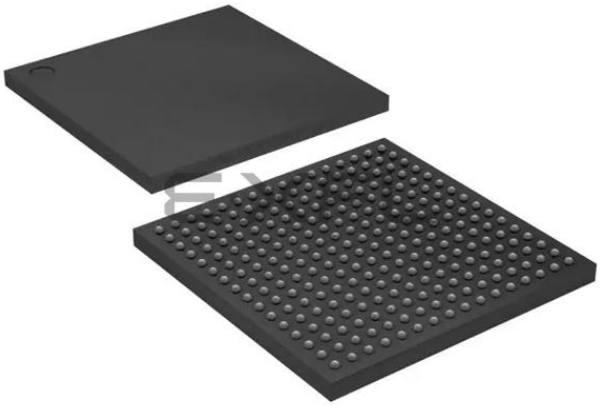


Welcome to E-XFL.COM

Understanding Embedded - FPGAs (Field Programmable Gate Array)

Embedded - FPGAs, or Field Programmable Gate Arrays, are advanced integrated circuits that offer unparalleled flexibility and performance for digital systems. Unlike traditional fixed-function logic devices, FPGAs can be programmed and reprogrammed to execute a wide array of logical operations, enabling customized functionality tailored to specific applications. This reprogrammability allows developers to iterate designs quickly and implement complex functions without the need for custom hardware.

Applications of Embedded - FPGAs

The versatility of Embedded - FPGAs makes them indispensable in numerous fields. In telecommunications.

Details	
Product Status	Active
Number of LABs/CLBs	963
Number of Logic Elements/Cells	15408
Total RAM Bits	516096
Number of I/O	165
Number of Gates	-
Voltage - Supply	1.15V ~ 1.25V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 100°C (TJ)
Package / Case	256-LFBGA
Supplier Device Package	256-UBGA (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/intel/ep4ce15u14i7n

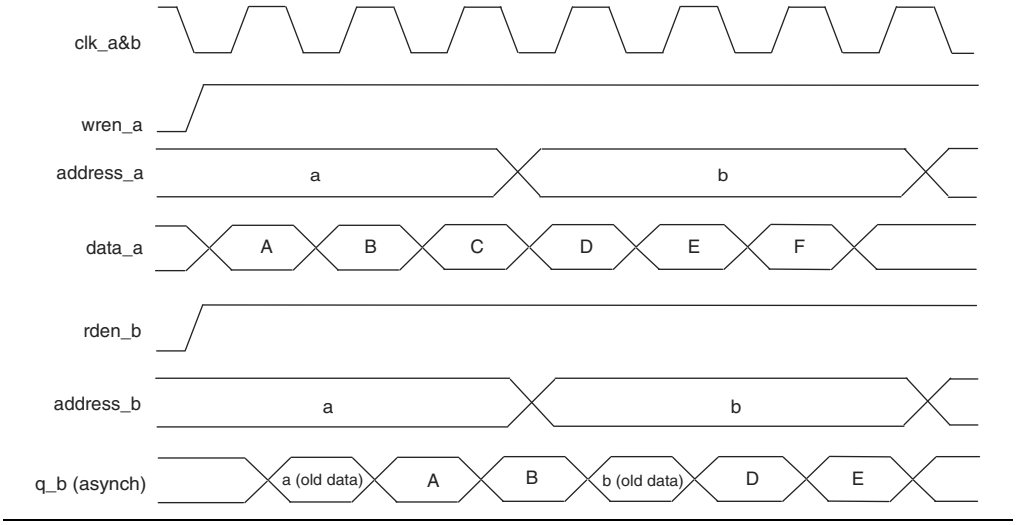
Contents

In this mode, you also have two output choices: **Old Data** mode or **Don't Care** mode. In **Old Data** mode, a read-during-write operation to different ports causes the RAM outputs to reflect the old data at that address location. In **Don't Care** mode, the same operation results in a "Don't Care" or unknown value on the RAM outputs.

For more information about how to implement the desired behavior, refer to the *RAM Megafunction User Guide*.

Figure 3–16 shows a sample functional waveform of mixed port read-during-write behavior for **Old Data** mode. In **Don't Care** mode, the old data is replaced with "Don't Care".

Figure 3–16. Mixed Port Read-During-Write: Old Data Mode



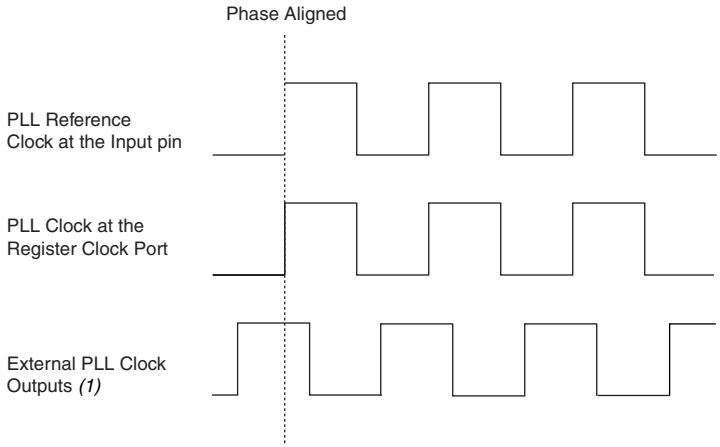
For mixed-port read-during-write operation with dual clocks, the relationship between the clocks determines the output behavior of the memory. If you use the same clock for the two clocks, the output is the old data from the address location. However, if you use different clocks, the output is unknown during the mixed-port read-during-write operation. This unknown value may be the old or new data at the address location, depending on whether the read happens before or after the write.

Conflict Resolution

When you are using M9K memory blocks in true dual-port mode, it is possible to attempt two write operations to the same memory location (address). Because there is no conflict resolution circuitry built into M9K memory blocks, this results in unknown data being written to that location. Therefore, you must implement conflict-resolution logic external to the M9K memory block.

Figure 5–14 shows a waveform example of the phase relationship of the PLL clocks in this mode.

