Intel - EP4CE15F17I7 Datasheet

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-LBGA
Supplier Device Package	256-FBGA (17x17)
Purchase URL	https://www.e-xfl.com/product-detail/intel/ep4ce15f17i7

Word-Wide Multi-Device AP Configuration . . . 8–26
Guidelines for Connecting Parallel Flash to Cyclone IV E Devices for an AP Interface . . . 8–28
Configuring With Multiple Bus Masters . . . 8–28
Estimating AP Configuration Time . . . 8–30
Programming Parallel Flash Memories . . . 8–31
PS Configuration . . . 8–32
PS Configuration Using an External Host . . . 8–33
PS Configuration Timing . . . 8–36
PS Configuration Using a Download Cable . . . 8–37
FPP Configuration . . . 8–40
FPP Configuration Using an External Host . . . 8–40
FPP Configuration Timing . . . 8–44
JTAG Configuration . . . 8–45
Configuring Cyclone IV Devices with Jam STAPL . . . 8–52
Configuring Cyclone IV Devices with the JRunner Software Driver . . . 8–52
Combining JTAG and AS Configuration Schemes . . . 8–53
Programming Serial Configuration Devices In-System with the JTAG Interface . . . 8–55
JTAG Instructions . . . 8–57
Device Configuration Pins . . . 8–62
Remote System Upgrade . . . 8–69
Functional Description . . . 8–69
Enabling Remote Update . . . 8–70
Configuration Image Types . . . 8–70
Remote System Upgrade Mode . . . 8–71
Remote Update Mode . . . 8–71
Dedicated Remote System Upgrade Circuitry . . . 8–74
Remote System Upgrade Registers . . . 8–75
Remote System Upgrade State Machine . . . 8–78
User Watchdog Timer . . . 8–79
Quartus II Software Support . . . 8–80
Document Revision History . . . 8–80

Chapter 9. SEU Mitigation in Cyclone IV Devices

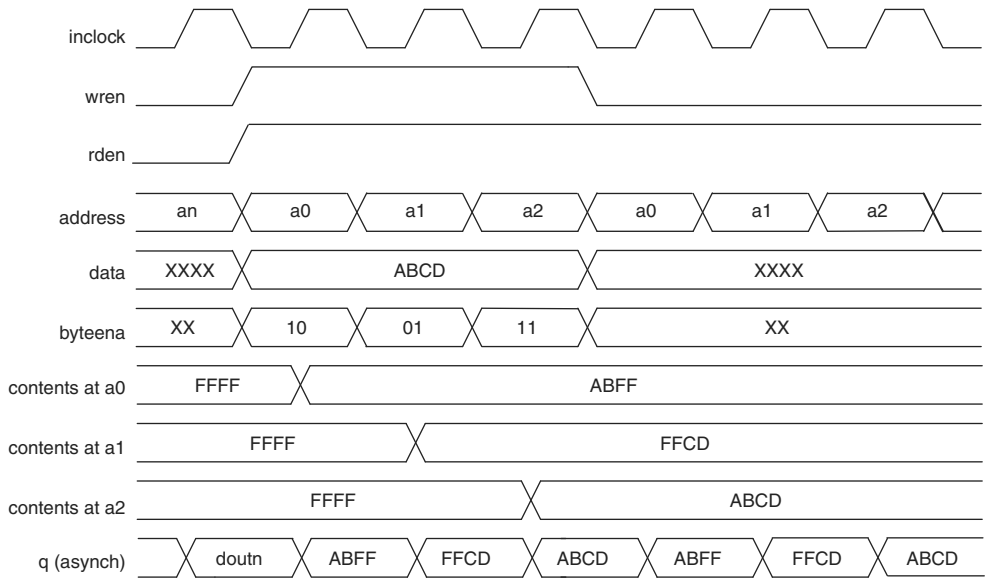
Configuration Error Detection . . . 9–1
User Mode Error Detection . . . 9–2
Automated SEU Detection . . . 9–3
CRC_ERROR Pin . . . 9–3
Error Detection Block . . . 9–4
Error Detection Registers . . . 9–4
Error Detection Timing . . . 9–5
Software Support . . . 9–6
Accessing Error Detection Block Through User Logic . . . 9–7
Recovering from CRC Errors . . . 9–9
Document Revision History . . . 9–10

Chapter 10. JTAG Boundary-Scan Testing for Cyclone IV Devices

IEEE Std. 1149.6 Boundary-Scan Register . . . 10–2
BST Operation Control . . . 10–3
EXTEST_PULSE . . . 10–5
EXTEST_TRAIN . . . 10–5
I/O Voltage Support in a JTAG Chain . . . 10–5
Boundary-Scan Description Language Support . . . 10–6
Document Revision History . . . 10–7

Figure 3–1 shows how the `wren` and `byteena` signals control the RAM operations.

