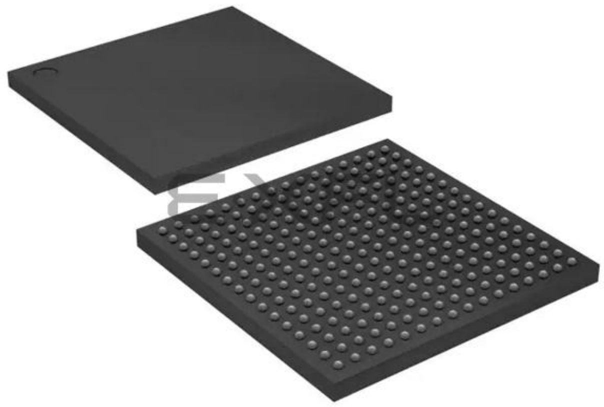
Intel - EP4CE10F17I8LN Datasheet



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	645
Number of Logic Elements/Cells	10320
Total RAM Bits	423936
Number of I/O	179
Number of Gates	-
Voltage - Supply	0.97V ~ 1.03V
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/ep4ce10f17i8ln

Chapter Revision Dates

The chapters in this document, Cyclone IV Device Handbook,, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1. Cyclone IV FPGA Device Family Overview
Revised: *March 2016*
Part Number: *CYIV-51001-2.0*

Chapter 2. Logic Elements and Logic Array Blocks in Cyclone IV Devices
Revised: *November 2009*
Part Number: *CYIV-51002-1.0*

Chapter 3. Memory Blocks in Cyclone IV Devices
Revised: *November 2011*
Part Number: *CYIV-51003-1.1*

Chapter 4. Embedded Multipliers in Cyclone IV Devices
Revised: *February 2010*
Part Number: *CYIV-51004-1.1*

Chapter 5. Clock Networks and PLLs in Cyclone IV Devices
Revised: *October 2012*
Part Number: *CYIV-51005-2.4*

Chapter 6. I/O Features in Cyclone IV Devices
Revised: *March 2016*
Part Number: *CYIV-51006-2.7*

Chapter 7. External Memory Interfaces in Cyclone IV Devices
Revised: *March 2016*
Part Number: *CYIV-51007-2.6*

Chapter 8. Configuration and Remote System Upgrades in Cyclone IV Devices
Revised: *May 2013*
Part Number: *CYIV-51008-1.7*

Chapter 9. SEU Mitigation in Cyclone IV Devices
Revised: *May 2013*
Part Number: *CYIV-51009-1.3*

Chapter 10. JTAG Boundary-Scan Testing for Cyclone IV Devices
Revised: *December 2013*
Part Number: *CYIV-51010-1.3*

Chapter 11. Power Requirements for Cyclone IV Devices
Revised: *May 2013*
Part Number: *CYIV-51011-1.3*

Clocking Modes

Cyclone IV devices M9K memory blocks support the following clocking modes:

- Independent
- Input or output
- Read or write
- Single-clock

When using read or write clock mode, if you perform a simultaneous read or write to the same address location, the output read data is unknown. If you require the output data to be a known value, use either single-clock mode or I/O clock mode and choose the appropriate read-during-write behavior in the MegaWizard Plug-In Manager.

☞ Violating the setup or hold time on the memory block input registers might corrupt the memory contents. This applies to both read and write operations.

☞ Asynchronous clears are available on read address registers, output registers, and output latches only.

Table 3–5 lists the clocking mode versus memory mode support matrix.

Table 3–5. Cyclone IV Devices Memory Clock Modes

Clocking Mode	True Dual-Port Mode	Simple Dual-Port Mode	Single-Port Mode	ROM Mode	FIFO Mode
Independent	✓	—	—	✓	—
Input or output	✓	✓	✓	✓	—
Read or write	—	✓	—	—	✓
Single-clock	✓	✓	✓	✓	✓

Independent Clock Mode

Cyclone IV devices M9K memory blocks can implement independent clock mode for true dual-port memories. In this mode, a separate clock is available for each port (port A and port B). `clock A` controls all registers on the port A side, while `clock B` controls all registers on the port B side. Each port also supports independent clock enables for port A and B registers.