Figure 5–14. Phase Relationship Between PLL Clocks in Normal Mode



Note to Figure 5–14:

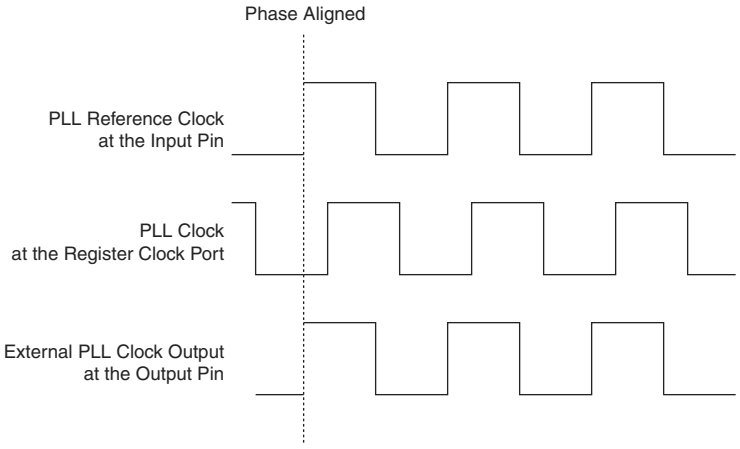
(1) The external clock output can lead or lag the PLL internal clock signals.

Zero Delay Buffer Mode

In zero delay buffer (ZDB) mode, the external clock output pin is phase-aligned with the clock input pin for zero delay through the device. When using this mode, use the same I/O standard on the input clock and output clocks to guarantee clock alignment at the input and output pins.

Figure 5–15 shows an example waveform of the phase relationship of the PLL clocks in ZDB mode.

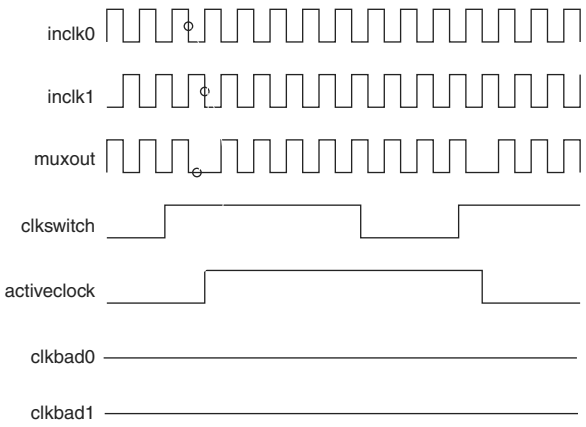
Figure 5–15. Phase Relationship Between PLL Clocks in ZDB Mode



In this mode, the activeclock signal mirrors the clkswitch signal. As both blocks are still functional during the manual switch, neither clkbad signals go 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 the automatic switch only works depending on the availability of the clock that is switched to. If the clock is unavailable, the state machine waits until the clock is available.

When CLKSWITCH = 1, it overrides the automatic switch-over function. As long as clkswitch signal is high, further switch-over action is blocked.

Figure 5–19. Clock Switchover Using the clkswitch Control *(1)*



Note to Figure 5–19:

(1) Both inclk0 and inclk1 must be running when the clkswitch signal goes high to start a manual clock switchover event.

Manual Clock Switchover

PLLs of Cyclone IV devices support manual switchover, in which the clkswitch signal controls whether inclk0 or inclk1 is the input clock to the PLL. The characteristics of a manual switchover are similar to the manual override feature in an automatic clock switchover, in which the switchover circuit is edge-sensitive. When the clkswitch signal goes high, the switchover sequence starts. The falling edge of the clkswitch signal does not cause the circuit to switch back to the previous input clock.

For more information about PLL software support in the Quartus II software, refer to the *ALTPLL Megafunction User Guide*.

Guidelines

Use the following guidelines to design with clock switchover in PLLs:

- Clock loss detection and automatic clock switchover require the inclk0 and inclk1 frequencies be within 20% of each other. Failing to meet this requirement causes the clkbad0 and clkbad1 signals to function improperly.

The user watchdog timer is disabled in factory configurations and during the configuration cycle of the application configuration. It is enabled after the application configuration enters user mode.

Quartus II Software Support

Implementation in your design requires a remote system upgrade interface between the Cyclone IV device logic array and remote system upgrade circuitry. You must also generate configuration files for production and remote programming of the system configuration memory. The Quartus II software provides these features.

The two implementation options, ALTREMOTE_UPDATE megafunction and remote system upgrade atom, are for the interface between the remote system upgrade circuitry and the device logic array interface. Using the megafunction block instead of creating your own logic saves design time and offers more efficient logic synthesis and device implementation.

For more information about the ALTREMOTE_UPDATE megafunction, refer to the *Remote Update Circuitry (ALTREMOTE_UPDATE) Megafunction User Guide*.

Document Revision History

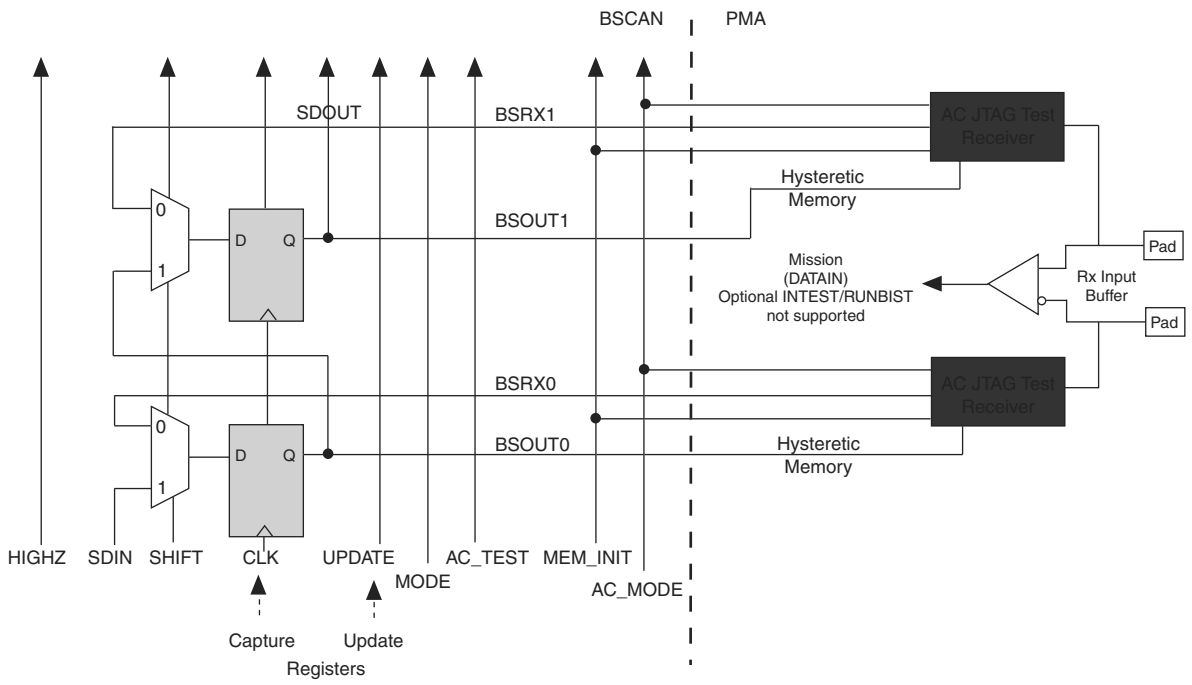
Table 8–28 lists the revision history for this chapter.

