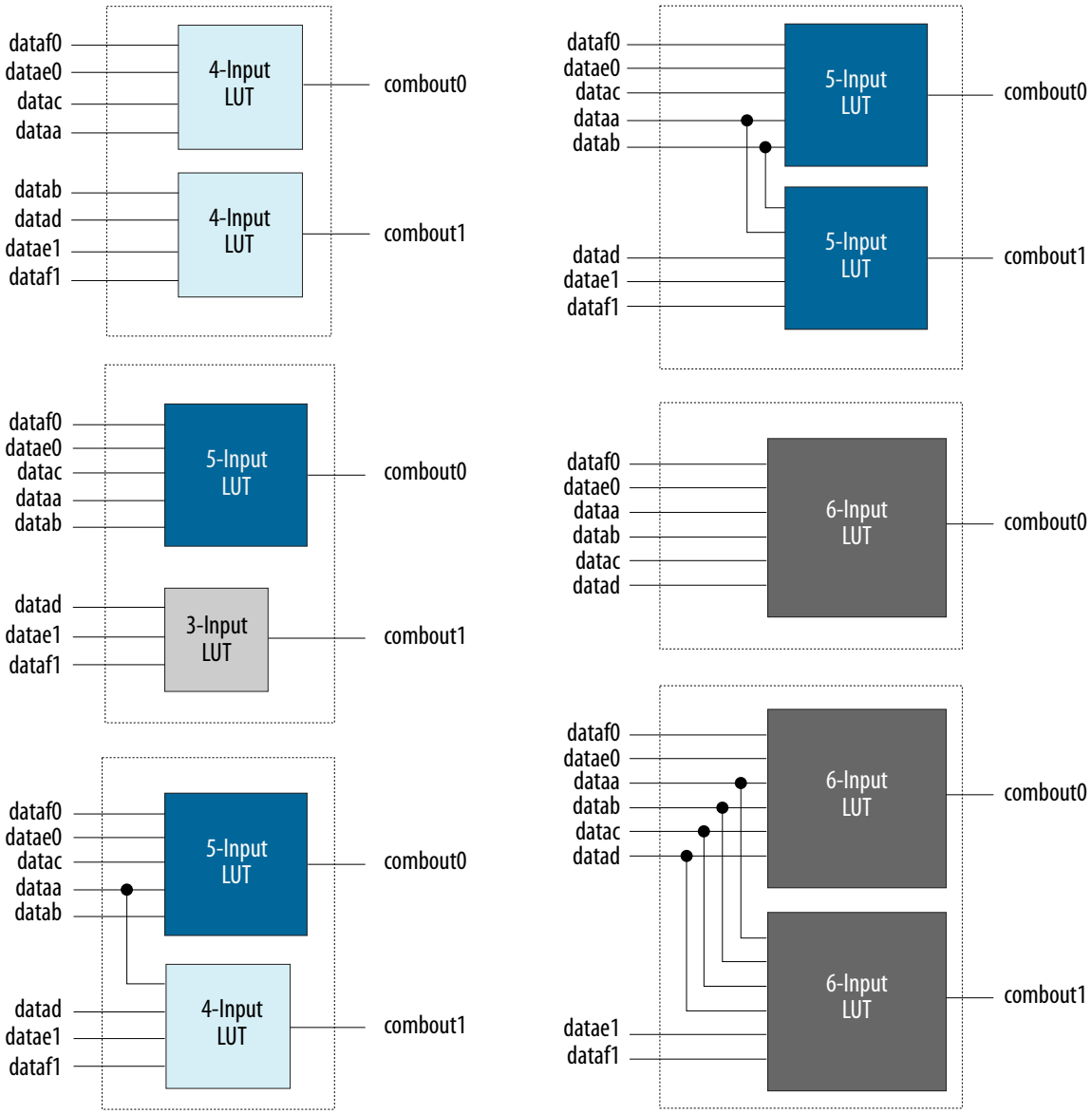
Normal mode allows two functions to be implemented in one Stratix V ALM, or a single function of up to six inputs.

Up to eight data inputs from the LAB local interconnect are inputs to the combinational logic.

The ALM can support certain combinations of completely independent functions and various combinations of functions that have common inputs.

Figure 1-8: ALM in Normal Mode

Combinations of functions with fewer inputs than those shown are also supported. For example, combinations of functions with the following number of inputs are supported: 4 and 3, 3 and 3, 3 and 2, and 5 and 2.



For the packing of 2 five-input functions into one ALM, the functions must have at least two common inputs. The common inputs are `dataa` and `datab`. The combination of a four-input function with a five-input function requires one common input (either `dataa` or `datab`).

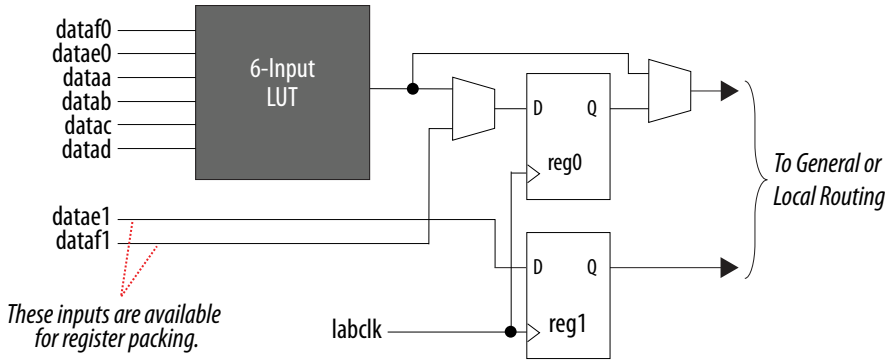
In the case of implementing 2 six-input functions in one ALM, four inputs must be shared and the combinational function must be the same. In a sparsely used device, functions that could be placed in one ALM may be implemented in separate ALMs by the Quartus II software to achieve the best possible performance. As a device begins to fill up, the Quartus II software automatically uses the full potential of the Stratix V ALM. The Quartus II Compiler automatically searches for functions using common inputs or completely independent functions to be placed in one ALM to make efficient use of device resources. In addition, you can manually control resource use by setting location assignments.

You can implement any six-input function using inputs `dataa`, `datab`, `datac`, `datad`, and either `datae0` and `dataf0` or `datae1` and `dataf1`. If you use `datae0` and `dataf0`, the output is either driven to `register0`, `register0` is bypassed, or the output driven to `register0` and `register0` is bypassed, and the data drives out to the interconnect using the top set of output drivers as shown in the following figure. If you use `datae1` and `dataf1`, the output either drives to `register1` or bypasses `register1`, and drives to the interconnect using the bottom set of output drivers. The Quartus II Compiler automatically selects the inputs to the LUT. ALMs in normal mode support register packing.

Figure 1-9: Input Function in Normal Mode

If you use `datae1` and `dataf1` as inputs to a six-input function, `datae0` and `dataf0` are available for register packing.

The `dataf1` input is available for register packing only if the six-input function is unregistered.

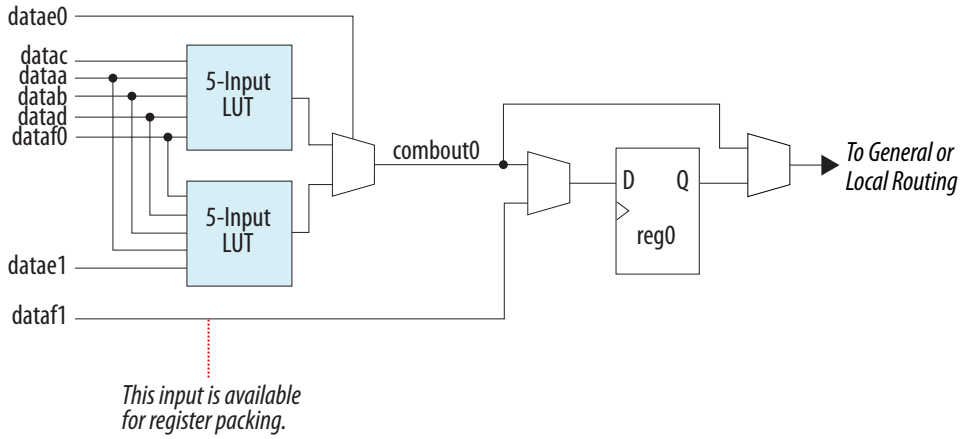


Extended LUT Mode

In this mode, if the 7-input function is unregistered, the unused eighth input is available for register packing.

Functions that fit into the template, as shown in the following figure, often appear in designs as “if-else” statements in Verilog HDL or VHDL code.

Figure 1-10: Template for Supported 7-Input Functions in Extended LUT Mode for Stratix V Devices



Arithmetic Mode

The ALM in arithmetic mode uses two sets of two 4-input LUTs along with two dedicated full adders.

The dedicated adders allow the LUTs to perform pre-adder logic; therefore, each adder can add the output of two 4-input functions.

The ALM supports simultaneous use of the adder's carry output along with combinational logic outputs. The adder output is ignored in this operation.

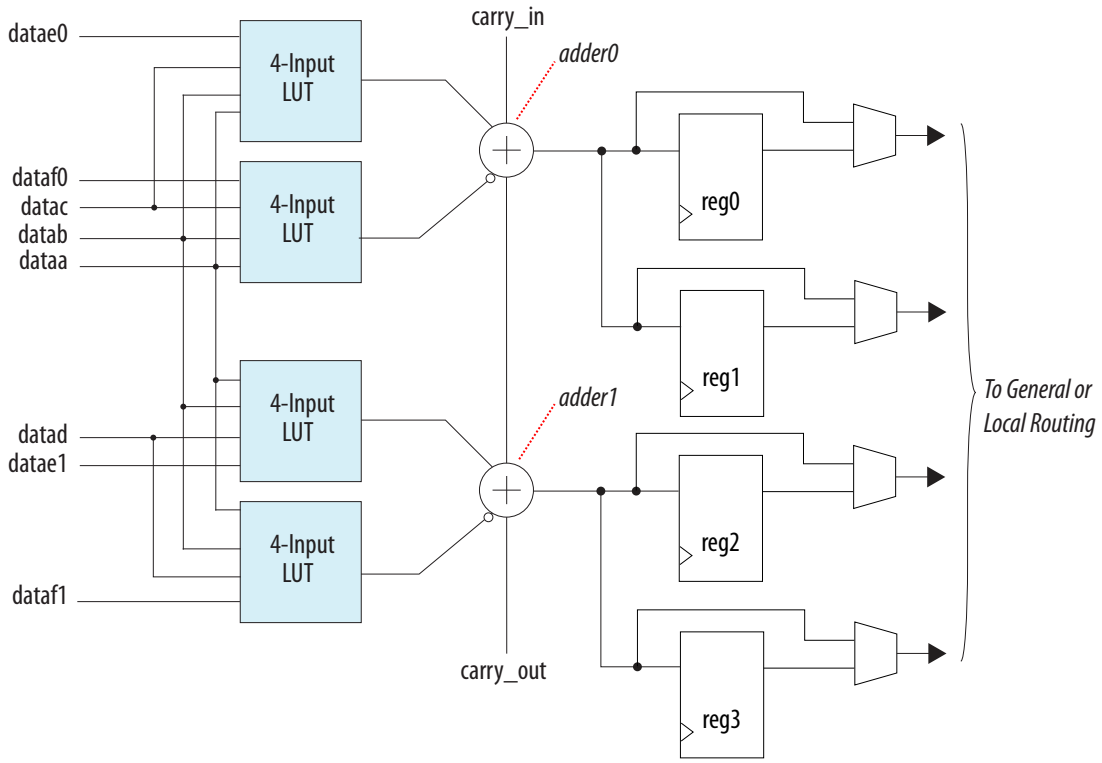
Using the adder with the combinational logic output provides resource savings of up to 50% for functions that can use this mode.

Arithmetic mode also offers clock enable, counter enable, synchronous up and down control, add and subtract control, synchronous clear, and synchronous load.

The LAB local interconnect data inputs generate the clock enable, counter enable, synchronous up/down, and add/subtract control signals. These control signals are good candidates for the inputs that are shared between the four LUTs in the ALM.

The synchronous clear and synchronous load options are LAB-wide signals that affect all registers in the LAB. You can individually disable or enable these signals for each register. The Quartus II software automatically places any registers that are not used by the counter into other LABs.

Figure 1-11: ALM in Arithmetic Mode for Stratix V Devices



Carry Chain

The carry chain provides a fast carry function between the dedicated adders in arithmetic or shared arithmetic mode.

The two-bit carry select feature in Stratix V devices halves the propagation delay of carry chains within the ALM. Carry chains can begin in either the first ALM or the fifth ALM in a LAB. The final carry-out signal is routed to an ALM, where it is fed to local, row, or column interconnects.

To avoid routing congestion in one small area of the device when a high fan-in arithmetic function is implemented, the LAB can support carry chains that only use either the top half or bottom half of the LAB before connecting to the next LAB. This leaves the other half of the ALMs in the LAB available for implementing narrower fan-in functions in normal mode. Carry chains that use the top five ALMs in the first LAB carry into the top half of the ALMs in the next LAB in the column. Carry chains that use the bottom five ALMs in the first LAB carry into the bottom half of the ALMs in the next LAB within the column. You can bypass the top-half of the LAB columns and bottom-half of the MLAB columns.

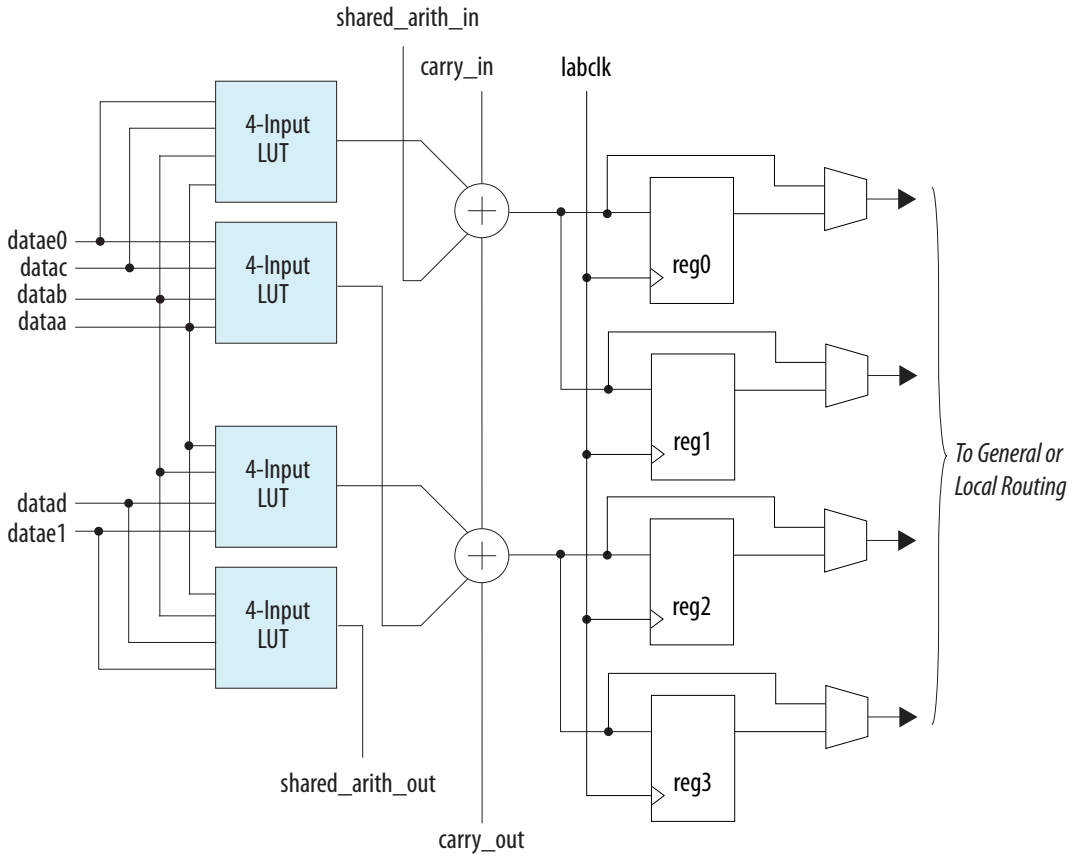
The Quartus II Compiler creates carry chains longer than 20 ALMs (10 ALMs in arithmetic or shared arithmetic mode) by linking LABs together automatically. For enhanced fitting, a long carry chain runs vertically, allowing fast horizontal connections to the TriMatrix memory and DSP blocks. A carry chain can continue as far as a full column.

Shared Arithmetic Mode

The ALM in shared arithmetic mode can implement a 3-input add in the ALM.

This mode configures the ALM with four 4-input LUTs. Each LUT either computes the sum of three inputs or the carry of three inputs. The output of the carry computation is fed to the next adder using a dedicated connection called the shared arithmetic chain.

Figure 1-12: ALM in Shared Arithmetic Mode for Stratix V Devices



Shared Arithmetic Chain

The shared arithmetic chain available in enhanced arithmetic mode allows the ALM to implement a 3-input adder. This significantly reduces the resources necessary to implement large adder trees or correlator functions.

The shared arithmetic chain can begin in either the first or sixth ALM in a LAB.

Similar to carry chains, the top and bottom half of the shared arithmetic chains in alternate LAB columns can be bypassed. This capability allows the shared arithmetic chain to cascade through half of the ALMs in an LAB while leaving the other half available for narrower fan-in functionality. In every LAB, the column is top-half bypassable; while in MLAB, columns are bottom-half bypassable.

The Quartus II Compiler creates shared arithmetic chains longer than 20 ALMs (10 ALMs in arithmetic or shared arithmetic mode) by linking LABs together automatically. To enhance fitting, a long shared arithmetic chain runs vertically, allowing fast horizontal connections to the TriMatrix memory and DSP blocks. A shared arithmetic chain can continue as far as a full column.

LAB Power Management Techniques

The following techniques are used to manage static and dynamic power consumption within the LAB:

- To save AC power, the Quartus II software forces all adder inputs low when the ALM adders are not in use.
- Stratix V LABs operate in high-performance mode or low-power mode. The Quartus II software automatically chooses the appropriate mode for the LAB, based on your design and to optimize speed versus leakage trade-offs.
- Clocks represent a significant portion of dynamic power consumption because of their high switching activity and long paths. The LAB clock that distributes a clock signal to registers within a LAB is a significant contributor to overall clock power consumption. Each LAB's clock and clock enable signals are linked. For example, a combinational ALUT or register in a particular LAB using the `labclk1` signal also uses the `labclkena1` signal. To disable a LAB-wide clock power consumption without disabling the entire clock tree, use the LAB-wide clock enable to gate the LAB-wide clock. The Quartus II software automatically promotes register-level clock enable signals to the LAB-level. All registers within the LAB that share a common clock and clock enable are controlled by a shared, gated clock. To take advantage of these clock enables, use a clock-enable construct in your HDL code for the registered logic.

Related Information

Power Optimization chapter, Quartus II Handbook
Provides more information about implementing static and dynamic power consumption within the LAB.

Document Revision History

Date	Version	Changes
January 2014	2014.01.10	Added multiplexers for the bypass paths and register outputs in the following diagrams: • ALM High-Level Block Diagram for Stratix V Devices • Input Function in Normal Mode • Template for Supported 7-Input Functions in Extended LUT Mode for Stratix V Devices • ALM in Arithmetic Mode for Stratix V Devices • ALM in Shared Arithmetic Mode for Stratix V Devices
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated the available LABs to use as a MLAB. • Removed register chain outputs information in ALM output section. • Moved all links to the Related Information section of respective topics for easy reference.
December 2012	2012.12.28	Reorganized content and updated template.

Date	Version	Changes
June 2012	1.4	• Updated Figure 1–5, Figure 1–6, and Figure 1–12. • Removed register chain expression. • Minor text edits.
November 2011	1.3	• Updated Figure 1–1, Figure 1–4, and Figure 1–6. • Removed “Register Chain” section.
May 2011	1.2	• Chapter moved to volume 2 for the 11.0 release. • Updated Figure 1–6. • Minor text edits.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

Embedded Memory Blocks in Stratix V Devices

2

2015.01.23

SV51003 Subscribe Send Feedback

The embedded memory blocks in the devices are flexible and designed to provide an optimal amount of small- and large-sized memory arrays to fit your design requirements.

Related Information

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

Types of Embedded Memory

The Stratix V devices contain two types of memory blocks:

- 20 Kb M20K blocks—blocks of dedicated memory resources. The M20K blocks are ideal for larger memory arrays while still providing a large number of independent ports.
- 640 bit memory logic array blocks (MLABs)—enhanced memory blocks that are configured from dual-purpose logic array blocks (LABs). The MLABs are ideal for wide and shallow memory arrays. The MLABs are optimized for implementation of shift registers for digital signal processing (DSP) applications, wide shallow FIFO buffers, and filter delay lines. Each MLAB is made up of ten adaptive logic modules (ALMs). In the Stratix V devices, you can configure these ALMs as ten 32 x 2 blocks, giving you one 32 x 20 simple dual-port SRAM block per MLAB. You can also configure these ALMs as ten 64 x 1 blocks, giving you one 64 x 10 simple dual-port SRAM block per MLAB.

Embedded Memory Capacity in Stratix V Devices

Table 2-1: Embedded Memory Capacity and Distribution in Stratix V Devices

Variant	Member Code	M20K		MLAB		Total RAM Bit (Kb)
		Block	RAM Bit (Kb)	Block	RAM Bit (Kb)	
Stratix V GX	A3	957	19,140	6,415	4,009	23,149
	A4	1,900	38,000	7,925	4,953	42,953
	A5	2,304	46,080	9,250	5,781	51,861
	A7	2,560	51,200	11,736	7,335	58,535
	A9	2,640	52,800	15,850	9,906	62,706
	AB	2,640	52,800	17,960	11,225	64,025
	B5	2,100	42,000	9,250	5,781	47,781
	B6	2,660	53,200	11,270	7,043	60,243
	B9	2,640	52,800	15,850	9,906	62,706
	BB	2,640	52,800	17,960	11,225	64,025
Stratix V GT	C5	2,304	46,080	8,020	5,012	51,092
	C7	2,560	51,200	11,735	7,334	58,534
Stratix V GS	D3	688	13,760	4,450	2,781	16,541
	D4	957	19,140	6,792	4,245	23,385
	D5	2,014	40,280	8,630	5,393	45,673
	D6	2,320	46,400	11,000	6,875	53,275
	D8	2,567	51,340	13,120	8,200	59,540
Stratix V E	E9	2,640	52,800	15,850	9,906	62,706
	EB	2,640	52,800	17,960	11,225	64,025

Embedded Memory Design Guidelines for Stratix V Devices

There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

Guideline: Consider the Memory Block Selection

The Quartus II software automatically partitions the user-defined memory into the memory blocks based on your design's speed and size constraints. For example, the Quartus II software may spread out the memory across multiple available memory blocks to increase the performance of the design.

To assign the memory to a specific block size manually, use the RAM megafunction in the MegaWizard™ Plug-In Manager.

For the memory logic array blocks (MLAB), you can implement single-port SRAM through emulation using the Quartus II software. Emulation results in minimal additional use of logic resources.

Because of the dual-purpose architecture of the MLAB, only data input and output registers are available in the block. The MLABs gain read address registers from the ALMs. However, the write address and read data registers are internal to the MLABs.

Guideline: Implement External Conflict Resolution

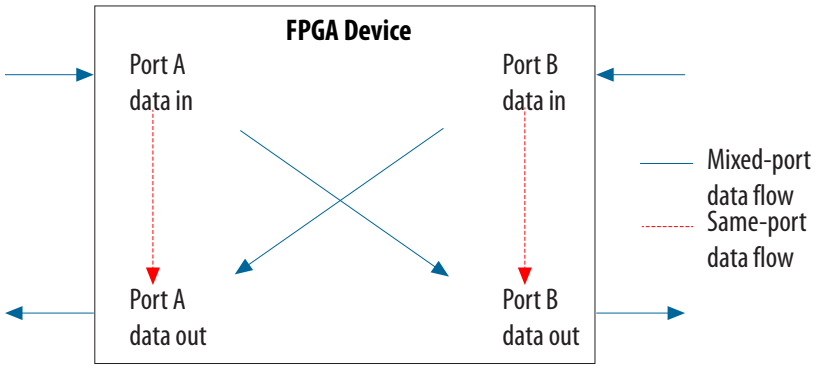
In the true dual-port RAM mode, you can perform two write operations to the same memory location. However, the memory blocks do not have internal conflict resolution circuitry. To avoid unknown data being written to the address, implement external conflict resolution logic to the memory block.

Guideline: Customize Read-During-Write Behavior

Customize the read-during-write behavior of the memory blocks to suit your design requirements.

Figure 2-1: Read-During-Write Data Flow

This figure shows the difference between the two types of read-during-write operations available—same port and mixed port.



Same-Port Read-During-Write Mode

The same-port read-during-write mode applies to a single-port RAM or the same port of a true dual-port RAM.

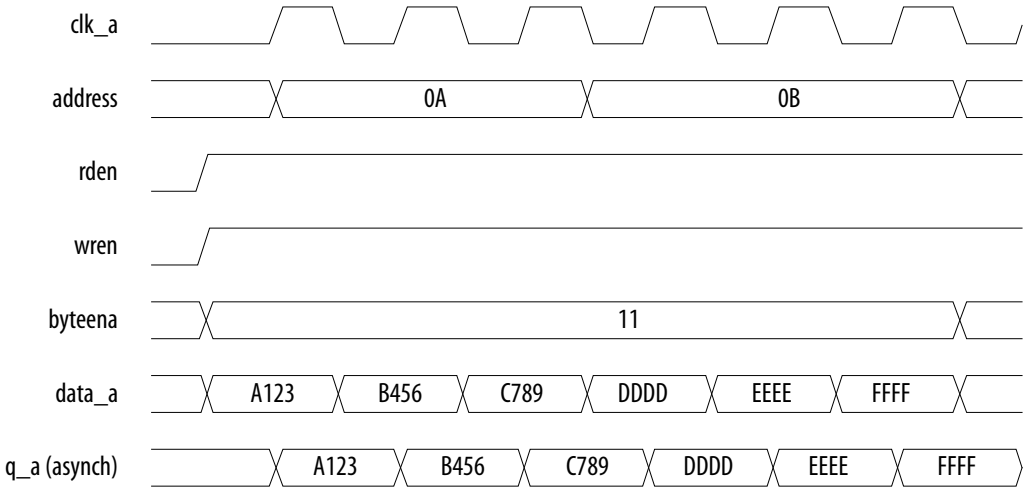
Table 2-2: Output Modes for Embedded Memory Blocks in Same-Port Read-During-Write Mode

This table lists the available output modes if you select the embedded memory blocks in the same-port read-during-write mode.

Output Mode	Memory Type	Description
"new data" (flow-through)	M20K	The new data is available on the rising edge of the same clock cycle on which the new data is written.
"don't care"	MLAB	The RAM outputs "don't care" values for a read-during-write operation.

Figure 2-2: Same-Port Read-During-Write: New Data Mode

This figure shows sample functional waveforms of same-port read-during-write behavior in the “new data” mode.



Mixed-Port Read-During-Write Mode

The mixed-port read-during-write mode applies to simple and true dual-port RAM modes where two ports perform read and write operations on the same memory address using the same clock—one port reading from the address, and the other port writing to it.

Table 2-3: Output Modes for RAM in Mixed-Port Read-During-Write Mode

Output Mode	Memory Type	Description
"new data"	MLAB	A read-during-write operation to different ports causes the MLAB registered output to reflect the “new data” on the next rising edge after the data is written to the MLAB memory. This mode is available only if the output is registered.
"old data"	M20K, MLAB	A read-during-write operation to different ports causes the RAM output to reflect the “old data” value at the particular address. For MLAB, this mode is available only if the output is registered.

Output Mode	Memory Type	Description
"don't care"	M20K, MLAB	The RAM outputs “don’t care” or “unknown” value. • For M20K memory, the Quartus II software does not analyze the timing between write and read operations. • For MLAB, the Quartus II software analyzes the timing between write and read operations by default. To disable this behavior, turn on the **Do not analyze the timing between write and read operation. Metastability issues are prevented by never writing and reading at the same address at the same time** option.
"constrained don't care"	MLAB	The RAM outputs “don’t care” or “unknown” value. The Quartus II software analyzes the timing between write and read operations in the MLAB.

Figure 2-3: Mixed-Port Read-During-Write: New Data Mode

This figure shows a sample functional waveform of mixed-port read-during-write behavior for the “new data” mode.

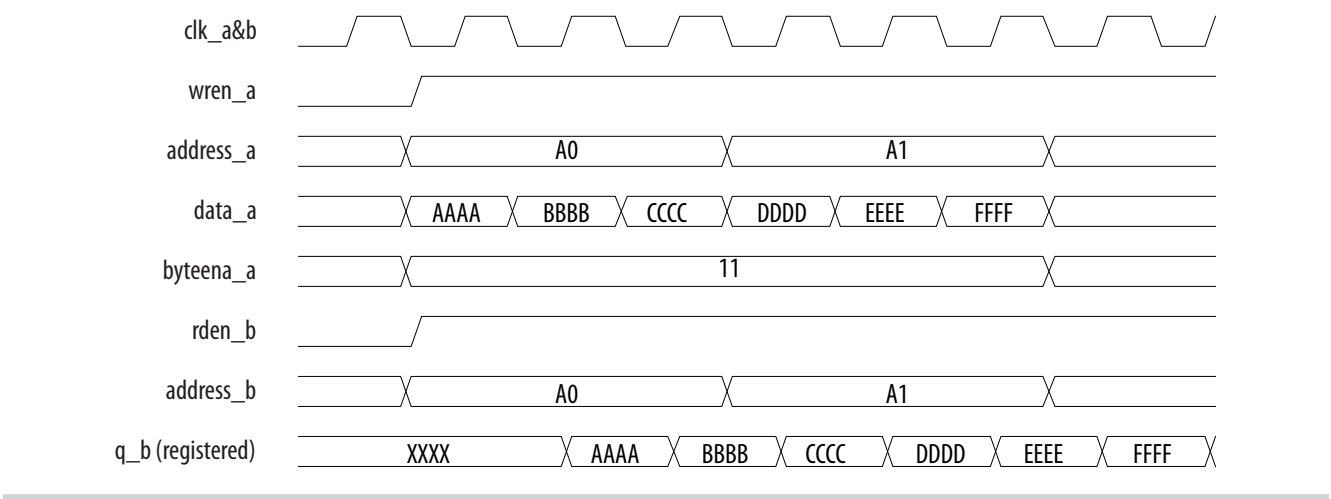


Figure 2-4: Mixed-Port Read-During-Write: Old Data Mode

This figure shows a sample functional waveform of mixed-port read-during-write behavior for the "old data" mode.

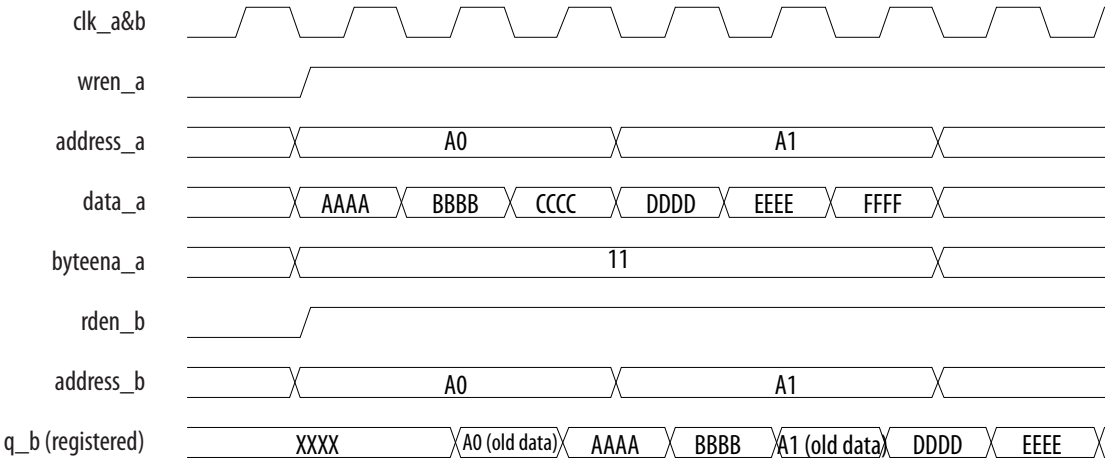
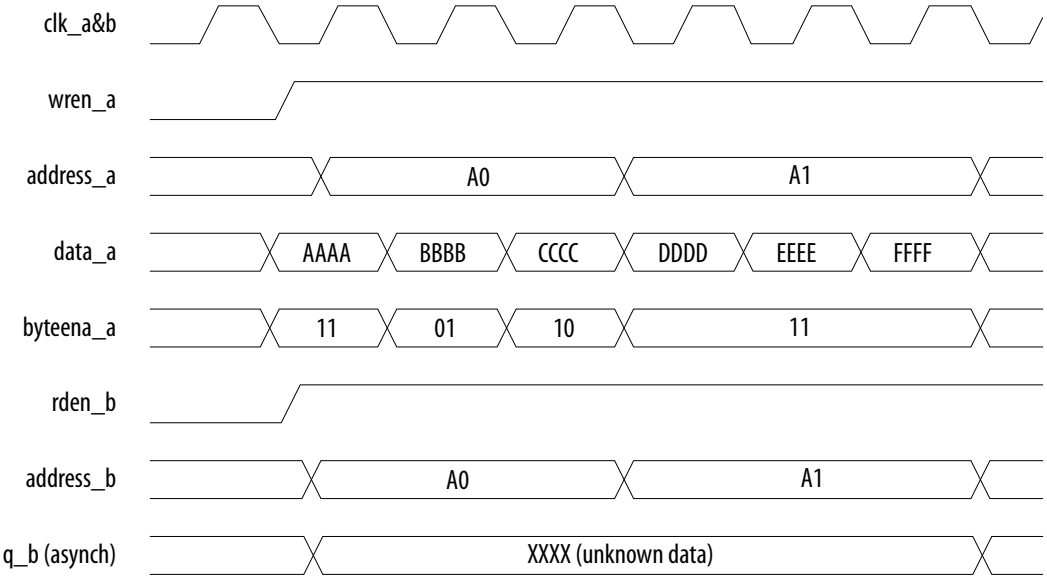


Figure 2-5: Mixed-Port Read-During-Write: Don't Care or Constrained Don't Care Mode

This figure shows a sample functional waveform of mixed-port read-during-write behavior for the "don't care" or "constrained don't care" mode.



In the dual-port RAM mode, the mixed-port read-during-write operation is supported if the input registers have the same clock. The output value during the operation is "unknown."

Related Information

Internal Memory (RAM and ROM) User Guide

Provides more information about the RAM megafunction that controls the read-during-write behavior.

Guideline: Consider Power-Up State and Memory Initialization

Consider the power up state of the different types of memory blocks if you are designing logic that evaluates the initial power-up values, as listed in the following table.

Table 2-4: Initial Power-Up Values of Embedded Memory Blocks

Memory Type	Output Registers	Power Up Value
MLAB	Used	Zero (cleared)
	Bypassed	Read memory contents
M20K	Used	Zero (cleared)
	Bypassed	Zero (cleared)

By default, the Quartus II software initializes the RAM cells in Stratix V devices to zero unless you specify a **.mif**.

All memory blocks support initialization with a **.mif**. You can create **.mif** files in the Quartus II software and specify their use with the RAM megafunction when you instantiate a memory in your design. Even if a memory is pre-initialized (for example, using a **.mif**), it still powers up with its output cleared.

Related Information

- **Internal Memory (RAM and ROM) User Guide**
  Provides more information about **.mif** files.
- **Quartus II Handbook**
  Provides more information about **.mif** files.

Guideline: Control Clocking to Reduce Power Consumption

Reduce AC power consumption in your design by controlling the clocking of each memory block:

- Use the read-enable signal to ensure that read operations occur only when necessary. If your design does not require read-during-write, you can reduce your power consumption by deasserting the read-enable signal during write operations, or during the period when no memory operations occur.
- Use the Quartus II software to automatically place any unused memory blocks in low-power mode to reduce static power.

Embedded Memory Features

Table 2-5: Memory Features in Stratix V Devices

This table summarizes the features supported by the embedded memory blocks.

Features	M20K	MLAB
Maximum operating frequency	600 MHz	600 MHz
Capacity per block (including parity bits)	20,480	640

Features	M20K	MLAB
Parity bits	Supported	Supported
Byte enable	Supported	Supported
Packed mode	Supported	—
Address clock enable	Supported	Supported
Simple dual-port mixed width	Supported	—
True dual-port mixed width	Supported	—
FIFO buffer mixed width	Supported	—
Memory Initialization File (.mif)	Supported	Supported
Mixed-clock mode	Supported	Supported
Fully synchronous memory	Supported	Supported
Asynchronous memory	—	Only for flow-through read memory operations.
Power-up state	Output ports are cleared.	• Registered output ports—Cleared. • Unregistered output ports—Read memory contents.
Asynchronous clears	Output registers and output latches	Output registers and output latches
Write/read operation triggering	Rising clock edges	Rising clock edges
Same-port read-during-write	Output ports set to "new data".	Output ports set to "don't care".
Mixed-port read-during-write	Output ports set to "old data" or "don't care".	Output ports set to "old data", "new data", "don't care", or "constrained don't care".
ECC support	Soft IP support using the Quartus II software. Built-in support in x32-wide simple dual-port mode.	Soft IP support using the Quartus II software.

Related Information

Internal Memory (RAM and ROM) User Guide
Provides more information about the embedded memory features.

Embedded Memory Configurations

Table 2-6: Supported Embedded Memory Block Configurations for Stratix V Devices

This table lists the maximum configurations supported for the embedded memory blocks. The information is applicable only to the single-port RAM and ROM modes.

Memory Block	Depth (bits)	Programmable Width
MLAB	32	x16, x18, or x20
	64	x8, x9, x10
M20K	512	x40, x32
	1K	x20, x16
	2K	x10, x8
	4K	x5, x4
	8K	x2
	16K	x1

Mixed-Width Port Configurations

The mixed-width port configuration is supported in the simple dual-port RAM and true dual-port RAM memory modes.

Note: MLABs do not support mixed-width port configurations.

Related Information

Internal Memory (RAM and ROM) User Guide
Provides more information about dual-port mixed width support.

M20K Blocks Mixed-Width Configurations

The following table lists the mixed-width configurations of the M20K blocks in the simple dual-port RAM mode.

Table 2-7: M20K Block Mixed-Width Configurations (Simple Dual-Port RAM Mode)

Read Port	Write Port									
	16K x 1	8K x 2	4K x 4	4K x 5	2K x 8	2K x 10	1K x 16	1K x 20	512 x 32	512 x 40
16K x 1	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
8K x 2	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
4K x 4	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
4K x 5	—	—	—	Yes	—	Yes	—	Yes	—	Yes
2K x 8	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
2K x 10	—	—	—	Yes	—	Yes	—	Yes	—	Yes
1K x 16	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—

Read Port	Write Port									
	16K x 1	8K x 2	4K x 4	4K x 5	2K x 8	2K x 10	1K x 16	1K x 20	512 x 32	512 x 40
1K x 20	—	—	—	Yes	—	Yes	—	Yes	—	Yes
512 x 32	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
512 x 40	—	—	—	Yes	—	Yes	—	Yes	—	Yes

The following table lists the mixed-width configurations of the M20K blocks in true dual-port mode.

Table 2-8: M20K Block Mixed-Width Configurations (True Dual-Port Mode)

Port A	Port B							
	16K x 1	8K x 2	4K x 4	4K x 5	2K x 8	2K x 10	1K x 16	1K x 20
16K x 1	Yes	Yes	Yes	—	Yes	—	Yes	—
8K x 2	Yes	Yes	Yes	—	Yes	—	Yes	—
4K x 4	Yes	Yes	Yes	—	Yes	—	Yes	—
4K x 5	—	—	—	Yes	—	Yes	—	Yes
2K x 8	Yes	Yes	Yes	—	Yes	—	Yes	—
2K x 10	—	—	—	Yes	—	Yes	—	Yes
1K x 16	Yes	Yes	Yes	—	Yes	—	Yes	—
1K x 20	—	—	—	Yes	—	Yes	—	Yes

Embedded Memory Modes

Caution: To avoid corrupting the memory contents, do not violate the setup or hold time on any of the memory block input registers during read or write operations. This is applicable if you use the memory blocks in single-port RAM, simple dual-port RAM, true dual-port RAM, or ROM mode.

Table 2-9: Memory Modes Supported in the Embedded Memory Blocks

This table lists and describes the memory modes that are supported in the Stratix V embedded memory blocks.

Memory Mode	M20K Support	MLAB Support	Description
Single-port RAM	Yes	Yes	You can perform only one read or one write operation at a time. Use the read enable port to control the RAM output ports behavior during a write operation: • To retain the previous values that are held during the most recent active read enable—create a read-enable port and perform the write operation with the read enable port deasserted. • To show the new data being written, the old data at that address, or a "Don't Care" value when read-during-write occurs at the same address location—do not create a read-enable signal, or activate the read enable during a write operation.
Simple dual-port RAM	Yes	Yes	You can simultaneously perform one read and one write operations to different locations where the write operation happens on port A and the read operation happens on port B.
True dual-port RAM	Yes	—	You can perform any combination of two port operations: two reads, two writes, or one read and one write at two different clock frequencies.
Shift-register	Yes	Yes	You can use the memory blocks as a shift-register block to save logic cells and routing resources. This is useful in DSP applications that require local data storage such as finite impulse response (FIR) filters, pseudo-random number generators, multi-channel filtering, and auto- and cross- correlation functions. Traditionally, the local data storage is implemented with standard flip-flops that exhaust many logic cells for large shift registers. The input data width (w), the length of the taps (m), and the number of taps (n) determine the size of a shift register ($w \times m \times n$). You can cascade memory blocks to implement larger shift registers.

Memory Mode	M20K Support	MLAB Support	Description
ROM	Yes	Yes	You can use the memory blocks as ROM. • Initialize the ROM contents of the memory blocks using a **.mif** or **.hex**. • The address lines of the ROM are registered on M20K blocks but can be unregistered on MLABs. • The outputs can be registered or unregistered. • The output registers can be asynchronously cleared. • The ROM read operation is identical to the read operation in the single-port RAM configuration.
FIFO	Yes	Yes	You can use the memory blocks as FIFO buffers. Use the SCFIFO and DCFIFO megafunctions to implement single- and dual-clock asynchronous FIFO buffers in your design. For designs with many small and shallow FIFO buffers, the MLABs are ideal for the FIFO mode. However, the MLABs do not support mixed-width FIFO mode.

Related Information

- **Internal Memory (RAM and ROM) User Guide**
  Provides more information memory modes.
- **RAM-Based Shift Register (ALTSHIFT_TAPS) Megafunction User Guide**
  Provides more information about implementing the shift register mode.
- **SCFIFO and DCFIFO Megafunctions User Guide**
  Provides more information about implementing FIFO buffers.

Embedded Memory Clocking Modes

This section describes the clocking modes for the Stratix V memory blocks.

Caution: To avoid corrupting the memory contents, do not violate the setup or hold time on any of the memory block input registers during read or write operations.

Clocking Modes for Each Memory Mode

Table 2-10: Memory Blocks Clocking Modes Supported for Each Memory Mode

Clocking Mode	Memory Mode				
	Single-Port	Simple Dual-Port	True Dual-Port	ROM	FIFO
Single clock mode	Yes	Yes	Yes	Yes	Yes
Read/write clock mode	—	Yes	—	—	Yes
Input/output clock mode	Yes	Yes	Yes	Yes	—

Clocking Mode	Memory Mode				
	Single-Port	Simple Dual-Port	True Dual-Port	ROM	FIFO
Independent clock mode	—	—	Yes	Yes	—

Note: The clock enable signals are not supported for write address, byte enable, and data input registers on MLAB blocks.

Single Clock Mode

In the single clock mode, a single clock, together with a clock enable, controls all registers of the memory block.

Read/Write Clock Mode

In the read/write clock mode, a separate clock is available for each read and write port. A read clock controls the data-output, read-address, and read-enable registers. A write clock controls the data-input, write-address, write-enable, and byte enable registers.

Input/Output Clock Mode

In input/output clock mode, a separate clock is available for each input and output port. An input clock controls all registers related to the data input to the memory block including data, address, byte enables, read enables, and write enables. An output clock controls the data output registers.

Independent Clock Mode

In the independent clock mode, a separate clock is available for each port (A and B). Clock A controls all registers on the port A side; clock B controls all registers on the port B side.

Note: You can create independent clock enable for different input and output registers to control the shut down of a particular register for power saving purposes. From the parameter editor, click **More Options** (beside the clock enable option) to set the available independent clock enable that you prefer.

Asynchronous Clears in Clocking Modes

In all clocking modes, asynchronous clears are available only for output latches and output registers. For the independent clock mode, this is applicable on both ports.

Output Read Data in Simultaneous Read/Write

If you perform a simultaneous read/write to the same address location using the read/write clock mode, the output read data is unknown. If you require the output read data to be a known value, use single-clock or input/output clock mode and select the appropriate read-during-write behavior in the MegaWizard™ Plug-In Manager.

Note: MLAB memory blocks only support simultaneous read/write operations when operating in single clock mode.

Independent Clock Enables in Clocking Modes

Independent clock enables are supported in the following clocking modes:

- Read/write clock mode—supported for both the read and write clocks.
- Independent clock mode—supported for the registers of both ports.

To save power, you can control the shut down of a particular register using the clock enables.

Related Information

Guideline: Control Clocking to Reduce Power Consumption on page 2-7

Parity Bit in Memory Blocks

Table 2-11: Parity Bit Support for the Embedded Memory Blocks

This table describes the parity bit support for the memory blocks.

M20K	MLAB
• The parity bit is the fifth bit associated with each 4 data bits in data widths of 5, 10, 20, and 40 (bits 4, 9, 14, 19, 24, 29, 34, and 39). • In non-parity data widths, the parity bits are skipped during read or write operations. • Parity function is not performed on the parity bit.	• The parity bit is the ninth bit associated with each byte. • The ninth bit can store a parity bit or serve as an additional bit. • Parity function is not performed on the parity bit.

Byte Enable in Embedded Memory Blocks

The embedded memory blocks support byte enable controls:

- The byte enable controls mask the input data so that only specific bytes of data are written. The unwritten bytes retain the values written previously.
- The write enable (`wren`) signal, together with the byte enable (`byteena`) signal, control the write operations on the RAM blocks. By default, the `byteena` signal is high (enabled) and only the `wren` signal controls the writing.
- The byte enable registers do not have a `clear` port.
- If you are using parity bits, on the M20K blocks, the byte enable function controls 8 data bits and 2 parity bits; on the MLABs, the byte enable function controls all 10 bits in the widest mode.
- The LSB of the `byteena` signal corresponds to the LSB of the data bus.
- The byte enables are active high.

Byte Enable Controls in Memory Blocks

Table 2-12: byteena Controls in x20 Data Width

byteena[1:0]	Data Bits Written	
11 (default)	[19:10]	[9:0]
10	[19:10]	—
01	—	[9:0]

Table 2-13: byteena Controls in x40 Data Width

byteena[3:0]	Data Bits Written			
1111 (default)	[39:30]	[29:20]	[19:10]	[9:0]
1000	[39:30]	—	—	—
0100	—	[29:20]	—	—
0010	—	—	[19:10]	—
0001	—	—	—	[9:0]

Note: If you use the ECC feature on the M20K blocks, you cannot use the byte enable feature.

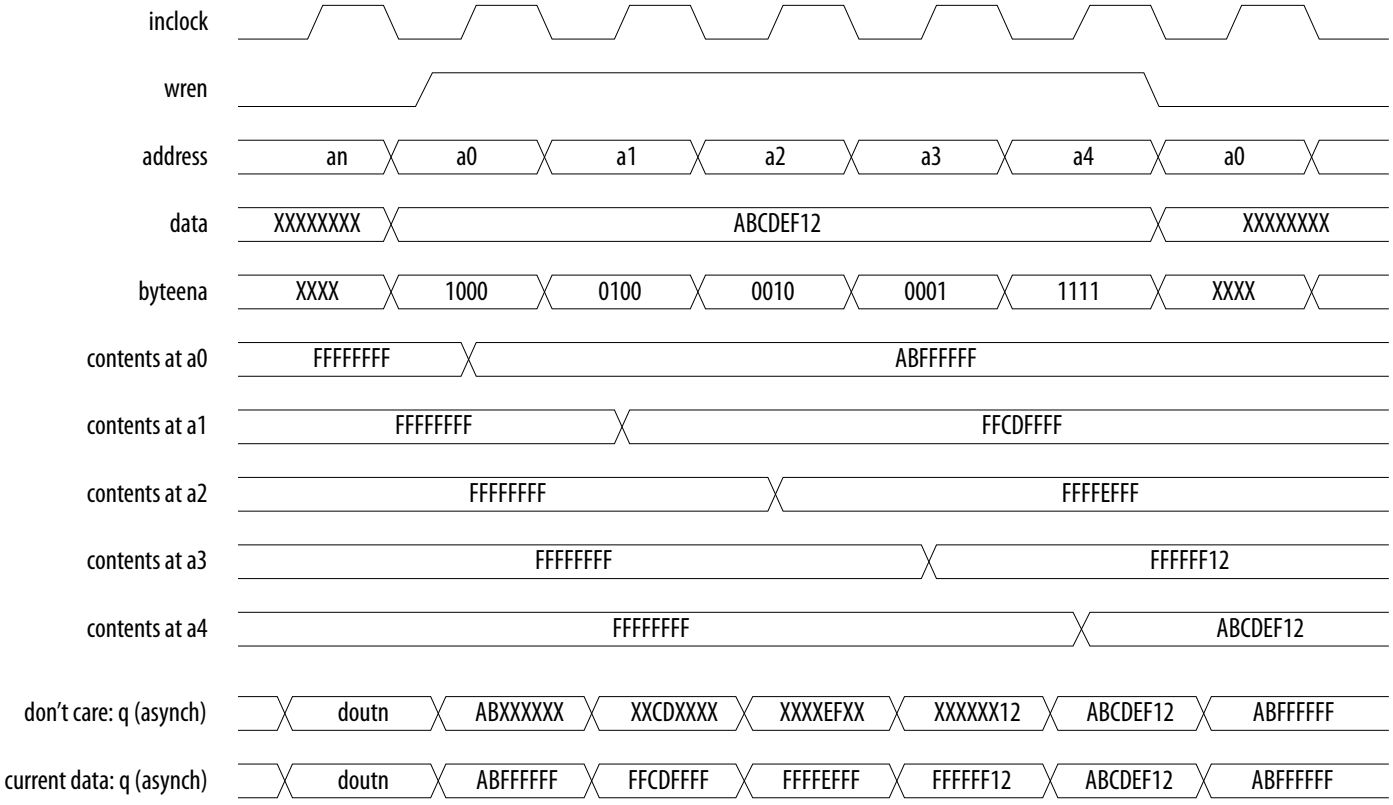
Data Byte Output

In M20K blocks or MLABs, when you de-assert a byte-enable bit during a write cycle, the corresponding data byte output appears as either a “don't care” value or the current data at that location. You can control the output value for the masked byte in the M20K blocks or MLABs by using the Quartus II software.

RAM Blocks Operations

Figure 2-6: Byte Enable Functional Waveform

This figure shows how the `wren` and `byteena` signals control the operations of the RAM blocks.



Memory Blocks Packed Mode Support

The M20K memory blocks support packed mode.

The packed mode feature packs two independent single-port RAM blocks into one memory block. The Quartus II software automatically implements packed mode where appropriate by placing the physical RAM block in true dual-port mode and using the MSB of the address to distinguish between the two logical RAM blocks. The size of each independent single-port RAM must not exceed half of the target block size.

Memory Blocks Address Clock Enable Support

The embedded memory blocks support address clock enable, which holds the previous address value for as long as the signal is enabled (`addressstall = 1`). When the memory blocks are configured in dual-port mode, each port has its own independent address clock enable. The default value for the address clock enable signal is low (disabled).

Figure 2-7: Address Clock Enable

This figure shows an address clock enable block diagram. The address clock enable is referred to by the port name `addressstall`.

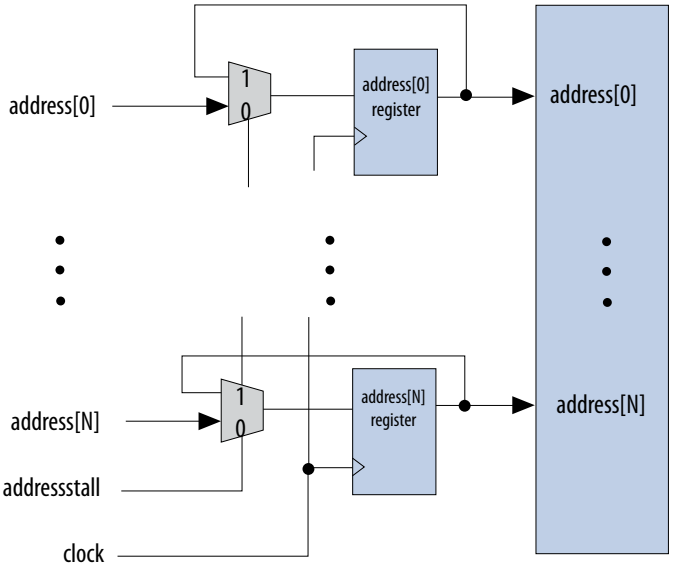


Figure 2-8: Address Clock Enable During Read Cycle Waveform

This figure shows the address clock enable waveform during the read cycle.

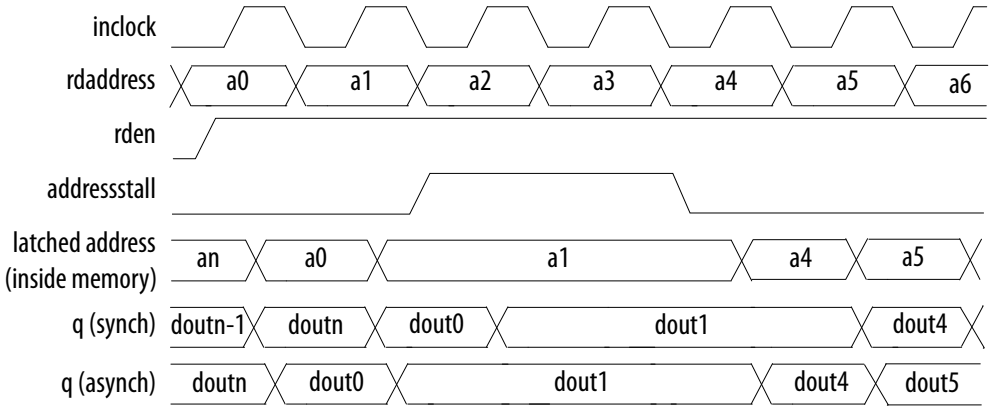
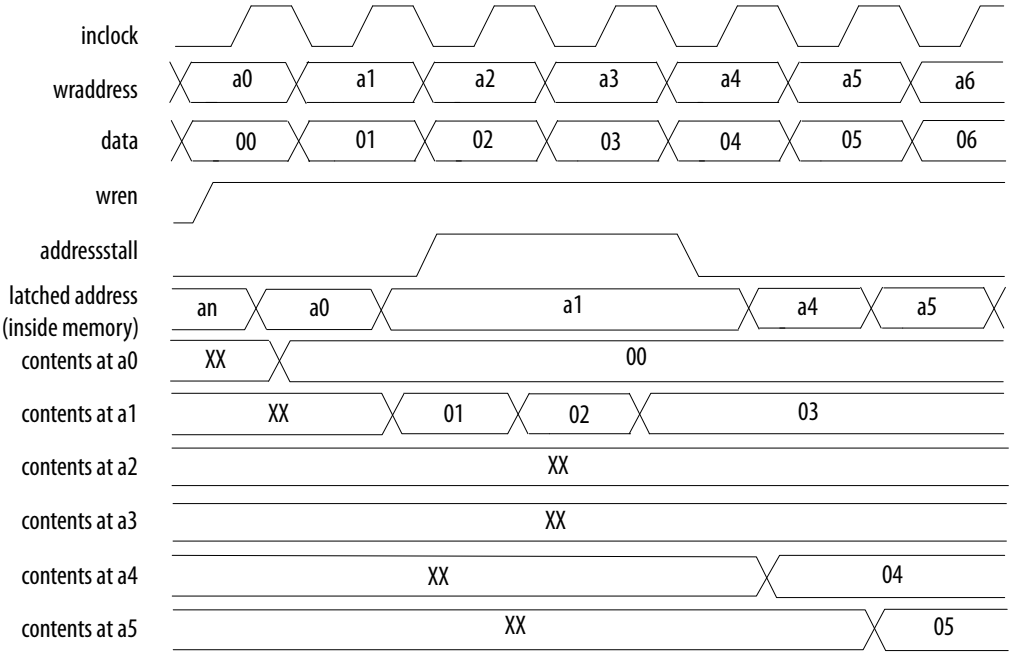


Figure 2-9: Address Clock Enable During the Write Cycle Waveform

This figure shows the address clock enable waveform during the write cycle.

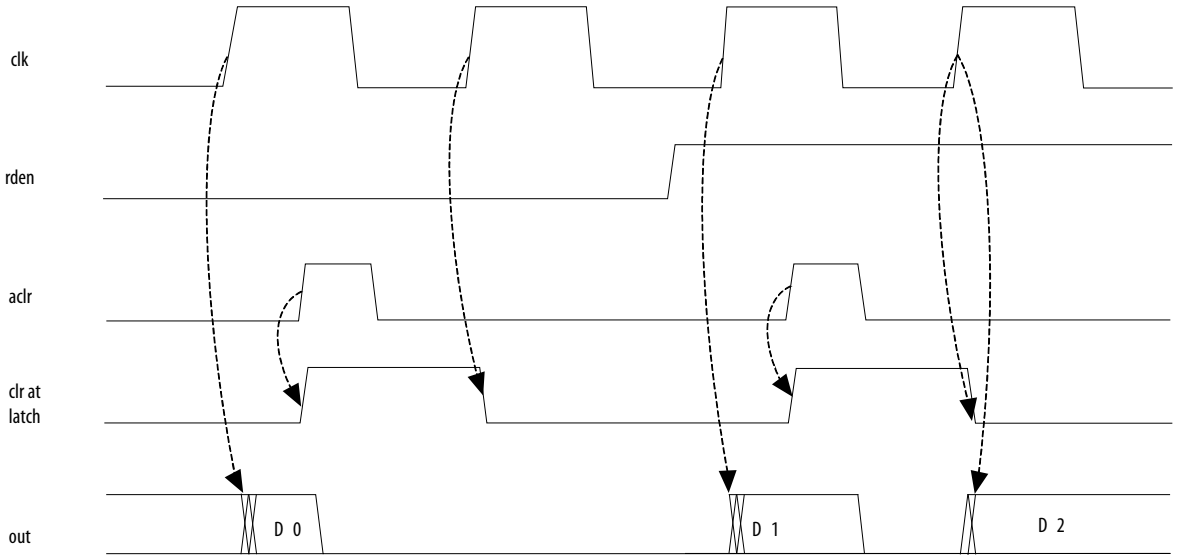


Memory Blocks Asynchronous Clear

The M20K memory blocks support asynchronous clear on output latches and output registers. If your RAM does not use output registers, clear the RAM outputs using the output latch asynchronous clear.

The clear is an asynchronous signal and it is generated at any time. The internal logic extends the clear pulse until the next rising edge of the output clock. When the clear is asserted, the outputs are cleared and stay cleared until the next read cycle.

Figure 2-10: Output Latch Clear in Stratix V Devices



Memory Blocks Error Correction Code Support

ECC allows you to detect and correct data errors at the output of the memory. ECC can perform single-error correction, double-adjacent-error correction, and triple-adjacent-error detection in a 32-bit word. However, ECC cannot detect four or more errors.

The M20K blocks have built-in support for ECC when in x32-wide simple dual-port mode:

- The M20K runs slower than non-ECC simple-dual port mode when ECC is engaged. However, you can enable optional ECC pipeline registers before the output decoder to achieve the same performance as non-ECC simple-dual port mode at the expense of one cycle of latency.
- The M20K ECC status is communicated with two ECC status flag signals—e (error) and ue (uncorrectable error). The status flags are part of the regular output from the memory block. When ECC is engaged, you cannot access two of the parity bits because the ECC status flag replaces them.

Error Correction Code Truth Table

Table 2-14: ECC Status Flags Truth Table

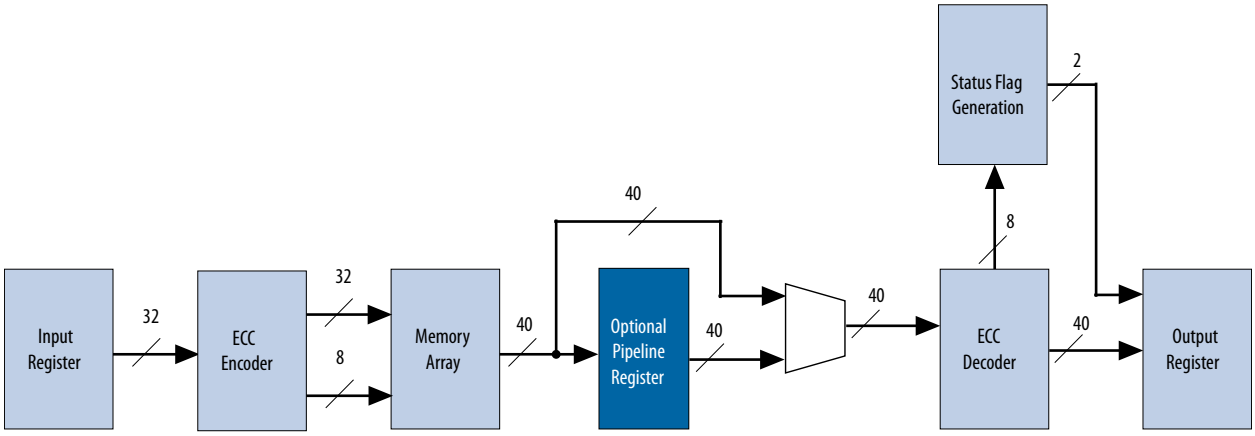
e (error) `eccstatus[1]`	ue (uncorrectable error) `eccstatus[0]`	Status
0	0	No error.
0	1	Illegal.
1	0	A correctable error occurred and the error has been corrected at the outputs; however, the memory array has not been updated.

e (error) eccstatus[1]	ue (uncorrectable error) eccstatus[0]	Status
1	1	An uncorrectable error occurred and uncorrectable data appears at the outputs.

If you engage ECC:

- You cannot use the byte enable feature.
- Read-during-write old data mode is not supported.

Figure 2-11: ECC Block Diagram for M20K Memory



Document Revision History

Date	Version	Changes
January 2015	2015.01.23	• Reword Total RAM bits in Memory Features in Stratix V Devices table to Capacity per Block.
June 2014	2014.06.30	Removed the term "one-hot" fashion for byte enables operation. The term one-hot indicates that only one bit can be active at a time.

Date	Version	Changes
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Corrected the description about the "don't care" output mode for RAM in mixed-port read-during-write. • Reorganized the structure of the supported memory configurations topics (single-port and mixed-width dual-port) to improve clarity about maximum data widths supported for each configuration. • Added a description to the table listing the maximum embedded memory configurations to clarify that the information applies only to the single port or ROM mode. • Removed the topic about mixed-width configurations for MLABs and added a note to clarify that MLABs do not support mixed-width configuration.
December 2012	2012.12.28	• Reorganized content and updated template. • Updated memory capacity information for accuracy (kilobits instead of megabits). • Moved information about supported memory block configurations into its own table. • Removed some information that is available in the **Internal Memory (RAM and ROM) User Guide**.
June 2012	1.4	Updated Table 2–1 and Table 2–2.
November 2011	1.3	• Updated Table 2–1 and Table 2–2. • Updated “Mixed-Port Read-During-Write Mode” section.
May 2011	1.2	• Chapter moved to volume 2 for the 11.0 release. • Updated Table 2–1, Table 2–2, and Table 2–5. • Updated Figure 2–1 and Figure 2–8. • Updated “Read-During-Write Behavior” section. • Minor text edits.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

Variable Precision DSP Blocks in Stratix V Devices

3

2015.01.23

SV51004 Subscribe Send Feedback

This chapter describes how the variable-precision digital signal processing (DSP) blocks in Stratix V devices are optimized to support higher bit precision in high-performance DSP applications.

Related Information

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

Features

Each Stratix V variable precision DSP block spans one logic array block (LAB) row height.

The Stratix V variable precision DSP blocks offer the following features:

- High-performance, power-optimized, and fully registered multiplication operations
- 9-bit, 18-bit, 27-bit, and 36-bit word lengths
- 18 x 25 complex multiplications for FFTs
- Floating-point arithmetic formats
- Built-in addition, subtraction, and 64-bit accumulation unit to combine multiplication results
- Cascading 18-bit and 27-bit input bus to form the tap-delay line for filtering applications
- Cascading 64-bit output bus to propagate output results from one block to the next block without external logic support
- Hard pre-adder supported in 18-bit and 27-bit mode for symmetric filters
- Supports 18-bit and 27-bit with internal coefficient register bank for filter implementation
- 18-bit and 27-bit systolic finite impulse response (FIR) filters with distributed output adder

Related Information

Stratix V Device Overview
Provides more information about the number of multipliers in each Stratix V device.

Supported Operational Modes in Stratix V Devices

Table 3-1: Variable Precision DSP Blocks Operational Modes for Stratix V Devices

Variable Precision DSP Block Resources	Operational Mode	Supported Instance	Pre-adder Support	Coefficient Support	Input Cascade Support	Chainout Support
1 variable precision DSP block	Independent 9 x 9 multiplication	3	No	No	No	No
	Independent 16 x 16 multiplication	2	Yes	Yes	Yes	No
	Independent 18 x 18 partial multiplication (32-bit)	2	Yes	Yes	Yes	No
	Independent 18 x 18 multiplication	1	Yes	Yes	Yes	No
	Independent 27 x 27 multiplication	1	Yes	Yes	Yes	Yes
	Independent 36 x 18 multiplication	1	No	Yes	No	Yes
	Two 18 x 18 multiplier adder	1	Yes	Yes	Yes	Yes
	Two 16 x 16 multiplier adder	1	Yes	Yes	Yes	Yes
	Sum of 2 square	1	Yes(1)	No	No	Yes
	18 x 18 multiplication summed with 36-bit input	1	No	No	No	Yes

(1) The pre-adder feature for this mode is automatically enabled.

Variable Precision DSP Block Resources	Operational Mode	Supported Instance	Pre-adder Support	Coefficient Support	Input Cascade Support	Chainout Support
2 variable precision DSP blocks	Independent 18 x 18 multiplication	3	No	No	No	No
	Independent 36 x 36 multiplication	1	No	No	No	No
	Complex 18 x 18 multiplication	1	Yes	Yes	Yes	Yes
	Four 18 x 18 multiplier adder	1	Yes	Yes	Yes	No
	Two 27 x 27 multiplier adder	1	Yes	Yes	Yes	No
	Two 18 x 36 multiplier adder	1	No	Yes	No	No
3 variable precision DSP blocks	Complex 18 x 25 multiplication	1	Yes(1)	No	No	No
4 variable precision DSP blocks	Complex 27 x 27 multiplication	1	Yes	Yes	Yes	No

Resources

Table 3-2: Number of Multipliers in Stratix V Devices

The table lists the variable-precision DSP resources by bit precision for each Stratix V device.

Variant	Member Code	Variable-precision DSP Block	Independent Input and Output Multiplications Operator					18 x 18 Multiplier Adder Mode	18 x 18 Multiplier Summed with 36-bit Input
			9 x 9 Multiplier	16 x 16 Multiplier	18 x 18 Multiplier with 32-bit Resolution	27 x 27 Multiplier	36 x 18 Multiplier		
Stratix V GX	A3	256	768	512	512	256	256	512	256
	A4	256	768	512	512	256	256	512	256
	A5	256	768	512	512	256	256	512	256
	A7	256	768	512	512	256	256	512	256
	A9	352	1,056	704	704	352	352	704	352
	AB	352	1,056	704	704	352	352	704	352
	B5	399	1,197	798	798	399	399	798	399
	B6	399	1,197	798	798	399	399	798	399
	B9	352	1,056	704	704	352	352	704	352
	BB	352	1,056	704	704	352	352	704	352
Stratix V GT	C5	256	768	512	512	256	256	512	256
	C7	256	768	512	512	256	256	512	256
Stratix V GS	D3	600	1,800	1,200	1,200	600	600	1,200	600
	D4	1,044	3,132	2,088	2,088	1,044	1,044	2,088	1,044
	D5	1,590	4,770	3,180	3,180	1,590	1,590	3,180	1,590
	D6	1,775	5,325	3,550	3,550	1,775	1,775	3,550	1,775
	D8	1,963	5,889	3,926	3,926	1,963	1,963	3,926	1,963
Stratix V E	E9	352	1,056	704	704	352	352	704	352
	EB	352	1,056	704	704	352	352	704	352

Design Considerations

You should consider the following elements in your design:

- Operational modes
- Internal coefficient and pre-adder
- Accumulator
- Chainout adder

Operational Modes

The Quartus II software includes megafunctions that you can use to control the operation mode of the multipliers. After entering the parameter settings with the MegaWizard Plug-In Manager, the Quartus II software automatically configures the variable precision DSP block.

Altera provides two methods for implementing various modes of the Stratix V variable precision DSP block in a design—using the Quartus II DSP megafunction and HDL inferring.

The following Quartus II megafunctions are supported for the Stratix V variable precision DSP blocks implementation:

- LPM_MULT
- ALTERA_MULT_ADD
- ALTMULT_COMPLEX

Related Information

- **Introduction to Megafunction User Guide**
- **Integer Arithmetic Megafunctions User Guide**
- **Floating-Point Megafunctions User Guide**
- **Quartus II Software Help**

Internal Coefficient and Pre-Adder

Mode	Stratix V
18-bit	The coefficient feature must be enabled when the pre-adder feature is enabled.
27-bit	The coefficient feature and pre-adder feature can be used independently. With pre-adder enabled: • If the multiplicand input comes from dynamic input due to input limitations—the input data width is restricted to 22 bits. • If the multiplicand input comes from the internal coefficients—the data width of the multiplicand is 27 bits.

Note: When you enable the pre-adder feature, all input data must have the same clock setting.

Accumulator

The accumulator feature is applicable to the following modes:

- One sum of two 18 x 18 multipliers
- 27 x 27 independent multiplier
- 36 x 18 independent multiplier
- 18 x 18 multiplication summed with 36-bit input mode
- Sum of square mode

Chainout Adder

You can use the output chaining path to add results from other DSP blocks.

Block Architecture

The Stratix V variable precision DSP block consists of the following elements:

- Input register bank
- Pre-adder
- Internal coefficient
- Multipliers
- Accumulator and chainout adder
- Systolic registers
- Output register bank

Figure 3-1: Variable Precision DSP Block Architecture in 18 x 18 Mode for Stratix V Devices

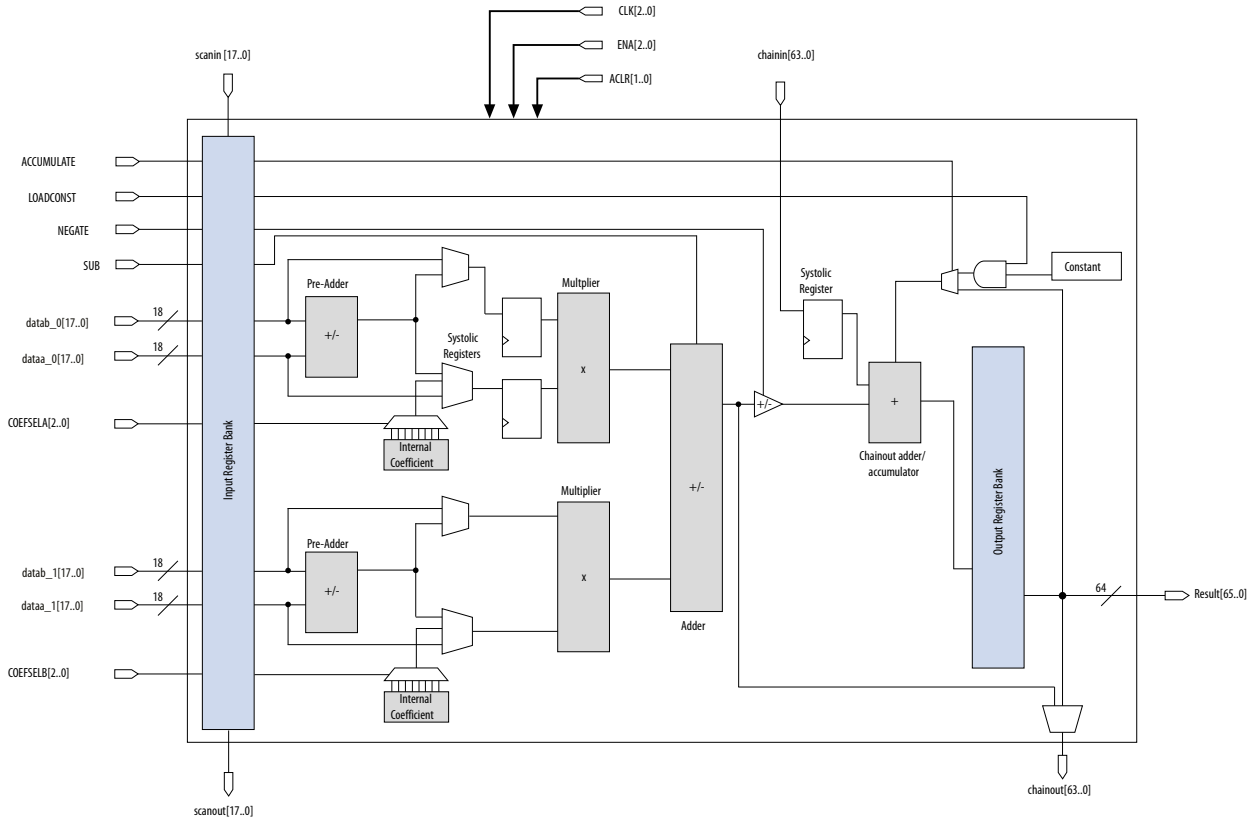
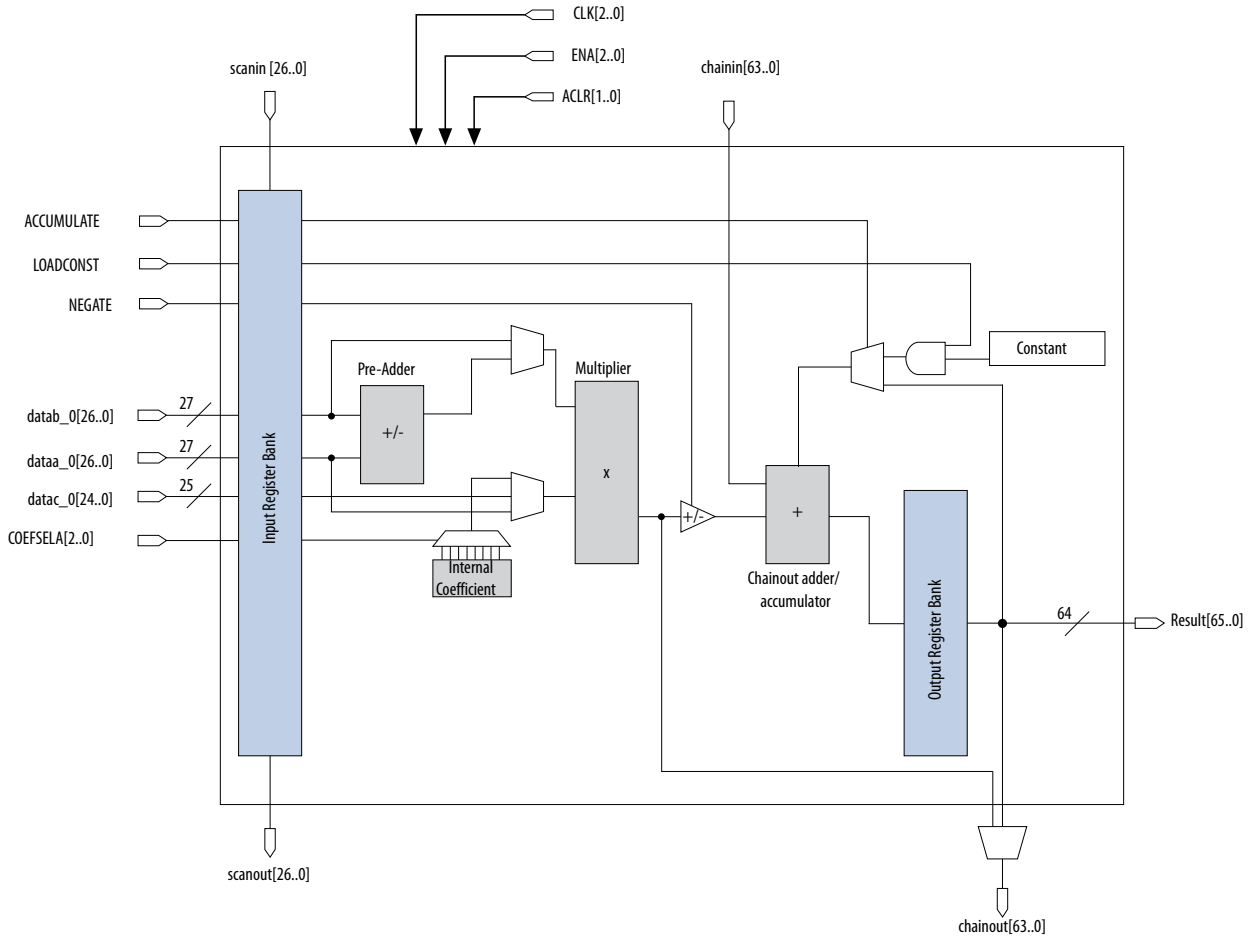


Figure 3-2: Variable Precision DSP Block Architecture in 27 x 27 Mode for Stratix V Devices



Input Register Bank

The input register bank consists of data, dynamic control signals, and two sets of delay registers.

All the registers in the DSP blocks are positive-edge triggered and cleared on power up. Each multiplier operand can feed an input register or a multiplier directly, bypassing the input registers.

The following variable precision DSP block signals control the input registers within the variable precision DSP block:

- `CLK[2..0]`
- `ENA[2..0]`
- `ACLR[0]`

In 18 x 18 mode, you can use the delay registers to balance the latency requirements when you use both the input cascade and chainout features.

One feature of the input register bank is to support a tap delay line; therefore, you can drive the top leg of the multiplier input (B) from general routing or from the cascade chain, as shown in the following figures. The Stratix V variable precision DSP block supports 18-bit and 27-bit input cascading.

Figure 3-3: Input Register of a Variable Precision DSP Block in 18 x 18 Mode for Stratix V Devices

The figures show the data registers only. Registers for the control signals are not shown.

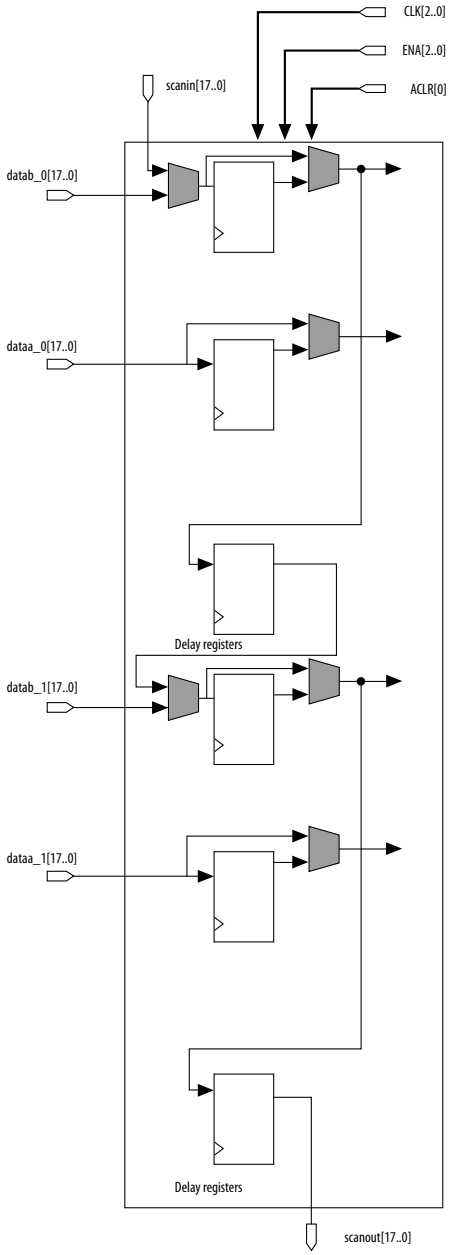
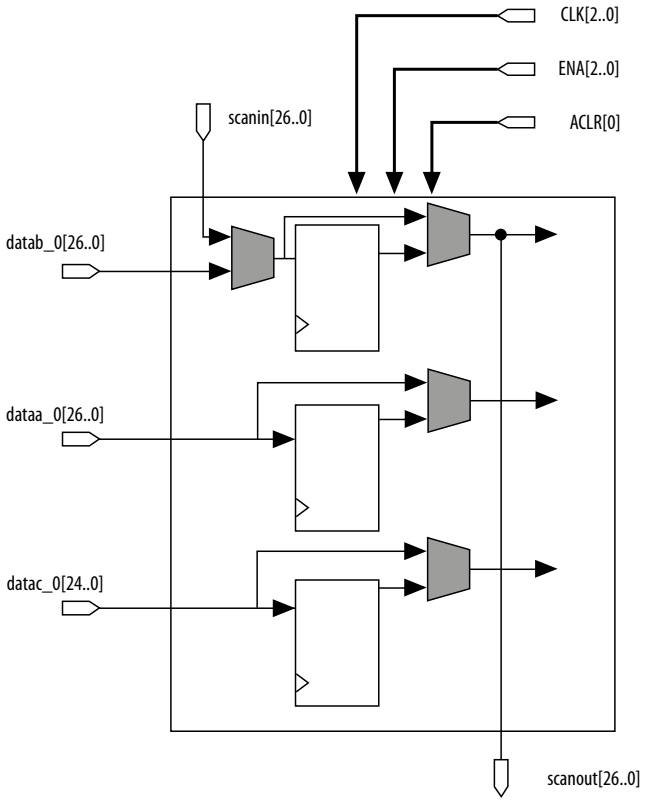


Figure 3-4: Input Register of a Variable Precision DSP Block in 27x 27 Mode for Stratix V Devices

The figures show the data registers only. Registers for the control signals are not shown.



Pre-Adder

Stratix V Devices

The pre-adder supports both addition and subtraction, which you must choose during compilation time.

Each variable precision DSP block has two 18-bit pre-adders. You can configure these pre-adders in the following configurations:

- Two independent 18-bit adders for 18-bit applications
- One 26-bit adder for 27-bit applications

Internal Coefficient

The Stratix V variable precision DSP block has the flexibility of selecting the multiplicand from either the dynamic input or the internal coefficient.

The internal coefficient can support up to eight constant coefficients for the multiplicands in 18-bit and 27-bit modes. When you enable the internal coefficient feature, `COEFSELA/COEFSELB` are used to control the selection of the coefficient multiplexer.

Multipliers

A single variable precision DSP block can perform many multiplications in parallel, depending on the data width of the multiplier.

There are two multipliers (upper multiplier and bottom multiplier) per variable precision DSP block. You can configure these two multipliers in several operational modes:

- One 27 x 27 multiplier
- Two 18 x 18 multipliers
- Three 9 x 9 multipliers

Related Information

Operational Mode Descriptions on page 3-11
Provides more information about the operational modes of the multipliers.

Accumulator and Chainout Adder

The Stratix V variable precision DSP block supports a 64-bit accumulator and a 64-bit adder.

For Stratix V devices, you can use the 64-bit adder as full adder.

The following signals can dynamically control the function of the accumulator:

- `NEGATE`
- `LOADCONST`
- `ACCUMULATE`

Table 3-3: Accumulator Functions and Dynamic Control Signals for 64-Bit Accumulator in Stratix V Devices

Function	Description	NEGATE	LOADCONST	ACCUMULATE
Zeroing	Disables the accumulator.	0	0	0
Preload	Loads an initial value to the accumulator. Only one bit of the 64-bit preload value can be "1". It can be used as rounding the DSP result to any position of the 64-bit result.	0	1	0
Accumulation	Adds the current result to the previous accumulate result.	0	0	1
Decimation	This function takes the current result, converts it into two's complement, and adds it to the previous result.	1	0	1

Systolic Registers

There are two systolic registers per variable precision DSP block. If the variable precision DSP block is not configured in systolic FIR mode, both systolic registers are bypassed.

The first systolic register has two 18-bit registers that are used to register the upper multiplier's two 18-bit inputs. You must clock these registers with the same clock source as the output register bank.

The second set of systolic registers are used to delay the chainout output to the next variable precision DSP block.

Output Register Bank

The positive edge of the clock signal triggers the 64-bit bypassable output register bank and is cleared after power up.

The following variable precision DSP block signals control the output register per variable precision DSP block:

- `CLK[2..0]`
- `ENA[2..0]`
- `ACLR[1]`

Operational Mode Descriptions

This section describes how you can configure an Stratix V variable precision DSP block to efficiently support the following operational modes:

- Independent Multiplier Mode
- Independent Complex Multiplier Mode
- Multiplier Adder Sum Mode
- Sum of Square Mode
- 18 x 18 Multiplication Summed with 36-Bit Input Mode
- Systolic FIR Mode

Independent Multiplier Mode

In independent input and output multiplier mode, the variable precision DSP blocks perform individual multiplication operations for general purpose multipliers.

You can configure each variable precision DSP block multiplier for 9-, 16-, 18-, 27-Bit, or 36 x 18 multiplication.

For some operational modes, the unused inputs require zero padding.

Table 3-4: Variable Precision DSP Block Independent Multiplier Mode Configurations

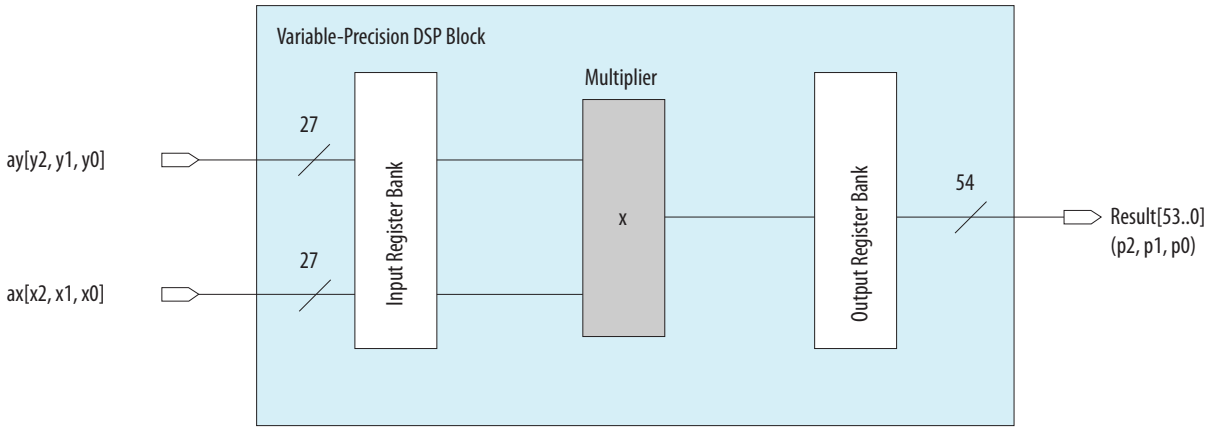
Configuration	Multipliers per block
9 x 9	3
16 x 16	2
18 x 18 (partial)	2

Configuration	Multipliers per block
18 x 18	1
27 x 27	1
36 x 18	1

9 x 9 Independent Multiplier

Figure 3-5: Three 9 x 9 Independent Multiplier Mode per Variable Precision DSP Block for Stratix V Devices

Three pairs of data are packed into the ax and ay ports; `result` contains three 18-bit products.