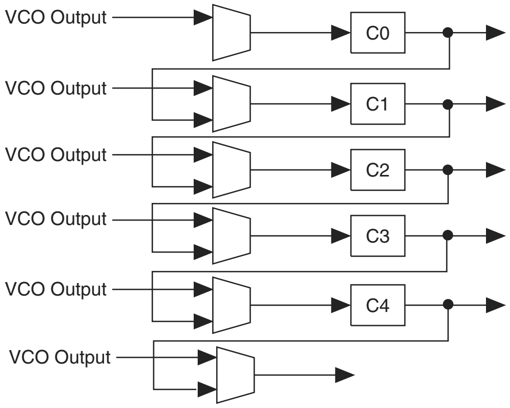
Input or Output Clock Mode

Cyclone IV devices M9K memory blocks can implement input or output clock mode for FIFO, single-port, true, and simple dual-port memories. In this mode, an input clock controls all input registers to the memory block including data, address, `byteena`, `wren`, and `rden` registers. An output clock controls the data-output registers. Each memory block port also supports independent clock enables for input and output registers.

Post-Scale Counter Cascading

PLLs of Cyclone IV devices support post-scale counter cascading to create counters larger than 512. This is implemented by feeding the output of one C counter into the input of the next C counter, as shown in Figure 5–16.

Figure 5–16. Counter Cascading



When cascading counters to implement a larger division of the high-frequency VCO clock, the cascaded counters behave as one counter with the product of the individual counter settings.

For example, if C0 = 4 and C1 = 2, the cascaded value is C0 × C1 = 8.

☞ Post-scale counter cascading is automatically set by the Quartus II software in the configuration file. Post-scale counter cascading cannot be performed using the PLL reconfiguration.

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. You can achieve the duty cycle setting by a low and high time count setting for the post-scale counters. The Quartus II software uses the frequency input and the required multiply or divide rate to determine the duty cycle choices. The post-scale counter value determines the precision of the duty cycle. The precision is defined by 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 between 5 to 90%.

Combining the programmable duty cycle with programmable phase shift allows the generation of precise non-overlapping clocks.

PLL Control Signals

You can use the `pfdena`, `areset`, and `locked` signals to observe and control the PLL operation and resynchronization.

For more information about the PLL control signals, refer to the *ALTPLL Megafunction User Guide*.

The IOE registers in each I/O block share the same source for the preset or clear features. You can program preset or clear for each individual IOE, but you cannot use both features simultaneously. You can also program the registers to power-up high or low after configuration is complete. If programmed to power-up low, an asynchronous clear can control the registers. If programmed to power-up high, an asynchronous preset can control the registers. This feature prevents the inadvertent activation of the active-low input of another device upon power-up. If one register in an IOE uses a preset or clear signal, all registers in the IOE must use that same signal if they require preset or clear. Additionally, a synchronous reset signal is available for the IOE registers.

For more information about the input and output pin delay settings, refer to the *Area and Timing Optimization* chapter in volume 2 of the *Quartus II Handbook*.

PCI-Clamp Diode

Cyclone IV devices provide an optional PCI-clamp diode enabled input and output for each I/O pin. Dual-purpose configuration pins support the diode in user mode if the specific pins are not used as configuration pins for the selected configuration scheme. For example, if you are using the active serial (AS) configuration scheme, you cannot use the clamp diode on the `ASDO` and `nCSO` pins in user mode. Dedicated configuration pins do not support the on-chip diode.