Table 8–28. Document Revision History (Part 1 of 2)

Date	Version	Changes
May 2013	1.7	■ Added Table 8–6. ■ Updated Table 8–9 to add new device options and packages. ■ Updated Figure 8–16 and Figure 8–22 to include user mode. ■ Updated the “Dedicated” column for DATA[0] and DCLK in Table 8–19. ■ Updated the “User Mode” and “Pin Type” columns for DCLK in Table 8–20.
February 2013	1.6	Updated Table 8–9 to add new device options and packages.
October 2012	1.5	■ Updated “AP Configuration Supported Flash Memories”, “Configuration Data Decompression”, and “Overriding the Internal Oscillator” sections. ■ Updated Figure 8–3, Figure 8–4, Figure 8–5, Figure 8–7, Figure 8–8, Figure 8–9, Figure 8–10, and Figure 8–11. ■ Updated Table 8–2, Table 8–8, Table 8–12, Table 8–13, Table 8–18, and Table 8–19.
November 2011	1.4	■ Added information about how to gain control of EPCS pins. ■ Updated “Reset”, “Single-Device AS Configuration”, “Single-Device AP Configuration”, and “Overriding the Internal Oscillator” sections. ■ Added Table 8–7. ■ Updated Table 8–6 and Table 8–19. ■ Updated Figure 8–3, Figure 8–4, and Figure 8–5.
December 2010	1.3	■ Updated for the Quartus II software version 10.1 release. ■ Added Cyclone IV E new device package information. ■ Updated Table 8–7, Table 8–10, and Table 8–11. ■ Minor text edits.

Figure 10–2 shows the Cyclone IV GX HSSI receiver BSC.

Figure 10–2. HSSI Receiver BSC with IEEE Std. 1149.6 BST Circuitry for the Cyclone IV GX Devices



For more information about Cyclone IV devices user I/O boundary-scan cells, refer to the *IEEE 1149.1 (JTAG) Boundary-Scan Testing for Cyclone III Devices* chapter.

BST Operation Control

Table 10–1 lists the boundary-scan register length for Cyclone IV devices.

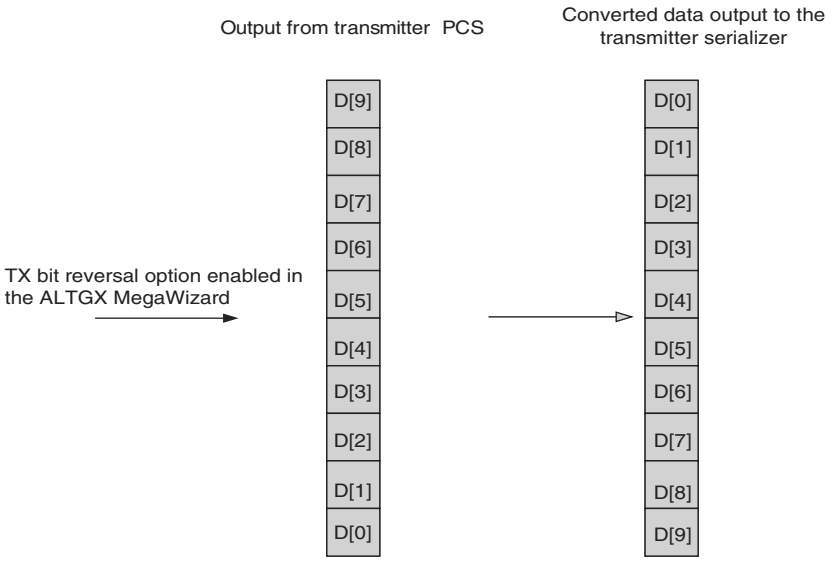
Table 10–1. Boundary-Scan Register Length for Cyclone IV Devices (Part 1 of 2)

Device	Boundary-Scan Register Length
EP4CE6	603
EP4CE10	603
EP4CE15	1080
EP4CE22	732
EP4CE30	1632
EP4CE40	1632
EP4CE55	1164
EP4CE75	1314
EP4CE115	1620
EP4CGX15	260
EP4CGX22	494
EP4CGX30 (1)	494
EP4CGX50	1006

Chapter 2. Cyclone IV Reset Control and Power Down

- Bit reversal—reverses the transmit bit order from LSB-to-MSB (default) to MSB-to-LSB at the input to the serializer. For example, input data to serializer `D[7..0]` is rewired to `D[0..7]` for 8-bit data width, and `D[9..0]` is rewired to `D[0..9]` for 10-bit data width. Figure 1–10 shows the transmitter bit reversal feature.

Figure 1–10. Transmitter Bit Reversal Operation in Basic Single-Width Mode



- Input bit-flip—reverses the bit order at a byte level at the input of the transmitter phase compensation FIFO. For example, if the 16-bit parallel transmitter data at the `tx_datain` port is '10111100 10101101' (16'hBCAD), selecting this option reverses the input data to the transmitter phase compensation FIFO to '00111101 10110101' (16'h3DB5).
- Bit-slip control—delays the data transmission by a number of specified bits to the serializer with the `tx_bitslipboundaryselect` port. For usage details, refer to the "Transmit Bit-Slip Control" on page 1–76.

Serializer

The serializer converts the low-speed parallel 8-bit or 10-bit data from the transmitter PCS to high-speed serial data for the transmitter output buffer. The serializer operates with a high-speed clock at half of the serial data rate. The serializer transmission sequence is LSB to MSB.