Figure 3–1. Cyclone IV Devices byteena Functional Waveform *(1)*



Note to Figure 3–1:

(1) For this functional waveform, **New Data** mode is selected.

When a `byteena` bit is deasserted during a write cycle, the old data in the memory appears in the corresponding data-byte output. When a `byteena` bit is asserted during a write cycle, the corresponding data-byte output depends on the setting chosen in the Quartus® II software. The setting can either be the newly written data or the old data at that location.

☞ Byte enables are only supported for True Dual-Port memory configurations when both the PortA and PortB data widths of the individual M9K memory blocks are multiples of 8 or 9 bits.

Packed Mode Support

Cyclone IV devices M9K memory blocks support packed mode. You can implement two single-port memory blocks in a single block under the following conditions:

- Each of the two independent block sizes is less than or equal to half of the M9K block size. The maximum data width for each independent block is 18 bits wide.
- Each of the single-port memory blocks is configured in single-clock mode. For more information about packed mode support, refer to "Single-Port Mode" on page 3–8 and "Single-Clock Mode" on page 3–15.

Table 5–1. GCLK Network Connections for EP4CGX15, EP4CGX22, and EP4CGX30 (1), (2) (Part 2 of 2)

GCLK Network Clock Sources	GCLK Networks																			
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
PLL_3_C1	—	—	—	—	—	—	✓	—	—	✓	—	—	—	—	—	—	✓	—	—	✓
PLL_3_C2	—	—	—	—	—	✓	—	✓	—	—	—	—	—	—	—	✓	—	✓	—	—
PLL_3_C3	—	—	—	—	—	—	✓	—	✓	—	—	—	—	—	—	—	✓	—	✓	—
PLL_3_C4	—	—	—	—	—	—	—	✓	—	✓	—	—	—	—	—	—	—	✓	—	✓
PLL_4_C0 (3)	—	—	—	—	—	✓	—	—	✓	—	✓	—	—	✓	—	—	—	—	—	—
PLL_4_C1 (3)	—	—	—	—	—	—	✓	—	—	✓	—	✓	—	—	✓	—	—	—	—	—
PLL_4_C2 (3)	—	—	—	—	—	✓	—	✓	—	—	✓	—	✓	—	—	—	—	—	—	—
PLL_4_C3 (3)	—	—	—	—	—	—	✓	—	✓	—	—	✓	—	✓	—	—	—	—	—	—
PLL_4_C4 (3)	—	—	—	—	—	—	—	✓	—	✓	—	—	✓	—	✓	—	—	—	—	—
DPCLK2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—
DPCLK3 (4)	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—
DPCLK4 (4)	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—
DPCLK5	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓
DPCLK6 (4)	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	—	—	—	—
DPCLK7	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	—	—	—	—	—	—
DPCLK8	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	—	—	—
DPCLK9 (4)	—	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	—	—	—	—	—
DPCLK10	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—
DPCLK11 (4)	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—	—
DPCLK12 (4)	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—
DPCLK13	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	—

Notes to Table 5–1:

(1) EP4CGX30 information in this table refers to all EP4CGX30 packages except F484 package.

(2) PLL_1 and PLL_2 are multipurpose PLLs while PLL_3 and PLL_4 are general purpose PLLs.

(3) PLL_4 is only available in EP4CGX22 and EP4CGX30 devices in F324 package.

(4) This pin applies to EP4CGX22 and EP4CGX30 devices.

Table 6–2 lists the I/O standards that support impedance matching and series termination.

Table 6–2. Cyclone IV Device I/O Features Support (Part 1 of 2)