The PCI-clamp diode is available for the following I/O standards:

- 3.3-V LVTTL
- 3.3-V LVCMOS
- 3.0-V LVTTL
- 3.0-V LVCMOS
- 2.5-V LVTTL/LVCMOS
- PCI
- PCI-X

If the input I/O standard is one of the listed standards, the PCI-clamp diode is enabled by default in the Quartus II software.

OCT Support

Cyclone IV devices feature OCT to provide I/O impedance matching and termination capabilities. OCT helps prevent reflections and maintain signal integrity while minimizing the need for external resistors in high pin-count ball grid array (BGA) packages. Cyclone IV devices provide I/O driver on-chip impedance matching and R_S OCT for single-ended outputs and bidirectional pins.

When using R_S OCT, programmable current strength is not available.

There are two ways to implement OCT in Cyclone IV devices:

- OCT with calibration
- OCT without calibration

External Memory Interfacing

Cyclone IV devices support I/O standards required to interface with a broad range of external memory interfaces, such as DDR SDRAM, DDR2 SDRAM, and QDR II SRAM.

For more information about Cyclone IV devices external memory interface support, refer to the *External Memory Interfaces in Cyclone IV Devices* chapter.

Pad Placement and DC Guidelines

You can use the Quartus II software to validate your pad and pin placement.

Pad Placement

Altera recommends that you create a Quartus II design, enter your device I/O assignments and compile your design to validate your pin placement. The Quartus II software checks your pin connections with respect to the I/O assignment and placement rules to ensure proper device operation. These rules depend on device density, package, I/O assignments, voltage assignments and other factors that are not fully described in this chapter.

For more information about how the Quartus II software checks I/O restrictions, refer to the *I/O Management* chapter in volume 2 of the *Quartus II Handbook*.

DC Guidelines

For the Quartus II software to automatically check for illegally placed pads according to the DC guidelines, set the DC current sink or source value to **Electromigration Current** assignment on each of the output pins that are connected to the external resistive load.

The programmable current strength setting has an impact on the amount of DC current that an output pin can source or sink. Determine if the current strength setting is sufficient for the external resistive load condition on the output pin.

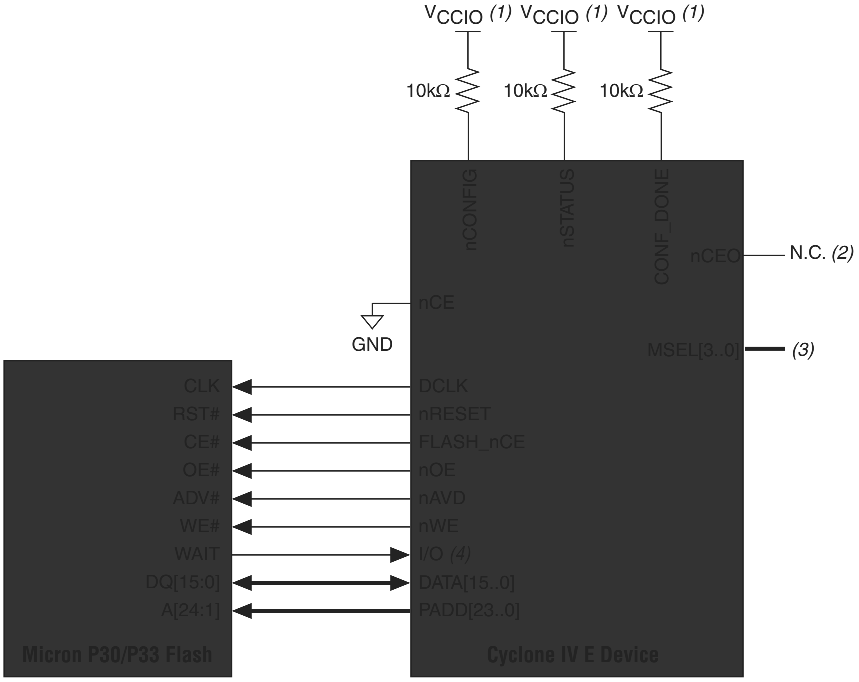
Clock Pins Functionality

Cyclone IV clock pins have multiple purposes, as per listed:

- `CLK` pins—Input support for single-ended and voltage-referenced standards. For I/O standard support, refer to Table 6–3 on page 6–11.
- `DIFFCLK` pins—Input support for differential standards. For I/O standard support, refer to Table 6–3 on page 6–11. When used as `DIFFCLK` pins, DC or AC coupling can be used depending on the interface requirements and external termination is required. For more information, refer to “High-Speed I/O Standards Support” on page 6–28.
- `REFCLK` pins—Input support for high speed differential reference clocks used by the transceivers in Cyclone IV GX devices. For I/O support, coupling, and termination requirements, refer to Table 6–10 on page 6–29.

Figure 8–7 shows the interface for the Micron P30 flash memory and P33 flash memory to the Cyclone IV E device pins.

Figure 8–7. Single-Device AP Configuration Using Micron P30 and P33 Flash Memory



Notes to Figure 8–7:

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

(2) 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.

(3) The `MSEL` pin settings vary for different configuration voltage standards and POR time. To connect `MSEL[3..0]`, refer to Table 8–5 on page 8–9. Connect the `MSEL` pins directly to V_{CCA} or GND.

(4) AP configuration ignores the `WAIT` signal during configuration mode. However, if you are accessing flash during user mode with user logic, you can optionally use normal I/O to monitor the `WAIT` signal from the Micron P30 or P33 flash.

☞ To tri-state the configuration bus for AP configuration schemes, you must tie `nCE` high and `nCONFIG` low.

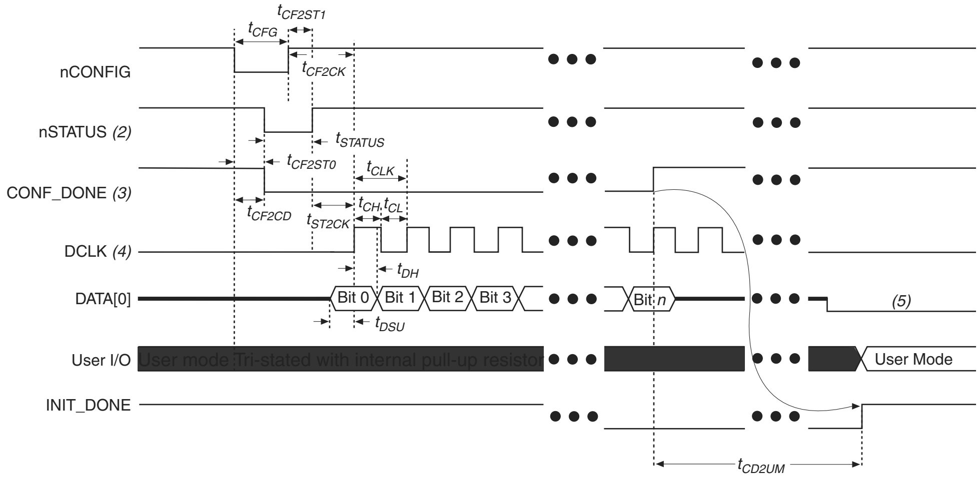
☞ In a single-device AP configuration, the maximum board loading and board trace length between supported parallel flash and Cyclone IV E devices must follow the recommendations listed in Table 8–11 on page 8–28.

☞ If you use the AP configuration scheme for Cyclone IV E devices, the V_{CCIO} of I/O banks 1, 6, 7, and 8 must be 3.3, 3.0, 2.5, or 1.8 V. Altera does not recommend using the level shifter between the Micron P30 or P33 flash and the Cyclone IV E device in the AP configuration scheme.

PS Configuration Timing

A PS configuration must meet the setup and hold timing parameters and the maximum clock frequency. When using a microprocessor or another intelligent host to control the PS interface, ensure that you meet these timing requirements. Figure 8–16 shows the timing waveform for PS configuration when using an external host device.