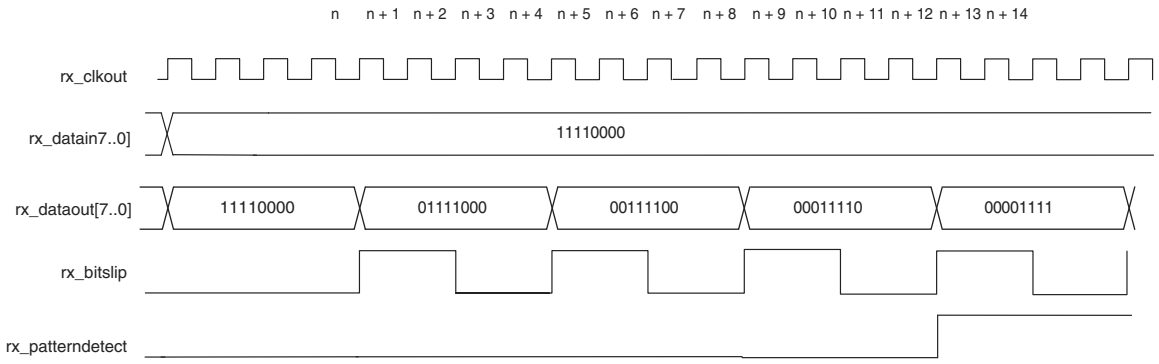
Bit-Slip Mode

In bit-slip mode, the rx_bitslip port controls the word aligner operation. At every rising edge of the rx_bitslip signal, the bit-slip circuitry slips one bit into the received data stream, effectively shifting the word boundary by one bit. When the received data after bit-slipping matches the programmed word alignment pattern, the rx_patterndetect signal is driven high for one parallel clock cycle.

☞ You can implement a bit-slip controller in the user logic that monitors either the rx_patterndetect signal or the receiver data output (rx_dataout), and controls the rx_bitslip port to achieve word alignment.

Figure 1–18 shows an example of the word aligner configured in bit-slip mode. 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 1–18. Word Aligner Configured in Bit-Slip Mode



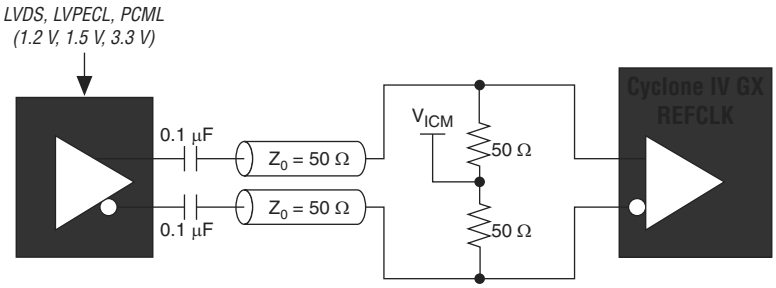
Automatic Synchronization State Machine Mode

In automatic synchronization state machine mode, the word aligner achieves synchronization after receiving a specific number of synchronization code groups, and falls out of synchronization after receiving a specific number of erroneous code groups. This mode provides hysteresis during link synchronization, which is required by protocols such as PCIe, GbE, XAUI, and Serial RapidIO.

☞ This mode is only supported using the 8B/10B encoded data with 10-bit input to the word aligner.

Figure 1–27 shows an example of the termination scheme for AC-coupled connections for REFCLK pins.

Figure 1–27. AC-Coupled Termination Scheme for a Reference Clock

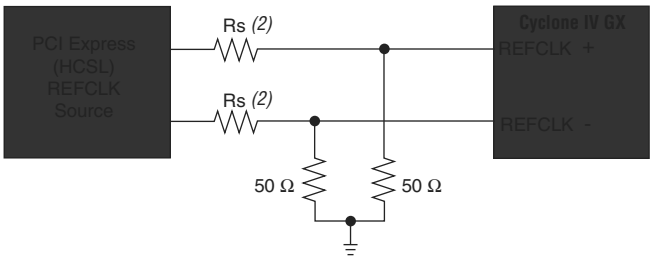


Note to Figure 1–27:

(1) For more information about the V_{ICM} value, refer to the *Cyclone IV Device Datasheet* chapter.

Figure 1–28 shows an example termination scheme for the REFCLK pin when configured as a **HCSL** input.

Figure 1–28. Termination Scheme for a Reference Clock When Configured as HCSL (1)



Notes to Figure 1–28:

(1) No biasing is required if the reference clock signals are generated from a clock source that conforms to the PCIe specification.

(2) Select values as recommended by the PCIe clock source vendor.

Transceiver Channel Datapath Clocking

Channel datapath clocking varies with channel configuration options and PCS configurations. This section describes the clock distribution from the left PLLs for transceiver channels and the datapath clocking in various supported configurations.

Table 1–7 lists the clocks generated by the PLLs for transceiver datapath.

Table 1–7. PLL Clocks for Transceiver Datapath

Clock	Usage
CDR clocks	Receiver CDR unit
High-speed clock	Transmitter serializer block in PMA
Low-speed clock	Transmitter PCS blocks Receiver PCS blocks when rate match FIFO enabled

Table 1–9 lists the high- and low-speed clock sources for each channel.

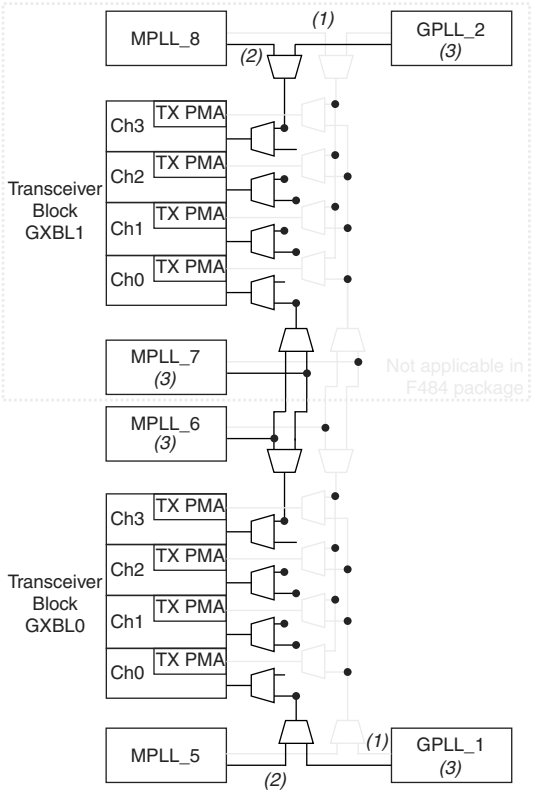
Table 1–9. High- and Low-Speed Clock Sources for Each Channel in Non-Bonded Channel Configuration

Package	Transceiver Block	Transceiver Channel	High- and Low-Speed Clocks Sources	
			Option 1	Option 2
F324 and smaller	GXBL0	All channels	MPLL_1	MPLL_2
F484 and larger	GXBL0	Channels 0, 1	MPLL_5/GPLL_1	MPLL_6
		Channels 2, 3	MPLL_5	MPLL_6/MPLL_7 (1)
	GXBL1 (1)	Channels 0, 1	MPLL_7/MPLL_6	MPLL_8
		Channels 2, 3	MPLL_7	MPLL_8/GPLL_2

Note to Table 1–9:

(1) MPLL_7 and GXBL1 are not applicable for transceivers in F484 package

Figure 1–32. Clock Distribution in Non-Bonded Channel Configuration for Transceivers in F484 and Larger Packages



Notes to Figure 1–32:

(1) High-speed clock.