I/O Standard	IOH/IOL Current Strength Setting (mA) (1), (9)		R_S OCT with Calibration Setting, Ohm (Ω)		R_S OCT Without Calibration Setting, Ohm (Ω)		Cyclone IV E I/O Banks Support	Cyclone IV GX I/O Banks Support	Slew Rate Option (6)	PCI-clamp Diode Support
	Column I/O	Row I/O	Column I/O	Row I/O (8)	Column I/O	Row I/O (8)				
3.3-V LVTTL	4,8	4,8	—	—	—	—	1,2,3,4, 5,6,7,8	3,4,5,6, 7,8,9	—	✓
3.3-V LVCMOS	2	2	—	—	—	—			—	✓
3.0-V LVTTL	4,8,12,16	4,8,12,16	50,25	50,25	50,25	50,25			0,1,**2**	✓
3.0-V LVCMOS	4,8,12,16	4,8,12,16	50,25	50,25	50,25	50,25				✓
3.0-V PCI/PCI-X	—	—	—	—	—	—			—	✓
2.5-V LVTTL/LVCMOS	4,8,12,16	4,8,12,16	50,25	50,25	50,25	50,25			0,1,**2**	✓
1.8-V LVTTL/LVCMOS	2,4,6,8,10,12,16	2,4,6,8,10,12,16	50,25	50,25	50,25	50,25				—
1.5-V LVCMOS	2,4,6,8,10,12,16	2,4,6,8,10,12,16	50,25	50,25	50,25	50,25				—
1.2-V LVCMOS	2,4,6,8,10,12	2,4,6,8,10	50,25	50	50,25	50		4,5,6,7, 8		—
SSTL-2 Class I	8,12	8,12	50	50	50	50		3,4,5,6, 7,8,9		—
SSTL-2 Class II	16	16	25	25	25	25				—
SSTL-18 Class I	8,10,12	8,10,12	50	50	50	50				—
SSTL-18 Class II	12,16	12,16	25	25	25	25				—
HSTL-18 Class I	8,10,12	8,10,12	50	50	50	50				—
HSTL-18 Class II	16	16	25	25	25	25				—
HSTL-15 Class I	8,10,12	8,10,12	50	50	50	50				—
HSTL-15 Class II	16	16	25	25	25	25				—
HSTL-12 Class I	8,10,12	8,10	50	50	50	50		4,5,6,7, 8		—
HSTL-12 Class II	14	—	25	—	25	—	3,4,7,8	4,7,8		—
Differential SSTL-2 Class I (2), (7)	8,12	8,12	50	50	50	50	1,2,3,4, 5,6,7,8	3,4,5,6, 7,8	0,1,**2**	—
Differential SSTL-2 Class II (2), (7)	16	16	25	25	25	25				—
Differential SSTL-18 (2), (7)	8,10,12	—	50	—	50	—				—
Differential HSTL-18 (2), (7)	8,10,12	—	50	—	50	—				—
Differential HSTL-15 (2), (7)	8,10,12	—	50	—	50	—				—
Differential HSTL-12 (2), (7)	8,10,12	—	50	—	50	—	3,4,7,8	4,7,8		—

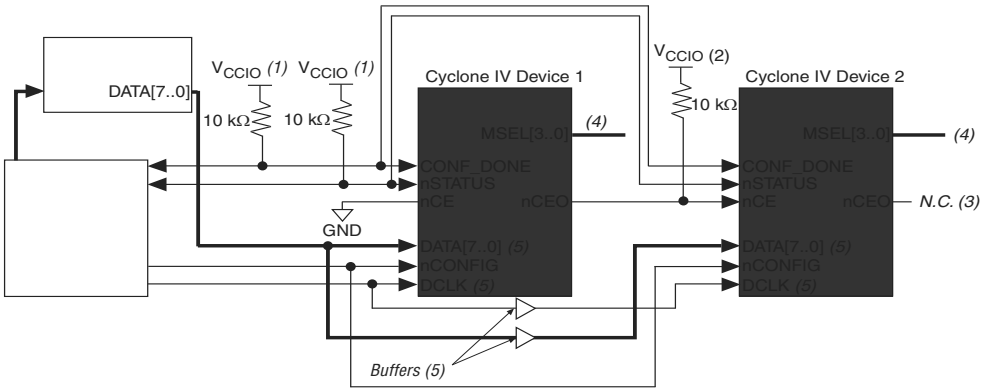
To ensure that DCLK and DATA[0] are not left floating at the end of the configuration, the MAX II device must drive them either high or low, whichever is convenient on your board. The DATA[0] pin is available as a user I/O pin after configuration. When you choose the FPP scheme in the Quartus II software, the DATA[0] pin is tri-stated by default in user mode and must be driven by the external host device. To change this default option in the Quartus II software, select the **Dual-Purpose Pins** tab of the **Device and Pin Options** dialog box.

The DCLK speed must be below the specified system frequency to ensure correct configuration. No maximum DCLK period exists, which means you can pause configuration by halting DCLK for an indefinite amount of time.

The external host device can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The CONF_DONE pin must be monitored by the external device to detect errors and to determine when programming is complete. If all configuration data is sent, but CONF_DONE or INIT_DONE has not gone high, the external device must reconfigure the target device.