Figure 8–16. PS Configuration Timing Waveform *(1)*



Notes to Figure 8–16:

(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) In user mode, drive `DCLK` either high or low when using the PS configuration scheme, whichever is more convenient. When using the AS configuration scheme, `DCLK` is a Cyclone IV device output pin and must not be driven externally.

(5) Do not leave the `DATA[0]` pin floating after configuration. Drive the `DATA[0]` pin high or low, whichever is more convenient.

Table 8–12 lists the PS configuration timing parameters for Cyclone IV devices.

Table 8–12. PS Configuration 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

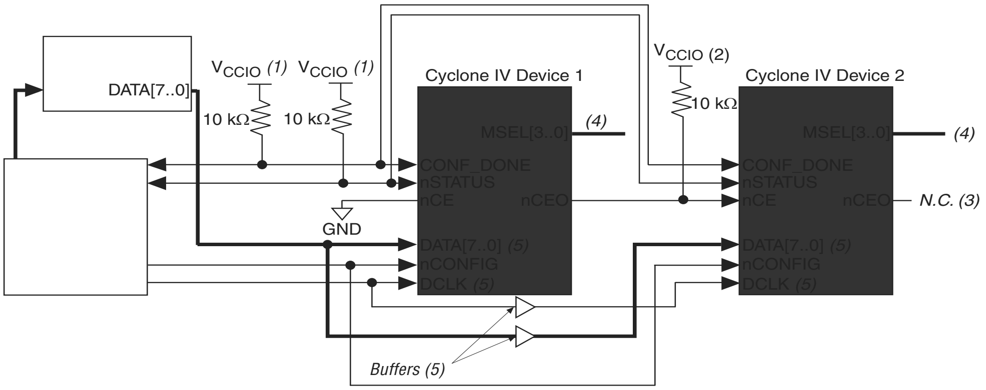
To ensure that `DCLK` and `DATA[0]` are not left floating at the end of the configuration, the MAX II device must drive them either high or low, whichever is convenient on your board. The `DATA[0]` pin is available as a user I/O pin after configuration. When you choose the FPP scheme in the Quartus II 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`,

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

Symbol	Parameter	Minimum		Maximum		Unit
		Cyclone IV *(1)*	Cyclone IV E *(2)*	Cyclone IV *(1)*	Cyclone IV E *(2)*	
t_{ST2CK}	`nSTATUS` high to first rising edge of `DCLK`	2		—		μs
t_{DH}	Data hold time after rising edge on `DCLK`	0		—		ns
t_{CD2UM}	`CONF_DONE` high to user mode *(5)*	300		650		μs
t_{CD2CU}	`CONF_DONE` high to `CLKUSR` enabled	4 × maximum `DCLK` period		—		—
t_{CD2UMC}	`CONF_DONE` high to user mode with **CLKUSR** option on	t_{CD2CU} + (3,192 × `CLKUSR` period)		—		—
t_{DSU}	Data setup time before rising edge on `DCLK`	5	8	—	—	ns
t_{CH}	`DCLK` high time	3.2	6.4	—	—	ns
t_{CL}	`DCLK` low time	3.2	6.4	—	—	ns
t_{CLK}	`DCLK` period	7.5	15	—	—	ns
f_{MAX}	`DCLK` frequency *(6)*	—	—	133	66	MHz

Notes to Table 8–13:

(1) Applicable for Cyclone IV GX and Cyclone IV E with 1.2-V core voltage.

(2) Applicable for Cyclone IV E with 1.0-V core voltage.

(3) This value is applicable if you do not delay configuration by extending the `nCONFIG` or `nSTATUS` low pulse width.

(4) This value is applicable if you do not delay configuration by externally holding the `nSTATUS` low.

(5) The minimum and maximum numbers apply only if you choose the internal oscillator as the clock source for starting the device.