18 x 18 Independent Multiplier

Figure 3-6: One 18 x 18 Independent Multiplier Mode with One Variable Precision DSP Block for Stratix V Devices

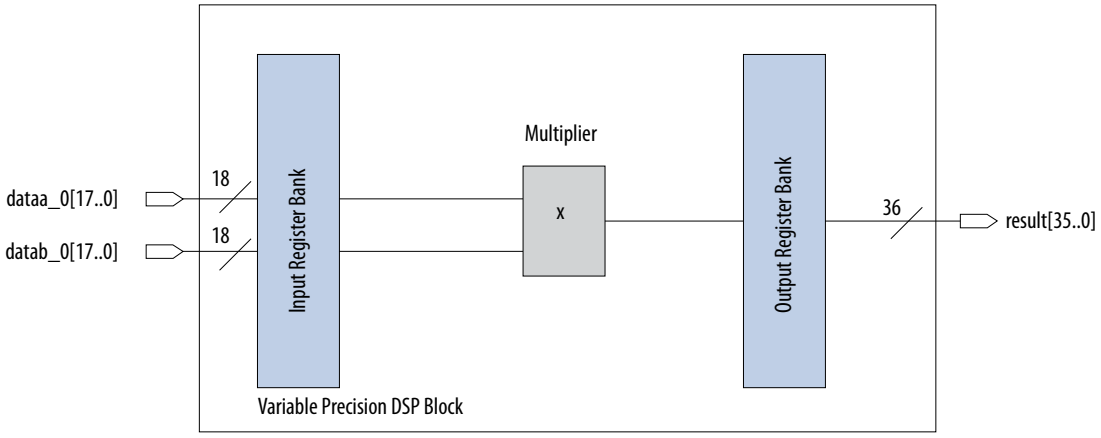
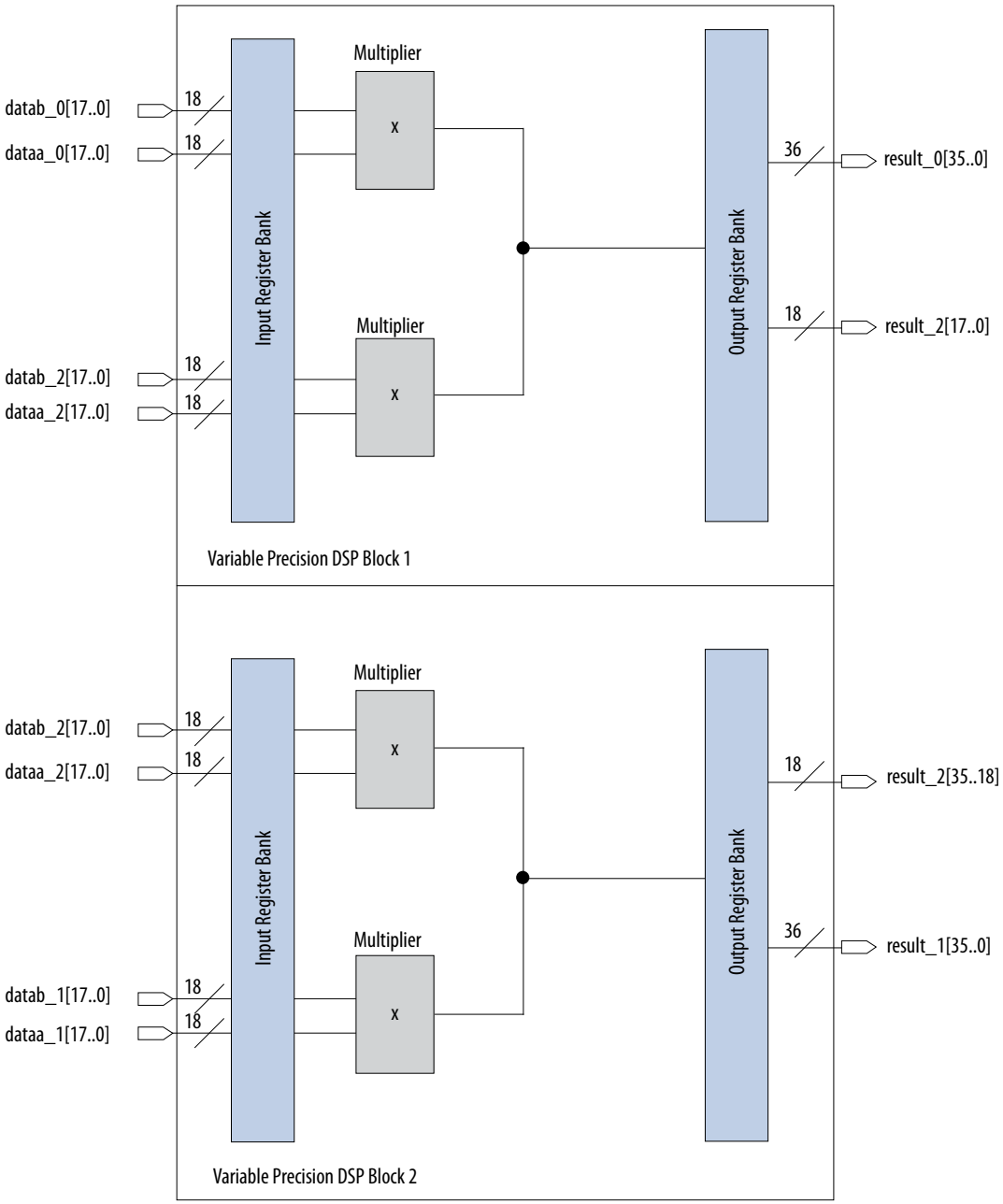


Figure 3-7: Three 18 x 18 Independent Multiplier Mode with Two Variable Precision DSP Blocks for Stratix V Devices

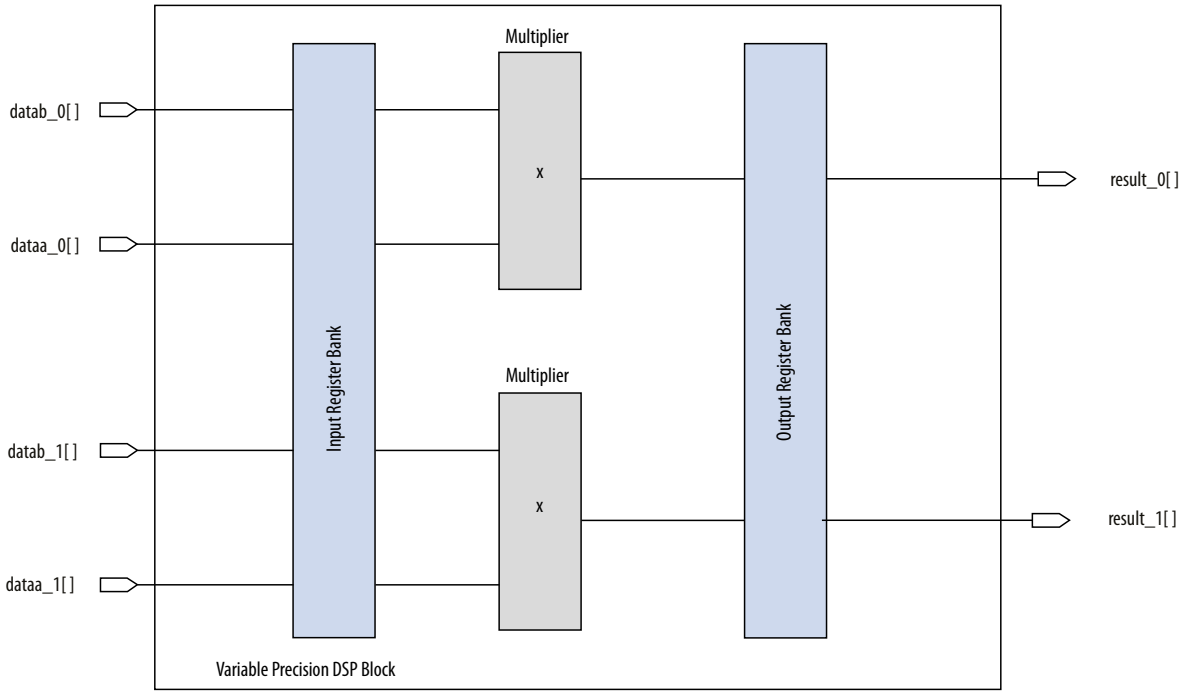


16 x 16 Independent Multiplier or 18 x 18 Independent Partial Multiplier

Figure 3-8: Two 16 x 16 Independent Multiplier Mode or Two 18 x 18 Independent Partial Multiplier Mode for Stratix V Devices

In this figure, the inputs for 16-bit independent multiplier mode are `data[15..0]`. The unused input bits require padding with zero.

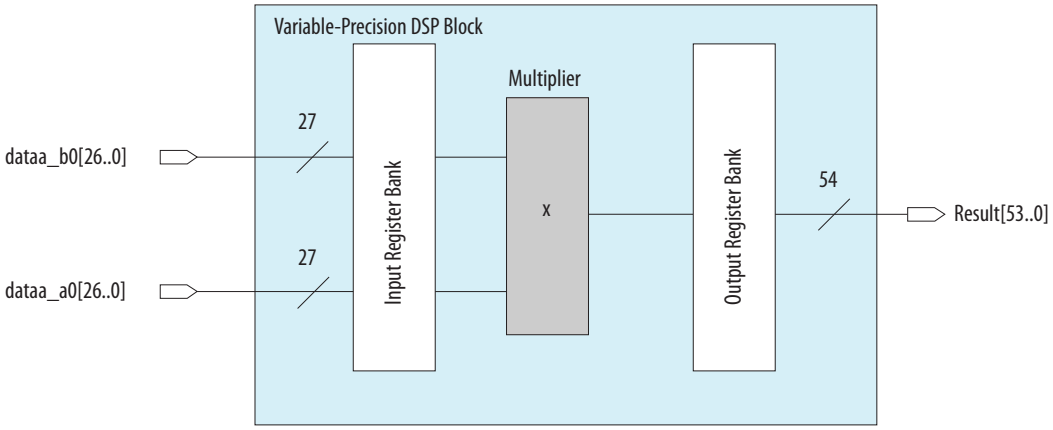
For two independent 18 x 18 partial multiplier mode, only 32-bit LSB result for each multiplication operation is routed to the output.



27 x 27 Independent Multiplier

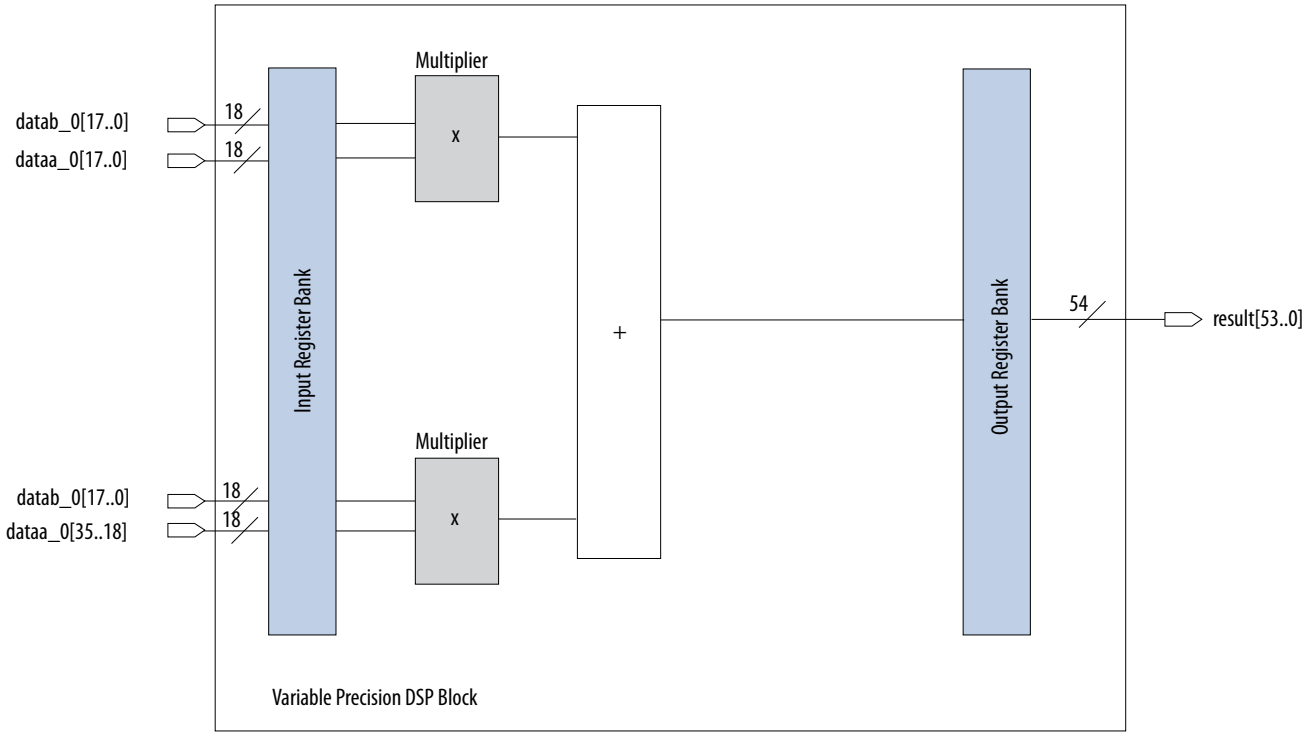
Figure 3-9: One 27 x 27 Independent Multiplier Mode per Variable Precision DSP Block for Stratix V Devices

In this mode, the `result` can be up to 64 bits when combined with a chainout adder or accumulator.



36 x 18 Independent Multiplier

Figure 3-10: One 36 x 18 Independent Multiplier Mode for Stratix V Devices

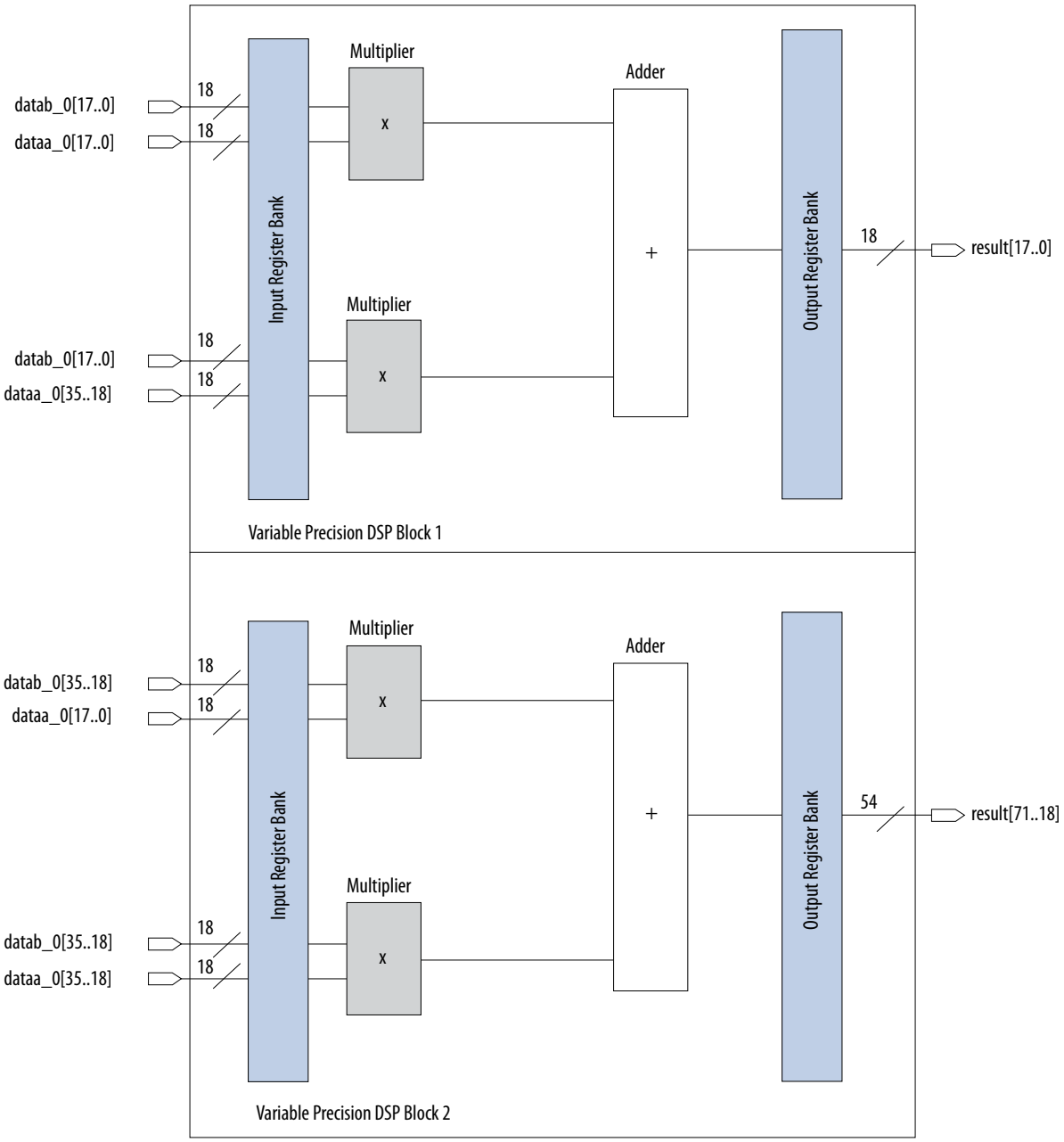


36-Bit Independent Multiplier

You can efficiently construct an individual 36-bit multiplier with two adjacent variable precision DSP blocks. The 36 x 36 multiplication consists of four 18 x 18 multipliers, as shown in **Figure 3-11**.

The 36-bit multiplier is useful for applications requiring more than 18-bit precision; for example, for the mantissa multiplication portion of very high precision fixed-point arithmetic applications.

Figure 3-11: 36-Bit Independent Multiplier Mode with Two Variable Precision DSP Blocks for Stratix V Devices



Independent Complex Multiplier Mode

The Stratix V variable precision DSP block provides the means for a complex multiplication.

Figure 3-12: Sample of Complex Multiplication Equation

$$(a + jb) \times (c + jd) = [(a \times c) - (b \times d)] + j[(a \times d) + (b \times c)]$$

The Stratix V variable precision DSP block can support the following:

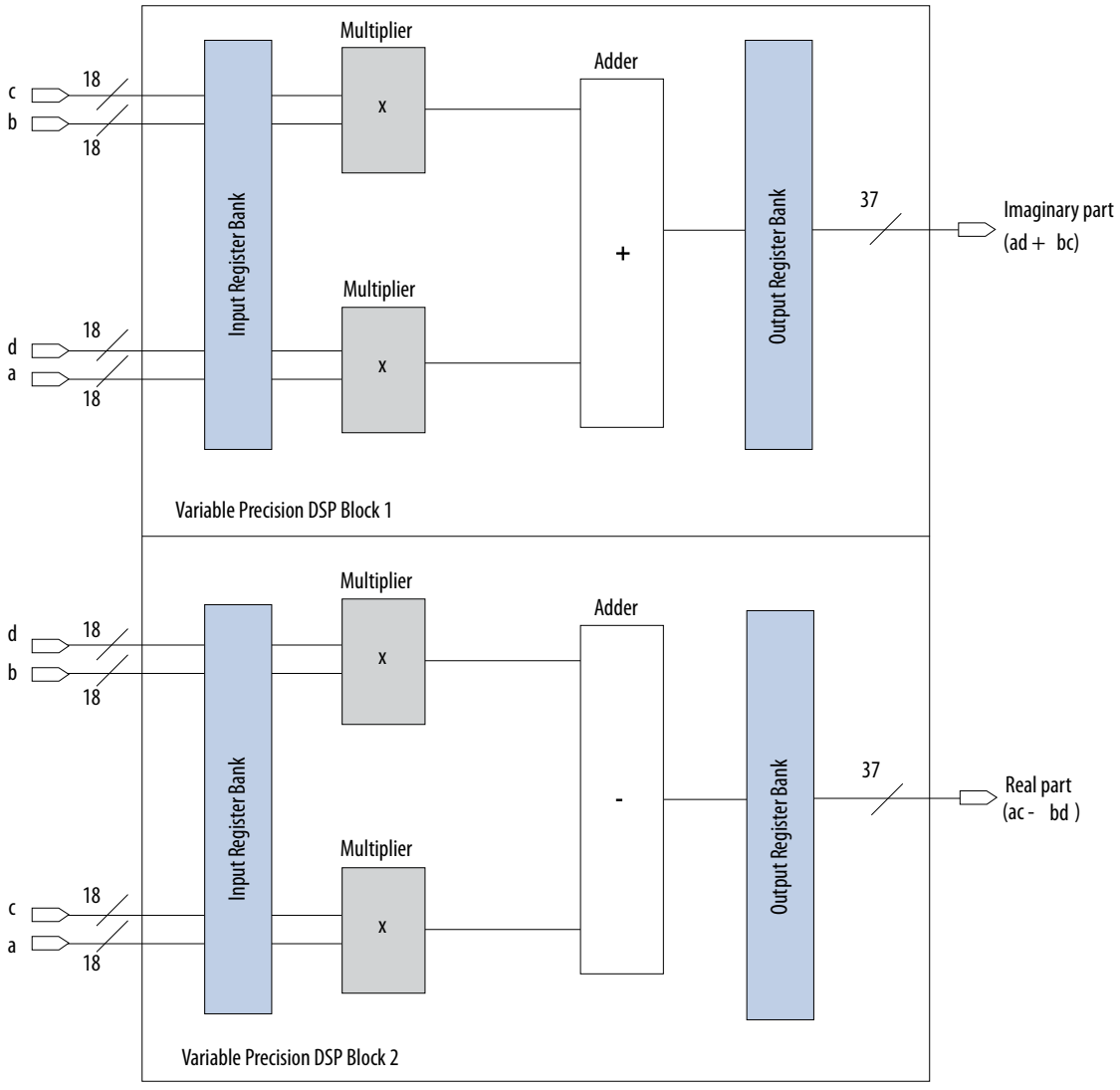
- one 18 x 18 complex multiplier
- one 18 x 25 complex multiplier
- one 27 x 27 complex multiplier

18 x 18 Complex Multiplier

For 18 x 18 complex multiplication mode, you require two variable precision DSP blocks to perform this multiplication.

You can implement the imaginary part [(a × d) + (b × c)] in the first variable precision DSP block, and you can implement the real part [(a × c) – (b × d)] in the second variable precision DSP block.

Figure 3-13: 18 x 18 Complex Multiplier with Two Variable Precision DSP Blocks for Stratix V Devices



18 x 25 Complex Multiplier

Stratix V devices support an individual 18 x 25 complex multiplication mode.

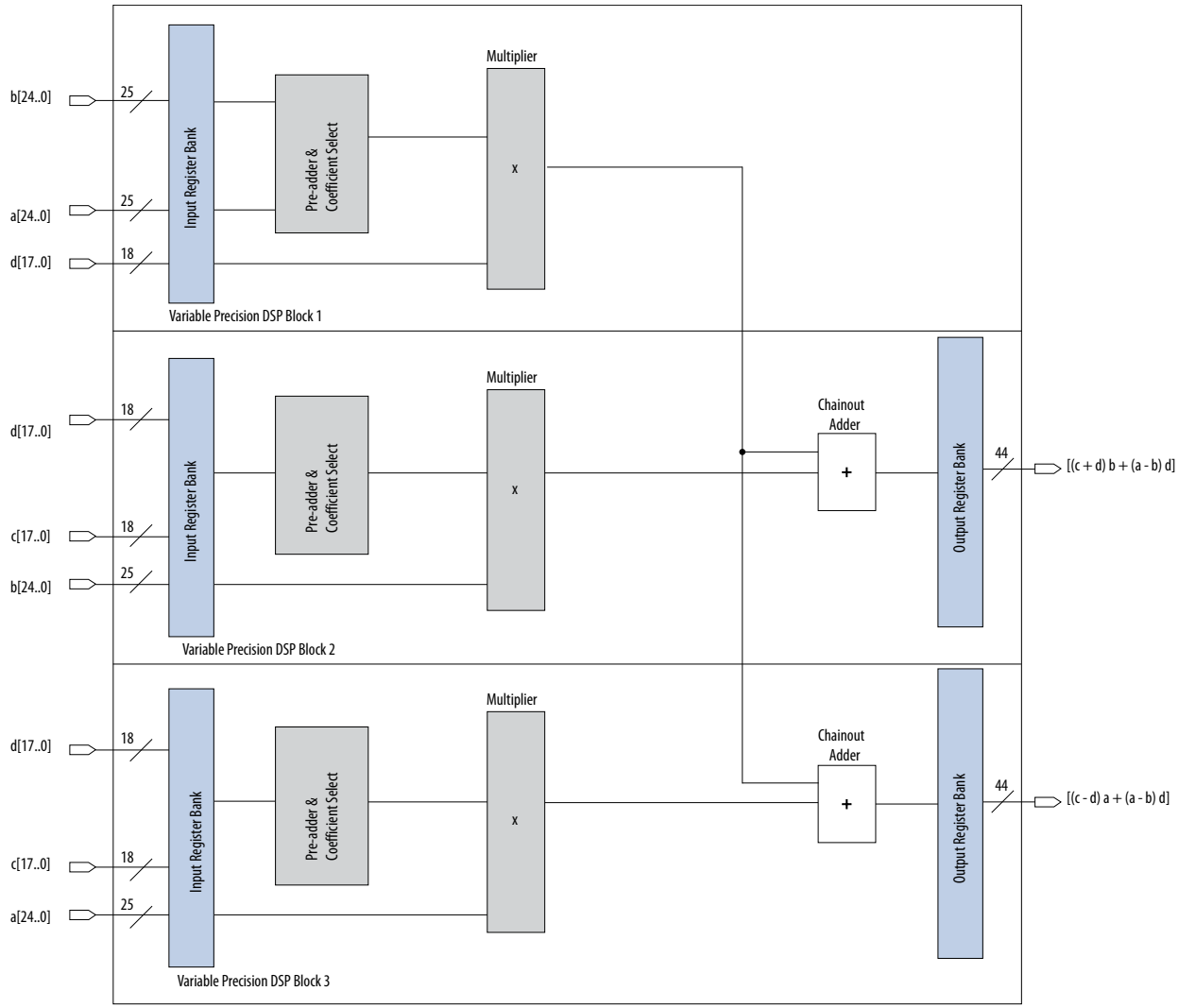
A 27 x 27 multiplier allows you to implement an individual 18 x 25 complex multiplication mode with three variable precision DSP blocks only. The pre-adder feature is automatically enabled for you to implement an individual 18 x 25 complex multiplication mode efficiently.

You can implement an 18 x 25 complex multiplication with three variable precision DSP blocks, as shown in **Figure 3-14**

Figure 3-14: 18 x 25 Complex Multiplication Equation

$$(a + jb) \times (c + jd) = (c - d) \times a + (a - b) \times d + j[(c + d) \times b + (a - b) \times d]$$

Figure 3-15: 18 x 25 Complex Multiplier with Three Variable Precision DSP Blocks for Stratix V Devices



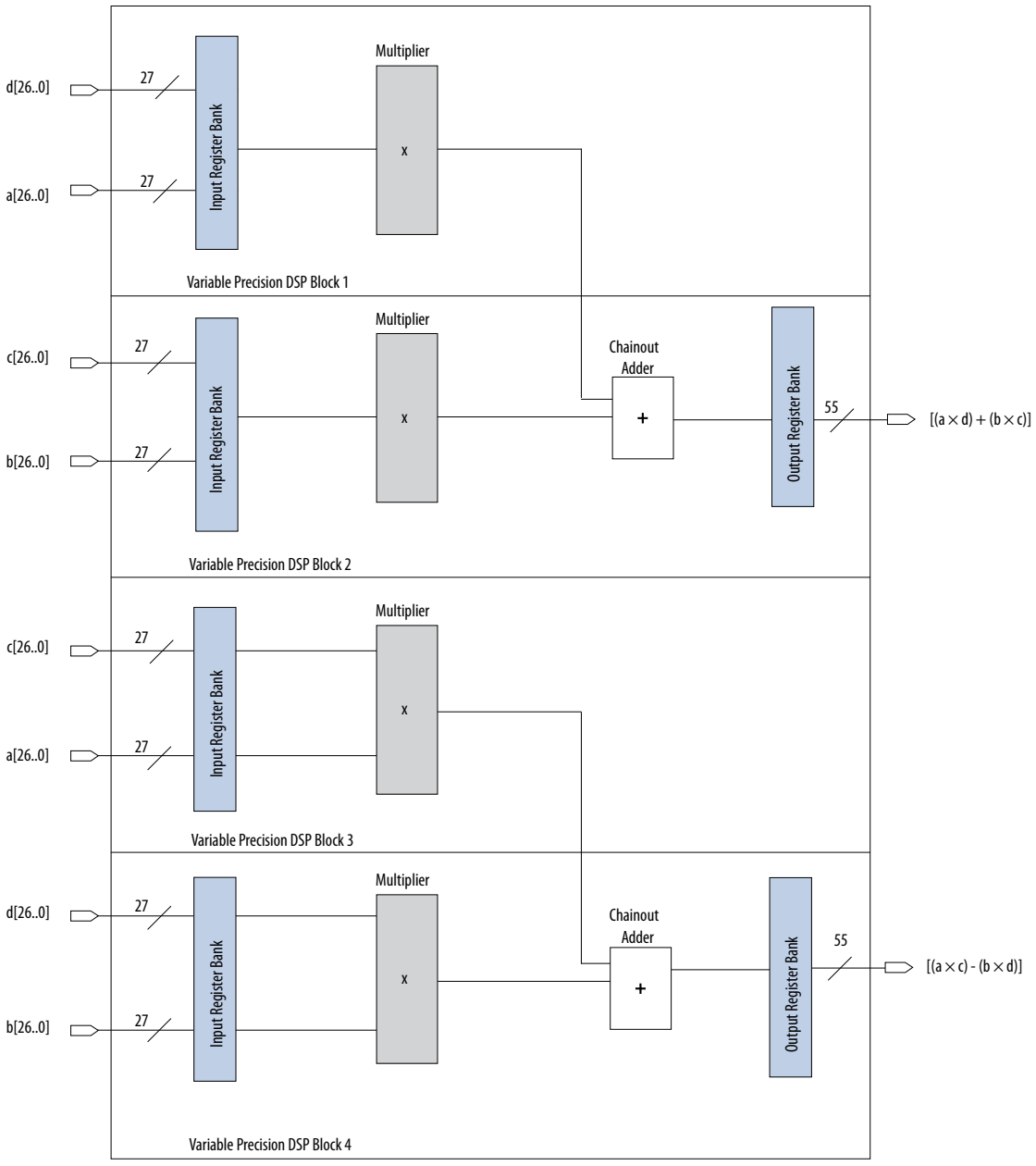
27 x 27 Complex Multiplier

Stratix V devices support an individual 27 x 27 complex multiplication mode. You require four variable precision DSP blocks to implement an individual 27 x 27 complex multiplication mode.

You can implement the imaginary part [(a x d) + (b x c)] in the first and second variable precision DSP blocks, and you can implement the real part [(a x c) - (b x d)] in the third and fourth variable precision DSP blocks.

You can achieve the difference of two 27 x 27 multiplications by enabling the NEGATE control signal in the fourth variable precision DSP block.

Figure 3-16: 27 x 27 Complex Multiplier with Four Variable Precision Blocks for Stratix V Devices



Multiplier Adder Sum Mode

Table 3-5: Variable Precision DSP Block Multiplier Adder Sum Mode Configurations for Stratix V Devices

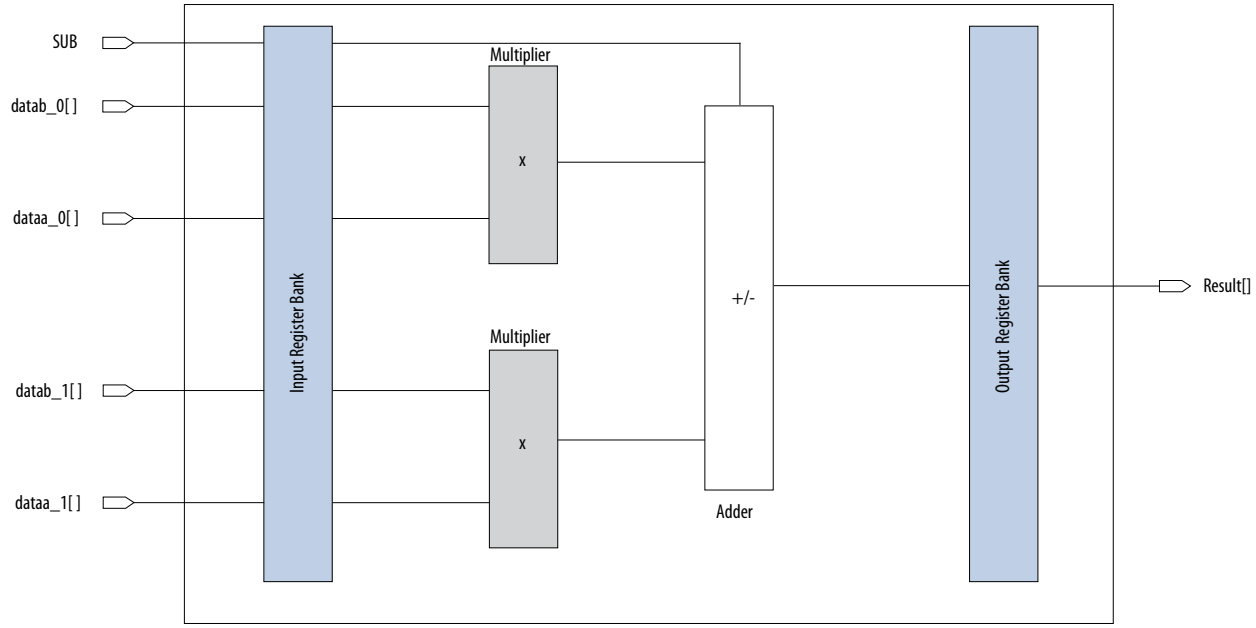
Mode	Configuration	Number of DSP Blocks Required
Two-multiplier Adder Sum	16 x 16	1
	18 x 18	1
	27 x 27	2
	18 x 36	2
Four-multiplier Adder Sum	18 x 18	2

One Sum of Two 18 x 18 Multipliers or Two 16 x 16 Multipliers

Figure 3-17: One Sum of Two 18 x 18 Multipliers or Two 16 x 16 Multipliers with One Variable Precision DSP Block for Stratix V Devices

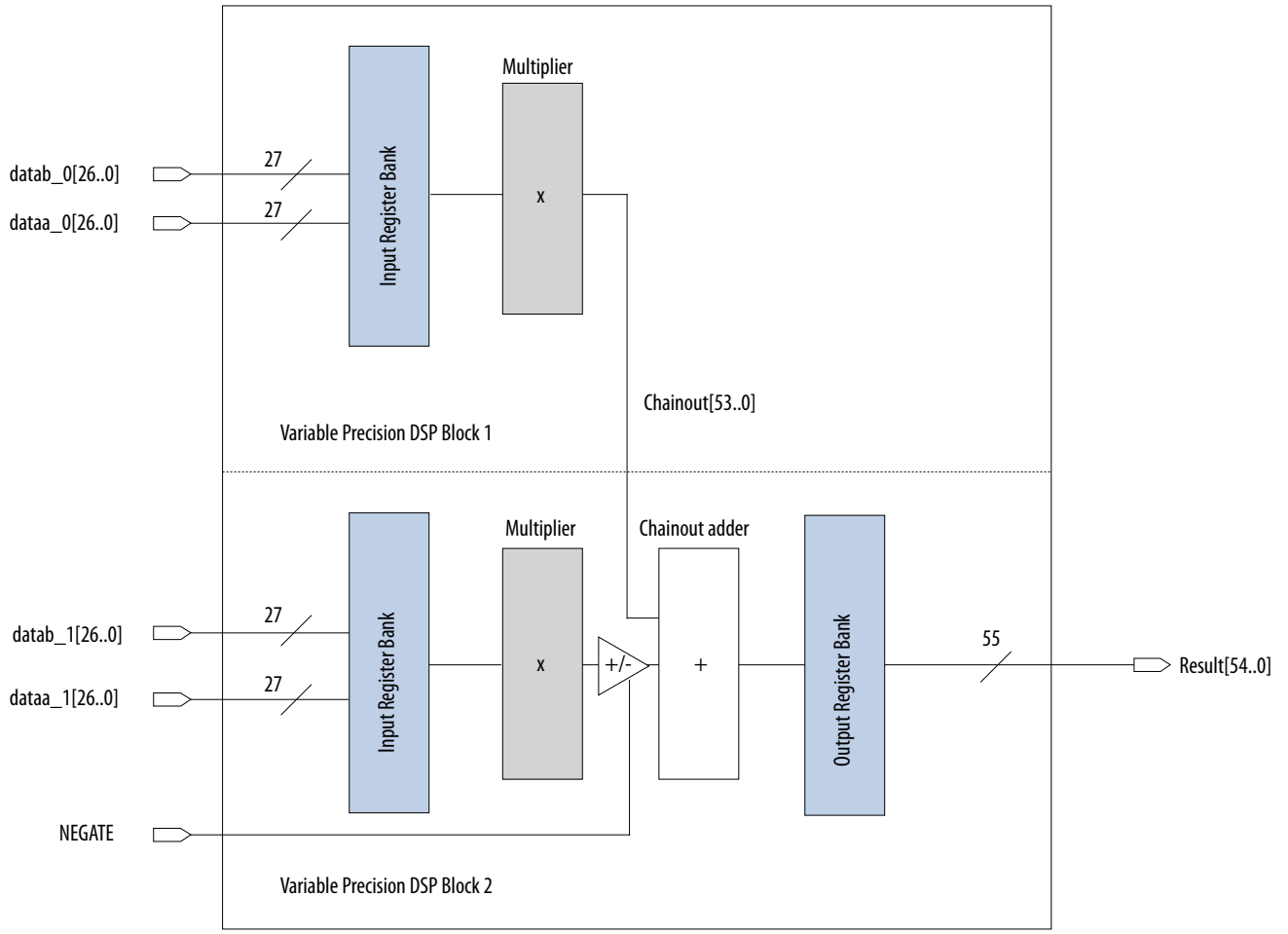
In this figure, for 18-bit multiplier adder sum mode, the input data width is 18 bits and the output data width is 37 bits.

For 16-bit multiplier adder sum mode, the input data width is 16 bits and the unused input bit requires padding with zeroes. The output data width is 33 bits.



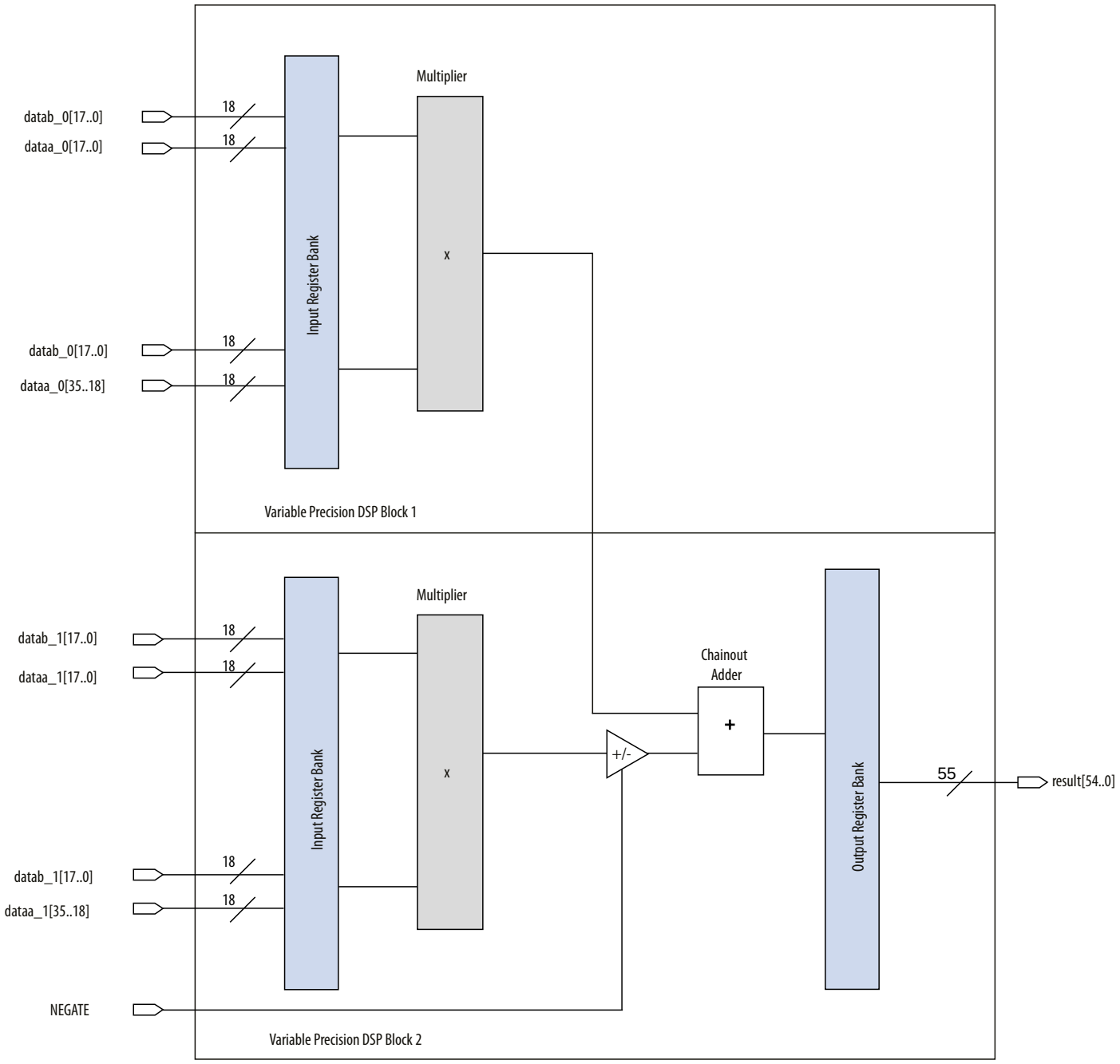
One Sum of Two 27 x 27 Multipliers

Figure 3-18: One Sum of Two 27 x 27 Multipliers with Two Variable Precision DSP Blocks for Stratix V Devices



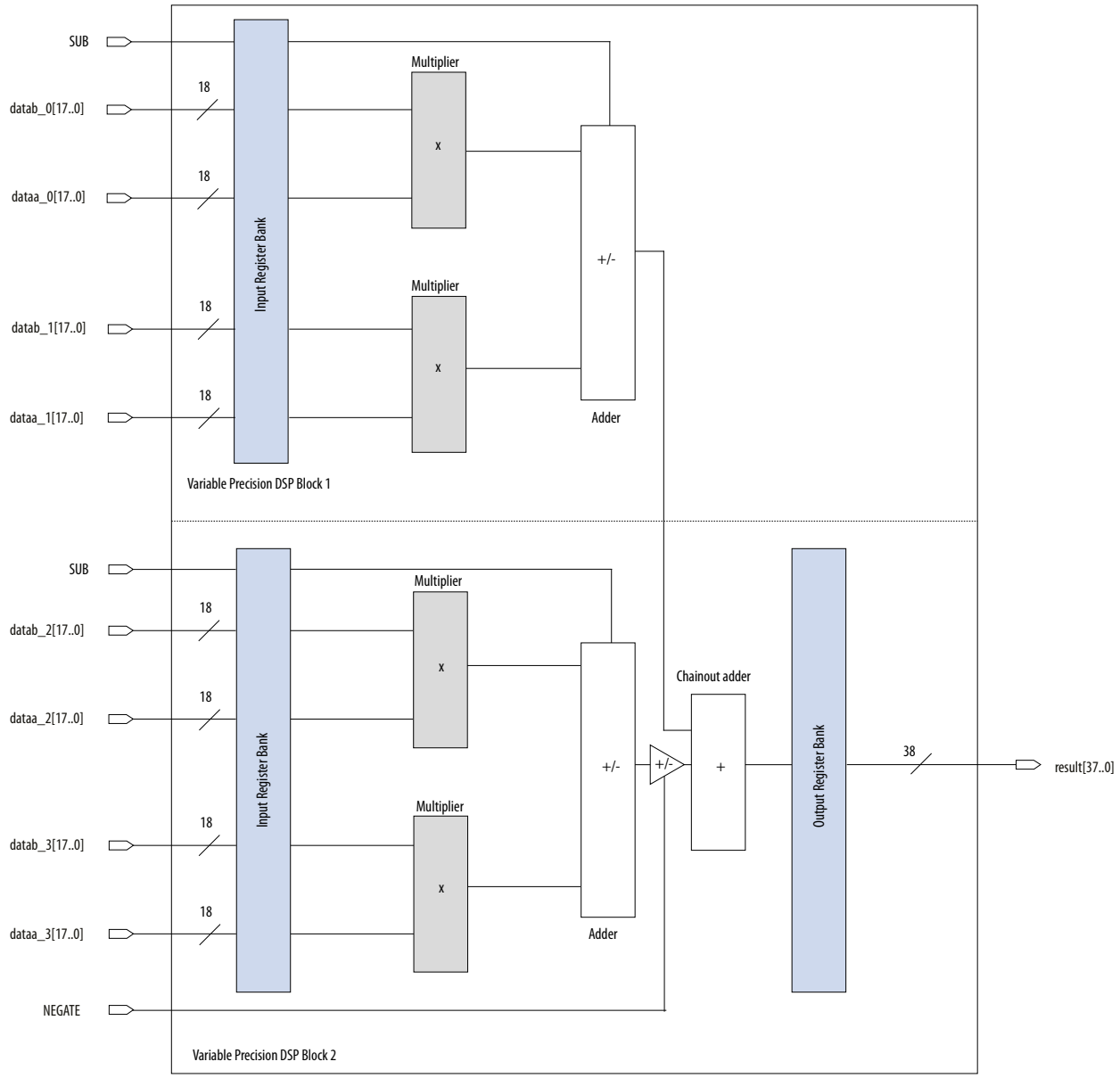
One Sum of Two 36 x 18 Multipliers

Figure 3-19: One Sum of Two 36 x 18 Multipliers with Two Variable Precision DSP Blocks for Stratix V Devices



One Sum of Four 18 x 18 Multipliers

Figure 3-20: One Sum of Four 18 x 18 Multipliers with Two Variable Precision DSP Blocks for Stratix V Devices



Sum of Square Mode

The Stratix V variable precision DSP block can implement one sum of square mode.

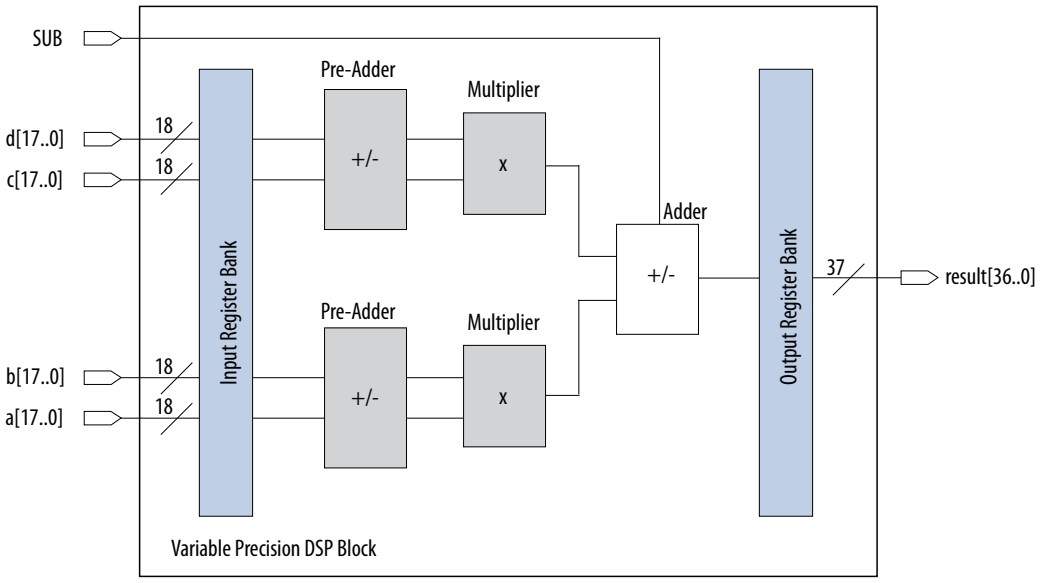
Figure 3-21: One Sum of Square Mode Equation

$$(a \pm b)^2 \times (c \pm d)^2$$

You can feed the four 18-bit inputs into the pre-adder block to convert b and d input as two's complement numbers to perform subtraction, if required.

You can feed each 18-bit pre-adder block output into both multiplicand and multiplier inputs of an 18 x 18 multiplier to generate a square result.

Figure 3-22: One Sum of Square Mode in a Variable Precision DSP Block for Stratix V Devices



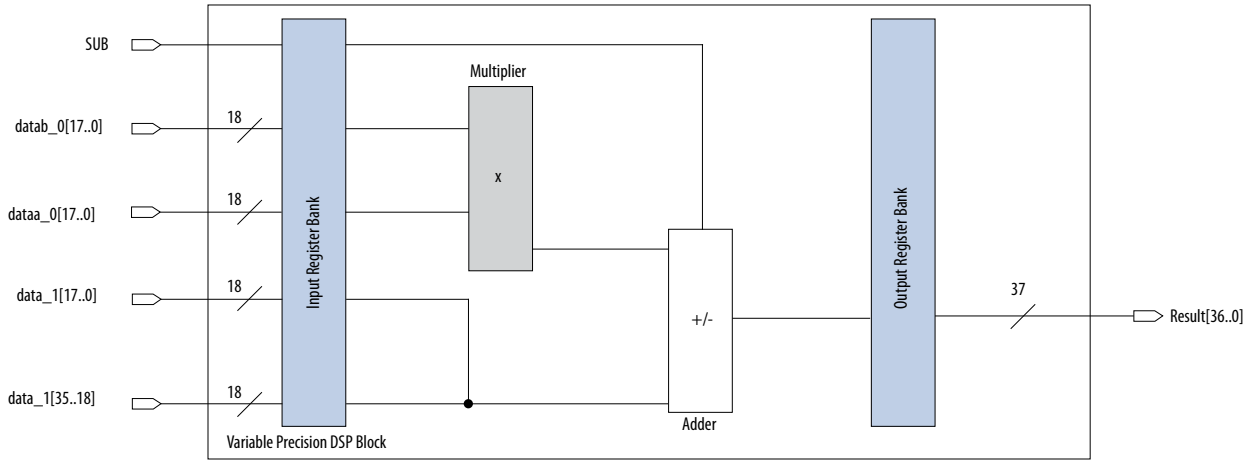
18 x 18 Multiplication Summed with 36-Bit Input Mode

Stratix V variable precision DSP blocks support one 18 x 18 multiplication summed to a 36-bit input.

Use the upper multiplier to provide the input for an 18 x 18 multiplication, while the bottom multiplier is bypassed.

The data1[17..0] and data1[35..18] signals are concatenated to produce a 36-bit input.

Figure 3-23: One 18 x 18 Multiplication Summed with 36-Bit Input Mode for Stratix V Devices



Systolic FIR Mode

Stratix V variable precision DSP blocks support the following systolic FIR structures:

- 18-bit
- 27-bit

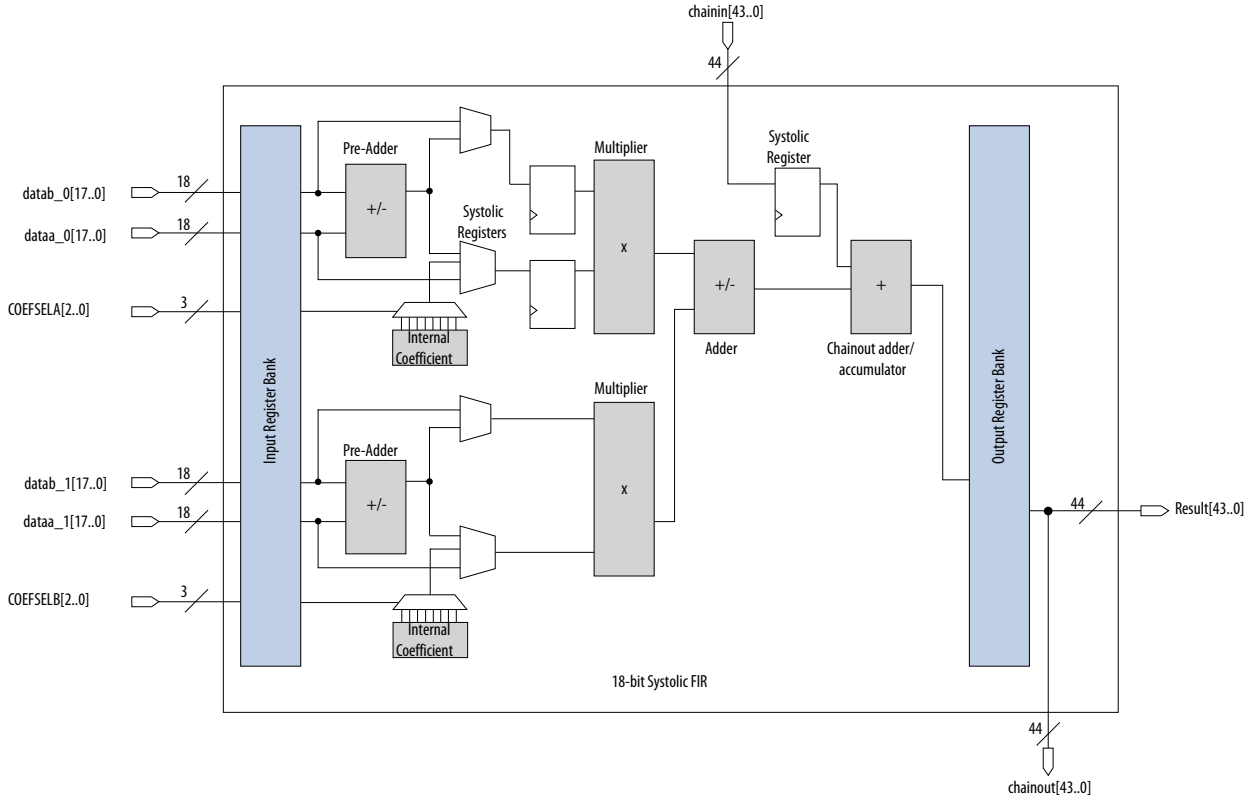
In systolic FIR mode, the input of the multiplier can come from three different sets of sources:

- Two dynamic inputs
- One dynamic input and one coefficient input
- One coefficient input and one pre-adder output

18-Bit Systolic FIR Mode

In 18-bit systolic FIR mode, the adders are configured as dual 44-bit adders, thereby giving 8 bits of overhead when using an 18-bit operation (36-bit products). This allows a total of 256 multiplier products.

Figure 3-24: 18-Bit Systolic FIR Mode with Two Dynamic Inputs for Stratix V Devices

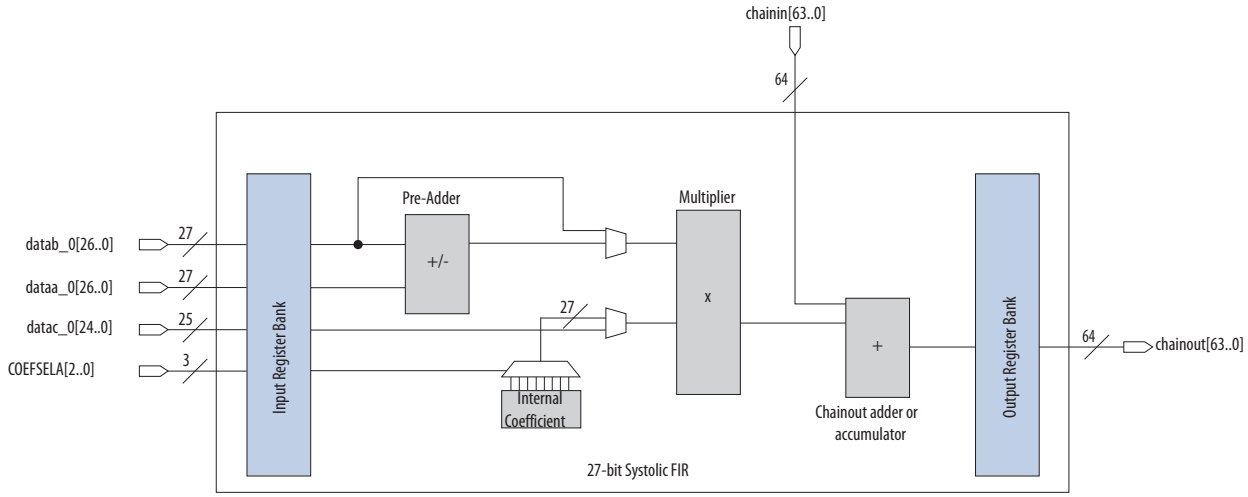


27-Bit Systolic FIR Mode

In 27-bit systolic FIR mode, the chainout adder or accumulator is configured for a 64-bit operation, providing 10 bits of overhead when using a 27-bit data (54-bit products). This allows a total of 1,024 multiplier products.

The 27-bit systolic FIR mode allows the implementation of one stage systolic filter per DSP block.

Figure 3-25: 27-Bit Systolic FIR Mode for Stratix V Devices



Variable Precision DSP Block Control Signals

The Stratix V variable precision DSP block has a total of 14 dynamic control signal inputs. The variable precision DSP block dynamic signals are user-configurable and can be set to toggle or not at run time.

The Stratix V variable precision DSP block supports 18-bit and 27-bit input cascading.

Table 3-6: Variable Precision DSP Block Dynamic Signals for Stratix V Devices

Signal Name	Function	Count
NEGATE	Control the operation of the decimation	1
LOADCONST	Preload an initial value to the accumulator	1
ACCUMULATE	Enable accumulation	1
SUB	This signal has two functions: • Controls add or subtract of the two 18 x 18 multiplier results • Controls dynamic switch between 36 x 36 mode and complex 18 x 18	1
COEFSELA COEFSELB	Controls the internal coefficient select multiplexer along with select signals provided through the MSB of each 18-bit data input	2
CLK0 CLK1 CLK2	Variable precision DSP-block-wide clock signals	3

Signal Name	Function	Count
ENA0 ENA1 ENA2	Variable precision DSP-block-wide clock enable signals	3
ACLR0 ACLR1	Variable precision DSP-block-wide asynchronous clear signals	2
Total Count per DSP Block		**14**

Document Revision History

Date	Version	Changes
July 2014	2014.07.22	Reinstated input register bank and systolic registers to the block architecture.
June 2014	2014.06.30	• Updated the supported megafunctions from ALTMULT_ADD and ALTMULT _ACCUM to ALTERA_MULT_ADD. • Updated modes applicable to the accumulator
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Moved all links to the Related Information section of respective topics for easy reference.
December 2012	2012.12.28	• Added "Design Considerations" • Updated Figure 3-1 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-6 changed Mult_L to Multiplier • Updated Figure 3-7 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-8 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-10 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-11 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-13 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-17 changed Mult_L and Mult_H to Multiplier • Updated Figure 3-19 changed 54 to 55 • Updated Figure 3-20 changed 19 to 18 and deleted Chainout [38..0] • Updated Figure 3-23 changed Mult_L to Multiplier • Updated Figure 3-24 changed Mult_L and Mult_H to Multiplier and added 44 • Updated Figure 3-25 added 64 • Reorganized content and updated template.

Date	Version	Changes
June 2012	1.4	• Added Figure 3–2. • Updated Figure 3–7, Figure 3–16, and Figure 3–18. • Updated Table 3–1. • Updated "Chainout Adder and Accumulator" and "18 x 25 Complex Multiplier" sections.
November 2011	1.3	• Added Figure 3–21. • Updated Figure 3–1, Figure 3–2, Figure 3–11, Figure 3–12, Figure 3–14, Figure 3–16, Figure 3–17, Figure 3–18, Figure 3–19, Figure 3–20, and Figure 3–21. • Updated Table 3–1 and Table 3–5. • Updated "Pre-Adder and Coefficient Select", "Systolic Register", "Systolic FIR Mode", and "Software Support" sections.
May 2011	1.2	• Updated chapter for Quartus II software 11.0 release. • Chapter moved to volume 2 for the 11.0 release. • Updated Table 3–1, Table 3–2, and Table 3–5. • Added Table 3–3. • Updated all figures in the chapter. • Added Figure 3–3. • Updated "Software Support" section.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

Clock Networks and PLLs in Stratix V Devices

4

2015.01.23

SV51005

Subscribe Send Feedback

This chapter describes the advanced features of hierarchical clock networks and phase-locked loops (PLLs) in Stratix V devices. The Quartus II software enables the PLLs and their features without external devices.

Related Information

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

Clock Networks

The Stratix V devices contain the following clock networks that are organized into a hierarchical structure:

- Global clock (GCLK) networks
- Regional clock (RCLK) networks
- Periphery clock (PCLK) networks

Clock Resources in Stratix V Devices

Table 4-1: Clock Resources in Stratix V Devices

Clock Resource	Device	Number of Resources Available	Source of Clock Resource
Clock input pins	All	48 single-ended or 24 differential	`CLK[0..23][p,n]` pins
GCLK networks	All	16	`CLK[0..23][p,n]` pins, PLL clock outputs, and logic array
RCLK networks	All	92	`CLK[0..23][p,n]` pins, PLL clock outputs, and logic array

Clock Resource	Device	Number of Resources Available	Source of Clock Resource
PCLK networks	• Stratix V GS D3 and D4 • Stratix V GX A3 (with 24 transceivers)	210	DPA clock outputs, PLD-transceiver interface clocks, I/O pins, and logic array
	• Stratix V GS D5 • Stratix V GX A3 (with 36 transceivers), A4, B5, and A6	282	
	• Stratix V GS D6 and D8 • Stratix V GT C5 and C7 • Stratix V GX A5 and A7	306	
	• Stratix V E E9 and EB • Stratix V GX A9, AB, B9, and BB	342	

For more information about the clock input pins connections, refer to the pin connection guidelines.

Related Information

- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
- **Stratix V GT Device Family Pin Connection Guidelines**

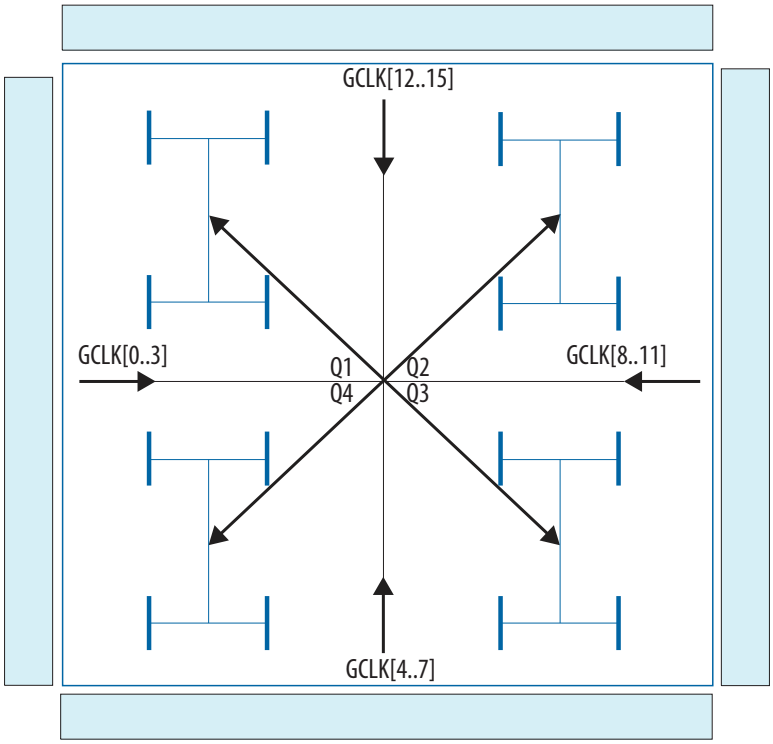
Types of Clock Networks

Global Clock Networks

Stratix V devices provide GCLKs that can drive throughout the device. The GCLKs serve as low-skew clock sources for functional blocks, such as adaptive logic modules (ALMs), digital signal processing (DSP), embedded memory, and PLLs. Stratix V I/O elements (IOEs) and internal logic can also drive GCLKs to create internally-generated global clocks and other high fan-out control signals, such as synchronous or asynchronous clear and clock enable signals.

Figure 4-1: GCLK Networks in Stratix V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

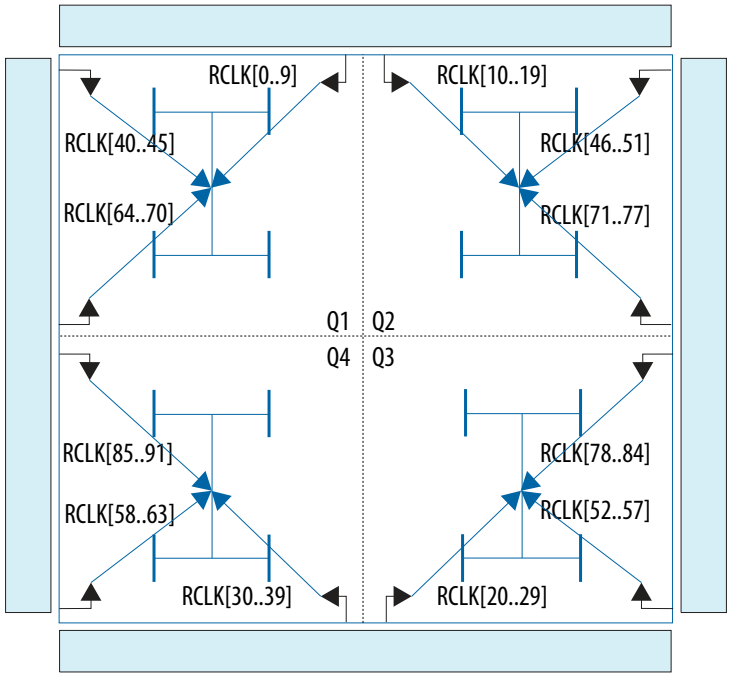


Regional Clock Networks

RCLK networks are only applicable to the quadrant they drive into. RCLK networks provide the lowest clock insertion delay and skew for logic contained within a single device quadrant. The Stratix V IOEs and internal logic within a given quadrant can also drive RCLKs to create internally generated regional clocks and other high fan-out control signals.

Figure 4-2: RCLK Networks in Stratix V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Periphery Clock Networks

Depending on the routing direction, Stratix V devices provide vertical PCLKs from the top and bottom periphery, and horizontal PCLKs from the left and right periphery.

Clock outputs from the dynamic phase aligner (DPA) block, programmable logic device (PLD)-transceiver interface clocks, I/O pins, and internal logic can drive the PCLK networks.

PCLKs have higher skew when compared with GCLK and RCLK networks. You can use PCLKs for general purpose routing to drive signals into and out of the Stratix V device.

Figure 4-3: PCLK Networks for Stratix V GS D5 Device, and Stratix V GX A3 (with 36 transceivers) and A4 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

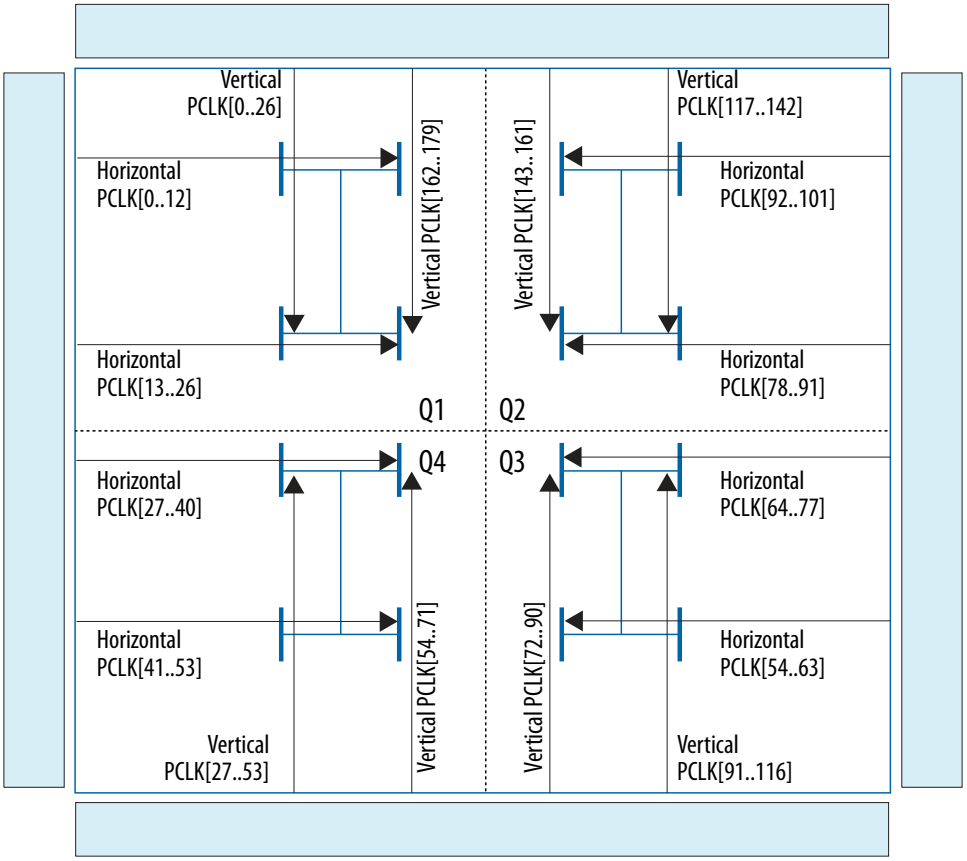


Figure 4-4: PCLK Networks for Stratix V GX B5 and B6 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

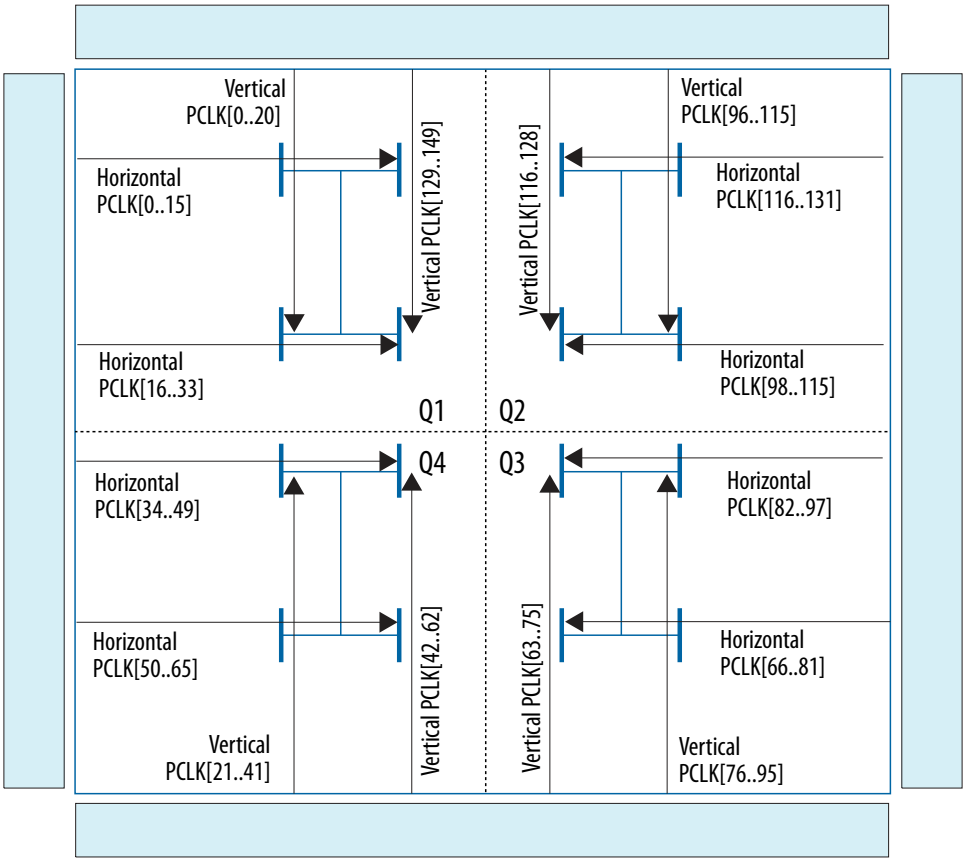


Figure 4-5: PCLK Networks for Stratix V GT C5 and C7 Devices, and Stratix V GX A5 and A7 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

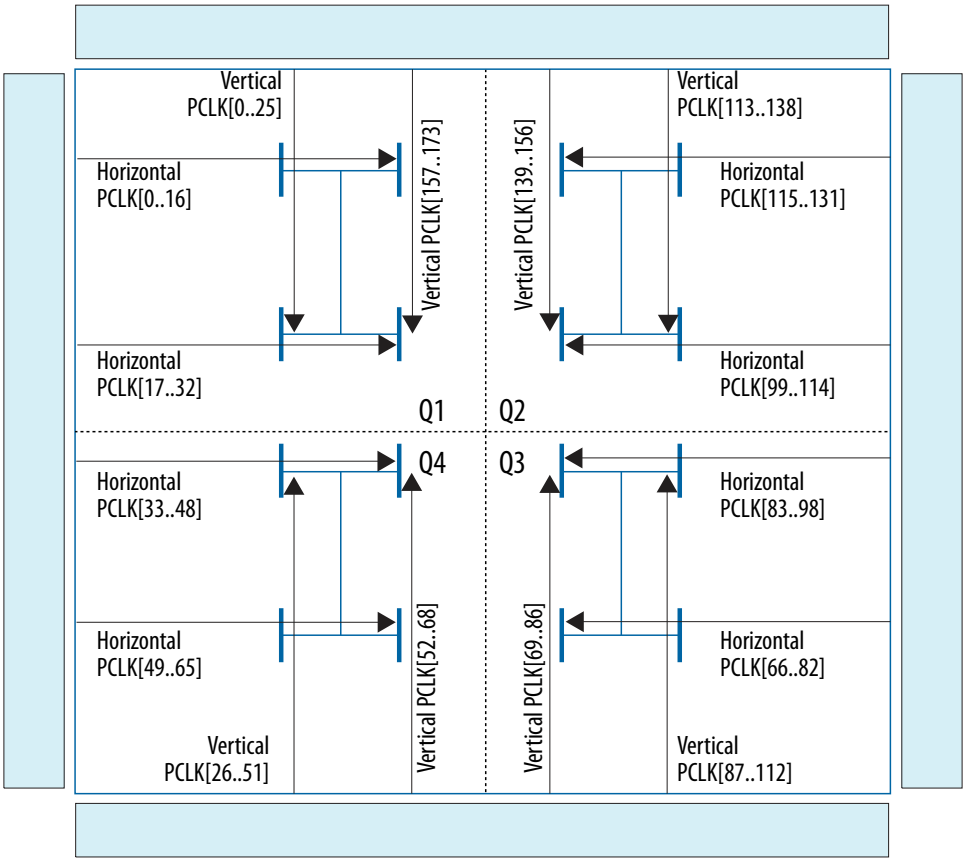


Figure 4-6: PCLK Networks for Stratix V GS D3 and D4 Devices, and Stratix V GX A3 (with 24 transceivers) Device

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

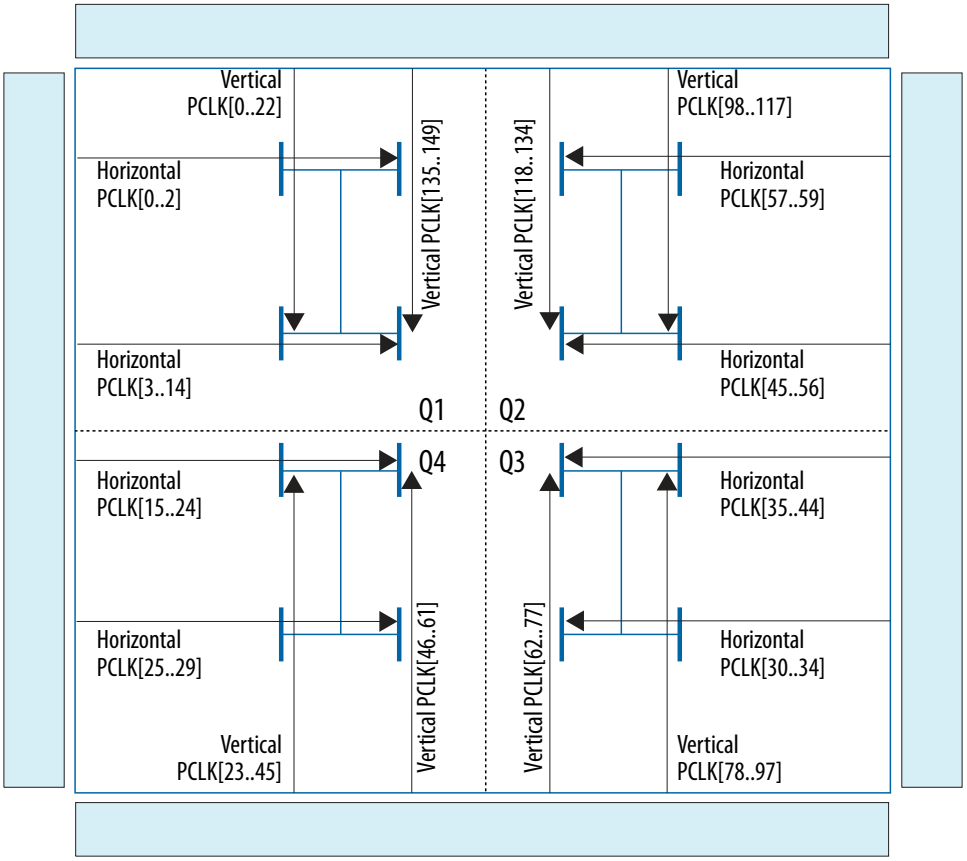


Figure 4-7: PCLK Networks for Stratix V GS D6 and D8 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

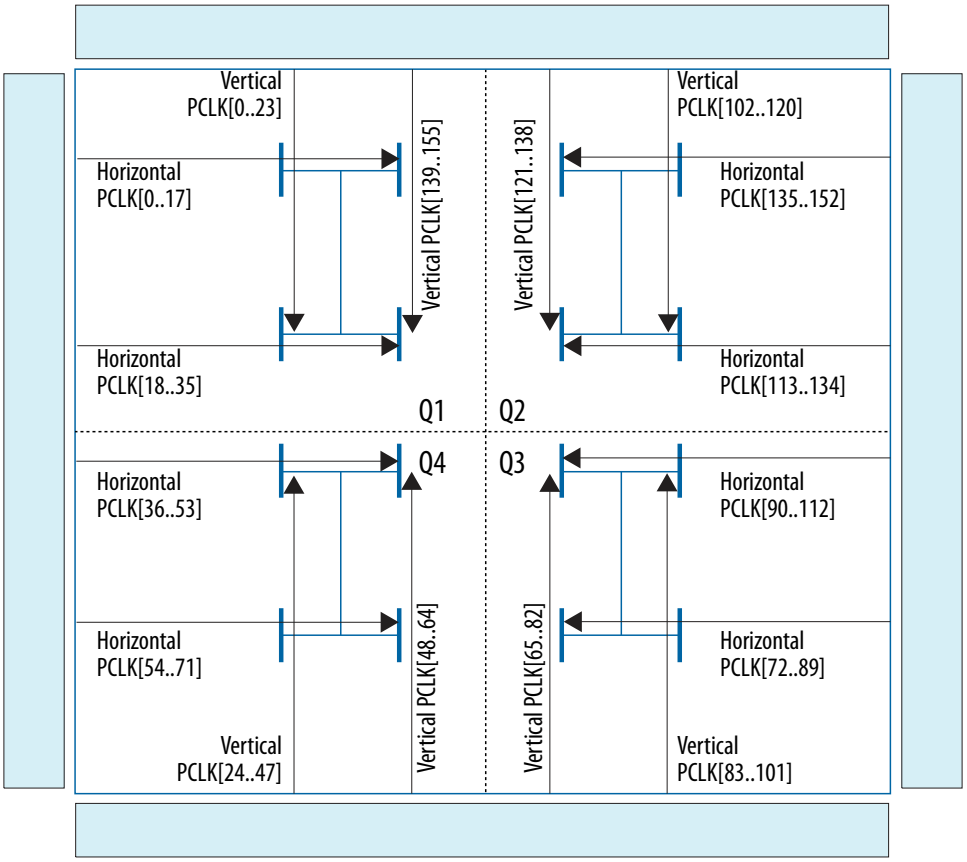
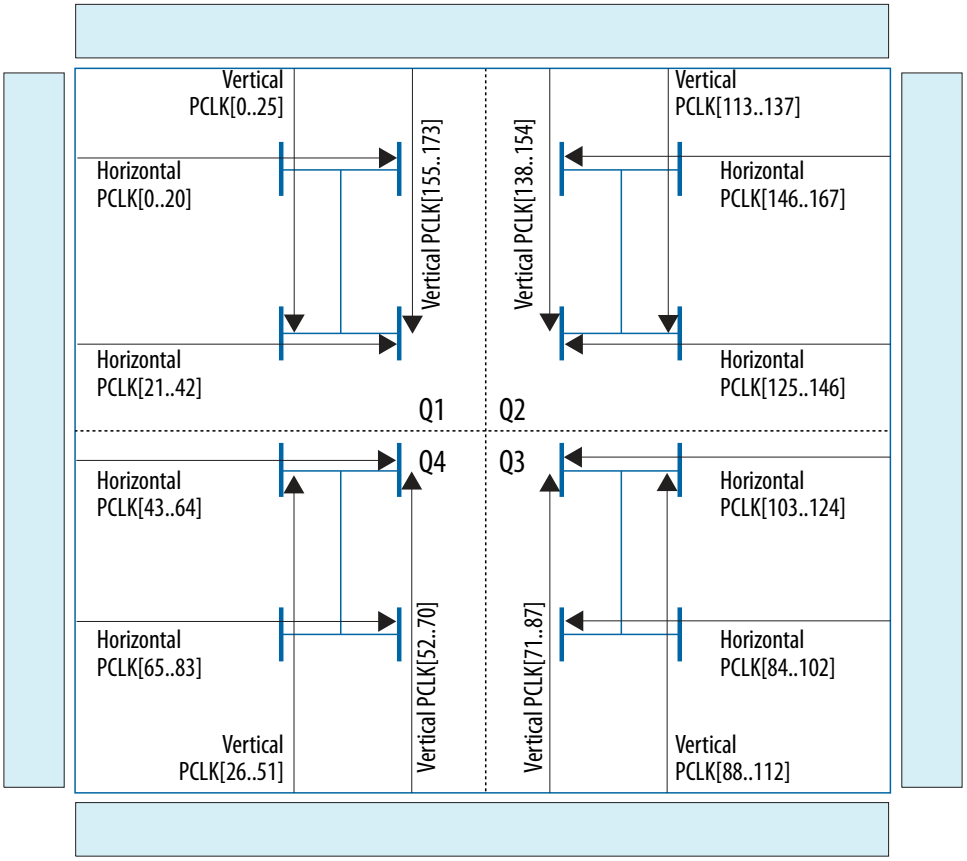


Figure 4-8: PCLK Networks for Stratix V E E9 and EB Devices, and Stratix V GX A9, AB, BB, and B9 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



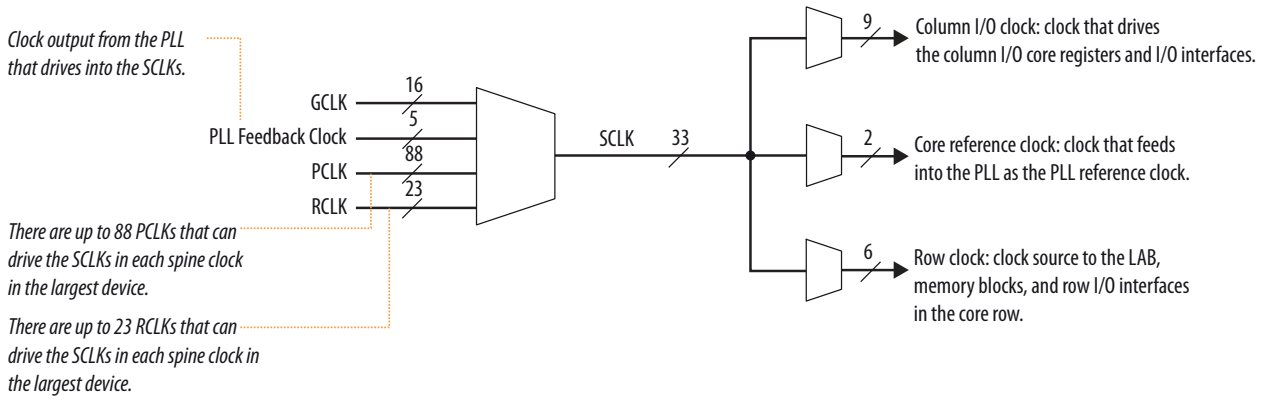
Clock Sources Per Quadrant

The Stratix V devices provide 33 section clock (SCLK) networks in each spine clock per quadrant. The SCLK networks can drive six row clocks in each logic array block (LAB) row, nine column I/O clocks, and two core reference clocks. The SCLKs are the clock resources to the core functional blocks, PLLs, and I/O interfaces of the device.

A spine clock is another layer of routing between the GCLK, RCLK, and PCLK networks before each clock is connected to the clock routing for each LAB row. The settings for spine clocks are transparent. The Quartus II software automatically routes the spine clock based on the GCLK, RCLK, and PCLK networks.

The following figure shows SCLKs driven by the GCLK, RCLK, PCLK, or the PLL feedback clock networks in each spine clock per quadrant. The GCLK, RCLK, PCLK, and PLL feedback clocks share the same routing to the SCLKs. To ensure successful design fitting in the Quartus II software, the total number of clock resources must not exceed the SCLK limits in each region.

Figure 4-9: Hierarchical Clock Networks in Each Spine Clock Per Quadrant



Types of Clock Regions

This section describes the types of clock regions in Stratix V devices.

Entire Device Clock Region

To form the entire device clock region, a source drives a signal in a GCLK network that can be routed through the entire device. The source is not necessarily a clock signal. This clock region has the maximum insertion delay when compared with other clock regions, but allows the signal to reach every destination in the device. It is a good option for routing global reset and clear signals or routing clocks throughout the device.

Regional Clock Region

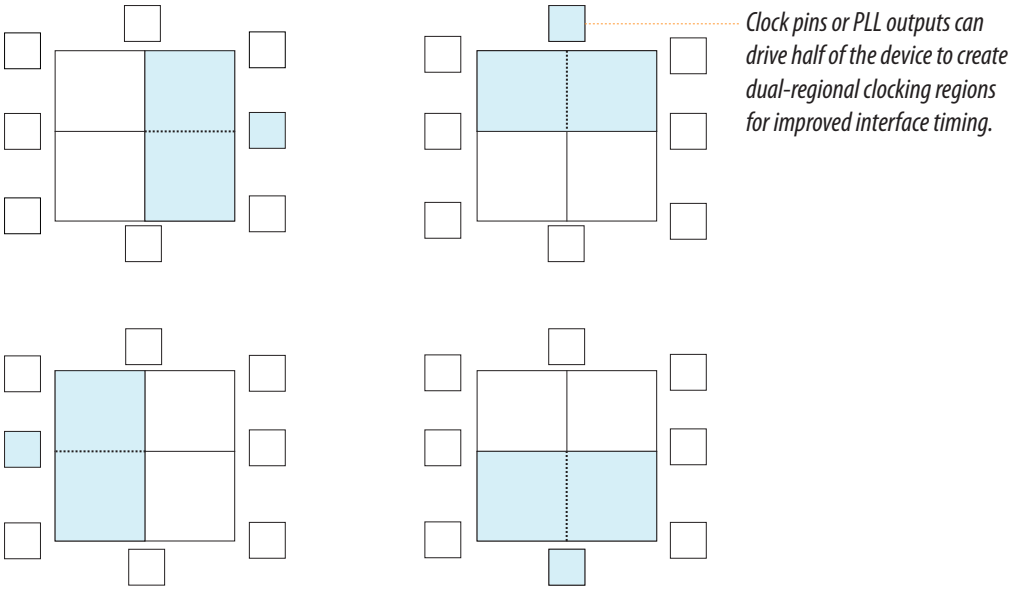
To form a regional clock region, a source drives a signal in a RCLK network that you can route throughout one quadrant of the device. This clock region provides the lowest skew in a quadrant. It is a good option if all the destinations are in a single quadrant.

Dual-Regional Clock Region

To form a dual-regional clock region, a single source (a clock pin or PLL output) generates a dual-regional clock by driving two RCLK networks (one from each quadrant). This technique allows destinations across two adjacent device quadrants to use the same low-skew clock. The routing of this signal on an entire side has approximately the same delay as a RCLK region. Internal logic can also drive a dual-regional clock network.

Figure 4-10: Dual-Regional Clock Region for Stratix V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Clock Network Sources

In Stratix V devices, clock input pins, PLL outputs, high-speed serial interface (HSSI) outputs, DPA outputs, and internal logic can drive the GCLK, RCLK, and PCLK networks.

Dedicated Clock Input Pins

You can use the dedicated clock input pins (`CLK[0..23][p,n]`) for high fan-out control signals, such as asynchronous clears, presets, and clock enables, for protocol signals through the GCLK or RCLK networks.

`CLK` pins can be either differential clocks or single-ended clocks. When you use the `CLK` pins as single-ended clock inputs, only the `CLK<#>p` pins have dedicated connections to the PLL. The `CLK<#>n` pins drive the PLLs over global or regional clock networks and do not have dedicated routing paths to the PLLs.

Driving a PLL over a global or regional clock can lead to higher jitter at the PLL input, and the PLL will not be able to fully compensate for the global or regional clock. Altera recommends using the `CLK<#>p` pins for optimal performance when you use single-ended clock inputs to drive the PLLs.

Internal Logic

You can drive each GCLK, RCLK, and horizontal PCLK network using LAB-routing and row clock to enable internal logic to drive a high fan-out, low-skew signal.

Note: Internally-generated GCLKs, RCLKs, or PCLKs cannot drive the Stratix V PLLs. The input clock to the PLL has to come from dedicated clock input pins, PLL-fed GCLKs, or PLL-fed RCLKs.

DPA Outputs

Every DPA generates one PCLK to the core.

Related Information

High-Speed I/O Design Guidelines for Stratix V Devices on page 6-8
Provides more information about DPA and HSSI outputs.

HSSI Outputs

Every three HSSI outputs generate a group of six PCLKs to the core.

Related Information

High-Speed I/O Design Guidelines for Stratix V Devices on page 6-8
Provides more information about DPA and HSSI outputs.

PLL Clock Outputs

The Stratix V PLL clock outputs can drive both GCLK and RCLK networks.

Clock Input Pin Connections to GCLK and RCLK Networks

Table 4-2: Dedicated Clock Input Pin Connectivity to the GCLK Networks for Stratix V Devices

Clock Resources	CLK (p/n Pins)
GCLK[0,1,2,3]	CLK[0,1,2,3,20,21,22,23]
GCLK[4,5,6,7]	CLK[4,5,6,7]
GCLK[8,9,10,11]	CLK[8,9,10,11,12,13,14,15]
GCLK[12,13,14,15]	CLK[16,17,18,19]

Table 4-3: Dedicated Clock Input Pin Connectivity to the RCLK Networks for Stratix V Devices

A given clock input pin can drive two adjacent RCLK networks to create a dual-regional clock network.

Clock Resources	CLK (p/n Pins)
RCLK[58,59,60,61,62,63,64,68,85,89]	CLK[0]
RCLK[58,59,60,61,62,63,65,69,86,90]	CLK[1]
RCLK[58,59,60,61,62,63,66,70,87,91]	CLK[2]
RCLK[58,59,60,61,62,63,67,88]	CLK[3]
RCLK[20,24,28,30,34,38]	CLK[4]
RCLK[21,25,29,31,35,39]	CLK[5]
RCLK[22,26,32,36]	CLK[6]
RCLK[23,27,33,37]	CLK[7]
RCLK[52,53,54,55,56,57,71,75,78,82]	CLK[8]
RCLK[52,53,54,55,56,57,72,76,79,83]	CLK[9]
RCLK[52,53,54,55,56,57,73,77,80,84]	CLK[10]

Clock Resources	CLK (p/n Pins)
RCLK[52,53,54,55,56,57,74,81]	CLK[11]
RCLK[46,47,48,49,50,51,71,75,78,82]	CLK[12]
RCLK[46,47,48,49,50,51,72,76,79,83]	CLK[13]
RCLK[46,47,48,49,50,51,73,77,80,84]	CLK[14]
RCLK[46,47,48,49,50,51,74,81]	CLK[15]
RCLK[0,4,8,10,14,18]	CLK[16]
RCLK[1,5,9,11,15,19]	CLK[17]
RCLK[2,6,12,16]	CLK[18]
RCLK[3,7,13,17]	CLK[19]
RCLK[40,41,42,43,44,45,64,68,85,89]	CLK[20]
RCLK[40,41,42,43,44,45,65,69,86,90]	CLK[21]
RCLK[40,41,42,43,44,45,66,70,87,91]	CLK[22]
RCLK[40,41,42,43,44,45,67,88]	CLK[23]

Clock Output Connections

For Stratix V PLL connectivity to GCLK and RCLK networks, refer to the PLL connectivity to GCLK and RCLK networks spreadsheet.

Related Information

PLL Connectivity to GCLK and RCLK Networks for Stratix V Devices

Clock Control Block

Every GCLK, RCLK, and PCLK network has its own clock control block. The control block provides the following features:

- Clock source selection (dynamic selection available only for GCLKs)
- Global clock multiplexing
- Clock power down (static or dynamic clock enable or disable available only for GCLKs and RCLKs)

Pin Mapping in Stratix V Devices

Table 4-4: Mapping Between the Input Clock Pins, PLL Counter Outputs, and Clock Control Block Inputs

Clock	Fed by
inclk[0] and inclk[1]	Any of the four dedicated clock pins on the same side of the Stratix V device.
inclk[2]	PLL counters C0 and C2 from the two center PLLs on the same side of the Stratix V devices.
inclk[3]	PLL counters C1 and C3 from the two center PLLs on the same side of the Stratix V devices.

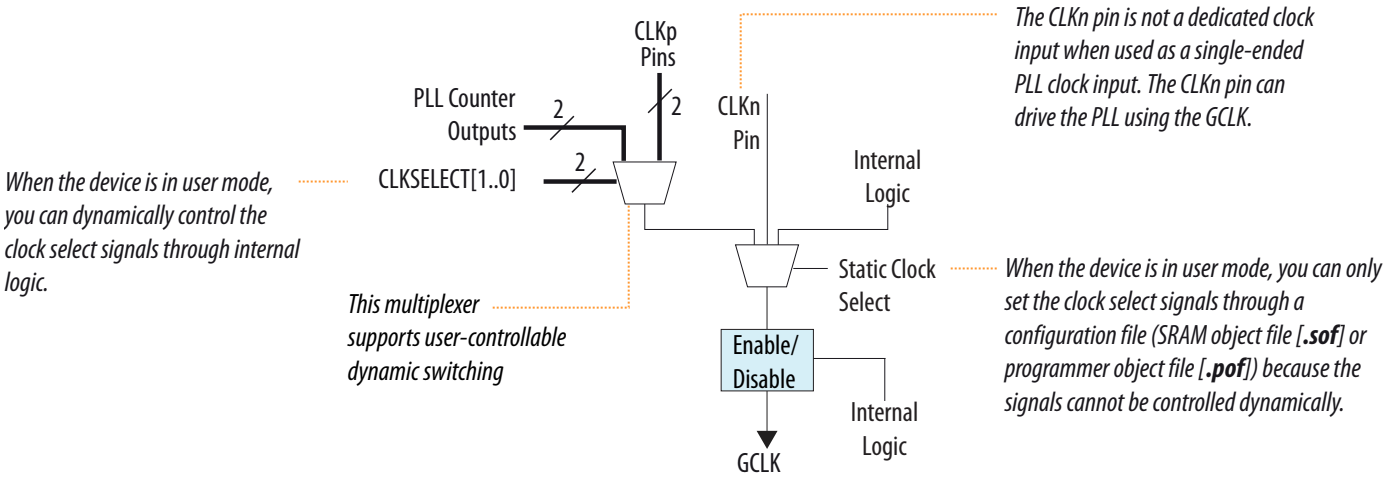
Note: You cannot use corner PLLs for dynamic clock control selection.