Figure 8–20 shows how to configure multiple devices with a MAX II device. This circuit is similar to the FPP configuration circuit for a single device, except the Cyclone IV devices are cascaded for multi-device configuration.

Figure 8–20. Multi-Device FPP Configuration Using an External Host



Notes to Figure 8–20:

(1) The pull-up resistor must be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} must be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

(2) Connect the pull-up resistor to the V_{CCIO} supply voltage of the I/O bank in which the nCE pin resides.

(3) The nCEO pin is left unconnected or used as a user I/O pin when it does not feed the nCE pin of another device.

(4) The MSEL pin settings vary for different configuration voltage standards and POR time. To connect the MSEL pins, refer to Table 8–4 on page 8–8 and Table 8–5 on page 8–9. Connect the MSEL pins directly to V_{CCA} or GND.

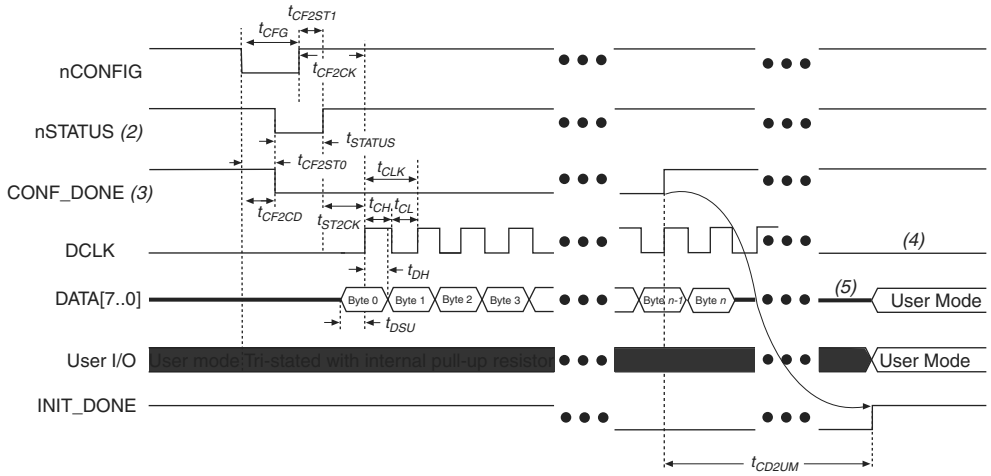
(5) All I/O inputs must maintain a maximum AC voltage of 4.1 V. DATA[7..0] and DCLK must fit the maximum overshoot outlined in Equation 8–1 on page 8–5.

After the first device completes configuration in a multi-device configuration chain, its nCEO pin drives low to activate the nCE pin of the second device, which prompts the second device to begin configuration. The second device in the chain begins configuration in one clock cycle; therefore, the transfer of data destinations is transparent to the MAX II device. All other configuration pins (nCONFIG, nSTATUS,

FPP Configuration Timing

Figure 8–22 shows the timing waveform for the FPP configuration when using an external host.

Figure 8–22. FPP Configuration Timing Waveform *(1)*



Notes to Figure 8–22:

(1) The beginning of this waveform shows the device in user mode. In user mode, `nCONFIG`, `nSTATUS`, and `CONF_DONE` are at logic-high levels. When `nCONFIG` is pulled low, a reconfiguration cycle begins.

(2) After power up, the Cyclone IV device holds `nSTATUS` low during POR delay.

(3) After power up, before and during configuration, `CONF_DONE` is low.

(4) Do not leave `DCLK` floating after configuration. It must be driven high or low, whichever is more convenient.

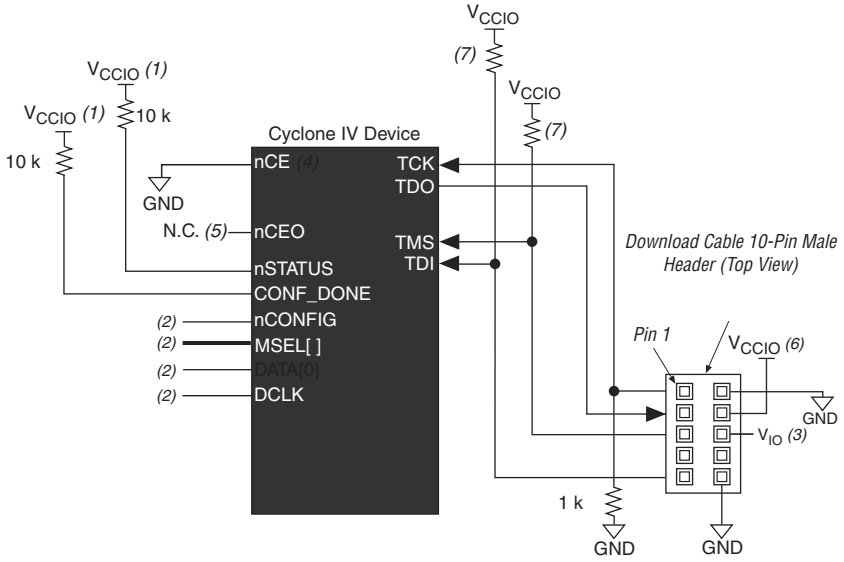
(5) `DATA[7..0]` is available as a user I/O pin after configuration; the state of the pin depends on the dual-purpose pin settings.