(2) Low-speed clock.

(3) These PLLs have restricted clock driving capability and may not reach all connected channels. For details, refer to Table 1–9.

The transceiver datapath clocking varies in non-bonded channel configuration depending on the PCS configuration.

Figure 1–33 shows the datapath clocking in transmitter only operation. In this mode, each channel selects the high- and low-speed clock from one of the supported PLLs. The high-speed clock feeds to the serializer for parallel to serial operation. The low-speed clock feeds to the following blocks in the transmitter PCS:

- 8B/10B encoder
- read clock of the byte serializer
- read clock of the TX phase compensation FIFO

Figure 1–35 shows the datapath clocking in the transmitter and receiver operation mode with the rate match FIFO. The receiver datapath clocking in configuration without the rate match FIFO is identical to Figure 1–34.

In configuration with the rate match FIFO, the CDR unit in the receiver channel recovers the clock from received serial data and generates the high-speed recovered clock for the deserializer, and low-speed recovered clock for forwarding to the receiver PCS. The low-speed recovered clock feeds to the following blocks in the receiver PCS:

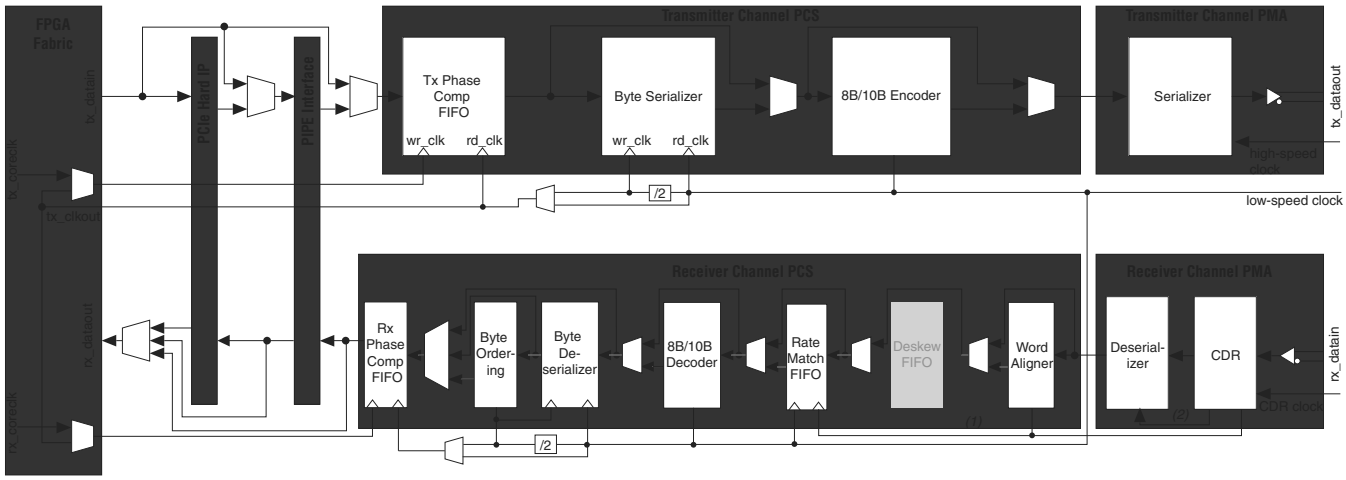
- word aligner
- write clock of rate match FIFO

The low-speed clock that is used in the transmitter PCS datapath feeds the following blocks in the receiver PCS:

- read clock of rate match FIFO
- 8B/10B decoder
- write clock of byte deserializer
- byte ordering
- write clock of RX phase compensation FIFO

When the byte deserializer is enabled, the low-speed clock frequency is halved before feeding into the write clock of RX phase compensation FIFO. The low-speed clock is available in the FPGA fabric as `tx_clkout` port, which can be used in the FPGA fabric to send transmitter data and control signals, and capture receiver data and status signals.

Figure 1–35. Transmitter and Receiver Datapath Clocking with Rate Match FIFO in Non-Bonded Channel Configuration



Notes to Figure 1–35:

(1) Low-speed recovered clock.

(2) High-speed recovered clock.

- transmitter in electrical idle
- receiver signal detect
- receiver spread spectrum clocking