GCLK Control Block

You can select the clock source for the GCLK select block either statically or dynamically using internal logic to drive the multiplexer-select inputs.

When selecting the clock source dynamically, you can select either PLL outputs (such as `C0` or `C1`), or a combination of clock pins or PLL outputs.

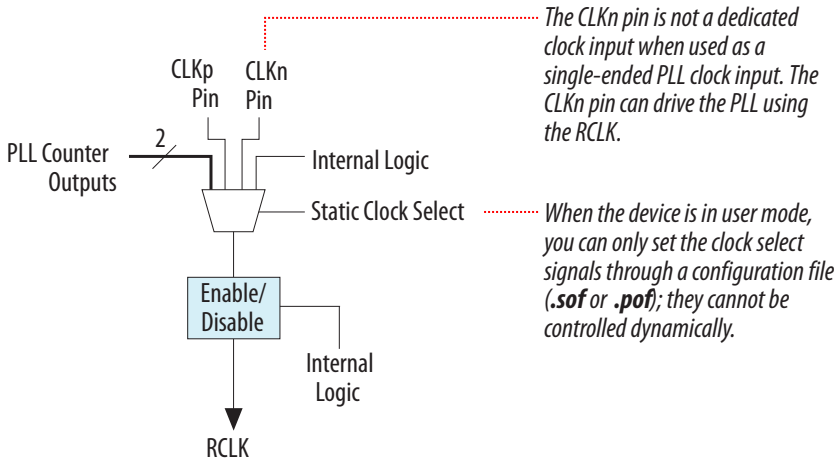
Figure 4-11: GCLK Control Block for Stratix V Devices



RCLK Control Block

You can only control the clock source selection for the RCLK select block statically using configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Quartus II software.

Figure 4-12: RCLK Control Block for Stratix V Devices



You can set the input clock sources and the `clkena` signals for the GCLK and RCLK network multiplexers through the Quartus II software using the ALTCLKCTRL megafunction.

Note: When selecting the clock source dynamically using the ALTCLKCTRL megafunction, choose the inputs using the `CLKSELECT[0..1]` signal. The inputs from the clock pins feed the `inclk[0..1]` ports of the multiplexer, and the PLL outputs feed the `inclk[2..3]` ports.

Related Information

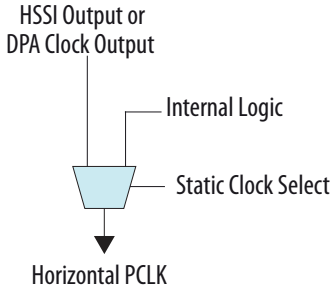
Clock Control Block (ALTCLKCTRL) Megafunction User Guide
Provides more information about ALTCLKCTRL megafunction.

PCLK Control Block

To drive the HSSI horizontal PCLK control block, select the HSSI output or internal logic .

To drive the DPA horizontal PCLK, select the DPA clock output or internal logic. You can only use the DPA clock output to generate the vertical PCLK to the core.

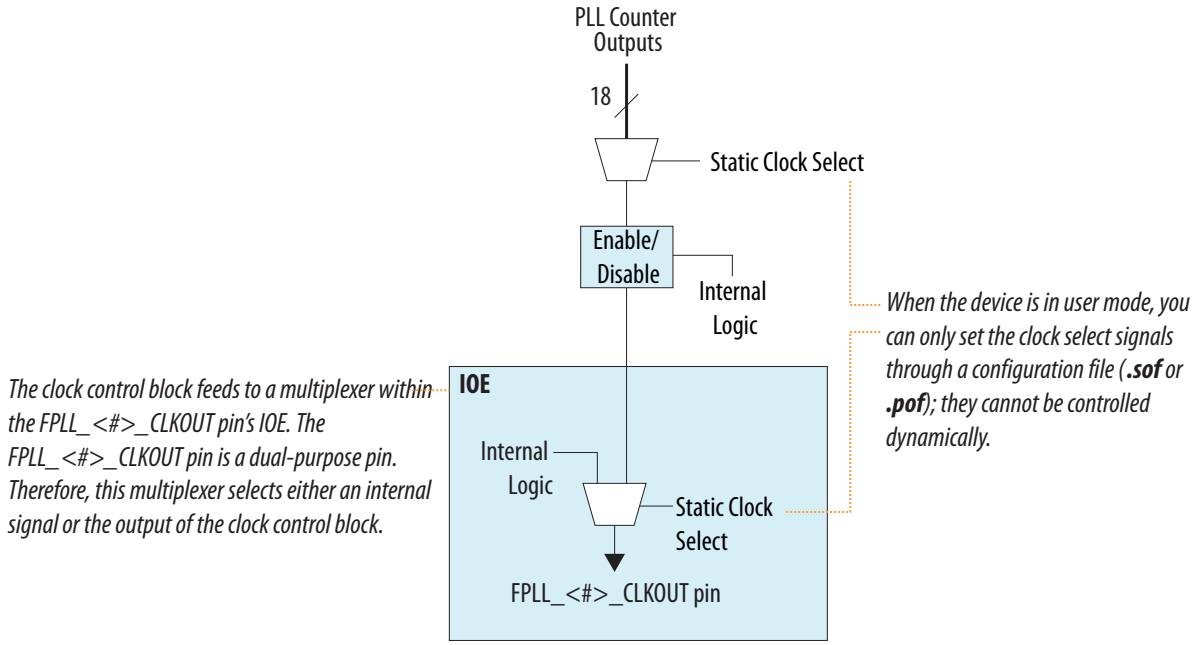
Figure 4-13: Horizontal PCLK Control Block for Stratix V Devices



External PLL Clock Output Control Block

You can enable or disable the dedicated external clock output pins using the ALTCLKCTRL megafunction.

Figure 4-14: External PLL Output Clock Control Block for Stratix V Devices



Related Information

Clock Control Block (ALTCLKCTRL) Megafunction User Guide
Provides more information about ALTCLKCTRL megafunction.

Clock Power Down

You can power down the GCLK and RCLK clock networks using both static and dynamic approaches.

When a clock network is powered down, all the logic fed by the clock network is in off-state, reducing the overall power consumption of the device. The unused GCLK, RCLK, and PCLK networks are automatically powered down through configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Quartus II software.

The dynamic clock enable or disable feature allows the internal logic to control power-up or power-down synchronously on the GCLK and RCLK networks, including dual-regional clock regions. This feature is independent of the PLL and is applied directly on the clock network.

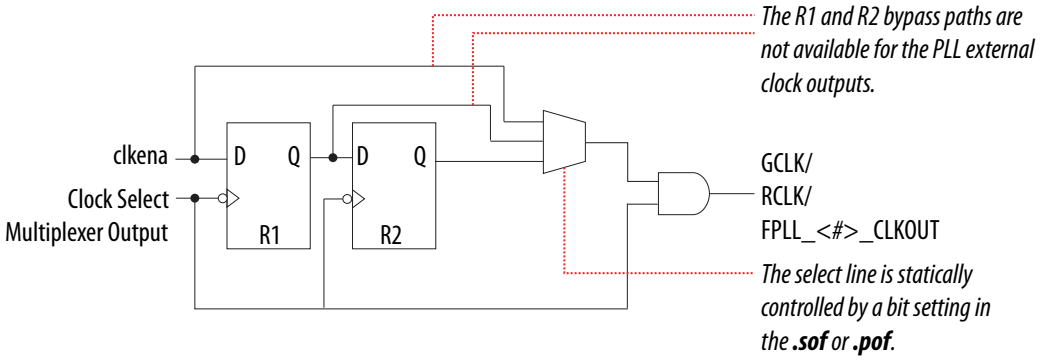
Note: You cannot dynamically enable or disable GCLK or RCLK networks that drive PLLs.

Clock Enable Signals

You cannot use the clock enable and disable circuit of the clock control block if the GCLK or RCLK output drives the input of a PLL.

Figure 4-15: `clkena` Implementation with Clock Enable and Disable Circuit

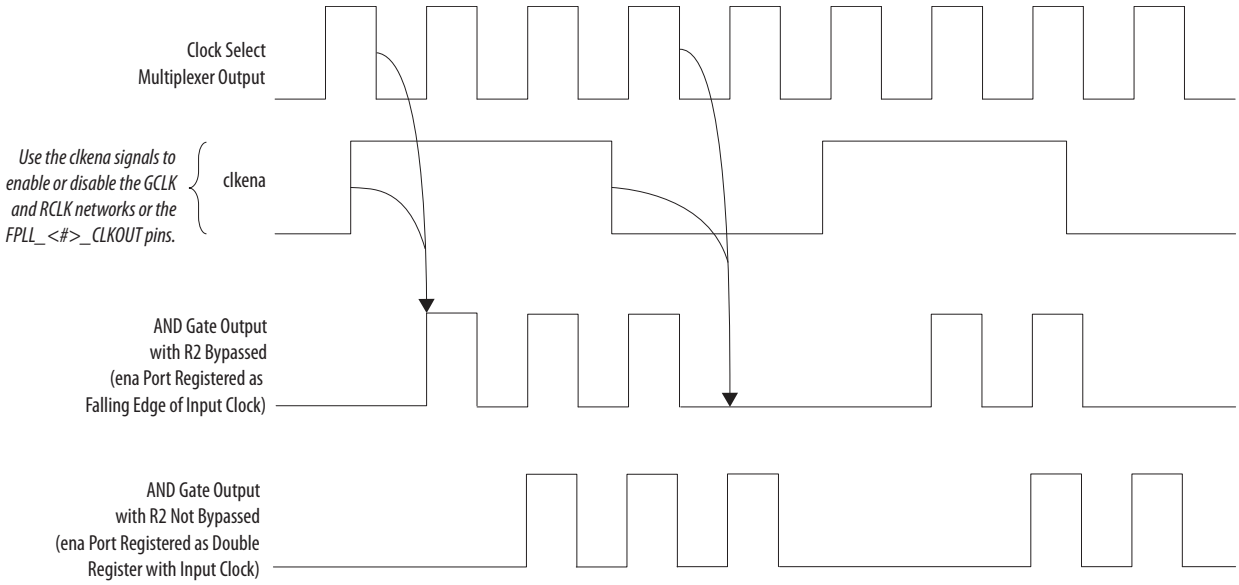
This figure shows the implementation of the clock enable and disable circuit of the clock control block.



The `clkena` signals are supported at the clock network level instead of at the PLL output counter level. This allows you to gate off the clock even when you are not using a PLL. You can also use the `clkena` signals to control the dedicated external clocks from the PLLs.

Figure 4-16: Example of `clkena` Signals

This figure shows a waveform example for a clock output enable. The `clkena` signal is synchronous to the falling edge of the clock output.



Stratix V devices have an additional metastability register that aids in asynchronous enable and disable of the GCLK and RCLK networks. You can optionally bypass this register in the Quartus II software.

The PLL can remain locked, independent of the `clkena` signals, because the loop-related counters are not affected. This feature is useful for applications that require a low-power or sleep mode. The `clkena` signal can also disable clock outputs if the system is not tolerant of frequency overshoot during resynchronization.

Stratix V PLLs

PLLs provide robust clock management and synthesis for device clock management, external system clock management, and high-speed I/O interfaces.

The Stratix V device family contains fractional PLLs that can function as fractional PLLs or integer PLLs. The output counters in Stratix V devices are dedicated to each fractional PLL that support integer or fractional frequency synthesis.

Two adjacent PLLs share 18 C output counters. Any number of C counters can be assigned to each PLL, as long as the total number used by the two PLLs is 18 or less.

The Stratix V devices offer up to 32 fractional PLLs in the larger densities. All Stratix V fractional PLLs have the same core analog structure and features support.

Table 4-5: PLL Features in Stratix V Devices

Feature	Support
Integer PLL	Yes
Fractional PLL	Yes
C output counters	18
M, N, C counter sizes	1 to 512
Dedicated external clock outputs	4 single-ended or 2 single-ended and 1 differential
Dedicated clock input pins	4 single-ended or 4 differential
External feedback input pin	Single-ended or differential
Spread-spectrum input clock tracking	Yes (2)
Source synchronous compensation	Yes
Direct compensation	Yes
Normal compensation	Yes
Zero-delay buffer compensation	Yes
External feedback compensation	Yes
LVDS compensation	Yes
Voltage-controlled oscillator (VCO) output drives the DPA clock	Yes
Phase shift resolution	78.125 ps (3)
Programmable duty cycle	Yes
Power down mode	Yes

(2) Provided input clock jitter is within input jitter tolerance specifications.

(3) The smallest phase shift is determined by the VCO period divided by eight. For degree increments, the Stratix V device can shift all output frequencies in increments of at least 45°. Smaller degree increments are possible depending on the frequency and divide parameters.

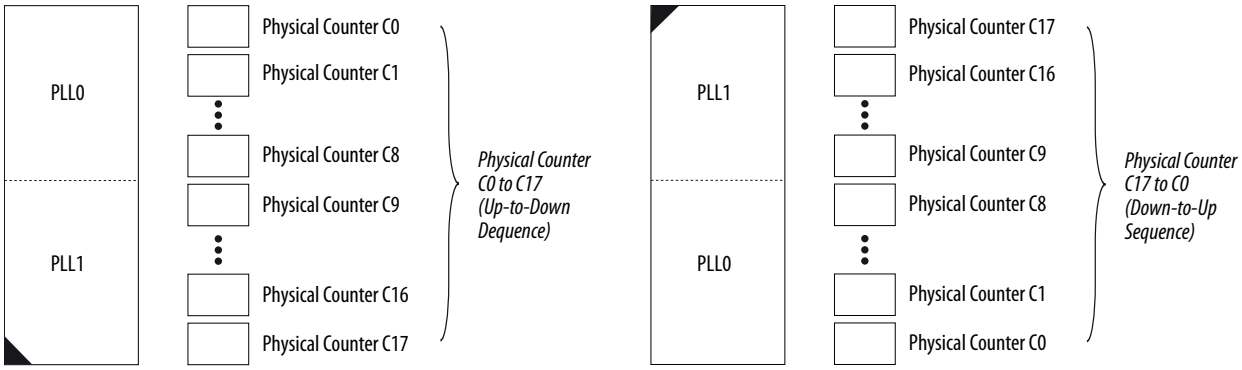
PLL Physical Counters in Stratix V Devices

The physical counters for the fractional PLLs are arranged in the following sequences:

- Up-to-down
- Down-to-up

Figure 4-17: PLL Physical Counters Orientation for Stratix V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



PLL Locations in Stratix V Devices

Stratix V devices provide PLLs for the transceiver channels. These PLLs are located in a strip, where the strip refers to an area in the FPGA.

The total number of PLLs in the Stratix V devices includes the PLLs in the PLL strip. However, the transceivers can only use the PLLs located in the strip.

The following figures show the physical locations of the fractional PLLs. Every index represents one fractional PLL in the device. The physical locations of the fractional PLLs correspond to the coordinates in the Quartus II Chip Planner.

Figure 4-18: PLL Locations for Stratix V GS D5 Device, and Stratix V GX A3 (with 36 transceivers) and A4 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

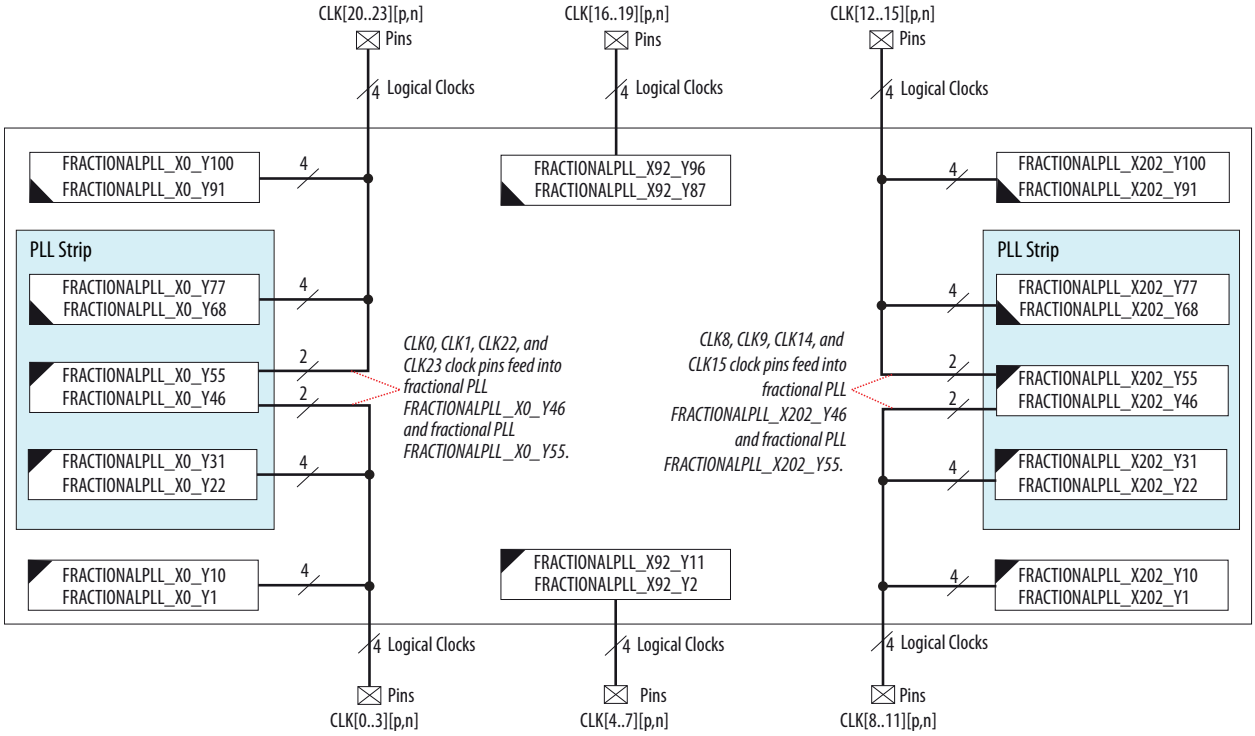


Figure 4-19: PLL Locations for Stratix V GX B5 and B6 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

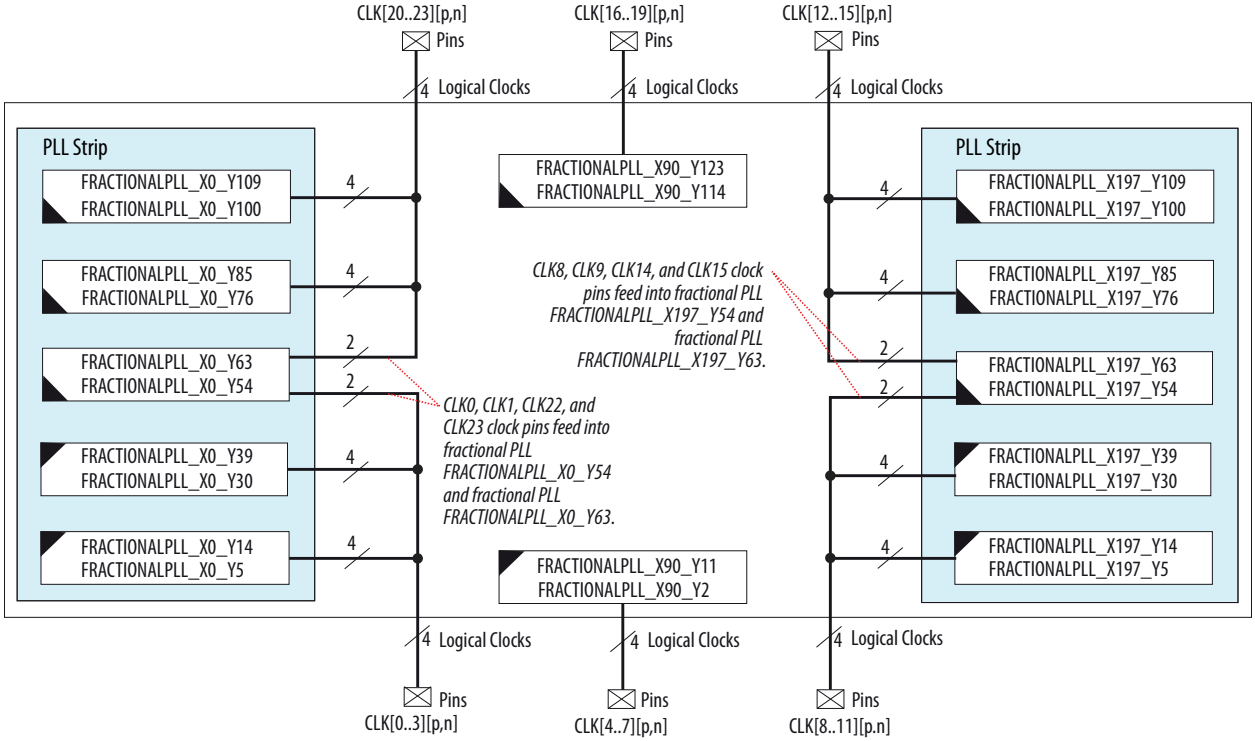


Figure 4-20: PLL Locations for Stratix V GT C5 and C7 Devices, and Stratix V GX A5 and A7 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

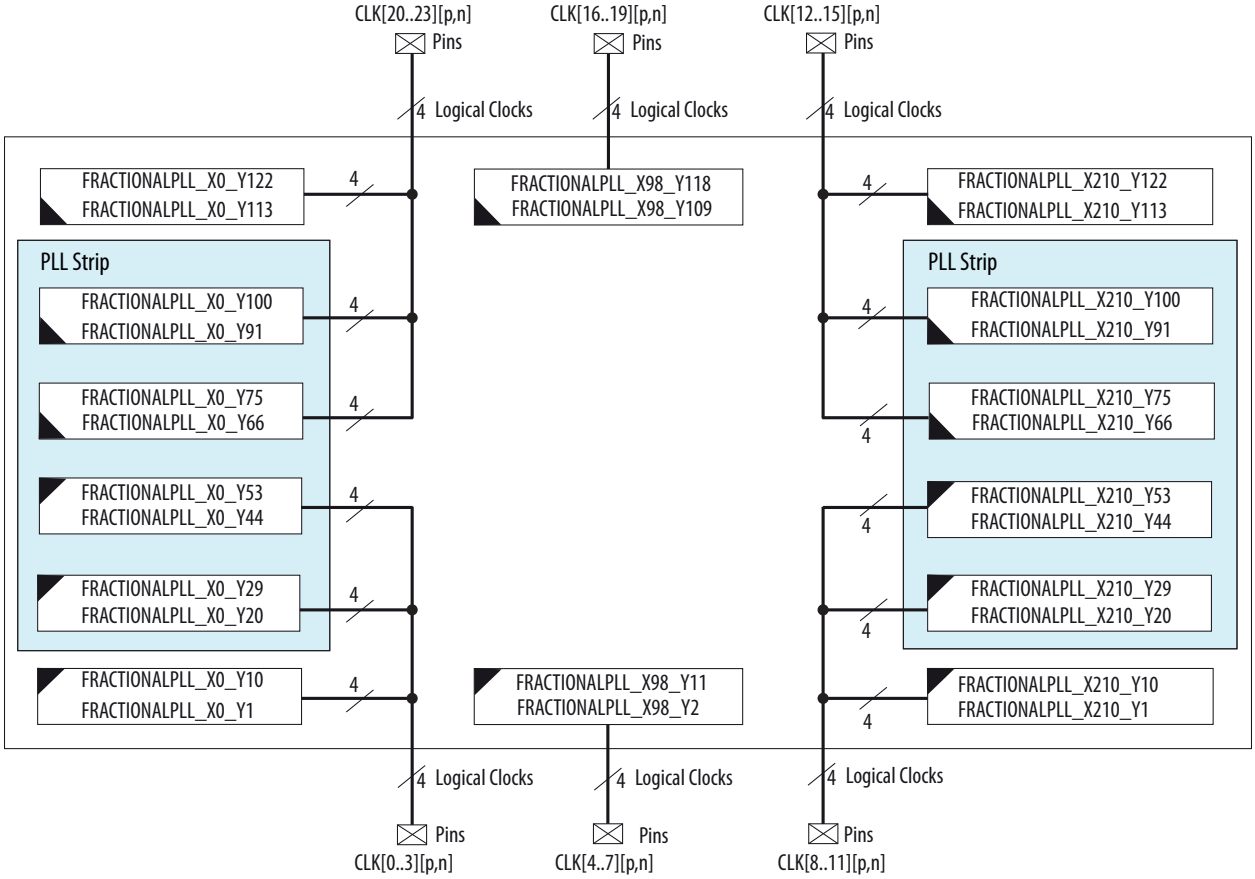


Figure 4-21: PLL Locations for Stratix V GS D3 and D4 Devices, and Stratix V GX A3 (with 24 transceivers) Device

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

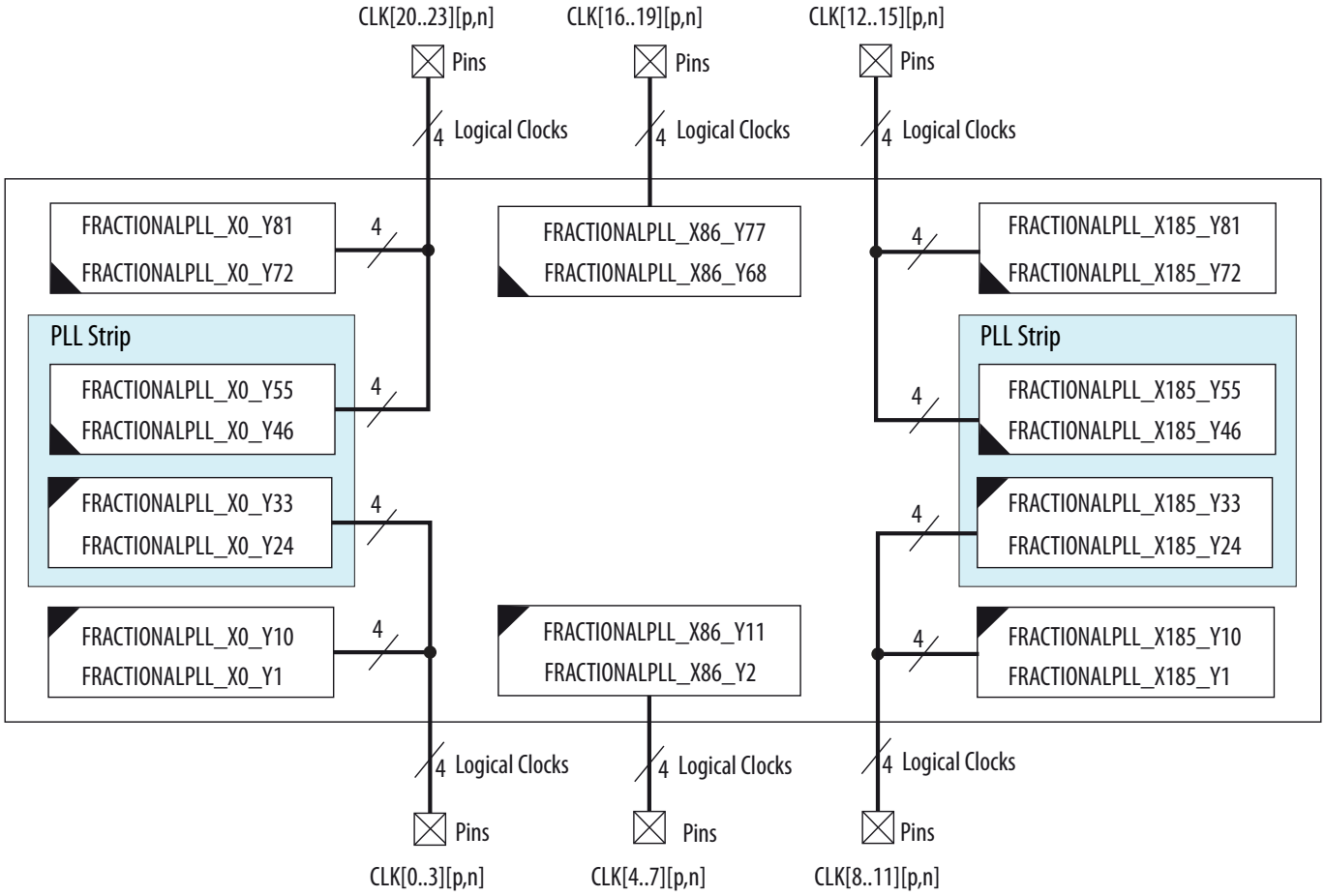


Figure 4-22: PLL Locations for Stratix V GS D6 and D8 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

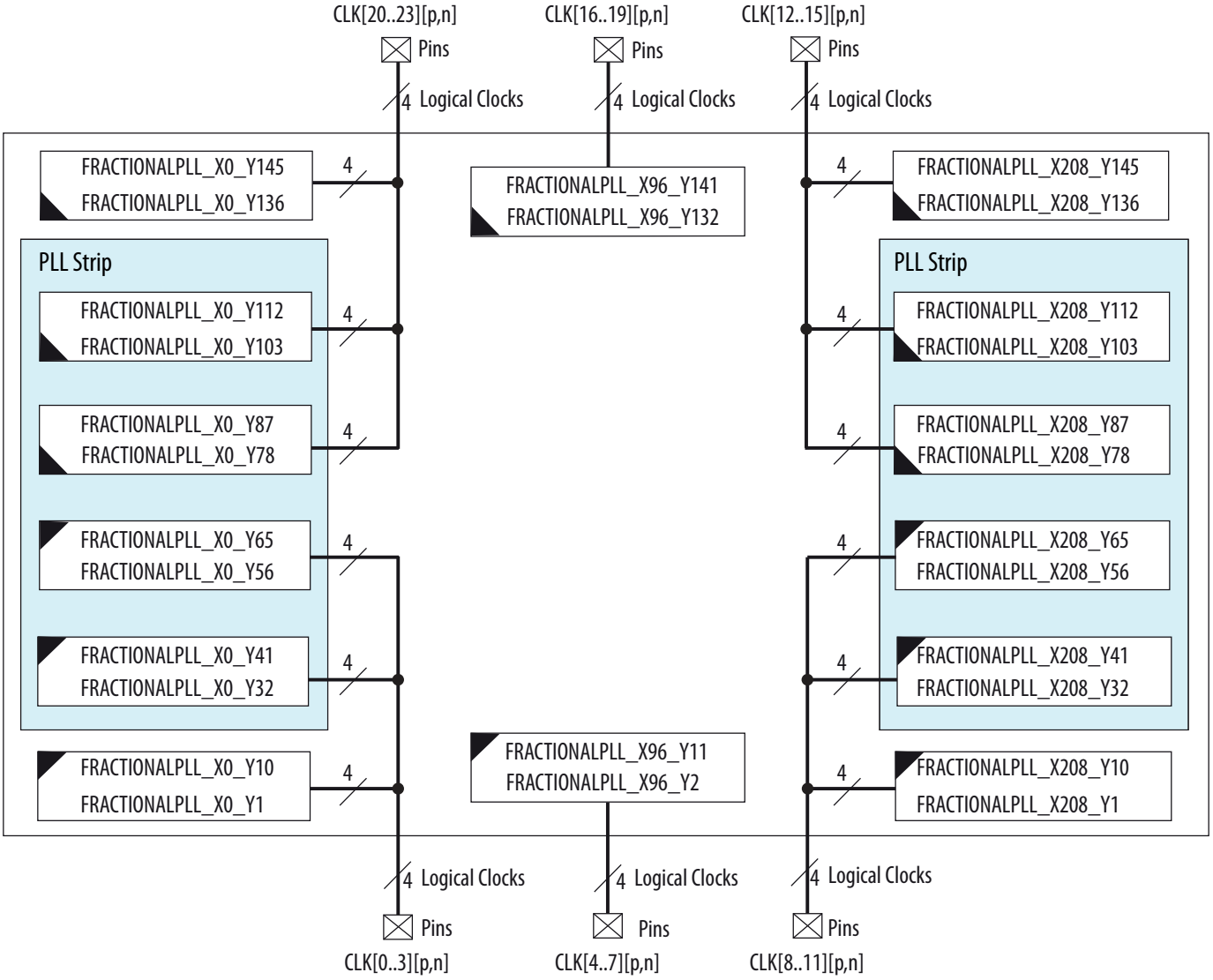
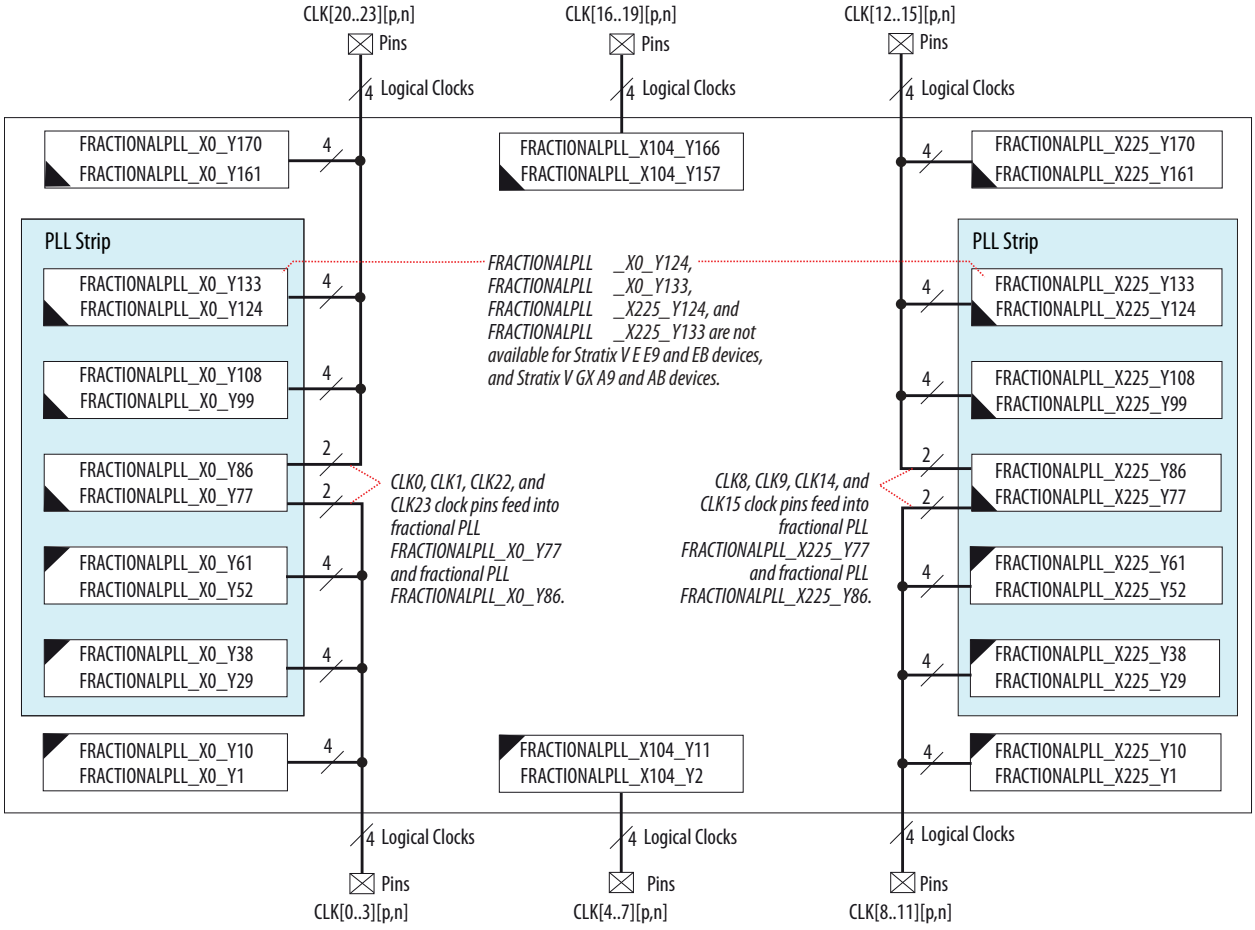


Figure 4-23: PLL Locations for Stratix V E E9 and EB Devices, and Stratix V GX A9, AB, B9, and BB Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Related Information

PLL Migration Guidelines on page 4-26
Provides more information about PLL migration between Stratix V GX A5, A7, A9, AB, B9, BB, D6, and D8 devices.

PLL Migration Guidelines

If you plan to migrate your design between Stratix V GX A5, A7, A9, AB, B9, BB, D6, and D8 devices with 48 transceiver channels, and your design requires a PLL to drive the HSSI and clock network (GCLK or RCLK) simultaneously, use the 2 middle PLLs on the left or right side of the device.

Table 4-6: Location of Middle PLLs for PLL Migration

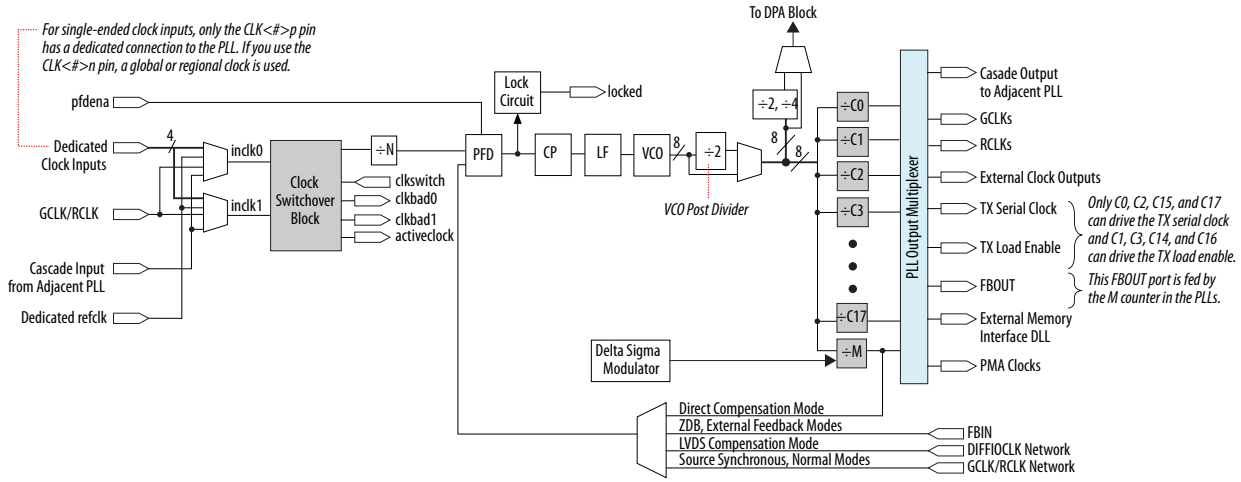
Variant	Member Code	Middle PLL Location	
		Left Side	Right Side
Stratix V GX	A5	FRACTIONALPLL_X0_Y53, FRACTIONALPLL_X0_Y66	FRACTIONALPLL_X210_Y53, FRACTIONALPLL_X210_Y66
	A7		
	A9	FRACTIONALPLL_X0_Y77, FRACTIONALPLL_X0_Y86	FRACTIONALPLL_X225_Y77, FRACTIONALPLL_X225_Y86
	AB		
	B9		
	BB		
	D6	FRACTIONALPLL_X0_Y65, FRACTIONALPLL_X0_Y78	FRACTIONALPLL_X208_Y65, FRACTIONALPLL_X208_Y78
	D8		

Related Information

PLL Locations in Stratix V Devices on page 4-20
Provides more information about CLKIN pin connectivity to the middle PLLs.

Fractional PLL Architecture

Figure 4-24: Fractional PLL High-Level Block Diagram for Stratix V Devices



Fractional PLL Usage

You can configure the fractional PLL to function either in the integer or in the enhanced fractional mode. One fractional PLL can use up to 18 output counters and all external clock outputs. Two adjacent fractional PLLs share the 18 output counters.

Fractional PLLs can be used as follows:

- Reduce the number of required oscillators on the board
- Reduce the clock pins used in the FPGA by synthesizing multiple clock frequencies from a single reference clock source
- Compensate clock network delay
- Zero delay buffering
- Transmit clocking for transceivers

PLL Cascading

Stratix V devices support two types of PLL cascading.

PLL-to-PLL Cascading

This cascading mode synthesizes a more precise output frequency than a single PLL in integer mode. Cascading two PLLs in integer mode expands the effective range of the pre-scale counter, `N` and the multiply counter, `M`.

Stratix V devices use two types of input clock sources.

- The `adjpllin` input clock source is used for inter-cascading between fracturable fractional PLLs.
- The `cclk` input clock source is used for intra-cascading within fracturable fractional PLLs.

Altera recommends using a low bandwidth setting for the source (upstream) PLL and a high bandwidth setting for destination (downstream) PLL.

Counter-Output-to-Counter-Output Cascading

This cascading mode synthesizes a lower frequency output than a single post-scale counter, `C`. Cascading two `C` counters expands the effective range of `C` counters.

PLL External Clock I/O Pins

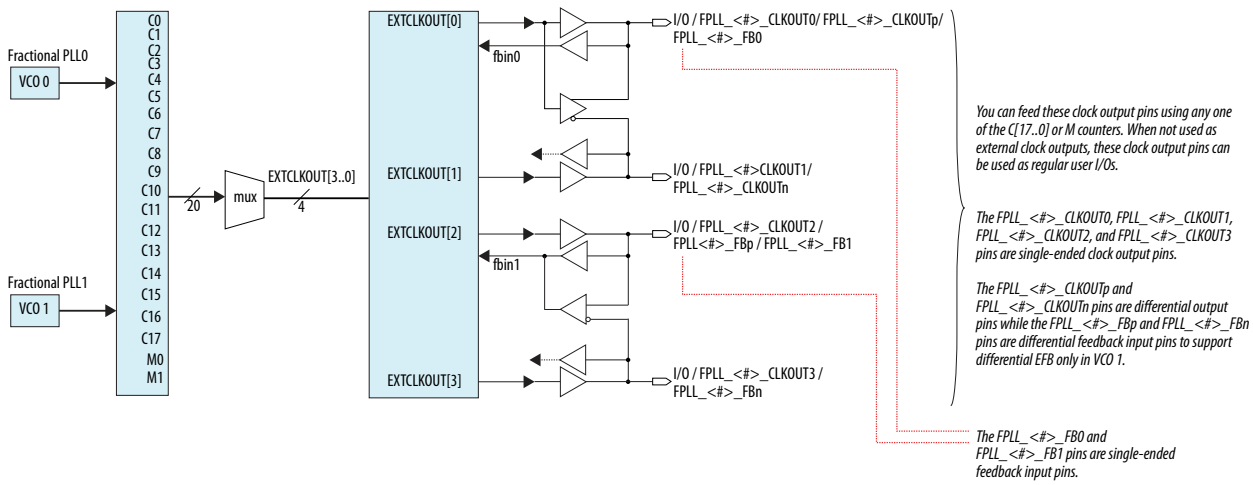
Two adjacent corner and center fractional PLLs share four dual-purpose clock I/O pins, organized as one of the following combinations:

- Four single-ended clock outputs
- Two single-ended outputs and one differential clock output
- Four single-ended clock outputs and two single-ended feedback inputs in the I/O driver feedback for zero delay buffer (ZDB) mode support
- Two single-ended clock outputs and two single-ended feedback inputs for single-ended external feedback (EFB) mode support
- One differential clock output and one differential feedback input for differential EFB support (only one of the two adjacent fractional PLLs can support differential EFB at one time while the other fractional PLL can be used for general-purpose clocking)

Note: All left and right fractional PLLs in Stratix V devices do not support external clock outputs.

The following figure shows that any of the output counters (`C[0..17]`) or the `M` counter on the PLLs can feed the dedicated external clock outputs. Therefore, one counter or frequency can drive all output pins available from a given PLL.

Figure 4-25: Dual-Purpose Clock I/O Pins Associated with PLL for Stratix V Devices



Each pin of a single-ended output pair can be either in-phase or 180° out-of-phase. To implement the 180° out-of-phase pin in a pin pair, the Quartus II software places a NOT gate in the design into the IOE.

The clock output pin pairs support the following I/O standards:

- Same I/O standard for the pin pairs
- LVDS
- Differential high-speed transceiver logic (HSTL)
- Differential SSTL

Stratix V PLLs can drive out to any regular I/O pin through the GCLK or RCLK network. You can also use the external clock output pins as user I/O pins if you do not require external PLL clocking.

Related Information

- **I/O Features in Stratix V Devices**
  Provides more information about I/O standards supported by the PLL clock input and output pins.
- **Zero-Delay Buffer Mode** on page 4-33
- **External Feedback Mode** on page 4-35

PLL Control Signals

You can use the `areset` signal to control PLL operation and resynchronization, and use the `locked` signal to observe the status of the PLL.

areset

The `areset` signal is the reset or resynchronization input for each PLL. The device input pins or internal logic can drive these input signals.

When `areset` is driven high, the PLL counters reset, clearing the PLL output and placing the PLL out-of-lock. The VCO is then set back to its nominal setting. When `areset` is driven low again, the PLL resynchronizes to its input as it re-locks.

You must assert the `areset` signal every time the PLL loses lock to guarantee the correct phase relationship between the PLL input and output clocks. You can set up the PLL to automatically reset (self-reset) after a loss-of-lock condition using the Quartus II MegaWizard Plug-In Manager.

You must include the `areset` signal if either of the following conditions is true:

- PLL reconfiguration or clock switchover is enabled in the design
- Phase relationships between the PLL input and output clocks must be maintained after a loss-of-lock condition

Note: If the input clock to the PLL is not toggling or is unstable after power up, assert the `areset` signal after the input clock is stable and within specifications.

locked

The `locked` signal output of the PLL indicates the following conditions:

- The PLL has locked onto the reference clock.
- The PLL clock outputs are operating at the desired phase and frequency set in the MegaWizard Plug-In Manager.

The lock detection circuit provides a signal to the core logic. The signal indicates when the feedback clock has locked onto the reference clock both in phase and frequency.

Clock Feedback Modes

This section describes the following clock feedback modes:

- Source synchronous
- LVDS compensation
- Direct
- Normal compensation
- ZDB
- EFB

Each mode allows clock multiplication and division, phase shifting, and programmable duty cycle.

The input and output delays are fully compensated by a PLL only when using the dedicated clock input pins associated with a given PLL as the clock source.

The input and output delays may not be fully compensated in the Quartus II software for the following conditions:

- When a GCLK or RCLK network drives the PLL
- When the PLL is driven by a dedicated clock pin that is not associated with the PLL

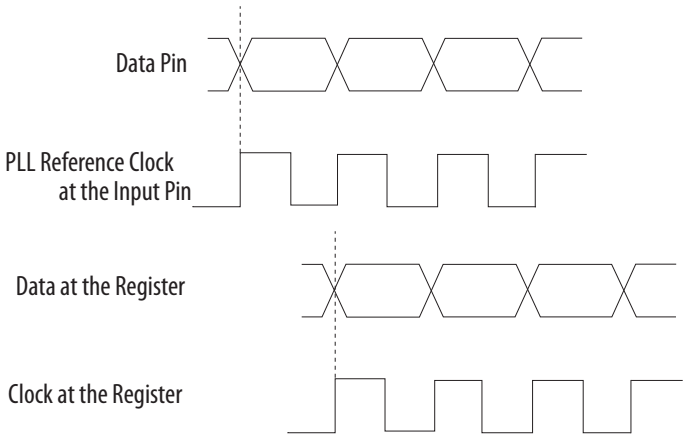
For example, when you configure a PLL in ZDB mode, the PLL input is driven by an associated dedicated clock input pin. In this configuration, a fully compensated clock path results in zero delay between the clock input and one of the clock outputs from the PLL. However, if the PLL input is fed by a non-dedicated input (using the GCLK network), the output clock may not be perfectly aligned with the input clock.

Source Synchronous Mode

If the data and clock arrive at the same time on the input pins, the same phase relationship is maintained at the clock and data ports of any IOE input register. Data and clock signals at the IOE experience similar buffer delays as long as you use the same I/O standard.

Altera recommends source synchronous mode for source synchronous data transfers.

Figure 4-26: Example of Phase Relationship Between Clock and Data in Source Synchronous Mode



The source synchronous mode compensates for the delay of the clock network used and any difference in the delay between the following two paths:

- Data pin to the IOE register input
- Clock input pin to the PLL phase frequency detector (PFD) input

The Stratix V PLL can compensate multiple pad-to-input-register paths, such as a data bus when it is set to use source synchronous compensation mode.

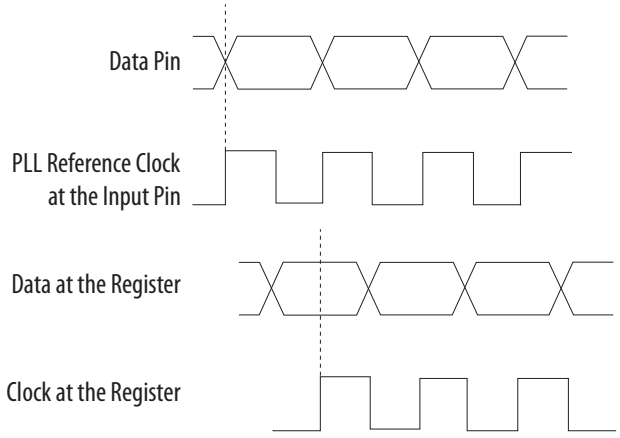
LVDS Compensation Mode

The purpose of LVDS compensation mode is to maintain the same data and clock timing relationship seen at the pins of the internal serializer/deserializer (SERDES) capture register, except that the clock is inverted (180° phase shift). Thus, LVDS compensation mode ideally compensates for the delay of the LVDS clock network, including the difference in delay between the following two paths:

- Data pin-to-SERDES capture register
- Clock input pin-to-SERDES capture register

The output counter must provide the 180° phase shift.

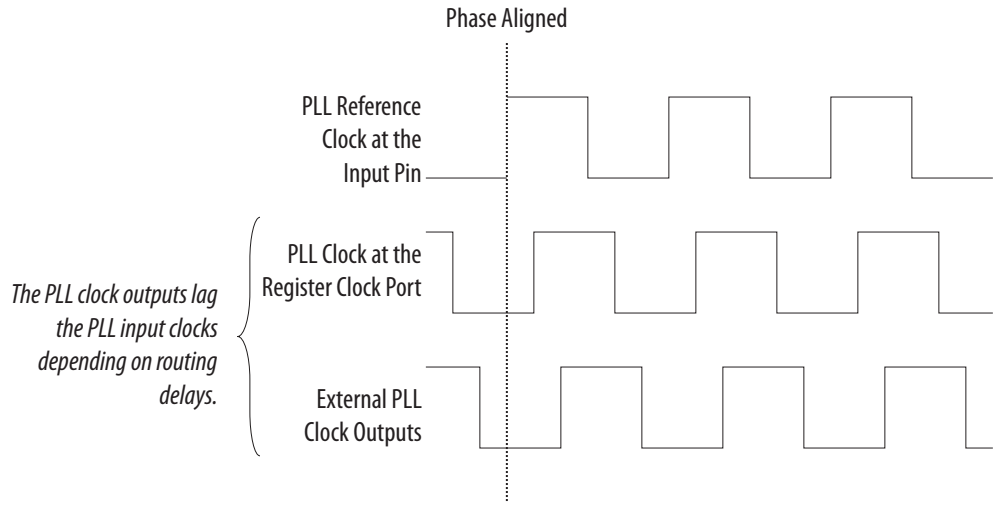
Figure 4-27: Example of Phase Relationship Between the Clock and Data in LVDS Compensation Mode



Direct Mode

In direct mode, the PLL does not compensate for any clock networks. This mode provides better jitter performance because the clock feedback into the PFD passes through less circuitry. Both the PLL internal- and external-clock outputs are phase-shifted with respect to the PLL clock input.

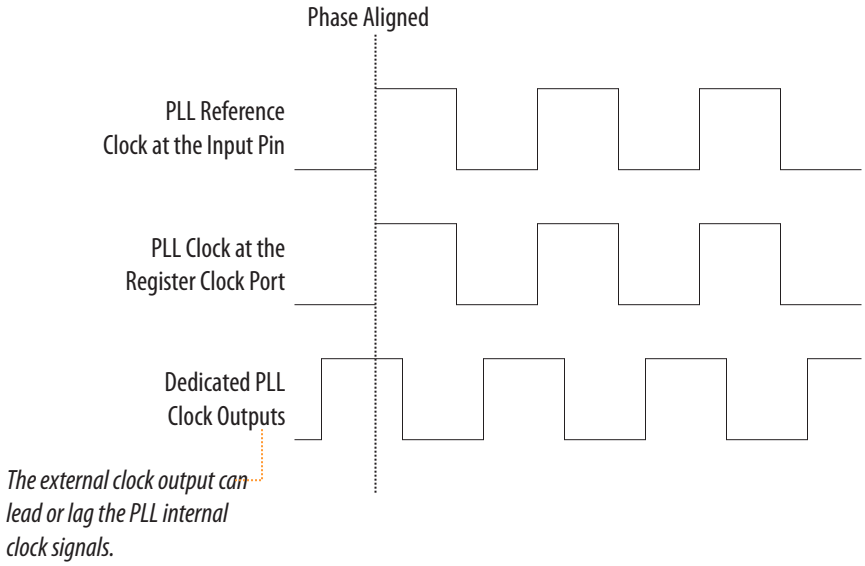
Figure 4-28: Example of Phase Relationship Between the PLL Clocks in Direct Mode



Normal Compensation Mode

An internal clock in normal compensation mode is phase-aligned to the input clock pin. The external clock output pin has a phase delay relative to the clock input pin if connected in this mode. The Quartus II TimeQuest Timing Analyzer reports any phase difference between the two. In normal compensation mode, the delay introduced by the GCLK or RCLK network is fully compensated.

Figure 4-29: Example of Phase Relationship Between the PLL Clocks in Normal Compensation Mode



Zero-Delay Buffer Mode

In ZDB mode, the external clock output pin is phase-aligned with the clock input pin for zero delay through the device. This mode is supported only on the center and corner PLLs in Stratix V devices.

When using this mode, you must use the same I/O standard on the input clocks and clock outputs to guarantee clock alignment at the input and output pins. You cannot use differential I/O standards on the PLL clock input or output pins.

To ensure phase alignment between the `clk` pin and the external clock output (`CLKOUT`) pin in ZDB mode, instantiate a bidirectional I/O pin in the design. The bidirectional I/O pin serves as the feedback path connecting the `fbout` and `fbin` ports of the PLL. The bidirectional I/O pin must always be assigned a single-ended I/O standard. The PLL uses this bidirectional I/O pin to mimic and compensate for the output delay from the clock output port of the PLL to the external clock output pin.

Note: To avoid signal reflection when using ZDB mode, do not place board traces on the bidirectional I/O pin.

Figure 4-30: ZDB Mode in Stratix V PLLs

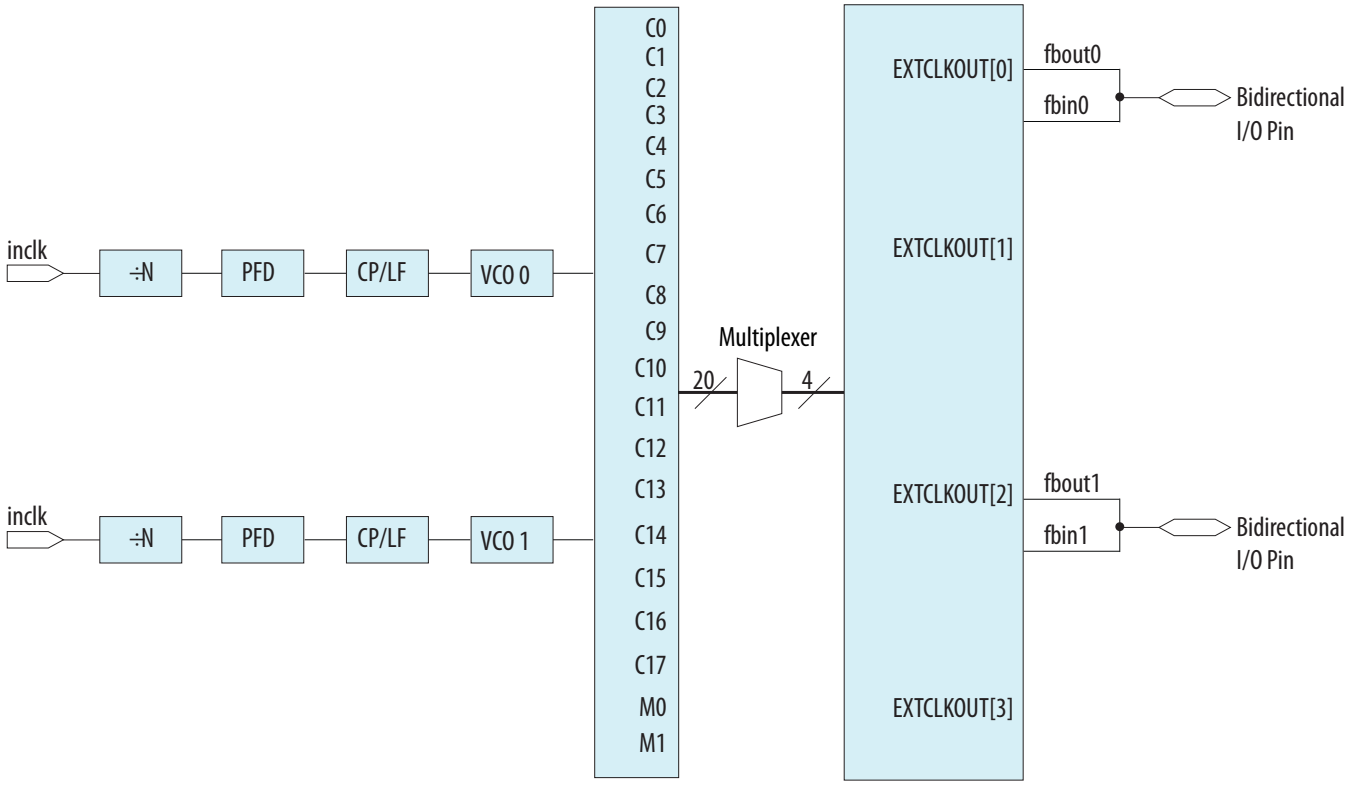
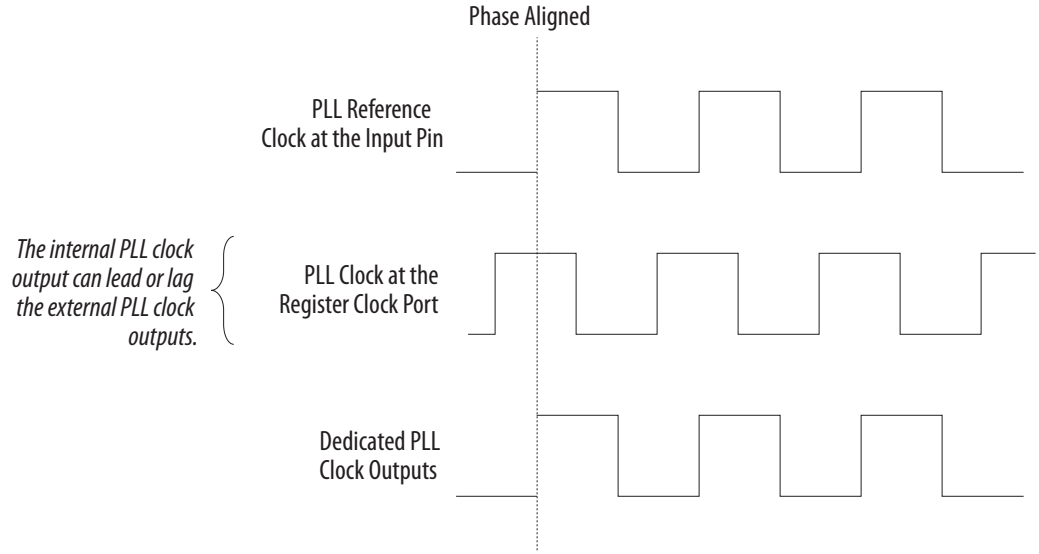


Figure 4-31: Example of Phase Relationship Between the PLL Clocks in ZDB Mode



Related Information

PLL External Clock I/O Pins on page 4-28
Provides more information about PLL clock outputs.

External Feedback Mode

In EFB mode, the output of the `M` counter (`fbout`) feeds back to the PLL `fbin` input (using a trace on the board) and becomes part of the feedback loop.

One of the dual-purpose external clock outputs becomes the `fbin` input pin in this mode. The external feedback input pin, `fbin` is phase-aligned with the clock input pin. Aligning these clocks allows you to remove clock delay and skew between devices.

When using EFB mode, you must use the same I/O standard on the input clock, feedback input, and clock outputs.

This mode is supported only on the center and corner fractional PLLs in Stratix V devices.

Figure 4-32: EFB Mode in Stratix V Devices

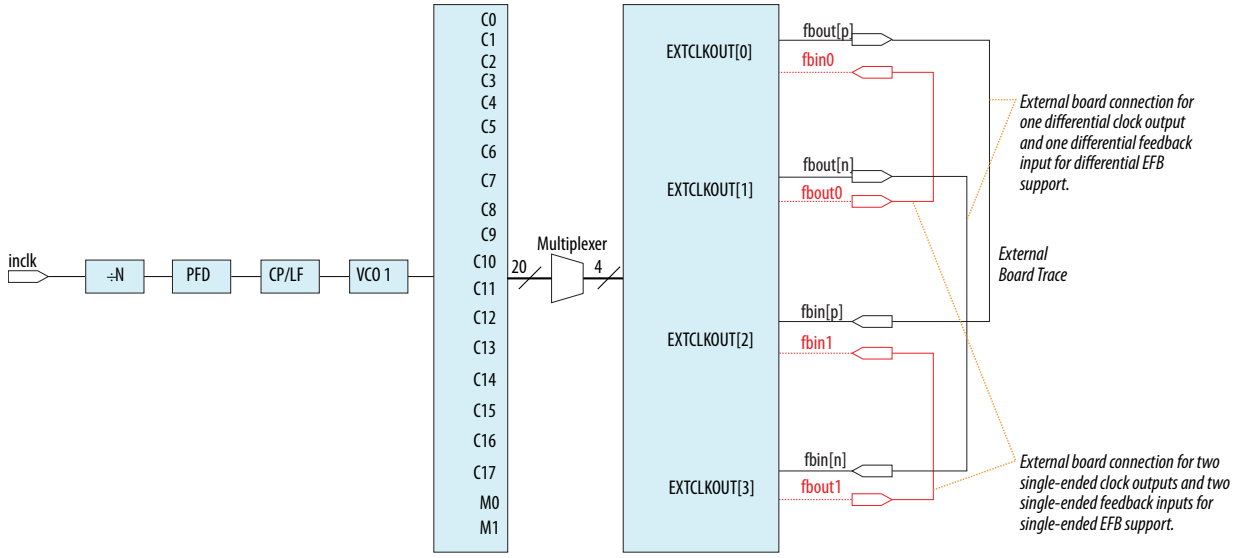
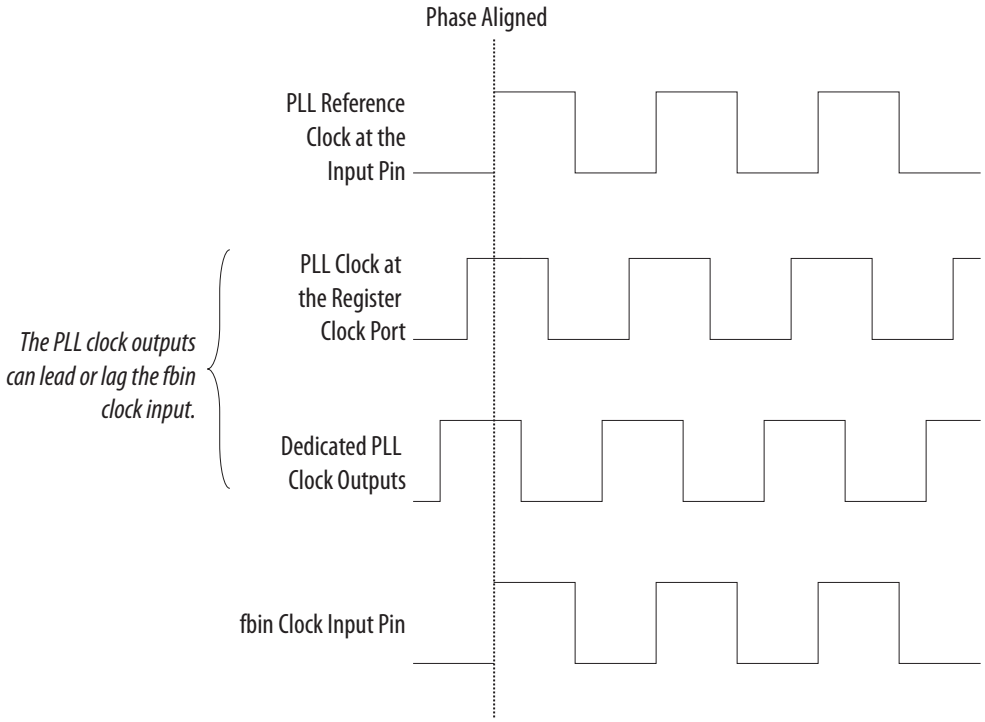


Figure 4-33: Example of Phase Relationship Between the PLL Clocks in EFB Mode



Related Information

PLL External Clock I/O Pins on page 4-28
Provides more information about PLL clock outputs.

Multiple PLLs in Normal Mode and Source Synchronous Mode

Normal and source synchronous compensation feedback mode require GCLK or RCLK feedback path to achieve the required phase relationship. Source synchronous mode for LVDS compensation does not require the GCLK or RCLK feedback path.

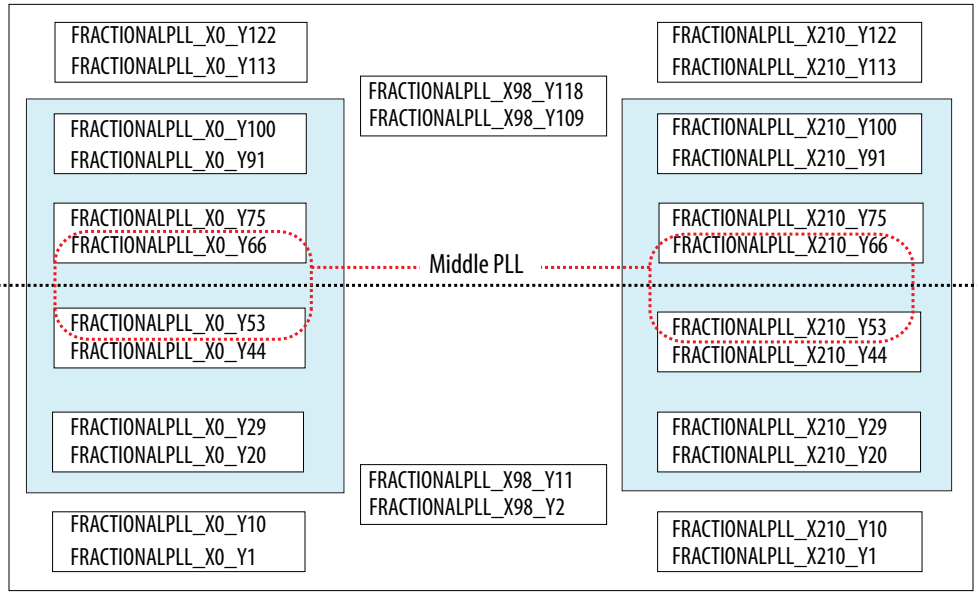
The GCLK or RCLK network feedback paths are fewer than the PLLs available on the device. You cannot implement the compensation mode that requires GCLK or RCLK feedback path on all the PLLs available on the device simultaneously.

Consider the following guidelines when implementing normal compensation or source synchronous compensation mode on multiple PLLs for the device:

- You can implement normal compensation or source synchronous compensation mode on all the center PLLs simultaneously.
- The Stratix V device has two middle PLLs on the left and right side of the device. All PLLs that reside on each side of the device can be divided equally into 2 groups as shown in the following figure.

Figure 4-34: Example of the PLL Grouping for Stratix V GX A5 and A7 Devices, and Stratix V GT C5 and C7 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



From the PLL grouping example, the PLLs can be divided into 4 different sections (upper left, lower left, upper right, and lower right). The PLLs in each of these sections can be further divided into first and second group. The first group consists of the 2 corner PLLs and one middle PLL located in each section. The remaining PLLs in the same section are grouped into the second group. For each section, you can use up to 3 PLLs to implement source synchronous or normal compensation mode in the following combinations:

- Any of the 3 PLLs in the first group
- Any of the 2 PLLs in the first group and 1 PLL in the second group

Table 4-7: Example of the PLL Grouping for Stratix V GX A5 and A7 Devices, and Stratix V GT C5 and C7 Devices

PLL Section	PLL Location	
	First Group	Second Group
Upper left	FRACTIONALPLL_X0_Y122, FRACTIONALPLL_X0_Y113, FRACTIONALPLL_X0_Y66	FRACTIONALPLL_X0_Y100, FRACTIONALPLL_X0_Y91, FRACTIONALPLL_X0_Y75
Lower left	FRACTIONALPLL_X0_Y53, FRACTIONALPLL_X0_Y10, FRACTIONALPLL_X0_Y1	FRACTIONALPLL_X0_Y44, FRACTIONALPLL_X0_Y29, FRACTIONALPLL_X0_Y20

PLL Section	PLL Location	
	First Group	Second Group
Upper right	FRACTIONALPLL_X210_Y122, FRACTIONALPLL_X210_Y113, FRACTIONALPLL_X210_Y66	FRACTIONALPLL_X210_Y100, FRACTIONALPLL_X210_Y91, FRACTIONALPLL_X210_Y75
Lower right	FRACTIONALPLL_X210_Y53, FRACTIONALPLL_X210_Y10, FRACTIONALPLL_X210_Y1	FRACTIONALPLL_X210_Y44, FRACTIONALPLL_X210_Y29, FRACTIONALPLL_X210_Y20

Clock Multiplication and Division

Each Stratix V PLL provides clock synthesis for PLL output ports using the M/(N × C) scaling factors. The input clock is divided by a pre-scale factor, N, and is then multiplied by the M feedback factor. The control loop drives the VCO to match $f_{in} \times (M/N)$.

The Quartus II software automatically chooses the appropriate scaling factors according to the input frequency, multiplication, and division values entered into the ALTERA_PLL megafunction.

VCO Post Divider

A VCO post divider is inserted after the VCO. When you enable the VCO post divider, the VCO post divider divides the VCO frequency by two. When the VCO post divider is bypassed, the VCO frequency goes to the output port without being divided by two.

Post-Scale Counter, C

Each output port has a unique post-scale counter, C, that divides down the output from the VCO post divider. For multiple PLL outputs with different frequencies, the VCO is set to the least common multiple of the output frequencies that meets its frequency specifications. For example, if the output frequencies required from one PLL are 33 and 66 MHz, the Quartus II software sets the VCO to 660 MHz (the least common multiple of 33 and 66 MHz within the VCO range). Then the post-scale counters, C, scale down the VCO frequency for each output port.

Pre-Scale Counter, N and Multiply Counter, M

Each PLL has one pre-scale counter, N, and one multiply counter, M, with a range of 1 to 512 for both M and N. The N counter does not use duty-cycle control because the only purpose of this counter is to calculate frequency division. The post-scale counters have a 50% duty cycle setting. The high- and low-count values for each counter range from 1 to 256. The sum of the high- and low-count values chosen for a design selects the divide value for a given counter.

Delta-Sigma Modulator

The delta-sigma modulator (DSM) is used together with the M multiply counter to enable the PLL to operate in fractional mode. The DSM dynamically changes the M counter divide value on a cycle to cycle basis. The different M counter values allow the "average" M counter value to be a non-integer.

Fractional Mode

In fractional mode, the M counter divide value equals to the sum of the "clock high" count, "clock low" count, and the fractional value. The fractional value is equal to K/2^*X*, where K is an integer between 0 and (2^*X* – 1), and *X* = 8, 16, 24, or 32.

Integer Mode

For PLL operating in integer mode, `M` is an integer value and DSM is disabled.

Related Information

Altera Phase-Locked Loop (ALTERA_PLL) Megafunction User Guide
Provides more information about PLL software support in the Quartus II software.

Programmable Phase Shift

The programmable phase shift feature allows the PLLs to generate output clocks with a fixed phase offset.

The VCO frequency of the PLL determines the precision of the phase shift. The minimum phase shift increment is 1/8 of the VCO period. For example, if a PLL operates with a VCO frequency of 1000 MHz, phase shift steps of 125 ps are possible.

The Quartus II software automatically adjusts the VCO frequency according to the user-specified phase shift values entered into the megafunction.

Programmable Duty Cycle

The programmable duty cycle allows PLLs to generate clock outputs with a variable duty cycle. This feature is supported on the PLL post-scale counters.

The duty-cycle setting is achieved by a low and high time-count setting for the post-scale counters. To determine the duty cycle choices, the Quartus II software uses the frequency input and the required multiply or divide rate.

The post-scale counter value determines the precision of the duty cycle. The precision is defined as 50% divided by the post-scale counter value. For example, if the `C0` counter is 10, steps of 5% are possible for duty-cycle choices from 5% to 90%. If the PLL is in external feedback mode, set the duty cycle for the counter driving the `fbin` pin to 50%.

Combining the programmable duty cycle with programmable phase shift allows the generation of precise non-overlapping clocks.

Clock Switchover

The clock switchover feature allows the PLL to switch between two reference input clocks. Use this feature for clock redundancy or for a dual-clock domain application where a system turns on the redundant clock if the previous clock stops running. The design can perform clock switchover automatically when the clock is no longer toggling or based on a user control signal, `clkswitch`.

The following clock switchover modes are supported in Stratix V PLLs:

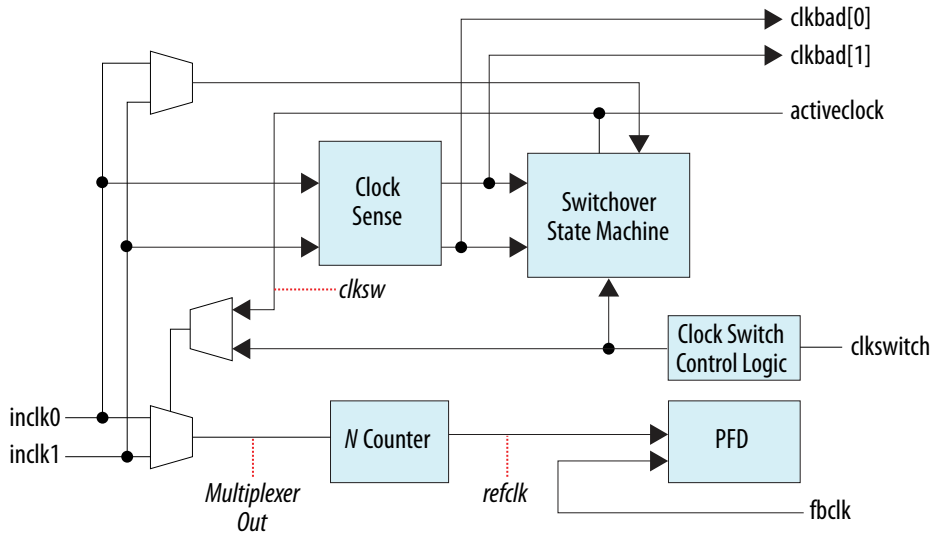
- Automatic switchover—The clock sense circuit monitors the current reference clock. If the current reference clock stops toggling, the reference clock automatically switches to `inclk0` or `inclk1` clock.
- Manual clock switchover—Clock switchover is controlled using the `clkswitch` signal. When the `clkswitch` signal goes from logic low to logic high, and stays high for at least three clock cycles, the reference clock to the PLL is switched from `inclk0` to `inclk1`, or vice-versa.
- Automatic switchover with manual override—This mode combines automatic switchover and manual clock switchover. When the `clkswitch` signal goes high, it overrides the automatic clock switchover function. As long as the `clkswitch` signal is high, further switchover action is blocked.

Automatic Switchover

Stratix V PLLs support a fully configurable clock switchover capability.

Figure 4-35: Automatic Clock Switchover Circuit Block Diagram

This figure shows a block diagram of the automatic switchover circuit built into the PLL.



When the current reference clock is not present, the clock sense block automatically switches to the backup clock for PLL reference. You can select a clock source as the backup clock by connecting it to the `inclk1` port of the PLL in your design.

The clock switchover circuit sends out three status signals—`clkbad[0]`, `clkbad[1]`, and `activeclock`—from the PLL to implement a custom switchover circuit in the logic array.

In automatic switchover mode, the `clkbad[0]` and `clkbad[1]` signals indicate the status of the two clock inputs. When they are asserted, the clock sense block detects that the corresponding clock input has stopped toggling. These two signals are not valid if the frequency difference between `inclk0` and `inclk1` is greater than 20%.

The `activeclock` signal indicates which of the two clock inputs (`inclk0` or `inclk1`) is being selected as the reference clock to the PLL. When the frequency difference between the two clock inputs is more than 20%, the `activeclock` signal is the only valid status signal.

Note: Glitches in the input clock may cause the frequency difference between the input clocks to be more than 20%.

Use the switchover circuitry to automatically switch between `inclk0` and `inclk1` when the current reference clock to the PLL stops toggling. You can switch back and forth between `inclk0` and `inclk1` any number of times when one of the two clocks fails and the other clock is available.

For example, in applications that require a redundant clock with the same frequency as the reference clock, the switchover state machine generates a signal (`clksw`) that controls the multiplexer select input. In this case, `inclk1` becomes the reference clock for the PLL.

When using automatic clock switchover mode, the following requirements must be satisfied:

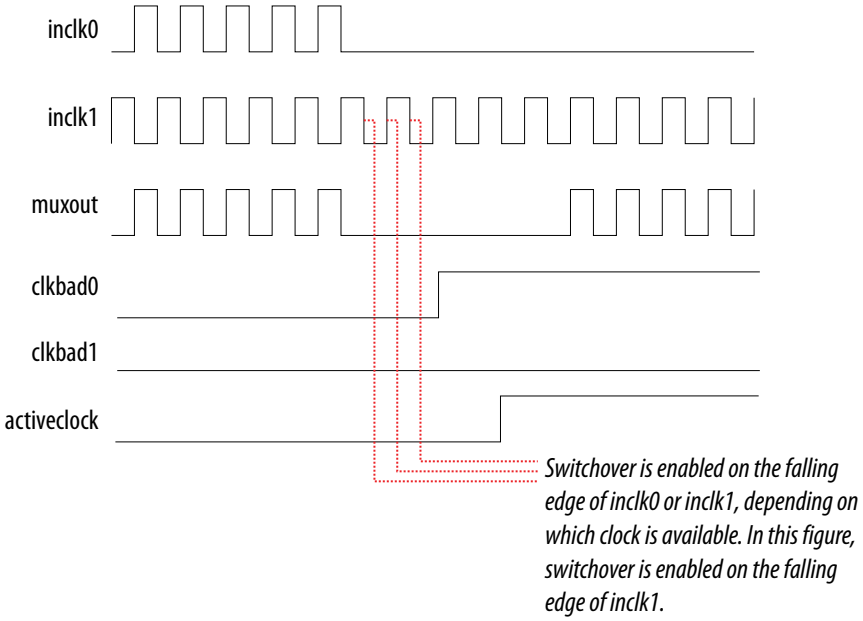
- Both clock inputs must be running when the FPGA is configured.
- The period of the two clock inputs can differ by no more than 20%.

If the current clock input stops toggling while the other clock is also not toggling, switchover is not initiated and the `clkbad[0..1]` signals are not valid. If both clock inputs are not the same frequency, but their period difference is within 20%, the clock sense block detects when a clock stops toggling. However, the PLL may lose lock after the switchover is completed and needs time to relock.

Note: Altera recommends resetting the PLL using the `areset` signal to maintain the phase relationships between the PLL input and output clocks when using clock switchover.

Figure 4-36: Automatic Switchover After Loss of Clock Detection

This figure shows an example waveform of the switchover feature in automatic switchover mode. In this example, the `inclk0` signal is stuck low. After the `inclk0` signal is stuck at low for approximately two clock cycles, the clock sense circuitry drives the `clkbad[0]` signal high. Since the reference clock signal is not toggling, the switchover state machine controls the multiplexer through the `clkswitch` signal to switch to the backup clock, `inclk1`.



Automatic Switchover with Manual Override

In automatic switchover with manual override mode, you can use the `clkswitch` signal for user- or system-controlled switch conditions. You can use this mode for same-frequency switchover, or to switch between inputs of different frequencies.

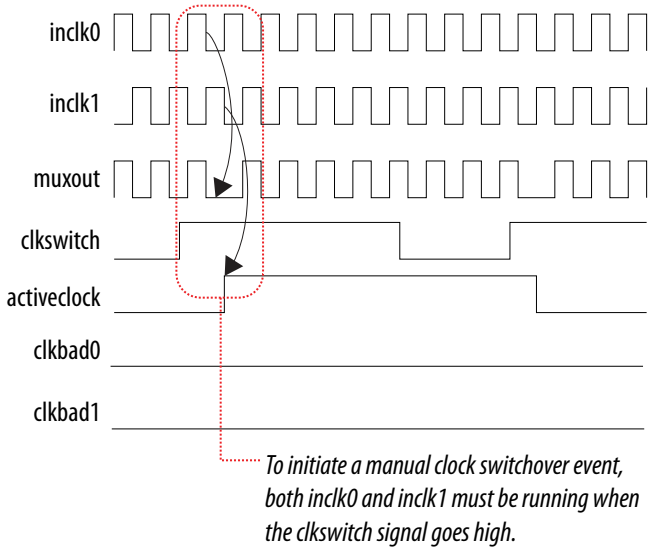
For example, if `inclk0` is 66 MHz and `inclk1` is 200 MHz, you must control switchover using the `clkswitch` signal. The automatic clock-sense circuitry cannot monitor clock input (`inclk0` and `inclk1`) frequencies with a frequency difference of more than 100% (2×).

This feature is useful when the clock sources originate from multiple cards on the backplane, requiring a system-controlled switchover between the frequencies of operation.

You must choose the backup clock frequency and set the M, N, C, and K counters so that the VCO operates within the recommended operating frequency range. The ALTERA_PLL MegaWizard Plug-in Manager notifies you if a given combination of inclk0 and inclk1 frequencies cannot meet this requirement.

Figure 4-37: Clock Switchover Using the clkswitch (Manual) Control

This figure shows a clock switchover waveform controlled by the clkswitch signal. In this case, both clock sources are functional and inclk0 is selected as the reference clock; the clkswitch signal goes high, which starts the switchover sequence. On the falling edge of inclk0, the counter's reference clock, muxout, is gated off to prevent clock glitching. On the falling edge of inclk1, the reference clock multiplexer switches from inclk0 to inclk1 as the PLL reference. The activeclock signal changes to indicate the clock which is currently feeding the PLL.



In automatic override with manual switchover mode, the activeclock signal mirrors the clkswitch signal. Since both clocks are still functional during the manual switch, neither clkbad signal goes high. Because the switchover circuit is positive-edge sensitive, the falling edge of the clkswitch signal does not cause the circuit to switch back from inclk1 to inclk0. When the clkswitch signal goes high again, the process repeats.

The clkswitch signal and automatic switch work only if the clock being switched to is available. If the clock is not available, the state machine waits until the clock is available.

Related Information

Altera Phase-Locked Loop (ALTERA_PLL) Megafunction User Guide
Provides more information about PLL software support in the Quartus II software.

Manual Clock Switchover

In manual clock switchover mode, the clkswitch signal controls whether inclk0 or inclk1 is selected as the input clock to the PLL. By default, inclk0 is selected.

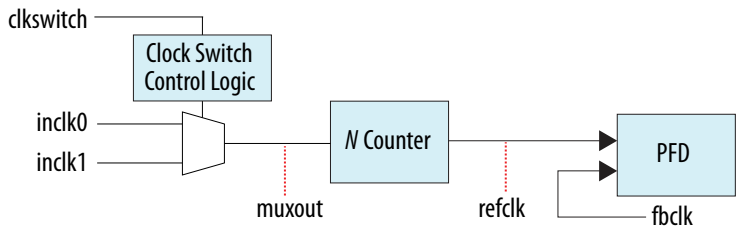
A clock switchover event is initiated when the clkswitch signal transitions from logic low to logic high, and being held high for at least three inclk cycles.

You must bring the clkswitch signal back low again to perform another switchover event. If you do not require another switchover event, you can leave the clkswitch signal in a logic high state after the initial switch.

Pulsing the clkswitch signal high for at least three inclk cycles performs another switchover event.

If inclk0 and inclk1 are different frequencies and are always running, the clkswitchsignal minimum high time must be greater than or equal to three of the slower frequency inclk0 and inclk1 cycles.

Figure 4-38: Manual Clock Switchover Circuitry in Stratix V PLLs



You can delay the clock switchover action by specifying the switchover delay in the ALTERA_PLL megafunction. When you specify the switchover delay, the clkswitch signal must be held high for at least three inclk cycles plus the number of the delay cycles that has been specified to initiate a clock switchover.

Related Information

Altera Phase-Locked Loop (ALTERA_PLL) Megafunction User Guide
Provides more information about PLL software support in the Quartus II software.

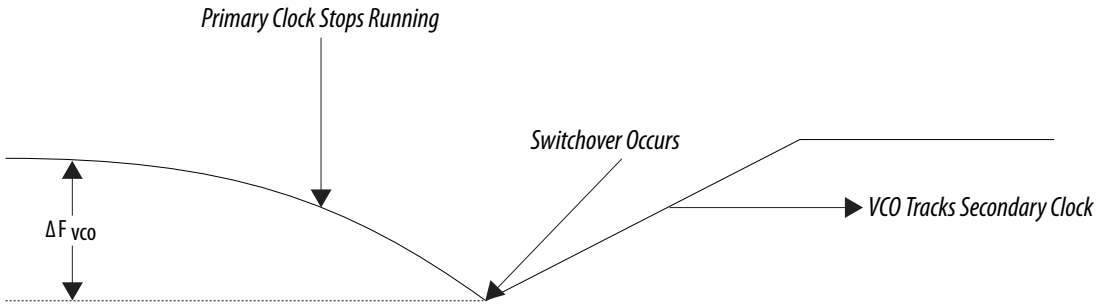
Guidelines

When implementing clock switchover in Stratix V PLLs, use the following guidelines:

- Automatic clock switchover requires that the inclk0 and inclk1 frequencies be within 20% of each other. Failing to meet this requirement causes the clkbad[0] and clkbad[1] signals to not function properly.
- When using manual clock switchover, the difference between inclk0 and inclk1 can be more than 100% (2×). However, differences in frequency, phase, or both, of the two clock sources will likely cause the PLL to lose lock. Resetting the PLL ensures that you maintain the correct phase relationships between the input and output clocks.
- Both inclk0 and inclk1 must be running when the clkswitch signal goes high to initiate the manual clock switchover event. Failing to meet this requirement causes the clock switchover to not function properly.
- Applications that require a clock switchover feature and a small frequency drift must use a low-bandwidth PLL. When referencing input clock changes, the low-bandwidth PLL reacts more slowly than a high-bandwidth PLL. When switchover happens, a low-bandwidth PLL propagates the stopping of the clock to the output more slowly than a high-bandwidth PLL. However, be aware that the low-bandwidth PLL also increases lock time.

- After a switchover occurs, there may be a finite resynchronization period for the PLL to lock onto a new clock. The time it takes for the PLL to relock depends on the PLL configuration.
- The phase relationship between the input clock to the PLL and the output clock from the PLL is important in your design. Assert `areset` for at least 10 ns after performing a clock switchover. Wait for the locked signal to go high and be stable before re-enabling the output clocks from the PLL.
- The VCO frequency gradually decreases when the current clock is lost and then increases as the VCO locks on to the backup clock, as shown in the following figure.

Figure 4-39: VCO Switchover Operating Frequency



PLL Reconfiguration and Dynamic Phase Shift

For more information about PLL reconfiguration and dynamic phase shifting, refer to AN661.

Related Information

AN661: Implementing Fractional PLL Reconfiguration with ALTERA_PLL and ALTERA_PLL_RECONFIG Megafunctions

Document Revision History

Date	Version	Changes
January 2014	2014.01.10	• Removed Preliminary tags for clock resources, clock input pin connections to GCLK and RCLK networks, and PLL features tables. • Updated information on dual-regional clock region. • Added label for PLL strip in PLL locations diagrams. • Added descriptions for PLLs located in a strip. • Updated VCO post-scale counter, K, to VCO post divider. • Added information on PLL cascading. • Added information on programmable phase shift. • Updated automatic clock switchover mode requirement.

Date	Version	Changes
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated PCLK clock sources per device quadrant. • Added PCLK networks resources and diagram for Stratix V E devices. • Updated PCLK clock sources in hierarchical clock networks in each spine clock per quadrant diagram. • Added PCLK networks in clock network sources section. • Updated dedicated clock input pins in clock network sources section. • Added information on C output counters for PLLs. • Added power down mode in PLL features table. • Added information on PLL physical counters. • Updated the PLL locations index from CEN_X<#>_Y<#>, COR_X<#>_Y<#>, and LR_X<#>_Y<#> to FRACTIONALPLL_X<#>_Y<#>. • Removed LVPECL I/O standard support for clock output pin pairs. • Updated PLL support for EFB mode. • Updated the scaling factors for PLL output ports. • Updated the fractional value for PLL in fractional mode. • Moved all links to the Related Information section of respective topics for easy reference. • Reorganized content.
December 2012	2012.12.28	• Added note to indicate that the figures shown are the top view of the silicon die. • Added diagram for PLL physical counter orientation. • Updated PLL locations diagrams. • Removed information on pfdena PLL control signal. • Removed information on PLL Compensation assignment in the Quartus II software. • Updated the fractional value for PLL in fractional mode. • Reorganized content and updated template.
June 2012	1.4	• Added Table 4–5 and Table 4–6. • Added Figure 4–6, Figure 4–8, Figure 4–20, Figure 4–22, and Figure 4–33. • Updated Table 4–1, Table 4–2, and Table 4–3. • Updated Figure 4–3, Figure 4–5, Figure 4–17, Figure 4–18, Figure 4–19, and Figure 4–21. • Added "PLL Migration Guidelines", "Implementing Multiple PLLs in Normal Mode and Source Synchronous Mode", "Clock Switchover", and "PLL Reconfiguration and Dynamic Phase Shift" sections. • Updated "Clock Networks in Stratix V Devices", "Clock Network Sources", and "Clock Multiplication and Division" sections.
November 2011	1.3	Updated Figure 4–19 and Figure 4–28.

Date	Version	Changes
May 2011	1.2	• Chapter moved to volume 2 for the 11.0 release. • Updated Table 4–1. • Updated Figure 4–3, Figure 4–4, Figure 4–5, Figure 4–6, Figure 4–15, Figure 4–17, Figure 4–18, Figure 4–20, Figure 4–25, and Figure 4–28. • Updated "Zero-Delay Buffer Mode" and "External Feedback Mode" sections. • Added "PLL Clock Outputs" section.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

I/O Features in Stratix V Devices

5

2015.01.23

SV51006 Subscribe Send Feedback

This chapter provides details about the features of the Stratix V I/O elements (IOEs) and how the IOEs work in compliance with current and emerging I/O standards and requirements.

The Stratix V I/Os support the following features:

- True LVDS channels in all I/O banks support SGMII, SPI-4.2, and XSBI applications
- Hard dynamic phase alignment (DPA) and serializer/deserializer (SERDES) support in I/O banks on all sides of the device with DPA
- Single-ended, non-voltage-referenced, and voltage-referenced I/O standards
- Low-voltage differential signaling (LVDS), RSDS, mini-LVDS, HSTL, HSUL, and SSTL I/O standards across all I/O banks
- Double data rate (DDR), single data rate (SDR), and half data rate input and output options
- Serializer/deserializer (SERDES)
- Deskew, read and write leveling, and clock-domain crossing functionality for high-performance memory interface
- Programmable output current strength
- Programmable slew-rate
- Programmable bus-hold
- Programmable pull-up resistor
- Programmable pre-emphasis
- Programmable I/O delay
- Programmable voltage output differential (V_{OD})
- Open-drain output
- On-chip series termination (R_S OCT) with and without calibration
- On-chip parallel termination (R_T OCT)
- On-chip differential termination (R_D OCT)

Note: The information in this chapter is applicable to all Stratix V variants, unless noted otherwise.