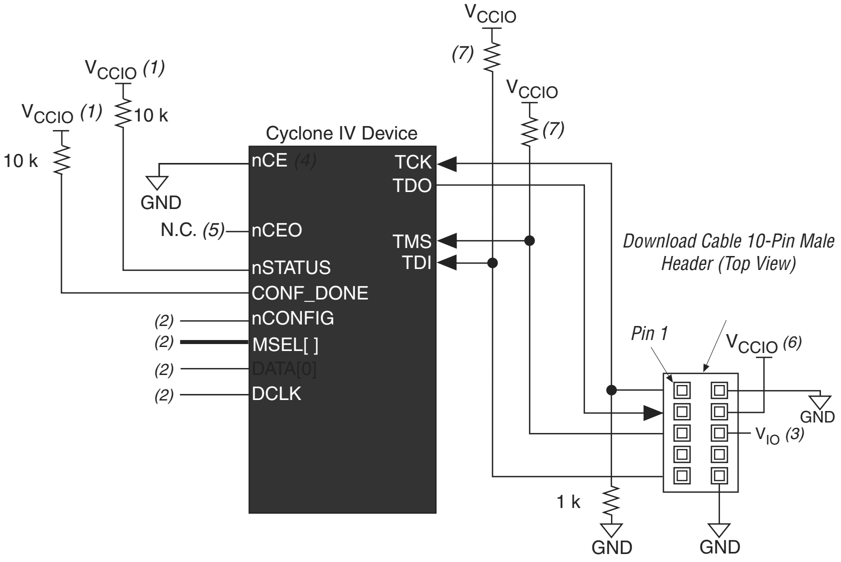
(6) Cyclone IV E devices with 1.0-V core voltage have slower F_{MAX} when compared with Cyclone IV GX devices with 1.2-V core voltage.

JTAG Configuration

JTAG has developed a specification for boundary-scan testing (BST). The BST architecture offers the capability to efficiently test components on PCBs with tight lead spacing. The BST architecture can test pin connections without using physical test probes and capture functional data while a device is normally operating. You can also use the JTAG circuitry to shift configuration data into the device. The Quartus II software automatically generates **.sof** for JTAG configuration with a download cable in the Quartus II software Programmer.

For more information about the JTAG boundary-scan testing, refer to the *JTAG Boundary-Scan Testing for Cyclone IV Devices* chapter.

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.

Use the `ACTIVE_DISENGAGE` instruction with the `CONFIG_IO` instruction to interrupt configuration. Table 8–16 lists the sequence of instructions to use for various `CONFIG_IO` usage scenarios.

Table 8–16. JTAG CONFIG_IO (without JTAG_PROGRAM) Instruction Flows *(1)*

JTAG Instruction	Configuration Scheme and Current State of the Cyclone IV Device											
	Prior to User Mode (Interrupting Configuration)				User Mode				Power Up			
	PS	FPP	AS	AP	PS	FPP	AS	AP	PS	FPP	AS	AP
`ACTIVE_DISENGAGE`	O	O	O	O	O	O	O	O	—	—	—	—
`CONFIG_IO`	R	R	R	R	R	R	R	R	NA	NA	NA	NA
JTAG Boundary Scan Instructions (no `JTAG_PROGRAM`)	O	O	O	O	O	O	O	O	—	—	—	—
`ACTIVE_ENGAGE`	A	A	R *(2)*	R *(2)*	A	A	R *(2)*	R *(2)*	—	—	—	—
`PULSE_NCONFIG`			A *(3)*	A *(3)*			O	O	—	—	—	—
Pulse `nCONFIG` pin			A *(3)*	A *(3)*			O	O	—	—	—	—
JTAG TAP Reset	R	R	R	R	R	R	R	R	—	—	—	—

Notes to Table 8–16:

(1) You must execute "R" indicates that the instruction before the next instruction, "O" indicates the optional instruction, "A" indicates that the instruction must be executed, and "NA" indicates that the instruction is not allowed in this mode.