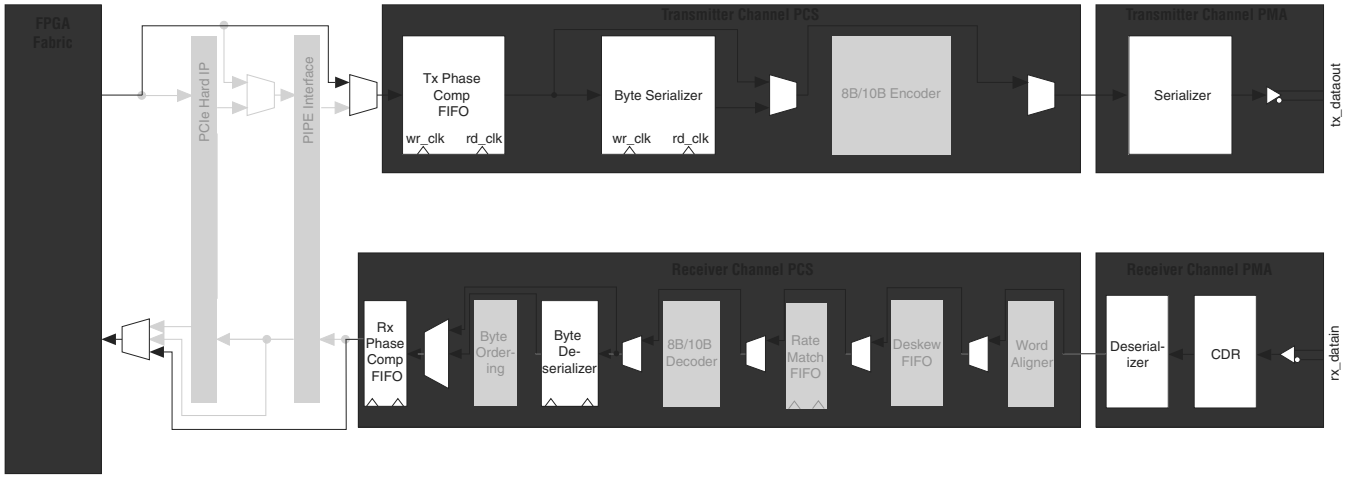
Low-Latency PCS Operation

When configured in low-latency PCS operation, the following blocks in the transceiver PCS are bypassed, resulting in a lower latency PCS datapath:

- 8B/10B encoder and decoder
- word aligner
- rate match FIFO
- byte ordering

Figure 1–47 shows the transceiver channel datapath in Basic mode with low-latency PCS operation.

Figure 1–47. Transceiver Channel Datapath in Basic Mode with Low-Latency PCS Operation



Transmitter in Electrical Idle

The transmitter buffer supports electrical idle state, where when enabled, the differential output buffer driver is tri-stated. During electrical idle, the output buffer assumes the common mode output voltage levels. For details about the electrical idle features, refer to "PCI Express (PIPE) Mode" on page 1–52.

The transmitter in electrical idle feature is required for compliance to the version 2.00 of PHY Interface for the PCI Express (PIPE) Architecture specification for PCIe protocol implementation.

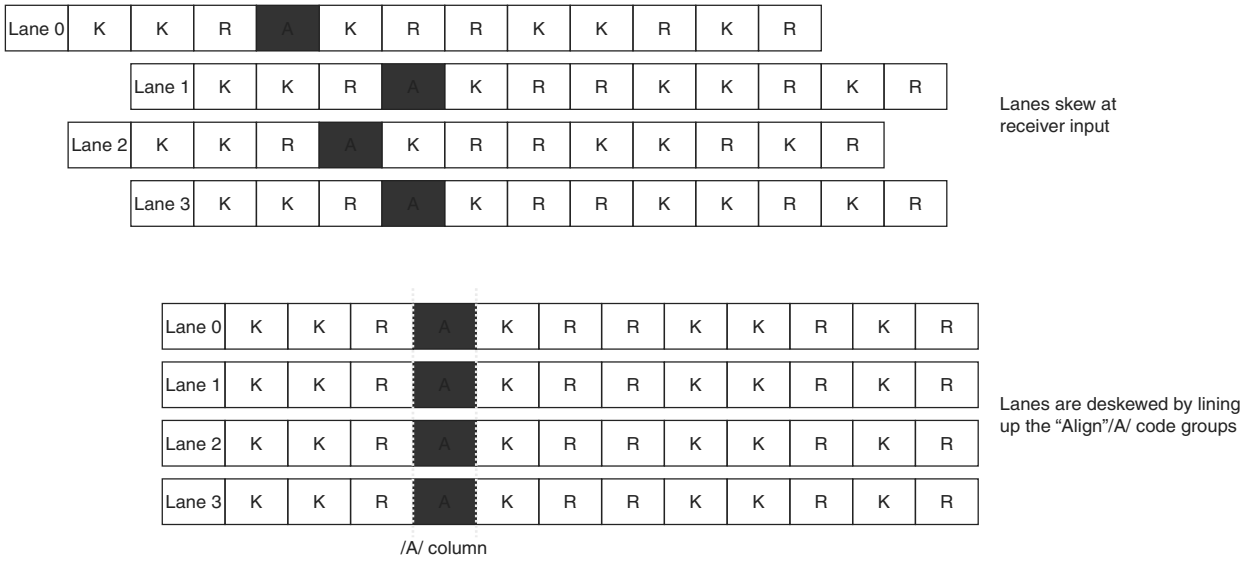
Signal Detect at Receiver

Signal detect at receiver is only supported when 8B/10B encoder/decoder block is enabled.

- Channel alignment is acquired if three additional aligned ||A|| columns are observed at the output of the deskew FIFOs of the four channels after alignment of the first ||A|| column.
- Channel alignment is indicated by the assertion of `rx_channelaligned` signal.
- After acquiring channel alignment, if four misaligned ||A|| columns are seen at the output of the deskew FIFOs in all four channels with no aligned ||A|| columns in between, the `rx_channelaligned` signal is deasserted, indicating loss of channel alignment.

Figure 1–65 shows lane skew at the receiver input and how the deskew FIFO uses the /A/ code group to align the channels.

Figure 1–65. Deskew FIFO–Lane Skew at the Receiver Input



Lane Synchronization

In XAUI mode, the word aligner is configured in automatic synchronization state machine mode that is compliant to the PCS synchronization state diagram specified in clause 48 of the IEEE P802.3ae specification. Table 1–23 lists the synchronization state machine parameters that implements the lane synchronization in XAUI mode.

Table 1–23. Synchronization State Machine Parameters (1)

Parameter	Value
Number of valid synchronization (/K28.5/) code groups received to achieve synchronization	4
Number of erroneous code groups received to lose synchronization	4
Number of continuous good code groups received to reduce the error count by one	4

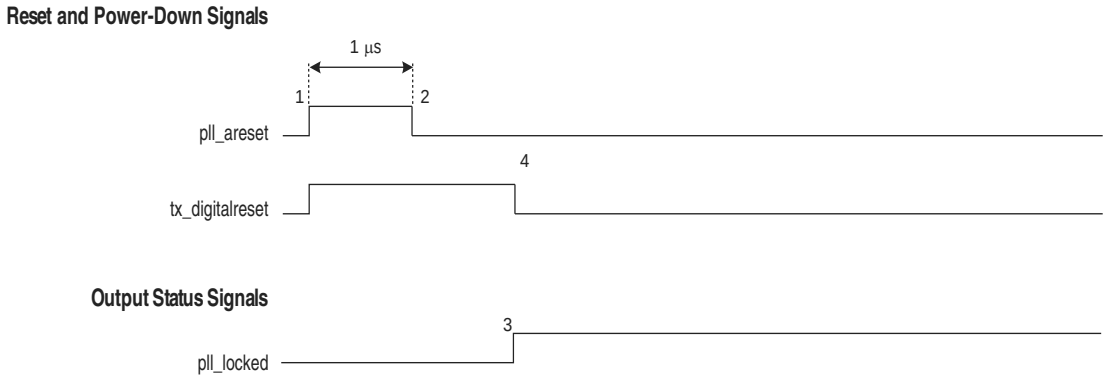
Note to Table 1–23:

(1) The word aligner supports 7-bit and 10-bit pattern lengths in XAUI mode.