Related Information

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

I/O Standards Support in Stratix V Devices

This section lists the I/O standards supported in the FPGA I/Os of Stratix V devices, the typical power supply values for each I/O standard, and the MultiVolt I/O interface feature.

I/O Standards Support in Stratix V Devices

Stratix V devices support a wide range of industry I/O standards. These devices support V_{CCIO} voltage levels of 3.0, 2.5, 1.8, 1.5, 1.35, 1.25, and 1.2 V.

Table 5-1: Supported I/O Standards for Stratix V Devices

This table lists the I/O standards for Stratix V devices, as well as the typical applications they support.

I/O Standard	Typical Applications	Standard Support
3.3 V LVTTL/3.3 V LVCMOS[4]	General purpose	JESD8-B
2.5 V LVCMOS	General purpose	JESD8-5
1.8 V LVCMOS	General purpose	JESD8-7
1.5 V LVCMOS	General purpose	JESD8-11
1.2 V LVCMOS	General purpose	JESD8-12
SSTL-2 Class I	DDR SDRAM	JESD8-9B
SSTL-2 Class II	DDR SDRAM	JESD8-9B
SSTL-18 Class I	DDR2 SDRAM	JESD8-15
SSTL-18 Class II	DDR2 SDRAM	JESD8-15
SSTL-15 Class I	DDR3 SDRAM	—
SSTL-15 Class II	DDR3 SDRAM	—
1.8 V HSTL Class I	QDR II/RLDRAM II	JESD8-6
1.8 V HSTL Class II	QDR II/RLDRAM II	JESD8-6
1.5 V HSTL Class I	QDR II/QDR II+/ RLDRAM II	JESD8-6
1.5 V HSTL Class II	QDR II/QDR II+/ RLDRAM II	JESD8-6
1.2 V HSTL Class I	General purpose	JESD8-16A
1.2 V HSTL Class II	General purpose	JESD8-16A
Differential SSTL-2 Class I	DDR SDRAM	JESD8-9B
Differential SSTL-2 Class II	DDR SDRAM	JESD8-9B
Differential SSTL-18 Class I	DDR2 SDRAM	JESD8-15
Differential SSTL-18 Class II	DDR2 SDRAM	JESD8-15
Differential SSTL-15 Class I	DDR3 SDRAM	—

(4) Supported using V_{CCIO} at 3.0 V.

I/O Standard	Typical Applications	Standard Support
Differential SSTL-15 Class II	DDR3 SDRAM	—
Differential 1.8 V HSTL Class I	Clock interfaces	JESD8-6
Differential 1.8 V HSTL Class II	Clock interfaces	JESD8-6
Differential 1.5 V HSTL Class I	Clock interfaces	JESD8-6
Differential 1.5 V HSTL Class II	Clock interfaces	JESD8-6
Differential 1.2 V HSTL Class I	Clock interfaces	JESD8-16A
Differential 1.2 V HSTL Class II	Clock interfaces	JESD8-16A
LVDS	High-speed communications	ANSI/TIA/EIA-644
RSDS	Flat panel display	—
Mini-LVDS	Flat panel display	—
LVPECL	Video graphics and clock distribution	—
SSTL-15	DDR3 SDRAM	JESD79-3D
SSTL-135	DDR3L SDRAM	—
SSTL-125	DDR3U SDRAM	—
SSTL-12	RLDRAM 3	—
HSUL-12	LPDDR2 SDRAM	—
Differential SSTL-15	DDR3 SDRAM	JESD79-3D
Differential SSTL-135	DDR3L SDRAM	—
Differential SSTL-125	DDR3U SDRAM	—
Differential SSTL-12	RLDRAM 3	—
Differential HSUL-12	LPDDR2 SDRAM	—

I/O Standards Voltage Levels in Stratix V Devices

Table 5-2: Stratix V I/O Standards Voltage Levels

This table lists the typical power supplies for each supported I/O standards in Stratix V devices.

I/O Standard	V_{CCIO} (V)		V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
	Input[5]	Output			
3.3 V LVTTL/3.3 V LVCMOS	3.0/2.5	3.0	3.0	—	—

(5) Input buffers for the SSTL, HSTL, Differential SSTL, Differential HSTL, LVDS, RSDS, Mini-LVDS, LVPECL, HSUL, and Differential HSUL are powered by V_{CCPD}

I/O Standard	V_{CCIO} (V)		V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
	Input[5]	Output			
2.5 V LVCMOS	3.0/2.5	2.5	2.5	—	—
1.8 V LVCMOS	1.8/1.5	1.8	2.5	—	—
1.5 V LVCMOS	1.8/1.5	1.5	2.5	—	—
1.2 V LVCMOS	1.2	1.2	2.5	—	—
SSTL-2 Class I	V_{CCPD}	2.5	2.5	1.25	1.25
SSTL-2 Class II	V_{CCPD}	2.5	2.5	1.25	1.25
SSTL-18 Class I	V_{CCPD}	1.8	2.5	0.9	0.9
SSTL-18 Class II	V_{CCPD}	1.8	2.5	0.9	0.9
SSTL-15 Class I	V_{CCPD}	1.5	2.5	0.75	0.75
SSTL-15 Class II	V_{CCPD}	1.5	2.5	0.75	0.75
1.8 V HSTL Class I	V_{CCPD}	1.8	2.5	0.9	0.9
1.8 V HSTL Class II	V_{CCPD}	1.8	2.5	0.9	0.9
1.5 V HSTL Class I	V_{CCPD}	1.5	2.5	0.75	0.75
1.5 V HSTL Class II	V_{CCPD}	1.5	2.5	0.75	0.75
1.2 V HSTL Class I	V_{CCPD}	1.2	2.5	0.6	0.6
1.2 V HSTL Class II	V_{CCPD}	1.2	2.5	0.6	0.6
Differential SSTL-2 Class I	V_{CCPD}	2.5	2.5	—	1.25
Differential SSTL-2 Class II	V_{CCPD}	2.5	2.5	—	1.25
Differential SSTL-18 Class I	V_{CCPD}	1.8	2.5	—	0.9
Differential SSTL-18 Class II	V_{CCPD}	1.8	2.5	—	0.9
Differential SSTL-15 Class I	V_{CCPD}	1.5	2.5	—	0.75
Differential SSTL-15 Class II	V_{CCPD}	1.5	2.5	—	0.75
Differential 1.8 V HSTL Class I	V_{CCPD}	1.8	2.5	—	0.9
Differential 1.8 V HSTL Class II	V_{CCPD}	1.8	2.5	—	0.9

(5) Input buffers for the SSTL, HSTL, Differential SSTL, Differential HSTL, LVDS, RSDS, Mini-LVDS, LVPECL, HSUL, and Differential HSUL are powered by V_{CCPD}

I/O Standard	V_{CCIO} (V)		V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
	Input[5]	Output			
Differential 1.5 V HSTL Class I	V_{CCPD}	1.5	2.5	—	0.75
Differential 1.5 V HSTL Class II	V_{CCPD}	1.5	2.5	—	0.75
Differential 1.2 V HSTL Class I	V_{CCPD}	1.2	2.5	—	0.6
Differential 1.2 V HSTL Class II	V_{CCPD}	1.2	2.5	—	0.6
LVDS	V_{CCPD}	2.5	2.5	—	—
RSDS	V_{CCPD}	2.5	2.5	—	—
Mini-LVDS	V_{CCPD}	2.5	2.5	—	—
LVPECL (Differential clock input only)	V_{CCPD}	—	2.5	—	—
SSTL-15	V_{CCPD}	1.5	2.5	0.75	Typically does not require board termination
SSTL-135	V_{CCPD}	1.35	2.5	0.675	
SSTL-125	V_{CCPD}	1.25	2.5	0.625	
SSTL-12	V_{CCPD}	1.2	2.5	0.6	
HSUL-12	V_{CCPD}	1.2	2.5	0.6	
Differential SSTL-15	V_{CCPD}	1.5	2.5	—	Typically does not require board termination
Differential SSTL-135	V_{CCPD}	1.35	2.5	—	
Differential SSTL-125	V_{CCPD}	1.25	2.5	—	
Differential SSTL-12	V_{CCPD}	1.2	2.5	—	
Differential HSUL-12	V_{CCPD}	1.2	2.5	—	

The Stratix V I/O buffers support 3.3 V I/O standards. You can use them as transmitters or receivers in your system. The output high voltage (V_{OH}), output low voltage (V_{OL}), input high voltage (V_{IH}), and input low voltage (V_{IL}) levels meet the 3.3 V I/O standards specifications defined by EIA/JEDEC Standard JESD8-B with margin when the Stratix V V_{CCIO} voltage is powered by 3.0 V.

Related Information

Guideline: Observe Device Absolute Maximum Rating for 3.3 V Interfacing on page 5-8
Provides more information about the 3.3 V LVTTL/LVCMOS I/O standard supported in Stratix V devices.

(5) Input buffers for the SSTL, HSTL, Differential SSTL, Differential HSTL, LVDS, RSDS, Mini-LVDS, LVPECL, HSUL, and Differential HSUL are powered by V_{CCPD}

MultiVolt I/O Interface in Stratix V Devices

The MultiVolt I/O interface feature allows Stratix V devices in all packages to interface with systems of different supply voltages.

You can connect the V_{CCIO} pins to a 1.2, 1.25, 1.35, 1.5, 1.8, 2.5, or 3.0 V power supply, depending on the output requirements. The output levels are compatible with systems of the same voltage as the power supply. For example, when V_{CCIO} pins are connected to a 1.5 V power supply, the output levels are compatible with 1.5 V systems.

For LVDS applications:

- The LVDS I/O standard is not supported when V_{CCIO} is 3.0 V.
- The LVDS input operations are supported when V_{CCIO} is 1.2, 1.25, 1.35, 1.5, 1.8, or 2.5 V.
- The LVDS output operations are only supported when V_{CCIO} is 2.5 V.

Table 5-3: MultiVolt I/O Support in Stratix V Devices

V_{CCIO} (V)	V_{CCPD} (V)	Input Signal (V)	Output Signal (V)
1.2	2.5	1.2	1.2
1.25	2.5	1.25	1.25
1.35	2.5	1.35	1.35
1.5	2.5	1.5, 1.8	1.5
1.8	2.5	1.5, 1.8	1.8
2.5	2.5	2.5, 3.0, 3.3	2.5
3.0	3.0	2.5, 3.0, 3.3	3.0, 3.3

The pin current may be slightly higher than the default value. Verify that the V_{OL} maximum and V_{OH} minimum voltages of the driving device do not violate the applicable V_{IL} maximum and V_{IH} minimum voltage specifications of the Stratix V device.

The V_{CCPD} power pins must be connected to a 2.5 V or 3.0 V power supply. Using these power pins to supply the pre-driver power to the output buffers increases the performance of the output pins.

Note: If the input signal is 3.0 V or 3.3 V, Altera recommends that you use an external clamping diode on the I/O pins.

I/O Design Guidelines for Stratix V Devices

There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

Mixing Voltage-Referenced and Non-Voltage-Referenced I/O Standards

Each I/O bank can simultaneously support multiple I/O standards. The following sections provide guidelines for mixing non-voltage-referenced and voltage-referenced I/O standards in the devices.

(6) Single-ended I/O standard at this voltage is not supported in the Stratix V devices. This information highlights that multiple single-ended I/O standards are not compatible with V_{CCIO} at this voltage.

Non-Voltage-Referenced I/O Standards

Each Stratix V I/O bank has its own V_{CCIO} pins and supports only one V_{CCIO} of 1.2, 1.25, 1.35, 1.5, 1.8, 2.5, or 3.0 V. An I/O bank can simultaneously support any number of input signals with different I/O standard assignments if the I/O standards support the V_{CCIO} level and V_{CCPD} requirement of the I/O bank.

For output signals, a single I/O bank supports non-voltage-referenced output signals that drive at the same voltage as V_{CCIO}. Because an I/O bank can only have one V_{CCIO} value, it can only drive out the value for non-voltage-referenced signals.

For example, an I/O bank with a 2.5 V V_{CCIO} setting can support 2.5 V standard inputs and outputs, and 3.0 V LVCMOS inputs only.

Voltage-Referenced I/O Standards

To accommodate voltage-referenced I/O standards:

- Each Stratix V I/O bank supports multiple dedicated VREF pins feeding a common V_{REF} bus.
- Each bank can have only a single V_{CCIO} voltage level and a single voltage reference (V_{REF}) level.

An I/O bank featuring single-ended or differential standards can support different voltage-referenced standards if all voltage-referenced standards use the same V_{REF} setting.

For performance reasons, voltage-referenced input standards use their own V_{CCPD} level as the power source. This feature allows you to place voltage-referenced input signals in an I/O bank with a V_{CCIO} of 2.5 V or below. For example, you can place HSTL-15 input pins in an I/O bank with 2.5 V V_{CCIO}. However, the voltage-referenced input with R_T OCT enabled requires the V_{CCIO} of the I/O bank to match the voltage of the input standard. R_T OCT cannot be supported for the HSTL-15 I/O standard when V_{CCIO} is 2.5 V.

Voltage-referenced bidirectional and output signals must be the same as the V_{CCIO} voltage of the I/O bank. For example, you can place only SSTL-2 output pins in an I/O bank with a 2.5 V V_{CCIO}.

Mixing Voltage-Referenced and Non-Voltage Referenced I/O Standards

An I/O bank can support voltage-referenced and non-voltage-referenced pins by applying each of the rule sets individually.

Examples:

- An I/O bank can support SSTL-18 inputs and outputs, and 1.8 V inputs and outputs with a 1.8 V V_{CCIO} and a 0.9 V V_{REF}.
- An I/O bank can support 1.5 V standards, 1.8 V inputs (but not outputs), and HSTL and 1.5 V HSTL I/O standards with a 1.5 V V_{CCIO} and 0.75 V V_{REF}.

Guideline: Use the Same V_{CCPD} for All I/O Banks in a Group

One V_{CCPD} is shared in a group of I/O banks. If one I/O bank in a group uses 3.0 V V_{CCPD}, other I/O banks in the same group must also use 3.0 V V_{CCPD}.

The I/O banks with the same bank number form a group. For example, I/O banks 7A, 7B, 7C, and 7D form a group and share the same V_{CCPD}. This sharing is applicable to all I/O banks, with the following exceptions:

- I/O banks 3A and 3B form a group with one V_{CCPD}.
- I/O banks 3C, 3D, and 3E (if available) form another group with its own V_{CCPD}.

If you are using an output or bidirectional pin with the 3.3 V LVTTL or 3.3 V LVCMOS I/O standard, you must adhere to this restriction manually with location assignments.

Related Information

- **Modular I/O Banks for Stratix V E Devices** on page 5-10
- **Modular I/O Banks for Stratix V GX Devices** on page 5-11
- **Modular I/O Banks for Stratix V GS Devices** on page 5-14
- **Modular I/O Banks for Stratix V GT Devices** on page 5-15

Guideline: Observe Device Absolute Maximum Rating for 3.3 V Interfacing

To ensure device reliability and proper operation when you use the device for 3.3 V I/O interfacing, do not violate the absolute maximum ratings of the device. For more information about absolute maximum rating and maximum allowed overshoot during transitions, refer to the device datasheet.

Tip: Perform IBIS or SPICE simulations to make sure the overshoot and undershoot voltages are within the specifications.

Transmitter Application

If you use the Stratix V device as a transmitter, use slow slew-rate and series termination to limit the overshoot and undershoot at the I/O pins. Transmission line effects that cause large voltage deviations at the receiver are associated with an impedance mismatch between the driver and the transmission lines. By matching the impedance of the driver to the characteristic impedance of the transmission line, you can significantly reduce overshoot voltage. You can use a series termination resistor placed physically close to the driver to match the total driver impedance to the transmission line impedance.

Receiver Application

If you use the Stratix V device as a receiver, use an off-chip clamping diode to limit the overshoot and undershoot voltage at the I/O pins.

The 3.3 V I/O standard is supported using the bank supply voltage (V_{CCIO}) at 3.0 V and a V_{CCPD} voltage of 3.0 V. In this method, the clamping diode can sufficiently clamp overshoot voltage to within the DC and AC input voltage specifications. The clamped voltage is expressed as the sum of the V_{CCIO} and the diode forward voltage.

Related Information

Stratix V Device Datasheet

Guideline: Use PLL Integer Mode for LVDS Applications

For LVDS applications, you must use the phase-locked loops (PLLs) in integer PLL mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

I/O Banks in Stratix V Devices

All I/O banks in Stratix V devices contain true differential input and output buffers and dedicated circuitry to support differential I/O standards:

- The number of I/O banks in a particular device depends on the device density.
- Each I/O bank supports a high-performance external memory interface.
- The I/O pins are organized in pairs to support differential I/O standards.
- Each I/O pin pair can support both differential input and output buffers.

Figure 5-1: I/0 Banks for Stratix V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

Related Information

- **Modular I/O Banks for Stratix V E Devices** on page 5-10
- **Modular I/O Banks for Stratix V GX Devices** on page 5-11
- **Modular I/O Banks for Stratix V GS Devices** on page 5-14
- **Modular I/O Banks for Stratix V GT Devices** on page 5-15

I/O Banks Groups in Stratix V Devices

The I/O pins in Stratix V devices are arranged in groups called modular I/O banks:

- Modular I/O banks have independent power supplies that allow each bank to support different I/O standards.
- Each modular I/O bank can support multiple I/O standards that use the same V_{CCIO} and V_{CCPD} voltages.

Modular I/O Banks for Stratix V E Devices

Table 5-4: Modular I/O Banks for Stratix V E Devices

Member Code		E9		EB	
Package		H40	F45	H40	F45
Bank	3A	36	36	36	36
	3B	48	48	48	48
	3C	48	48	48	48
	3D	48	48	48	48
	3E	—	36	—	36
	4A	24	24	24	24
	4B	48	48	48	48
	4C	48	48	48	48
	4D	48	48	48	48
	4E	—	36	—	36
	7A	24	24	24	24
	7B	48	48	48	48
	7C	48	48	48	48
	7D	48	48	48	48
	7E	—	36	—	36
	8A	36	36	36	36
	8B	48	48	48	48
	8C	48	48	48	48
	8D	48	48	48	48
	8E	—	36	—	36
Total		696	840	696	840

Related Information

- I/O Banks in Stratix V Devices on page 5-9

- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-7
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Stratix V GX Devices

Table 5-5: Modular I/O Banks for Stratix V GX A3 and A4 Devices

Member Code		A3				A4		
Package		EH29	HF35	KF35	KF40	HF35	KF35	KF40
Bank	3A	36	36	36	36	36	36	36
	3B	48	48	48	48	48	48	48
	3C	—	—	—	48	48	—	48
	3D	24	24	24	48	24	24	48
	4A	24	24	24	24	24	24	24
	4B	—	48	48	48	48	48	48
	4C	—	—	—	48	48	—	48
	4D	24	36	36	48	24	36	48
	7A	24	24	24	24	24	24	24
	7B	—	48	48	48	48	48	48
	7C	48	48	48	48	48	48	48
	7D	36	36	36	48	36	36	48
	8A	24	24	24	36	24	24	36
	8B	—	—	—	48	—	—	48
	8C	48	—	—	48	48	—	48
	8D	24	36	36	48	24	36	48
Total		360	432	432	696	552	432	696

Table 5-6: Modular I/O Banks for Stratix V GX A5 and A7 Devices

Member Code		A5					A7				
Package		HF35	KF35	KF40	NF40	NF45	HF35	KF35	KF40	NF40	NF45
Bank	3A	36	36	36	36	36	36	36	36	36	36
	3B	48	48	48	48	48	48	48	48	48	48
	3C	48	—	48	48	48	48	—	48	48	48
	3D	24	24	48	24	48	24	24	48	24	48
	3E	—	—	—	—	36	—	—	—	—	36
	4A	24	24	24	24	24	24	24	24	24	24
	4B	48	48	48	48	48	48	48	48	48	48
	4C	48	—	48	48	48	48	—	48	48	48
	4D	24	36	48	24	48	24	36	48	24	48
	4E	—	—	—	—	36	—	—	—	—	36
	7A	24	24	24	24	24	24	24	24	24	24
	7B	48	48	48	48	48	48	48	48	48	48
	7C	48	48	48	48	48	48	48	48	48	48
	7D	36	36	48	48	48	36	36	48	48	48
	7E	—	—	—	—	36	—	—	—	—	36
	8A	24	24	36	36	36	24	24	36	36	36
	8B	—	—	48	—	48	—	—	48	—	48
	8C	48	—	48	48	48	48	—	48	48	48
	8D	24	36	48	48	48	24	36	48	48	48
	8E	—	—	—	—	36	—	—	—	—	36
Total		552	432	696	600	840	552	432	696	600	840

Table 5-7: Modular I/O Banks for Stratix V GX A9, AB, B5, B6, B9, and BB Devices

Member Code		A9		AB		B5		B6		B9	BB
Package		KH40	NF45	KH40	NF45	RF40	RF43	RF40	RF43	RH43	RH43
Bank	3A	36	36	36	36	36	36	36	36	36	36
	3B	48	48	48	48	48	48	48	48	48	48
	3C	48	48	48	48	—	48	—	48	48	48
	3D	48	48	48	48	—	36	—	36	36	36
	3E	—	36	—	36	—	—	—	—	—	—
	4A	24	24	24	24	48	48	48	48	48	48
	4B	48	48	48	48	48	48	48	48	48	48
	4C	48	48	48	48	36	36	36	36	36	36
	4D	48	48	48	48	—	—	—	—	—	—
	4E	—	36	—	36	—	—	—	—	—	—
	7A	24	24	24	24	48	48	48	48	48	48
	7B	48	48	48	48	48	48	48	48	48	48
	7C	48	48	48	48	36	36	36	36	36	36
	7D	48	48	48	48	—	—	—	—	—	—
	7E	—	36	—	36	—	—	—	—	—	—
	8A	36	36	36	36	36	36	36	36	36	36
	8B	48	48	48	48	48	48	48	48	48	48
	8C	48	48	48	48	—	48	—	48	48	48
	8D	48	48	48	48	—	36	—	36	36	36
	8E	—	36	—	36	—	—	—	—	—	—
Total		696	840	696	840	432	600	432	600	600	600

Related Information

- **I/O Banks in Stratix V Devices** on page 5-9
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-7
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Stratix V GS Devices

Table 5-8: Modular I/O Banks for Stratix V GS Devices

Member Code		D3		D4			D5		D6		D8	
Package		EF29	HF35	EF29	HF35	KF40	HF35	KF40	KF40	NF45	KF40	NF45
Bank	3A	36	36	36	36	36	36	36	36	36	36	36
	3B	48	48	48	48	48	48	48	48	48	48	48
	3C	—	—	—	—	48	48	48	48	48	48	48
	3D	24	24	24	24	48	24	48	48	48	48	48
	3E	—	—	—	—	—	—	—	—	36	—	36
	4A	24	24	24	24	24	24	24	24	24	24	24
	4B	—	48	—	48	48	48	48	48	48	48	48
	4C	—	—	—	—	48	48	48	48	48	48	48
	4D	24	24	24	24	48	24	48	48	48	48	48
	4E	—	—	—	—	—	—	—	—	36	—	36
	7A	24	24	24	24	24	24	24	24	24	24	24
	7B	—	24	—	24	48	48	48	48	48	48	48
	7C	48	48	48	48	48	48	48	48	48	48	48
	7D	36	36	36	36	48	36	48	48	48	48	48
	7E	—	—	—	—	—	—	—	—	36	—	36
	8A	24	24	24	24	36	24	36	36	36	36	36
	8B	—	—	—	—	48	—	48	48	48	48	48
	8C	48	48	48	48	48	48	48	48	48	48	48
	8D	24	24	24	24	48	24	48	48	48	48	48
	8E	—	—	—	—	—	—	—	—	36	—	36
Total		360	432	360	432	696	552	696	696	840	696	840

Related Information

- **I/O Banks in Stratix V Devices** on page 5-9
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-7
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Stratix V GT Devices

Table 5-9: Modular I/O Banks for Stratix V GT Devices

Member Code		C5	C7
Package		KF40	KF40
Bank	3A	36	36
	3B	48	48
	3C	48	48
	3D	24	24
	3E	—	—
	4A	24	24
	4B	48	48
	4C	48	48
	4D	24	24
	4E	—	—
	7A	24	24
	7B	48	48
	7C	48	48
	7D	48	48
	7E	—	—
	8A	36	36
	8B	—	—
	8C	48	48
	8D	48	48
	8E	—	—
Total		600	600

Related Information

- **I/O Banks in Stratix V Devices** on page 5-9
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-7
  Provides guidelines about V_{CCPD} and I/O banks groups.

I/O Element Structure in Stratix V Devices

The I/O elements (IOEs) in Stratix V devices contain a bidirectional I/O buffer and I/O registers to support a complete embedded bidirectional single data rate (SDR) or double data rate (DDR) transfer.

The IOEs are located in I/O blocks around the periphery of the Stratix V device.

I/O Buffer and Registers in Stratix V Devices

I/O registers are composed of the input path for handling data from the pin to the core, the output path for handling data from the core to the pin, and the output enable (OE) path for handling the OE signal to the output buffer. These registers allow faster source-synchronous register-to-register transfers and resynchronization.

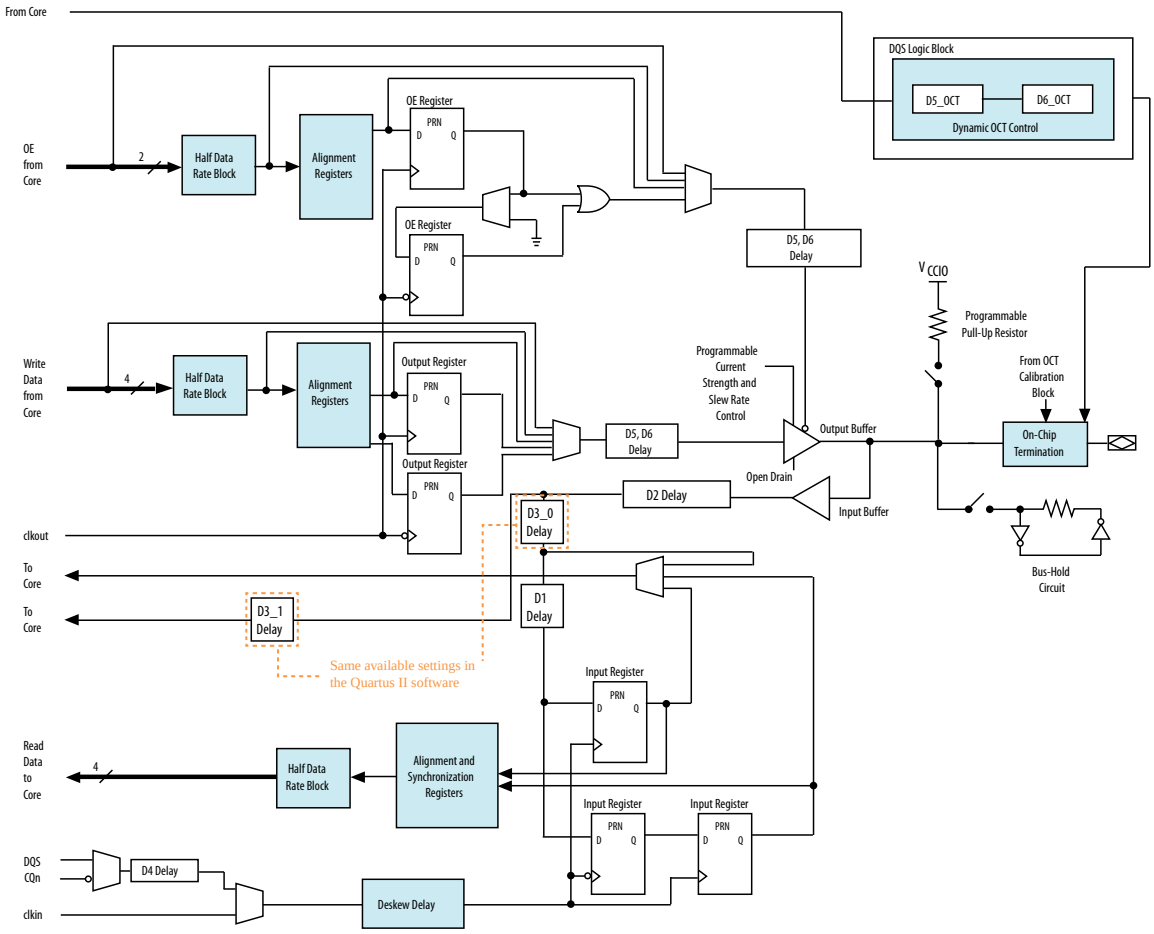
Table 5-10: Input and Output Paths in Stratix V Devices

This table summarizes the input and output path in the Stratix V devices.

Input Path	Output Path
Consists of: • DDR input registers • Alignment and synchronization registers • Half data rate blocks	Consists of: • Output or OE registers • Alignment registers • Half data rate blocks
You can bypass each block in the input path. The input path uses the deskew delay to adjust the input register clock delay across process, voltage, and temperature (PVT) variations.	You can bypass each block of the output and OE paths.

Figure 5-2: IOE Structure for Stratix V Devices

This figure shows the Stratix V FPGA IOE structure. In the figure, one dynamic on-chip termination (OCT) control is available for each DQ/DQS group.



External Memory Interfaces

In addition to the I/O registers in each IOE, Stratix V devices also have dedicated registers and phase-shift circuitry on all I/O banks to interface with external memory. Stratix V devices support I/O standards such as SSTL-12, SSTL-15, SSTL-125, SSTL-135, and HSUL-12.

High-Speed Differential I/O with DPA Support

To support high-speed differential I/O, Stratix V devices contain the following dedicated circuitries:

- Differential I/O buffer
- Transmitter serializer
- Receiver deserializer
- Data realignment
- DPA
- Synchronizer (FIFO buffer)
- Phase-locked loops (PLLs)

Programmable IOE Features in Stratix V Devices

Table 5-11: Summary of Supported Stratix V Programmable IOE Features and Settings

Feature	Setting	Condition
Slew Rate Control	0 (Slow), 1 (Fast). Default is 1.	Disabled if you use the R_S OCT feature.
I/O Delay	Refer to the device datasheet.	—
Open-Drain Output	On, Off (default)	—
Bus-Hold	On, Off (default)	Disabled if you use the pull-up resistor feature.
Pull-up Resistor	On, Off (default)	Disabled if you use the bus-hold feature.
Pre-Emphasis	0 (disabled), 1 (enabled). Default is 1.	—
Differential Output Voltage	0 (low), 1 (medium low), 2 (medium high), 3 (high). Default is 1.	—

Related Information

- **Stratix V Device Datasheet**
- **Programmable Current Strength** on page 5-18
- **Programmable Output Slew-Rate Control** on page 5-19
- **Programmable IOE Delay** on page 5-20
- **Programmable Output Buffer Delay** on page 5-20
- **Programmable Pre-Emphasis** on page 5-20
- **Programmable Differential Output Voltage** on page 5-21
- **Stratix V Device Datasheet**

Programmable Current Strength

You can use the programmable current strength to mitigate the effects of high signal attenuation that is caused by a long transmission line or a legacy backplane.

Table 5-12: Programmable Current Strength Settings for Stratix V Devices

The output buffer for each Stratix V device I/O pin has a programmable current strength control for the I/O standards listed in this table.

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA) (Default setting in bold)
3.3-V LVTTL	16, **12**, 8, 4
3.3-V LVCMOS	16, **12**, 8, 4

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA) (Default setting in bold)
2.5-V LVCMOS	16, **12**, 8, 4
1.8-V LVCMOS	**12**, 10, 8, 6, 4, 2
1.5-V LVCMOS	**12**, 10, 8, 6, 4, 2
1.2-V LVCMOS	**8**, 6, 4, 2
SSTL-2 Class I	12, 10, **8**
SSTL-2 Class II	**16**
SSTL-18 Class I	12, 10, **8**, 6, 4
SSTL-18 Class II	**16**, 8
SSTL-15 Class I	12, 10, **8**, 6, 4
SSTL-15 Class II	**16**, 8
1.8-V HSTL Class I	12, 10, **8**, 6, 4
1.8-V HSTL Class II	**16**
1.5-V HSTL Class I	12, 10, **8**, 6, 4
1.5-V HSTL Class II	**16**
1.2-V HSTL Class I	12, 10, **8**, 6, 4
1.2-V HSTL Class II	**16**

The 3.3 V LVTTL and 3.3 V LVCMOS I/O standards are supported using V_{CCIO} and V_{CCPD} at 3.0 V.

Note: Altera recommends that you perform IBIS or SPICE simulations to determine the best current strength setting for your specific application.

Related Information

Programmable IOE Features in Stratix V Devices on page 5-18

Programmable Output Slew-Rate Control

Programmable output slew-rate is available for single-ended I/O standards and emulated LVDS output standards.

The programmable output slew-rate control in the output buffer of each regular- and dual-function I/O pin allows you to configure the following:

- Fast slew-rate—provides high-speed transitions for high-performance systems. Fast slew rates improve the available timing margin in memory-interface applications or when the output pin has high-capacitive loading.
- Slow slew-rate—reduces system noise and crosstalk but adds a nominal delay to the rising and falling edges.

You can specify the slew-rate on a pin-by-pin basis because each I/O pin contains a slew-rate control.

Note: Altera recommends that you perform IBIS or SPICE simulations to determine the best slew rate setting for your specific application.

Related Information

Programmable IOE Features in Stratix V Devices on page 5-18

Programmable IOE Delay

You can activate the programmable IOE delays to ensure zero hold times, minimize setup times, or increase clock-to-output times. This feature helps read and write timing margins because it minimizes the uncertainties between signals in the bus.

Each single-ended and differential I/O pin can have a different input delay from pin-to-input register or a delay from output register-to-output pin values to ensure that the signals within a bus have the same delay going into or out of the device.

For more information about the programmable IOE delay specifications, refer to the device datasheet.

Related Information

- **Stratix V Device Datasheet**
- **Programmable IOE Features in Stratix V Devices** on page 5-18
- **Stratix V Device Datasheet**

Programmable Output Buffer Delay

The delay chains are built inside the single-ended output buffer. There are four levels of output buffer delay settings. By default, there is no delay.

The delay chains can independently control the rising and falling edge delays of the output buffer, allowing you to:

- Adjust the output-buffer duty cycle
- Compensate channel-to-channel skew
- Reduce simultaneous switching output (SSO) noise by deliberately introducing channel-to-channel skew
- Improve high-speed memory-interface timing margins

For more information about the programmable output buffer delay specifications, refer to the device datasheet.

Related Information

- **Stratix V Device Datasheet**
- **Programmable IOE Features in Stratix V Devices** on page 5-18
- **Stratix V Device Datasheet**

Programmable Pre-Emphasis

The V_{OD} setting and the output impedance of the driver set the output current limit of a high-speed transmission signal. At a high frequency, the slew rate may not be fast enough to reach the full V_{OD} level before the next edge, producing pattern-dependent jitter. With pre-emphasis, the output current is boosted momentarily during switching to increase the output slew rate.

Pre-emphasis increases the amplitude of the high-frequency component of the output signal, and thus helps to compensate for the frequency-dependent attenuation along the transmission line. The overshoot introduced by the extra current happens only during a change of state switching to increase the output

slew rate and does not ring, unlike the overshoot caused by signal reflection. The amount of pre-emphasis required depends on the attenuation of the high-frequency component along the transmission line.

Figure 5-3: Programmable Pre-Emphasis

This figure shows the LVDS output with pre-emphasis.

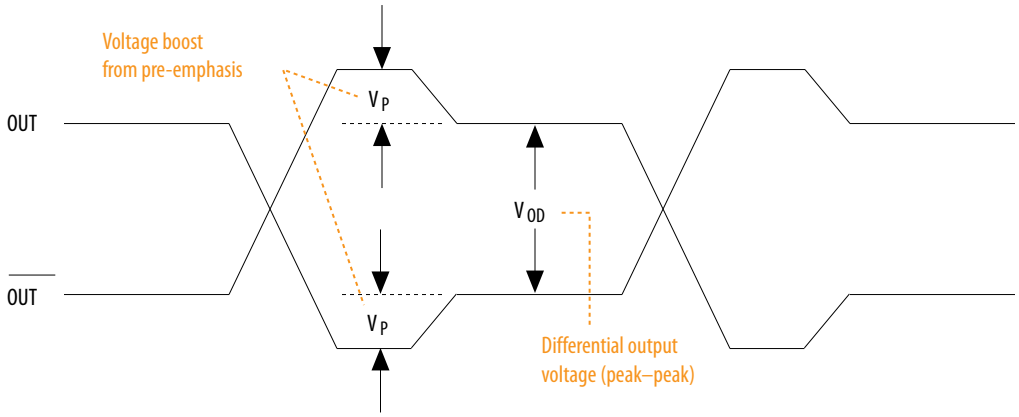


Table 5-13: Quartus II Software Assignment Editor—Programmable Pre-Emphasis

This table lists the assignment name for programmable pre-emphasis and its possible values in the Quartus II software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Pre-emphasis
Allowed values	0 (disabled), 1 (enabled). Default is 1.

Related Information

Programmable IOE Features in Stratix V Devices on page 5-18

Programmable Differential Output Voltage

The programmable V_{OD} settings allow you to adjust the output eye opening to optimize the trace length and power consumption. A higher V_{OD} swing improves voltage margins at the receiver end, and a smaller V_{OD} swing reduces power consumption. You can statically adjust the V_{OD} of the differential signal by changing the V_{OD} settings in the Quartus II software Assignment Editor.

Figure 5-4: Differential V_{OD}

This figure shows the V_{OD} of the differential LVDS output.

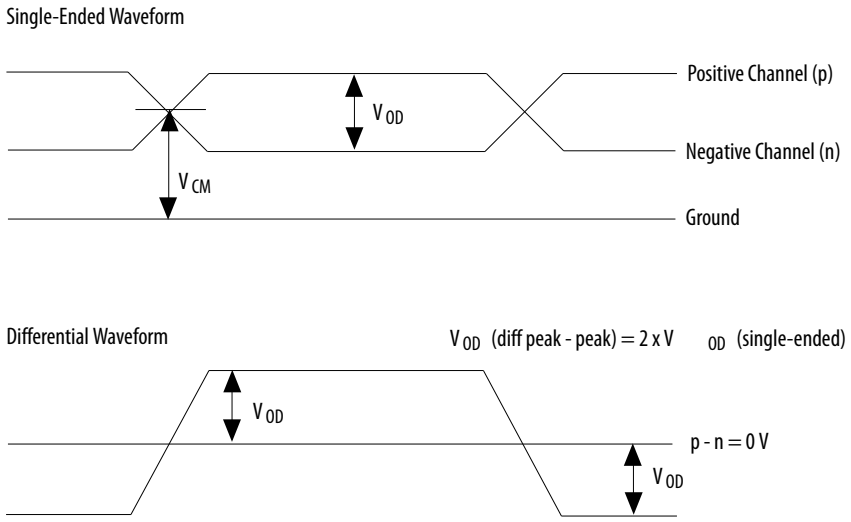


Table 5-14: Quartus II Software Assignment Editor—Programmable V_{OD}

This table lists the assignment name for programmable V_{OD} and its possible values in the Quartus II software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Differential Output Voltage (V_{OD})
Allowed values	0 (low), 1 (medium low), 2 (medium high), 3 (high). Default is 1.

Related Information

Programmable IOE Features in Stratix V Devices on page 5-18

Open-Drain Output

The optional open-drain output for each I/O pin is equivalent to an open collector output. If it is configured as an open drain, the logic value of the output is either high-Z or logic low.

You can attach several open-drain output to a wire. This connection type is like a logical OR function and is commonly called an active-low wired-OR circuit. If at least one of the outputs is in logic 0 state (active), the circuit sinks the current and brings the line to low voltage.

You can use open-drain output if you are connecting multiple devices to a bus. For example, you can use the open-drain output for system-level control signals that can be asserted by any device or as an interrupt.

You can enable the open-drain output assignment using one these methods:

- Design the tristate buffer using OPNDRN primitive.
- Turn on the **Auto Open-Drain Pins** option in the Quartus II software.

Although you can design open-drain output without enabling the option assignment, you will not be using the open-drain output feature of the I/O buffer. The open-drain output feature in the I/O buffer provides you the best propagation delay from OE to output.

Bus-Hold Circuitry

Each I/O pin provides an optional bus-hold feature that is active only after configuration. When the device enters user mode, the bus-hold circuit captures the value that is present on the pin by the end of the configuration.

The bus-hold circuitry uses a resistor with a nominal resistance (R_{BH}), approximately 7 kΩ, to weakly pull the signal level to the last-driven state of the pin. The bus-hold circuitry holds this pin state until the next input signal is present. Because of this, you do not require an external pull-up or pull-down resistor to hold a signal level when the bus is tri-stated.

For each I/O pin, you can individually specify that the bus-hold circuitry pulls non-driven pins away from the input threshold voltage—where noise can cause unintended high-frequency switching. To prevent over-driving signals, the bus-hold circuitry drives the voltage level of the I/O pin lower than the V_{CCIO} level.

If you enable the bus-hold feature, you cannot use the programmable pull-up option. To configure the I/O pin for differential signals, disable the bus-hold feature.

Pull-up Resistor

Each I/O pin provides an optional programmable pull-up resistor during user mode. The pull-up resistor, typically 25 kΩ, weakly holds the I/O to the V_{CCIO} level. If you enable this option, you cannot use the bus-hold feature.

The Stratix V device supports programmable pull-up resistors only on user I/O pins.

For dedicated configuration pins or JTAG pins with internal pull-up resistors, these resistor values are not programmable. You can find more information related to the internal pull-up values for dedicated configuration pins or JTAG pins in the Stratix V Pin Connection Guidelines.

On-Chip I/O Termination in Stratix V Devices

Dynamic R_S and R_T OCT provides I/O impedance matching and termination capabilities. OCT maintains signal quality, saves board space, and reduces external component costs.

The Stratix V devices support OCT in all I/O banks.

Table 5-15: OCT Schemes Supported in Stratix V Devices

Direction	OCT Schemes
Output	R_S OCT with calibration
	R_S OCT without calibration
Input	R_T OCT with calibration
	R_D OCT (differential LVDS I/O standard only)

Direction	OCT Schemes
Bidirectional	Dynamic R_S OCT and R_T OCT

R_S OCT without Calibration in Stratix V Devices

The Stratix V devices support R_S OCT for single-ended I/O standards. R_S OCT without calibration is supported on output only.

Table 5-16: Selectable I/O Standards for R_S OCT Without Calibration

This table lists the output termination settings for uncalibrated OCT on different I/O standards.

I/O Standard	Uncalibrated OCT (Output)
	R_S (Ω)
3.3 V LVTTL/3.3 V LVCMOS	25/50
2.5 V LVCMOS	25/50
1.8 V LVCMOS	25/50
1.5 V LVCMOS	25/50
1.2 V LVCMOS	25/50
SSTL-2 Class I	50
SSTL-2 Class II	25
SSTL-18 Class I	50
SSTL-18 Class II	25
SSTL-15 Class I	50
SSTL-15 Class II	25
1.8 V HSTL Class I	50
1.8 V HSTL Class II	25
1.5 V HSTL Class I	50
1.5 V HSTL Class II	25
1.2 V HSTL Class I	50
1.2 V HSTL Class II	25
Differential SSTL-2 Class I	50
Differential SSTL-2 Class II	25
Differential SSTL-18 Class I	50
Differential SSTL-18 Class II	25
Differential SSTL-15 Class I	50
Differential SSTL-15 Class II	25
Differential 1.8 V HSTL Class I	50
Differential 1.8 V HSTL Class II	25

I/O Standard	Uncalibrated OCT (Output)
	R_S (Ω)
Differential 1.5 V HSTL Class I	50
Differential 1.5 V HSTL Class II	25
Differential 1.2 V HSTL Class I	50
Differential 1.2 V HSTL Class II	25
SSTL-15	25, 34, 40, 50
SSTL-135	34, 40
SSTL-125	34, 40
SSTL-12	40, 60, 240
HSUL-12	34.3, 40, 48, 60, 80

The following list specifies the default settings for R_S OCT without calibration in the Quartus II software:

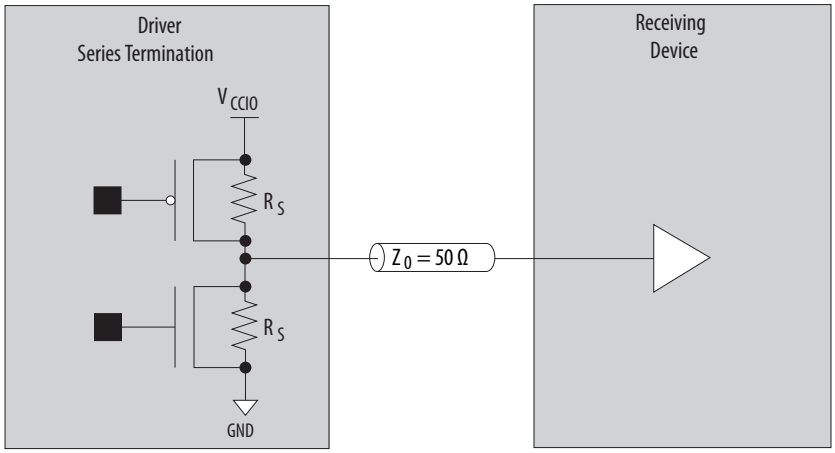
- For all non-voltage-referenced, HSTL Class I, and SSTL Class I I/O standards—50 Ω.
- For HSTL Class II and SSTL Class II I/O standards—25 Ω.

Driver-impedance matching provides the I/O driver with controlled output impedance that closely matches the impedance of the transmission line. As a result, you can significantly reduce signal reflections on PCB traces.

If you select matching impedance, current strength is no longer selectable.

Figure 5-5: R_S OCT Without Calibration

This figure shows the R_S as the intrinsic impedance of the output transistors. Typical R_S values are 25 Ω and 50 Ω.



To use OCT for the SSTL Class I I/O standard, you must select the 50 Ω R_S OCT setting, thus eliminating the external 25 Ω R_S (to match the 50 Ω transmission line). For the SSTL Class II I/O standard, you must select the 25 Ω R_S OCT setting (to match the 50 Ω transmission line and the near-end external 50 Ω pull-up to V_{TT}).

R_S OCT with Calibration in Stratix V Devices

The Stratix V devices support R_S OCT with calibration in all banks.

Table 5-17: Selectable I/O Standards for R_S OCT With Calibration

This table lists the output termination settings for calibrated OCT on different I/O standards.

I/O Standard	Calibrated OCT (Output)	
	R_S (Ω)	RZQ (Ω)
3.3 V LVTTL/3.3 V LVCMOS	25/50	100
2.5 V LVCMOS	25/50	100
1.8 V LVCMOS	25/50	100
1.5 V LVCMOS	25/50	100
1.2 V LVCMOS	25/50	100
SSTL-2 Class I	50	100
SSTL-2 Class II	25	100
SSTL-18 Class I	50	100
SSTL-18 Class II	25	100
SSTL-15 Class I	50	100
SSTL-15 Class II	25	100
1.8 V HSTL Class I	50	100
1.8 V HSTL Class II	25	100
1.5 V HSTL Class I	50	100
1.5 V HSTL Class II	25	100
1.2 V HSTL Class I	50	100
1.2 V HSTL Class II	25	100
Differential SSTL-2 Class I	50	100
Differential SSTL-2 Class II	25	100
Differential SSTL-18 Class I	50	100
Differential SSTL-18 Class II	25	100
Differential SSTL-15 Class I	50	100
Differential SSTL-15 Class II	25	100
Differential 1.8 V HSTL Class I	50	100
Differential 1.8 V HSTL Class II	25	100
Differential 1.5 V HSTL Class I	50	100
Differential 1.5 V HSTL Class II	25	100
Differential 1.2 V HSTL Class I	50	100

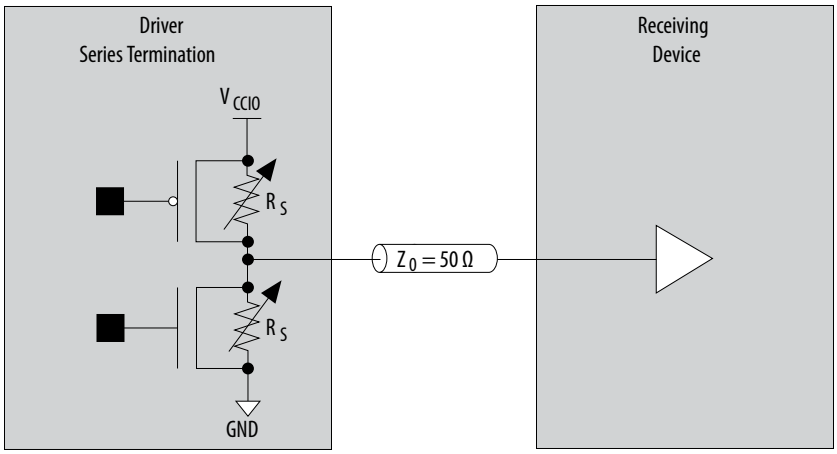
I/O Standard	Calibrated OCT (Output)	
	R_S (Ω)	RZQ (Ω)
Differential 1.2 V HSTL Class II	25	100
SSTL-15	25, 50	100
	34, 40	240
SSTL-135	34, 40	240
SSTL-125	34, 40	240
SSTL-12	40, 60, 240	240
HSUL-12	34, 40, 48, 60, 80	240
Differential SSTL-15	25, 50	100
	34, 40	240
Differential SSTL-135	34, 40	240
Differential SSTL-125	34, 40	240
Differential SSTL-12	40, 60, 240	240
Differential HSUL-12	34, 40, 48, 60, 80	240

The R_S OCT calibration circuit compares the total impedance of the I/O buffer to the external reference resistor connected to the RZQ pin and dynamically enables or disables the transistors until they match.

Calibration occurs at the end of device configuration. When the calibration circuit finds the correct impedance, the circuit powers down and stops changing the characteristics of the drivers.

Figure 5-6: R_S OCT with Calibration

This figure shows the R_S as the intrinsic impedance of the output transistors.



R_T OCT with Calibration in Stratix V Devices

The Stratix V devices support R_T OCT with calibration in all banks. R_T OCT with calibration is available only for configuration of input and bidirectional pins. Output pin configurations do not support R_T OCT

with calibration. If you use R_T OCT, the V_{CCIO} of the bank must match the I/O standard of the pin where you enable the R_T OCT.

Table 5-18: Selectable I/O Standards for R_T OCT With Calibration

This table lists the input termination settings for calibrated OCT on different I/O standards.

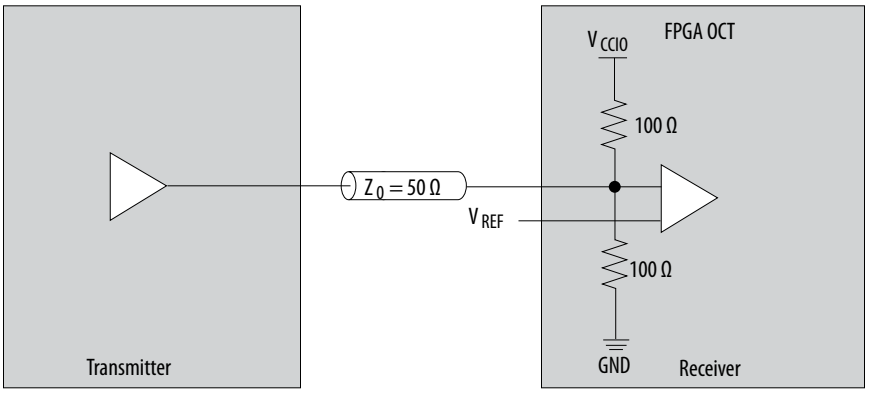
I/O Standard	Calibrated OCT (Input)	
	R_T (Ω)	RZQ (Ω)
SSTL-2 Class I	50	100
SSTL-2 Class II	50	100
SSTL-18 Class I	50	100
SSTL-18 Class II	50	100
SSTL-15 Class I	50	100
SSTL-15 Class II	50	100
1.8 V HSTL Class I	50	100
1.8 V HSTL Class II	50	100
1.5 V HSTL Class I	50	100
1.5 V HSTL Class II	50	100
1.2 V HSTL Class I	50	100
1.2 V HSTL Class II	50	100
Differential SSTL-2 Class I	50	100
Differential SSTL-2 Class II	50	100
Differential SSTL-18 Class I	50	100
Differential SSTL-18 Class II	50	100
Differential SSTL-15 Class I	50	100
Differential SSTL-15 Class II	50	100
Differential 1.8 V HSTL Class I	50	100
Differential 1.8 V HSTL Class II	50	100
Differential 1.5 V HSTL Class I	50	100
Differential 1.5 V HSTL Class II	50	100
Differential 1.2 V HSTL Class I	50	100
Differential 1.2 V HSTL Class II	50	100
SSTL-15	20, 30, 40, 60,120	240
SSTL-135	20, 30, 40, 60, 120	240
SSTL-125	20, 30, 40, 60, 120	240
SSTL-12	60, 120	240
HSUL-12	34, 40, 48, 60, 80	240

I/O Standard	Calibrated OCT (Input)	
	R_T (Ω)	RZQ (Ω)
Differential SSTL-15	20, 30, 40, 60,120	240
Differential SSTL-135	20, 30, 40, 60, 120	240
Differential SSTL-125	20, 30, 40, 60, 120	240
Differential SSTL-12	60, 120	240
Differential HSUL-12	34, 40, 48, 60, 80	240

The R_T OCT calibration circuit compares the total impedance of the I/O buffer to the external resistor connected to the RZQ pin. The circuit dynamically enables or disables the transistors until the total impedance of the I/O buffer matches the external resistor.

Calibration occurs at the end of the device configuration. When the calibration circuit finds the correct impedance, the circuit powers down and stops changing the characteristics of the drivers.

Figure 5-7: R_T OCT with Calibration



Dynamic OCT in Stratix V Devices

Dynamic OCT is useful for terminating a high-performance bidirectional path by optimizing the signal integrity depending on the direction of the data. Dynamic OCT also helps save power because device termination is internal—termination switches on only during input operation and thus draw less static power.

Note: If you use the HSUL-12, SSTL-12, SSTL-15, SSTL-135, and SSTL-125 I/O standards with the DDR3 memory interface, Altera recommends that you use dynamic OCT with these I/O standards to save board space and cost. Dynamic OCT reduces the number of external termination resistors used.

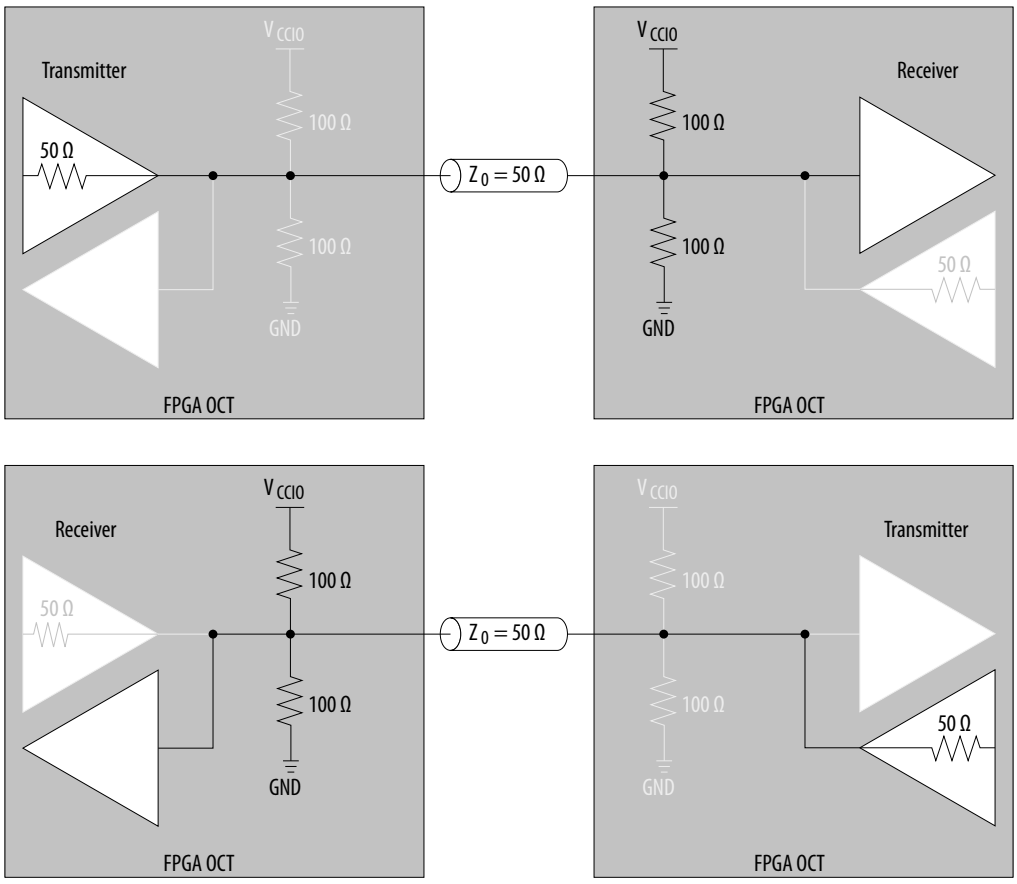
Table 5-19: Dynamic OCT Based on Bidirectional I/O

Dynamic R_T OCT or R_S OCT is enabled or disabled based on whether the bidirectional I/O acts as a receiver or driver.

Dynamic OCT	Bidirectional I/O	State
Dynamic R_T OCT	Acts as a receiver	Enabled
	Acts as a driver	Disabled

Dynamic OCT	Bidirectional I/O	State
Dynamic R_S OCT	Acts as a receiver	Disabled
	Acts as a driver	Enabled

Figure 5-8: Dynamic R_T OCT in Stratix V Devices



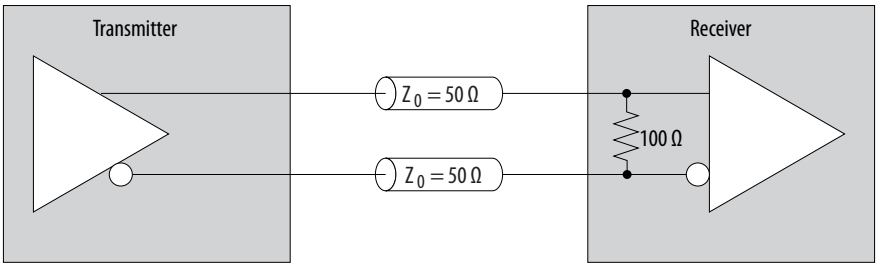
LVDS Input R_D OCT in Stratix V Devices

The Stratix V devices support R_D OCT in all I/O banks.

You can only use R_D OCT if you set the V_{CCPD} to 2.5 V.

Figure 5-9: Differential Input OCT

The Stratix V devices support OCT for differential LVDS input buffers with a nominal resistance value of 100 Ω, as shown in this figure.



OCT Calibration Block in Stratix V Devices

You can calibrate the OCT using any of the available four to eight OCT calibration blocks, depending on the device density. Each calibration block contains one RZQ pin.

You can use R_S and R_T OCT in the same I/O bank for different I/O standards if the I/O standards use the same V_{CCIO} supply voltage. You cannot configure the R_S OCT and the programmable current strength for the same I/O buffer.

The OCT calibration process uses the RZQ pin that is available in every calibration block in a given I/O bank for series- and parallel-calibrated termination:

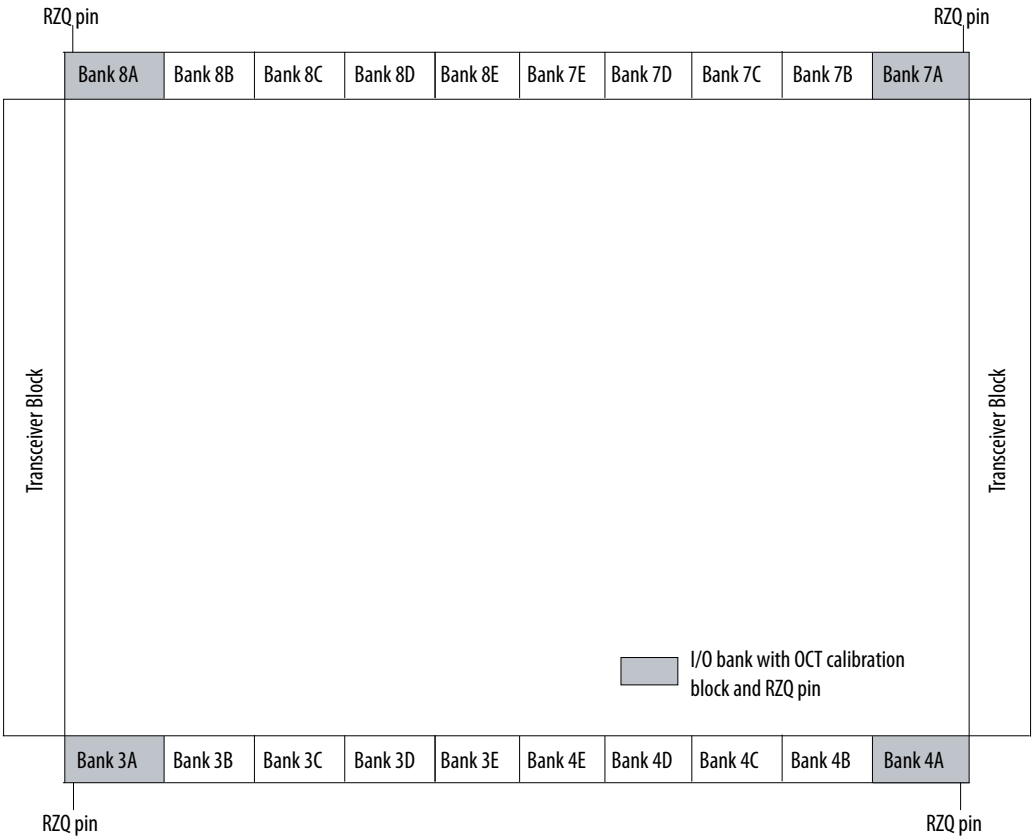
- Connect the RZQ pin to GND through an external 100 Ω or 240 Ω resistor (depending on the R_S or R_T OCT value).
- The RZQ pin shares the same V_{CCIO} supply voltage with the I/O bank where the pin is located.
- The RZQ pin is a dual-purpose I/O pin and functions as a general purpose I/O pin if you do not use the calibration circuit.

Stratix V devices support calibrated R_S and calibrated R_T OCT on all I/O pins except for dedicated configuration pins.

Calibration Block Locations in Stratix V Devices

Figure 5-10: OCT Calibration Block and RZQ Pin Location

This figure shows the location of I/O banks with OCT calibration blocks and RZQ pins in the Stratix V device. This figure represents the top view of the silicon die that corresponds to a reverse view of the device package and illustrates the highest density device in the device family.



Sharing an OCT Calibration Block on Multiple I/O Banks

An OCT calibration block has the same V_{CCIO} as the I/O bank that contains the block. All I/O banks with the same V_{CCIO} can share one OCT calibration block, even if that particular I/O bank has an OCT calibration block.

I/O banks that do not have calibration blocks share the calibration blocks in the I/O banks that have calibration blocks.

All I/O banks support OCT calibration with different V_{CCIO} voltage standards, up to the number of available OCT calibration blocks.

You can configure the I/O banks to receive calibration codes from any OCT calibration block with the same V_{CCIO}. If a group of I/O banks has the same V_{CCIO} voltage, you can use one OCT calibration block to calibrate the group of I/O banks placed around the periphery.

Related Information

- **OCT Calibration Block Sharing Example** on page 5-33

- **Dynamic Calibrated On-Chip Termination (ALTOCT) Megafunction User Guide**
  Provides more information about the OCT calibration block.

OCT Calibration Block Sharing Example

Figure 5-11: Example of Calibrating Multiple I/O Banks with One Shared OCT Calibration Block

As an example, this figure shows a group of I/O banks that has the same V_{CCIO} voltage. The figure does not show transceiver calibration blocks. This figure represents the top view of the silicon die that corresponds to a reverse view of the device package and illustrates the highest density device in the device family.

Because banks 3B, 4C, and 7B have the same V_{CCIO} as bank 7A, you can calibrate all four I/O banks (3B, 4C, 7A, and 7B) with the OCT calibration block (CB7) located in bank 7A.

To enable this calibration, serially shift out the R_S OCT calibration codes from the OCT calibration block in bank 7A to the I/O banks around the periphery.

Related Information

- **Sharing an OCT Calibration Block on Multiple I/O Banks** on page 5-32
- **Dynamic Calibrated On-Chip Termination (ALTOCT) Megafunction User Guide**
  Provides more information about the OCT calibration block.

OCT Calibration in Power-Up Mode

In power-up mode, OCT calibration is automatically performed at power up. Calibration codes are shifted to selected I/O buffers before transitioning to user mode.

OCT Calibration in User Mode

In user mode, the OCTUSRCLK, ENAOCT, nCLRUSR, and ENASER signals are used to calibrate and serially transfer calibration codes from each OCT calibration block to any I/O.

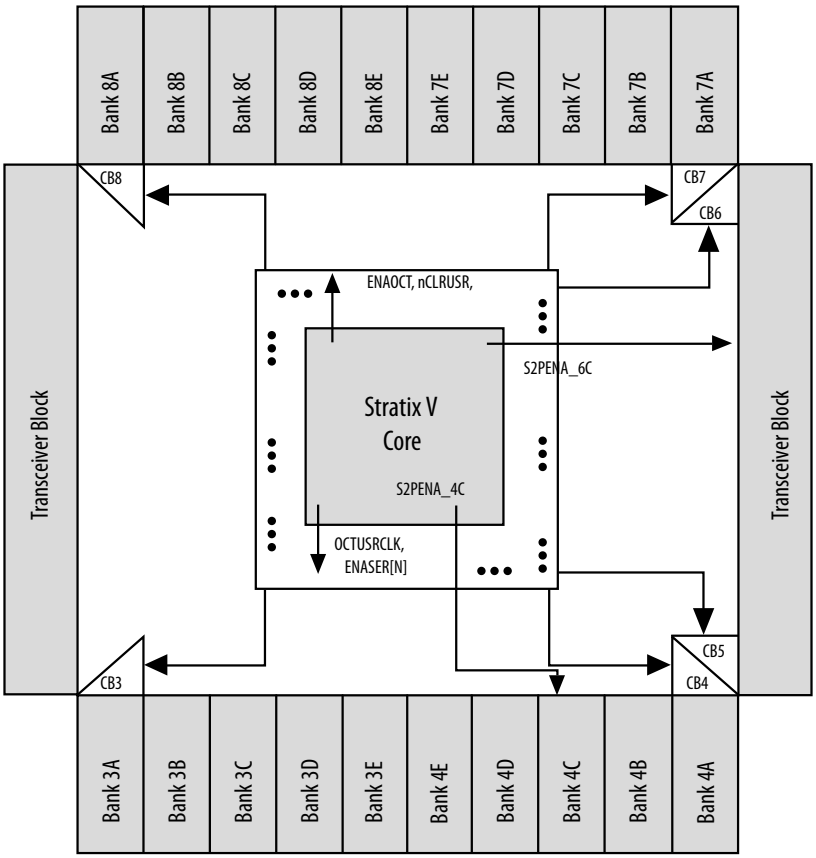
Table 5-20: OCT Calibration Block Ports for User Control

This table lists the user-controlled calibration block signal names and their descriptions

Signal Name	Description
OCTUSRCLK	Clock for OCT block.
ENAOCT	Enable OCT Calibration (generated by user IP).
ENASER[7..0]	• ENOCT is 0—each signal enables the OCT serializer for the corresponding OCT calibration block. • ENAOCT is 1—each signal enables OCT calibration for the corresponding OCT calibration block.
S2PENA_*bank#*	Serial-to-parallel load enable per I/O bank.
nCLRUSR	Clear user.

Figure 5-12: Signals Used for User Mode Calibration

This figure shows the flow of the user signal.

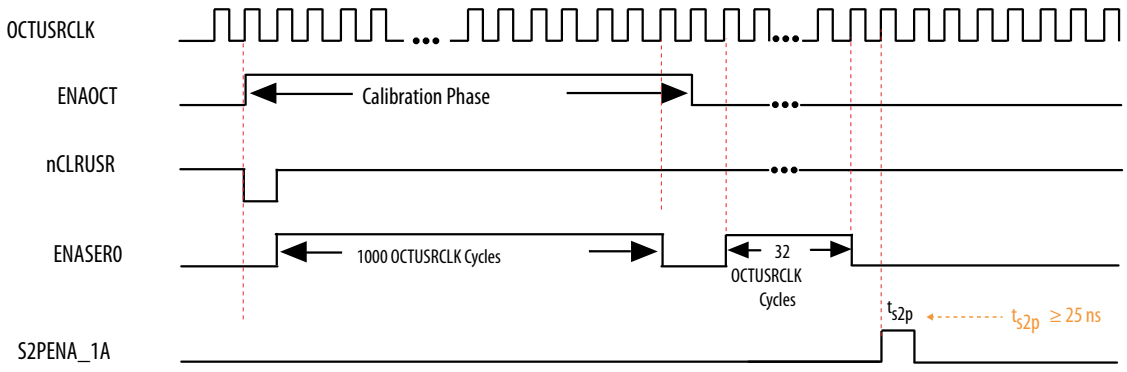


When ENAOCT is 1, all OCT calibration blocks are in calibration mode. When ENAOCT is 0, all OCT calibration blocks are in serial data transfer mode. The OCTUSRCLK clock frequency must be 20 MHz or less.

Note: You must generate all user signals on the rising edge of the OCTUSRCLK signal.

Figure 5-13: OCT User Mode Signal—Timing Waveform for One OCT Block

This figure shows the user mode signal-timing waveforms.



OCT Calibration

To calibrate OCT block *N* (where *N* is a calibration block number), you must assert ENAOCT one cycle before asserting ENASER*N*. You must also set nCLRUSR low for one OCTUSRCLK cycle before the ENASER*N* signal is asserted. Assert the ENASER*N* signals for 1,000 OCTUSRCLK cycles to perform R_S OCT and R_T OCT calibration. You can deassert ENAOCT one clock cycle after the last ENASER is deasserted.

Serial Data Transfer

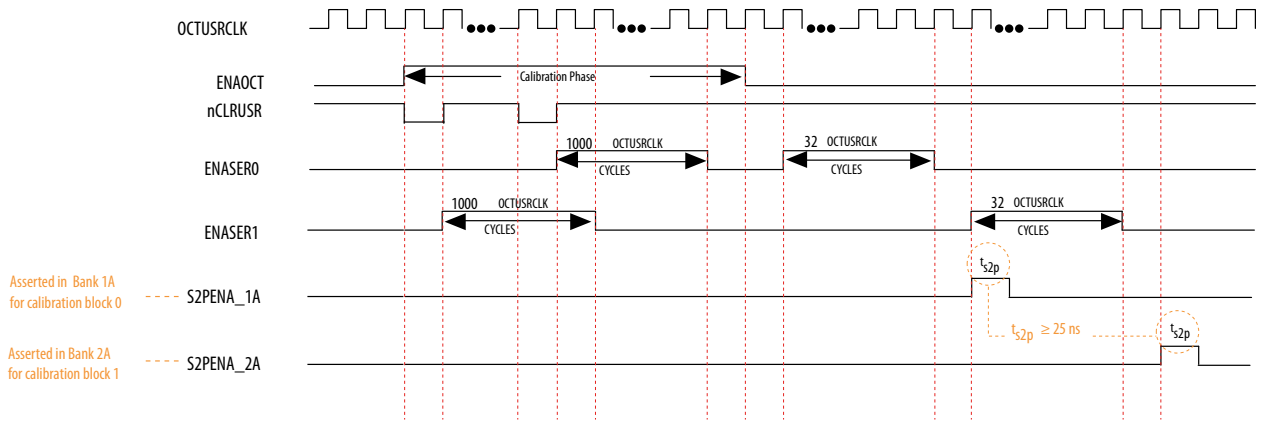
After you complete calibration, you must serially shift out the 32 bit OCT calibration codes (16 bit R_S OCT and 16 bit R_T OCT) from each OCT calibration block to the corresponding I/O buffers. Only one OCT calibration block can send out the codes at any time by asserting only one ENASER*N* signal at a time. After you deassert ENAOCT, wait at least one OCTUSRCLK cycle to enable any ENASER*N* signal to begin serial transfer. To shift the 32 bit code from the OCT calibration block *N*, you must assert ENASER*N* for exactly 32 OCTUSRCLK cycles. Between two consecutive asserted ENASER signals, there must be at least one OCTUSRCLK cycle gap, as shown in the preceding figure.

After calibrated codes are shifted in serially to each I/O bank, the calibrated codes must be converted from serial to parallel format before being used in the I/O buffers. The preceding figure shows the S2PENA signals that can be asserted at any time to update the calibration codes in each I/O bank. All I/O banks that received the codes from the same OCT calibration block can have S2PENA asserted at the same time, or at a different time, even while another OCT calibration block is calibrating and serially shifting codes. The S2PENA signal is asserted one OCTUSRCLK cycle after ENASER is deasserted for at least 25 ns. You cannot use I/Os for transmitting or receiving data when their S2PENA is asserted for parallel codes transfer.

Example of Using Multiple OCT Calibration Blocks

Figure 5-14: OCT User-Mode Signal Timing Waveform for Two OCT Blocks

This figure shows a signal timing waveform for two OCT calibration blocks doing R_S and R_T calibration.



Calibration blocks can start calibrating at different times by asserting the ENASER signals at different times. ENAOCT must remain asserted while any calibration is ongoing. You must set nCLRUSR low for one OCTUSRCLK cycle before each ENASER*N* signal is asserted. As shown in the preceding figure, when you set nCLRUSR to 0 for the second time to initialize OCT calibration block 0, this does not affect OCT calibration block 1, whose calibration is already in progress.

I/O Termination Schemes for Stratix V Devices

Table 5-21: Termination Schemes for Different I/O Standards

I/O Standard	External Termination Scheme
3.3-V LVTTL/3.3-V LVCMOS	No external termination required
2.5-V LVCMOS	
1.8-V LVCMOS	
1.5-V LVCMOS	
1.2-V LVCMOS	
SSTL-2 Class I	Single-Ended SSTL I/O Standard Termination
SSTL-2 Class II	
SSTL-18 Class I	
SSTL-18 Class II	
SSTL-15 Class I	
SSTL-15 Class II	
1.8-V HSTL Class I	Single-Ended HSTL I/O Standard Termination
1.8-V HSTL Class II	
1.5-V HSTL Class I	
1.5-V HSTL Class II	
1.2-V HSTL Class I	
1.2-V HSTL Class II	
Differential SSTL-2 Class I	Differential SSTL I/O Standard Termination
Differential SSTL-2 Class II	
Differential SSTL-18 Class I	
Differential SSTL-18 Class II	
Differential SSTL-15 Class I	
Differential SSTL-15 Class II	

I/O Standard	External Termination Scheme
Differential 1.8-V HSTL Class I	Differential HSTL I/O Standard Termination
Differential 1.8-V HSTL Class II	
Differential 1.5-V HSTL Class I	
Differential 1.5-V HSTL Class II	
Differential 1.2-V HSTL Class I	
Differential 1.2-V HSTL Class II	
LVDS	LVDS I/O Standard Termination
RSDS	RSDS/mini-LVDS I/O Standard Termination
Mini-LVDS	
LVPECL	Differential LVPECL I/O Standard Termination
SSTL-15 (7)	No external termination required
SSTL-135 (7)	
SSTL-125 (7)	
SSTL-12	
HSUL-12	
Differential SSTL-15 (7)	
Differential SSTL-135 (7)	
Differential SSTL-125 (7)	
Differential SSTL-12	
Differential HSUL-12	

Single-ended I/O Termination

Voltage-referenced I/O standards require an input V_{REF} and a termination voltage (V_{TT}). The reference voltage of the receiving device tracks the termination voltage of the transmitting device.

The supported I/O standards such as SSTL-12, SSTL-125, SSTL-135, and SSTL-15 typically do not require external board termination.

Altera recommends that you use dynamic OCT with these I/O standards to save board space and cost. Dynamic OCT reduces the number of external termination resistors used.

Note: You cannot use R_S and R_T OCT simultaneously. For more information, refer to the related information.

(7) Altera recommends that you use dynamic OCT with these I/O standards to save board space and cost. Dynamic OCT reduces the number of external termination resistors used.

Figure 5-15: SSTL I/O Standard Termination

This figure shows the details of SSTL I/O termination on Stratix V devices. This is not applicable for SSTL-12, SSTL-15, SSTL-125, and SSTL-135 I/O standards.

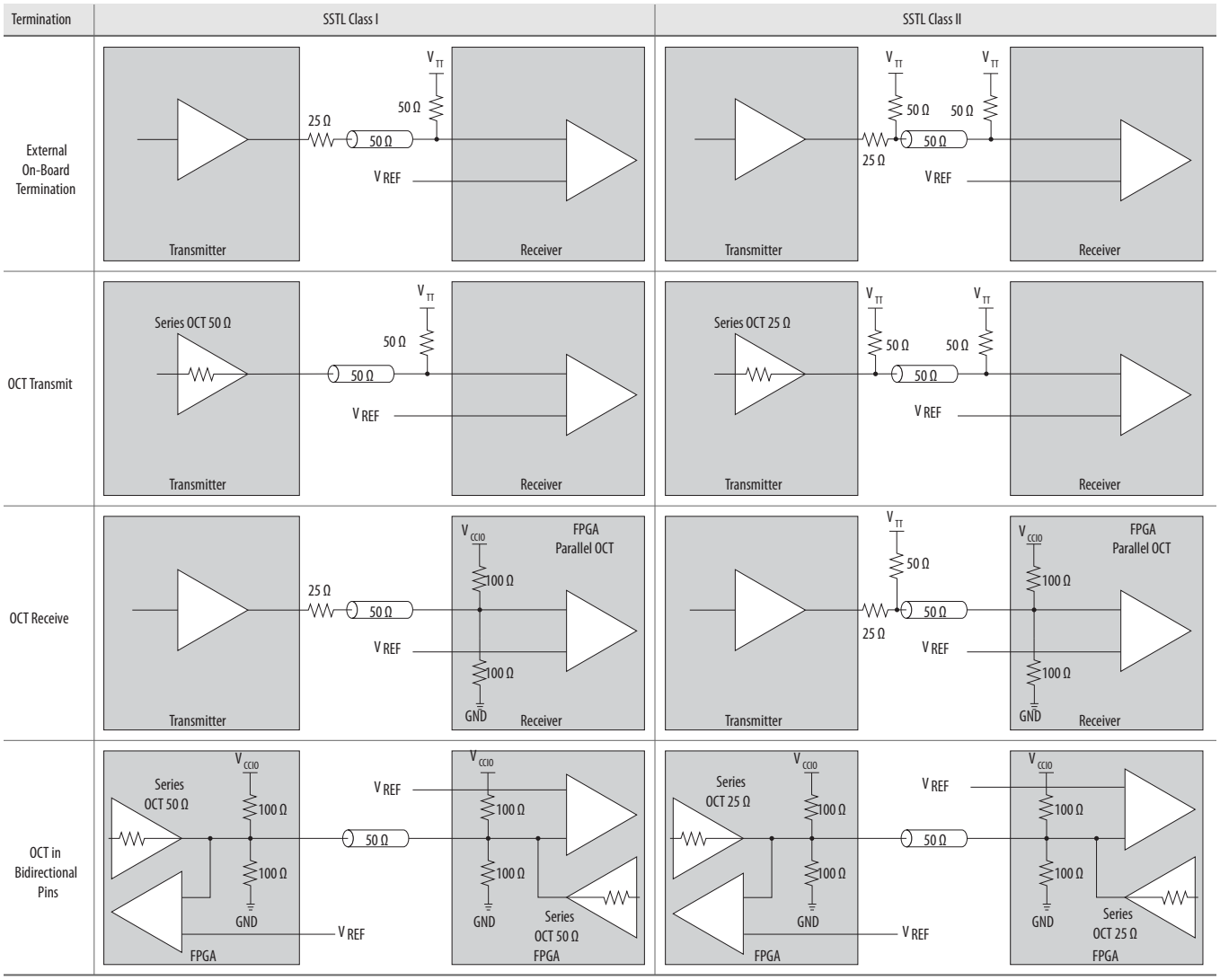
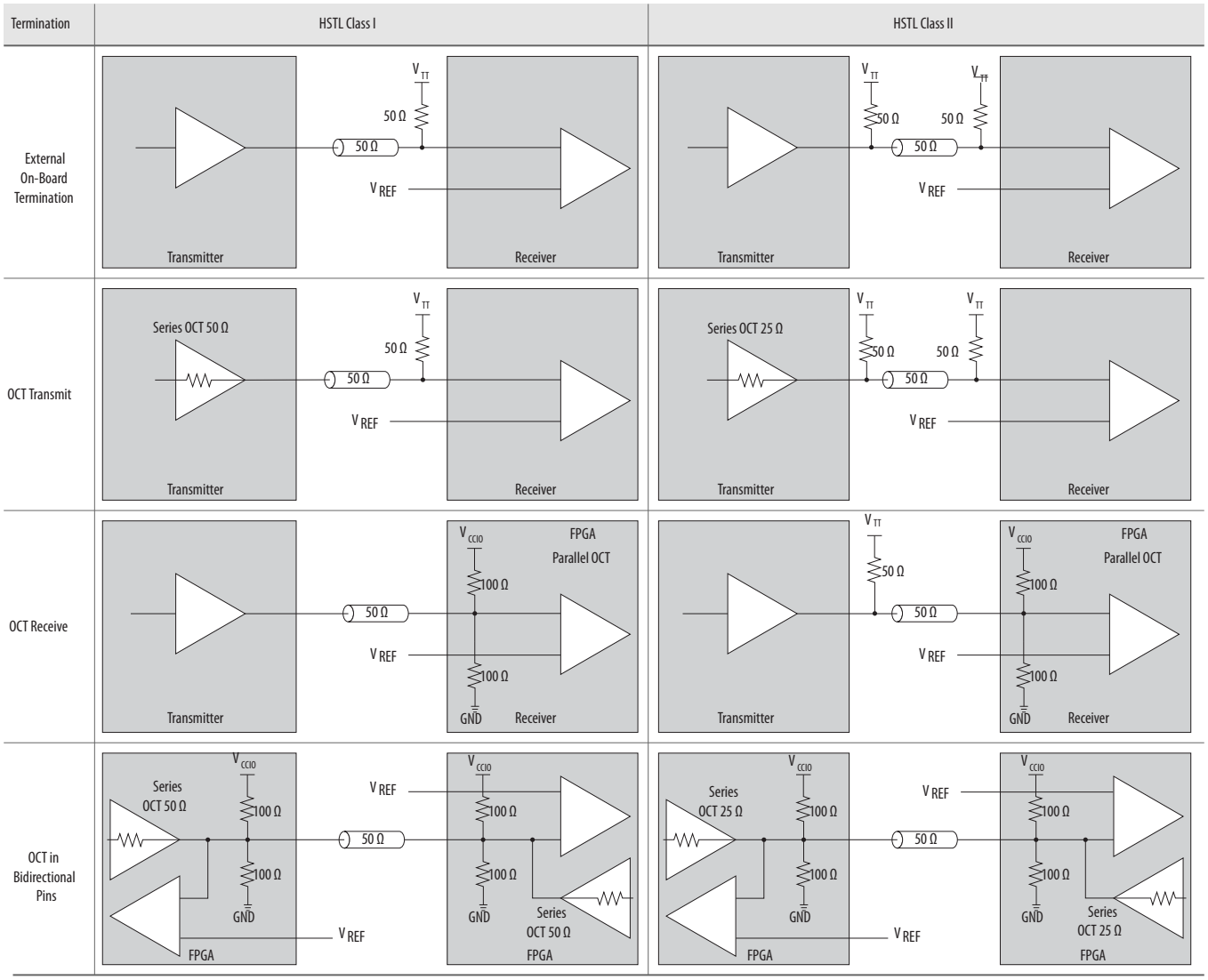


Figure 5-16: HSTL I/O Standard Termination

This figure shows the details of HSTL I/O termination on the Stratix V devices. This is not applicable for HSUL-12 I/O standard.



Related Information

Dynamic OCT in Stratix V Devices on page 5-29

Differential I/O Termination

The I/O pins are organized in pairs to support differential I/O standards. Each I/O pin pair can support differential input and output buffers.

The supported I/O standards such as Differential SSTL-12, Differential SSTL-15, Differential SSTL-125, and Differential SSTL-135 typically do not require external board termination.

Altera recommends that you use dynamic OCT with these I/O standards to save board space and cost. Dynamic OCT reduces the number of external termination resistors used.

Differential HSTL, SSTL, and HSUL Termination

Differential HSTL, SSTL, and HSUL inputs use LVDS differential input buffers with R_D support.

Differential HSTL, SSTL, and HSUL outputs are not true differential outputs. These I/O standards use two single-ended outputs with the second output programmed as inverted.

Figure 5-17: Differential SSTL I/O Standard Termination

This figure shows the details of Differential SSTL I/O termination on Stratix V devices. This is not applicable for differential SSTL-12, differential SSTL-15, differential SSTL-125, differential SSTL-135, and differential HSUL-12 I/O standards.

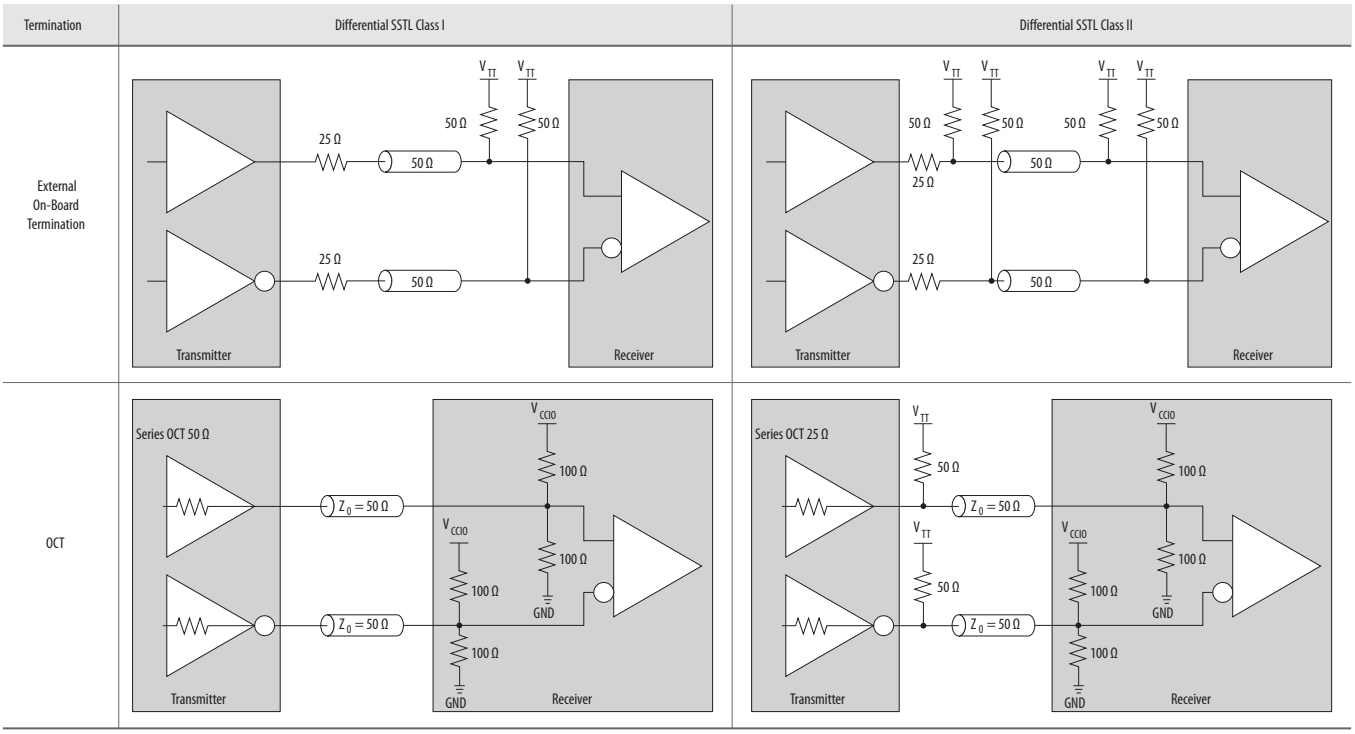
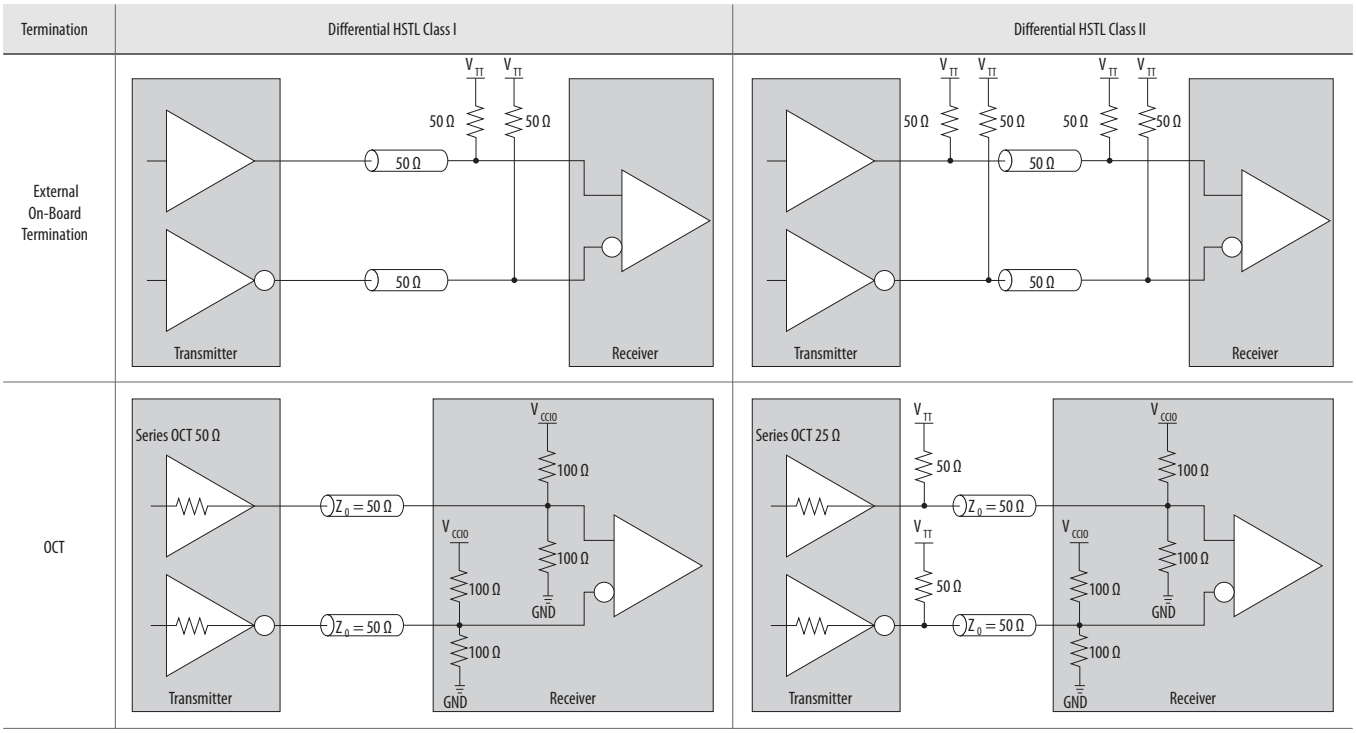


Figure 5-18: Differential HSTL I/O Standard Termination

This figure shows the details of Differential HSTL I/O standard termination on Stratix V devices. This is not applicable for differential HSUL-12 I/O standard.