Table 8–13 lists the FPP configuration timing parameters for Cyclone IV devices.

Table 8–13. FPP Timing Parameters for Cyclone IV Devices (Part 1 of 2)

Symbol	Parameter	Minimum		Maximum		Unit
		Cyclone IV *(1)*	Cyclone IV E *(2)*	Cyclone IV *(1)*	Cyclone IV E *(2)*	
t_{CF2CD}	`nCONFIG` low to `CONF_DONE` low	—		500		ns
t_{CF2ST0}	`nCONFIG` low to `nSTATUS` low	—		500		ns
t_{CFG}	`nCONFIG` low pulse width	500		—		ns
t_{STATUS}	`nSTATUS` low pulse width	45		230 *(3)*		µs
t_{CF2ST1}	`nCONFIG` high to `nSTATUS` high	—		230 *(4)*		µs
t_{CF2CK}	`nCONFIG` high to first rising edge on `DCLK`	230 *(3)*		—		µs

Figure 8–24. JTAG Configuration of a Single Device Using a Download Cable (1.5-V or 1.8-V V_{CCIO} Powering the JTAG Pins)



Notes to Figure 8–24:

(1) Connect these pull-up resistors to the V_{CCIO} supply of the bank in which the pin resides.

(2) Connect the `nCONFIG` and `MSEL` pins to support a non-JTAG configuration scheme. If you only use JTAG configuration, connect the `nCONFIG` pin to logic-high and the `MSEL` pins to GND. In addition, pull `DCLK` and `DATA[0]` to either high or low, whichever is convenient on your board.

(3) In the USB-Blaster and ByteBlaster II cables, this pin is connected to `nCE` when it is used for AS programming; otherwise it is a no connect.

(4) The `nCE` must be connected to GND or driven low for successful JTAG configuration.

(5) The `nCEO` pin is left unconnected or used as a user I/O pin when it does not feed the `nCE` pin of another device.

(6) Power up the V_{CC} of the EthernetBlaster, ByteBlaster II or USB-Blaster cable with supply from V_{CCIO}. The Ethernet-Blaster, ByteBlaster II, and USB-Blaster cables do not support a target supply voltage of 1.2 V. For the target supply voltage value, refer to the *ByteBlaster II Download Cable User Guide*, the *USB-Blaster Download Cable User Guide*, and the *EthernetBlaster Communications Cable User Guide*.

(7) Resistor value can vary from 1 kΩ to 10 kΩ.

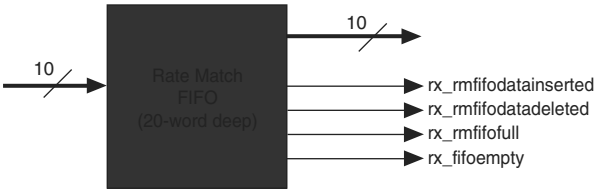
To configure a single device in a JTAG chain, the programming software places all other devices in bypass mode. In bypass mode, devices pass programming data from the `TDI` pin to the `TDO` pin through a single bypass register without being affected internally. This scheme enables the programming software to program or verify the target device. Configuration data driven into the device appears on the `TDO` pin one clock cycle later.

The Quartus II software verifies successful JTAG configuration after completion. At the end of configuration, the software checks the state of `CONF_DONE` through the JTAG port. When Quartus II generates a **.jam** for a multi-device chain, it contains instructions so that all the devices in the chain are initialized at the same time. If `CONF_DONE` is not high, the Quartus II software indicates that configuration has failed. If `CONF_DONE` is high, the software indicates that configuration was successful. After the configuration bitstream is serially sent using the JTAG `TDI` port, the `TCK` port clocks an additional clock cycles to perform device initialization.

Rate Match FIFO

In asynchronous systems, the upstream transmitter and local receiver can be clocked with independent reference clocks. Frequency differences in the order of a few hundred ppm can corrupt the data when latching from the recovered clock domain (the same clock domain as the upstream transmitter reference clock) to the local receiver reference clock domain. Figure 1–21 shows the rate match FIFO block diagram.