Transmitter Only Channel

This configuration contains only a transmitter channel. If you create a **Transmitter Only** instance in the ALTGX MegaWizard Plug-In Manager in Basic ×4 functional mode, use the reset sequence shown in Figure 2–3.

Figure 2–3. Sample Reset Sequence for Bonded and Non-Bonded Configuration Transmitter Only Channels



As shown in Figure 2–3, perform the following reset procedure for the **Transmitter Only** channel configuration:

1. After power up, assert `pll_areset` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset` signal asserted during this time period. After you de-assert the `pll_areset` signal, the multipurpose PLL starts locking to the transmitter input reference clock.
3. When the multipurpose PLL locks, as indicated by the `pll_locked` signal going high (marker 3), de-assert the `tx_digitalreset` signal (marker 4). At this point, the transmitter is ready for transmitting data.

Table 3–2. Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 4 of 7)

<table>
<tr><th>Port Name</th><th>Input/ Output</th><th>Description</th></tr>
<tr><td colspan="3">Analog Settings Control/Status Signals</td></tr>
<tr><td>tx_vodctrl[2..0] (1)</td><td>Input</td><td>This is an optional transmit buffer V_{OD} control signal. It is 3 bits per transmitter channel. The number of settings varies based on the transmit buffer supply setting and the termination resistor setting on the TX Analog screen of the ALTGX MegaWizard Plug-In Manager.

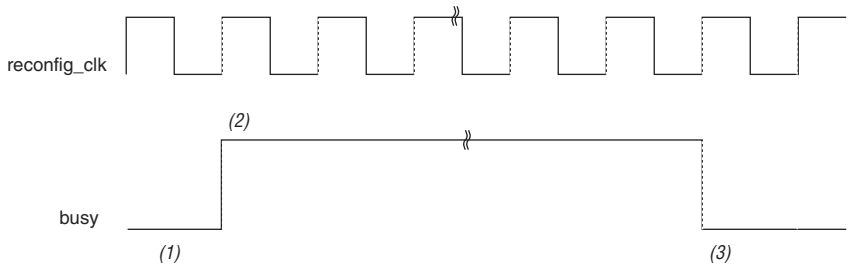
The width of this signal is fixed to 3 bits if you enable either the Use 'logical_channel_address' port for Analog controls reconfiguration option or the Use same control signal for all the channels option in the Analog controls screen. Otherwise, the width of this signal is 3 bits per channel.

The following shows the V_{OD} values corresponding to the tx_vodctrl settings for 100-Ω termination.

For more information, refer to the “Programmable Output Differential Voltage” section of the Cyclone IV GX Device Datasheet chapter.
<table>
<tr><th>tx_vodctrl[2:0]</th><th>Corresponding ALTGX instance settings</th><th>Corresponding V_{OD} settings (mV)</th></tr>
<tr><td>3’b001</td><td>1</td><td>400</td></tr>
<tr><td>3’b010</td><td>2</td><td>600</td></tr>
<tr><td>3’b011</td><td>3</td><td>800</td></tr>
<tr><td>3’b111</td><td>4 (2)</td><td>900 (2)</td></tr>
<tr><td>3’b100</td><td>5</td><td>1000</td></tr>
<tr><td>3’b101</td><td>6</td><td>1200</td></tr>
<tr><td colspan="3">All other values => N/A</td></tr>
</table>
</td></tr>
</table>

Figure 3–3 shows the timing diagram for a offset cancellation process.

Figure 3–3. Dynamic Reconfiguration Signals Transition during Offset Cancellation



Notes to Figure 3–3:

(1) After device power up, the busy signal remains low for the first reconfig_clk cycle.
(2) The busy signal then gets asserted for the second reconfig_clk cycle, when the dynamic reconfiguration controller initiates the offset cancellation process.
(3) The deassertion of the busy signal indicates the successful completion of the offset cancellation process.

Functional Simulation of the Offset Cancellation Process

You must connect the ALTGX_RECONFIG instances to the ALTGX instances in your design for functional simulation. Functional simulation uses a reduced timing model of the dynamic reconfiguration controller. Therefore, the duration of the offset cancellation process is 16 reconfig_clk clock cycles for functional simulation only. The gxb_powerdown signal must not be asserted during the offset cancellation sequence (for functional simulation and silicon).

Dynamic Reconfiguration Modes

When you enable the dynamic reconfiguration feature, you can reconfigure the following portions of each transceiver channel dynamically, without powering down the other transceiver channels or the FPGA fabric of the device:

- Analog (PMA) controls reconfiguration
- Channel reconfiguration
- PLL reconfiguration

Table 3–3 lists the supported dynamic reconfiguration modes for Cyclone IV GX devices.

Table 3–3. Cyclone IV GX Supported Dynamic Reconfiguration Mode (Part 1 of 2)

Dynamic Reconfiguration Supported Mode	Operational Mode			Quartus II Instances			.mif Requirements
	Transmitter Only	Receiver Only	Transmitter and Receiver Only	ALTGX	ALTGX_RECONFIG	ALTPLL_RECONFIG	
Offset Cancellation	—	✓	✓	✓	✓	—	—
Analog (PMA) Controls Reconfiguration	✓	✓	✓	✓	✓	—	—

Option 1: Share a Single Transmitter Core Clock Between Transmitters

- Enable this option if you want `tx_clkout` of the first channel (channel 0) of the transceiver block to provide the write clock to the Transmitter Phase Compensation FIFOs of the remaining channels in the transceiver block.
- This option is typically enabled when all the channels of a transceiver block have the same functional mode and data rate and are reconfigured to the identical functional mode and data rate.

Figure 3–11 shows the sharing of channel 0's `tx_clkout` between all four regular channels of a transceiver block.