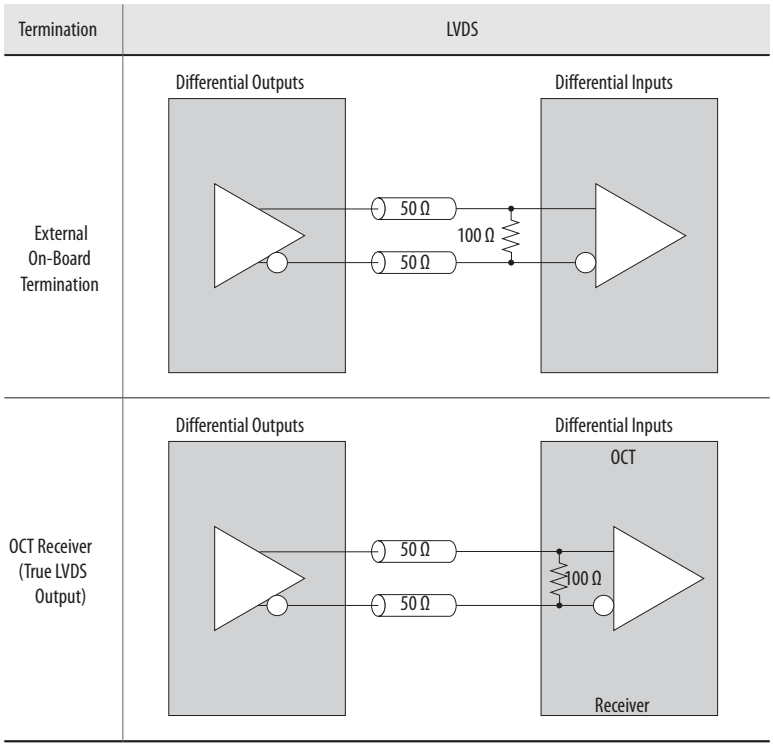
LVDS, RSDS, and Mini-LVDS Termination

All I/O banks have dedicated circuitry to support the true LVDS, RSDS, and mini-LVDS I/O standards by using true LVDS output buffers without resistor networks.

In Stratix V devices, the LVDS I/O standard requires a 2.5 V V_{CCIO} level. The LVDS input buffer requires 2.5 V V_{CCPD}. The LVDS receiver requires a 100 Ω termination resistor between the two signals at the input buffer. Stratix V devices provide an optional 100 Ω differential termination resistor in the device using R_D OCT if V_{CCPD} is set to 2.5 V.

Figure 5-19: LVDS I/O Standard Termination

This figure shows the LVDS I/O standard termination. The on-chip differential resistor is available in all I/O banks.



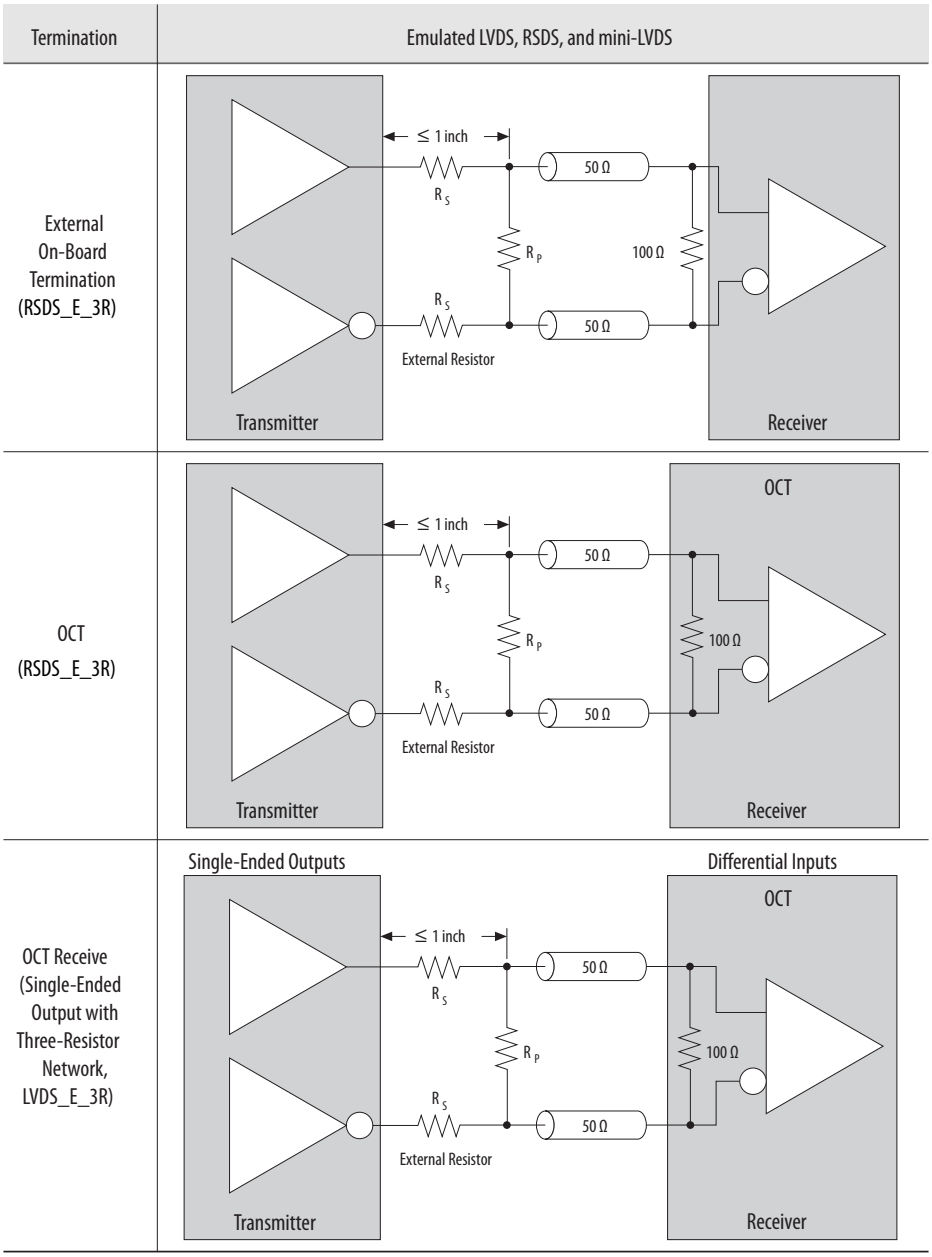
Emulated LVDS, RSDS, and Mini-LVDS Termination

The I/O banks also support emulated LVDS, RSDS, and mini-LVDS I/O standards.

Emulated LVDS, RSDS and mini-LVDS output buffers use two single-ended output buffers with an external three-resistor network, and can be tri-stated.

Figure 5-20: Emulated LVDS, RSDS, or Mini-LVDS I/O Standard Termination

The output buffers, as shown in this figure, are available in all I/O banks. For LVDS output with a three-resistor network, R_S is 120 Ω and R_P is 170 Ω. For RSDS and Mini-LVDS output, R_S and R_P values are pending characterization.



To meet the RSDS or mini-LVDS specifications, you require a resistor network to attenuate the output-voltage swing.

You can modify the three-resistor network values to reduce power or improve the noise margin. Choose resistor values that satisfy the following equation.

Figure 5-21: Resistor Network Calculation

$$\frac{R_S \times \frac{R_P}{2}}{R_S + \frac{R_P}{2}} = 50\ \Omega$$

Note: Altera recommends that you perform additional simulations with IBIS or SPICE models to validate that the custom resistor values meet the RSDS or mini-LVDS I/O standard requirements.

For information about the data rates supported for external three-resistor network, refer to the device datasheet.

Related Information

- **Stratix V Device Datasheet**
- **National Semiconductor (www.national.com)**
  For more information about the RSDS I/O standard, refer to the *RSDS Specification* on the National Semiconductor web site.

LVPECL Termination

The Stratix V devices support the LVPECL I/O standard on input clock pins only:

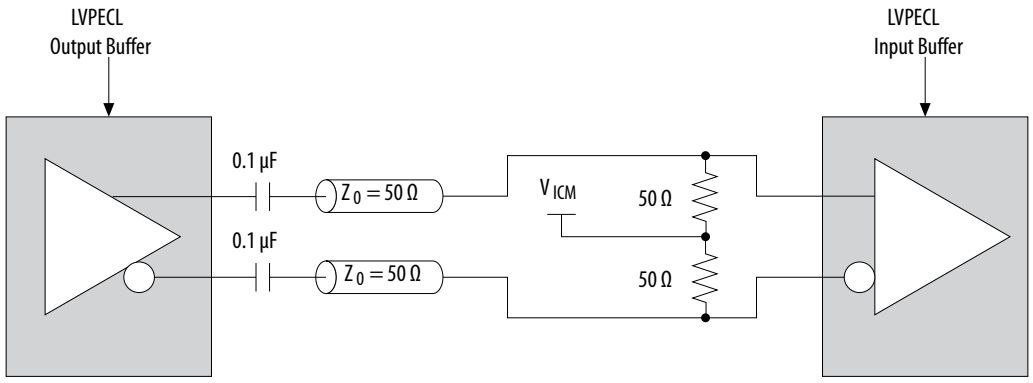
- LVPECL input operation is supported using LVDS input buffers.
- LVPECL output operation is not supported.

Use AC coupling if the LVPECL common-mode voltage of the output buffer does not match the LVPECL input common-mode voltage.

Note: Altera recommends that you use IBIS models to verify your LVPECL AC/DC-coupled termination.

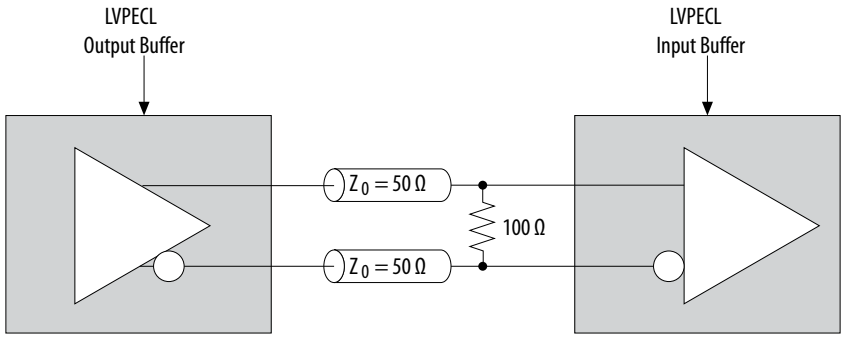
Figure 5-22: LVPECL AC-Coupled Termination

The 50 Ω resistors used at the receiver end are external to the device.



Support for DC-coupled LVPECL is available if the LVPECL output common mode voltage is within the Stratix V LVPECL input buffer specification.

Figure 5-23: LVPECL DC-Coupled Termination



Document Revision History

Date	Version	Changes
January 2015	2015.01.23	• Corrected truncated sentence in the note about the recommendation to use dynamic OCT for several I/O standards with DDR3 external memory interface. • Clarified that dedicated configuration pins, clock pins and JTAG pins do not support programmable pull-up resistor but these pins have fixed value of internal pull-up resistors. • Moved the Open-Drain Output, Bus-Hold Circuitry and Pull-up Resistor sections to Programmable IOE Features in Stratix V Devices. • Update Open-Drain Output section with steps to enable open-drain output in Assignment Editor.
June 2014	2014.06.30	• Added footnote to clarify that some of the voltage levels listed in the MultiVolt I/O support table are for showing that multiple single-ended I/O standards are not compatible with certain V_{CCIO} voltages. • Added information to clarify that programmable output slew-rate is available for single-ended and emulated LVDS I/O standards. • Finalized calibrated R_S and R_T OCT values and updated the R_T OCT values for HSUL-12 and Differential HSUL-12 I/O standards.
January 2014	2014.01.10	• Updated statements in several topics to clarify that each modular I/O bank can support multiple I/O standards that use the same voltages. • Updated the guideline topic about using the same V_{CCPD} for I/O banks in the same V_{CCPD} group to improve clarity. • Clarified that you can only use R_D OCT if V_{CCPD} is 2.5 V. • Corrected the topic about LVDS, RSDS, and Mini-LVDS termination to remove the requirement of 2.5 V V_{CCIO}. Only V_{CCPD} of 2.5 V is required for using R_D OCT. • Removed all "preliminary" marks.

Date	Version	Changes
June 2013	2013.06.21	• Updated the topic about LVDS input R_D OCT to remove the requirement for setting the V_{CCIO} to 2.5 V. R_D OCT now requires only that the V_{CCPD} is 2.5 V. • Updated the topic about LVPECL termination to improve clarity.
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Removed all references to column and row I/Os. Stratix V devices have I/O banks on the top and bottom only.
January 2013	2013.01.22	• Corrected the guideline about using the same V_{CCPD} for all I/O banks in a group. • Removed references to LVDS single-ended output with single-resistor network (LVDS_E_1R). The Stratix V devices do not support LVDS_E_1R.
December 2012	2012.12.28	• Reorganized content and updated template. • Added table about the termination schemes for different I/O standards. • Updated the SSTL and HSTL I/O termination figures to add V_{REF} inputs for OCT in bidirectional pins. • Added OCT diagram for LVDS single-ended output with single-resistor network (LVDS_E_1R). • Removed the "Summary of OCT Assignments" table and merged the information into the restructured OCT tables.
June 2012	1.5	• Added "Summary of OCT Assignments" and "LVDS Channels" sections. • Updated Table 5-2, Table 5-3, Table 5-4, Table 5-5, and Table 5-8. • Updated "Pull-Up Resistor", "Differential Output Voltage", and "Programmable IOE Delay" sections.
November 2011	1.4	• Updated Figure 5-2. • Updated Table 5-3, Table 5-4, and Table 5-5.
May 2011	1.3	• Chapter moved to volume 2 for the 11.0 release. • Added Table 5-4, Table 5-5, Table 5-6, Table 5-7, and Table 5-8. • Updated "Single-Ended I/O Standards Termination", "Differential I/O Standards Termination", and "VCCPD Restriction" sections. • Updated Table 5-3 and Table 5-11. • Updated Figure 5-1, Figure 5-8, Figure 5-9, Figure 5-10, Figure 5-17, Figure 5-20, and Figure 5-21. • Minor text edits.

Date	Version	Changes
January 2011	1.2	Updated Table 5-2.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

High-Speed Differential I/O Interfaces and DPA in Stratix V Devices

6

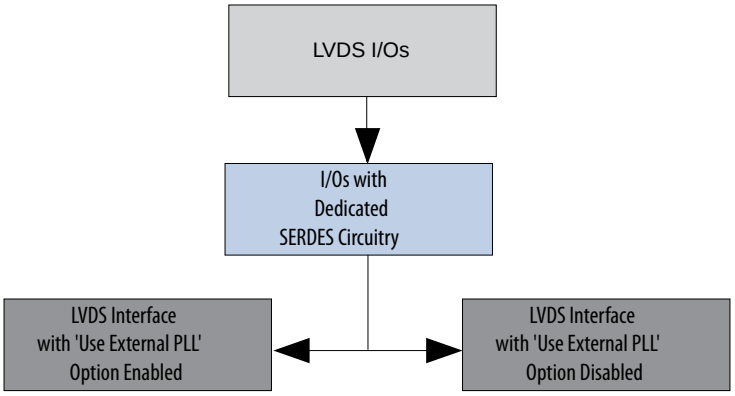
2015.01.23

SV51007 Subscribe Send Feedback

The high-speed differential I/O interfaces and dynamic phase alignment (DPA) features in Stratix V devices provide advantages over single-ended I/Os and contribute to the achievable overall system bandwidth. Stratix V devices support low-voltage differential signaling (LVDS), mini-LVDS, and reduced swing differential signaling (RSDS) differential I/O standards.

The following figure shows the I/O bank support for high-speed differential I/O in the Stratix V devices.

Figure 6-1: I/O Bank Support for High-Speed Differential I/O



Related Information

- **I/O Standards Support in Stratix V Devices** on page 5-2
  Provides information about the supported differential I/O standards.
- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the *Stratix V Device Handbook* chapters.

Dedicated High-Speed Circuitries in Stratix V Devices

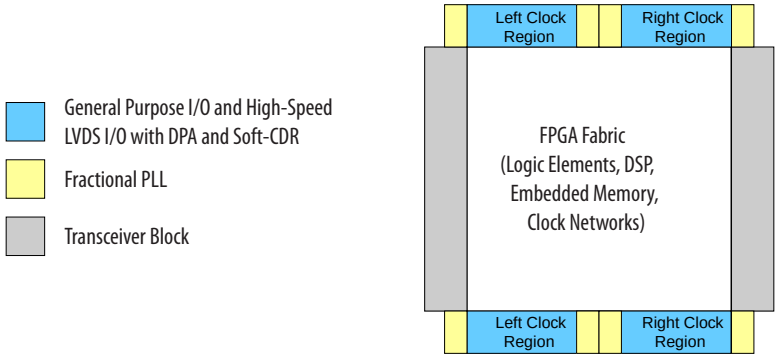
The following dedicated circuitries are available in the Stratix V device family to support high-speed differential I/O:

- Differential I/O buffer
- Transmitter serializer
- Receiver deserializer
- Data realignment (Bit-slip)
- DPA
- Synchronizer (FIFO buffer)
- Phase-locked loops (PLLs)

SERDES and DPA Bank Locations in Stratix V Devices

The dedicated serializer/deserializer (SERDES) and DPA circuitry that supports high-speed differential I/Os is located in the top and bottom banks of the Stratix V devices.

Figure 6-2: High-Speed Differential I/Os with DPA Locations in Stratix V Devices



Related Information

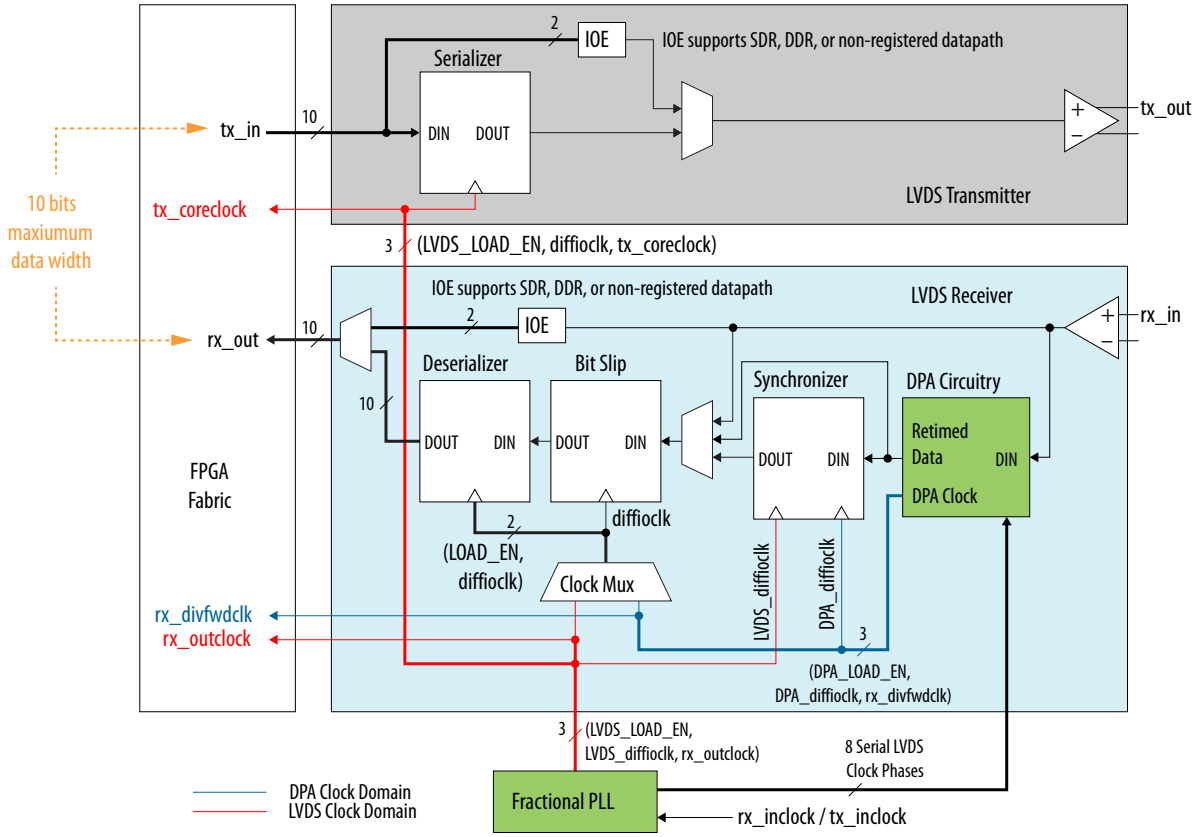
PLLs and Clocking for Stratix V Devices on page 6-8

LVDS SERDES Circuitry

The Stratix V devices have built-in serializer/deserializer (SERDES) circuitry that supports high-speed LVDS interfaces. You can configure the SERDES circuitry to support source-synchronous communication protocols such as RapidIO®, XSBI, serial peripheral interface (SPI), and asynchronous protocols such as Gigabit Ethernet (GbE) and SGMII.

The following figure shows a transmitter and receiver block diagram for the LVDS SERDES circuitry with the interface signals of the transmitter and receiver data paths.

Figure 6-3: LVDS SERDES



The preceding figure shows a shared PLL between the transmitter and receiver. If the transmitter and receiver do not share the same PLL, you require two fractional PLLs. In single data rate (SDR) and double data rate (DDR) modes, the data width is 1 and 2 bits, respectively.

The ALTLVDS transmitter and receiver requires various clock and load enable signals from a fractional PLL. The Quartus II software configures the PLL settings automatically. The software is also responsible for generating the various clock and load enable signals based on the input reference clock and selected data rate.

Note: For the maximum data rate supported by the Stratix V devices, refer to the device overview.

Related Information

- **Stratix V Device Overview**
- **LVDS SERDES Transmitter/Receiver (ALTLVDS_TX and ALTLVDS_RX) Megafunction User Guide**
  Provides a list of the LVDS transmitter and receiver ports and settings using ALTLVDS.
- **Guideline: Use PLLs in Integer PLL Mode for LVDS** on page 6-8

SERDES I/O Standards Support in Stratix V Devices

The following tables list the I/O standards supported by the SERDES receiver and transmitter, and the respective Quartus II software assignment values.

The SERDES receiver and transmitter also support all differential HSTL, differential HSUL, and differential SSTL I/O standards.

Table 6-1: SERDES Receiver I/O Standards Support

I/O Standard	Quartus II Software Assignment Value
True LVDS	LVDS
Differential 1.2 V HSTL Class I	Differential 1.2-V HSTL Class I
Differential 1.2 V HSTL Class II	Differential 1.2-V HSTL Class II
Differential HSUL-12	Differential 1.2-V HSUL
Differential SSTL-12	Differential 1.2-V SSTL
Differential SSTL-125	Differential 1.25-V SSTL
Differential SSTL-135	Differential 1.35-V SSTL
Differential 1.5 V HSTL Class I	Differential 1.5-V HSTL Class I
Differential 1.5 V HSTL Class II	Differential 1.5-V HSTL Class II
Differential SSTL-15	Differential 1.5-V SSTL
Differential SSTL-15 Class I	Differential 1.5-V SSTL Class I
Differential SSTL-15 Class II	Differential 1.5-V SSTL Class II
Differential 1.8 V HSTL Class I	Differential 1.8-V HSTL Class I
Differential 1.8 V HSTL Class II	Differential 1.8-V HSTL Class II
Differential SSTL-18 Class I	Differential 1.8-V SSTL Class I
Differential SSTL-18 Class II	Differential 1.8-V SSTL Class II
Differential SSTL-2 Class I	Differential 2.5-V SSTL Class I
Differential SSTL-2 Class II	Differential 2.5-V SSTL Class II

Table 6-2: SERDES Transmitter I/O Standards Support

I/O Standard	Quartus II Software Assignment Value
True LVDS	LVDS
Differential 1.2 V HSTL Class I	Differential 1.2-V HSTL Class I
Differential 1.2 V HSTL Class II	Differential 1.2-V HSTL Class II
Differential HSUL-12	Differential 1.2-V HSUL
Differential SSTL-12	Differential 1.2-V SSTL
Differential SSTL-125	Differential 1.25-V SSTL
Differential SSTL-135	Differential 1.35-V SSTL
Differential 1.5 V HSTL Class I	Differential 1.5-V HSTL Class I
Differential 1.5 V HSTL Class II	Differential 1.5-V HSTL Class II
Differential SSTL-15	Differential 1.5-V SSTL

I/O Standard	Quartus II Software Assignment Value
Differential SSTL-15 Class I	`Differential 1.5-V SSTL Class I`
Differential SSTL-15 Class II	`Differential 1.5-V SSTL Class II`
Differential 1.8 V HSTL Class I	`Differential 1.8-V HSTL Class I`
Differential 1.8 V HSTL Class II	`Differential 1.8-V HSTL Class II`
Differential SSTL-18 Class I	`Differential 1.8-V SSTL Class I`
Differential SSTL-18 Class II	`Differential 1.8-V SSTL Class II`
Differential SSTL-2 Class I	`Differential 2.5-V SSTL Class I`
Differential SSTL-2 Class II	`Differential 2.5-V SSTL Class II`
Emulated LVDS	`LVDS_E_3R`
mini-LVDS	`mini-LVDS`
Emulated mini-LVDS	`mini-LVDS_E_3R`
RSDS	`RSDS`
Emulated RSDS	`RSDS_E_3R`

True LVDS Buffers in Stratix V Devices

The Stratix V device family supports LVDS on all I/O banks:

- All I/Os support true LVDS input buffers with R_D OCT or true LVDS output buffers.
- Stratix V devices offer single-ended I/O reference clock support for the fractional PLL that drives the SERDES.

The following tables list the number of true LVDS buffers supported in Stratix V devices with these conditions:

- The LVDS channel count does not include dedicated clock pins.
- Dedicated SERDES and DPA is available for top and bottom banks only.

Table 6-3: LVDS Channels Supported in Stratix V E Devices

Member Code	Package	Side	TX	RX
E9 and EB	H40-H1517	Top	87	87
		Bottom	87	87
	F45-F1932	Top	105	105
		Bottom	105	105

Table 6-4: LVDS Channels Supported in Stratix V GX Devices

Member Code	Package	Side	TX	RX
A3	EH29-H780	Top	51	51
		Bottom	39	39
	HF35-F1152	Top	57	57
		Bottom	51	51
	KF35-F1152	Top	54	54
		Bottom	54	54
	KF40-F1517	Top	87	87
		Bottom	87	87
A4	HF35-F1152	Top	63	63
		Bottom	75	75
	KF35-F1152	Top	54	54
		Bottom	54	54
	KF40-F1517	Top	87	87
		Bottom	87	87
A5 and A7	HF35-F1152	Top	63	63
		Bottom	75	75
	KF35-F1152	Top	54	54
		Bottom	54	54
	KF40-F1517	Top	87	87
		Bottom	87	87
	NF40-F1517	Top	75	75
		Bottom	75	75
	NF45-F1932	Top	105	105
		Bottom	105	105
A9 and AB	KH40-1517	Top	87	87
		Bottom	87	87
	NF45-F1932	Top	105	105
		Bottom	105	105
B5 and B6	RF40-F1517	Top	54	54
		Bottom	54	54
	RF43-F1760	Top	75	75
		Bottom	75	75

Member Code	Package	Side	TX	RX
B9 and BB	RH43-H1760	Top	75	75
		Bottom	75	75

Table 6-5: LVDS Channels Supported in Stratix V GS Devices

Member Code	Package	Side	TX	RX
D3	EH29-H780	Top	51	51
		Bottom	39	39
	HF35-F1152	Top	57	57
		Bottom	51	51
D4	EH29-H780	Top	51	51
		Bottom	39	39
	HF35-F1152	Top	57	57
		Bottom	51	51
	KF40-F1517	Top	87	87
		Bottom	87	87
D5	HF35-F1152	Top	63	63
		Bottom	75	75
	KF40-F1517	Top	87	87
		Bottom	87	87
D6 and D8	KF40-F1517	Top	87	87
		Bottom	87	87
	NF45-F1932	Top	105	105
		Bottom	105	105

Table 6-6: LVDS Channels Supported in Stratix V GT Devices

Member Code	Package	Side	TX	RX
C5 and C7	KF40-F1517	Top	75	75
		Bottom	75	75

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Emulated LVDS Buffers in Stratix V Devices

The Stratix V device family supports emulated LVDS on all I/O banks:

- You can use unutilized true LVDS input channels as emulated LVDS output buffers (eTX), which use two single-ended output buffers with an external resistor network to support LVDS, mini-LVDS, and RSDS I/O standards.
- The emulated differential output buffers support tri-state capability.

High-Speed I/O Design Guidelines for Stratix V Devices

There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

PLLs and Clocking for Stratix V Devices

To generate the parallel clocks (`rx_outclock` and `tx_outclock`) and high-speed clocks (`diffioclk`), the Stratix V devices provide fractional PLLs in the high-speed differential I/O receiver and transmitter channels.

Related Information

- **SERDES and DPA Bank Locations in Stratix V Devices** on page 6-2
  Provides information about the PLL locations available for each Stratix V device.
- **Guideline: Use High-Speed Clock from PLL to Clock LVDS SERDES Only** on page 6-8
- **Guideline: Use PLLs in Integer PLL Mode for LVDS** on page 6-8

Guideline: Use PLLs in Integer PLL Mode for LVDS

To drive the LVDS channels, you must use the PLLs in integer PLL mode. The center or corner PLLs can drive the LVDS receiver and transmitter channels.

However, the clock tree network cannot cross over to different I/O regions. For example, the top left corner PLL cannot cross over to drive the LVDS receiver and transmitter channels on the top right I/O bank.

Related Information

Pin Placement Guidelines for DPA Differential Channels on page 6-13
Provides more information about the fractional PLL clocking restrictions.

Guideline: Use High-Speed Clock from PLL to Clock LVDS SERDES Only

The high-speed clock generated from the PLL is intended to clock the LVDS SERDES circuitry only. Do not use the high-speed clock to drive other logic because the allowed frequency to drive the core logic is restricted by the PLL F_{OUT} specification.

For more information about the F_{OUT} specification, refer to the device datasheet.

Related Information

Stratix V Device Datasheet

LVDS Interface with External PLL Mode

The MegaWizard Plug-In Manager provides an option for implementing the LVDS interface with the **Use External PLL** option. With this option enabled you can control the PLL settings, such as dynamically reconfiguring the PLL to support different data rates, dynamic phase shift, and other settings. You must also instantiate the an Altera_PLL megafunction to generate the various clock and load enable signals.

If you enable the **Use External PLL** option with the ALTLVDS transmitter and receiver, the following signals are required from the Altera_PLL megafunction:

- Serial clock input to the SERDES of the ALTLVDS transmitter and receiver
- Load enable to the SERDES of the ALTLVDS transmitter and receiver
- Parallel clock used to clock the transmitter FPGA fabric logic and parallel clock used for the receiver
- Asynchronous PLL reset port of the ALTLVDS receiver

Altera_PLL Signal Interface with ALTLVDS Megafunction

Table 6-7: Signal Interface Between Altera_PLL and ALTLVDS Megafunctions

This table lists the signal interface between the output ports of the Altera_PLL megafunction and the input ports of the ALTLVDS transmitter and receiver. As an example, the table lists the serial clock output, load enable output, and parallel clock output generated on ports outclk0, outclk1, and outclk2, along with the locked signal of the Altera_PLL instance. You can choose any of the PLL output clock ports to generate the interface clocks.

From the Altera_PLL Megafunction	To the ALTLVDS Transmitter	To the ALTLVDS Receiver
Serial clock output (outclk0) The serial clock output (outclk0) can only drive `tx_inclock` on the ALTLVDS transmitter, and `rx_inclock` and `rx_dpaclock` on the ALTLVDS receiver. This clock cannot drive the core logic.	`tx_inclock` (serial clock input to the transmitter)	`rx_inclock` (serial clock input) `rx_dpaclock`
Load enable output (outclk1)	`tx_enable` (load enable to the transmitter)	`rx_enable` (load enable for the deserializer)
Parallel clock output (outclk2)	Parallel clock used inside the transmitter core logic in the FPGA fabric	`rx_syncclock` (parallel clock input) and parallel clock used inside the receiver core logic in the FPGA fabric
~(locked)	—	`pll_areset` (asynchronous PLL reset port) The `pll_areset` signal is automatically enabled for the LVDS receiver in external PLL mode. This signal does not exist for LVDS transmitter instantiation when the external PLL option is enabled.

Note: With soft SERDES, a different clocking requirement is needed.

Related Information

LVDS SERDES Transmitter/Receiver (ALTLVDS_RX/TX) Megafunction User Guide
More information about the different clocking requirement for soft SERDES.

Altera_PLL Parameter Values for External PLL Mode

The following examples show the clocking requirements to generate output clocks for ALTLVDS_TX and ALTLVDS_RX using the Altera_PLL megafunction. The examples set the phase shift with the assumption that the clock and data are edge aligned at the pins of the device.

Note: For other clock and data phase relationships, Altera recommends that you first instantiate your ALTLVDS_RX and ALTLVDS_TX interface without using the external PLL mode option. Compile the megafunctions in the Quartus II software and take note of the frequency, phase shift, and duty cycle settings for each clock output. Enter these settings in the Altera_PLL megafunction parameter editor and then connect the appropriate output to the ALTLVDS_RX and ALTLVDS_TX megafunctions.

Table 6-8: Example: Generating Output Clocks Using an Altera_PLL Megafunction (No DPA and Soft-CDR Mode)

This table lists the parameter values that you can set in the Altera_PLL parameter editor to generate three output clocks using an Altera_PLL megafunction if you are not using DPA and soft-CDR mode.

Parameter	outclk0 (Connects to the tx_inclock port of ALTLVDS_TX and the rx_inclock port of ALTLVDS_RX)	outclk1 (Connects to the tx_enable port of ALTLVDS_TX and the rx_enable port of ALTLVDS_RX)	outclk2 (Used as the core clock for the parallel data registers for both transmitter and receiver, and connects to the rx_syncclock port of ALTLVDS_RX)
Frequency	data rate	data rate/serialization factor	data rate/serialization factor
Phase shift	–180°	[(deserialization factor – 2)/ deserialization factor] x 360°	–180/serialization factor (outclk0 phase shift divided by the serialization factor)
Duty cycle	50%	100/serialization factor	50%

The calculations for phase shift, using the RSKM equation, assume that the input clock and serial data are edge aligned. Introducing a phase shift of –180° to sampling clock (c0) ensures that the input data is center-aligned with respect to the outclk0, as shown in the following figure.

Figure 6-4: Phase Relationship for External PLL Interface Signals

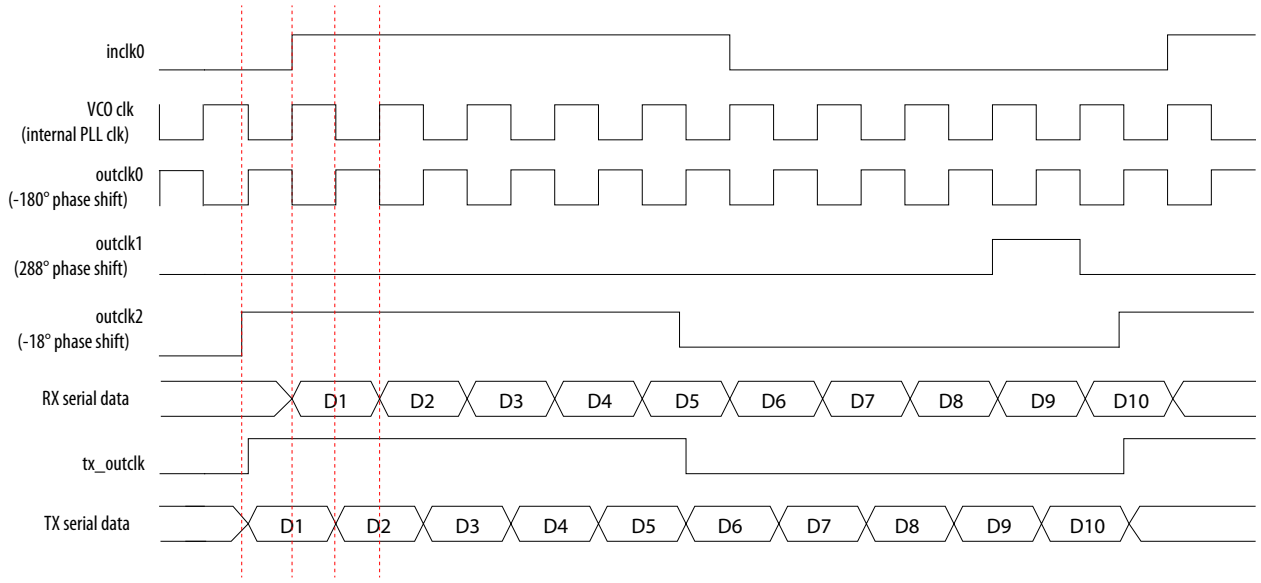


Table 6-9: Example: Generating Output Clocks Using an Altera_PLL Megafunction (With DPA and Soft-CDR Mode)

This table lists the parameter values that you can set in the Altera_PLL parameter editor to generate four output clocks using an Altera_PLL megafunction if you are using DPA and soft-CDR mode. The `locked` output port of Altera_PLL must be inverted and connected to the `pll_areset` port of the ALTLVDS_RX megafunction if you are using DPA and soft-CDR mode.

Parameter	outclk0 (Connects to the `tx_inclock` port of ALTLVDS_TX and the `rx_inclock` port of ALTLVDS_RX)	outclk1 (Connects to the `tx_enable` port of ALTLVDS_TX and the `rx_enable` port of ALTLVDS_RX)	outclk2 (Used as the core clock for the parallel data registers for both transmitter and receiver, and connects to the `rx_syncclock` port of ALTLVDS_RX)	outclk3 (Connects to the `rx_dpaclock` port of ALTLVDS_RX)
Frequency	data rate	data rate/serialization factor	data rate/serialization factor	data rate
Phase shift	–180°	[(deserialization factor - 2)/deserialization factor] x 360°	–180/serialization factor (outclk0 phase shift divided by the serialization factor)	–180°
Duty cycle	50%	100/serialization factor	50%	50%

Related Information

Receiver Skew Margin for Non-DPA Mode on page 6-33
RSKM equation used for the phase shift calculations.

Connection between Altera_PLL and ALTLVDS

Figure 6-5: LVDS Interface with the Altera_PLL Megafunction (Without DPA and Soft-CDR Mode)

This figure shows the connections between the Altera_PLL and ALTLVDS megafunction if you are not using DPA and soft-CDR mode.

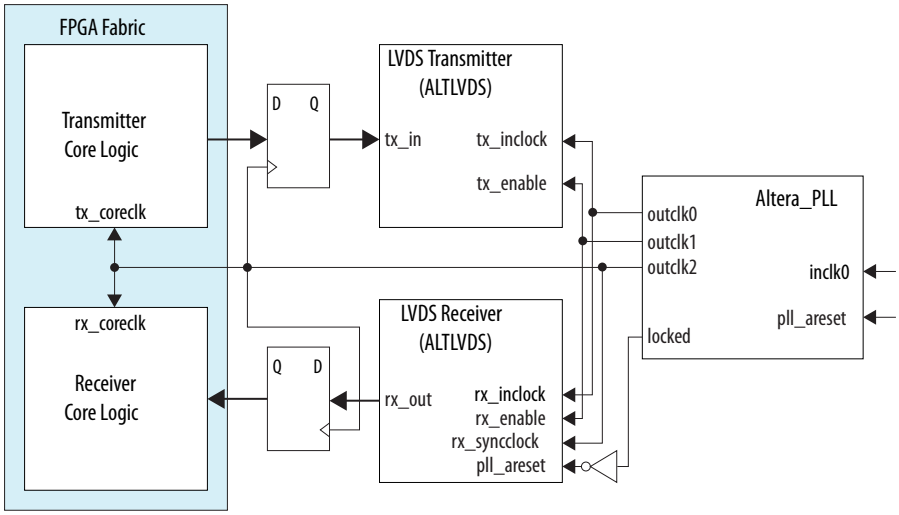


Figure 6-6: LVDS Interface with the Altera_PLL Megafunction (With DPA)

This figure shows the connections between the Altera_PLL and ALTLVDS megafunction if you are using DPA. The `locked` output port must be inverted and connected to the `pll_areset` port.

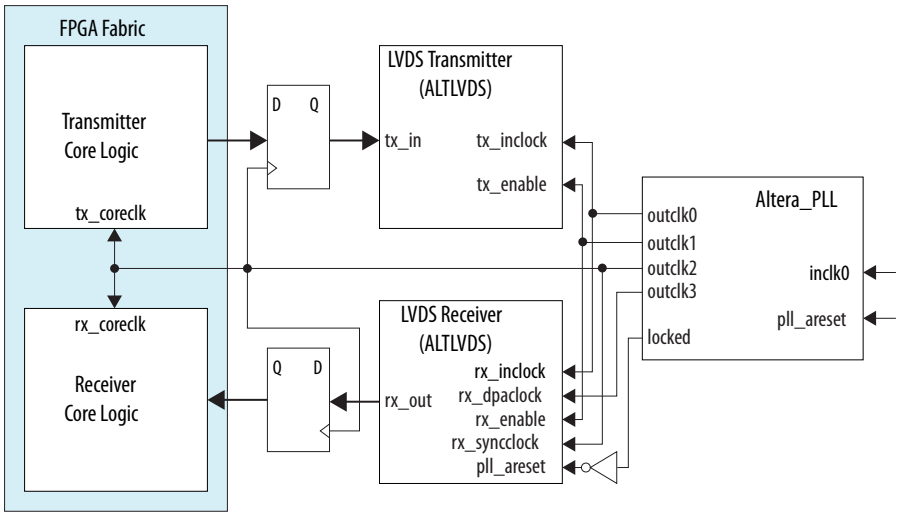
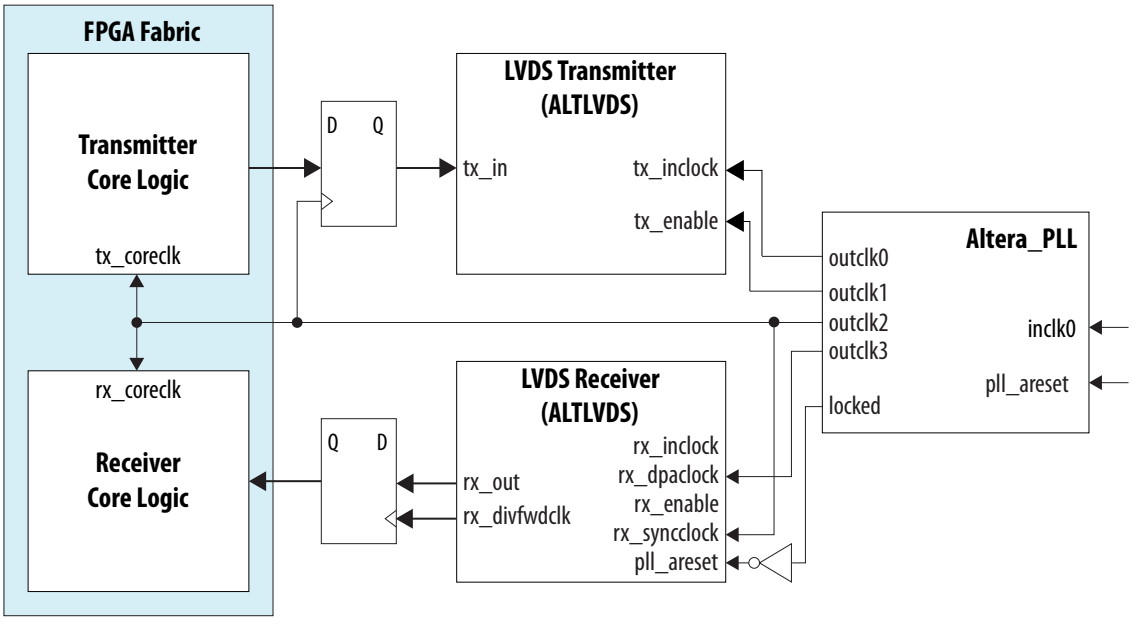


Figure 6-7: LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode)

This figure shows the connections between the Altera_PLL and ALTLVDS megafunction if you are using soft-CDR mode. The `locked` output port must be inverted and connected to the `pll_areset` port.



When generating the Altera_PLL megafunction, the **Left/Right PLL** option is configured to set up the PLL in LVDS mode. Instantiation of `pll_areset` is optional.

The `rx_enable` and `rx_inclock` input ports are not used and can be left unconnected.

Pin Placement Guidelines for DPA Differential Channels

DPA usage adds some constraints on the placement of high-speed differential channels. If DPA-enabled or DPA-disabled differential channels[8] in the differential banks are used, you must adhere to the differential pin placement guidelines to ensure the proper high-speed operation. The Quartus II compiler automatically checks the design and issues an error message if the guidelines are not followed.

Note: The figures in this section show guidelines for using corner and center PLLs but do not necessarily represent the exact locations of the high-speed LVDS I/O banks.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Guideline: Using DPA-Enabled Differential Channels

Each differential receiver in an I/O block has a dedicated DPA circuit to align the phase of the clock to the data phase of its associated channel. If you enable a DPA channel in a bank, you can use both single-ended I/Os and differential I/O standards in the bank.

[8] DPA-enabled differential channels refer to DPA mode or soft-CDR mode while DPA disabled channels refer to non-DPA mode.

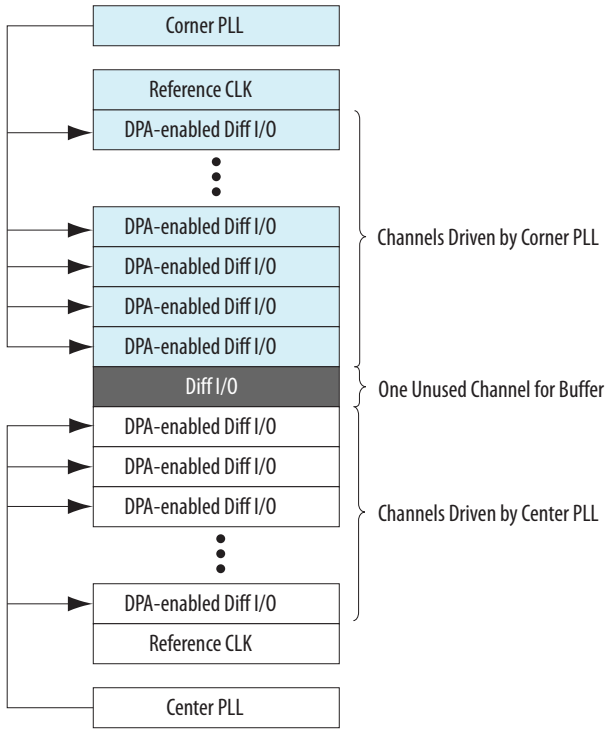
You can place double data rate I/O (DDIO) output pins within I/O modules that have the same pad group number as a SERDES differential channel. However, you cannot place SDR I/O output pins within I/O modules that have the same pad group number as a receiver SERDES differential channel. You must implement the input register within the FPGA fabric logic.

If you use DPA-enabled channels in differential banks, adhere to the following guidelines.

Using Center and Corner PLLs

If two PLLs drive the DPA-enabled channels in a bank—the corner and center PLL drive one group each—there must be at least one row (one differential channel) of separation between the two groups of DPA-enabled channels, as shown in the following figure.

Figure 6-8: Center and Corner PLLs Driving DPA-enabled Differential I/Os in the Same Bank



This separation prevents noise mixing because the two groups can operate at independent frequencies. No separation is necessary if a single PLL is driving both the DPA-enabled channels and DPA-disabled channels.

Using Both Center PLLs

You can use center PLLs to drive DPA-enabled channels simultaneously, if they drive these channels in their adjacent banks only, as shown in the previous figure. The center PLLs cannot drive cross-banks simultaneously. Refer to the following figures.

Figure 6-9: Center PLLs Driving DPA-enabled Differential I/Os

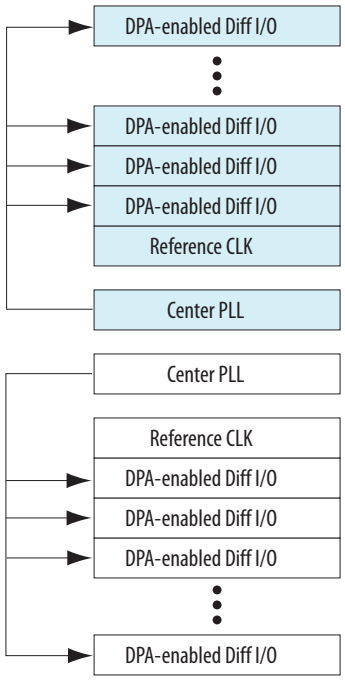
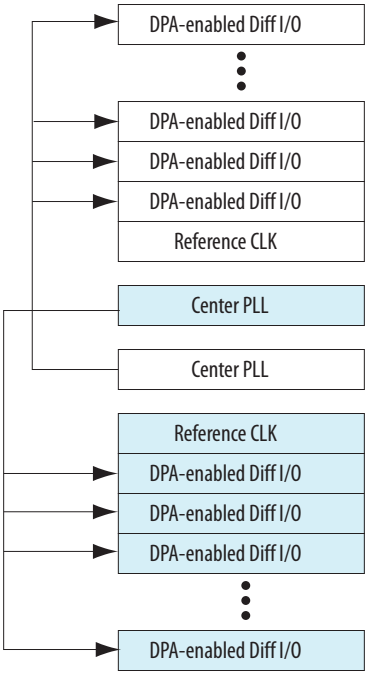


Figure 6-10: Invalid Placement of DPA-enabled Differential I/Os Driven by Both Center PLLs



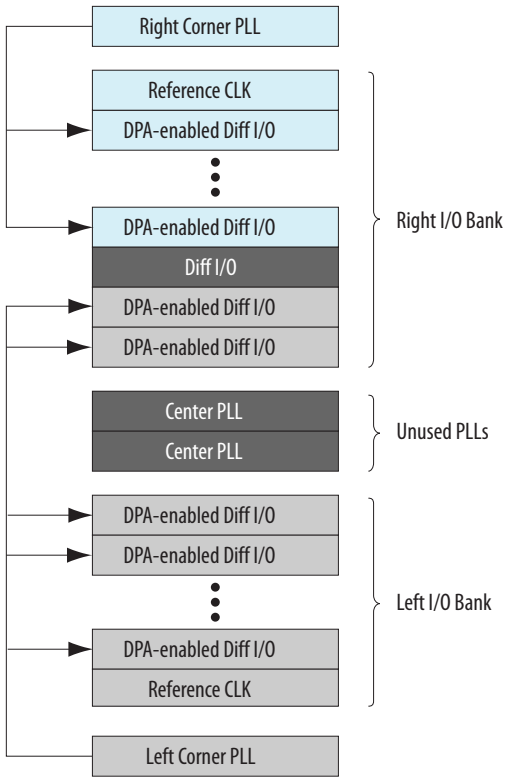
Using Both Corner PLLs

You can use both corner PLLs to drive DPA-enabled channels simultaneously, if they drive the channels in their adjacent banks only. There must be at least one row of separation between the two groups of DPA-enabled channels.

If one of the corner PLLs drives DPA-enabled channels in the left and right I/O banks, you cannot use the center PLLs to drive DPA-enabled channels. You can use the other corner PLL to drive DPA-enabled channels in their adjacent bank only. There must be at least one row of separation between the two groups of DPA-enabled channels.

If the left corner PLL drives DPA-enabled channels in the right I/O bank, the right corner PLL cannot drive DPA-enabled channels in the left I/O bank, and vice versa. In other words, the corner PLLs cannot drive cross-banks simultaneously, as shown in the following figure.

Figure 6-11: Corner PLLs Driving DPA-enabled Differential I/Os



DPA Restrictions

Because there is only a single DPA clock bus, a PLL drives a continuous series of DPA channels.

To prevent noise mixing, use one row of separation between two groups of DPA channels.

Guideline: Using DPA-Disabled Differential Channels

If you use DPA-disabled channels, adhere to the following guidelines.

DPA-Disabled Channel Driving Distance

Each PLL can drive all the DPA-disabled channels in the entire bank.

Using Corner and Center PLLs

You can use a corner PLL to drive all transmitter channels and a center PLL to drive all DPA-disabled receiver channels in the same I/O bank. You can drive a transmitter channel and a receiver channel in the same LAB row by two different PLLs. A corner PLL and a center PLL can drive duplex channels in the same I/O bank if the channels that are driven by each PLL are not interleaved. You do not require separation between the group of channels that are driven by the corner and center, left and right PLLs. Refer to the following figures.

Figure 6-12: Corner and Center PLLs Driving DPA-Disabled Differential I/Os in the Same Bank

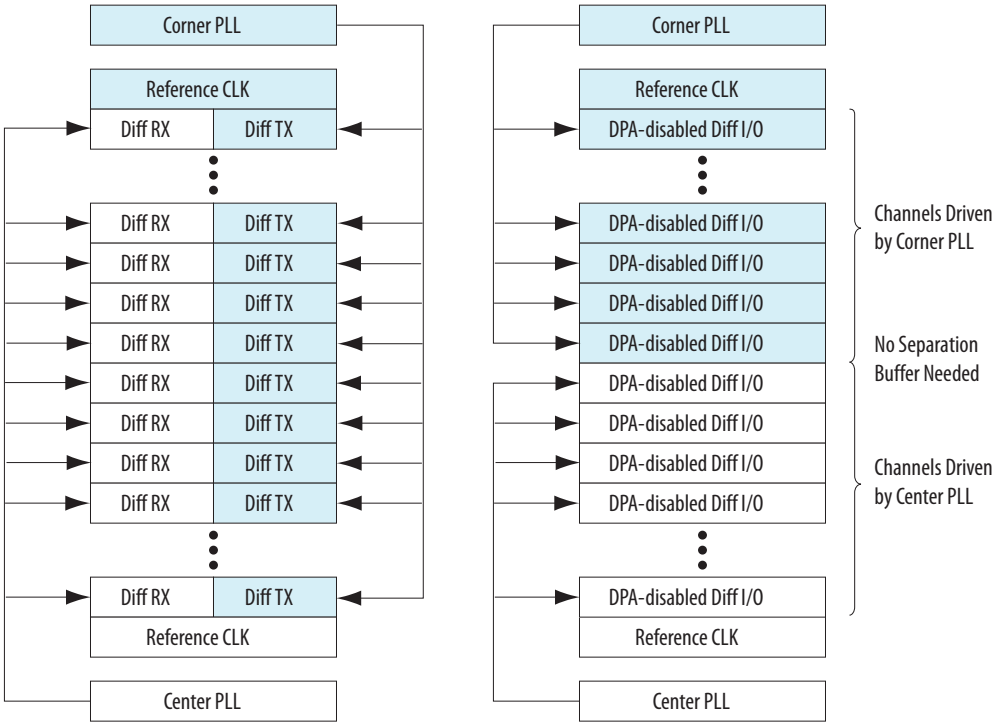
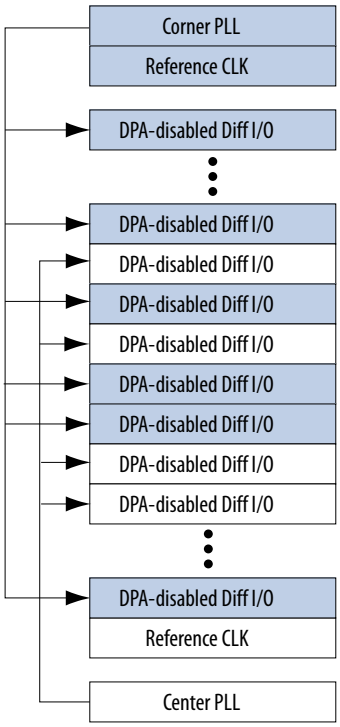


Figure 6-13: Invalid Placement of DPA-disabled Differential I/Os Due to Interleaving of Channels Driven by the Corner and Center PLLs



Using Both Corner PLLs

You can use both corner PLLs to drive DPA-disabled channels simultaneously. You can use a corner PLL to drive all the transmitter channels and the other corner PLL to drive all the DPA-disabled receiver channels in the same I/O bank. Both corner PLLs can drive duplex channels in the same I/O bank if the channels that are driven by each PLL are not interleaved. You do not require separation between the groups of channels that are driven by both corner PLLs.

Differential Transmitter in Stratix V Devices

The Stratix V transmitter contains dedicated circuitry to support high-speed differential signaling. The differential transmitter buffers support the following features:

- LVDS signaling that can drive out LVDS, mini-LVDS, and RSDS signals
- Programmable V_{OD} and programmable pre-emphasis

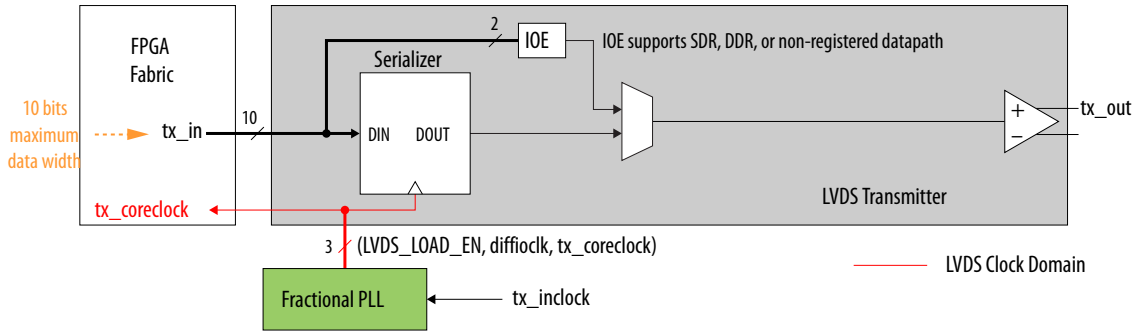
Transmitter Blocks

The dedicated circuitry consists of a true differential buffer, a serializer, and fractional PLLs that you can share between the transmitter and receiver. The serializer takes up to 10 bits wide parallel data from the FPGA fabric, clocks it into the load registers, and serializes it using shift registers that are clocked by the fractional PLL before sending the data to the differential buffer. The MSB of the parallel data is transmitted first.

Note: To drive the LVDS channels, you must use the PLLs in integer PLL mode.

The following figure shows a block diagram of the transmitter. In SDR and DDR modes, the data width is 1 and 2 bits, respectively.

Figure 6-14: LVDS Transmitter



Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Transmitter Clocking

The fractional PLL generates the load enable (`LVDS_LOAD_EN`) signal and the `diffioclk` signal (the clock running at serial data rate) that clocks the load and shift registers. You can statically set the serialization factor to x3, x4, x5, x6, x7, x8, x9, or x10 using the Quartus II software. The load enable signal is derived from the serialization factor setting.

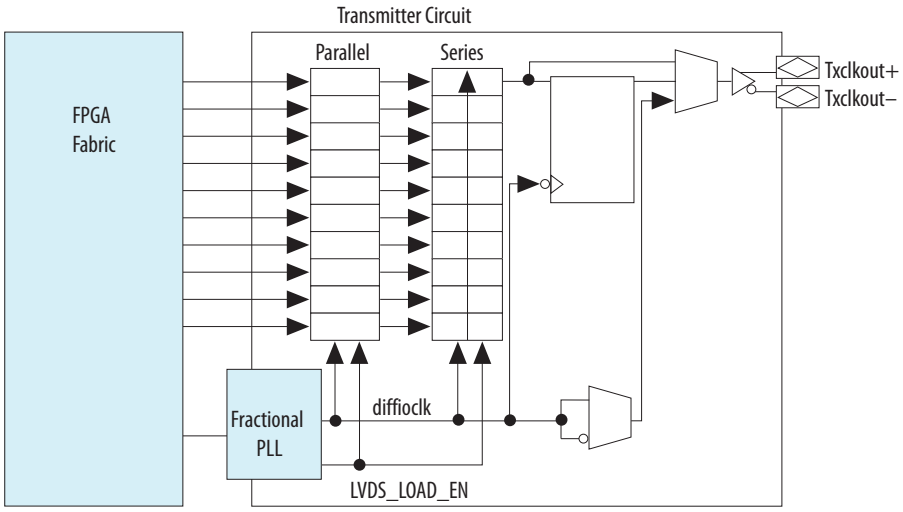
You can configure any Stratix V transmitter data channel to generate a source-synchronous transmitter clock output. This flexibility allows the placement of the output clock near the data outputs to simplify board layout and reduce clock-to-data skew.

Different applications often require specific clock-to-data alignments or specific data-rate-to-clock-rate factors. You can specify these settings statically in the Quartus II MegaWizard Plug-In Manager:

- The transmitter can output a clock signal at the same rate as the data—with a maximum output clock frequency that each speed grade of the device supports.
- You can divide the output clock by a factor of 1, 2, 4, 6, 8, or 10, depending on the serialization factor.
- You can set the phase of the clock in relation to the data using internal PLL option of the ALTLVDS megafunction. The fractional PLLs provide additional support for other phase shifts in 45° increments.

The following figure shows the transmitter in clock output mode. In clock output mode, you can use an LVDS channel as a clock output channel.

Figure 6-15: Transmitter in Clock Output Mode



Related Information

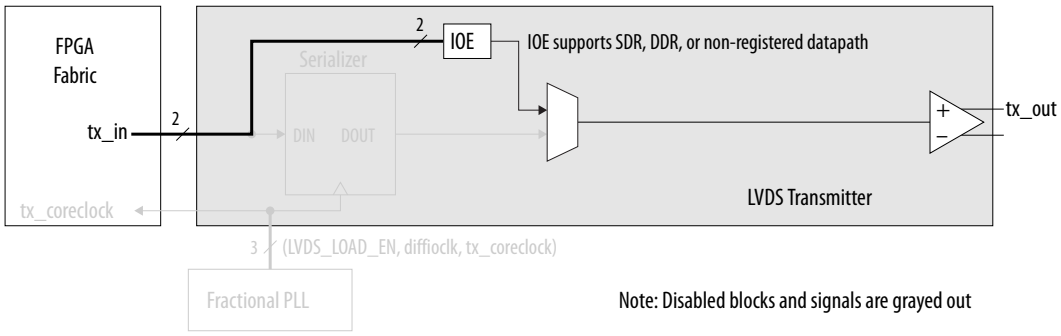
Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Serializer Bypass for DDR and SDR Operations

You can bypass the serializer to support DDR (x2) and SDR (x1) operations to achieve a serialization factor of 2 and 1, respectively. The I/O element (IOE) contains two data output registers that can each operate in either DDR or SDR mode.

Figure 6-16: Serializer Bypass

This figure shows the serializer bypass path. In DDR mode, `tx_inclock` clocks the IOE register. In SDR mode, data is passed directly through the IOE. In SDR and DDR modes, the data width to the IOE is 1 and 2 bits, respectively.



Programmable Differential Output Voltage

The programmable V_{OD} settings allow you to adjust the output eye opening to optimize the trace length and power consumption. A higher V_{OD} swing improves voltage margins at the receiver end, and a smaller V_{OD} swing reduces power consumption. You can statically adjust the V_{OD} of the differential signal by changing the V_{OD} settings in the Quartus II software Assignment Editor.

Figure 6-17: Differential V_{OD}

This figure shows the V_{OD} of the differential LVDS output.

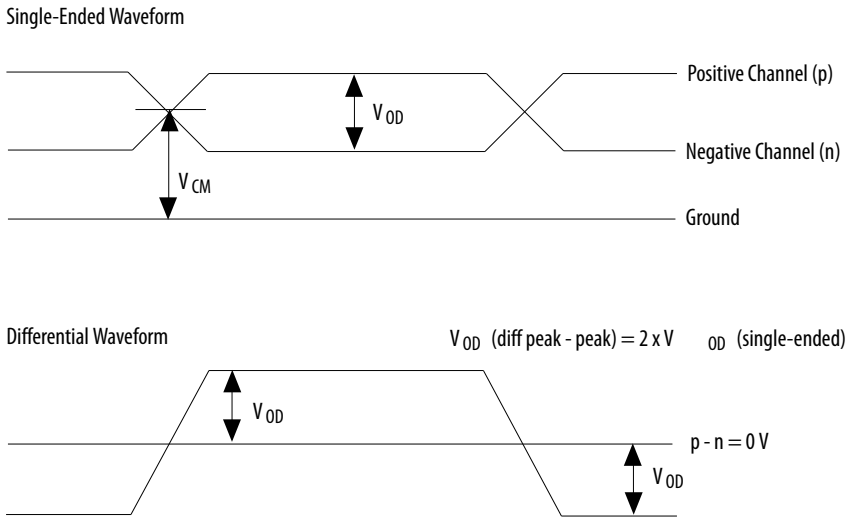


Table 6-10: Quartus II Software Assignment Editor—Programmable V_{OD}

This table lists the assignment name for programmable V_{OD} and its possible values in the Quartus II software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Differential Output Voltage (V_{OD})
Allowed values	0 (low), 1 (medium low), 2 (medium high), 3 (high). Default is 1.

Related Information

Programmable IOE Features in Stratix V Devices on page 5-18

Programmable Pre-Emphasis

The V_{OD} setting and the output impedance of the driver set the output current limit of a high-speed transmission signal. At a high frequency, the slew rate may not be fast enough to reach the full V_{OD} level before the next edge, producing pattern-dependent jitter. With pre-emphasis, the output current is boosted momentarily during switching to increase the output slew rate.

Pre-emphasis increases the amplitude of the high-frequency component of the output signal, and thus helps to compensate for the frequency-dependent attenuation along the transmission line. The overshoot introduced by the extra current happens only during a change of state switching to increase the output slew rate and does not ring, unlike the overshoot caused by signal reflection. The amount of pre-emphasis required depends on the attenuation of the high-frequency component along the transmission line.

Figure 6-18: Programmable Pre-Emphasis

This figure shows the LVDS output with pre-emphasis.

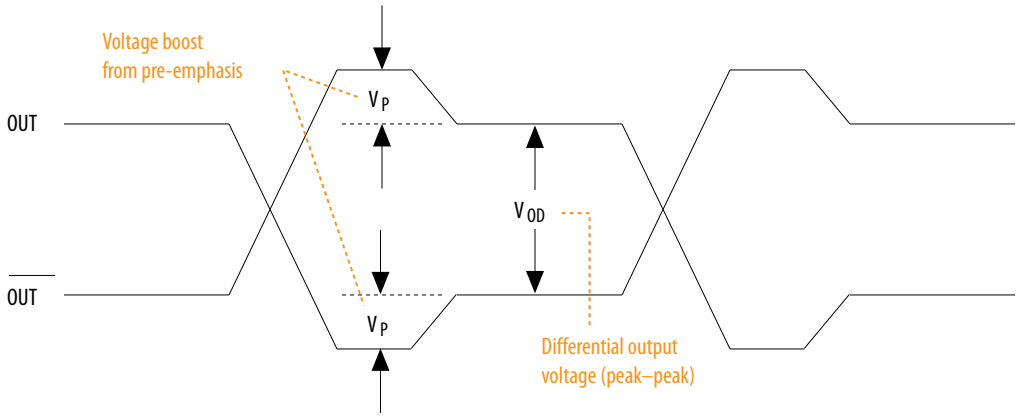


Table 6-11: Quartus II Software Assignment Editor—Programmable Pre-Emphasis

This table lists the assignment name for programmable pre-emphasis and its possible values in the Quartus II software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Pre-emphasis
Allowed values	0 (disabled), 1 (enabled). Default is 1.

Related Information

Programmable IOE Features in Stratix V Devices on page 5-18

Differential Receiver in Stratix V Devices

The receiver has a differential buffer and fractional PLLs that you can share among the transmitter and receiver, a DPA block, a synchronizer, a data realignment block, and a deserializer. The differential buffer can receive LVDS, mini-LVDS, and RSDS signal levels. You can statically set the I/O standard of the receiver pins to LVDS, mini-LVDS, or RSDS in the Quartus II software Assignment Editor.

Note: To drive the LVDS channels, you must use the PLLs in integer PLL mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

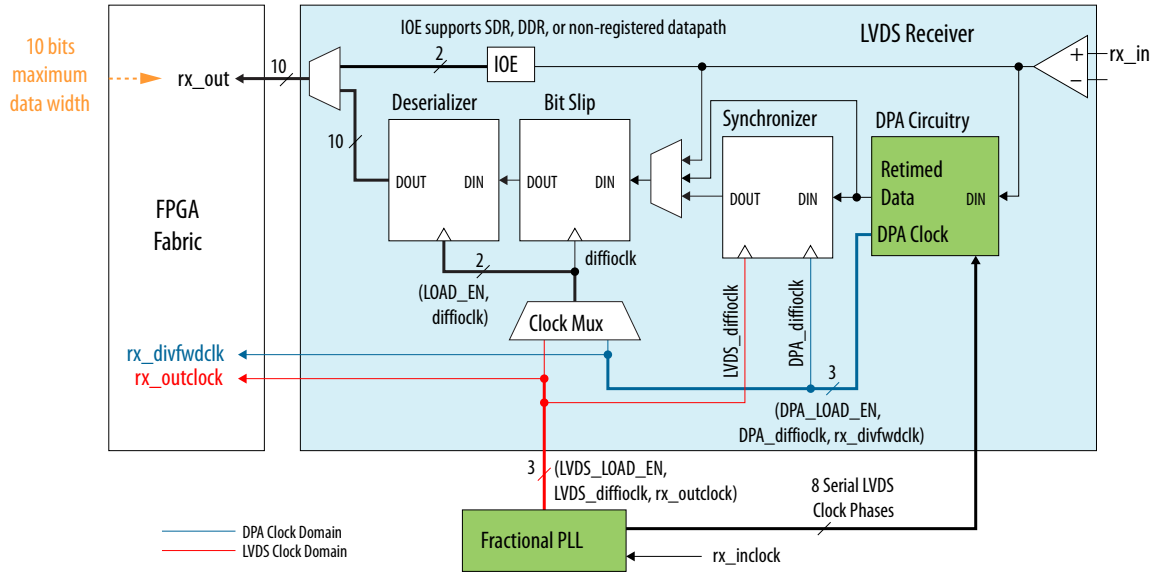
Receiver Blocks in Stratix V Devices

The Stratix V differential receiver has the following hardware blocks:

- DPA block
- Synchronizer
- Data realignment block (bit slip)
- Deserializer

The following figure shows the hardware blocks of the receiver. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively. The deserializer includes shift registers and parallel load registers, and sends a maximum of 10 bits to the internal logic.

Figure 6-19: Receiver Block Diagram

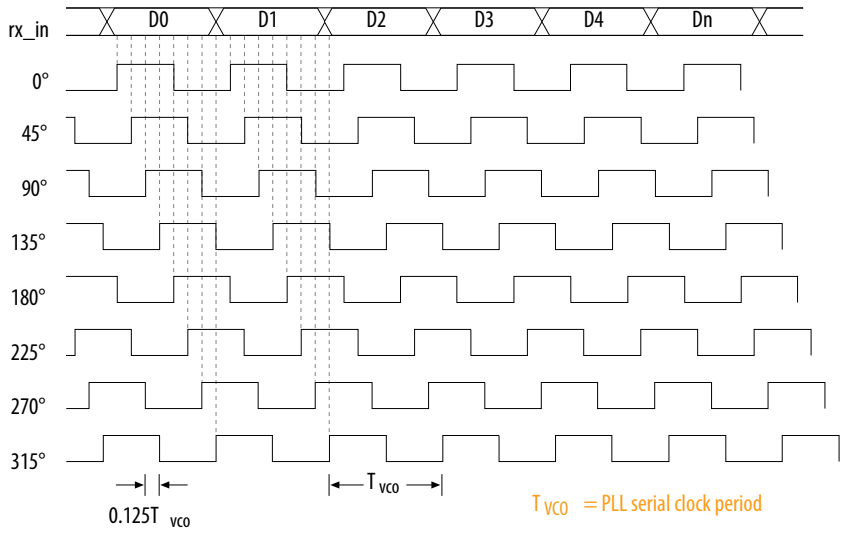


DPA Block

The DPA block takes in high-speed serial data from the differential input buffer and selects one of the eight phases that the fractional PLLs generate to sample the data. The DPA chooses a phase closest to the phase of the serial data. The maximum phase offset between the received data and the selected phase is 1/8 UI, which is the maximum quantization error of the DPA. The eight phases of the clock are equally divided, offering a 45° resolution.

The following figure shows the possible phase relationships between the DPA clocks and the incoming serial data.

Figure 6-20: DPA Clock Phase to Serial Data Timing Relationship



The DPA block continuously monitors the phase of the incoming serial data and selects a new clock phase if it is required. You can prevent the DPA from selecting a new clock phase by asserting the optional `RX_DPLL_HOLD` port, which is available for each channel.

DPA circuitry does not require a fixed training pattern to lock to the optimum phase out of the eight phases. After reset or power up, the DPA circuitry requires transitions on the received data to lock to the optimum phase. An optional output port, `RX_DPA_LOCKED`, is available to indicate an initial DPA lock condition to the optimum phase after power up or reset. This signal is not deasserted if the DPA selects a new phase out of the eight clock phases to sample the received data. Do not use the `rx_dpa_locked` signal to determine a DPA loss-of-lock condition. Use data checkers such as a cyclic redundancy check (CRC) or diagonal interleaved parity (DIP-4) to validate the data.

An independent reset port, `RX_RESET`, is available to reset the DPA circuitry. You must retrain the DPA circuitry after reset.

Note: The DPA block is bypassed in non-DPA mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Synchronizer

The synchronizer is a 1 bit wide and 6 bit deep FIFO buffer that compensates for the phase difference between `DPA_diffioclk`—the optimal clock that the DPA block selects—and the `LVDS_diffioclk` that the fractional PLLs produce. The synchronizer can only compensate for phase differences, not frequency differences, between the data and the receiver's input reference clock.

An optional port, `RX_FIFO_RESET`, is available to the internal logic to reset the synchronizer. The synchronizer is automatically reset when the DPA first locks to the incoming data. Altera recommends using `RX_FIFO_RESET` to reset the synchronizer when the data checker indicates that the received data is corrupted.

Note: The synchronizer circuit is bypassed in non-DPA and soft-CDR mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Data Realignment Block (Bit Slip)

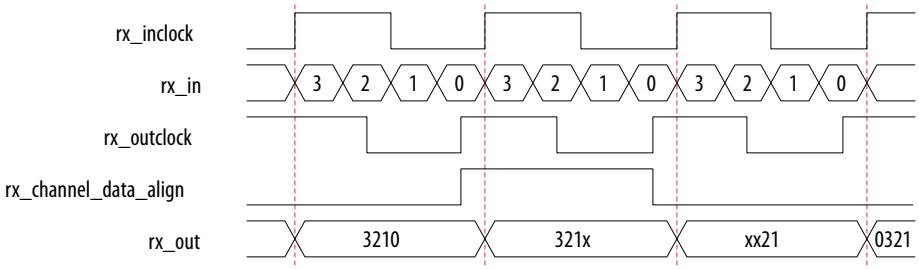
Skew in the transmitted data along with skew added by the link causes channel-to-channel skew on the received serial data streams. If you enable the DPA, the received data is captured with different clock phases on each channel. This difference may cause misalignment of the received data from channel to channel. To compensate for this channel-to-channel skew and establish the correct received word boundary at each channel, each receiver channel has a dedicated data realignment circuit that realigns the data by inserting bit latencies into the serial stream.

An optional `RX_CHANNEL_DATA_ALIGN` port controls the bit insertion of each receiver independently controlled from the internal logic. The data slips one bit on the rising edge of `RX_CHANNEL_DATA_ALIGN`. The requirements for the `RX_CHANNEL_DATA_ALIGN` signal include the following items:

- The minimum pulse width is one period of the parallel clock in the logic array.
- The minimum low time between pulses is one period of the parallel clock.
- The signal is an edge-triggered signal.
- The valid data is available two parallel clock cycles after the rising edge of `RX_CHANNEL_DATA_ALIGN`.

Figure 6-21: Data Realignment Timing

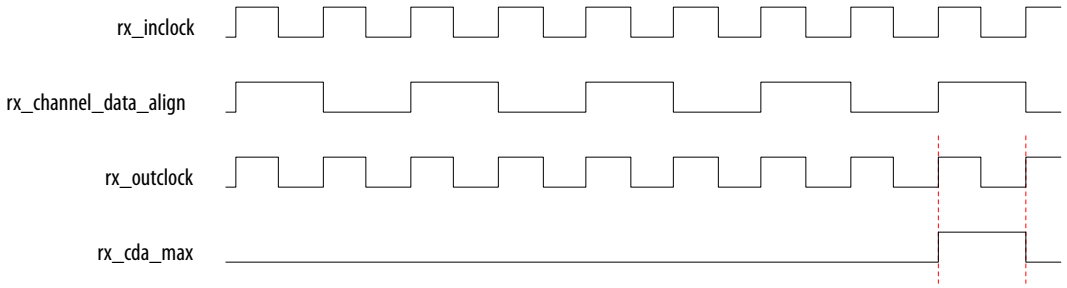
This figure shows receiver output (`RX_OUT`) after one bit slip pulse with the deserialization factor set to **4**.



The data realignment circuit can have up to 11 bit-times of insertion before a rollover occurs. The programmable bit rollover point can be from 1 to 11 bit-times, independent of the deserialization factor. Set the programmable bit rollover point equal to, or greater than, the deserialization factor—allowing enough depth in the word alignment circuit to slip through a full word. You can set the value of the bit rollover point using the MegaWizard Plug-In Manager. An optional status port, `RX_CDA_MAX`, is available to the FPGA fabric from each channel to indicate the reaching of the preset rollover point.

Figure 6-22: Receiver Data Realignment Rollover

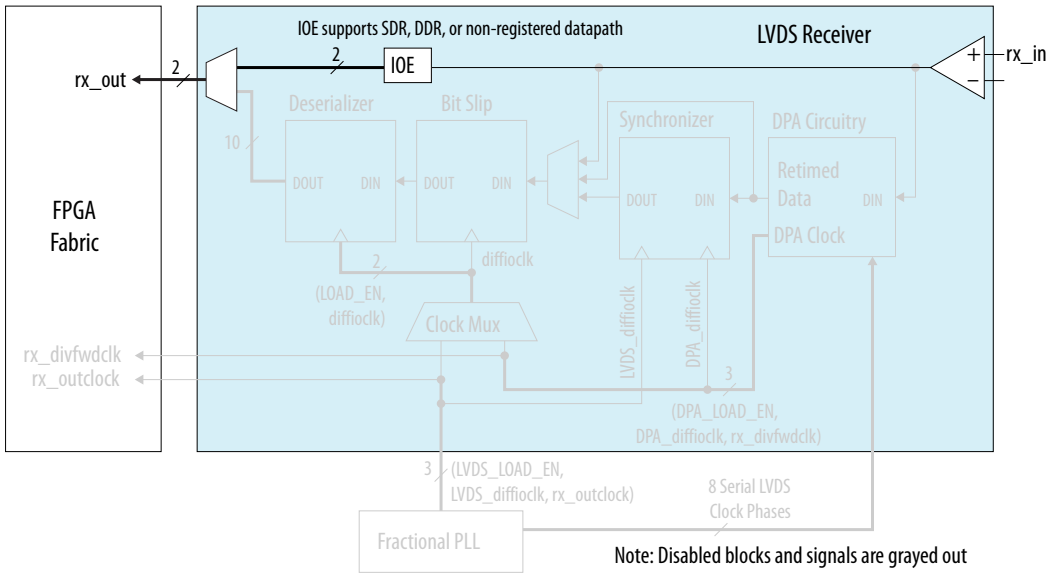
This figure shows a preset value of four bit-times before rollover occurs. The `rx_cda_max` signal pulses for one `rx_outclock` cycle to indicate that rollover has occurred.



Deserializer

You can statically set the deserialization factor to x3, x4, x5, x6, x7, x8, x9, or x10 by using the Quartus II software. You can bypass the deserializer in the Quartus II MegaWizard Plug-In Manager to support DDR (x2) or SDR (x1) operations, as shown in the following figure.

Figure 6-23: Deserializer Bypass



The IOE contains two data input registers that can operate in DDR or SDR mode. In DDR mode, `rx_inclock` clocks the IOE register. In SDR mode, data is directly passed through the IOE. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.

You cannot use the DPA and data realignment circuit when you bypass the deserializer.

Receiver Modes in Stratix V Devices

The Stratix V devices support the following receiver modes:

- Non-DPA mode
- DPA mode
- Soft-CDR mode

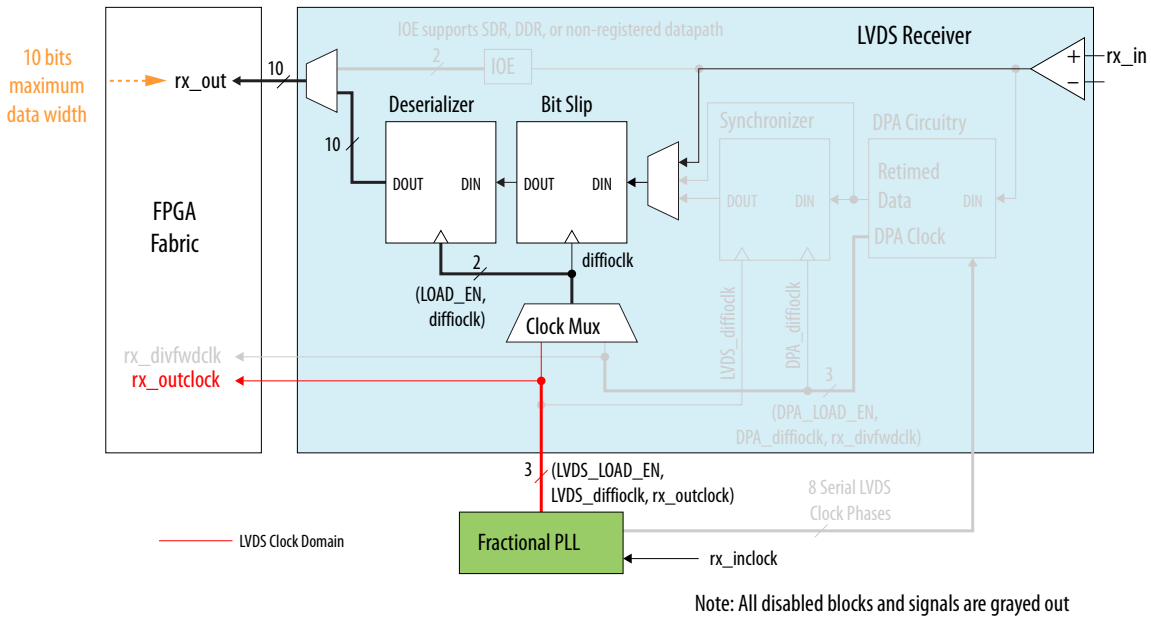
Non-DPA Mode

The non-DPA mode disables the DPA and synchronizer blocks. Input serial data is registered at the rising edge of the serial `LVDS_diffioclk` clock that is produced by the left and right PLLs.

You can select the rising edge option with the Quartus II MegaWizard Plug-In Manager. The `LVDS_diffioclk` clock that is generated by the left and right PLLs clocks the data realignment and deserializer blocks.

The following figure shows the non-DPA datapath block diagram. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.

Figure 6-24: Receiver Data Path in Non-DPA Mode



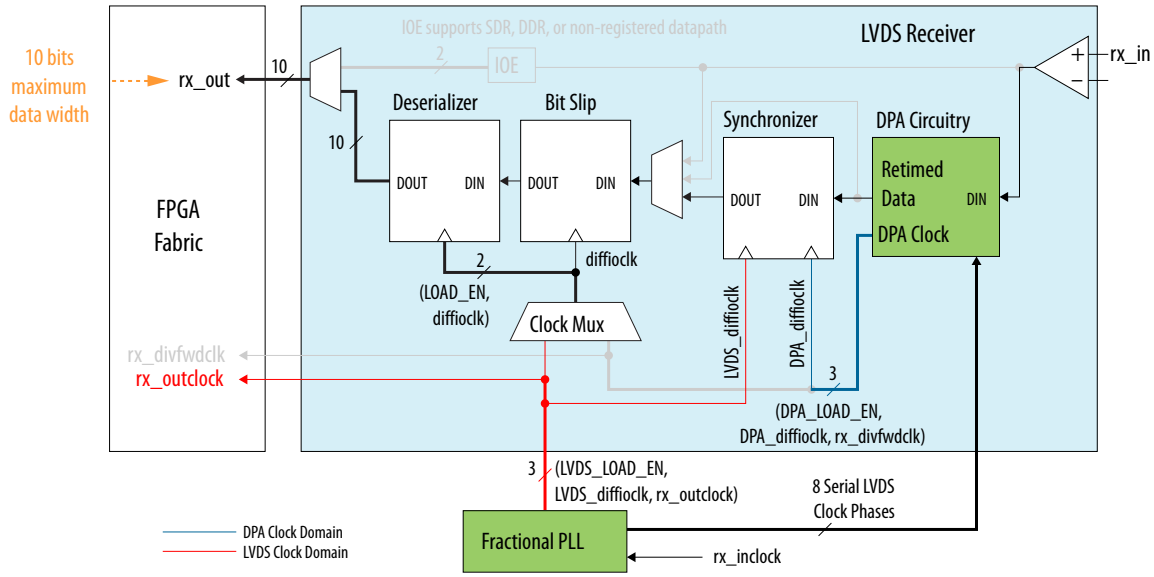
DPA Mode

The DPA block chooses the best possible clock (`DPA_diffioclk`) from the eight fast clocks that the fractional PLL sent. This serial `DPA_diffioclk` clock is used for writing the serial data into the synchronizer. A serial `LVDS_diffioclk` clock is used for reading the serial data from the synchronizer. The same `LVDS_diffioclk` clock is used in data realignment and deserializer blocks.

The following figure shows the DPA mode datapath. In the figure, all the receiver hardware blocks are active.

Figure 6-25: Receiver Datapath in DPA Mode

In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.



Related Information

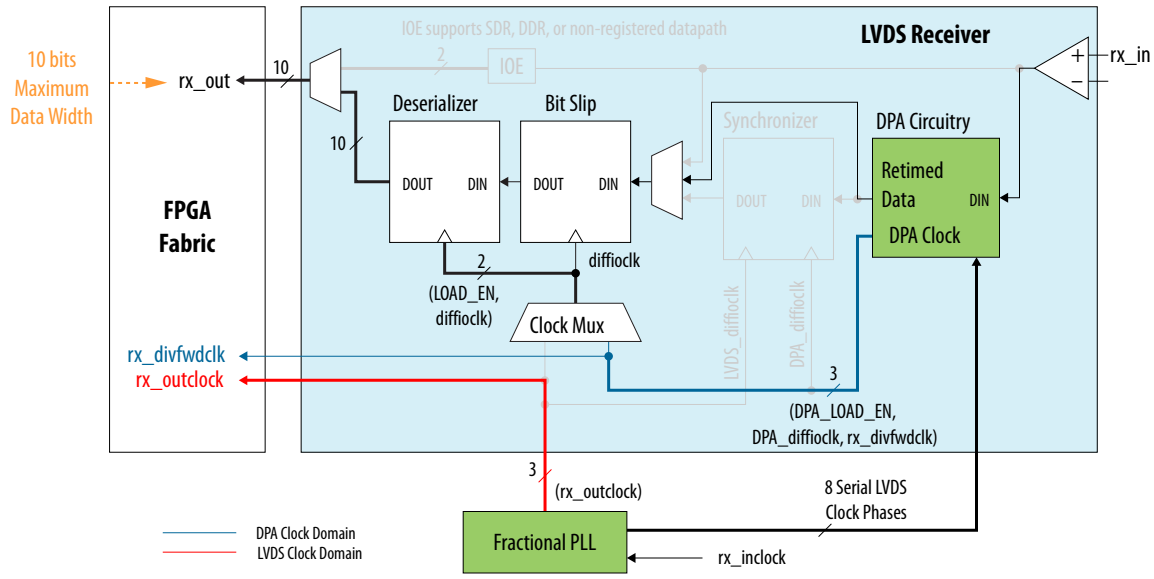
- **Receiver Blocks in Stratix V Devices** on page 6-22
  Lists and describes the receiver hardware blocks.
- **Guideline: Use PLLs in Integer PLL Mode for LVDS** on page 6-8

Soft-CDR Mode

The Stratix V LVDS channel offers the soft-CDR mode to support the GbE and SGMII protocols. A receiver PLL uses the local clock source for reference.

The following figure shows the soft-CDR mode datapath. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.

Figure 6-26: Receiver Datapath in Soft-CDR Mode



Note: All disabled blocks and signals are grayed out

In soft-CDR mode, the synchronizer block is inactive. The DPA circuitry selects an optimal DPA clock phase to sample the data. Use the selected DPA clock for bit-slip operation and deserialization. The DPA block also forwards the selected DPA clock, divided by the deserialization factor called `rx_divfwdclk`, to the FPGA fabric, along with the deserialized data. This clock signal is put on the periphery clock (PCLK) network.

If you use the soft-CDR mode, do not assert the `rx_reset` port after the DPA has trained. The DPA continuously chooses new phase taps from the PLL to track parts per million (PPM) differences between the reference clock and incoming data.

You can use every LVDS channel in soft-CDR mode and drive the FPGA fabric using the PCLK network in the Stratix V device family. The `rx_dpa_locked` signal is not valid in soft-CDR mode because the DPA continuously changes its phase to track PPM differences between the upstream transmitter and the local receiver input reference clocks. The parallel clock, `rx_outclock`, generated by the left and right PLLs, is also forwarded to the FPGA fabric.

Related Information

Periphery Clock Networks on page 4-4
Provides more information about PCLK networks.

Receiver Clocking for Stratix V Devices

The fractional PLL receives the external clock input and generates different phases of the same clock. The DPA block automatically chooses one of the clocks from the fractional PLL and aligns the incoming data on each channel.

The synchronizer circuit is a 1 bit wide by 6 bit deep FIFO buffer that compensates for any phase difference between the DPA clock and the data realignment block. If necessary, the user-controlled data realignment circuitry inserts a single bit of latency in the serial bit stream to align to the word boundary.

The deserializer includes shift registers and parallel load registers, and sends a maximum of 10 bits to the internal logic.

The physical medium connecting the transmitter and receiver LVDS channels may introduce skew between the serial data and the source-synchronous clock. The instantaneous skew between each LVDS channel and the clock also varies with the jitter on the data and clock signals as seen by the receiver. The three different modes—non-DPA, DPA, and soft-CDR—provide different options to overcome skew between the source synchronous clock (non-DPA, DPA) /reference clock (soft-CDR) and the serial data.

Non-DPA mode allows you to statically select the optimal phase between the source synchronous clock and the received serial data to compensate skew. In DPA mode, the DPA circuitry automatically chooses the best phase to compensate for the skew between the source synchronous clock and the received serial data. Soft-CDR mode provides opportunities for synchronous and asynchronous applications for chip-to-chip and short reach board-to-board applications for SGMII protocols.

Note: Only the non-DPA mode requires manual skew adjustment.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Differential I/O Termination for Stratix V Devices

The Stratix V devices provide a 100 Ω, on-chip differential termination option on each differential receiver channel for LVDS standards. On-chip termination saves board space by eliminating the need to add external resistors on the board. You can enable on-chip termination in the Quartus II software Assignment Editor.

All I/O pins and dedicated clock input pins support on-chip differential termination, R_D OCT.

Figure 6-27: On-Chip Differential I/O Termination

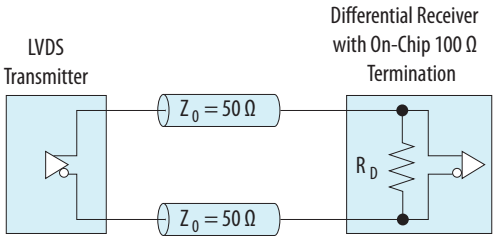


Table 6-12: Quartus II Software Assignment Editor—On-Chip Differential Termination

This table lists the assignment name for on-chip differential termination in the Quartus II software Assignment Editor.

Field	Assignment
To	rx_in
Assignment name	Input Termination
Value	Differential

Source-Synchronous Timing Budget

The topics in this section describe the timing budget, waveforms, and specifications for source-synchronous signaling in the Stratix V device family.

The LVDS I/O standard enables high-speed transmission of data, resulting in better overall system performance. To take advantage of fast system performance, you must analyze the timing for these high-speed signals. Timing analysis for the differential block is different from traditional synchronous timing analysis techniques.

The basis of the source synchronous timing analysis is the skew between the data and the clock signals instead of the clock-to-output setup times. High-speed differential data transmission requires the use of timing parameters provided by IC vendors and is strongly influenced by board skew, cable skew, and clock jitter.