Figure 1–21. Rate Match FIFO Block Diagram



The rate match FIFO compensates for small clock frequency differences of up to ±300 ppm (600 ppm total) between the upstream transmitter and the local receiver clocks by performing the following functions:

- Insert skip symbols when the local receiver reference clock frequency is greater than the upstream transmitter reference clock frequency
- Delete skip symbols when the local receiver reference clock frequency is less than the upstream transmitter reference clock frequency

The 20-word deep rate match FIFO and logics control insertion and deletion of skip symbols, depending on the ppm difference. The operation begins after the word aligner synchronization status (`rx_syncstatus`) is asserted.

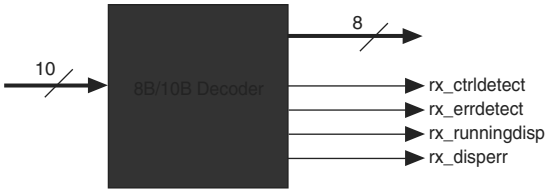
☞ Rate match FIFO is only supported with 8B/10B encoded data and the word aligner in automatic synchronization state machine mode.

8B/10B Decoder

The 8B/10B decoder receives 10-bit data and decodes it into an 8-bit data and a 1-bit control identifier. The decoder is compliant with Clause 36 of the IEEE 802.3 specification.

Figure 1–22 shows the 8B/10B decoder block diagram.

Figure 1–22. 8B/10B Decoder Block Diagram



Byte Deserializer

The byte deserializer halves the FPGA fabric-transceiver interface frequency while doubles the parallel data width to the FPGA fabric.

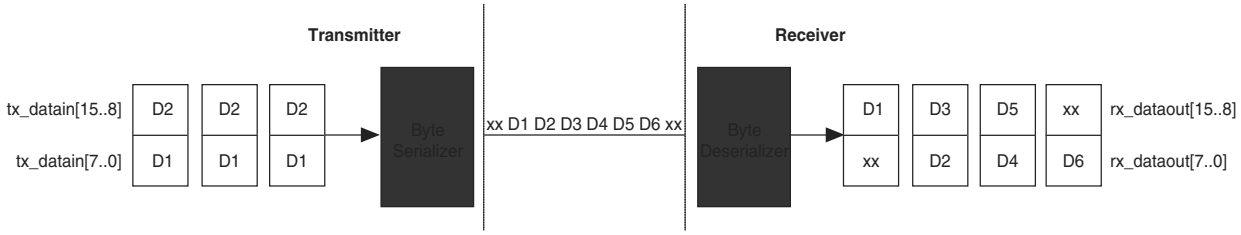
For example, when operating an EP4CGX150 receiver channel at 3.125 Gbps with deserialization factor of 10, the receiver PCS datapath runs at 312.5 MHz. The byte deserializer converts the 10-bit data at 312.5 MHz into 20-bit data at 156.25 MHz before forwarding the data to the FPGA fabric.

Byte Ordering

In the 16- or 20-bit FPGA fabric-transceiver interface, the byte deserializer receives one data byte (8 or 10 bits) and deserializes it 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 not match the original byte ordering of the transmitted data. The byte misalignment resulting from byte deserialization is unpredictable because it depends on which byte is being received by the byte deserializer when it comes out of reset.

Figure 1–23 shows a scenario where the most significant byte and the least significant byte of the two-byte transmitter data appears straddled across two word boundaries after the data is deserialized at the receiver.

Figure 1–23. Example of Byte Deserializer at the Receiver



The byte ordering block restores the proper byte ordering by performing the following actions:

- Look for the user-programmed byte ordering pattern in the byte-deserialized data
- Inserts a user-programmed pad byte if the user-programmed byte ordering pattern is found in the most significant byte position

You must select a byte ordering pattern that you know appears at the least significant byte position of the parallel transmitter data.

The byte ordering block is supported in the following receiver configurations:

- 16-bit FPGA fabric-transceiver interface, 8B/10B disabled, and the word aligner in manual alignment mode. Program a custom 8-bit byte ordering pattern and 8-bit pad byte.
- 16-bit FPGA fabric-transceiver interface, 8B/10B enabled, and the word aligner in automatic synchronization state machine mode. Program a custom 9-bit byte ordering pattern and 9-bit pad byte. The MSB of the 9-bit byte ordering pattern and pad byte represents the control identifier of the 8B/10B decoded data.