Figure 3–11. Option 1 for Transmitter Core Clocking (Channel Reconfiguration Mode)

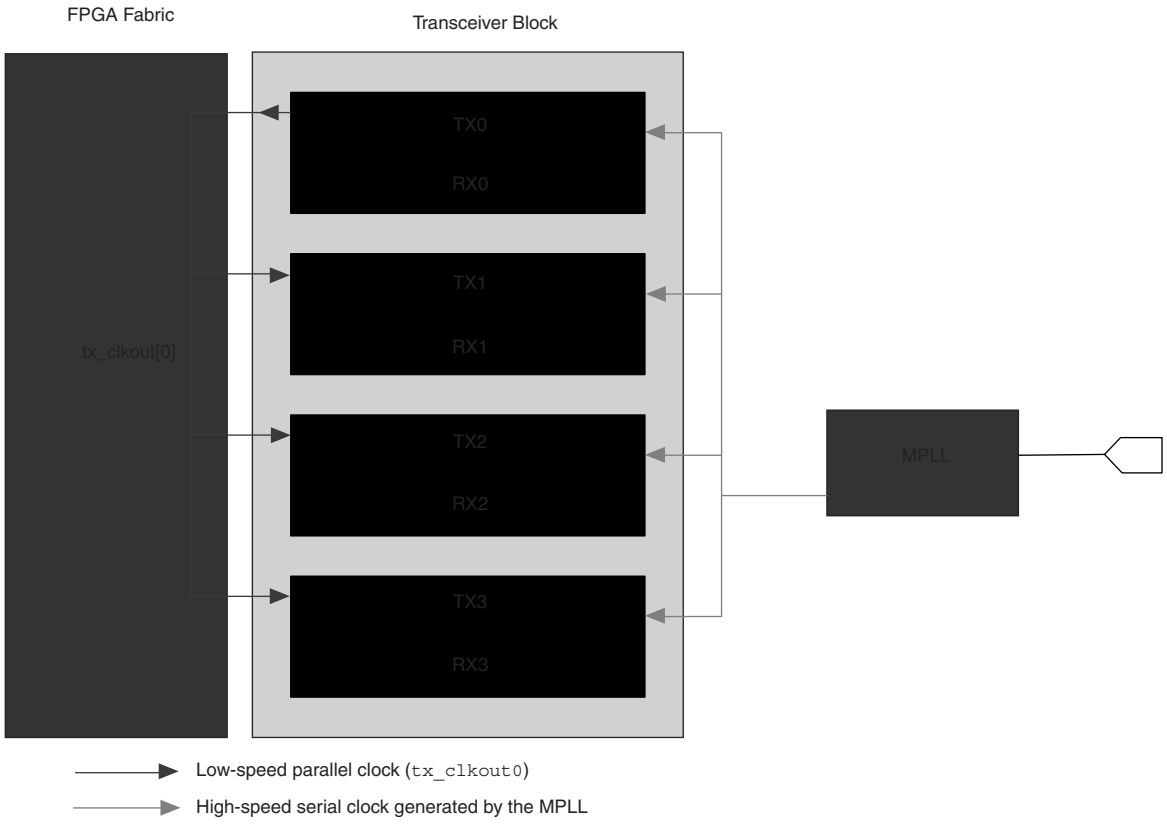


Table 1–35. Emulated LVDS Transmitter Timing Specifications for Cyclone IV Devices (1), (3) (Part 2 of 2)

Symbol	Modes	C6		C7, I7		C8, A7		C8L, I8L		C9L		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
t_{DUTY}	—	45	55	45	55	45	55	45	55	45	55	%
TCCS	—	—	200	—	200	—	200	—	200	—	200	ps
Output jitter (peak to peak)	—	—	500	—	500	—	550	—	600	—	700	ps
t_{LOCK} (2)	—	—	1	—	1	—	1	—	1	—	1	ms

Notes to Table 1–35:

(1) Cyclone IV E—emulated LVDS transmitter is supported at the output pin of all I/O Banks.
Cyclone IV GX—emulated LVDS transmitter is supported at the output pin of I/O Banks 3, 4, 5, 6, 7, 8, and 9.

(2) t_{LOCK} is the time required for the PLL to lock from the end-of-device configuration.

(3) Cyclone IV E 1.0 V core voltage devices only support C8L, C9L, and I8L speed grades. Cyclone IV E 1.2 V core voltage devices only support C6, C7, C8, I7, and A7 speed grades. Cyclone IV GX devices only support C6, C7, C8, and I7 speed grades.

Table 1–36. LVDS Receiver Timing Specifications for Cyclone IV Devices (1), (3)

Symbol	Modes	C6		C7, I7		C8, A7		C8L, I8L		C9L		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
f_{HSCLK} (input clock frequency)	×10	10	437.5	10	370	10	320	10	320	10	250	MHz
	×8	10	437.5	10	370	10	320	10	320	10	250	MHz
	×7	10	437.5	10	370	10	320	10	320	10	250	MHz
	×4	10	437.5	10	370	10	320	10	320	10	250	MHz
	×2	10	437.5	10	370	10	320	10	320	10	250	MHz
	×1	10	437.5	10	402.5	10	402.5	10	362	10	265	MHz
HSIODR	×10	100	875	100	740	100	640	100	640	100	500	Mbps
	×8	80	875	80	740	80	640	80	640	80	500	Mbps
	×7	70	875	70	740	70	640	70	640	70	500	Mbps
	×4	40	875	40	740	40	640	40	640	40	500	Mbps
	×2	20	875	20	740	20	640	20	640	20	500	Mbps
	×1	10	437.5	10	402.5	10	402.5	10	362	10	265	Mbps
SW	—	—	400	—	400	—	400	—	550	—	640	ps
Input jitter tolerance	—	—	500	—	500	—	550	—	600	—	700	ps
t_{LOCK} (2)	—	—	1	—	1	—	1	—	1	—	1	ms

Notes to Table 1–36:

(1) Cyclone IV E—LVDS receiver is supported at all I/O Banks.
Cyclone IV GX—LVDS receiver is supported at I/O Banks 3, 4, 5, 6, 7, 8, and 9.

(2) t_{LOCK} is the time required for the PLL to lock from the end-of-device configuration.

(3) Cyclone IV E 1.0 V core voltage devices only support C8L, C9L, and I8L speed grades. Cyclone IV E 1.2 V core voltage devices only support C6, C7, C8, I7, and A7 speed grades. Cyclone IV GX devices only support C6, C7, C8, and I7 speed grades.

External Memory Interface Specifications

The external memory interfaces for Cyclone IV devices are auto-calibrating and easy to implement.