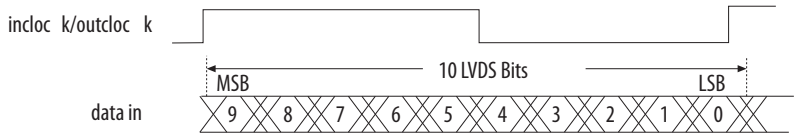
This section defines the source-synchronous differential data orientation timing parameters, the timing budget definitions for the Stratix V device family, and how to use these timing parameters to determine the maximum performance of a design.

Differential Data Orientation

There is a set relationship between an external clock and the incoming data. For operations at 1 Gbps and a serialization factor of 10, the external clock is multiplied by 10. You can set phase-alignment in the PLL to coincide with the sampling window of each data bit. The data is sampled on the falling edge of the multiplied clock.

Figure 6-28: Bit Orientation in the Quartus II Software

This figure shows the data bit orientation of the x10 mode.



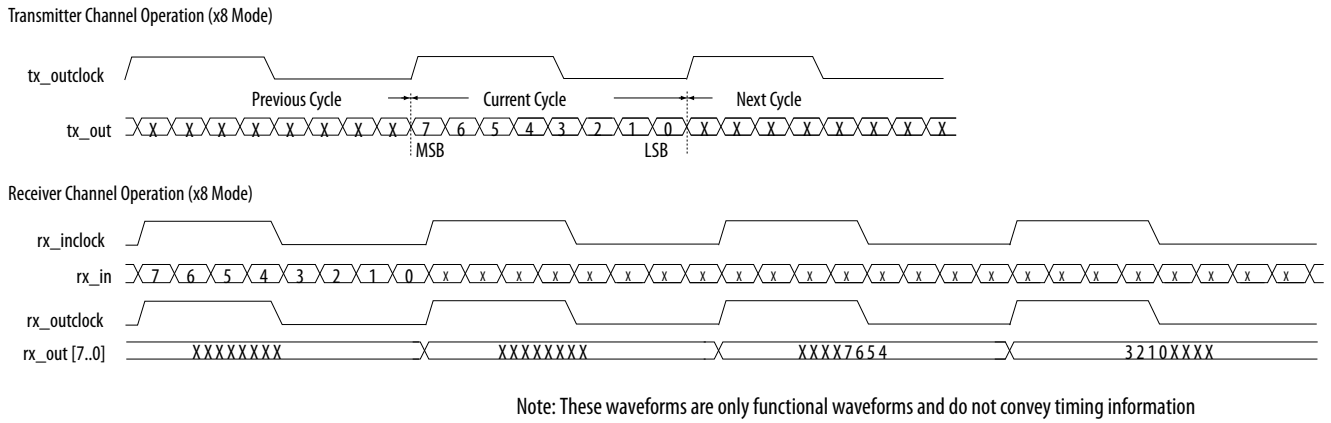
Differential I/O Bit Position

Data synchronization is necessary for successful data transmission at high frequencies.

The following figure shows the data bit orientation for a channel operation and is based on the following conditions:

- The serialization factor is equal to the clock multiplication factor.
- The phase alignment uses edge alignment.
- The operation is implemented in hard SERDES.

Figure 6-29: Bit-Order and Word Boundary for One Differential Channel



Note: These waveforms are only functional waveforms and do not convey timing information

For other serialization factors, use the Quartus II software tools to find the bit position within the word.

Differential Bit Naming Conventions

The following table lists the conventions for differential bit naming for 18 differential channels. The MSB and LSB positions increase with the number of channels used in a system.

Table 6-13: Differential Bit Naming

This table lists the conventions for differential bit naming for 18 differential channels. The MSB and LSB positions increase with the number of channels used in a system.

Receiver Channel Data Number	Internal 8-Bit Parallel Data	
	MSB Position	LSB Position
1	7	0
2	15	8
3	23	16
4	31	24
5	39	32
6	47	40
7	55	48
8	63	56
9	71	64
10	79	72
11	87	80
12	95	88
13	103	96
14	111	104

Receiver Channel Data Number	Internal 8-Bit Parallel Data	
	MSB Position	LSB Position
15	119	112
16	127	120
17	135	128
18	143	136

Transmitter Channel-to-Channel Skew

The receiver skew margin calculation uses the transmitter channel-to-channel skew (TCCS)—an important parameter based on the Stratix V transmitter in a source-synchronous differential interface:

- TCCS is the difference between the fastest and slowest data output transitions, including the T_{CO} variation and clock skew.
- For LVDS transmitters, the TimeQuest Timing Analyzer provides the TCCS value in the TCCS report (`report_TCCS`) in the Quartus II compilation report, which shows TCCS values for serial output ports.
- You can also get the TCCS value from the device datasheet.

Note: For the Stratix V devices, perform PCB trace compensation to adjust the trace length of each LVDS channel to improve channel-to-channel skew when interfacing with non-DPA receivers at data rate above 840 Mbps.

The Quartus II software Fitter Report panel reports the amount of delay you must add to each trace for the Stratix V device. You can use the recommended trace delay numbers published under the LVDS Transmitter/Receiver Package Skew Compensation panel and manually compensate the skew on the PCB board trace to reduce channel-to-channel skew, thus meeting the timing budget between LVDS channels.

Related Information

- **Stratix V Device Datasheet**
- **LVDS SERDES Transmitter/Receiver (ALTLVDS_TX and ALTLVDS_RX) Megafunction User Guide**
  Provides more information about the LVDS Transmitter/Receiver Package Skew Compensation report panel.

Receiver Skew Margin for Non-DPA Mode

Different modes of LVDS receivers use different specifications, which can help in deciding the ability to sample the received serial data correctly:

- In DPA mode, use DPA jitter tolerance instead of the receiver skew margin (RSKM).
- In non-DPA mode, use RSKM, TCCS, and sampling window (SW) specifications for high-speed source-synchronous differential signals in the receiver data path.

The following equation expresses the relationship between RSKM, TCCS, and SW.

Figure 6-30: RSKM Equation

$$RSKM = \frac{TUI - SW - TCCS}{2}$$

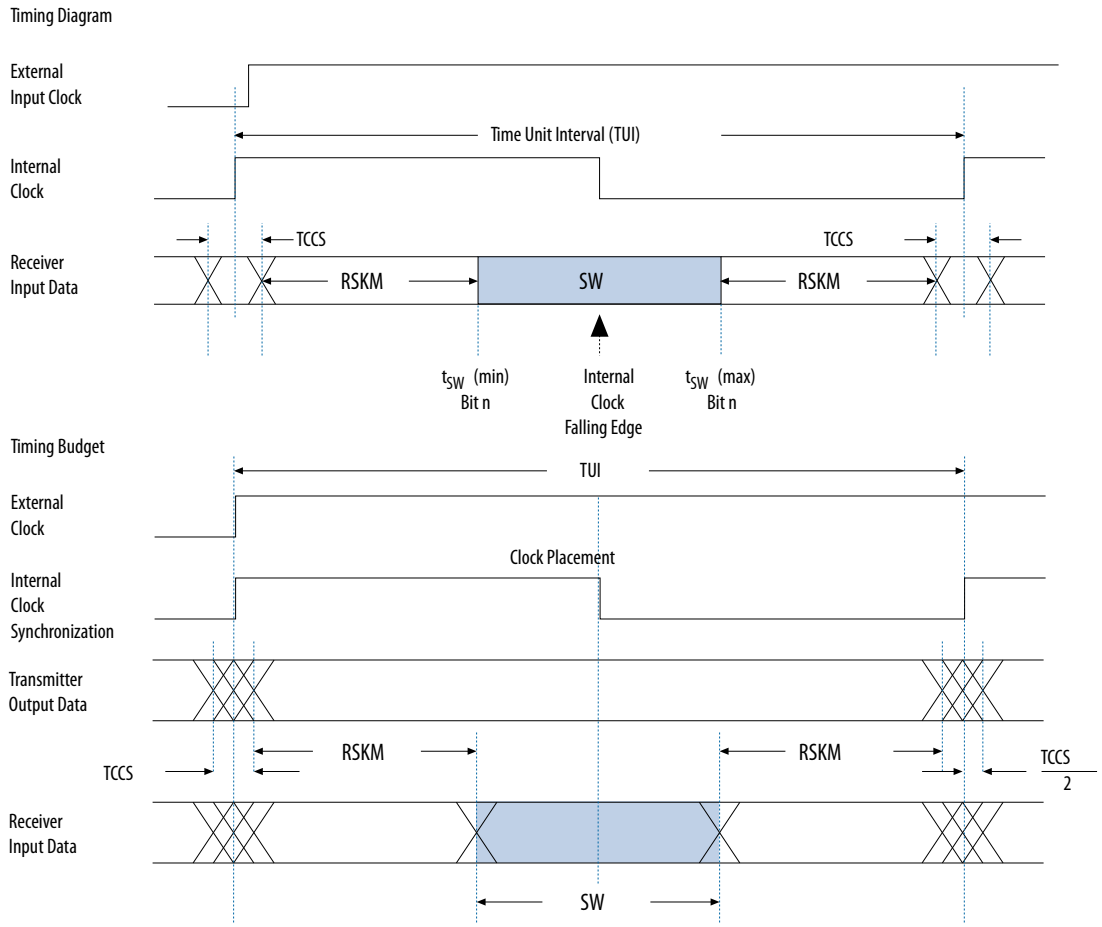
Conventions used for the equation:

- RSKM—the timing margin between the receiver's clock input and the data input sampling window.
- Time unit interval (TUI)—time period of the serial data.
- SW—the period of time that the input data must be stable to ensure that data is successfully sampled by the LVDS receiver. The SW is a device property and varies with device speed grade.
- TCCS—the timing difference between the fastest and the slowest output edges, including t_{CO} variation and clock skew, across channels driven by the same PLL. The clock is included in the TCCS measurement.

You must calculate the RSKM value to decide whether the LVDS receiver can sample the data properly or not, given the data rate and device. A positive RSKM value indicates that the LVDS receiver can sample the data properly, whereas a negative RSKM indicates that it cannot sample the data properly.

The following figure shows the relationship between the RSKM, TCCS, and the SW of the receiver.

Figure 6-31: Differential High-Speed Timing Diagram and Timing Budget for Non-DPA Mode



For LVDS receivers, the Quartus II software provides an RSKM report showing the SW, TUI, and RSKM values for non-DPA LVDS mode:

- You can generate the RSKM report by executing the `report_RSKM` command in the TimeQuest Timing Analyzer. You can find the RSKM report in the Quartus II compilation report in the TimeQuest Timing Analyzer section.
- To obtain the RSKM value, assign the input delay to the LVDS receiver through the constraints menu of the TimeQuest Timing Analyzer. The input delay is determined according to the data arrival time at the LVDS receiver port, with respect to the reference clock.
- If you set the input delay in the settings parameters for the **Set Input Delay** option, set the clock name to the clock that reference the source synchronous clock that feeds the LVDS receiver.
- If you do not set any input delay in the TimeQuest Timing Analyzer, the receiver channel-to-channel skew defaults to zero.
- You can also directly set the input delay in a Synopsys Design Constraint file (**.sdc**) using the `set_input_delay` command.

Example 6-1: RSKM Calculation Example

This example shows the RSKM calculation for Stratix V devices at 1 Gbps data rate with a 200 ps board channel-to-channel skew.

- TCCS = 100 ps (pending characterization)
- SW = 300 ps (pending characterizatoin
- TUI = 1000 ps
- Total RCCS = TCCS + Board channel-to-channel skew = 100 ps + 200 ps = 300 ps
- RSKM = (TUI – SW – RCCS) / 2 = (1000 ps – 300 ps – 300 ps) / 2 = 200 ps

Because the RSKM is greater than 0 ps, the receiver non-DPA mode will work correctly.

Related Information

- **LVDS SERDES Transmitter/Receiver (ALTLVDS_TX and ALTLVDS_RX) Megafunction User Guide**
  Provides more information about the RSKM equation and calculation.
- **Quartus II TimeQuest Timing Analyzer chapter, Quartus II Development Software Handbook**
  Provides more information about **.sdc** commands and the TimeQuest Timing Analyzer.

Assigning Input Delay to LVDS Receiver Using TimeQuest Timing Analyzer

To obtain the RSKM value, assign an appropriate input delay to the LVDS receiver from the TimeQuest Timing Analyzer constraints menu.

1. On the menu in the TimeQuest Timing Analyzer, select **Constraints** > **Set Input Delay**.
2. In the **Set Input Delay** window, select the desired clock using the pull-down menu. The clock name must reference the source synchronous clock that feeds the LVDS receiver.
3. Click the **Browse** button (next to the **Targets** field).
4. In the **Name Finder** window, click **List** to view a list of all available ports. Select the LVDS receiver serial input ports according to the input delay you set, and click **OK**.
5. In the **Set Input Delay** window, set the appropriate values in the **Input delay** options and **Delay value** fields.
6. Click **Run** to incorporate these values in the TimeQuest Timing Analyzer.
7. Repeat from **step 1** to assign the appropriate delay for all the LVDS receiver input ports. If you have already assigned Input Delay and you need to add more delay to that input port, turn on the **Add Delay** option.

Document Revision History

Date	Version	Changes
January 2015	2015.01.23	• Removed statement on explanation related to rx_synclock for figure "LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode)". • Updated figure LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode) and figure Receiver Datapath in Soft-CDR Mode. • Added a note to leave rx_enable and rx_inclock to be unconnected for figure LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode). • Updated timing diagram for Phase Relationship for External PLL Interface Signals to reflect the correct phase shift and frequency for outclk2.
January 2014	2014.01.10	• Updated the statement about setting the phase of the clock in relation to data in the topic about transmitter clocking. • Updated the figure that shows the phase relationship for the external PLL interface signals. • Clarified that "one row of separation" between two groups of DPA-enabled channels means a separation of one differential channel. • Clarified that "internal PLL option" refers to the option in the ALTLVDS megafunction. • Updated the topic about emulated LVDS buffers to clarify that you can use unutilized true LVDS input channels (instead "buffers") as emulated LVDS output buffers.
June 2013	2013.06.21	Updated the figure about data realignment timing to correct the data pattern after a bit slip.

Date	Version	Changes
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Removed all references to column and row I/Os. Stratix V devices have I/O banks on the top and bottom only. • Changed the color of the transceiver blocks in the high-speed differential I/O location diagram for clarity. • Updated the pin placement guidelines section to add figures and new topic about using DPA-disabled differential channels. • Added a topic about emulated LVDS buffers. • Edited the topic about true LVDS buffers. • Added a topic that lists the SERDES I/O standards support and the respective Quartus II assignment values. • Corrected the outclk2 waveform in **Figure 6-4** to show -18° phase shift (as labeled). • Clarified that the programmable V_{OD} assignment value of "0" is also applicable for mini-LVDS. • Updated the data realignment timing figure to improve clarity. • Updated the receiver data realignment rollover figure to improve clarity.
December 2012	2012.12.28	• Reorganized content and updated template. • Added Altera_PLL settings for external PLL usage in DPA and non-DPA modes. • Moved the PLL and clocking section into design guideline topics. • Updated external PLL clocking examples without DPA and soft-CDR. Altera_PLL now supports entering negative phase shift. • Added external PLL clocking example and settings for DPA and soft-CDR mode. • Updated the LVDS channel tables to list the number of channels per side for each device package instead of just for the largest package. • Removed the “LVDS Direct Loopback Mode” section.
June 2012	1.4	• Added Table 6–2. • Updated Table 6–1, Table 6–3, Table 6–4, and Table 6–5. • Updated Figure 6–21. • Updated “Non-DPA Mode”, “Soft-CDR Mode”, and “PLLs and Stratix V Clocking” sections.
November 2011	1.3	• Updated Table 6–2. • Updated Example 6–1. • Updated “LVDS Direct Loopback Mode” and “LVDS Interface with the Use External PLL Option Enabled” sections.

Date	Version	Changes
May 2011	1.2	• Chapter moved to volume 2 for the 11.0 release. • Added Table 6–2 and Table 6–3. • Updated Table 6–1. • Updated Figure 6–2 and Figure 6–23. • Updated "Locations of the I/O Banks", "Programmable Pre-Emphasis", "Differential Receiver", "Fractional PLLs and Stratix V Clocking", and "DPA-Enabled Channels, DPA-Disabled Channels, and Single-Ended I/Os" sections. • Minor text edits.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

External Memory Interfaces in Stratix V Devices

7

2015.01.23

SV51008 Subscribe Send Feedback

The Stratix V devices provide an efficient architecture that allows you to fit wide external memory interfaces to support a high level of system bandwidth within the small modular I/O bank structure. The I/Os are designed to provide high-performance support for existing and emerging external memory standards.

Table 7-1: Supported External Memory Standards in Stratix V Devices

Memory Standard	Soft Memory Controller
DDR3 SDRAM	Half rate and quarter rate
DDR2 SDRAM	Full rate and half rate
RLDRAM 3	Half rate and quarter rate
RLDRAM II	Full rate and half rate
QDR II+ SRAM	Full rate and half rate
QDR II SRAM	Full rate and half rate

Related Information

- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the *Stratix V Device Handbook* chapters.
- **External Memory Interface Spec Estimator**
  For the latest information and to estimate the external memory system performance specification, use Altera's External Memory Interface Spec Estimator tool.
- **External Memory Interface Handbook**
  Provides more information about the memory types supported, board design guidelines, timing analysis, simulation, and debugging information.

External Memory Performance

Table 7-2: External Memory Interface Performance in Stratix V Devices

Interface	Voltage (V)	Soft Controller (MHz)
DDR3 SDRAM	1.5	933
	1.35	800
DDR2 SDRAM	1.8	400
RLDRAM 3	1.2	800
RLDRAM II	1.8	533
	1.5	533
QDR II+ SRAM	1.8	550
	1.5	550
QDR II SRAM	1.8	350
	1.5	350

Related Information

External Memory Interface Spec Estimator
For the latest information and to estimate the external memory system performance specification, use Altera's External Memory Interface Spec Estimator tool.

Memory Interface Pin Support in Stratix V Devices

In the Stratix V devices, the memory interface circuitry is available in every I/O bank that does not support transceivers. The devices offer differential input buffers for differential read-data strobe and clock operations.

Stratix V devices also provide an independent DQS logic block for each CQn pin for complementary read-data strobe and clock operations

The memory clock pins are generated with double data rate input/output (DDRIO) registers.

Related Information

Planning Pin and FPGA Resources chapter, External Memory Interface Handbook
Provides more information about which pins to use for memory clock pins and pin location requirements.

Guideline: Using DQ/DQS Pins

The following list provides guidelines on using the DQ/DQS pins:

- The devices support DQ and DQS signals with DQ bus modes of x4, x8/x9, x16/x18, or x32/x36.
- You can use the DQSn or CQn pins that are not used for clocking as DQ pins.
- If you do not use the DQ/DQS pins for memory interfacing, you can use these pins as user I/Os.
- Some pins have multiple functions such as RZQ or DQ. If you need extra RZQ pins, you can use the DQ/DQNs pins in some of the x4 groups as RZQ pins instead.
- You cannot use a x4 DQ/DQS group for memory interfaces if any of its members are used as RZQ pins for OCT calibration.
- There is no restriction on using x8/x9, x16/x18, or x32/x36 DQ/DQS groups that include the x4 groups whose pins are used as RZQ pins because there are enough extra pins that you can use as DQS pins.

Note: For the x8, x16/x18, or x32/x36 DQ/DQS groups whose members are used as RZQ pins, Altera recommends that you assign the DQ and DQS pins manually. Otherwise, the Quartus II software might not be able to place the DQ and DQS pins, resulting in a "no-fit" error.

DQ pins can be bidirectional signals, as in DDR3 and DDR2 SDRAM, and RLDRAM II common I/O interfaces, or unidirectional signals, as in QDR II+ and QDR II SRAM, and RLDRAM II separate I/O devices. Connect the unidirectional read-data signals to Stratix V DQ pins and the unidirectional write-data signals to a different DQ/DQS group than the read DQ/DQS group. You must assign the write clocks to the DQS/DQSn pins associated to this write DQ/DQS group. Do not use the CQ/CQn pin-pair for write clocks.

Note: Using a DQ/DQS group for the write-data signals minimizes output skew, allows access to the write-leveling circuitry (for DDR3 SDRAM interfaces), and allows vertical migration. These pins also have access to deskewing circuitry (using programmable delay chains) that can compensate for delay mismatch between signals on the bus.

Reading the Pin Table

For the maximum number of DQ pins and the exact number per group for a particular Stratix V device, refer to the pin table in the Stratix V page of the Altera website. In the pin tables, the DQS and DQSn pins denote the differential data strobe/clock pin pairs, while the CQ and CQn pins denote the complementary echo clock signals. The pin table lists the parity, DM, BWSn, NWSn, ECC, and QVLD pins as DQ pins.

In the Stratix V pin tables, DQSn and CQn pins are marked separately. Each CQn pin connects to a DQS logic block and the phase-shifted CQn signals go to the negative half cycle input registers in the DQ IOE registers.

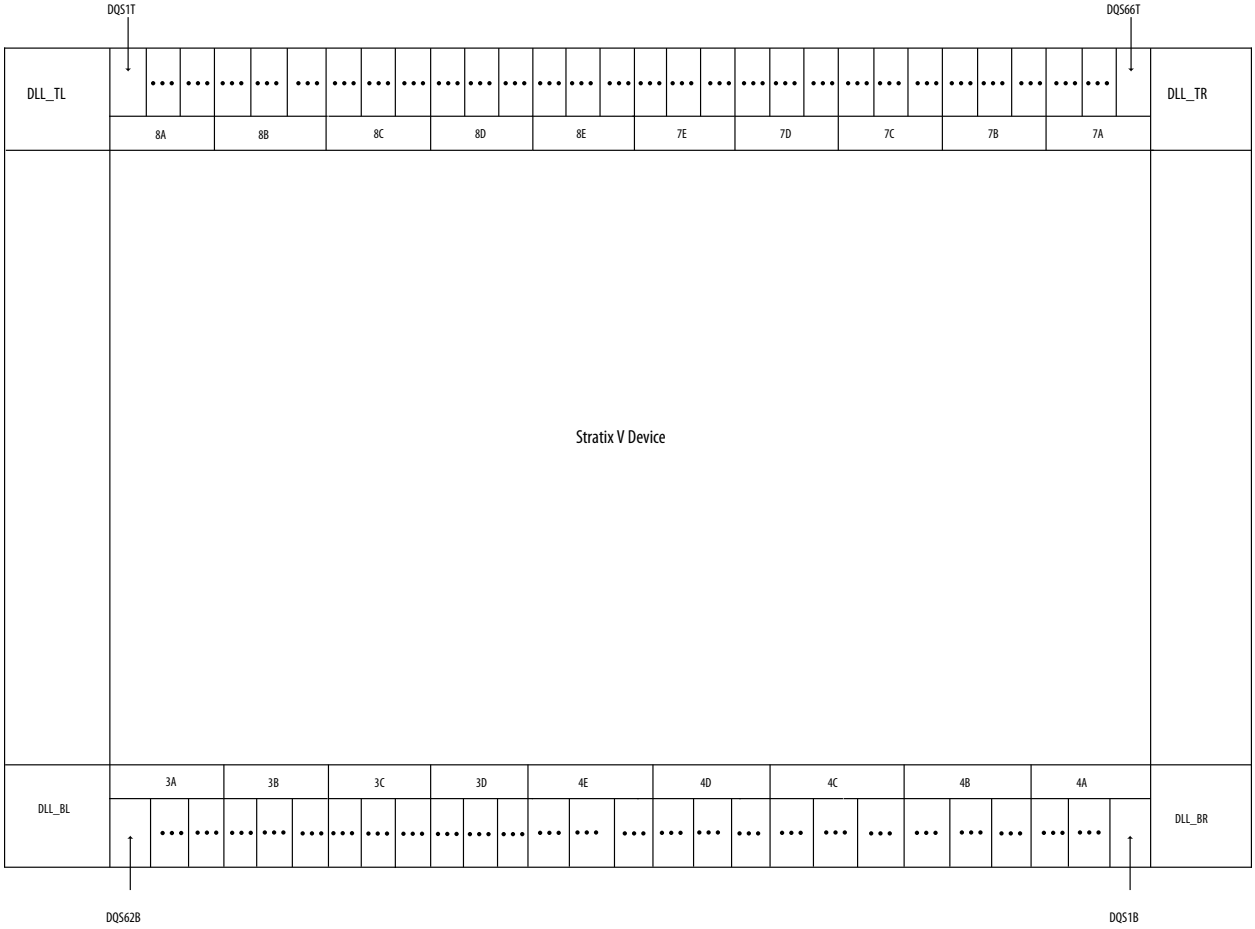
The DQS and DQSn pins are listed respectively in the Stratix V pin tables as `DQS`*XY* and `DQSn`*XY*. *X* indicates the DQ/DQS grouping number and *Y* indicates whether the group is located on the top (T), bottom (B), left (L), or right (R) side of the device. The DQ/DQS pin numbering is based on the x4 mode.

The corresponding DQ pins are marked as `DQ`*XY*, where *X* indicates which DQS group the pins belong to and *Y* indicates whether the group is located on the top (T) or bottom (B) side of the device.

For example, `DQS1T` indicates a DQS pin located on the top side of the device. The DQ pins belonging to that group are shown as `DQ1T` in the pin table.

Figure 7-1: DQS Pins in Stratix V I/O Banks

This figure shows the DQ/DQS groups numbering in a die-top view of the device where the numbering scheme starts from the top-left corner of the device going clockwise.



DQ/DQS Bus Mode Pins for Stratix V Devices

The following table list the pin support per DQ/DQS bus mode, including the DQS/CQ and DQSn/CQn pins. The maximum number of data pins per group listed in the table may vary according to the following conditions:

- Single-ended DQS signaling—the maximum number of DQ pins includes parity, data mask, and QVLD pins connected to the DQS bus network.
- Differential or complementary DQS signaling—the maximum number of data pins per group decreases by one. This number may vary per DQ/DQS group in a particular device. Check the pin table for the exact number per group.
- DDR3 and DDR2 interfaces—the maximum number of pins is further reduced for an interface larger than x8 because you require one DQS pin for each x8/x9 group to form the x16/x18 and x32/x36 groups.

Table 7-3: DQ/DQS Bus Mode Pins for Stratix V Devices

Mode	DQSn Support	CQn Support	Parity or Data Mask (Optional)	QVLD [9] (Optional)	Data Pins per Group		Notes
					Typical	Maximum	
x4	Yes	—	—	—	4	5	If you do not use differential DQS and the group does not have additional signals, the data mask (DM) pin is supported.
x8/x9	Yes	Yes	Yes	Yes	8 or 9	11	Two x4 DQ/DQS groups are stitched to create a x8/x9 group, so there are a total of 12 pins in this group.
x16/x18	Yes	Yes	Yes	Yes	16 or 18	23	Four x4 DQ/DQS groups are stitched to create a x16/x18 group; so there are a total of 24 pins in this group.
x32/x36	Yes	Yes	Yes	Yes	32 or 36	47	Eight x4 DQ/DQS groups are stitched to create a x32/x36 group, so there are a total of 48 pins in this group.

DQ/DQS Groups in Stratix V E

Table 7-4: Number of DQ/DQS Groups Per Side in Stratix V E Devices

Some of the x4 groups are using RZQ pins. If you use the Stratix V calibrated OCT feature, you cannot use these groups.

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
E9	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6
EB	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6

DQ/DQS Groups in Stratix V GX

[9] The QVLD pin is not used in the UniPHY megafunction.

Table 7-5: Number of DQ/DQS Groups Per Side in Stratix V GX Devices

Some of the x4 groups are using RZQ pins. If you use the Stratix V calibrated OCT feature, you cannot use these groups.

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
A3	780-pin FineLine BGA	Top	34	13	8	2
		Bottom	26	17	6	1
	1152-pin FineLine BGA (with 24 transceivers)	Top	42	21	10	3
		Bottom	50	25	12	4
	1152-pin FineLine BGA (with 36 transceivers)	Top	36	18	8	2
		Bottom	36	18	8	2
	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
A4	1152-pin FineLine BGA (with 24 transceivers)	Top	42	21	10	3
		Bottom	50	25	12	4
	1152-pin FineLine BGA (with 36 transceivers)	Top	36	18	8	2
		Bottom	36	18	8	2
	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
A5	1152-pin FineLine BGA (with 24 transceivers)	Top	42	21	10	3
		Bottom	50	25	12	4
	1152-pin FineLine BGA (with 36 transceivers)	Top	36	18	8	2
		Bottom	36	18	8	2
	1517-pin FineLine BGA (with 36 transceivers)	Top	58	29	14	6
		Bottom	58	29	14	6
	1517-pin FineLine BGA (with 48 transceivers)	Top	50	25	12	5
		Bottom	50	25	12	4
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
A7	1152-pin FineLine BGA (with 24 transceivers)	Top	42	21	10	3
		Bottom	50	25	12	4
	1152-pin FineLine BGA (with 36 transceivers)	Top	36	18	8	2
		Bottom	36	18	8	2
	1517-pin FineLine BGA (with 36 transceivers)	Top	58	29	14	6
		Bottom	58	29	14	6
	1517-pin FineLine BGA (with 48 transceivers)	Top	50	25	12	5
		Bottom	50	25	12	4
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6
A9	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6
AB	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6
B5	1517-pin FineLine BGA	Top	36	18	8	3
		Bottom	36	18	8	3
	1760-pin FineLine BGA	Top	50	25	11	4
		Bottom	50	25	11	4
B6	1517-pin FineLine BGA	Top	36	18	8	3
		Bottom	36	18	8	3
	1760-pin FineLine BGA	Top	50	25	11	4
		Bottom	50	25	11	4
B9	1760-pin FineLine BGA	Top	50	25	11	3
		Bottom	50	25	11	3
BB	1760-pin FineLine BGA	Top	50	25	11	3
		Bottom	50	25	11	3

DQ/DQS Groups in Stratix V GS

Table 7-6: Number of DQ/DQS Groups Per Side in Stratix V GS Devices

Some of the x4 groups are using RZQ pins. If you use the Stratix V calibrated OCT feature, you cannot use these groups.

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
D3	780-pin FineLine BGA	Top	34	13	8	2
		Bottom	26	17	6	1
	1152-pin FineLine BGA	Top	38	19	9	2
		Bottom	34	17	8	2
D4	780-pin FineLine BGA	Top	34	13	8	2
		Bottom	26	17	6	1
	1152-pin FineLine BGA	Top	38	19	9	2
		Bottom	34	17	8	2
	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
D5	1152-pin FineLine BGA	Top	42	21	10	3
		Bottom	50	25	12	4
	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
D6	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6
D8	1517-pin FineLine BGA	Top	58	29	14	6
		Bottom	58	29	14	6
	1932-pin FineLine BGA	Top	70	35	16	6
		Bottom	70	35	16	6

DQ/DQS Groups in Stratix V GT

Table 7-7: Number of DQ/DQS Groups Per Side in Stratix V GT Devices

Some of the x4 groups are using RZQ pins. If you use the Stratix V calibrated OCT feature, you cannot use these groups.

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
C5	1517-pin FineLine BGA	Top	50	25	12	5
		Bottom	50	25	12	4

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
C7	1517-pin FineLine BGA	Top	50	25	12	5
		Bottom	50	25	12	4

External Memory Interface Features in Stratix V Devices

The Stratix V I/O elements (IOE) provide built-in functionality required for a rapid and robust implementation of external memory interfacing.

The following device features are available for external memory interfaces:

- DQS phase-shift circuitry
- PHY Clock (PHYCLK) networks
- DQS logic block
- Dynamic on-chip termination (OCT) control
- IOE registers
- Delay chains
- Delay-locked loops (DLLs)
- Read- and write-leveling support
- Trace mismatch compensation
- Read FIFO blocks
- Slew rate adjustment
- Programmable drive strength

UniPHY IP

The high-performance memory interface solution includes the self-calibrating UniPHY IP that is optimized to take advantage of the Stratix V I/O structure and the Quartus II software TimeQuest Timing Analyzer. The UniPHY IP helps set up the physical interface (PHY) best suited for your system. This provides the total solution for the highest reliable frequency of operation across process, voltage, and temperature (PVT) variations.

The UniPHY IP instantiates a PLL to generate related clocks for the memory interface. The UniPHY IP can also dynamically choose the number of delay chains that are required for the system. The amount of delay is equal to the sum of the intrinsic delay of the delay element and the product of the number of delay steps and the value of the delay steps.

The UniPHY IP and the Altera memory controller MegaCore® functions can run at half or quarter of the I/O interface frequency of the memory devices, allowing better timing management in high-speed memory interfaces. The Stratix V devices contain built-in circuitry in the IOE to convert data from full rate (the I/O frequency) to half rate (the controller frequency) and vice versa.

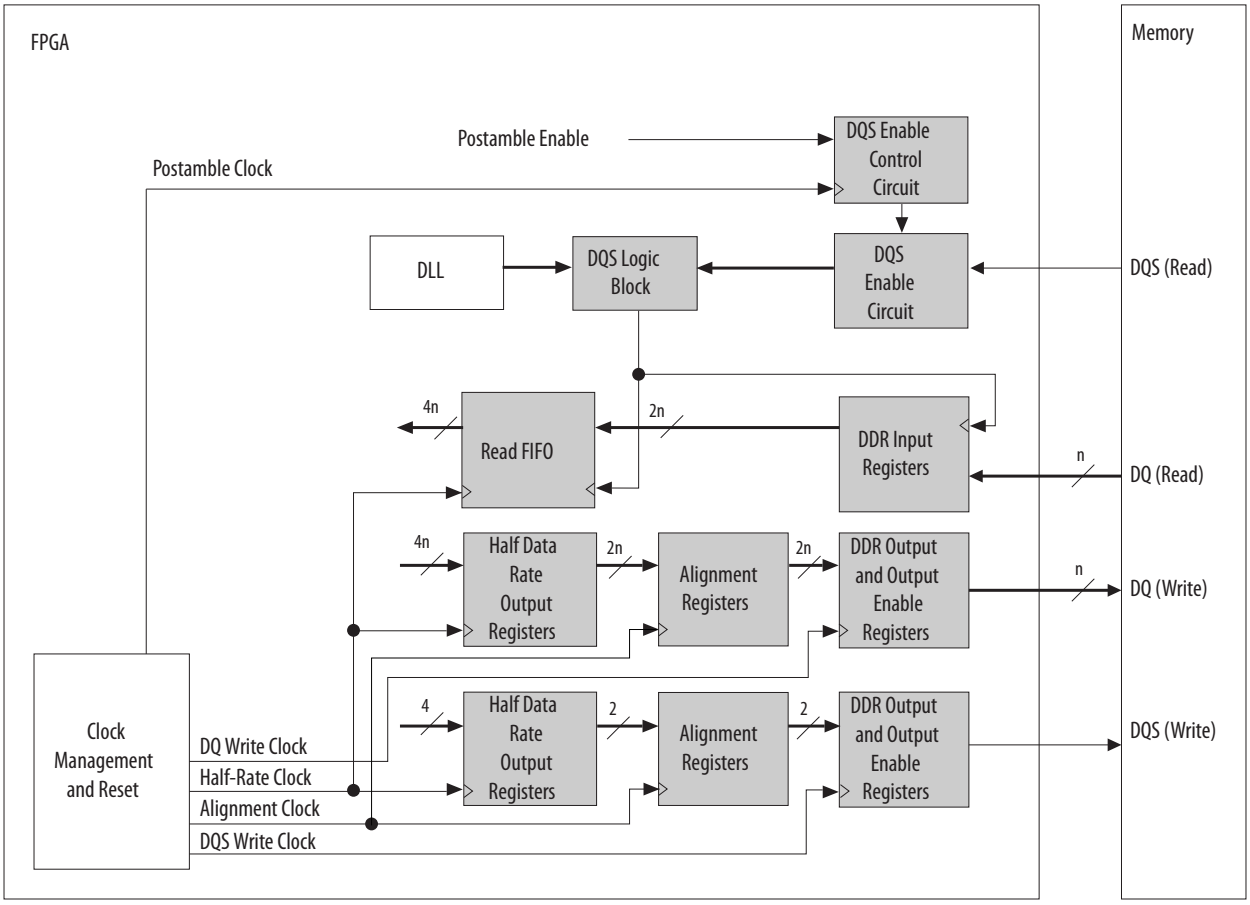
Related Information

Reference Material volume, External Memory Interface Handbook
Provides more information about the UniPHY IP.

External Memory Interface Datapath

The following figure shows an overview of the memory interface datapath that uses the Stratix V I/O elements. In the figure, the DQ/DQS read and write signals may be bidirectional or unidirectional, depending on the memory standard. If the signal is bidirectional, it is active during read and write operations. You can bypass each register block.

Figure 7-2: External Memory Interface Datapath Overview for Stratix V Devices



Note: There are slight block differences for different memory interface standards. The shaded blocks are part of the I/O elements.

DQS Phase-Shift Circuitry

The Stratix V phase-shift circuitry provides phase shift to the DQS/CQ and CQn pins on read transactions if the DQS/CQ and CQn pins are acting as input clocks or strobes to the FPGA. The DQS phase-shift circuitry consists of DLLs that are shared between multiple DQS pins and the phase-offset module to further fine-tune the DQS phase shift for different sides of the device.

The following figures show how the DQS phase-shift circuitry is connected to the DQS/CQ and CQn pins in the Stratix V variants.

Figure 7-3: DQS/CQ and CQn Pins and DQS Phase-Shift Circuitry in Stratix V E Devices

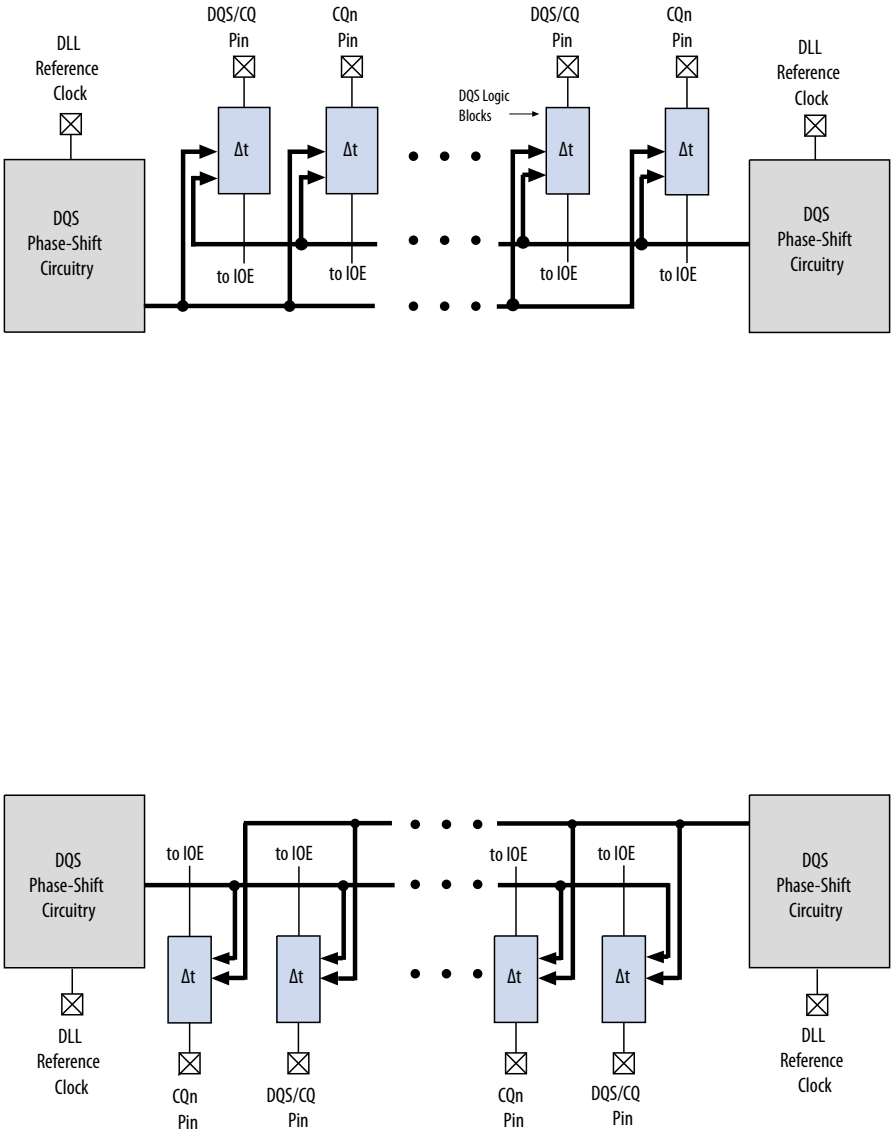
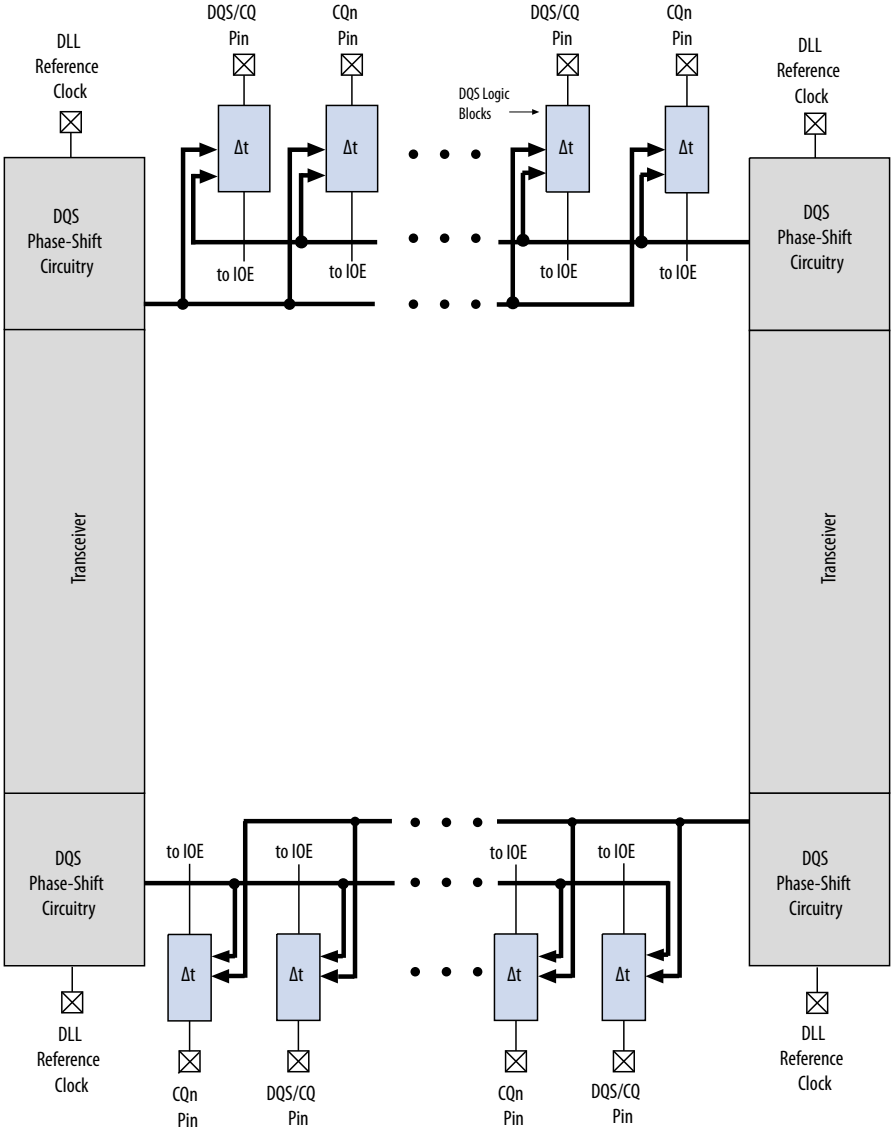


Figure 7-4: DQS/CQ and CQn Pins and DQS Phase-Shift Circuitry in Stratix V GX, GS, and GT Devices



The DQS phase-shift circuitry is connected to the DQS logic blocks that control each DQS/CQ or CQn pin. DQS logic blocks allow the DQS delay settings to be updated concurrently at every DQS/CQ or CQn pin.

Delay-Locked Loop

The DQS phase-shift circuitry uses a delay-locked loop (DLL) to dynamically control the clock delay required by the DQS/CQ and CQn pin.

The DLL uses a frequency reference to dynamically generate control signals for the delay chains in each of the DQS/CQ and CQn pins, allowing the delay to compensate for process, voltage, and temperature (PVT) variations. The DQS delay settings are gray-coded to reduce jitter if the DLL updates the settings.

There are a maximum of four DLLs, located in each corner of the Stratix V devices. You can clock each DLL using different frequencies. Each DLL can have two outputs with different phase offsets, which allows one Stratix V device to have eight different DLL phase shift settings.

You can have two different interfaces with the same frequency sharing a DLL, where the DLL controls the DQS delay settings for both interfaces.

Each I/O bank is accessible by two DLLs, giving more flexibility to create multiple frequencies and multiple-type interfaces. Each bank can use settings from one or both adjacent DLLs. For example, DQS1T can get its phase-shift settings from DLL_TR, while DQS2T can get its phase-shift settings from DLL_TL.

The reference clock for each DLL may come from the PLL output clocks or clock input pins.

Note: If you have a dedicated PLL that only generates the DLL input reference clock, set the PLL mode to **No Compensation** to achieve better performance (or the Quartus II software automatically changes it). Because the PLL does not use any other outputs, it does not have to compensate for any clock paths.

DLL Reference Clock Input for Stratix V Devices

Table 7-8: DLL Reference Clock Input for Stratix V E E9 and EB, and Stratix V GX A9, AB, B9, and BB Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X104_Y166 CEN_X104_Y157	COR_X0_Y170 COR_X0_Y161	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X104_Y166 CEN_X104_Y157	COR_X225_Y170 COR_X225_Y161	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X104_Y11 CEN_X104_Y2	COR_X225_Y10 COR_X225_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X104_Y11 CEN_X104_Y2	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-9: DLL Reference Clock Input for Stratix V GX A3 (with 36 Transceivers) and A4, and Stratix V GS D5 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X92_Y96 CEN_X92_Y87	COR_X0_Y100 COR_X0_Y91	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X92_Y96 CEN_X92_Y87	COR_X202_Y100 COR_X202_Y91	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X92_Y11 CEN_X92_Y2	COR_X202_Y10 COR_X202_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X92_Y11 CEN_X92_Y1	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-10: DLL Reference Clock Input for Stratix V GX B5 and B6 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X90_Y123 CEN_X90_Y114	LR_X0_Y109 LR_X0_Y100	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TR	CEN_X90_Y123 CEN_X90_Y114	LR_X197_Y109 LR_X197_Y100	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X90_Y11 CEN_X90_Y2	LR_X197_Y14 LR_X197_Y5	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X90_Y11 CEN_X90_Y2	LR_X0_Y14 LR_X0_Y5	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-11: DLL Reference Clock Input for Stratix V GX A5 and A7, and Stratix V GT C5 and C7 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X98_Y118 CEN_X98_Y109	COR_X0_Y122 COR_X0_Y113	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X98_Y118 CEN_X98_Y109	COR_X210_Y122 COR_X210_Y113	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X98_Y11 CEN_X98_Y2	COR_X210_Y10 COR_X210_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_BL	CEN_X98_Y11 CEN_X98_Y2	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-12: DLL Reference Clock Input for Stratix V GX A3 (with 24 Transceivers), and Stratix V GS D3 and D4 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X84_Y77 CEN_X84_Y68	COR_X0_Y81 COR_X0_Y72	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X84_Y77 CEN_X84_Y68	COR_X185_Y81 COR_X185_Y72	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X84_Y11 CEN_X84_Y2	COR_X185_Y10 COR_X185_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X84_Y11 CEN_X84_Y2	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-13: DLL Reference Clock Input for Stratix V GS D6 and D8 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X96_Y141 CEN_X96_Y132	COR_X0_Y145 COR_X0_Y136	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X96_Y141 CEN_X96_Y132	COR_X208_Y145 COR_X208_Y136	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X96_Y11 CEN_X96_Y2	COR_X208_Y10 COR_X208_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X96_Y11 CEN_X96_Y2	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

DQS Phase-Shift

The DLL can shift the incoming DQS signals by 0°, 45°, 90°, or 135°. The shifted DQS signal is then used as the clock for the DQ IOE input registers.

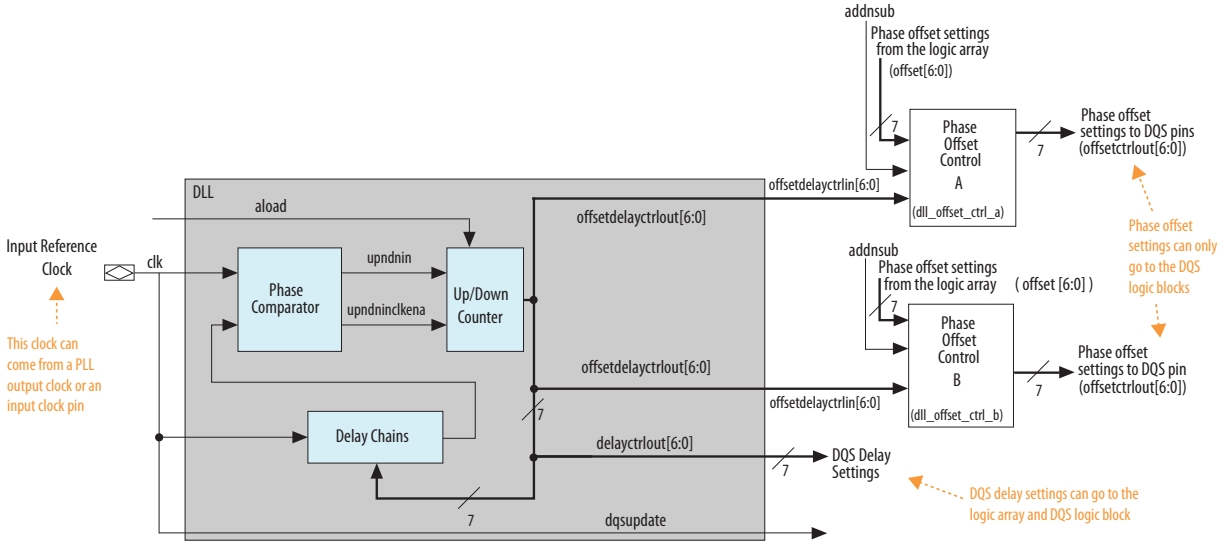
All DQS/CQ/CQn pins referenced to the same DLL, can have their input signal phase shifted by a different degree amount but all must be referenced at one particular frequency. For example, you can have a 90° phase shift on DQS1T and a 45° phase shift on DQS2T, referenced from a 300-MHz clock. However, not all phase-shift combinations are supported. The phase shifts on the DQS pins referenced by the same DLL must all be a multiple of 45° (up to 135°).

The 7-bit DQS delay settings from the DLL vary with PVT to implement the phase-shift delay. For example, with a 0° shift, the DQS/CQ signal bypasses both the DLL and DQS logic blocks. The Quartus II software automatically sets the DQ input delay chains, so that the skew between the DQ and DQS/CQ pins at the DQ IOE registers is negligible if a 0° shift is implemented. You can feed the DQS delay settings to the DQS logic block and logic array.

The shifted DQS/CQ signal goes to the DQS bus to clock the IOE input registers of the DQ pins. The signal can also go into the logic array for resynchronization if you are not using IOE resynchronization registers.

Figure 7-5: Simplified Diagram of the DQS Phase-Shift Circuitry

This figure shows a simple block diagram of the DLL. All features of the DQS phase-shift circuitry are accessible from the UniPHY megafunction in the Quartus II software.



The input reference clock goes into the DLL to a chain of up to eight delay elements. The phase comparator compares the signal coming out of the end of the delay chain block to the input reference clock. The phase comparator then issues the upndn signal to the Gray-code counter. This signal increments or decrements a 7-bit delay setting (DQS delay settings) that increases or decreases the delay through the delay element chain to bring the input reference clock and the signals coming out of the delay element chain in phase.

Note: In the Quartus II assignment, the phase offset control block 'A' is designated as `DLLOFFSETCTRL_CoordinateX_CoordinateY_N1` and phase offset control block 'B' is designated as `DLLOFFSETCTRL_CoordinateX_CoordinateY_N2`.

The DLL can be reset from either the logic array or a user I/O pin (if 2,560 or 512 clock cycles applies). Each time the DLL is reset, you must wait for 2,560 (low-jitter mode) or 512 clock cycles for the DLL to lock before you can capture the data properly.

You can still use DQS phase-shift circuitry for memory interfaces running on frequencies below the minimum DLL input frequency, which is 300 MHz. The frequency of the clock feeding the DLL should be doubled when the interface frequency is between 150 MHz and 299 MHz or multiplied by four when the interface frequency is between 75 MHz and 149 MHz. Because of the changes on the DLL input clock frequency, the DQS delay chain can only shift up to 67.5° for the interface frequency between 150 MHz and 299 MHz and 33.75° for the interface frequency between 75 MHz and 149 MHz. Depending on your design, while the DQS signal might not shift exactly to the middle of the DQ valid window, the IOE is still able to capture the data accurately in low-frequency applications, where a large amount of timing margin is available.

For the frequency range of each DLL frequency mode, refer to the device datasheet.

Related Information

Stratix V Device Datasheet

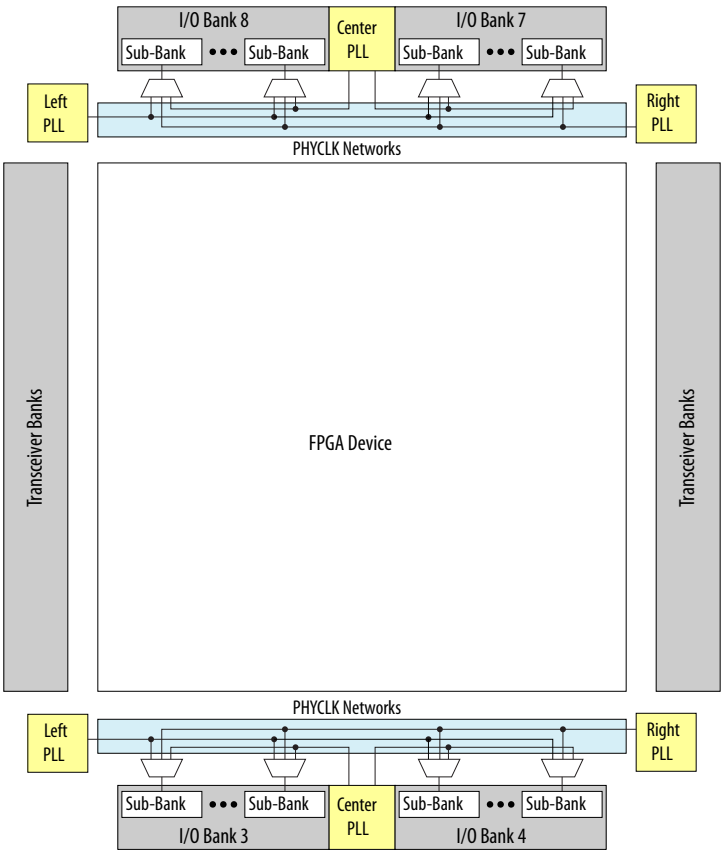
PHY Clock (PHYCLK) Networks

The PHYCLK network is a dedicated high-speed, low-skew balanced clock tree designed for a high-performance external memory interface.

The top and bottom sides of the Stratix V devices three PHYCLK networks each. Each PHYCLK network spans across one I/O bank and is driven by one of the left, right, or center PLLs located at that device side.

The following figure shows the PHYCLK networks available in the Stratix V devices.

Figure 7-6: PHYCLK Networks in Stratix V Devices



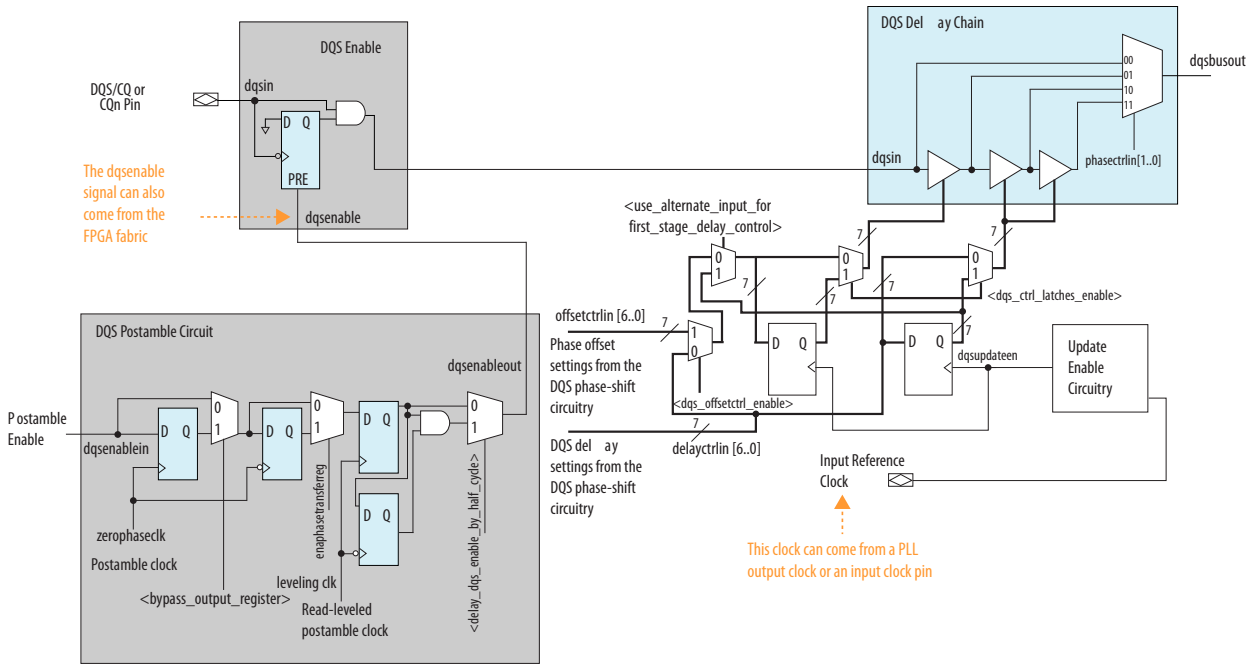
The PHYCLK network can be used to drive I/O sub-banks in each I/O bank. Each I/O sub-bank can be driven by only one PHYCLK network—all I/O pins in an I/O sub-bank are driven by the same PHYCLK network. The UniPHY IP for Stratix V devices uses the PHYCLK network to improve external memory interface performance.

DQS Logic Block

Each DQS/CQ and CQn pin is connected to a separate DQS logic block, which consists of the DQS delay chains, update enable circuitry, and DQS postamble circuitry.

The following figure shows the DQS logic block.

Figure 7-7: DQS Logic Block in Stratix V Devices



Update Enable Circuitry

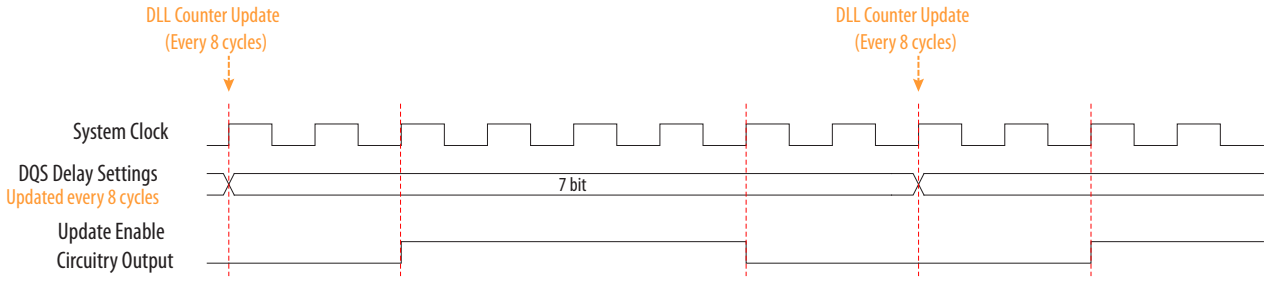
The update enable circuitry enables the registers to allow enough time for the DQS delay settings to travel from the DQS phase-shift circuitry or core logic to all the DQS logic blocks before the next change.

Both the DQS delay settings and the phase-offset settings pass through a register before going into the DQS delay chains. The registers are controlled by the update enable circuitry to allow enough time for any changes in the DQS delay setting bits to arrive at all the delay elements, which allows them to be adjusted at the same time.

The circuitry uses the input reference clock or a user clock from the core to generate the update enable output. The UniPHY intellectual property (IP) uses this circuit by default.

Figure 7-8: DQS Update Enable Waveform

This figure shows an example waveform of the update enable circuitry output.



DQS Delay Chain

DQS delay chains consist of a set of variable delay elements to allow the input DQS/CQ and CQn signals to be shifted by the amount specified by the DQS phase-shift circuitry or the logic array.

There are four delay elements in the DQS delay chain that have the same characteristics:

- Delay elements in the DQS logic block
- Delay elements in the DLL

The first delay chain closest to the DQS/CQ pin is shifted either by the DQS delay settings or by the sum of the DQS delay setting and the phase-offset setting. The DQS delay settings can come from the DQS phase-shift circuitry on either end of the I/O banks or from the logic array.

The number of delay chains required is transparent because the UniPHY IP automatically sets it when you choose the operating frequency.

In Stratix V devices, if you do not use the DLL to control the DQS delay chains, you can input your own gray-coded 7 bit settings using the `delayctrlin[6..0]` signals available in the UniPHY IP. These settings control 1, 2, 3, or all 4 delay elements in the DQS delay chains. The UniPHY megafunction can also dynamically choose the number of DQS delay chains required for the system. The amount of delay is equal to the sum of the intrinsic delay of the delay element and the product of the number of delay steps and the value of the delay steps. You can also bypass the DQS delay chain to achieve a 0° phase shift.

Related Information

- **ALTDQ_DQS2 Megafunction User Guide**
  Provides more information about programming the delay chains.
- **Delay Chains** on page 7-26

DQS Postamble Circuitry

There are preamble and postamble specifications for both read and write operations in DDR3 and DDR2 SDRAM. The DQS postamble circuitry ensures that data is not lost if there is noise on the DQS line during the end of a read operation that occurs while DQS is in a postamble state.

The Stratix V devices contain dedicated postamble registers that you can control to ground the shifted DQS signal that is used to clock the DQ input registers at the end of a read operation. This function ensures that any glitches on the DQS input signal during the end of a read operation and occurring while DQS is in a postamble state do not affect the DQ IOE registers.

- For preamble state, the DQS is low, just after a high-impedance state.
- For postamble state, the DQS is low, just before it returns to a high-impedance state.

For external memory interfaces that use a bidirectional read strobe (DDR3 and DDR2 SDRAM), the DQS signal is low before going to or coming from a high-impedance state.

Half Data Rate Block

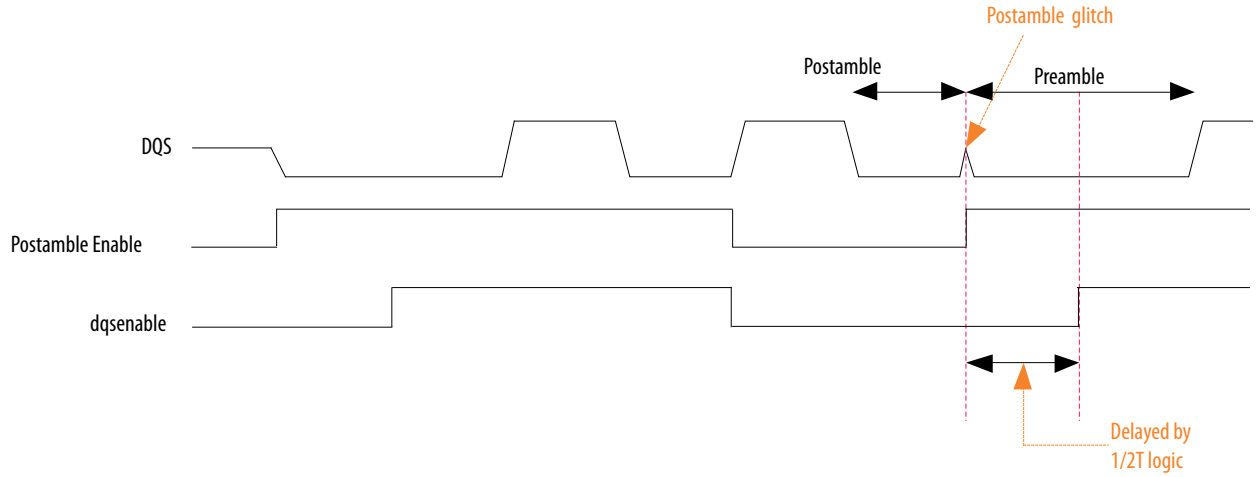
The Stratix V devices contain a half data rate (HDR) block in the postamble enable circuitry.

The HDR block is clocked by the half-rate resynchronization clock, which is the output of the I/O clock divider circuit. There is an AND gate after the postamble register outputs to avoid postamble glitches from a previous read burst on a non-consecutive read burst. This scheme allows half-a-clock cycle latency for `dqsenable` assertion and zero latency for `dqsenable` deassertion.

Using the HDR block as the first stage capture register in the postamble enable circuitry block is optional. Altera recommends using these registers if the controller is running at half the frequency of the I/Os.

Figure 7-9: Avoiding Glitch on a Non-Consecutive Read Burst Waveform

This figure shows how to avoid postamble glitches using the HDR block.

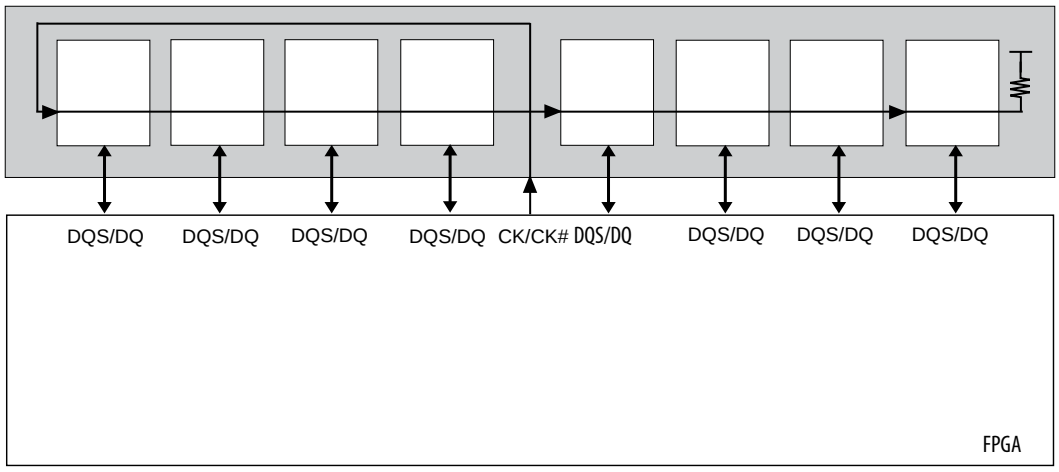


Leveling Circuitry

DDR3 SDRAM unbuffered modules use a fly-by clock distribution topology for better signal integrity. This means that the CK/CK# signals arrive at each DDR3 SDRAM device in the module at different times. The difference in arrival time between the first DDR3 SDRAM device and the last device on the module can be as long as 1.6 ns.

The following figure shows the clock topology in DDR3 SDRAM unbuffered modules.

Figure 7-10: DDR3 SDRAM Unbuffered Module Clock Topology



Because the data and read strobe signals are still point-to-point, take special care to ensure that the timing relationship between the CK/CK# and DQS signals (t_{DQSS}, t_{DSS}, and t_{DSH}) during a write is met at every

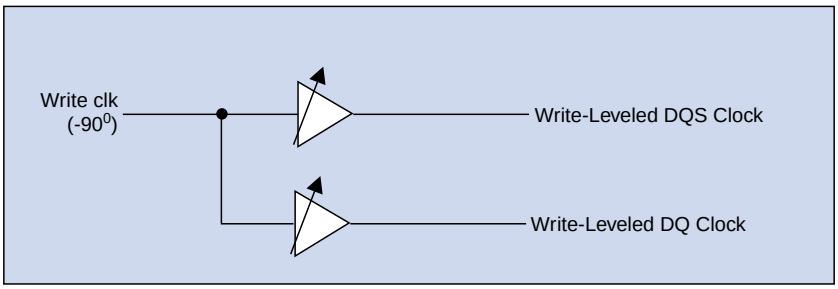
device on the modules. In a similar way, read data coming back into the FPGA from the memory is also staggered.

The Stratix V devices have leveling circuitry to address these two situations. There is one leveling circuit per I/O sub-bank (for example, I/O sub-bank 1A, 1B, and 1C each has one leveling circuitry). These delay chains are PVT-compensated by the same DQS delay settings as the DLL and DQS delay chains.

The DLL uses eight delay chain taps, such that each delay chain tap generates a 45° delay. The generated clock phases are distributed to every DQS logic block that is available in the I/O sub-bank. The delay chain taps then feed a multiplexer controlled by the UniPHY megafunction to select which clock phases are to be used for that x4 or x 8 DQS group. Each group can use a different tap output from the read-leveling and write-leveling delay chains to compensate for the different CK/CK# delay going into each device on the module.

Figure 7-11: Write-Leveling Delay Chains and Multiplexers

There is one leveling delay chain per I/O sub-bank (for example, I/O sub-banks 1A, 1B, and 1C). You can only have one memory interface in each I/O sub-bank when you use the leveling delay chain.



The –90° write clock of the UniPHY IP feeds the write-leveling circuitry to produce the clock to generate the DQS and DQ signals. During initialization, the UniPHY IP picks the correct write-leveled clock for the DQS and DQ clocks for each DQ/DQS group after sweeping all the available clocks in the write calibration process. The DQ clock output is –90° phase-shifted compared to the DQS clock output.

The UniPHY IP dynamically calibrates the alignment for read and write leveling during the initialization process.

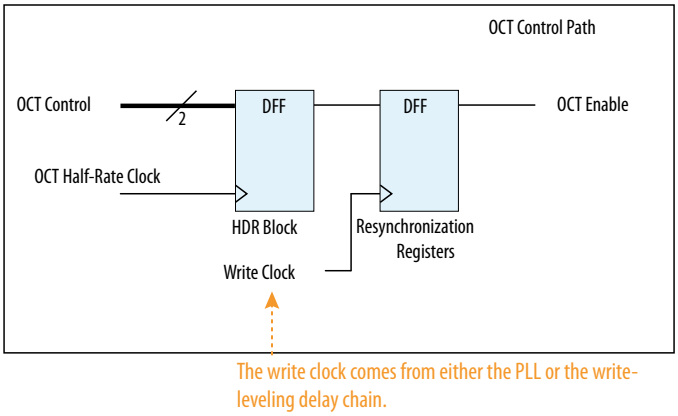
Related Information

- **Reference Material volume, External Memory Interface Handbook**
  Provides more information about the UniPHY IP.
- **DDR2 and DDR3 SDRAM Board Design Guidelines chapter, External Memory Interface Handbook**
  Provides layout guidelines for DDR3 SDRAM interface.

Dynamic OCT Control

The dynamic OCT control block includes all the registers that are required to dynamically turn the on-chip parallel termination (R_T OCT) on during a read and turn R_T OCT off during a write.

Figure 7-12: Dynamic OCT Control Block for Stratix V Devices



Related Information

Dynamic OCT in Stratix V Devices on page 5-29
Provides more information about dynamic OCT control.

IOE Registers

The IOE registers are expanded to allow source-synchronous systems to have faster register-to-FIFO transfers and resynchronization. All top, bottom, and right IOEs have the same capability.

Input Registers

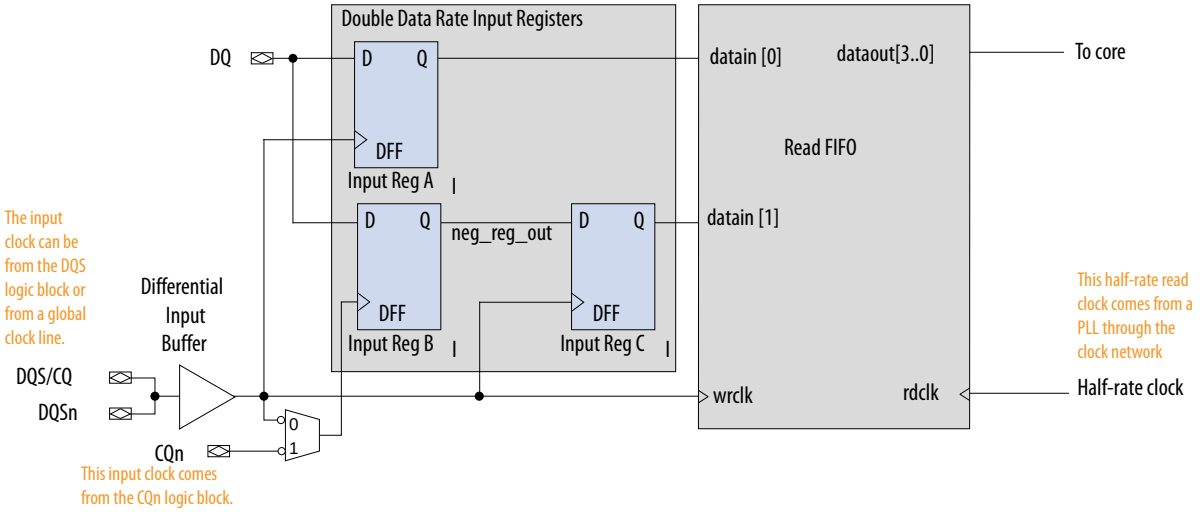
The input path consists of the DDR input registers and the read FIFO block. You can bypass each block of the input path.

There are three registers in the DDR input registers block. Two registers capture data on the positive and negative edges of the clock while the third register aligns the captured data. You can choose to use the same clock for the positive and negative edge registers or two complementary clocks (DQS/CQ for the positive-edge register and DQSn/CQn for the negative-edge register). The third register that aligns the captured data uses the same clock as the positive edge registers.

The read FIFO block resynchronizes the data to the system clock domain and lowers the data rate to half rate.

The following figure shows the registers available in the Stratix V input path. For DDR3 and DDR2 SDRAM interfaces, the DQS and DQSn signals must be inverted. If you use Altera's memory interface IPs, the DQS and DQSn signals are automatically inverted.

Figure 7-13: IOE Input Registers for Stratix V Devices



Output Registers

The Stratix V output and output-enable path is divided into the HDR block, alignment registers, and output and output-enable registers. The device can bypass each block of the output and output-enable path.

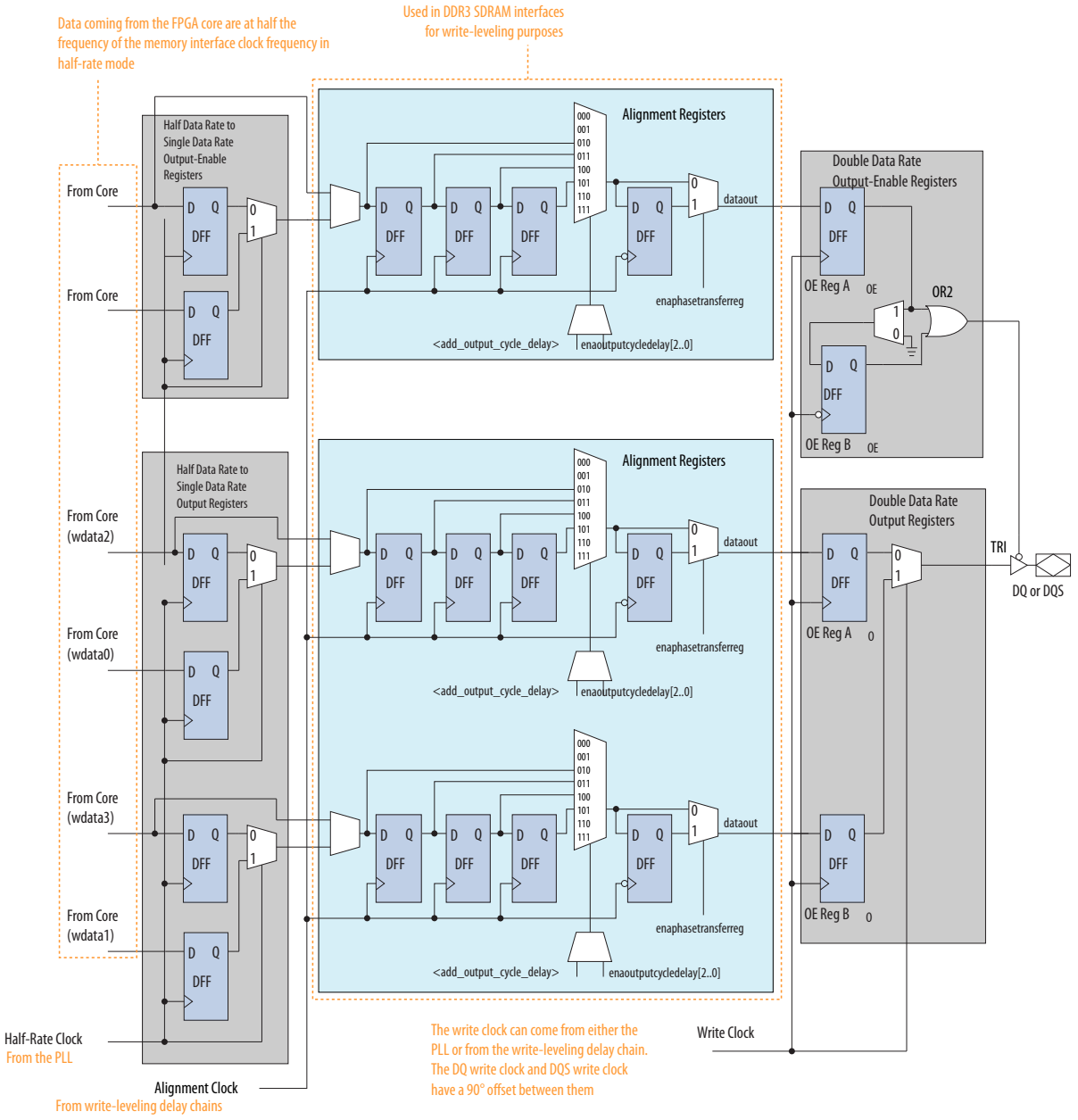
The output path is designed to route combinatorial or registered single data rate (SDR) outputs and full-rate or half-rate DDR outputs from the FPGA core. Half-rate data is converted to full-rate with the HDR block, clocked by the half-rate clock from the PLL.

The resynchronization registers are also clocked by the same 0° system clock, except in the DDR3 SDRAM interface. In DDR3 SDRAM interfaces, the leveling registers are clocked by the write-leveling clock.

The output-enable path has a structure similar to the output path—ensuring that the output-enable path goes through the same delay and latency as the output path.

Figure 7-14: IOE Output and Output-Enable Path Registers

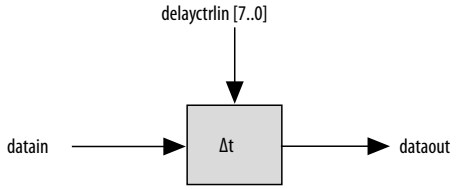
The following figure shows the registers available in the output and output-enable paths. You can bypass each register block of the output and output-enable paths.



Delay Chains

The Stratix V devices contain run-time adjustable delay chains in the I/O blocks and the DQS logic blocks. You can control the delay chain setting through the I/O or the DQS configuration block output.

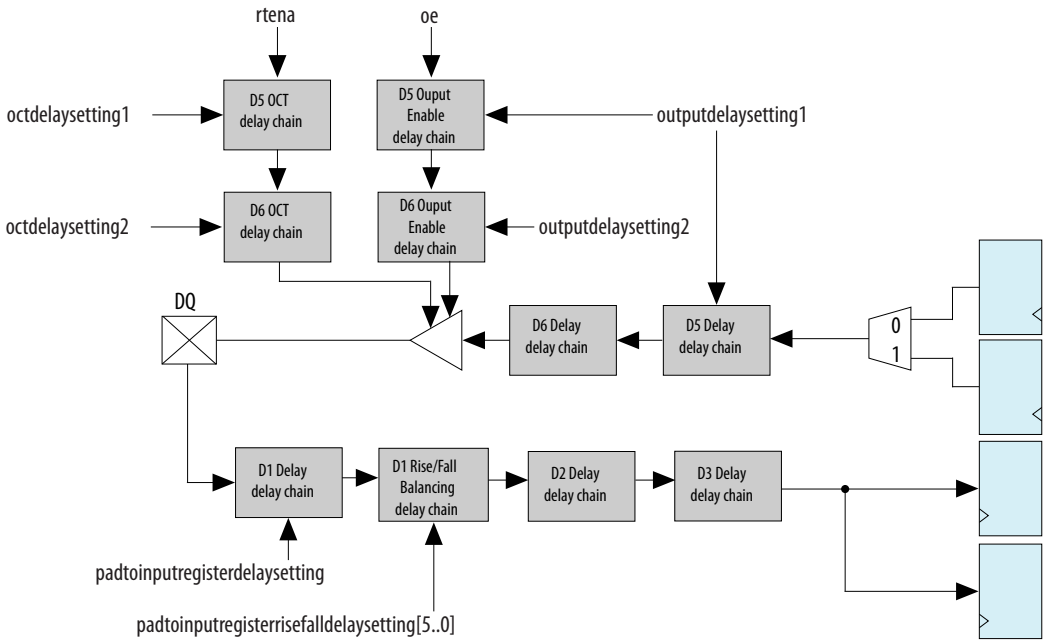
Figure 7-15: Delay Chain



Every I/O block contains two delay chains between the following elements:

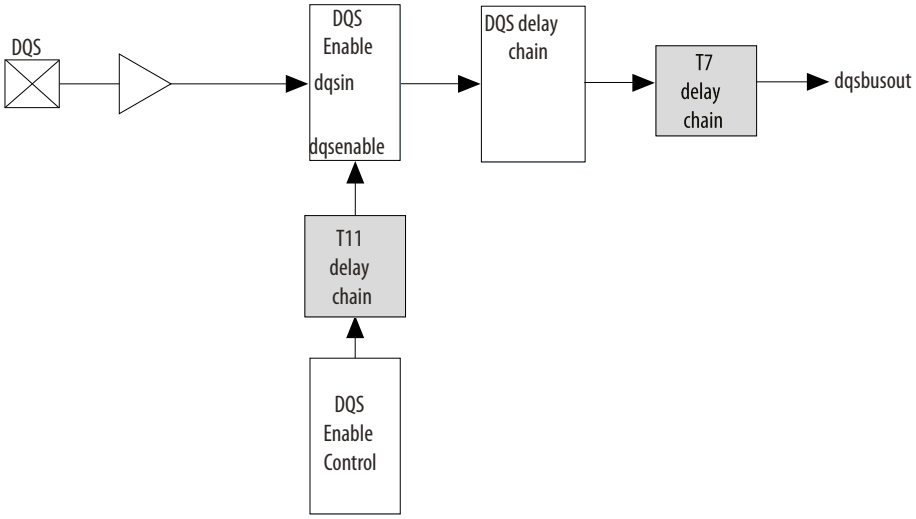
- The output registers and output buffer (in series)
- The input buffer and input register
- The output enable and output buffer
- The R_T OCT enable-control register and output buffer

Figure 7-16: Delay Chains in an I/O Block



Each DQS logic block contains a delay chain after the `dqsbusout` output and another delay chain before the `dqsenable` input.

Figure 7-17: Delay Chains in the DQS Input Path



Related Information

- **ALTDQ_DQS2 Megafunction User Guide**
  Provides more information about programming the delay chains.
- **DQS Delay Chain** on page 7-21

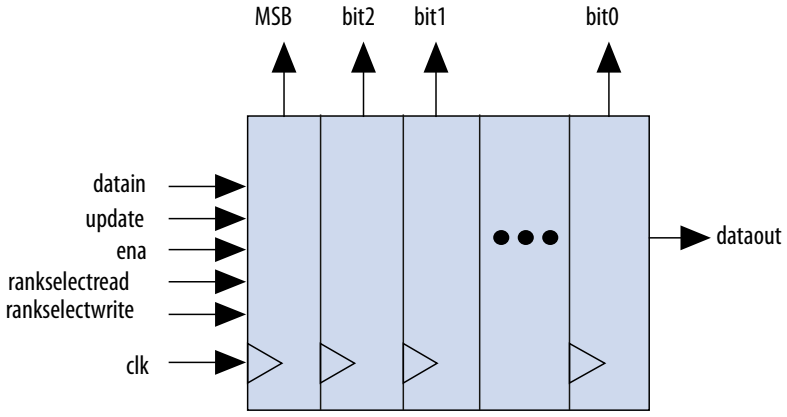
I/O and DQS Configuration Blocks

The I/O and DQS configuration blocks are shift registers that you can use to dynamically change the settings of various device configuration bits.

- The shift registers power-up low.
- Every I/O pin contains one I/O configuration register.
- Every DQS pin contains one DQS configuration block in addition to the I/O configuration register.

Figure 7-18: Configuration Block (I/O and DQS)

This figure shows the I/O configuration block and the DQS configuration block circuitry.



Related Information

ALTDQ_DQS2 Megafunction User Guide
Provides details about the I/O and DQS configuration block bit sequence.

Document Revision History

Date	Version	Changes
June 2014	2014.06.30	• Updated DDR3 1.35 V (DDR3L) performance from 933 MHz to 800 MHz.
January 2014	2014.01.10	• Updated the figure that shows the delay chains in the Stratix V I/O block. • Added related information link to **ALTDQ_DQS2 Megafunction User Guide** for more information about using the delay chains. • Added link to Altera's **External Memory Spec Estimator** tool to the topic listing the external memory interface performance.
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Added related information link to **DDR2 and DDR3 SDRAM Board Design Guidelines**. • Performed some minor text edits to improve accuracy.
December 2012	2012.11.28	• Reorganized content and updated template. • Added RLDRAM 3 support. • Added performance information for external memory interfaces. • Separated the DQ/DQS groups tables into separate topics for each device variant for easy reference. • Moved the PHYCLK networks pin placement guideline to the **Planning Pin and FPGA Resources** chapter of the *External Memory Interface Handbook*. • Removed guidelines on DDR2 and DDR3 SDRAM DIMM interfaces. Refer to the relevant sections in the **External Memory Interface Handbook** for the information. • Corrected “Gray-code” to “Binary-Code” in the “Phase Offset Control” section. • Removed the topic about phase offset control. • Removed the topics about I/O and DQS configuration block bit sequence. Refer to the relevant sections in the **ALTDQ_DQS2 Megafunction User Guide**.

Date	Version	Changes
June 2012	1.4	• Added Table 7–6, Table 7–8, and Table 7–9. • Updated Table 7–2, Table 7–3, and Table 7–7. • Updated Figure 7–18. • Updated the "PHY Clock (PHYCLK) Networks" section.
November 2011	1.3	• Added "PHY Clock (PHYCLK) Networks" section. • Updated "Delay-Locked Loop" section. • Updated Figure 7–3, Figure 7–5, and Figure 7–7. • Updated Table 7–2, Table 7–3, Table 7–4, Table 7–5, and Table 7–6. • Minor text edits.
May 2011	1.2	• Chapter moved to volume 2 for the 11.0 release. • Updated Figure 7–4, Figure 7–6, Figure 7–13, Figure 7–14, and Figure 7–17. • Updated Table 7–2, Table 7–7, and Table 7–8. • Minor text edits.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

Configuration, Design Security, and Remote System Upgrades in Stratix V Devices

8

2015.01.23

SV51010 Subscribe Send Feedback

This chapter describes the configuration schemes, design security, and remote system upgrade that are supported by the Stratix V devices.

Related Information

- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the *Stratix V Device Handbook* chapters.
- **Stratix V Device Overview**
  Provides more information about configuration features supported for each configuration scheme.
- **Stratix V Device Datasheet**
  Provides more information about the estimated uncompressed **.rbf** file sizes, FPP `DCLK-to-DATA[]` ratio, and timing parameters.
- **Configuration via Protocol (CvP) Implementation in Altera FPGAs User Guide**
  Provides more information about the CvP configuration scheme.
- **Design Planning for Partial Reconfiguration**
  Provides more information about partial reconfiguration.

Enhanced Configuration and Configuration via Protocol

Table 8-1: Configuration Modes and Features of Stratix V Devices

Stratix V devices support 1.8 V, 2.5 V, and 3.0 V programming voltages and several configuration modes.

Mode	Data Width	Max Clock Rate (MHz)	Max Data Rate (Mbps)	Decompressio	Design Security	Partial Reconfiguratio (10)	Remote System Update
AS through the EPCS and EPCQ serial configuration device	1 bit, 4 bits	100	—	Yes	Yes	—	Yes

(10) Partial reconfiguration is an advanced feature of the device family. If you are interested in using partial reconfiguration, contact Altera for support.

Mode	Data Width	Max Clock Rate (MHz)	Max Data Rate (Mbps)	Decompressio	Design Security	Partial Reconfiguratio (10)	Remote System Update
PS through CPLD or external microcontroller	1 bit	125	125	Yes	Yes	—	—
FPP	8 bits	125	—	Yes	Yes	—	Parallel flash loader
	16 bits	125	—	Yes	Yes	Yes	
	32 bits	100	—	Yes	Yes	—	
CvP (PCIe)	x1, x2, x4, and x8 lanes	—	—	Yes	Yes	Yes	—
JTAG	1 bit	33	33	—	—	—	—

Instead of using an external flash or ROM, you can configure the Stratix V devices through PCIe using CvP. The CvP mode offers the fastest configuration rate and flexibility with the easy-to-use PCIe hard IP block interface. The Stratix V CvP implementation conforms to the PCIe 100 ms power-up-to-active time requirement.

Related Information

Configuration via Protocol (CvP) Implementation in Altera FPGAs User Guide
Provides more information about the CvP configuration scheme.

MSEL Pin Settings

To select a configuration scheme, hardwire the `MSEL` pins to V_{CCPGM} or GND without pull-up or pull-down resistors.

Note: Altera recommends connecting the `MSEL` pins directly to V_{CCPGM} or GND. Driving the `MSEL` pins from a microprocessor or another controlling device may not guarantee the V_{IL} or V_{IH} of the `MSEL` pins. The V_{IL} or V_{IH} of the `MSEL` pins must be maintained throughout configuration stages.

(10) Partial reconfiguration is an advanced feature of the device family. If you are interested in using partial reconfiguration, contact Altera for support.
(11) Supported at a maximum clock rate of 62.5 MHz.

Table 8-2: MSEL Pin Settings for Each Configuration Scheme of Stratix V Devices

Configuration Scheme	Compression Feature	Design Security Feature	V_{CCPGM} (V)	Power-On Reset (POR) Delay	Valid MSEL[4..0]
FPP x8	Disabled	Disabled	1.8/2.5/3.0	Fast	10100
				Standard	11000
	Disabled	Enabled	1.8/2.5/3.0	Fast	10101
				Standard	11001
	Enabled	Enabled/ Disabled	1.8/2.5/3.0	Fast	10110
				Standard	11010
FPP x16	Disabled	Disabled	1.8/2.5/3.0	Fast	00000
				Standard	00100
	Disabled	Enabled	1.8/2.5/3.0	Fast	00001
				Standard	00101
	Enabled	Enabled/ Disabled	1.8/2.5/3.0	Fast	00010
				Standard	00110
FPP x32	Disabled	Disabled	1.8/2.5/3.0	Fast	01000
				Standard	01100
	Disabled	Enabled	1.8/2.5/3.0	Fast	01001
				Standard	01101
	Enabled	Enabled/ Disabled	1.8/2.5/3.0	Fast	01010
				Standard	01110
PS	Enabled/ Disabled	Enabled/ Disabled	1.8/2.5/3.0	Fast	10000
				Standard	10001
AS (x1 and x4)	Enabled/ Disabled	Enabled/ Disabled	3.0	Fast	10010
				Standard	10011
JTAG-based configuration	Disabled	Disabled	—	—	Use any valid `MSEL` pin settings above

Note: You must also select the configuration scheme in the **Configuration** page of the **Device and Pin Options** dialog box in the Quartus II software. Based on your selection, the option bit in the programming file is set accordingly.