Table 1–11. FPGA Fabric-Transceiver Interface Clocks (Part 2 of 2)

Clock Name	Clock Description	Interface Direction
cal_blk_clk (2)	Transceiver calibration block clock	FPGA fabric to transceiver

Notes to Table 1–11:

(1) Offset cancellation process that is executed after power cycle requires `reconfig_clk` clock. The `reconfig_clk` must be driven with a free-running clock and not derived from the transceiver blocks.

(2) For the supported clock frequency range, refer to the *Cyclone IV Device Data Sheet*.

In the transmitter datapath, TX phase compensation FIFO forms the FPGA fabric-transmitter interface. Data and control signals for the transmitter are clocked with the FIFO write clock. The FIFO write clock supports automatic clock selection by the Quartus II software (depending on channel configuration), or user-specified clock from `tx_coreclk` port. Table 1–12 details the automatic TX phase compensation FIFO write clock selection by the Quartus II software.

The Quartus II software assumes automatic clock selection for TX phase compensation FIFO write clock if you do not enable the `tx_coreclk` port.

Table 1–12. Automatic TX Phase Compensation FIFO Write Clock Selection

Channel Configuration	Quartus II Selection
Non-bonded	`tx_clkout` clock feeds the FIFO write clock. `tx_clkout` is forwarded through the transmitter channel from low-speed clock, which also feeds the FIFO read clock.
Bonded	`coreclkout` clock feeds the FIFO write clock for the bonded channels. `coreclkout` clock is the common bonded low-speed clock, which also feeds the FIFO read clock in the bonded channels.

When using user-specified clock option, ensure that the clock feeding `tx_coreclk` port has 0 ppm difference with the TX phase compensation FIFO read clock.

In the receiver datapath, RX phase compensation FIFO forms the receiver-FPGA fabric interface. Data and status signals from the receiver are clocked with the FIFO read clock. The FIFO read clock supports automatic clock selection by the Quartus II software (depending on channel configuration), or user-specified clock from `rx_coreclk` port. Table 1–13 details the automatic RX phase compensation FIFO read clock selection by the Quartus II software.

The Quartus II software assumes automatic clock selection for RX phase compensation FIFO read clock if you do not enable the `rx_coreclk` port.

Table 1–13. Automatic RX Phase Compensation FIFO Read Clock Selection (Part 1 of 2)

Channel Configuration		Quartus II Selection
Non-bonded	With rate match FIFO (1)	`tx_clkout` clock feeds the FIFO read clock. `tx_clkout` is forwarded through the receiver channel from low-speed clock, which also feeds the FIFO write clock and transmitter PCS.
	Without rate match FIFO	`rx_clkout` clock feeds the FIFO read clock. `rx_clkout` is forwarded through the receiver channel from low-speed recovered clock, which also feeds the FIFO write clock.

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

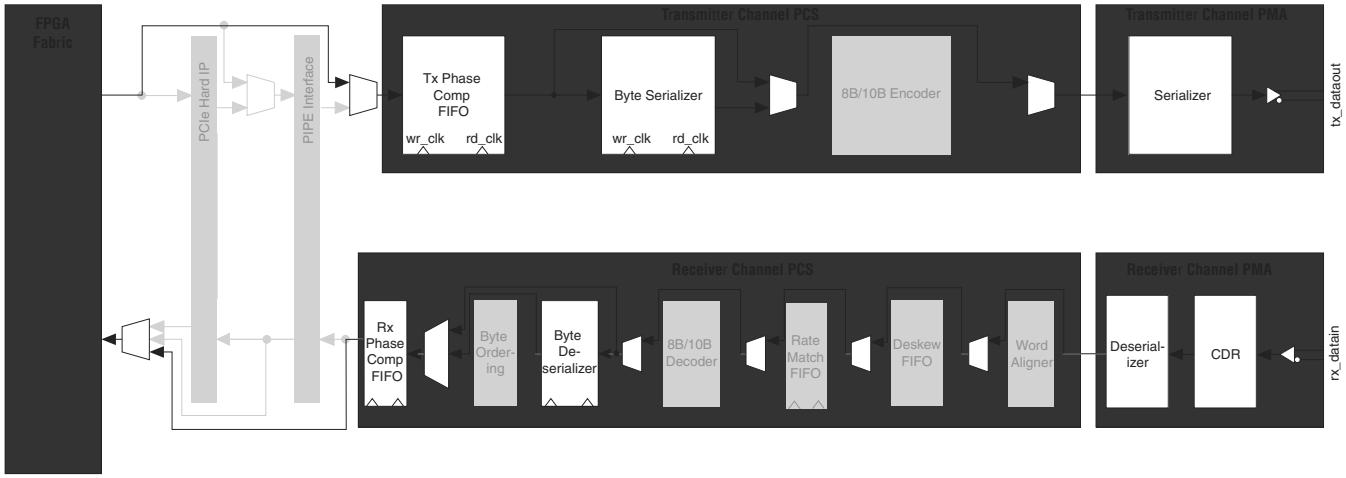
Low-Latency PCS Operation

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.