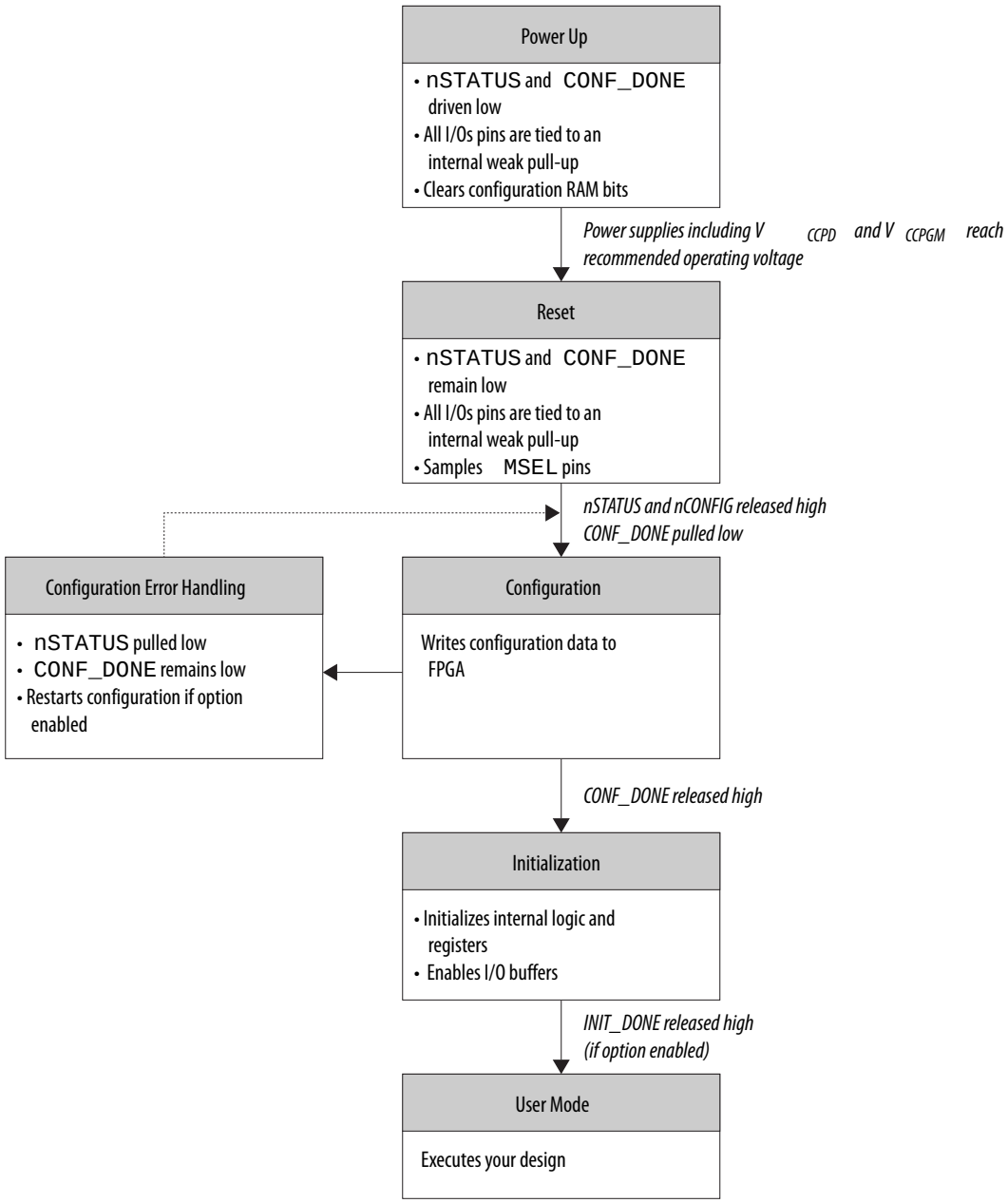
Related Information

- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides more information about JTAG pins voltage-level connection.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides more information about JTAG pins voltage-level connection.

Configuration Sequence

Describes the configuration sequence and each configuration stage.

Figure 8-1: Configuration Sequence for Stratix V Devices



You can initiate reconfiguration by pulling the nCONFIG pin low to at least the minimum t_{CFG} low-pulse width except for configuration using the partial reconfiguration operation. When this pin is pulled low, the nSTATUS and CONF_DONE pins are pulled low and all I/O pins are tied to an internal weak pull-up.

Power Up

Power up all the power supplies that are monitored by the POR circuitry. All power supplies, including V_{CCPGM} and V_{CCPD}, must ramp up from 0 V to the recommended operating voltage level within the ramp-up time specification. Otherwise, hold the `nCONFIG` pin low until all the power supplies reach the recommended voltage level.

V_{CCPGM} Pin

The configuration input buffers do not have to share power lines with the regular I/O buffers in Stratix V devices.

The operating voltage for the configuration input pin is independent of the I/O banks power supply, V_{CCIO}, during configuration. Therefore, Stratix V devices do not require configuration voltage constraints on V_{CCIO}.

V_{CCPD} Pin

Use the V_{CCPD} pin, a dedicated programming power supply, to power the I/O pre-drivers and JTAG I/O pins (`TCK`, `TMS`, `TDI`, `TRST`, and `TDO`).

If V_{CCIO} of the bank is set to 2.5 V or lower, V_{CCPD} must be powered up at 2.5 V. If V_{CCIO} is set greater than 2.5 V, V_{CCPD} must be greater than V_{CCIO}. For example, when V_{CCIO} is set to 3.0 V, V_{CCPD} must be set at 3.0 V.

Related Information

- **Stratix V Device Datasheet**
  Provides more information about the ramp-up time specifications.
- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides more information about configuration pin connections.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides more information about configuration pin connections.
- **Device Configuration Pins** on page 8-6
  Provides more information about configuration pins.
- **I/O Standards Voltage Levels in Stratix V Devices** on page 5-3
  Provides more information about typical power supplies for each supported I/O standards in Stratix V devices.

Reset

POR delay is the time frame between the time when all the power supplies monitored by the POR circuitry reach the recommended operating voltage and when `nSTATUS` is released high and the Stratix V device is ready to begin configuration.

Set the POR delay using the `MSEL` pins.

The user I/O pins are tied to an internal weak pull-up until the device is configured.

Related Information

- **MSEL Pin Settings** on page 8-2
- **Stratix V Device Datasheet**
  Provides more information about the POR delay specification.

Configuration

For more information about the DATA[] pins for each configuration scheme, refer to the appropriate configuration scheme.

Configuration Error Handling

To restart configuration automatically, turn on the **Auto-restart configuration after error** option in the **General** page of the **Device and Pin Options** dialog box in the Quartus II software.

If you do not turn on this option, you can monitor the nSTATUS pin to detect errors. To restart configuration, pull the nCONFIG pin low for at least the duration of t_{CFG}.

Related Information

Stratix V Device Datasheet
Provides more information about t_{STATUS} and t_{CFG} timing parameters.

Initialization

The initialization clock source is from the internal oscillator, CLKUSR pin, or DCLK pin. By default, the internal oscillator is the clock source for initialization. If you use the internal oscillator, the Stratix V device will be provided with enough clock cycles for proper initialization.

Note: If you use the optional CLKUSR pin as the initialization clock source and the nCONFIG pin is pulled low to restart configuration during device initialization, ensure that the CLKUSR or DCLK pin continues toggling until the nSTATUS pin goes low and then goes high again.

The CLKUSR pin provides you with the flexibility to synchronize initialization of multiple devices or to delay initialization. Supplying a clock on the CLKUSR pin during initialization does not affect configuration. After the CONF_DONE pin goes high, the CLKUSR or DCLK pin is enabled after the time specified by t_{CD2CU}. When this time period elapses, Stratix V devices require a minimum number of clock cycles as specified by T_{init} to initialize properly and enter user mode as specified by the t_{CD2UMC} parameter.

Related Information

Stratix V Device Datasheet
Provides more information about t_{CD2CU}, t_{init}, and t_{CD2UMC} timing parameters, and initialization clock source.

User Mode

You can enable the optional INIT_DONE pin to monitor the initialization stage. After the INIT_DONE pin is pulled high, initialization completes and your design starts executing. The user I/O pins will then function as specified by your design.

Device Configuration Pins

Configuration Pins Summary

The following table lists the Stratix V configuration pins and their power supply.

Note: The TDI, TMS, TCK, TDO, and TRST pins are powered by V_{CCPD} of the bank in which the pin resides.

Note: The CLKUSR, DEV_OE, DEV_CLRn, DATA[31..1], and DATA0 pins are powered by V_{CCPGM} during configuration and by V_{CCIO} of the bank in which the pin resides if you use it as a user I/O pin.

Table 8-3: Configuration Pin Summary for Stratix V Devices

Configuration Pin	Configuration Scheme	Input/Output	User Mode	Powered By
TDI	JTAG	Input	—	V_{CCPD}
TMS	JTAG	Input	—	V_{CCPD}
TCK	JTAG	Input	—	V_{CCPD}
TDO	JTAG	Output	—	V_{CCPD}
TRST	JTAG	Input	—	V_{CCPD}
CLKUSR	All schemes	Input	I/O	V_{CCPGM}/V_{CCIO} (12)
CRC_ERROR	Optional, all schemes	Output	I/O	Pull-up
CONF_DONE	All schemes	Bidirectional	—	V_{CCPGM}/Pull-up
DCLK	FPP and PS	Input	—	V_{CCPGM}
	AS	Output	—	V_{CCPGM}
DEV_OE	Optional, all schemes	Input	I/O	V_{CCPGM}/V_{CCIO} (12)
DEV_CLRn	Optional, all schemes	Input	I/O	V_{CCPGM}/V_{CCIO} (12)
INIT_DONE	Optional, all schemes	Output	I/O	Pull-up
MSEL[4..0]	All schemes	Input	—	V_{CCPGM}
nSTATUS	All schemes	Bidirectional	—	V_{CCPGM}/Pull-up
nCE	All schemes	Input	—	V_{CCPGM}
nCEO	All schemes	Output	I/O	Pull-up

(12) This pin is powered by V_{CCPGM} during configuration and powered by V_{CCIO} of the bank in which the pin resides when you use this pin as a user I/O pin.

Configuration Pin	Configuration Scheme	Input/Output	User Mode	Powered By
nCONFIG	All schemes	Input	—	V_{CCPGM}
DATA[31..1]	FPP	Input	I/O	V_{CCPGM}/V_{CCIO} (12)
DATA0	FPP and PS	Bidirectional	I/O	V_{CCPGM}/V_{CCIO} (12)
nCSO	AS	Output	—	V_{CCPGM}
nIO_PULLUP	All schemes	Input	—	V_{CCPGM}
AS_DATA[3..1]	AS	Bidirectional	—	V_{CCPGM}
AS_DATA0/ASDO	AS	Bidirectional	—	V_{CCPGM}
PR_REQUEST	Partial Reconfiguration	Input	I/O	V_{CCPGM}/V_{CCIO} (12)
PR_READY	Partial Reconfiguration	Output	I/O	V_{CCPGM}/V_{CCIO} (12)
PR_ERROR	Partial Reconfiguration	Output	I/O	V_{CCPGM}/V_{CCIO} (12)
PR_DONE	Partial Reconfiguration	Output	I/O	V_{CCPGM}/V_{CCIO} (12)

Related Information

- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides more information about each configuration pin.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides more information about each configuration pin.

Configuration Pin Options in the Quartus II Software

The following table lists the dual-purpose configuration pins available in the **Device and Pin Options** dialog box in the Quartus II software.

Table 8-4: Configuration Pin Options

Configuration Pin	Category Page	Option
CLKUSR	General	Enable user-supplied start-up clock (CLKUSR)
DEV_CLRn	General	Enable device-wide reset (DEV_CLRn)

Configuration Pin	Category Page	Option
DEV_OE	General	Enable device-wide output enable (DEV_OE)
INIT_DONE	General	Enable INIT_DONE output
nCEO	General	Enable nCEO pin
CRC_ERROR	Error Detection CRC	Enable Error Detection CRC_ERROR pin
		Enable open drain on CRC_ERROR pin
		Enable internal scrubbing
PR_REQUEST	General	Enable PR pin
PR_READY		
PR_ERROR		
PR_DONE		

Related Information

Reviewing Printed Circuit Board Schematics with the Quartus II Software
Provides more information about the device and pin options dialog box setting.

Fast Passive Parallel Configuration

The FPP configuration scheme uses an external host, such as a microprocessor, MAX® II device, or MAX V device. This scheme is the fastest method to configure Stratix V devices. The FPP configuration scheme supports 8-, 16-, and 32-bits data width.

You can use an external host to control the transfer of configuration data from an external storage such as flash memory to the FPGA. The design that controls the configuration process resides in the external host. You can store the configuration data in Raw Binary File (**.rbf**), Hexadecimal (Intel-Format) File (**.hex**), or Tabular Text File (**.ttf**) formats.

You can use the PFL megafunction with a MAX II or MAX V device to read configuration data from the flash memory device and configure the Stratix V device.

Note: Two DCLK falling edges are required after the CONF_DONE pin goes high to begin the initialization of the device for both uncompressed and compressed configuration data in an FPP configuration.

Related Information

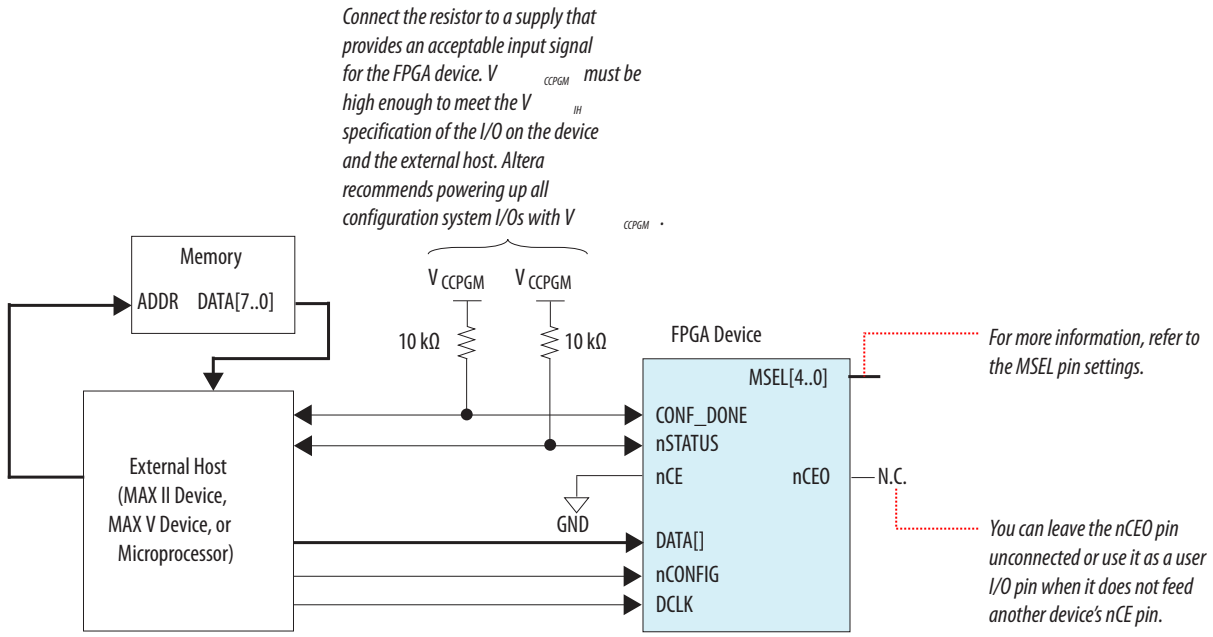
- **Parallel Flash Loader Megafunction User Guide**
- **Stratix V Device Datasheet**
  Provides more information about the FPP configuration timing.

Fast Passive Parallel Single-Device Configuration

To configure a Stratix V device, connect the device to an external host as shown in the following figure.

Note: If you are using the FPP x8 configuration mode, use `DATA[7..0]` pins. If you are using FPP x16 configuration mode, use `DATA[15..0]` pins.

Figure 8-2: Single Device FPP Configuration Using an External Host



Fast Passive Parallel Multi-Device Configuration

You can configure multiple Stratix V devices that are connected in a chain.

Pin Connections and Guidelines

Observe the following pin connections and guidelines for this configuration setup:

- Tie the following pins of all devices in the chain together:
  - `nCONFIG`
  - `nSTATUS`
  - `DCLK`
  - `DATA[]`
  - `CONF_DONE`

  By tying the `CONF_DONE` and `nSTATUS` pins together, the devices initialize and enter user mode at the same time. If any device in the chain detects an error, configuration stops for the entire chain and you must reconfigure all the devices. For example, if the first device in the chain flags an error on the `nSTATUS` pin, it resets the chain by pulling its `nSTATUS` pin low.
- Ensure that `DCLK` and `DATA[]` are buffered for every fourth device to prevent signal integrity and clock skew problems.
- All devices in the chain must use the same data width.
- If you are configuring the devices in the chain using the same configuration data, the devices must be of the same package and density.

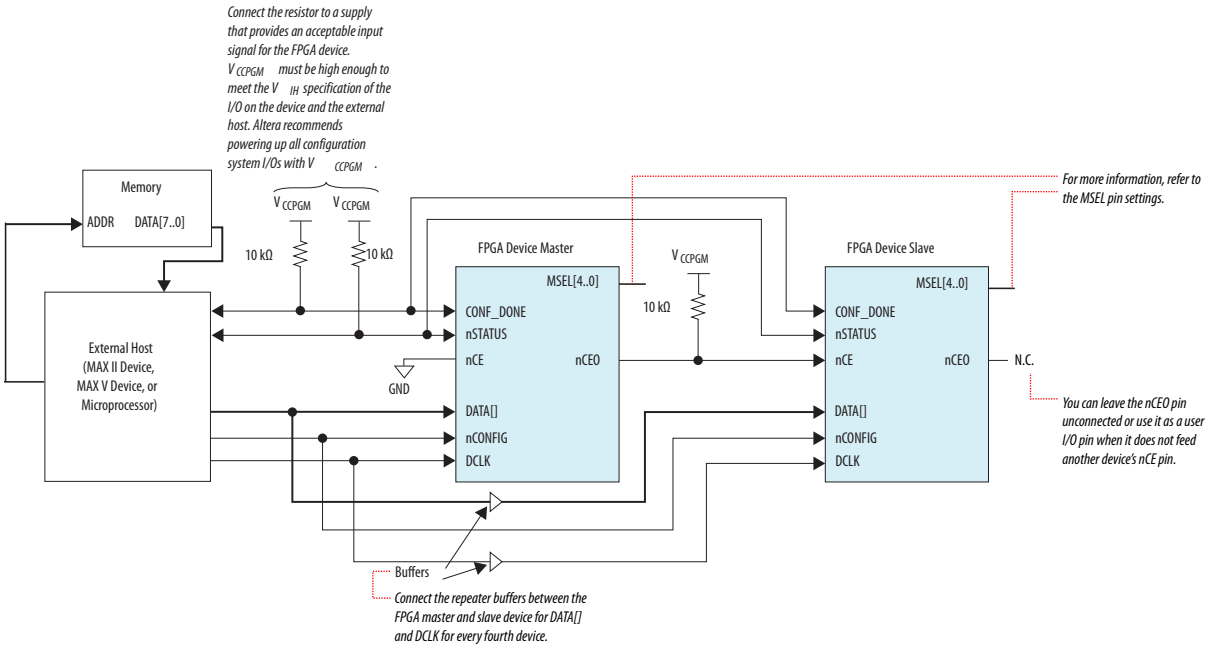
Using Multiple Configuration Data

To configure multiple Stratix V devices in a chain using multiple configuration data, connect the devices to an external host as shown in the following figure.

Note: If you are using the FPP x8 configuration mode, use `DATA[7..0]` pins. If you are using FPP x16 configuration mode, use `DATA[15..0]` pins.

Note: By default, the `nCEO` pin is disabled in the Quartus II software. For multi-device configuration chain, you must enable the `nCEO` pin in the Quartus II software. Otherwise, device configuration could fail.

Figure 8-3: Multiple Device FPP Configuration Using an External Host When Both Devices Receive a Different Set of Configuration Data



When a device completes configuration, its `nCEO` pin is released low to activate the `nCE` pin of the next device in the chain. Configuration automatically begins for the second device in one clock cycle.

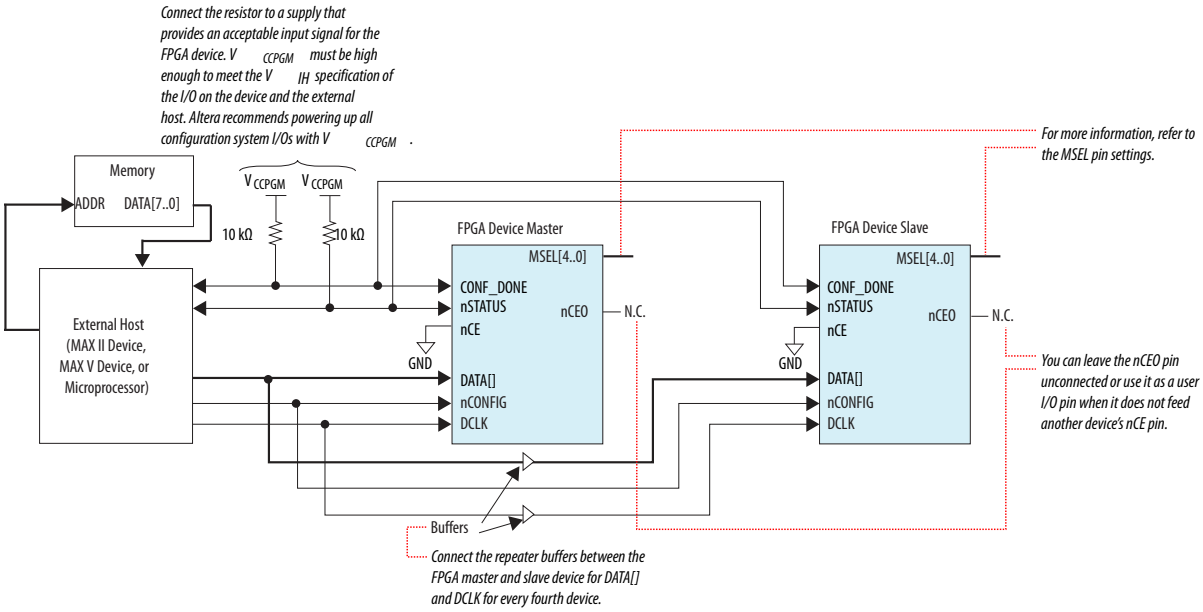
Using One Configuration Data

To configure multiple Stratix V devices in a chain using one configuration data, connect the devices to an external host as shown in the following figure.

Note: If you are using the FPP x8 configuration mode, use `DATA[7..0]` pins. If you are using FPP x16 configuration mode, use `DATA[15..0]` pins.

Note: By default, the `nCEO` pin is disabled in the Quartus II software. For multi-device configuration chain, you must enable the `nCEO` pin in the Quartus II software. Otherwise, device configuration could fail.

Figure 8-4: Multiple Device FPP Configuration Using an External Host When Both Devices Receive the Same Data



The nCE pins of the device in the chain are connected to GND, allowing configuration for these devices to begin and end at the same time.

Transmitting Configuration Data

This section describes how to transmit configuration data when you are using **.rbf** file for FPP x8, x16, and x32 configuration modes. The configuration data in the **.rbf** file is little endian.

For example, if the **.rbf** file contains the byte sequence 02 1B EE 01, refer to the following tables for details on how this data is transmitted in the FPP x8, x16, and x32 configuration modes.

Table 8-5: Transmitting Configuration Data for FPP x8 Configuration Mode

In FPP x8 configuration mode, the LSB of a byte is BIT0, and the MSB is BIT7.

BYTE0 = 02	BYTE1 = 1B	BYTE2 = EE	BYTE3 = 01
D[7..0]	D[7..0]	D[7..0]	D[7..0]
0000 0010	0001 1011	1110 1110	0000 0001

Table 8-6: Transmitting Configuration Data for FPP x16 Configuration Mode

In FPP x16 configuration mode, the first byte in the file is the LSB of the configuration word, and the second byte in the file is the MSB of the configuration word.

WORD0 = 1B02		WORD1 = 01EE	
LSB: BYTE0 = 02	MSB: BYTE1 = 1B	LSB: BYTE2 = EE	MSB: BYTE3 = 01
D[7..0]	D[15..8]	D[7..0]	D[15..8]

WORD0 = 1B02		WORD1 = 01EE	
LSB: BYTE0 = 02	MSB: BYTE1 = 1B	LSB: BYTE2 = EE	MSB: BYTE3 = 01
0000 0010	0001 1011	1110 1110	0000 0001

Table 8-7: Transmitting Configuration Data for FPP x32 Configuration Mode

In FPP x32 configuration mode, the first byte in the file is the LSB of the configuration double word, and the fourth byte is the MSB.

Double Word = 01EE1B02			
LSB: BYTE0 = 02	BYTE1 = 1B	BYTE2 = EE	MSB: BYTE3 = 01
D[7..0]	D[15..8]	D[23..16]	D[31..24]
0000 0010	0001 1011	1110 1110	0000 0001

Ensure that you do not swap the the upper bits or bytes with the lower bits or bytes when performing the FPP configuration. Sending incorrect configuration data during the configuration process may cause unexpected behavior on the CONF_DONE signal.

Active Serial Configuration

The AS configuration scheme supports AS x1 (1-bit data width) and AS x4 (4-bit data width) modes. The AS x4 mode provides four times faster configuration time than the AS x1 mode. In the AS configuration scheme, the Stratix V device controls the configuration interface.

Related Information

Stratix V Device Datasheet
Provides more information about the AS configuration timing.

DATA Clock (DCLK)

Stratix V devices generate the serial clock, DCLK, that provides timing to the serial interface. In the AS configuration scheme, Stratix V devices drive control signals on the falling edge of DCLK and latch the configuration data on the following falling edge of this clock pin.

The maximum DCLK frequency supported by the AS configuration scheme is 100 MHz except for the AS multi-device configuration scheme. You can source DCLK using CLKUSR or the internal oscillator. If you use the internal oscillator, you can choose a 12.5, 25, 50, or 100 MHz clock under the **Device and Pin Options** dialog box, in the **Configuration** page of the Quartus II software.

After power-up, DCLK is driven by a 12.5 MHz internal oscillator by default. The Stratix V device determines the clock source and frequency to use by reading the option bit in the programming file.

Related Information

Stratix V Device Datasheet
Provides more information about the DCLK frequency specification in the AS configuration scheme.

Active Serial Single-Device Configuration

To configure a Stratix V device, connect the device to a serial configuration (EPCS) device or quad-serial configuration (EPCQ) device, as shown in the following figures.

Figure 8-5: Single Device AS x1 Mode Configuration

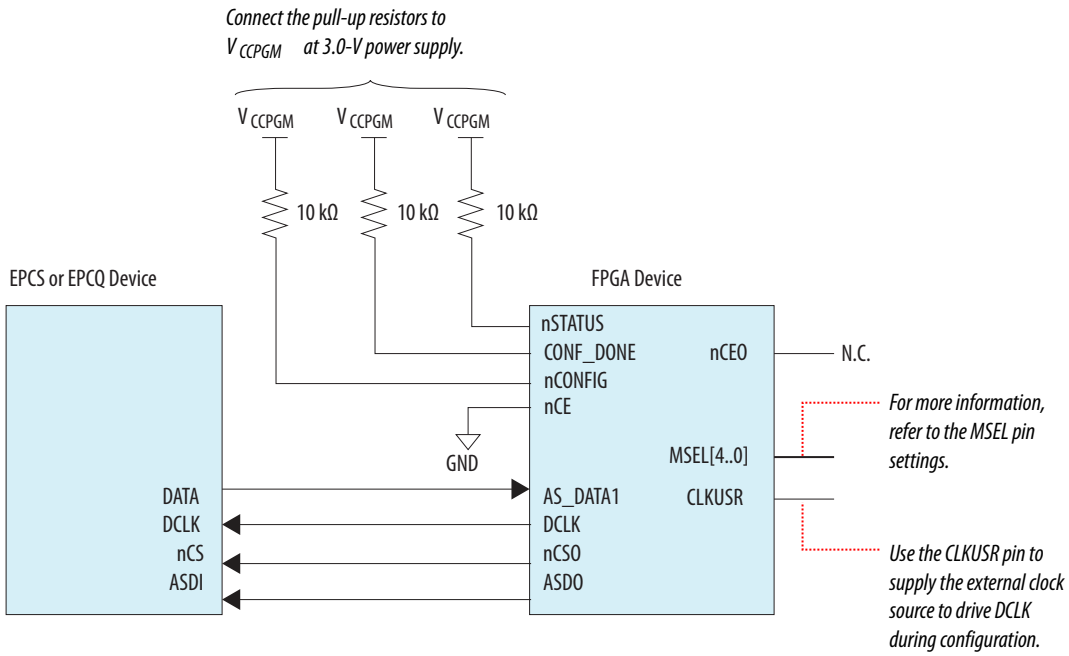
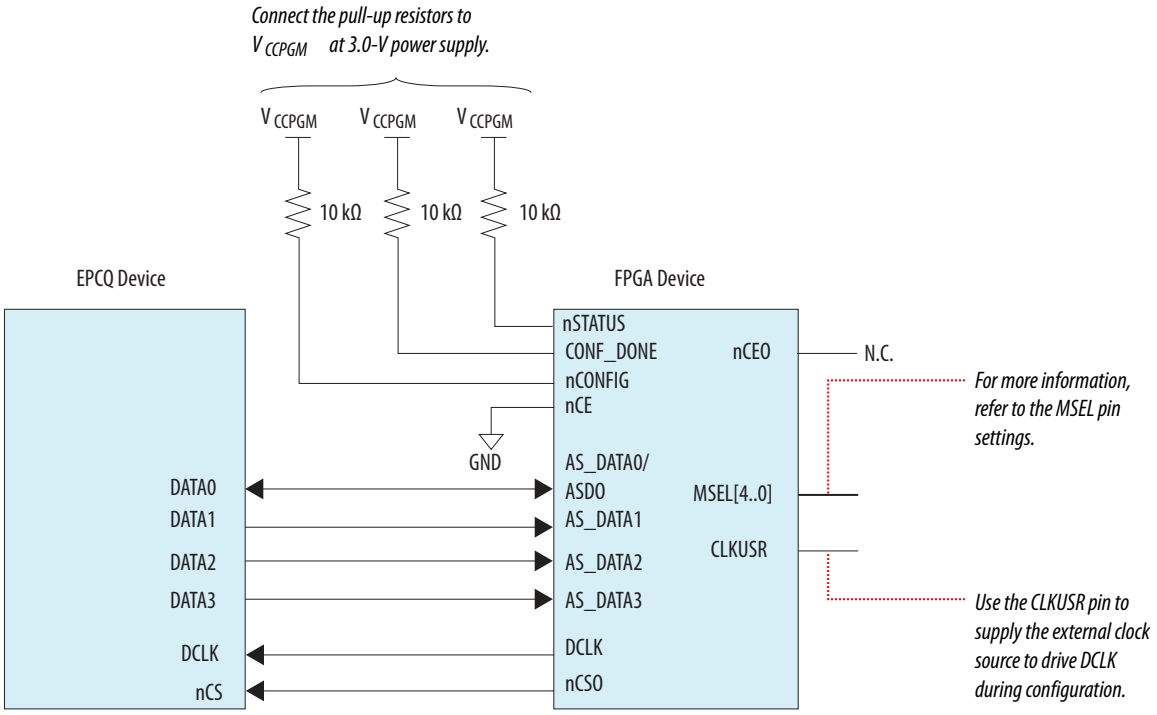


Figure 8-6: Single Device AS x4 Mode Configuration



Active Serial Multi-Device Configuration

You can configure multiple Stratix V devices that are connected to a chain. Only AS x1 mode supports multi-device configuration.

The first device in the chain is the configuration master. Subsequent devices in the chain are configuration slaves.

Note: The AS multi-device configuration scheme does not support 100 MHz DCLK frequency.

Pin Connections and Guidelines

Observe the following pin connections and guidelines for this configuration setup:

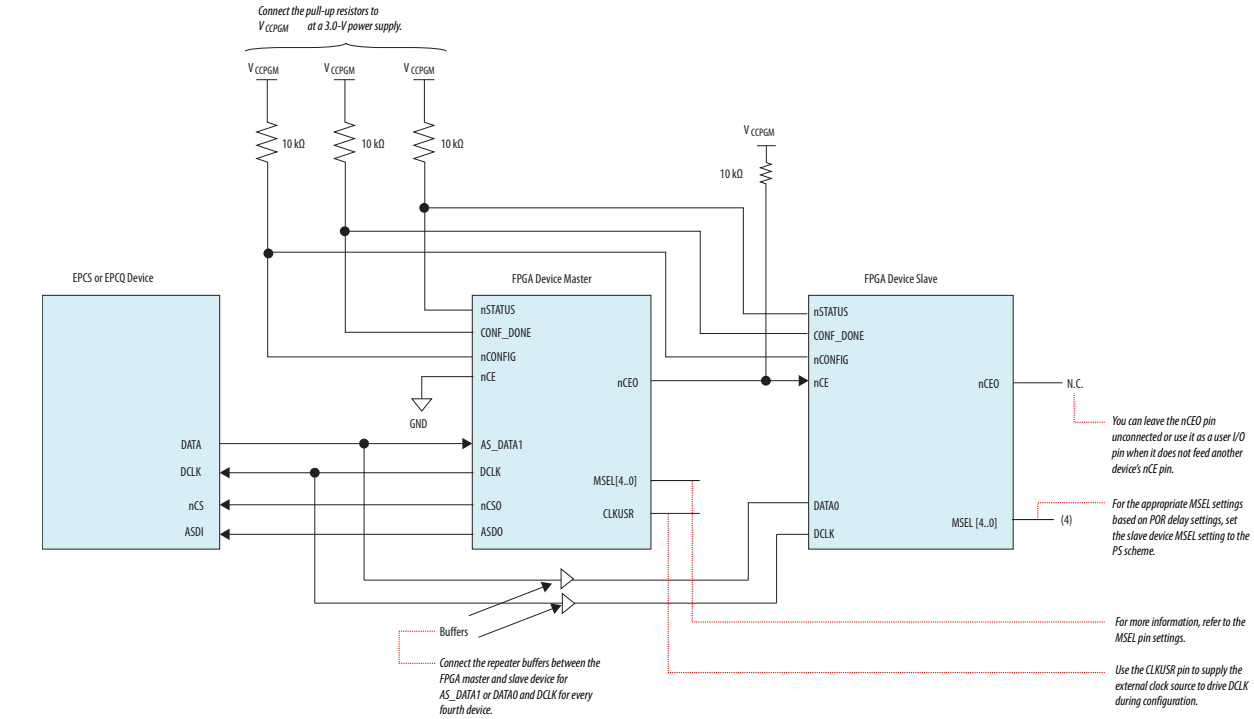
- Hardwire the MSEL pins of the first device in the chain to select the AS configuration scheme. For subsequent devices in the chain, hardwire their MSEL pins to select the PS configuration scheme. Any other Altera® devices that support the PS configuration can also be part of the chain as a configuration slave.
- Tie the following pins of all devices in the chain together:
  - nCONFIG
  - nSTATUS
  - DCLK
  - DATA[]
  - CONF_DONE

  By tying the CONF_DONE, nSTATUS, and nCONFIG pins together, the devices initialize and enter user mode at the same time. If any device in the chain detects an error, configuration stops for the entire chain and you must reconfigure all the devices. For example, if the first device in the chain flags an error on the nSTATUS pin, it resets the chain by pulling its nSTATUS pin low.
- Ensure that DCLK and DATA[] are buffered every fourth device to prevent signal integrity and clock skew problems.

Using Multiple Configuration Data

To configure multiple Stratix V devices in a chain using multiple configuration data, connect the devices to an EPCS or EPCQ device, as shown in the following figure.

Figure 8-7: Multiple Device AS Configuration When Both Devices in the Chain Receive Different Sets of Configuration Data



When a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device in the chain. Configuration automatically begins for the second device in one clock cycle.

Estimating the Active Serial Configuration Time

The AS configuration time is mostly the time it takes to transfer the configuration data from an EPCS or EPCQ device to the Stratix V device.

Use the following equations to estimate the configuration time:

- AS x1 mode

  .rbf Size x (minimum DCLK period / 1 bit per DCLK cycle) = estimated minimum configuration time.
- AS x4 mode

  .rbf Size x (minimum DCLK period / 4 bits per DCLK cycle) = estimated minimum configuration time.

Compressing the configuration data reduces the configuration time. The amount of reduction varies depending on your design.

Using EPCS and EPCQ Devices

EPCS devices support AS x1 mode and EPCQ devices support AS x1 and AS x4 modes.

Related Information

- **Serial Configuration (EPCS) Devices Datasheet**
- **Quad-Serial Configuration (EPCQ) Devices Datasheet**

Controlling EPCS and EPCQ Devices

During configuration, Stratix V devices enable the EPCS or EPCQ device by driving its nCSO output pin low, which connects to the chip select (nCS) pin of the EPCS or EPCQ device. Stratix V devices use the DCLK and ASDO pins to send operation commands and read address signals to the EPCS or EPCQ device. The EPCS or EPCQ device provides data on its serial data output (DATA[]) pin, which connects to the AS_DATA[] input of the Stratix V devices.

Note: If you wish to gain control of the EPCS pins, hold the nCONFIG pin low and pull the nCE pin high. This causes the device to reset and tri-state the AS configuration pins.

Trace Length and Loading

The maximum trace length and loading apply to both single- and multi-device AS configuration setups as listed in the following table. The trace length is the length from the Stratix V device to the EPCS or EPCQ device.

Table 8-8: Maximum Trace Length and Loading for AS x1 and x4 Configurations for Stratix V Devices

Stratix V Device AS Pins	Maximum Board Trace Length (Inches)		Maximum Board Load (pF)
	12.5/ 25/ 50 MHz	100 MHz	
DCLK	10	6	5
DATA[3..0]	10	6	10

Stratix V Device AS Pins	Maximum Board Trace Length (Inches)		Maximum Board Load (pF)
	12.5/ 25/ 50 MHz	100 MHz	
nCSO	10	6	10

Programming EPCS and EPCQ Devices

You can program EPCS and EPCQ devices in-system using a USB-Blaster™, EthernetBlaster, EthernetBlaster II, or ByteBlaster™ II download cable. Alternatively, you can program the EPCS or EPCQ using a microprocessor with the SRunner software driver.

In-system programming (ISP) offers you the option to program the EPCS or EPCQ either using an AS programming interface or a JTAG interface. Using the AS programming interface, the configuration data is programmed into the EPCS by the Quartus II software or any supported third-party software. Using the JTAG interface, an Altera IP called the serial flash loader (SFL) must be downloaded into the Stratix V device to form a bridge between the JTAG interface and the EPCS or EPCQ. This allows the EPCS or EPCQ to be programmed directly using the JTAG interface.

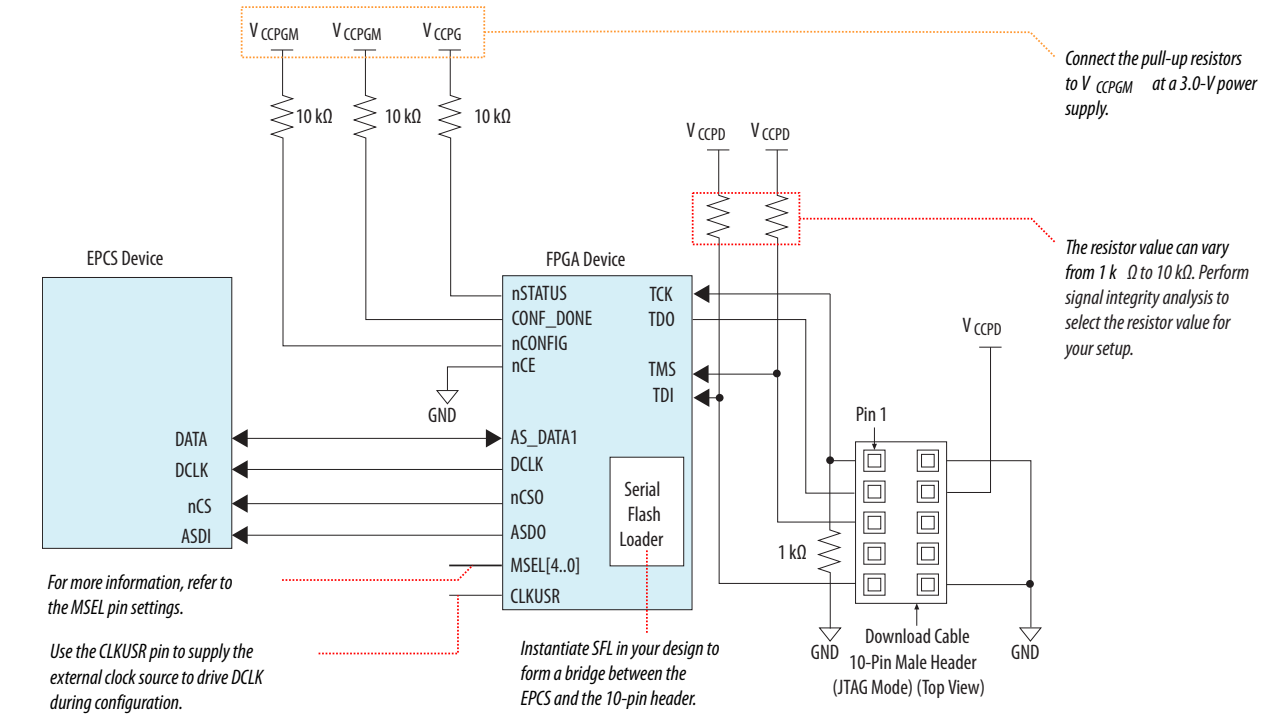
Related Information

- **AN 370: Using the Serial FlashLoader with the Quartus II Software**
- **AN 418: SRunner: An Embedded Solution for Serial Configuration Device Programming**

Programming EPCS Using the JTAG Interface

To program an EPCS device using the JTAG interface, connect the device as shown in the following figure.

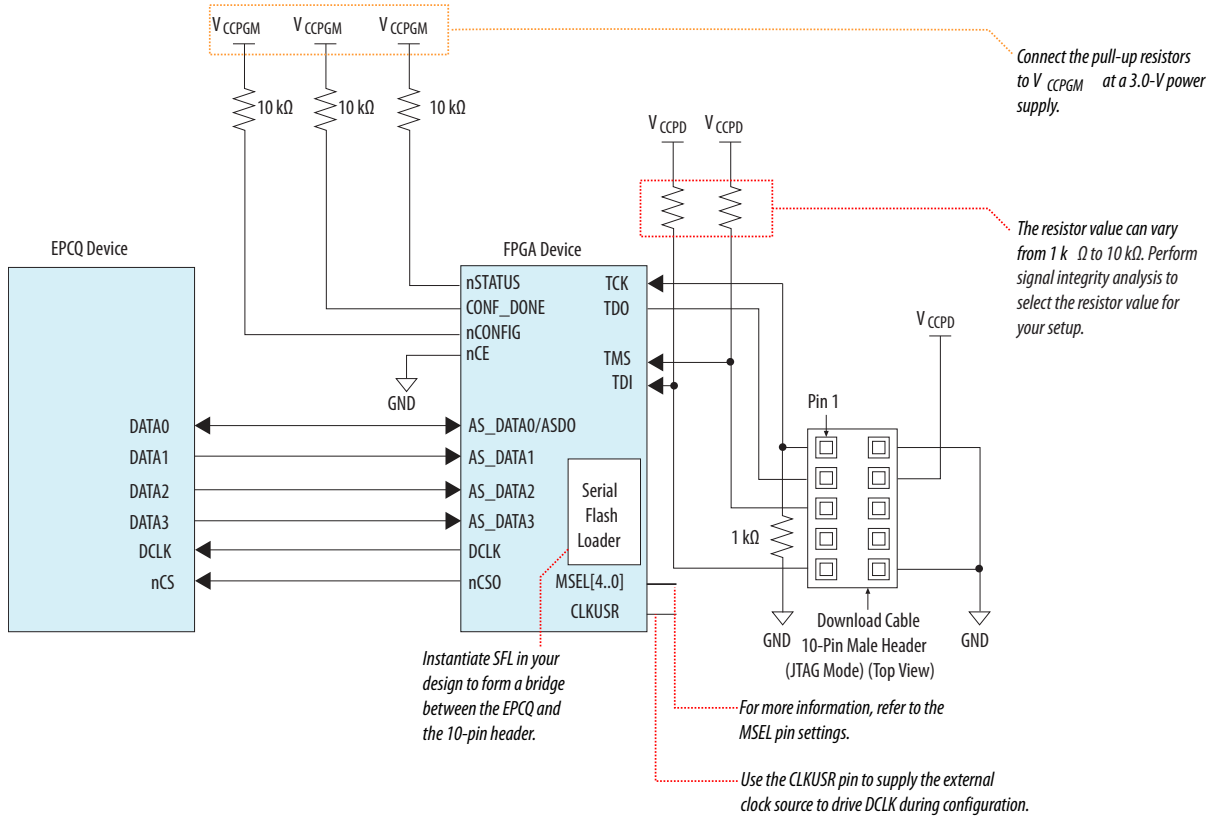
Figure 8-8: Connection Setup for Programming the EPCS Using the JTAG Interface



Programming EPCQ Using the JTAG Interface

To program an EPCQ device using the JTAG interface, connect the device as shown in the following figure.

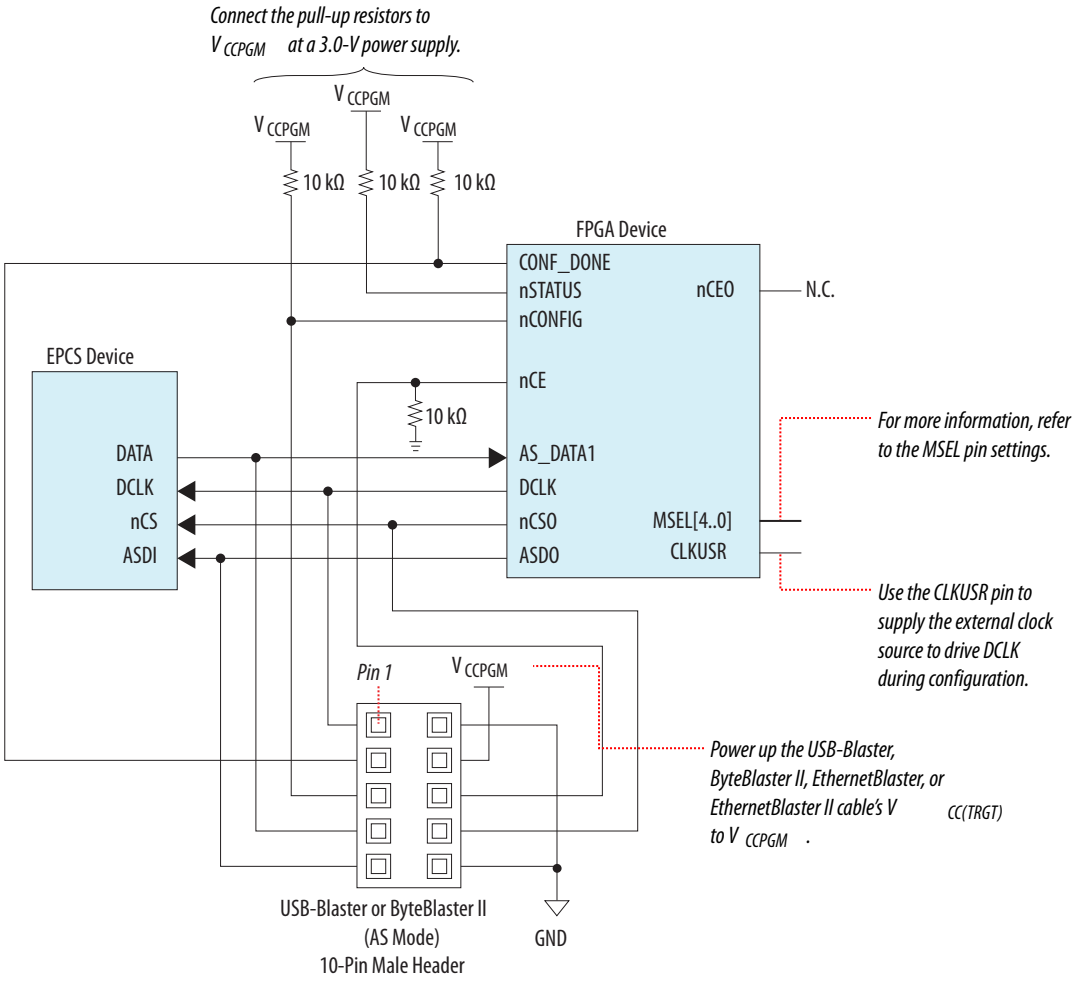
Figure 8-9: Connection Setup for Programming the EPCQ Using the JTAG Interface



Programming EPCS Using the Active Serial Interface

To program an EPCS device using the AS interface, connect the device as shown in the following figure.

Figure 8-10: Connection Setup for Programming the EPCS Using the AS Interface

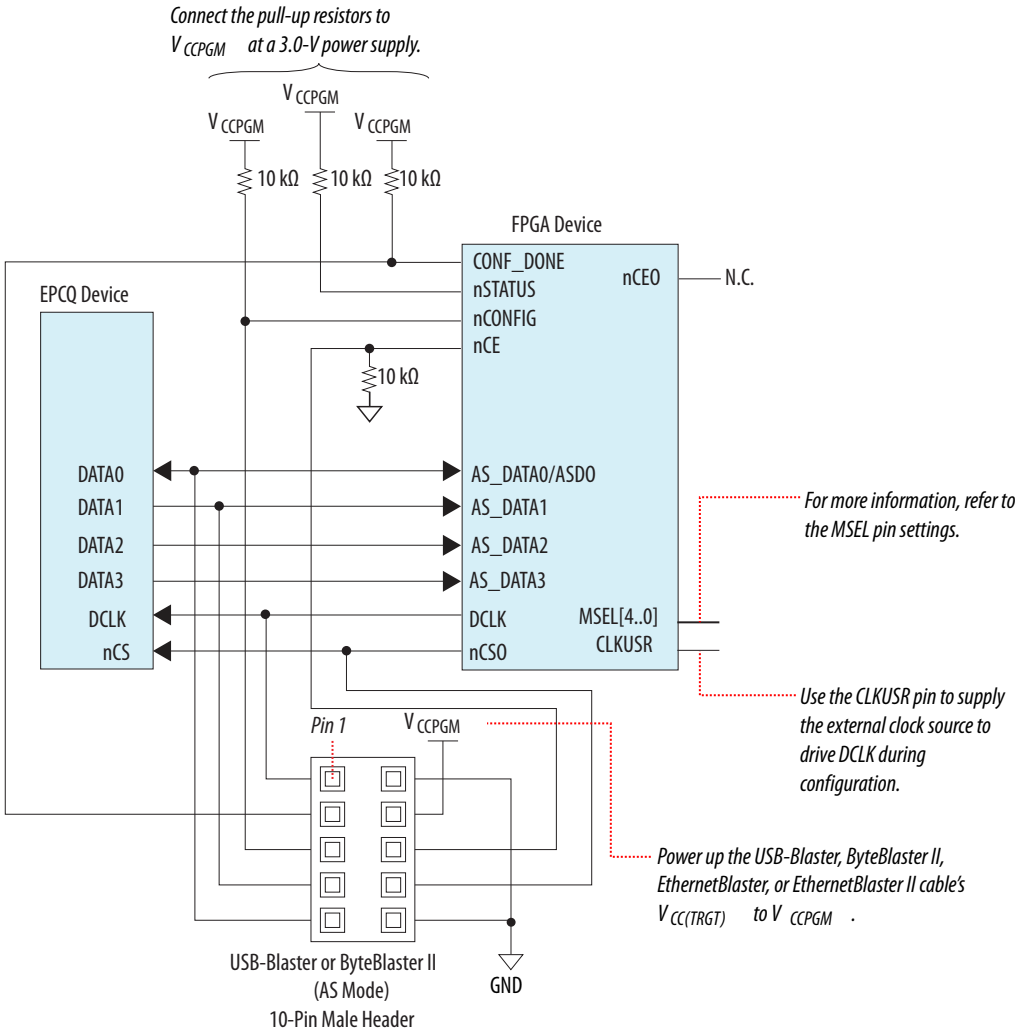


Programming EPCQ Using the Active Serial Interface

To program an EPCQ device using the AS interface, connect the device as shown in the following figure.

Figure 8-11: Connection Setup for Programming the EPCQ Using the AS Interface

Using the AS header, the programmer serially transmits the operation commands and configuration bits to the EPCQ on DATA0. This is equivalent to the programming operation for the EPCS.



When programming the EPCS and EPCQ devices, the download cable disables access to the AS interface by driving the nCE pin high. The nCONFIG line is also pulled low to hold the Stratix V device in the reset stage. After programming completes, the download cable releases nCE and nCONFIG, allowing the pull-down and pull-up resistors to drive the pin to GND and V_{CCPGM}, respectively.

During the EPCQ programming using the download cable, DATA0 transfers the programming data, operation command, and address information from the download cable into the EPCQ. During the EPCQ verification using the download cable, DATA1 transfers the programming data back to the download cable.

Passive Serial Configuration

The PS configuration scheme uses an external host. You can use a microprocessor, MAX II device, MAX V device, or a host PC as the external host.

You can use an external host to control the transfer of configuration data from an external storage such as flash memory to the FPGA. The design that controls the configuration process resides in the external host.

You can store the configuration data in Programmer Object File (**.pof**), **.rbf**, **.hex**, or **.ttf**. If you are using configuration data in **.rbf**, **.hex**, or **.ttf**, send the LSB of each data byte first. For example, if the **.rbf** contains the byte sequence 02 1B EE 01 FA, the serial data transmitted to the device must be 0100-0000 1101-1000 0111-0111 1000-0000 0101-1111.

You can use the PFL megafunction with a MAX II or MAX V device to read configuration data from the flash memory device and configure the Stratix V device.

For a PC host, connect the PC to the device using a download cable such as the Altera USB-Blaster USB port, ByteBlaster II parallel port, EthernetBlaster, and EthernetBlaster II download cables.

The configuration data is shifted serially into the `DATA0` pin of the device.

If you are using the Quartus II programmer and the `CLKUSR` pin is enabled, you do not need to provide a clock source for the pin to initialize your device.

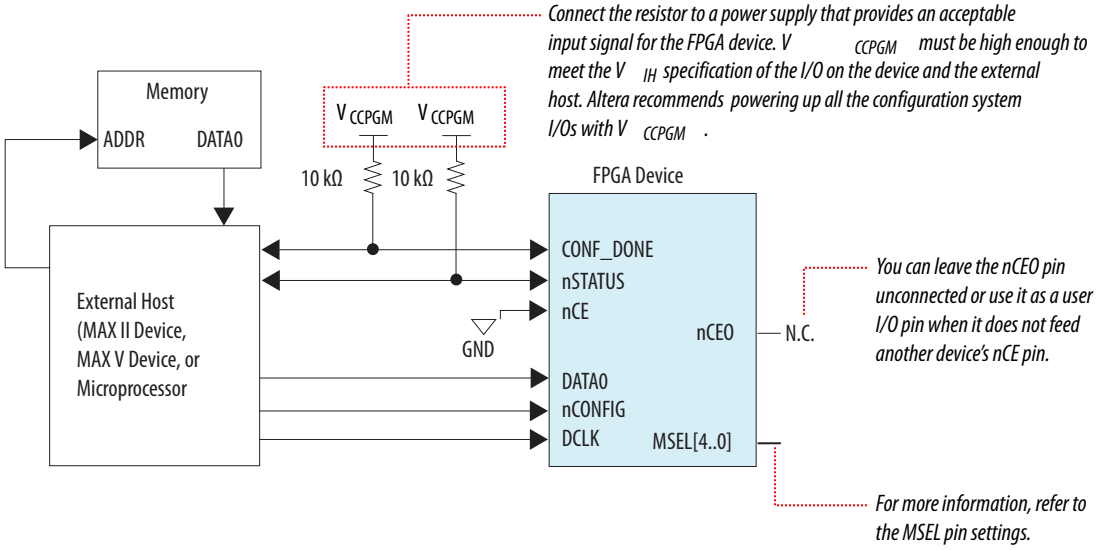
Related Information

- **Parallel Flash Loader Megafunction User Guide**
- **Stratix V Device Datasheet**
  Provides more information about the PS configuration timing.

Passive Serial Single-Device Configuration Using an External Host

To configure a Stratix V device, connect the device to an external host, as shown in the following figure.

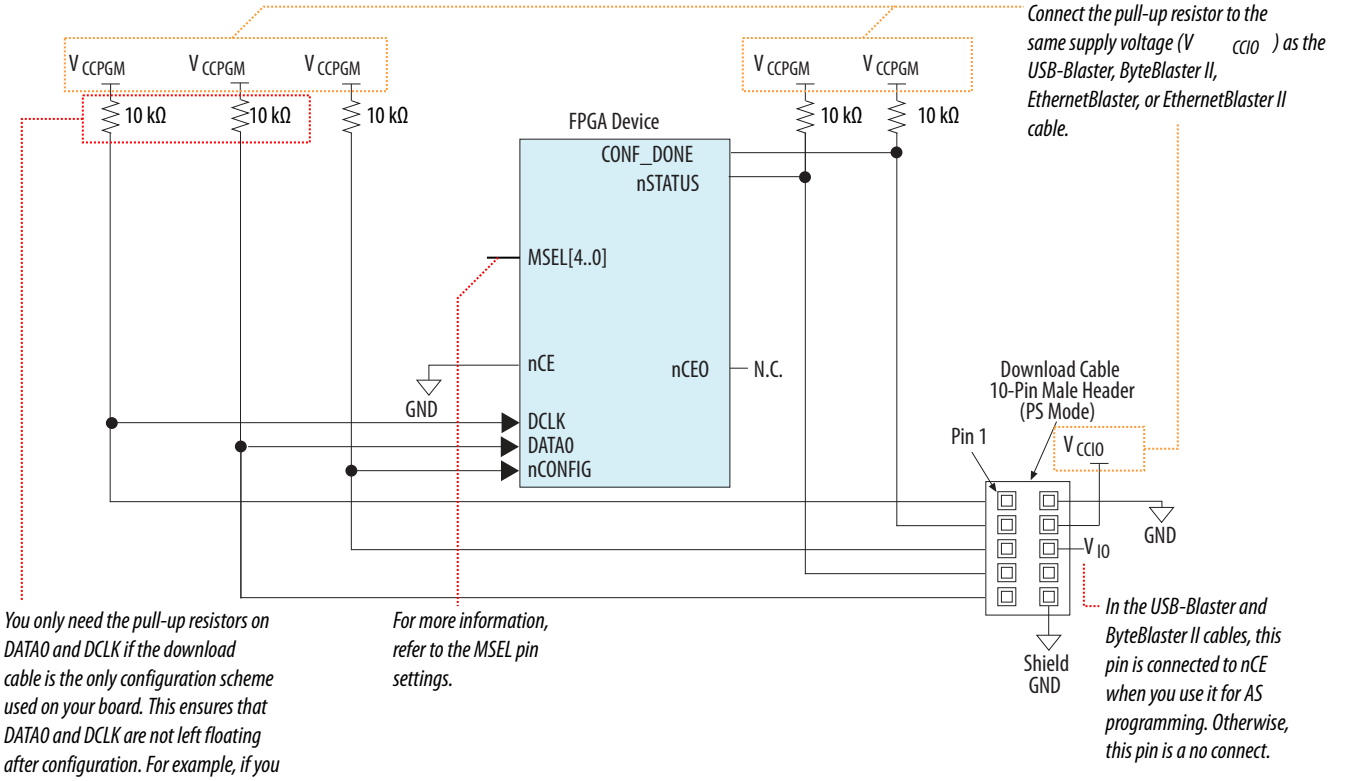
Figure 8-12: Single Device PS Configuration Using an External Host



Passive Serial Single-Device Configuration Using an Altera Download Cable

To configure a Stratix V device, connect the device to a download cable, as shown in the following figure.

Figure 8-13: Single Device PS Configuration Using an Altera Download Cable



Passive Serial Multi-Device Configuration

You can configure multiple Stratix V devices that are connected in a chain.

Pin Connections and Guidelines

Observe the following pin connections and guidelines for this configuration setup:

- Tie the following pins of all devices in the chain together:
  - nCONFIG
  - nSTATUS
  - DCLK
  - DATA0
  - CONF_DONE

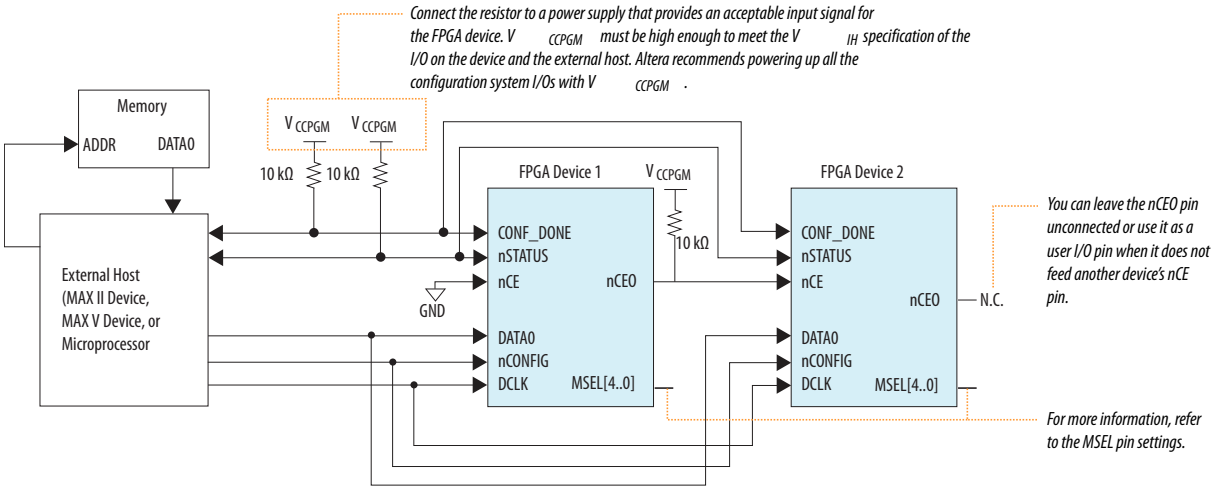
  By tying the CONF_DONE and nSTATUS pins together, the devices initialize and enter user mode at the same time. If any device in the chain detects an error, configuration stops for the entire chain and you must reconfigure all the devices. For example, if the first device in the chain flags an error on the nSTATUS pin, it resets the chain by pulling its nSTATUS pin low.
- If you are configuring the devices in the chain using the same configuration data, the devices must be of the same package and density.

Using Multiple Configuration Data

To configure multiple Stratix V devices in a chain using multiple configuration data, connect the devices to the external host as shown in the following figure.

Note: By default, the nCEO pin is disabled in the Quartus II software. For the multi-device configuration chain, you must enable the nCEO pin in the Quartus II software. Otherwise, device configuration could fail.

Figure 8-14: Multiple Device PS Configuration when Both Devices Receive Different Sets of Configuration Data



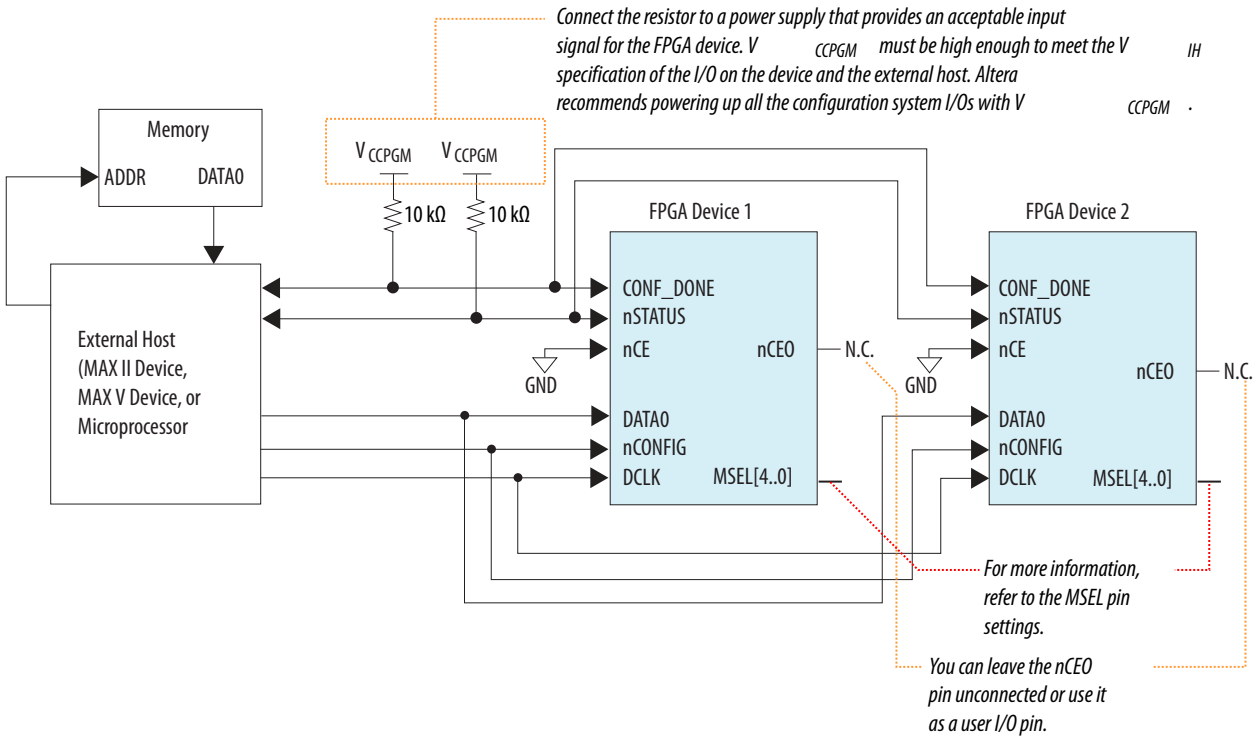
After a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device in the chain. Configuration automatically begins for the second device in one clock cycle.

Using One Configuration Data

To configure multiple Stratix V devices in a chain using one configuration data, connect the devices to an external host, as shown in the following figure.

Note: By default, the nCEO pin is disabled in the Quartus II software. For the multi-device configuration chain, you must enable the nCEO pin in the Quartus II software. Otherwise, device configuration could fail.

Figure 8-15: Multiple Device PS Configuration When Both Devices Receive the Same Set of Configuration Data



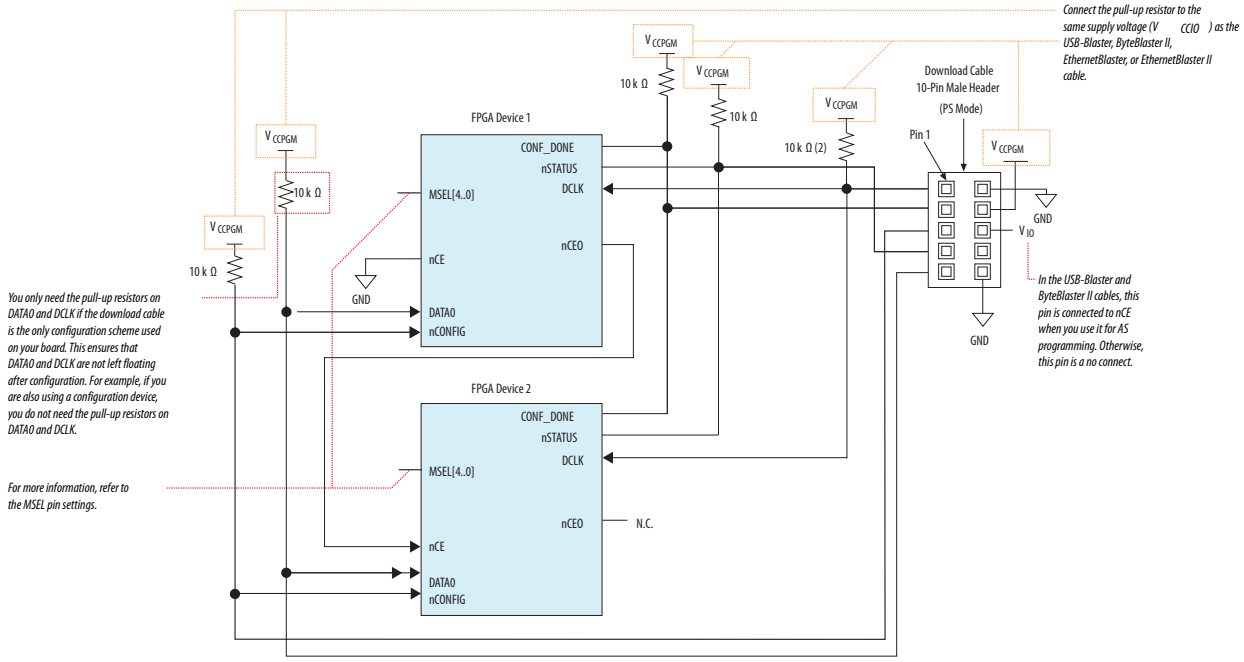
The nCE pins of the devices in the chain are connected to GND, allowing configuration for these devices to begin and end at the same time.

Using PC Host and Download Cable

To configure multiple Stratix V devices, connect the devices to a download cable, as shown in the following figure.

Note: By default, the nCEO pin is disabled in the Quartus II software. For the multi-device configuration chain, you must enable the nCEO pin in the Quartus II software. Otherwise, device configuration could fail.

Figure 8-16: Multiple Device PS Configuration Using an Altera Download Cable



When a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device. Configuration automatically begins for the second device.

JTAG Configuration

In Stratix V devices, JTAG instructions take precedence over other configuration schemes.

The Quartus II software generates an SRAM Object File (**.sof**) that you can use for JTAG configuration using a download cable in the Quartus II software programmer. Alternatively, you can use the JRunner software with **.rbf** or a JAM™ Standard Test and Programming Language (STAPL) Format File (**.jam**) or JAM Byte Code File (**.jbc**) with other third-party programmer tools.

Note: You cannot use the Stratix V decompression or design security features if you are configuring your Stratix V device using JTAG-based configuration.

The chip-wide reset (DEV_CLRn) and chip-wide output enable (DEV_OE) pins on Stratix V devices do not affect JTAG boundary-scan or programming operations.

Related Information

- **JTAG Boundary-Scan Testing in Stratix V Devices** on page 10-1
  Provides more information about JTAG boundary-scan testing.
- **Device Configuration Pins** on page 8-6
  Provides more information about JTAG configuration pins.
- **JTAG Secure Mode** on page 8-40
- **AN 425: Using the Command-Line Jam STAPL Solution for Device Programming**

- **Stratix V Device Datasheet**
  Provides more information about the JTAG configuration timing.
- **JTAG Boundary-Scan Testing in Stratix V Devices**
- **Programming Support for Jam STAPL Language**
- **USB-Blaster Download Cable User Guide**
- **ByteBlaster II Download Cable User Guide**
- **EthernetBlaster Communications Cable User Guide**
- **EthernetBlaster II Communications Cable User Guide**

JTAG Single-Device Configuration

To configure a single device in a JTAG chain, the programming software sets the other devices to the bypass mode. A device in a bypass mode transfers the programming data from the `TDI` pin to the `TDO` pin through a single bypass register. The configuration data is available on the `TDO` pin one clock cycle later.

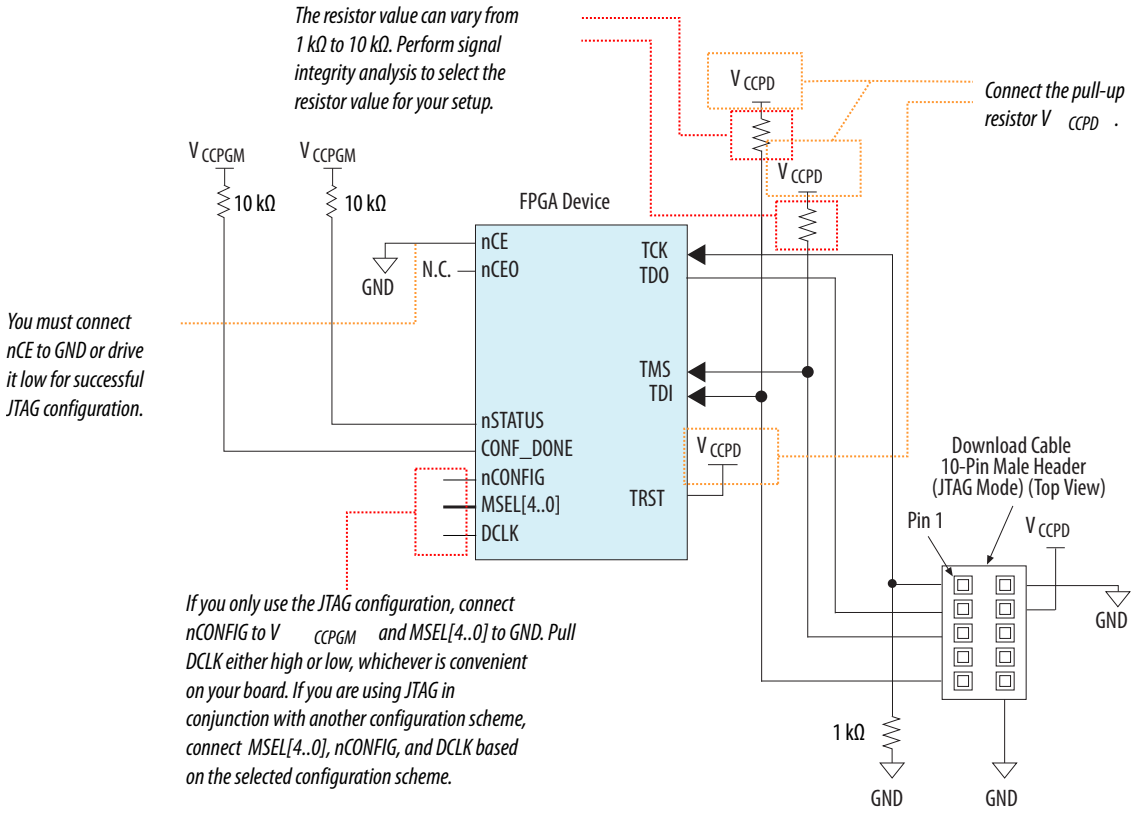
The Quartus II software can use the `CONF_DONE` pin to verify the completion of the configuration process through the JTAG port:

- `CONF_DONE` pin is low—indicates that configuration has failed.
- `CONF_DONE` pin is high—indicates that configuration was successful.

After the configuration data is transmitted serially using the JTAG `TDI` port, the `TCK` port is clocked an additional 1,222 cycles to perform device initialization.

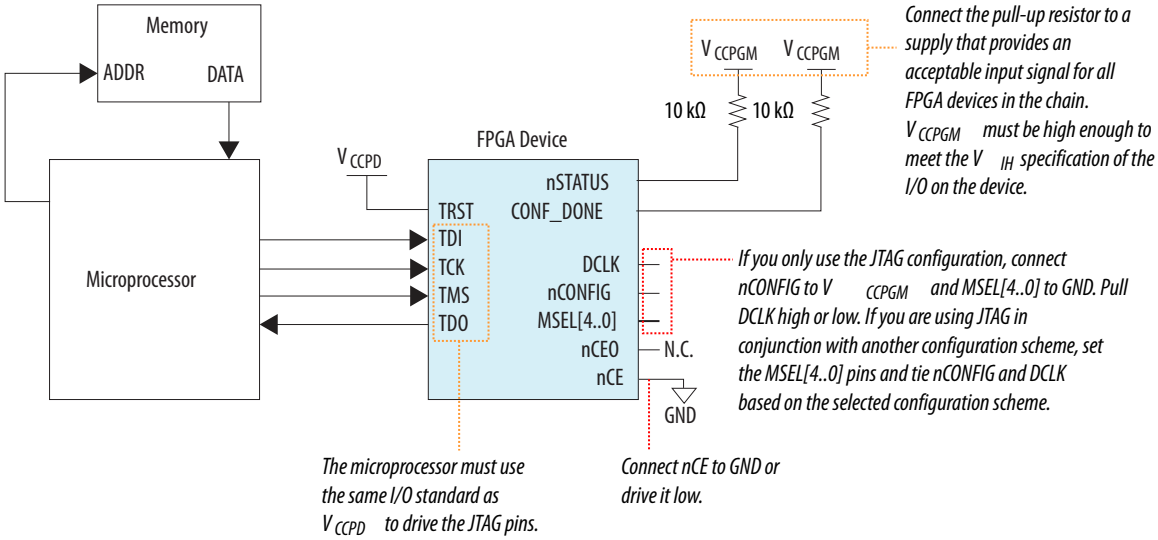
To configure a Stratix V device using a download cable, connect the device as shown in the following figure.

Figure 8-17: JTAG Configuration of a Single Device Using a Download Cable



To configure Stratix V device using a microprocessor, connect the device as shown in the following figure. You can use JRunner as your software driver.

Figure 8-18: JTAG Configuration of a Single Device Using a Microprocessor



Related Information

AN 414: The JRunner Software Driver: An Embedded Solution for PLD JTAG Configuration

JTAG Multi-Device Configuration

You can configure multiple devices in a JTAG chain.

Pin Connections and Guidelines

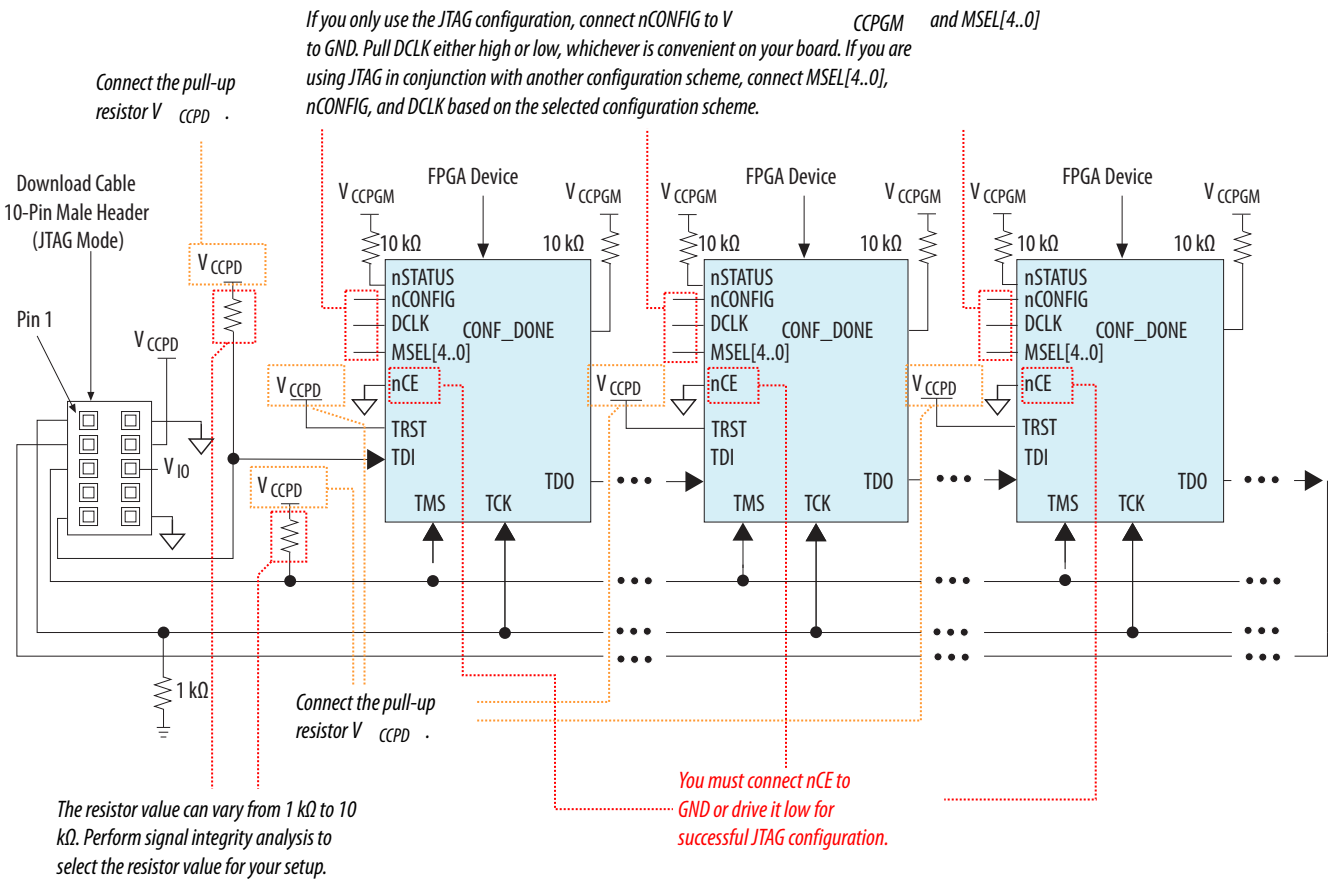
Observe the following pin connections and guidelines for this configuration setup:

- Isolate the `CONF_DONE` and `nSTATUS` pins to allow each device to enter user mode independently.
- One JTAG-compatible header is connected to several devices in a JTAG chain. The number of devices in the chain is limited only by the drive capability of the download cable.
- If you have four or more devices in a JTAG chain, buffer the `TCK`, `TDI`, and `TMS` pins with an on-board buffer. You can also connect other Altera devices with JTAG support to the chain.
- JTAG-chain device programming is ideal when the system contains multiple devices or when testing your system using the JTAG boundary-scan testing (BST) circuitry.

Using a Download Cable

The following figure shows a multi-device JTAG configuration.

Figure 8-19: JTAG Configuration of Multiple Devices Using a Download Cable



Related Information

AN 656: Combining Multiple Configuration Schemes
Provides more information about combining JTAG configuration with other configuration schemes.

CONFIG_IO JTAG Instruction

The CONFIO_IO JTAG instruction allows you to configure the I/O buffers using the JTAG port before or during device configuration. When you issue this instruction, it interrupts configuration and allows you to issue all JTAG instructions. Otherwise, you can only issue the BYPASS, IDCODE, and SAMPLE JTAG instructions.

You can use the CONFIO_IO JTAG instruction to interrupt configuration and perform board-level testing. After the board-level testing is completed, you must reconfigure your device. Use the following methods to reconfigure your device:

- JTAG interface—issue the PULSE_NCONFIG JTAG instruction.
- FPP, PS, or AS configuration scheme—pulse the nCONFIG pin low.

Configuration Data Compression

Stratix V devices can receive compressed configuration bitstream and decompress the data in real-time during configuration. Preliminary data indicates that compression typically reduces the configuration file size by 30% to 55% depending on the design.

Decompression is supported in all configuration schemes except the JTAG configuration scheme.

You can enable compression before or after design compilation.

Enabling Compression Before Design Compilation

To enable compression before design compilation, follow these steps:

1. On the Assignment Menu, click **Device**.
2. Select your Stratix V device and then click **Device and Pin Options**.
3. In the **Device and Pin Options** window, select **Configuration** under the **Category** list and turn on **Generate compressed bitstreams**.

Enabling Compression After Design Compilation

To enable compression after design compilation, follow these steps:

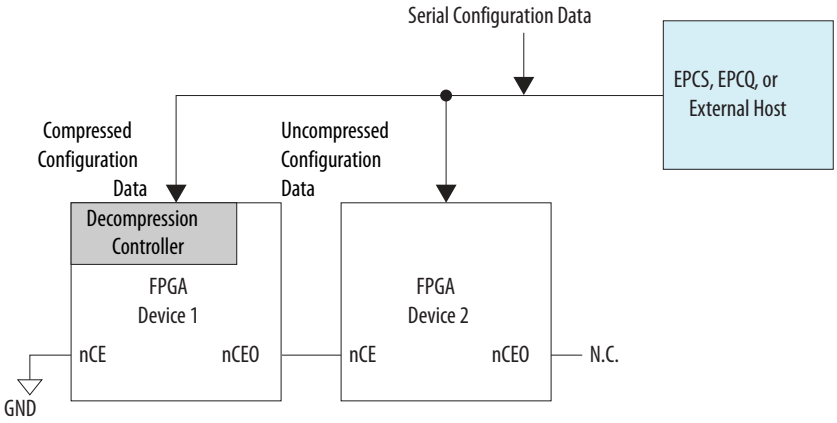
1. On the File menu, click **Convert Programming Files**.
2. Select the programming file type (**.pof**, **.sof**, **.hex**, **.hexout**, **.rbf**, or **.ttf**). For POF output files, select a configuration device.
3. Under the **Input files to convert** list, select **SOF Data**.
4. Click **Add File** and select a Stratix V device **.sof**.
5. Select the name of the file you added to the **SOF Data** area and click **Properties**.
6. Turn on the **Compression** check box.

Using Compression in Multi-Device Configuration

The following figure shows a chain of two Stratix V devices. Compression is only enabled for the first device.

This setup is supported by the AS or PS multi-device configuration only.

Figure 8-20: Compressed and Uncompressed Serial Configuration Data in the Same Configuration File

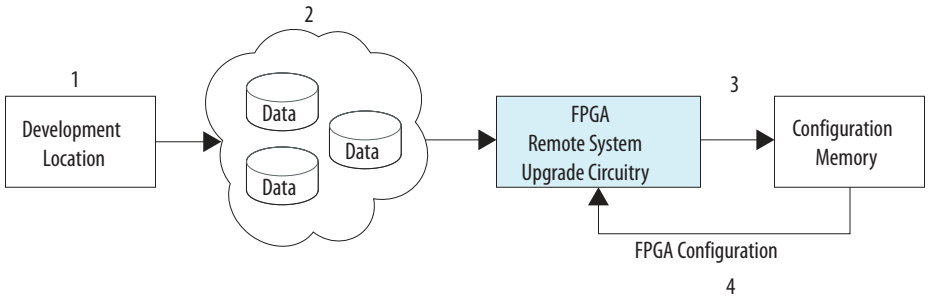


For the FPP configuration scheme, a combination of compressed and uncompressed configuration in the same multi-device configuration chain is not allowed because of the difference on the DCLK-to-DATA[] ratio.

Remote System Upgrades

Stratix V devices contain dedicated remote system upgrade circuitry. You can use this feature to upgrade your system from a remote location.

Figure 8-21: Stratix V Remote System Upgrade Block Diagram



You can design your system to manage remote upgrades of the application configuration images in the configuration device. The following list is the sequence of the remote system upgrade:

1. The logic (embedded processor or user logic) in the Stratix V device receives a configuration image from a remote location. You can connect the device to the remote source using communication protocols such as TCP/IP, PCI, user datagram protocol (UDP), UART, or a proprietary interface.
2. The logic stores the configuration image in non-volatile configuration memory.
3. The logic starts reconfiguration cycle using the newly received configuration image.
4. When an error occurs, the circuitry detects the error, reverts to a safe configuration image, and provides error status to your design.

Configuration Images

Each Stratix V device in your system requires one factory image. The factory image is a user-defined configuration image that contains logic to perform the following:

- Processes errors based on the status provided by the dedicated remote system upgrade circuitry.
- Communicates with the remote host, receives new application images, and stores the images in the local non-volatile memory device.
- Determines the application image to load into the Stratix V device.
- Enables or disables the user watchdog timer and loads its time-out value.
- Instructs the dedicated remote system upgrade circuitry to start a reconfiguration cycle.

You can also create one or more application images for the device. An application image contains selected functionalities to be implemented in the target device.

Store the images at the following locations in the EPCS or EPCQ devices:

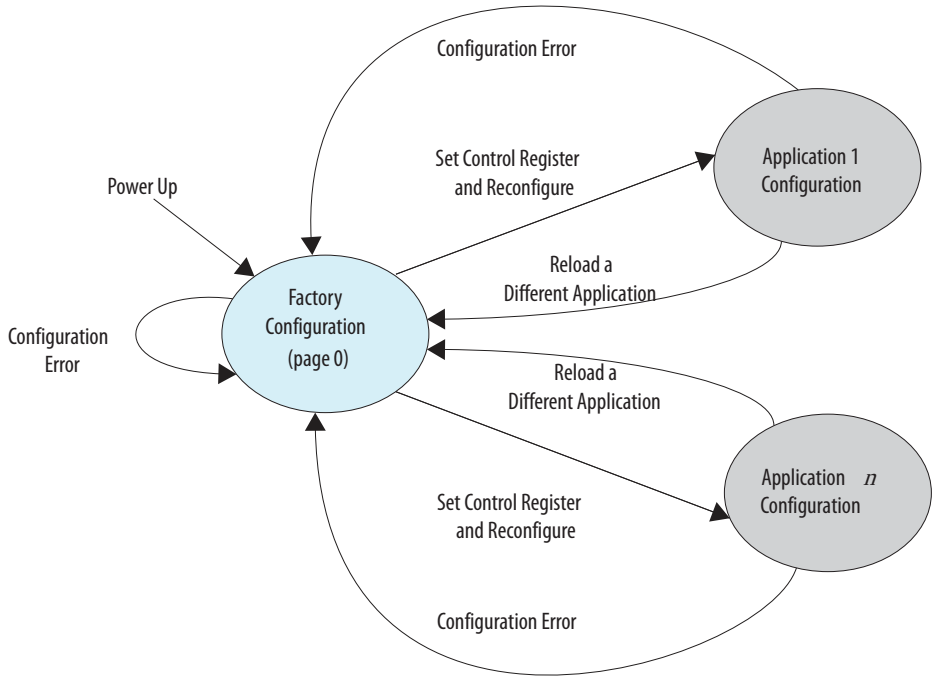
- Factory configuration image—`PGM[23..0]` = 24'h000000 start address on the EPCS or EPCQ device.
- Application configuration image—any sector boundary. Altera recommends that you store only one image at one sector boundary.

When you are using EPCQ 256, ensure that the application configuration image address granularity is 32'h00000100. The granularity requirement is having the most significant 24 bits of the 32 bits start address written to `PGM[23..0]` bits.

Note: If you are not using the Quartus II software or SRunner software for EPCQ 256 programming, put your EPCQ 256 device into four-byte addressing mode before you program and configure your device.

Configuration Sequence in the Remote Update Mode

Figure 8-22: Transitions Between Factory and Application Configurations in Remote Update Mode



Related Information

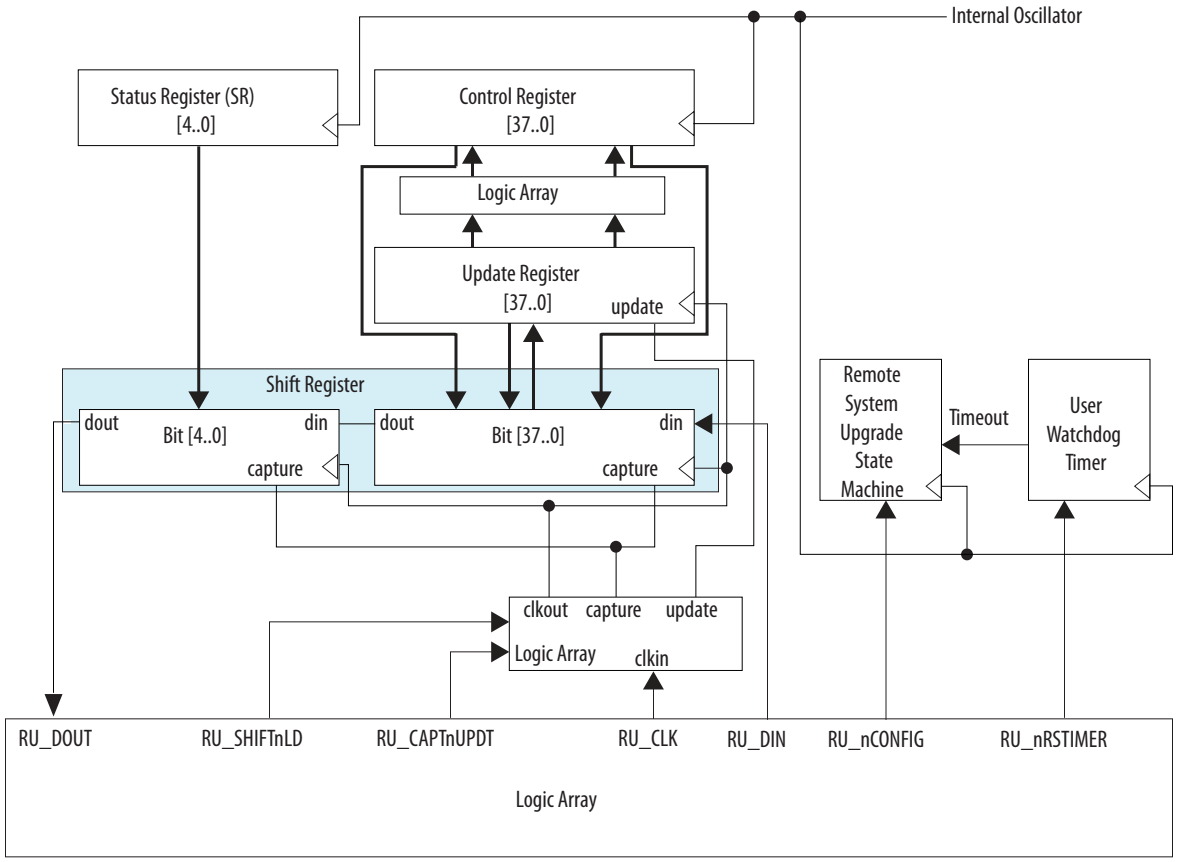
Remote System Upgrade State Machine on page 8-38
A detailed description of the configuration sequence in the remote update mode.

Remote System Upgrade Circuitry

The remote system upgrade circuitry contains the remote system upgrade registers, watchdog timer, and a state machine that controls these components.

Note: If you are using the ALTREMOTE_UPDATE megafunction, the megafunction controls the `RU_DOUT`, `RU_SHIFTnLD`, `RU_CAPTnUPDT`, `RU_CLK`, `RU_DIN`, `RU_nCONFIG`, and `RU_nRSTIMER` signals internally to perform all the related remote system upgrade operations.

Figure 8-23: Remote System Upgrade Circuitry



Related Information

Stratix V Device Datasheet
Provides more information about remote system upgrade circuitry timing specifications.

Enabling Remote System Upgrade Circuitry

To enable the remote system upgrade feature, follow these steps:

1. Select **Active Serial x1/x4** or **Configuration Device** from the Configuration scheme list in the **Configuration** page of the **Device and Pin Options** dialog box in the Quartus II software.
2. Select **Remote** from the Configuration mode list in the **Configuration** page of the **Device and Pin Options** dialog box in the Quartus II software.

Enabling this feature automatically turns on the **Auto-restart configuration after error** option.

Altera-provided ALTREMOTE_UPDATE megafunction provides a memory-like interface to the remote system upgrade circuitry and handles the shift register read and write protocol in the Stratix V device logic.

Related Information

Remote System Upgrade (ALTREMOTE_UPDATE) Megafunction User Guide

Remote System Upgrade Registers

Table 8-9: Remote System Upgrade Registers

Register	Description
Shift	Accessible by the logic array and clocked by RU_CLK. • Bits[4..0]—Contents of the status register are shifted into these bits. • Bits[37..0]—Contents of the update and control registers are shifted into these bits.
Control	This register is clocked by the 10-MHz internal oscillator. The contents of this register are shifted to the shift register for the user logic in the application configuration to read. When reconfiguration is triggered, this register is updated with the contents of the update register.
Update	This register is clocked by RU_CLK. The factory configuration updates this register by shifting data into the shift register and issuing an update. When reconfiguration is triggered, the contents of the update register are written to the control register.
Status	After each reconfiguration, the remote system upgrade circuitry updates this register to indicate the event that triggered the reconfiguration. This register is clocked by the 10-MHz internal oscillator.

Related Information

- **Control Register** on page 8-37
- **Status Register** on page 8-38

Control Register

Table 8-10: Control Register Bits

Bit	Name	Reset Value[13]	Description
0	AnF	1'b0	Application not Factory bit. Indicates the configuration image type currently loaded in the device; **0** for factory image and **1** for application image. When this bit is **1**, the access to the control register is limited to read only and the watchdog timer is enabled. Factory configuration design must set this bit to **1** before triggering reconfiguration using an application configuration image.
1..24	PGM[0..23]	24'h000000	Upper 24 bits of AS configuration start address (StAdd[31..8]), the 8 LSB are zero.

(13) This is the default value after the device exits POR and during reconfiguration back to the factory configuration image.

Bit	Name	Reset Value[13]	Description
25	Wd_en	1'b0	User watchdog timer enable bit. Set this bit to **1** to enable the watchdog timer.
26..37	Wd_timer[11..0]	12'b000000000000	User watchdog time-out value.

Status Register

Table 8-11: Status Register Bits

Bit	Name	Reset Value[14]	Description
0	CRC	1'b0	When set to **1**, indicates CRC error during application configuration.
1	nSTATUS	1'b0	When set to **1**, indicates that nSTATUS is asserted by an external device due to error.
2	Core_nCONFIG	1'b0	When set to **1**, indicates that reconfiguration has been triggered by the logic array of the device.
3	nCONFIG	1'b0	When set to **1**, indicates that nCONFIG is asserted.
4	Wd	1'b0	When set to **1**, indicates that the user watchdog time-out.

Remote System Upgrade State Machine

The operation of the remote system upgrade state machine is as follows:

1. After power-up, the remote system upgrade registers are reset to **0** and the factory configuration image is loaded.
2. The user logic sets the AnF bit to **1** and the start address of the application image to be loaded. The user logic also writes the watchdog timer settings.
3. When the configuration reset (RU_CONFIG) goes low, the state machine updates the control register with the contents of the update register, and triggers reconfiguration using the application configuration image.
4. If error occurs, the state machine falls back to the factory image. The control and update registers are reset to **0**, and the status register is updated with the error information.
5. After successful reconfiguration, the system stays in the application configuration.

User Watchdog Timer

The user watchdog timer prevents a faulty application configuration from stalling the device indefinitely. You can use the timer to detect functional errors when an application configuration is successfully loaded

(13) This is the default value after the device exits POR and during reconfiguration back to the factory configuration image.
(14) After the device exits POR and power-up, the status register content is 5'b00000.

into the device. The timer is automatically disabled in the factory configuration; enabled in the application configuration.

Note: If you do not want this feature in the application configuration, you need to turn off this feature by setting the `Wd_en` bit to **1'b0** in the update register during factory configuration user mode operation. You cannot disable this feature in the application configuration.

The counter is 29 bits wide and has a maximum count value of 2^{29}. When specifying the user watchdog timer value, specify only the most significant 12 bits. The granularity of the timer setting is 2^{17} cycles. The cycle time is based on the frequency of the user watchdog timer internal oscillator.

The timer begins counting as soon as the application configuration enters user mode. When the timer expires, the remote system upgrade circuitry generates a time-out signal, updates the status register, and triggers the loading of the factory configuration image. To reset the time, assert `RU_nRSTIMER`.

Related Information

Stratix V Device Datasheet
Provides more information about the operating range of the user watchdog internal oscillator's frequency.

Design Security

The Stratix V design security feature supports the following capabilities:

- Enhanced built-in advanced encryption standard (AES) decryption block to support 256-bit key industry-standard design security algorithm (FIPS-197 Certified)
- Volatile and non-volatile key programming support
- Secure operation mode for both volatile and non-volatile key through tamper protection bit setting
- Limited accessible JTAG instruction during power-up in the JTAG secure mode
- Supports board-level testing
- Supports in-socket key programming for non-volatile key
- Available in all configuration schemes except JTAG
- Supports both remote system upgrades and compression features

The Stratix V design security feature provides the following security protection for your designs:

- Security against copying—the security key is securely stored in the Stratix V device and cannot be read out through any interface. In addition, as configuration file read-back is not supported in Stratix V devices, your design information cannot be copied.
- Security against reverse engineering—reverse engineering from an encrypted configuration file is very difficult and time consuming because the Stratix V configuration file formats are proprietary and the file contains millions of bits that require specific decryption.
- Security against tampering—After you set the tamper protection bit, the Stratix V device can only accept configuration files encrypted with the same key. Additionally, programming through the JTAG interface and configuration interface is blocked.

When you use compression with the design security feature, the configuration file is first compressed and then encrypted using the Quartus II software. During configuration, the device first decrypts and then decompresses the configuration file.

When you use design security with Stratix V devices in an FPP configuration scheme, it requires a different `DCLK-to-DATA[]` ratio.

ALTCHIP_ID Megafunction

The ALTCHIP_ID megafunction provides the following features:

- Acquiring the chip ID of an FPGA device.
- Allowing you to identify your device in your design as part of a security feature to protect your design from an unauthorized device.

Related Information

ALTCHIP_ID Megafunction User Guide

JTAG Secure Mode

When you enable the tamper-protection bit, Stratix V devices are in the JTAG secure mode after power-up. During this mode, many JTAG instructions are disabled. Stratix V devices only allow mandatory JTAG 1149.1 and 1149.6 instructions to be exercised. These JTAG instructions are `SAMPLE/PRELOAD`, `BYPASS`, `EXTEST`, and optional instructions such as `IDCODE` and `SHIFT_EDERROR_REG`.

To enable the access of other JTAG instructions such as `USERCODE`, `HIGHZ`, `CLAMP`, `PULSE_nCONFIG`, and `CONFIG_IO`, you must issue the `UNLOCK` instruction to deactivate the JTAG secure mode. You can issue the `LOCK` instruction to put the device back into JTAG secure mode. You can only issue both the `LOCK` and `UNLOCK` JTAG instructions during user mode.

Related Information

- **Supported JTAG Instruction** on page 10-3
  Provides more information about JTAG binary instruction code related to the `LOCK` and `UNLOCK` instructions.
- **JTAG Boundary-Scan Testing in Stratix V Devices**
  Provides more information about JTAG binary instruction code related to the `LOCK` and `UNLOCK` instructions.

Security Key Types

Stratix V devices offer two types of keys—volatile and non-volatile. The following table lists the differences between the volatile key and non-volatile keys.

Table 8-12: Security Key Types

Key Types	Key Programmability	Power Supply for Key Storage	Programming Method
Volatile	• Reprogrammable • Erasable	Required external battery, V_{CCBAT} (15)	On-board
Non-volatile	One-time programming	Does not require an external battery	On-board and in-socket programming (16)

(15) V_{CCBAT} is a dedicated power supply for volatile key storage. V_{CCBAT} continuously supplies power to the volatile register regardless of the on-chip supply condition.
(16) Third-party vendors offer in-socket programming.

Both non-volatile and volatile key programming offers protection from reverse engineering and copying. If you set the tamper-protection bit, the design is also protected from tampering.

You can perform key programming through the JTAG pins interface. Ensure that the nSTATUS pin is released high before any key-programming attempts.

Note: To clear the volatile key, issue the KEY_CLR_VREG JTAG instruction. To verify the volatile key has been cleared, issue the KEY_VERIFY JTAG instruction.

Related Information

- **Supported JTAG Instruction** on page 10-3
  Provides more information about the KEY_CLR_VREG and KEY_VERIFY instructions.
- **JTAG Boundary-Scan Testing in Stratix V Devices**
  Provides more information about the KEY_CLR_VREG and KEY_VERIFY JTAG instructions.
- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides more information about the V_{CCBAT} pin connection recommendations.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides more information about the V_{CCBAT} pin connection recommendations.
- **Stratix V Device Datasheet**
  Provides more information about battery specifications.

Security Modes

Table 8-13: Supported Security Modes

There is no impact to the configuration time required when compared with unencrypted configuration modes except FPP with AES (and/or decompression), which requires a DCLK that is up to ×4 the data rate.

Security Mode	Tamper Protection Bit Setting	Device Accepts Unencrypted File	Device Accepts Encrypted File	Security Level
No key	—	Yes	No	—
Volatile Key	—	Yes	Yes	Secure
Volatile Key with Tamper Protection Bit Set	Set	No	Yes	Secure with tamper resistant
Non-volatile Key	—	Yes	Yes	Secure
Non-volatile Key with Tamper Protection Bit Set	Set	No	Yes	Secure with tamper resistant

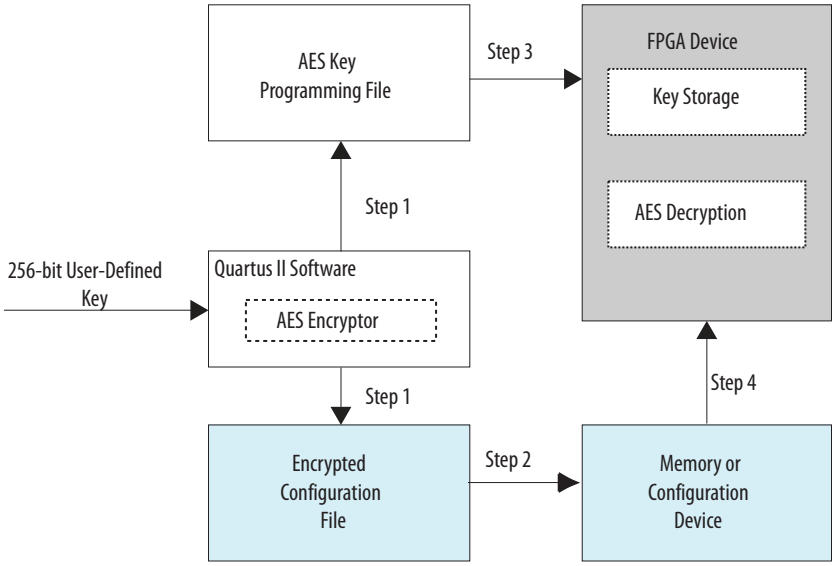
The use of unencrypted configuration bitstream in the volatile key and non-volatile key security modes is supported for board-level testing only.

Note: For the volatile key with tamper protection bit set security mode, Stratix V devices do not accept the encrypted configuration file if the volatile key is erased. If the volatile key is erased and you want to reprogram the key, you must use the volatile key security mode.

Enabling the tamper protection bit disables the test mode in Stratix V devices and disables programming through the JTAG interface. This process is irreversible and prevents Altera from carrying out failure analysis.

Design Security Implementation Steps

Figure 8-24: Design Security Implementation Steps



To carry out secure configuration, follow these steps:

1. The Quartus II software generates the design security key programming file and encrypts the configuration data using the user-defined 256-bit security key.
2. Store the encrypted configuration file in the external memory.
3. Program the AES key programming file into the Stratix V device through a JTAG interface.
4. Configure the Stratix V device. At the system power-up, the external memory device sends the encrypted configuration file to the Stratix V device.

Document Revision History

Date	Version	Changes
January 2015	2015.01.23	Added the Transmitting Configuration Data section.

Date	Version	Changes
June 2014	2014.06.30	• Updated Figure 8-17: JTAG Configuration of a Single Device Using a Download Cable. • Updated Figure 8-19: JTAG Configuration of Multiple Devices Using a Download Cable. • Updated the maximum clock rate for Partial Reconfiguration in Table 8-1. • Updated the MSEL pin settings recommendation in the MSEL Pin Settings section.
January 2014	2014.01.10	• Updated the Enabling Remote System Upgrade Circuitry section. • Updated the Configuration Pin Summary section. • Updated Figure 8-3, Figure 8-7, and Figure 8-14.
June 2013	2013.06.11	Updated the Configuration Error Handling section.
May 2013	2013.05.10	Removed support for active serial multi-device configuration using the same configuration data.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Added the ALTCHIP_ID megafunction section. • Added links for AS, PS, FPP, and JTAG configuration timing to device datasheet. • Updated "Connection Setup for Programming the EPCS Using the JTAG Interface" and "Connection Setup for Programming the EPCQ Using the JTAG Interface" figures. • Updated CvP support for partial reconfiguration in the Table 8-1: Configuration Modes and Features Supported by Stratix V Devices. • Moved all links to the Related Information section of respective topics for easy reference.
March 2013	2013.03.04	Remove a note to the `nIO_PULLUP` pin in Table 8-3: Configuration Pin Summary for Stratix V Devices.
December 2012	2012.12.28	• Added configuration modes and features for Stratix V devices. • Reorganized content and updated template.
June 2012	1.7	• Added MAX V devices. • Updated Figure 9-2, Figure 9-3, Figure 9-11, Figure 9-16, Figure 9-17, Figure 9-20, and Figure 9-23. • Updated Table 9-4, Table 9-5, Table 9-7, Table 9-11, and Table 9-12. • Updated "MSEL Pin Settings" and "FPP Multi-Device Configuration" sections.
February 2012	1.6	• Updated "Security Key Types" section. • Updated Table 9-10.

Date	Version	Changes
December 2011	1.5	• Updated "FPP Configuration Timing", "JTAG Secure Mode", and "Security Key Types" sections. • Updated Table 9-8.
November 2011	1.4	• Updated Table 9-5, Table 9-9, and Table 9-14. • Updated Figure 9-8, Figure 9-9, and Figure 9-21. • Updated "AS Multi-Device Configuration" and "Active Serial Configuration (Serial Configuration Devices)" sections.
May 2011	1.3	• Chapter moved to volume 2 for the 11.0 release. • Added "Remote System Upgrades Using EPCQ 256" and "JTAG Secure Mode" sections. • Updated Table 9-5. • Updated "Configuration", "Configuration Error", "Programming EPCS and EPCQ", "JTAG Configuration", "Remote Update Mode", and "Design Security" sections. • Minor text edits.
January 2011	1.2	• Updated Table 9-7, Table 9-8, Table 9-12, and Table 9-14. • Updated Figure 9-15 and Figure 9-21. • Updated "User Watchdog Timer", "DCLK-to-DATA[] Ratio for FPP Configuration", "VCCPD Pin", "POR Delay Specification", and "Programming EPCS and EPCQ" sections.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1.
July 2010	1.0	Initial release.

SEU Mitigation for Stratix V Devices

9

2015.01.23

SV51011 Subscribe Send Feedback

This chapter describes the error detection features in Stratix V devices. You can use these features to mitigate single event upset (SEU) or soft errors.

Related Information

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

Error Detection Features

The on-chip error detection CRC circuitry allows you to perform the following operations without any impact on the fitting or performance of the device:

- Auto-detection of CRC errors during configuration.
- Optional CRC error detection and identification in user mode.
- Optional internal scrubbing in user mode. When enabled, this feature corrects single-bit and double-adjacent errors automatically.
- Testing of error detection functions by deliberately injecting errors through the JTAG interface.

Configuration Error Detection

When the Quartus II software generates the configuration bitstream, the software also computes a 16-bit CRC value for each frame. A configuration bitstream can contain more than one CRC values depending on the number of data frames in the bitstream. The length of the data frame varies for each device.

When a data frame is loaded into the FPGA during configuration, the precomputed CRC value shifts into the CRC circuitry. At the same time, the CRC engine in the FPGA computes the CRC value for the data frame and compares it against the precomputed CRC value. If both CRC values do not match, the nSTATUS pin is set to low to indicate a configuration error.

You can test the capability of this feature by modifying the configuration bitstream or intentionally corrupting the bitstream during configuration.

User Mode Error Detection

In user mode, the contents of the configured CRAM bits may be affected by soft errors. These soft errors, which are caused by an ionizing particle, are not common in Altera devices. However, high-reliability applications that require the device to operate error-free may require that your designs account for these errors.

You can enable the error detection circuitry to detect soft errors. Each data frame stored in the CRAM contains a 32-bit precomputed CRC value. When this feature is enabled, the error detection circuitry continuously computes a 32-bit CRC value for each frame in the CRAM and compares the CRC value against the precomputed value.

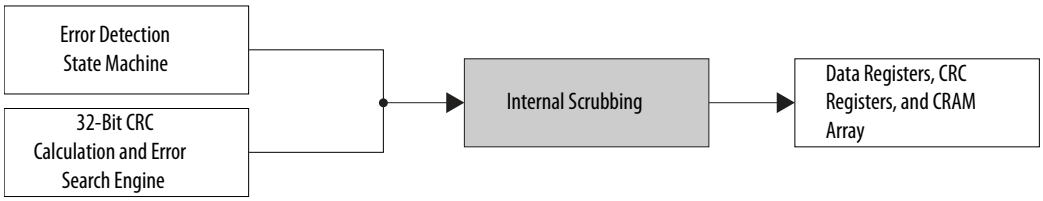
- If the CRC values match, the 32-bit CRC signature in the `syndrome` register is set to zero to indicate that no error is detected.
- Otherwise, the resulting 32-bit CRC signature in the `syndrome` register is non-zero to indicate a CRC error. The `CRC_ERROR` pin is pulled high, and the error type and location are identified.

Within a frame, the error detection circuitry can detect all single-, double-, triple-, quadruple-, and quintuple-bit errors. When a single-bit or double-adjacent error is detected, the error detection circuitry reports the bit location and determines the error type for single-bit and double-adjacent errors. The probability of other error patterns is very low and the reporting of bit location is not guaranteed. The probability of more than five CRAM bits being flipped by soft errors is very low. In general, the probability of detection for all error patterns is 99.9999%. The process of error detection continues until the device is reset by setting the `nCONFIG` signal low.

Internal Scrubbing

Internal scrubbing is the ability to internally correct soft errors in user mode. This feature corrects single-bit and double-adjacent errors detected in each data frame without the need to reconfigure the device.

Figure 9-1: Block Diagram



Specifications

This section lists the EMR update interval, error detection frequencies, and CRC calculation time for error detection in user mode.

Minimum EMR Update Interval

The interval between each update of the error message register depends on the device and the frequency of the error detection clock. Using a lower clock frequency increases the interval time, hence increasing the time required to recover from a single event upset (SEU).

Table 9-1: Estimated Minimum EMR Update Interval in Stratix V Devices

Variant	Member Code	Package	Timing Interval (µs)
Stratix V GX	A3	EH29-H780	3.13
		HF35-F1152	3.13
		KF35-F1152	3.13
		KF40-F1517/KH40-H1517	3.13
	A4	All	3.13
	A5		3.71
	A7		3.71
	A9		5.01
	AB		5.01
	B5		3.85
	B6		3.85
Stratix V GT	C5	All	3.71
	C7		3.71
Stratix V GS	D3	All	2.61
	D4	EH29-H780	2.61
		HF35-F1152	2.61
		KF40-F1517/KH40-H1517	3.13
	D5	All	3.13
	D6		4.33
	D8		4.33
Stratix V E	E9	All	5.01
	EB		5.01

Error Detection Frequency

You can control the speed of the error detection process by setting the division factor of the clock frequency in the Quartus II software. The divisor is 2^n, where n can be any value listed in the following table.

The speed of the error detection process for each data frame is determined by the following equation:

Figure 9-2: Error Detection Frequency Equation

$$\text{Error Detection Frequency} = \frac{\text{Internal Oscillator Frequency}}{2^n}$$

Table 9-2: Error Detection Frequency Range for Stratix V Devices

The following table lists the frequencies and valid values of n.

Internal Oscillator Frequency	Error Detection Frequency		n	Divisor Range
	Maximum	Minimum		
100 MHz	100 MHz	390 kHz	0, 1, 2, 3, 4, 5, 6, 7, 8	1 – 256

CRC Calculation Time For Entire Device

While the CRC calculation is done on a per frame basis, it is important to know the time taken to complete CRC calculations for the entire device. The entire device detection time is the time taken to do CRC calculations on every frame in the device. This time depends on the device and the error detection clock frequency. The error detection clock frequency also depends on the device and on the internal oscillator frequency, which varies from 42.6 MHz to 100 MHz.

You can calculate the minimum and maximum time for any number of divisor based on the following formula:

Maximum time (n) = 2^(n-8) * t_{MAX}

Minimum time (n) = 2^n * t_{MIN}

where the range of n is from 0 to 8.

Table 9-3: Device EDCRC Detection Time in Stratix V Devices

The following table lists the minimum and maximum time taken to calculate the CRC value:

- The minimum time is derived using the maximum clock frequency with a divisor of 0.
- The maximum time is derived using the minimum clock frequency with a divisor of 8.

Variant	Member Code	Package	t_{MIN} (ms)	t_{MAX} (s)
Stratix V GX	A3	EH29-H780	38	19.42
		HF35-F1152	38	19.42
		KF35-F1152	38	19.42
		KF40-F1517/KH40-H1517	38	19.42
	A4	All	38	19.42
	A5		47	24.20
	A7		47	24.20
	A9		68	35.21
	AB		68	35.21
	B5		45	23.52
	B6		45	23.52
Stratix V GT	C5	All	47	24.20
	C7		47	24.20
Stratix V GS	D3	All	29	14.91
	D4	EH29-H780	29	14.91
		HF35-F1152	38	19.42
		KF40-F1517/KH40-H1517	38	19.42
	D5	All	38	19.42
	D6		54	27.81
	D8		54	27.81
Stratix V E	E9	All	68	35.21
	EB		68	35.21

Using Error Detection Features in User Mode

This section describes the pin, registers, process flow, and procedures for error detection in user mode.

Enabling Error Detection and Internal Scrubbing

To enable user mode error detection and internal scrubbing in the Quartus II software, follow these steps:

1. On the Assignments menu, click **Device**.
2. In the Device dialog box, click **Device and Pin Options**.
3. In the **Category** list, click **Error Detection CRC**.
4. Turn on **Enable Error Detection CRC_ERROR pin**.
5. To set the CRC_ERROR pin as output open drain, turn on **Enable open drain on CRC_ERROR pin**. Turning off this option sets the CRC_ERROR pin as output.
6. To enable the on-chip error correction feature, turn on **Enable internal scrubbing**.
7. In the **Divide error check frequency by** list, select a valid divisor.
8. Click OK.

CRC_ERROR Pin

Table 9-4: Pin Description

Pin Name	Pin Type	Description
CRC_ERROR	I/O or output/ output open-drain	An active-high signal, when driven high indicates that an error is detected in the CRAM bits. This pin is only used when you enable error detection in user mode. Otherwise, the pin is used as a user I/O pin. When using the WYSIWYG function, you can route the crcerror port from the WYSIWYG atom to the dedicated CRC_ERROR pin or any user I/O pin. To route the crcerror port to a user I/O pin, insert a D-type flipflop between them.

Error Detection Registers

This section describes the registers used in user mode.

Figure 9-3: Block Diagram for Error Detection in User Mode

The block diagram shows the registers and data flow in user mode.

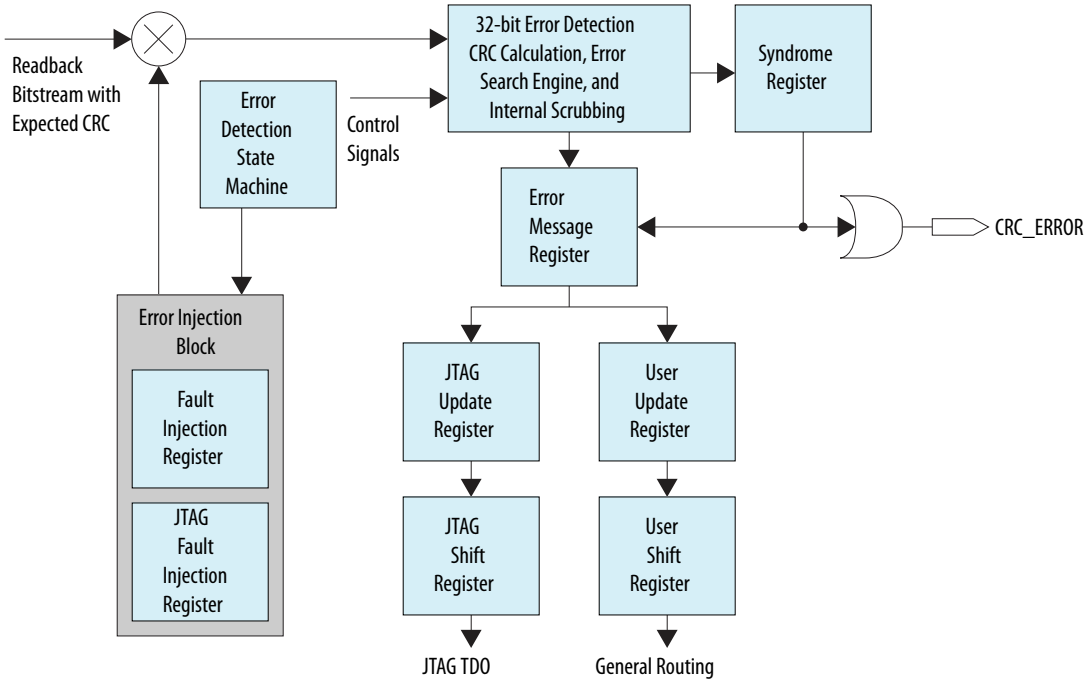


Table 9-5: Error Detection Registers

Name	Width (Bits)	Description
Syndrome register	32	Contains the 32-bit CRC signature calculated for the current frame. If the CRC value is 0, the CRC_ERROR pin is driven low to indicate no error. Otherwise, the pin is pulled high.
Error message register (EMR)	67	Contains error details for single-bit and double-adjacent errors. The error detection circuitry updates this register each time the circuitry detects an error. The Error Message Register Map figure shows the fields in this register and the Error Type in EMR table lists the possible error types.
JTAG update register	67	This register is automatically updated with the contents of the EMR one clock cycle after the content of this register is validated. The JTAG update register includes a clock enable, which must be asserted before its contents are written to the JTAG shift register. This requirement ensures that the JTAG update register is not overwritten when its contents are being read by the JTAG shift register.
JTAG shift register	67	This register allows you to access the contents of the JTAG update register via the JTAG interface using the SHIFT_EDERROR_REG JTAG instruction.

Name	Width (Bits)	Description
User update register	67	This register is automatically updated with the contents of the EMR one clock cycle after the contents of this register are validated. The user update register includes a clock enable, which must be asserted before its contents are written to the user shift register. This requirement ensures that the user update register is not overwritten when its contents are being read by the user shift register.
User shift register	67	This register allows user logic to access the contents of the user update register via the core interface.
JTAG fault injection register	46	You can use this register with the `EDERROR_INJECT JTAG` instruction to inject errors in the bitstream. The JTAG Fault Injection Register Map table lists the fields in this register.
Fault injection register	46	This register is updated with the contents of the JTAG fault injection register.

Figure 9-4: Error Message Register Map

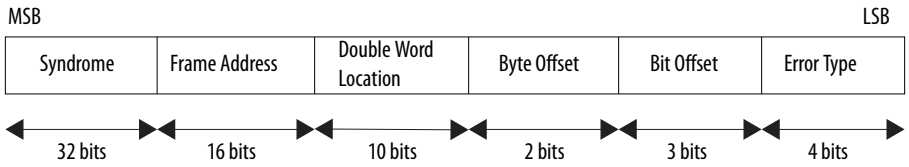


Table 9-6: Error Type in EMR

The following table lists the possible error types reported in the error type field in the EMR.

Error Type				Description
Bit 3	Bit 2	Bit 1	Bit 0	
0	0	0	0	No CRC error.
0	0	0	1	Location of a single-bit error is identified.
0	0	1	0	Location of a double-adjacent error is identified.
1	1	1	1	Error types other than single-bit and double-adjacent errors.

Table 9-7: JTAG Fault Injection Register Map

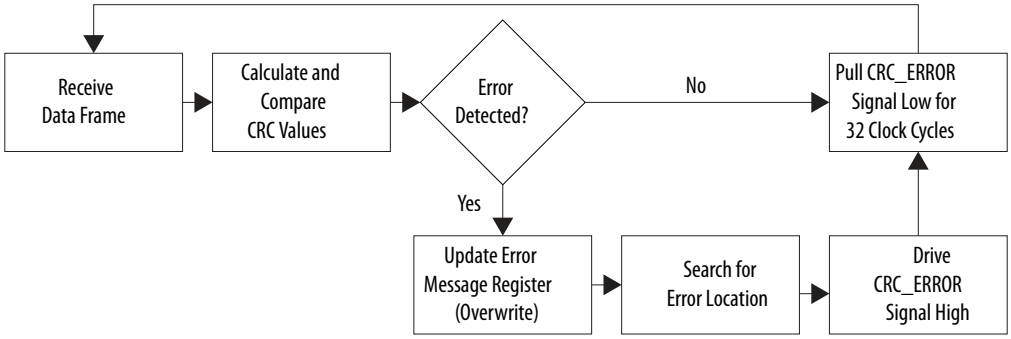
Field Name	Bit Range	Description
Error Byte Value	31:0	Contains the location of the bit error that corresponds to the error injection type to this field.
Byte Location	41:32	Contains the location of the injected error in the first data frame.

Field Name	Bit Range				Description
	45:42				Specifies the following error types.
	Bit 45	Bit 44	Bit 43	Bit 42	
Error Type	0	0	0	0	No error
	0	0	0	1	Single-bit error
	0	0	1	0	Double adjacent error

Error Detection Process

When enabled, the user mode error detection process activates automatically when the FPGA enters user mode. The process continues to run until the device is reset even when an error is detected in the current frame.

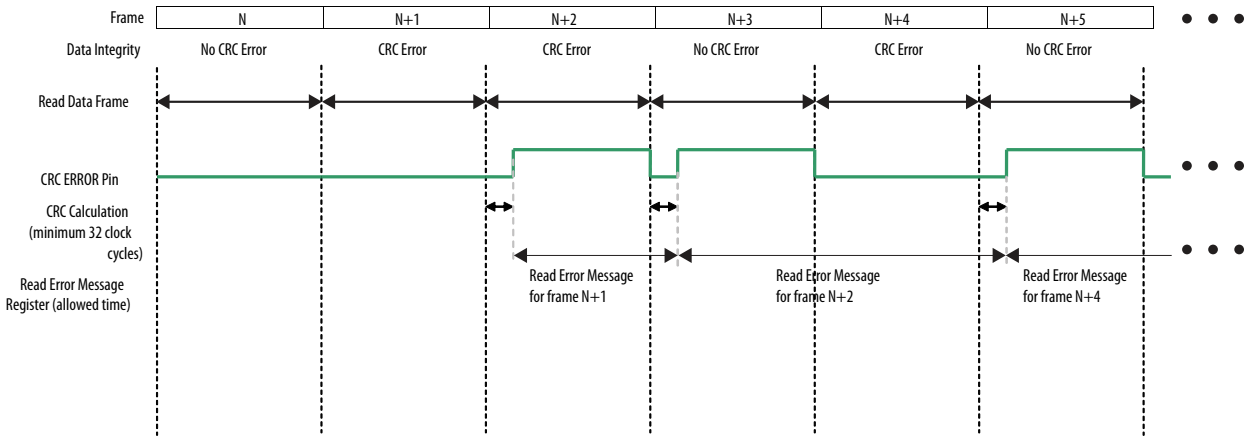
Figure 9-5: Error Detection Process Flow in User Mode



Timing

The CRC_ERROR pin is always driven low during CRC calculation for a minimum of 32 clock cycles. When an error occurs, the pin is driven high once the EMR is updated or 32 clock cycles have lapsed, whichever comes last. Therefore, you can start retrieving the contents of the EMR at the rising edge of the CRC_ERROR pin. The pin stays high until the current frame is read and then driven low again for a minimum of 32 clock cycles. To ensure information integrity, complete the read operation within one frame of the CRC verification. The following diagram shows the timing of these events.

Figure 9-6: Timing Requirements



Retrieving Error Information

You can retrieve the error information via the core interface or the JTAG interface using the `SHIFT_EDERROR_REG JTAG` instruction.

Recovering from CRC Errors

The system that hosts the FPGA must control device reconfiguration. To recover from a CRC error, drive the `nCONFIG` signal low. The system waits for a safe time before reconfiguring the device. When reconfiguration completes successfully, the FPGA operates as intended.

Related Information

- **Error Detection Frequency** on page 9-3
  Provides more information about the minimum and maximum error detection frequencies.
- **Minimum EMR Update Interval** on page 9-3
  Provides more information about the duration of each Stratix Vdevice.
- **Test Methodology of Error Detection and Recovery using CRC in Altera FPGA Devices**
  Provides more information about how to retrieve the error information.

Testing the Error Detection Block

You can inject errors into the configuration data to test the error detection block. This error injection methodology provides design verification and system fault tolerance characterization.

Testing via the JTAG Interface

You can intentionally inject single or double-adjacent errors into the configuration data using the `EDERROR_INJECT` JTAG instruction.

Table 9-8: EDERROR_INJECT instruction

JTAG Instruction	Instruction Code	Description
EDERROR_INJECT	00 0001 0101	Use this instruction to inject errors into the configuration data. This instruction controls the JTAG fault injection register, which contains the error you want to inject into the bitstream.

You can only inject errors into the first frame of the configuration data. However, you can monitor the error information at any time. Altera recommends that you reconfigure the FPGA after the test completes.

Automating the Testing Process

You can automate the testing process by creating a Jam™ file (**.jam**). Using this file, you can verify the CRC functionality in-system and on-the-fly without reconfiguring the device. You can then switch to the CRC circuitry to check for real errors caused by an SEU.

Related Information

Test Methodology of Error Detection and Recovery using CRC in Altera FPGA Devices
Provides more information about how to test the error detection block.

Document Revision History

Date	Version	Changes
January 2015	2015.01.23	Updated the description in the CRC Calculation Time section.
June 2014	2014.06.30	Updated the CRC Calculation Time section.
January 2014	2014.01.10	• Updated the CRC Calculation Time section to include a formula to calculate the minimum and maximum time. • Updated the maximum error detection frequency. • Removed preliminary and finalized the values for the Minimum EMR Update Interval and CRC Calculation Time.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Moved all links to the Related Information section of respective topics for easy reference.
December 2012	2012.12.28	• Updated the valid values of n in the error detection frequency equation. • Updated the width of the JTAG fault injection and fault injection registers.
June 2012	2.0	Minor text edits.
February 2012	1.4	Updated Table 10–9 and Table 10–10.

Date	Version	Changes
November 2011	1.3	• Chapter moved to Volume 2. • Updated Table 10–9 and Table 10–10. • Minor text edits.
May 2011	1.2	• Chapter moved to Volume 2. • Updated Table 10–9 and Table 10–10. • Minor text edits.
December 2010	1.1	No change.
July 2010	1.0	Initial release.

JTAG Boundary-Scan Testing in Stratix V Devices 10

2015.01.23

SV51012

This chapter describes the boundary-scan test (BST) features in Stratix V devices.

Related Information

- **JTAG Configuration** on page 8-28
  Provides more information about JTAG configuration.
- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the *Stratix V Device Handbook* chapters.

BST Operation Control

Stratix V devices support IEEE Std. 1149.1 and IEEE Std. 1149.6. The IEEE Std. 1149.6 is only supported on the high-speed serial interface (HSSI) transceivers in Stratix V devices. IEEE Std. 1149.6 enables board-level connectivity checking between transmitters and receivers that are AC coupled (connected with a capacitor in series between the source and destination).

IDCODE

The `IDCODE` is unique for each Stratix V device. Use this code to identify the devices in a JTAG chain.

Table 10-1: IDCODE Information for Stratix V Devices

Family	Member Code	IDCODE (32 Bits)			
		Version (4 Bits)	Part Number (16 Bits)	Manufacture Identity (11 Bits)	LSB (1 Bit)
Stratix V GX	A3 (17)	0000	0010 1001 0100 0111	000 0110 1110	1
	A3 (18)	0000	0010 1001 0010 0001	000 0110 1110	1
	A4	0000	0010 1001 0010 0111	000 0110 1110	1
	A5	0000	0010 1001 0001 0011	000 0110 1110	1
	A7	0000	0010 1001 0000 0011	000 0110 1110	1
	A9	0000	0010 1001 0100 0101	000 0110 1110	1
	AB	0000	0010 1001 0010 0101	000 0110 1110	1
	B5	0000	0010 1001 0001 0010	000 0110 1110	1
	B6	0000	0010 1001 0000 0010	000 0110 1110	1
	B9	0000	0010 1001 0001 0101	000 0110 1110	1
	BB	0000	0010 1001 0000 0101	000 0110 1110	1
Stratix V GT	C5	0000	0010 1001 0010 0011	000 0110 1110	1
	C7	0000	0010 1001 0100 0011	000 0110 1110	1

(17) The IDCODE is applicable for KF35 and KF40 packages only.
(18) The IDCODE is applicable for EH29 and HF35 packages only.

Family	Member Code	IDCODE (32 Bits)			
		Version (4 Bits)	Part Number (16 Bits)	Manufacture Identity (11 Bits)	LSB (1 Bit)
Stratix V GS	D3	0000	0010 1001 0001 0001	000 0110 1110	1
	D4 (19)	0000	0010 1001 0000 0001	000 0110 1110	1
	D4 (20)	0000	0010 1001 0001 0111	000 0110 1110	1
	D5	0000	0010 1001 0000 0111	000 0110 1110	1
	D6	0000	0010 1001 0001 0100	000 0110 1110	1
	D8	0000	0010 1001 0000 0100	000 0110 1110	1
Stratix V E	E9	0000	0010 1001 1001 0101	000 0110 1110	1
	EB	0000	0010 1001 1000 0101	000 0110 1110	1

Supported JTAG Instruction

Table 10-2: JTAG Instructions Supported by Stratix V Devices

JTAG Instruction	Instruction Code	Description
SAMPLE/PRELOAD	00 0000 0101	• Allows you to capture and examine a snapshot of signals at the device pins during normal device operation and permits an initial data pattern to be an output at the device pins. • Use this instruction to preload the test data into the update registers before loading the EXTEST instruction. • Used by the SignalTap™ II Embedded Logic Analyzer.

(19) The IDCODE is applicable for EH29 and HF35 packages only.
(20) The IDCODE is applicable for KF40 package only.

JTAG Instruction	Instruction Code	Description
EXTEST	00 0000 1111	• Allows you to test the external circuit and board-level interconnects by forcing a test pattern at the output pins, and capturing the test results at the input pins. Forcing known logic high and low levels on output pins allows you to detect opens and shorts at the pins of any device in the scan chain. • The high-impedance state of EXTEST is overridden by bus hold and weak pull-up resistor features.
BYPASS	11 1111 1111	Places the 1-bit bypass register between the TDI and TDO pins. During normal device operation, the 1-bit bypass register allows the BST data to pass synchronously through the selected devices to adjacent devices.
USERCODE	00 0000 0111	• Examines the user electronic signature (UES) within the devices along a JTAG chain. • Selects the 32-bit USERCODE register and places it between the TDI and TDO pins to allow serial shifting of USERCODE out of TDO. • The UES value is set to default value before configuration and is only user-defined after the device is configured.
IDCODE	00 0000 0110	• Identifies the devices in a JTAG chain. If you select IDCODE, the device identification register is loaded with the 32-bit vendor-defined identification code. • Selects the IDCODE register and places it between the TDI and TDO pins to allow serial shifting of IDCODE out of TDO. • IDCODE is the default instruction at power up and in the TAP RESET state. Without loading any instructions, you can go to the SHIFT_DR state and shift out the JTAG device ID.

JTAG Instruction	Instruction Code	Description
HIGHZ	00 0000 1011	• Sets all user I/O pins to an inactive drive state. • Places the 1-bit bypass register between the TDI and TDO pins. During normal operation, the 1-bit bypass register allows the BST data to pass synchronously through the selected devices to adjacent devices while tri-stating all I/O pins until a new JTAG instruction is executed. • If you are testing the device after configuration, the programmable weak pull-up resistor or the bus hold feature overrides the HIGHZ value at the pin.
CLAMP	00 0000 1010	• Places the 1-bit bypass register between the TDI and TDO pins. During normal operation, the 1-bit bypass register allows the BST data to pass synchronously through the selected devices to adjacent devices while holding the I/O pins to a state defined by the data in the boundary-scan register. • If you are testing the device after configuration, the programmable weak pull-up resistor or the bus hold feature overrides the CLAMP value at the pin. The CLAMP value is the value stored in the update register of the boundary-scan cell (BSC).
PULSE_NCONFIG	00 0000 0001	Emulates pulsing the nCONFIG pin low to trigger reconfiguration even though the physical pin is not affected.
CONFIG_IO	00 0000 1101	Allows I/O reconfiguration (after or during reconfigurations) through the JTAG ports using I/O configuration shift register (IOCSR) for JTAG testing. You can issue the CONFIG_IO instruction only after the nSTATUS pin goes high.

JTAG Instruction	Instruction Code	Description
LOCK	01 1111 0000	Put the device in JTAG secure mode. In this mode, only BYPASS, SAMPLE/PRELOAD, EXTEST, IDCODE, SHIFT_EDERROR_REG, and UNLOCK instructions are supported. This instruction can only be accessed through JTAG core access in user mode. It cannot be accessed through external JTAG pins in test or user mode.
UNLOCK	11 0011 0001	Release the device from the JTAG secure mode to enable access to all other JTAG instructions. This instruction can only be accessed through JTAG core access in user mode. It cannot be accessed through external JTAG pins in test or user mode.
KEY_CLR_VREG	00 0010 1001	Clears the volatile key.
KEY_VERIFY	00 0001 0011	Verifies the non-volatile key has been cleared.
EXTEST_PULSE	00 1000 1111	Enables board-level connectivity checking between the transmitters and receivers that are AC coupled by generating three output transitions: • Driver drives data on the falling edge of TCK in the UPDATE_IR/DR state. • Driver drives inverted data on the falling edge of TCK after entering the RUN_TEST/IDLE state. • Driver drives data on the falling edge of TCK after leaving the RUN_TEST/IDLE state. The EXTEST_PULSE JTAG instruction is only supported in user mode for Stratix V devices.
EXTEST_TRAIN	00 0100 1111	Behaves the same as the EXTEST_PULSE instruction except that the output continues to toggle on the TCK falling edge as long as the TAP controller is in the RUN_TEST/IDLE state. The EXTEST_TRAIN JTAG instruction is only supported in user mode for Stratix V devices.

Note: If the device is in a reset state and the nCONFIG or nSTATUS signal is low, the device IDCODE might not be read correctly. To read the device IDCODE correctly, you must issue the IDCODE JTAG instruction only when the nCONFIG and nSTATUS signals are high.

Note: If you use DC coupling on the HSSI signals, execute the EXTEST instruction. If you use AC coupling on the HSSI signals, execute the EXTEST_PULSE instruction. AC-coupled and DC-coupled HSSI are only supported in post-configuration mode.

Related Information

JTAG Secure Mode on page 8-40
Provides more information about PULSE_NCONFIG, CONFIG_IO, LOCK, and UNLOCK JTAG instructions.

JTAG Secure Mode

If you enable the tamper-protection bit, the Stratix V device is in JTAG secure mode after power up. In the JTAG secure mode, the JTAG pins support only the BYPASS, SAMPLE/PRELOAD, EXTEST, IDCODE, SHIFT_EDERROR_REG, and UNLOCK instructions. Issue the UNLOCK JTAG instruction to enable support for other JTAG instructions.

JTAG Private Instruction

Caution: Never invoke the following instruction codes. These instructions can damage and render the device unusable:

- 1100010000
- 0011001001
- 1100010011
- 1100010111
- 0111100000
- 1110110011

I/O Voltage for JTAG Operation

A Stratix V device operating in BST mode uses four required JTAG pins—TDI, TDO, TMS, TCK, and one optional pin, TRST.

The TCK pin has an internal weak pull-down resistor, while the TDI and TMS pins have internal weak pull-up resistors. The 3.0- or 2.5-V V_{CCPD} supply of I/O bank 3A powers the TDO, TDI, TMS, and TCK pins. All user I/O pins are tri-stated during JTAG configuration.

The JTAG chain supports several different devices. Use the supported TDO and TDI voltage combinations listed in the following table if the JTAG chain contains devices that have different V_{CCIO} levels. The output voltage level of the TDO pin must meet the specification of the TDI pin it drives.

Table 10-3: Supported TDO and TDI Voltage Combinations

The TDO output buffer for V_{CCPD} of 3.0 V meets V_{OH} (MIN) of 2.4 V, and the TDO output buffer for V_{CCPD} of 2.5 V meets V_{OH} (MIN) of 2.0 V.

Device	TDI Input Buffer Power (V)	Stratix V TDO V_{CCPD}	
		V_{CCPD} = 3.0 V	V_{CCPD} = 2.5 V
Stratix V	V_{CCPD} = 3.0 V	Yes	Yes
	V_{CCPD} = 2.5 V	Yes	Yes
Non-Stratix V(21)	V_{CC} = 3.3 V	Yes	Yes
	V_{CC} = 2.5 V	Yes	Yes
	V_{CC} = 1.8 V	Yes	Yes
	V_{CC} = 1.5 V	Yes	Yes

Performing BST

You can issue BYPASS, IDCODE, and SAMPLE JTAG instructions before, after, or during configuration without having to interrupt configuration.

To issue other JTAG instructions, follow these guidelines:

- To perform testing before configuration, hold the nCONFIG pin low.
- To perform BST during configuration, issue CONFIG_IO JTAG instruction to interrupt configuration. While configuration is interrupted, you can issue other JTAG instructions to perform BST. After BST is completed, issue the PULSE_CONFIG JTAG instruction or pulse nCONFIG low to reconfigure the device.

The chip-wide reset (DEV_CLRn) and chip-wide output enable (DEV_OE) pins on Stratix V devices do not affect JTAG boundary-scan or configuration operations. Toggling these pins does not disrupt BST operation (other than the expected BST behavior).

If you design a board for JTAG configuration of Stratix V devices, consider the connections for the dedicated configuration pins.

Related Information

- **Configuration, Design Security, and Remote System Upgrades in Stratix V Devices**
  Provides more information about JTAG configuration.
- **Stratix V Device Datasheet**
  Provides more information about JTAG configuration timing.

Enabling and Disabling IEEE Std. 1149.1 BST Circuitry

The IEEE Std. 1149.1 BST circuitry is enabled after the Stratix V device powers up.

(21) The input buffer must be tolerant to the TDO V_{CCPD} voltage.

To ensure that you do not inadvertently enable the IEEE Std. 1149.1 circuitry when it is not required, disable the circuitry permanently with pin connections as listed in the following table.

Table 10-4: Pin Connections to Permanently Disable the IEEE Std. 1149.1 Circuitry for Stratix V Devices

JTAG Pins[22]	Connection for Disabling
TMS	V_{CCPD} supply of Bank 3A
TCK	GND
TDI	V_{CCPD} supply of Bank 3A
TDO	Leave open

Guidelines for IEEE Std. 1149.1 Boundary-Scan Testing

Consider the following guidelines when you perform BST with IEEE Std. 1149.1 devices:

- If the "10..." pattern does not shift out of the instruction register through the TDO pin during the first clock cycle of the SHIFT_IR state, the TAP controller did not reach the proper state. To solve this problem, try one of the following procedures:
  - Verify that the TAP controller has reached the SHIFT_IR state correctly. To advance the TAP controller to the SHIFT_IR state, return to the RESET state and send the 01100 code to the TMS pin.
  - Check the connections to the VCC, GND, JTAG, and dedicated configuration pins on the device.
- Perform a SAMPLE/PRELOAD test cycle before the first EXTEST test cycle to ensure that known data is present at the device pins when you enter EXTEST mode. If the OEJ update register contains 0, the data in the OUTJ update register is driven out. The state must be known and correct to avoid contention with other devices in the system.
- Do not perform EXTEST testing during in-circuit reconfiguration because EXTEST is not supported during in-circuit reconfiguration. To perform testing, wait for the configuration to complete or issue the CONFIG_IO instruction to interrupt configuration.
- After configuration, you cannot test any pins in a differential pin pair. To perform BST after configuration, edit and redefine the BSC group that correspond to these differential pin pairs as an internal cell.

Related Information

IEEE 1149.6 BSDL Files

Provides more information about BSC group definitions.

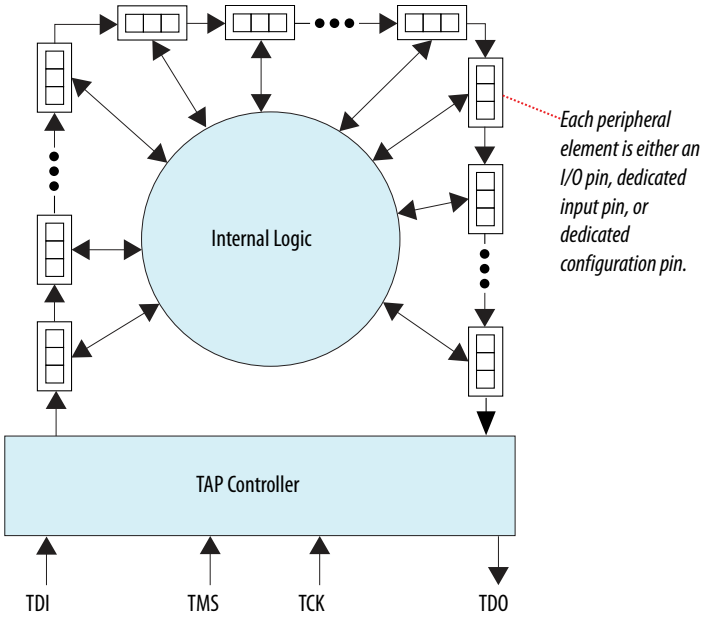
IEEE Std. 1149.1 Boundary-Scan Register

The boundary-scan register is a large serial shift register that uses the TDI pin as an input and the TDO pin as an output. The boundary-scan register consists of 3-bit peripheral elements that are associated with Stratix V I/O pins. You can use the boundary-scan register to test external pin connections or to capture internal data.

[22] The JTAG pins are dedicated. Software option is not available to disable JTAG in Stratix V devices.

Figure 10-1: Boundary-Scan Register

This figure shows how test data is serially shifted around the periphery of the IEEE Std. 1149.1 device.



Boundary-Scan Cells of a Stratix V Device I/O Pin

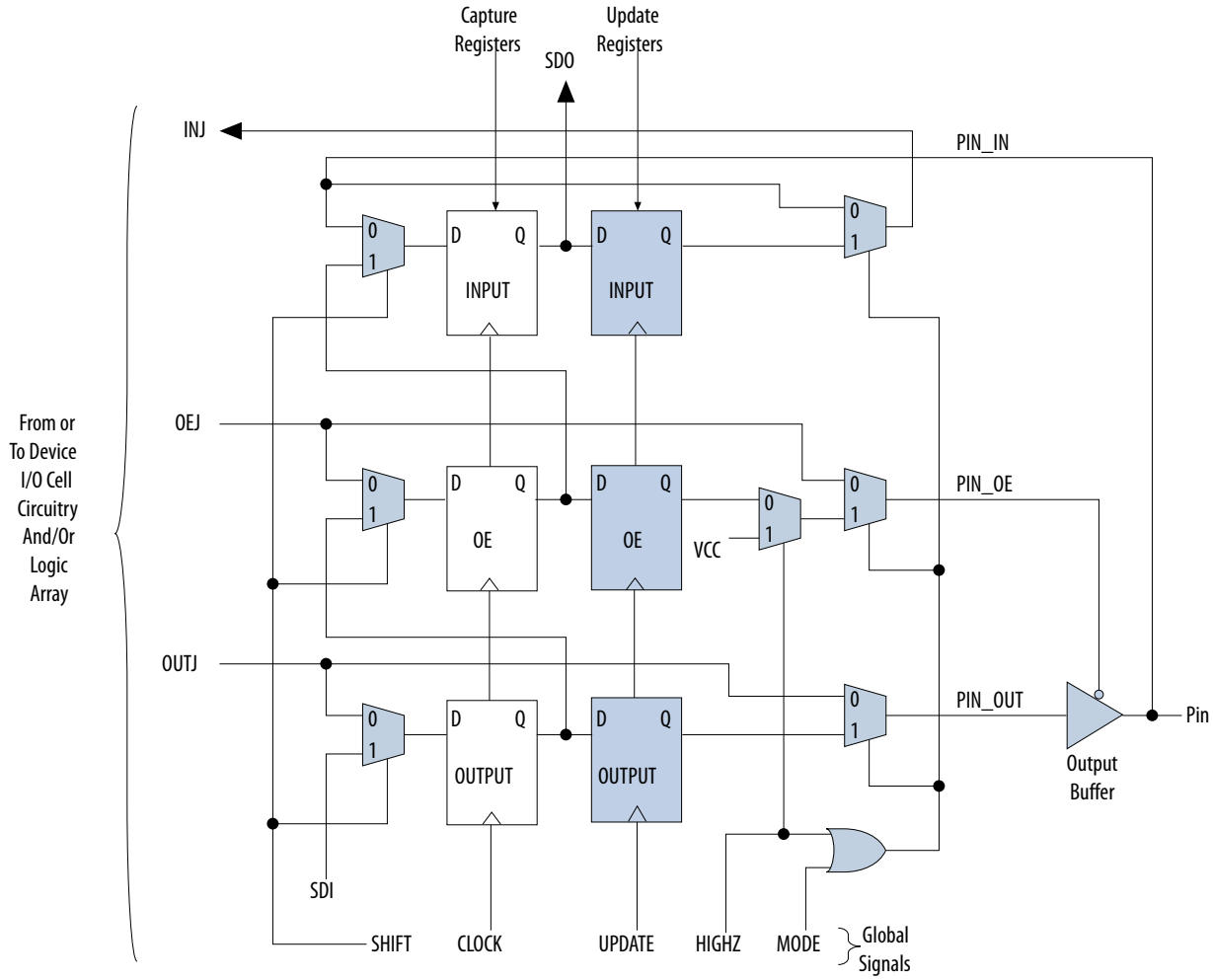
The Stratix V device 3-bit BSC consists of the following registers:

- Capture registers—Connect to internal device data through the `OUTJ`, `OEJ`, and `PIN_IN` signals.
- Update registers—Connect to external data through the `PIN_OUT` and `PIN_OE` signals.

The TAP controller generates the global control signals for the IEEE Std. 1149.1 BST registers (`shift`, `clock`, and `update`) internally. A decode of the instruction register generates the `MODE` signal.

The data signal path for the boundary-scan register runs from the serial data in (`SDI`) signal to the serial data out (`SDO`) signal. The scan register begins at the `TDI` pin and ends at the `TDO` pin of the device.

Figure 10-2: User I/O BSC with IEEE Std. 1149.1 BST Circuitry for Stratix V Devices



Note: TDI, TDO, TMS, and TCK pins, all VCC and GND pin types, and VREF pins do not have BSCs.

Table 10-5: Boundary-Scan Cell Descriptions for Stratix V Devices

This table lists the capture and update register capabilities of all BSCs within Stratix V devices.

Pin Type	Captures			Drives			Comments
	Output Capture Register	OE Capture Register	Input Capture Register	Output Update Register	OE Update Register	Input Update Register	
User I/O pins	OUTJ	OEJ	PIN_IN	PIN_OUT	PIN_OE	INJ	—
Dedicated clock input	0	1	PIN_IN	No Connect (N.C.)	N.C.	N.C.	PIN_IN drives to the clock network or logic array

Pin Type	Captures			Drives			Comments
	Output Capture Register	OE Capture Register	Input Capture Register	Output Update Register	OE Update Register	Input Update Register	
Dedicated input	0	1	PIN_IN	N.C.	N.C.	N.C.	PIN_IN drives to the control logic
Dedicated bidirectional (open drain) (23)	0	OEJ	PIN_IN	N.C.	N.C.	N.C.	PIN_IN drives to the configuration control
Dedicated bidirectional(24)	OUTJ	OEJ	PIN_IN	N.C.	N.C.	N.C.	PIN_IN drives to the configuration control and OUTJ drives to the output buffer
Dedicated output(25)	OUTJ	0	0	N.C.	N.C.	N.C.	OUTJ drives to the output buffer

IEEE Std. 1149.6 Boundary-Scan Register

The BSCs for HSSI transmitters (GXB_TX[p,n]) and receivers/input clock buffers (GXB_RX[p,n])/(REFCLK[p,n]) in Stratix V devices are different from the BSCs for the I/O pins.

(23) This includes the CONF_DONE and nSTATUS pins.
(24) This includes the DCLK pin.
(25) This includes the nCEO pin.

Figure 10-3: HSSI Transmitter BSC with IEEE Std. 1149.6 BST Circuitry for Stratix V Devices

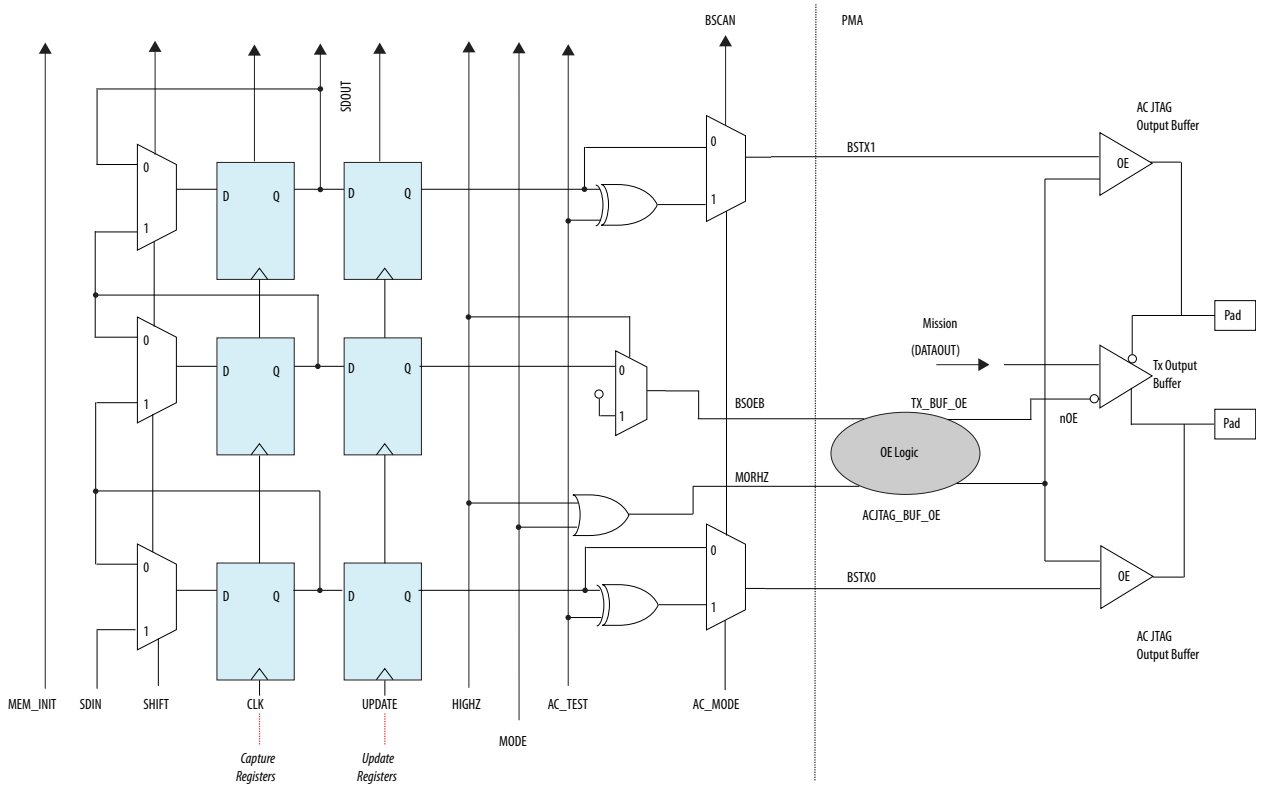
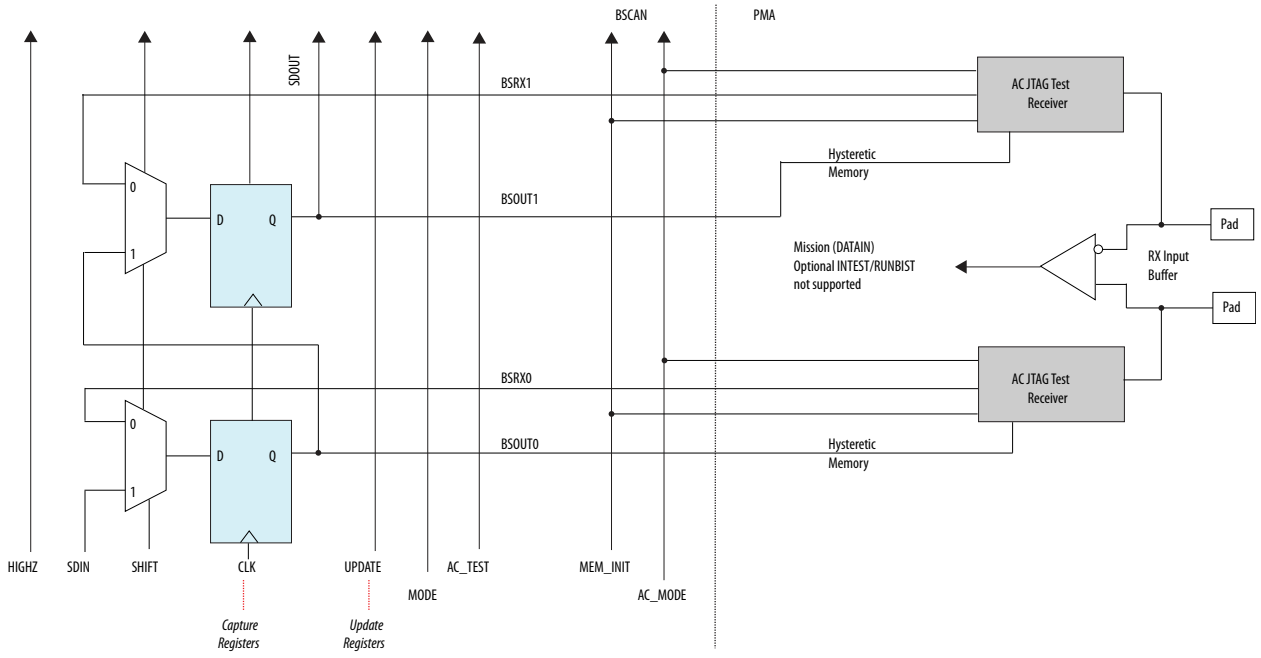


Figure 10-4: HSSI Receiver/Input Clock Buffer with IEEE Std. 1149.6 BST Circuitry for Stratix V Devices



Document Revision History

Date	Version	Changes
January 2014	2014.01.10	• Updated the Supported JTAG Instruction section. • Updated the KEY_CLR_VREG JTAG instruction.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated the description for EXTEST_TRAIN and EXTEST_PULSE JTAG instructions. • Moved all links to the Related Information section of respective topics for easy reference.
December 2012	2012.12.28	Reorganized content and updated template.
June 2012	1.5	Updated Table 11-1.
December 2011	1.4	Updated Table 11-2 to include KEY_CLR_VREG and KEY_VERIFY JTAG instructions.
November 2011	1.3	Updated Table 11-1 and Table 11-2.
May 2011	1.2	• Chapter moved to volume 2 for the 11.0 release. • Updated Table 11-1.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1 release.
July 2010	1.0	Initial release.

Power Management in Stratix V Devices 11

2015.01.23

SV51013

This chapter describes the programmable power technology, hot-socketing feature, power-on reset (POR) requirements, power-up sequencing recommendation, temperature sensing diode (TSD), and their implementation in Stratix V devices.

Related Information

- **Stratix V Device Handbook: Known Issues**
  Lists the planned updates to the Stratix V Device Handbook chapters.
- **PowerPlay Power Analysis**
  Provides more information about the Quartus®II PowerPlay Power Analyzer tool in volume 3 of the Quartus II Handbook.
- **Stratix V Device Datasheet**
  Provides more information about the recommended operating conditions of each power supply.
- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides detailed information about power supply pin connection guidelines and power regulator sharing.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides detailed information about power supply pin connection guidelines and power regulator sharing.
- **Board Design Resource Center**
  Provides detailed information about power supply design requirements.
- **PowerPlay Early Power Estimators (EPE) and Power Analyzer**
  Provides more information about the two supplies which make up the V_{CC} supply. They are V_{CCL} (core V_{CC}) and V_{CCP} (periphery V_{CC}). The sum of I_{CCL} and I_{CCP} equals to I_{CC}. I_{CCL} and I_{CCP} is found on the EPE report tab.
- **Stratix V Device Design Guidelines**
- **Stratix V GT Device Design Guidelines**

Power Consumption

The total power consumption of a Stratix V device consists of the following components:

- Static power—the power that the configured device consumes when powered up but no clocks are operating.
- Dynamic power— the additional power consumption of the device due to signal activity or toggling.

Dynamic Power Equation

Figure 11-1: Dynamic Power

The following equation shows how to calculate dynamic power where P is power, C is the load capacitance, and V is the supply voltage level.

$$P = \frac{1}{2} CV^2 \times frequency$$

The equation shows that power is design-dependent and is determined by the operating frequency of your design. Stratix V devices minimize static and dynamic power using advanced process optimizations. This technology allows Stratix V designs to meet specific performance requirements with the lowest possible power.

Programmable Power Technology

Stratix V devices offer the ability to configure portions of the core, called tiles, for high-speed or low-power mode of operation performed by the Quartus II software without user intervention. Setting a tile to high-speed or low-power mode is accomplished with on-chip circuitry and does not require extra power supplies brought into the Stratix V device. In a design compilation, the Quartus II software determines whether a tile should be in high-speed or low-power mode based on the timing constraints of the design.

Stratix V tiles consist of the following:

- Memory logic array block (MLAB)/ logic array block (LAB) pairs with routing to the pair
- MLAB/LAB pairs with routing to the pair and to adjacent digital signal processing (DSP)/ memory block routing
- TriMatrix memory blocks
- DSP blocks
- PCI Express® (PCIe®) hard IP
- Physical coding sublayer (PCS)

All blocks and routing associated with the tile share the same setting of either high-speed or low-power mode. By default, tiles that include DSP blocks or memory blocks are set to high-speed mode for optimum performance. Unused DSP blocks and memory blocks are set to low-power mode to minimize static power. Clock networks do not support programmable power technology.

With programmable power technology, faster speed grade FPGAs may require less power because there are fewer high-speed MLAB and LAB pairs, when compared with slower speed grade FPGAs. The slower speed grade device may have to use more high-speed MLAB and LAB pairs to meet performance requirements.

The Quartus II software sets unused device resources in the design to low-power mode to reduce the static power. It also sets the following resources to low-power mode when they are not used in the design:

- LABs and MLABs
- TriMatrix memory blocks
- DSP blocks

If a phase-locked loop (PLL) is instantiated in the design, you may assert the `areset` pin high to keep the PLL in low-power mode.

Altera recommends that you power down unused PCIe HIPs, per side, by connecting the PCIe HIP power to GND on the PCB for additional power savings. All of the HIPs on a side of the device must be unused to be powered down. For additional information refer to the pin connection guidelines.

Table 11-1: Programmable Power Capabilities for Stratix V Devices

This table lists the available Stratix V programmable power capabilities. Speed grade considerations can add to the permutations to give you flexibility in designing your system.

Feature	Programmable Power Technology
LAB	Yes
Routing	Yes
Memory Blocks	Fixed setting(26)
DSP Blocks	Fixed setting(26)
Clock Networks	No

Related Information

- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides more information about powering down PCIe HIPs.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides more information about powering down PCIe HIPs.

Temperature Sensing Diode

The Stratix V TSD uses the characteristics of a PN junction diode to determine die temperature. Knowing the junction temperature is crucial for thermal management. You can calculate junction temperature using ambient or case temperature, junction-to-ambient (ja) or junction-to-case (jc) thermal resistance, and device power consumption. Stratix V devices monitor its die temperature with the internal TSD with built-in analog-to-digital converter (ADC) circuitry or the external TSD with an external temperature sensor. This allows you to control the air flow to the device.

Internal Temperature Sensing Diode

You can use the Stratix V internal TSD in the following operations:

- Power-up mode—to read the die's temperature during configuration, enable the ALTTEMP_SENSE megafunction in your design.
- User mode—to read the die's temperature during user mode, assert the `clken` signal to the internal TSD circuitry.

Note: To reduce power consumption, disable the Stratix V internal TSD when you are not using it.

(26) Tiles with DSP blocks and memory blocks that are used in the design are always set to high-speed mode. By default, unused DSP blocks and memory blocks are set to low-power mode.

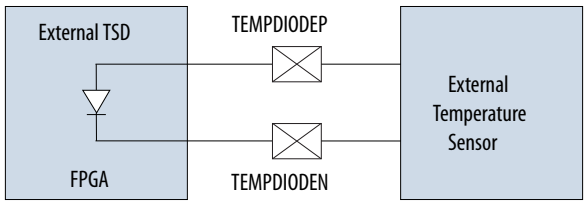
Related Information

- **Temperature Sensor (ALTTEMP_SENSE) Megafunction User Guide**
  Provides more information about using the ALTTEMP_SENSE megafunction.
- **Stratix V Device Datasheet**
  Provides more information about the Stratix V internal TSD specification.

External Temperature Sensing Diode

The Stratix V external TSD requires two pins for voltage reference. The following figure shows how to connect the external TSD with an external temperature sensor device, allowing external sensing of the Stratix V die temperature. For example, you can connect external temperature sensing devices, such as MAX1619, MAX1617A, MAX6627, and ADT7411 to the two external TSD pins for Stratix V device die temperature reading.

Figure 11-2: TSD External Pin Connections



The TSD is a very sensitive circuit that can be influenced by noise coupled from other traces on the board or within the device package itself, depending on your device usage. The interfacing signal from the Stratix V device to the external temperature sensor is based on millivolts (mV) of difference, as seen at the external TSD pins. Switching the I/O near the TSD pins can affect the temperature reading. Altera recommends taking temperature readings during periods of inactivity in the device or use the internal TSD with built-in ADC circuitry.

The following are board connection guidelines for the TSD external pin connections:

- The maximum trace lengths for the $TEMPDIODE_P$/$TEMPDIODE_N$ traces must be less than eight inches.
- Route both traces in parallel and place them close to each other with grounded guard tracks on each side.
- Altera recommends 10-mils width and space for both traces.
- Route traces through a minimum number of vias and crossunders to minimize the thermocouple effects.
- Ensure that the number of vias are the same on both traces.
- Ensure both traces are approximately the same length.
- Avoid coupling with toggling signals (for example, clocks and I/O) by having the GND plane between the diode traces and the high frequency signals.
- For high-frequency noise filtering, place an external capacitor (close to the external chip) between the $TEMPDIODE_P$/$TEMPDIODE_N$ trace. For Maxim devices, use an external capacitor between 2200 pF to 3300 pF.

- Place a 0.1 uF bypass capacitor close to the external device.
- You can use the internal TSD with built-in ADC circuitry and external TSD at the same time.
- If you only use internal ADC circuitry, the external TSD pins ($TEMPDIODE_P$/$TEMPDIODE_N$) can be connected to GND because the external TSD pins are not used.

For details about device specification and connection guidelines, refer to the external temperature sensor device datasheet from the device manufacturer.

Related Information

- **Stratix V Device Datasheet**
  Provides details about the external TSD specification.
- **Stratix V E, GS, and GX Device Family Pin Connection Guidelines**
  Provides details about the $TEMPDIODE_P$/$TEMPDIODE_N$ pin connection when you are not using an external TSD.
- **Stratix V GT Device Family Pin Connection Guidelines**
  Provides details about the $TEMPDIODE_P$/$TEMPDIODE_N$ pin connection when you are not using an external TSD.

Hot-Socketing Feature

Stratix V devices support hot socketing—also known as hot plug-in or hot swap.

The hot-socketing circuitry monitors the V_{CCIO}, V_{CCPD}, and V_{CC} power supplies and all V_{CCIO} and V_{CCPD} banks.

When powering up or powering down these power supplies, refer to the Power-Up Sequence section of this handbook.

During the hot-socketing operation, the I/O pin capacitance is less than 15 pF and the clock pin capacitance is less than 20 pF.

The hot-socketing capability removes some of the difficulty that designers face when using the Stratix V devices on PCBs that contain a mixture of devices with different voltage requirements.

The hot-socketing capability in Stratix V devices provides the following advantages:

- You can drive signals into the I/O, dedicated input, and dedicated clock pins before or during power up or power down without damaging the device. External input signals to the I/O pins of the unpowered device will not power the power supplies through internal paths within the device.
- The output buffers are tri-stated during system power up or power down. Because the Stratix V device does not drive signals out before or during power up, the device does not affect the other operating buses.
- You can insert or remove a Stratix V device from a powered-up system board without damaging or interfering with the system board's operation. This capability allows you to avoid sinking current through the device signal pins to the device power supply, which can create a direct connection to GND that causes power supply failures.
- During hot socketing, Stratix V devices are immune to latch up that can occur when a device is hot-socketed into an active system.

Altera uses GND as a reference for hot-socketing and I/O buffer circuitry designs. To ensure proper operation, connect GND between boards before connecting the power supplies. This prevents GND on

your board from being pulled up inadvertently by a path to power through other components on your board. A pulled up GND could otherwise cause an out-of-specification I/O voltage or over current condition in the Altera device.

Related Information

- **Power-Up Sequence** on page 11-7
- **Stratix V Device Datasheet**
  Provides details about the Stratix V hot-socketing specifications.

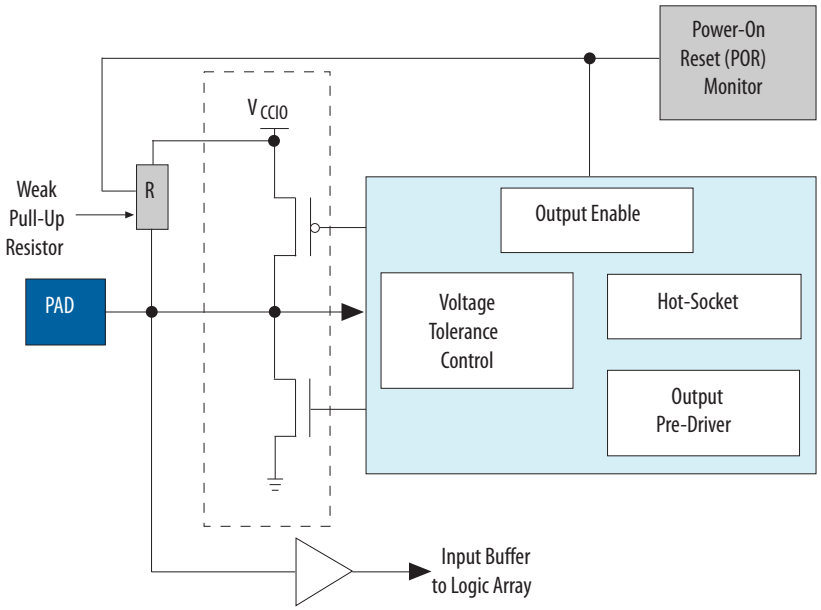
Hot-Socketing Implementation

The hot-socketing feature tri-state the output buffer during power up and power down of the power supplies. When these power supplies are below the threshold voltage, the hot-socketing circuitry generates an internal `HOTSCKT` signal.

Hot-socketing circuitry prevents excess I/O leakage during power up. When the voltage ramps up very slowly, I/O leakage is still relatively low, even after the release of the POR signal and configuration is complete.

Note: The output buffer cannot flip from the state set by the hot-socketing circuitry at very low voltage. To allow the `CONF_DONE` and `nSTATUS` pins to operate during configuration, the hot-socketing feature is not applied to these configuration pins. Therefore, these pins will drive out during power up and power down.

Figure 11-3: Hot-Socketing Circuitry for Stratix V Devices



The POR circuitry monitors the voltage level of the power supplies and keeps the I/O pins tri-stated until the device is in user mode. The weak pull-up resistor (R) in the Stratix V input/output element (IOE) is enabled during configuration download to keep the I/O pins from floating.

The 3.0-V tolerance control circuit allows the I/O pins to be driven by 3.0 V before the power supplies are powered and prevents the I/O pins from driving out before the device enters user mode.

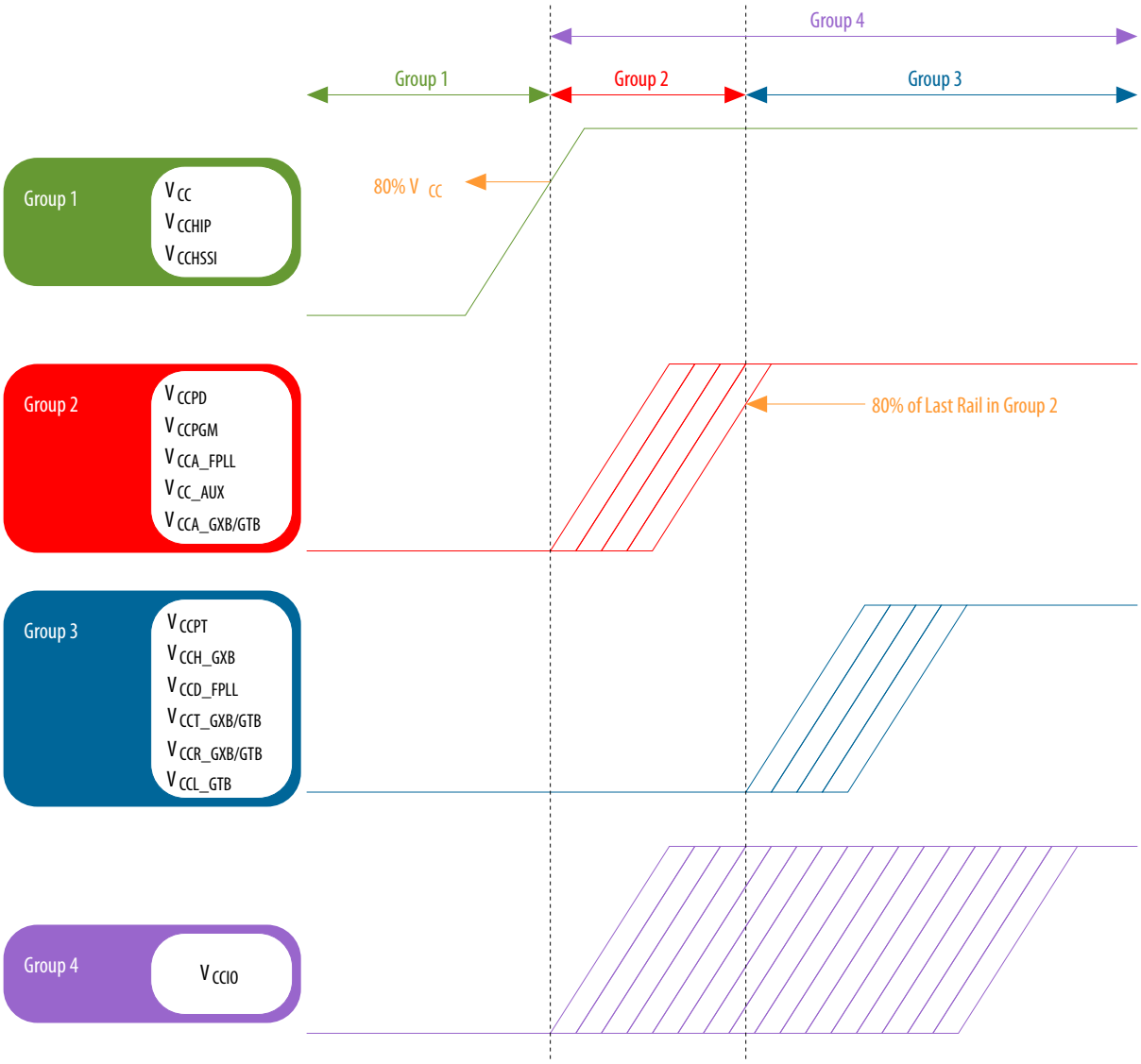
Note: For the V_{CC_AUX} power supply, POR only monitors one of the VCC_AUX pins. You must connect all the VCC_AUX pins.

Power-Up Sequence

The Stratix V devices require a power-up sequence as shown in the following figure to prevent excessive inrush current. This power-up sequence is divided into four power groups. Group 1 contains the first power rails to ramp. The V_{CC}, V_{CCHIP}, and V_{CCHSSI} power rails in this group must ramp to a minimum of 80% of their full rail before any other power rails may start. Group 1 power rails can continue to ramp to full rail. The power rails in Group 2 and Group 4 can start to ramp in any order after Group 1 has reached its minimum 80% threshold. When the last power rail in Group 2 reaches 80% of its full rail, the remaining power rails in Group 3 may start their ramp. During this time, Group 2 power rails may continue to ramp to full rail. Power rails in Group 3 may ramp in any order. All power rails must ramp monotonically. The complete power-up sequence must meet either the standard or fast POR delay time, depending on the POR delay setting that is used.

Figure 11-4: Power-Up Sequence Requirement for Stratix V Devices

Power up V_{CCBAT} at any time. If V_{CC}, V_{CCR_GXB}, and V_{CCT_GXB} have the same voltage level, they can be powered by the same regulator in Group 1 and ramp simultaneously.



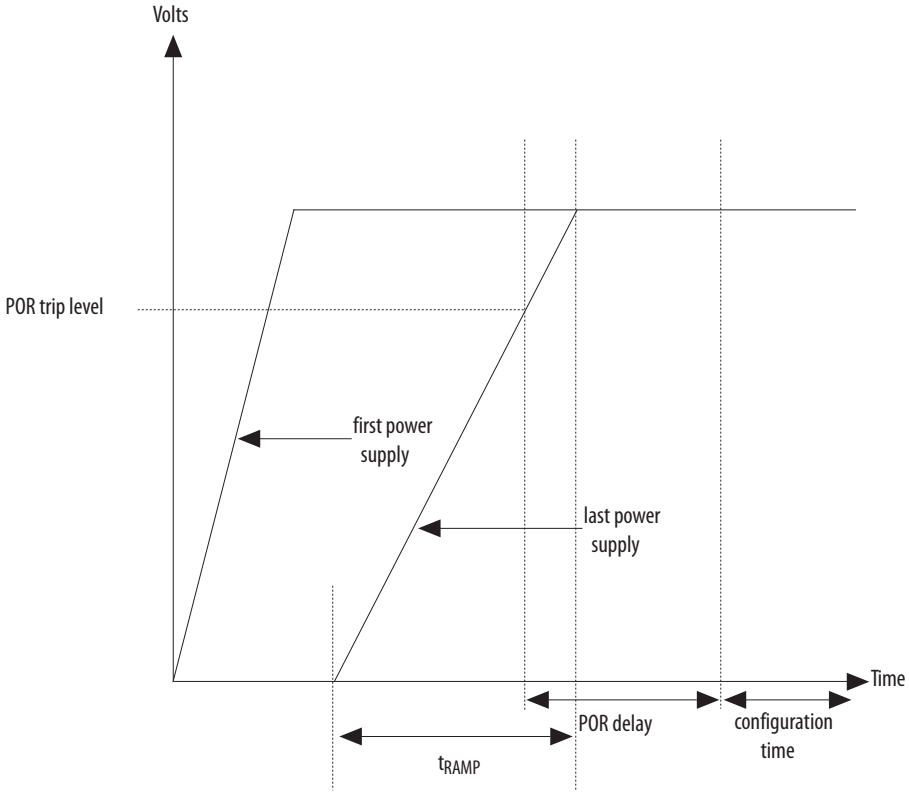
Stratix V devices may power down all power rails simultaneously. However, all rails must reach 0 V within 100 ms from the start of power-down.

Power-On Reset Circuitry

The POR circuitry keeps the Stratix V device in the reset state until the power supply outputs are within the recommended operating range.

A POR event occurs when you power up the Stratix V device until the power supplies reach the recommended operating range within the maximum power supply ramp time, t_{RAMP}. If t_{RAMP} is not met, the Stratix V device I/O pins and programming registers remain tri-stated, during which device configuration could fail.

Figure 11-5: Relationship Between t_{RAMP} and POR Delay

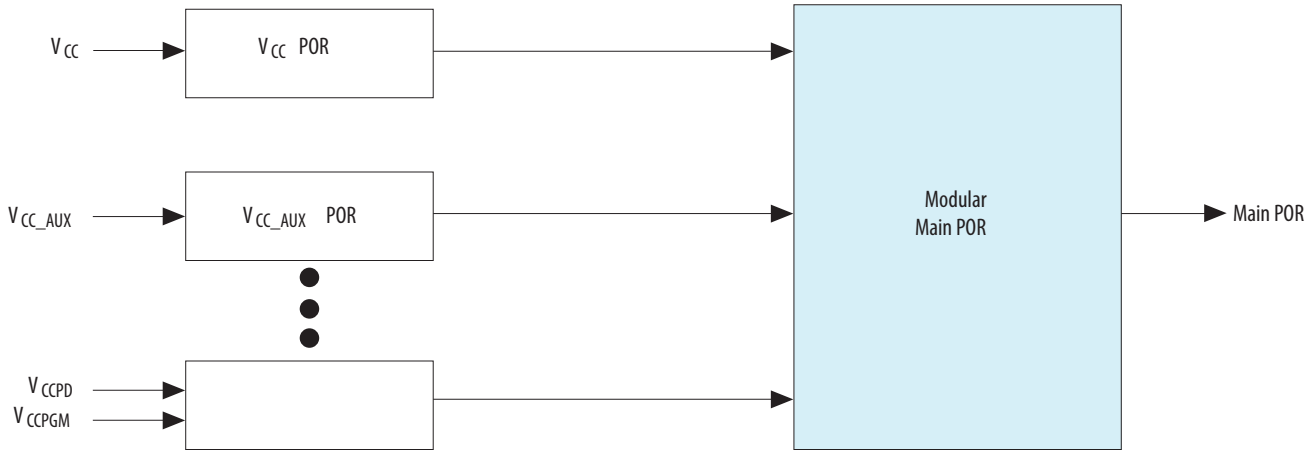


The Stratix V POR circuitry uses an individual detecting circuitry to monitor each of the configuration-related power supplies independently. The main POR circuitry is gated by the outputs of all the individual detectors. The main POR signal is asserted when the power starts to ramp up. This signal is released after the last ramp-up power reaches the POR trip level during power up.

In user mode, the main POR signal is asserted when any of the monitored power goes below its POR trip level. Asserting the POR signal forces the device into the reset state.

The POR circuitry checks the functionality of the I/O level shifters powered by the V_{CCPD} and V_{CCPGM} power supplies during power-up mode. The main POR circuitry waits for all the individual POR circuitries to release the POR signal before allowing the control block to start programming the device.

Figure 11-6: Simplified POR Diagram for Stratix V Devices



Related Information

Stratix V Device Datasheet
Provides more information about the POR delay specification and t_{RAMP}.

Power Supplies Monitored and Not Monitored by the POR Circuitry

Table 11-2: Power Supplies Monitored and Not Monitored by the Stratix V POR Circuitry

Power Supplies Monitored	Power Supplies Not Monitored
• V_{CC_AUX} • V_{CCBAT} • V_{CC} • V_{CCPT} • V_{CCPD} • V_{CCPGM}	• V_{CCT_GXB} • V_{CCH_GXB} • V_{CCR_GXB} • V_{CCA_GXB} • V_{CCA_FPLL} • V_{CCD_FPLL} • V_{CCIO} • V_{CCHIP}

Note: For the device to exit POR, you must power the V_{CCBAT} power supply even if you do not use the volatile key.

Related Information

Configuration, Design Security, and Remote System Upgrades in Stratix V Devices
Provides more information about the `MSEL` pin settings for each POR delay.

Document Revision History

Date	Version	Changes
January 2015	2015.01.23	Added links to the Stratix V Design Guidelines and Stratix V GT Design Guidelines.

Date	Version	Changes
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Moved all links to the Related Information section of respective topics for easy reference. • Added 'There are two supplies which make up the V_{CC} supply. They are V_{CCL} (core V_{CC}) and V_{CCP} (periphery V_{CC}). The sum of I_{CCL} and I_{CCP} equals to I_{CC}. You can refer to the Stratix V PowerPlay Early Power Estimators (EPE) and Power Analyzer for I_{CCL} and I_{CCP} on the EPE report tab.' to 'For detailed information about power supply design requirements, refer to the Board Design Resource Center page.' • Updated dynamic power in Power Consumption for improve clarity. • Added description on powering down unused PCIe HIPS in Programmable Power Technology • Updated Hot-Socketing Feature with ' When powering up these power supplies, you must follow the required power-up sequence as shown in the Power-Up Sequence section of this handbook.'
December 2012	2012.12.28	• Consolidated content from the Hot Socketing and Power-On Reset in Stratix V Devices chapter. • Reorganized content and updated template.
June 2012	1.3	Minor text edits.
May 2011	1.2	Chapter moved to volume 2 for the Quartus II software 11.0 release.
December 2010	1.1	No changes to the content of this chapter for the Quartus II software 10.1 release.
July 2010	1.0	Initial release.

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View 5SGXMA9N2F45C2N on WIN SOURCE
- Altera Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management