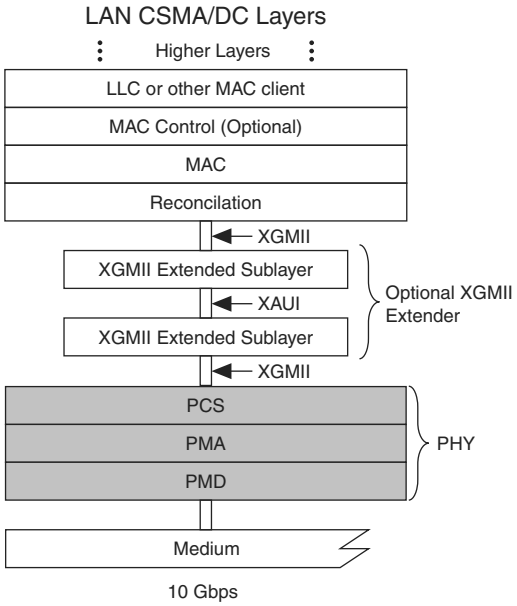
☞ The transmitter in electrical idle feature is required for compliance to the version 2.00 of PHY Interface for the PCI Express (PIPE) Architecture specification for PCIe protocol implementation.

Signal Detect at Receiver

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

converted within the XGMII extender sublayer into an 8B/10B encoded data stream. Each data stream is then transmitted across a single differential pair running at 3.125 Gbps. At the XAUI receiver, the incoming data is decoded and mapped back to the 32-bit XGMII format. This provides a transparent extension of the physical reach of the XGMII and also reduces the interface pin count.

Figure 1–62. XAUI in 10 Gbps LAN Layers

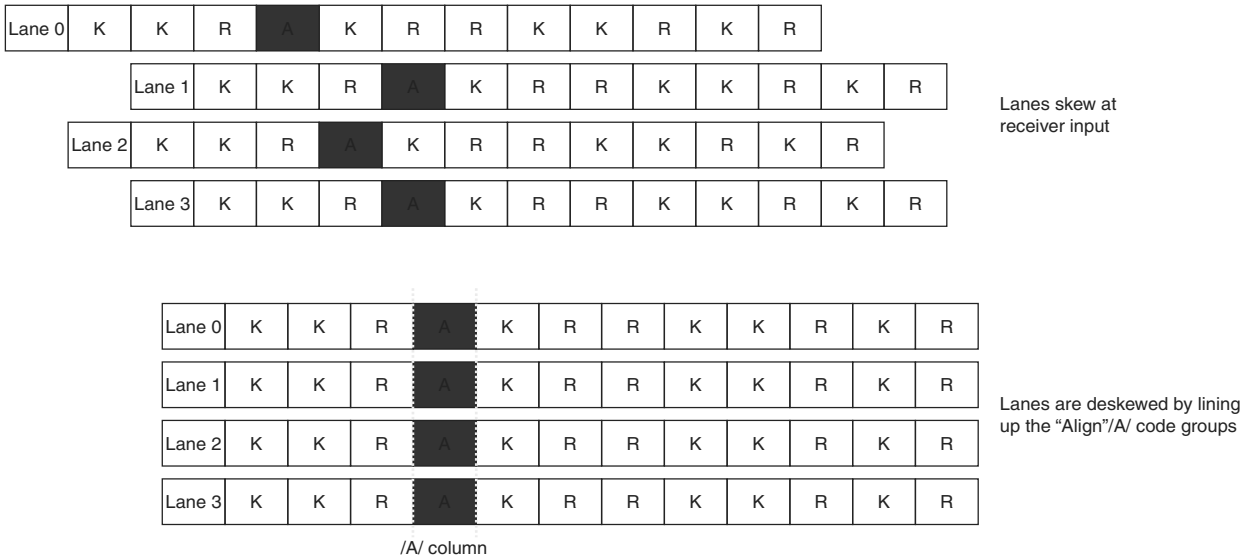


XAUI functions as a self-managed interface because code group synchronization, channel deskew, and clock domain decoupling is handled with no upper layer support requirements. This functionality is based on the PCS code groups that are used during the inter-packet gap time and idle periods.

- 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.