(2) Required if you use `ACTIVE_DISENGAGE`.

(3) Neither of the instruction is required if you use `ACTIVE_ENGAGE`.

The `CONFIG_IO` instruction does not hold `nSTATUS` low until reconfiguration. You must disengage the AS or AP configuration controller by issuing the `ACTIVE_DISENGAGE` and `ACTIVE_ENGAGE` instructions when active configuration is interrupted. You must issue the `ACTIVE_DISENGAGE` instruction alone or prior to the `CONFIG_IO` instruction if the `JTAG_PROGRAM` instruction is to be issued later (Table 8–17). This puts the active configuration controllers into the idle state. The active configuration controller is re-engaged after user mode is reached through JTAG programming (Table 8–17).

While executing the `CONFIG_IO` instruction, all user I/Os are tri-stated.

If reconfiguration after interruption is performed using configuration modes (rather than using `JTAG_PROGRAM`), it is not necessary to issue the `ACTIVE_DISENGAGE` instruction prior to `CONFIG_IO`. You can start reconfiguration by either pulling `nCONFIG` low for at least 500 ns or issuing the `PULSE_NCONFIG` instruction. If the `ACTIVE_DISENGAGE` instruction was issued and the `JTAG_PROGRAM` instruction fails to enter user mode, you must issue the `ACTIVE_ENGAGE` instruction to reactivate the active configuration controller. Issuing the `ACTIVE_ENGAGE` instruction also triggers reconfiguration in configuration modes; therefore, it is not necessary to pull `nCONFIG` low or issue the `PULSE_NCONFIG` instruction.

Table 9–4 defines the registers shown in Figure 9–1.

Table 9–4. Error Detection Registers

Register	Function
32-bit signature register	This register contains the CRC signature. The signature register contains the result of the user mode calculated CRC value compared against the pre-calculated CRC value. If no errors are detected, the signature register is all zeros. A non-zero signature register indicates an error in the configuration CRAM contents. The `CRC_ERROR` signal is derived from the contents of this register.
32-bit storage register	This register is loaded with the 32-bit pre-computed CRC signature at the end of the configuration stage. The signature is then loaded into the 32-bit CRC circuit (called the Compute and Compare CRC block, as shown in Figure 9–1) during user mode to calculate the CRC error. This register forms a 32-bit scan chain during execution of the `CHANGE_EDREG` JTAG instruction. The `CHANGE_EDREG` JTAG instruction can change the content of the storage register. Therefore, the functionality of the error detection CRC circuitry is checked in-system by executing the instruction to inject an error during the operation. The operation of the device is not halted when issuing the `CHANGE_EDREG` instruction.

Error Detection Timing

When the error detection CRC feature is enabled through the Quartus II software, the device automatically activates the CRC process upon entering user mode after configuration and initialization is complete.

The `CRC_ERROR` pin is driven low until the error detection circuitry detects a corrupted bit in the previous CRC calculation. After the pin goes high, it remains high during the next CRC calculation. This pin does not log the previous CRC calculation. If the new CRC calculation does not contain any corrupted bits, the `CRC_ERROR` pin is driven low. The error detection runs until the device is reset.

The error detection circuitry runs off an internal configuration oscillator with a divisor that sets the maximum frequency.

Table 9–5 lists the minimum and maximum error detection frequencies.

Table 9–5. Minimum and Maximum Error Detection Frequencies for Cyclone IV Devices

Error Detection Frequency	Maximum Error Detection Frequency	Minimum Error Detection Frequency	Valid Divisors (2^n)
80 MHz/2^n	80 MHz	312.5 kHz	0, 1, 2, 3, 4, 5, 6, 7, 8

You can set a lower clock frequency by specifying a division factor in the Quartus II software (for more information, refer to "Software Support"). The divisor is a power of two (2), where *n* is between 0 and 8. The divisor ranges from one through 256. Refer to Equation 9–1.

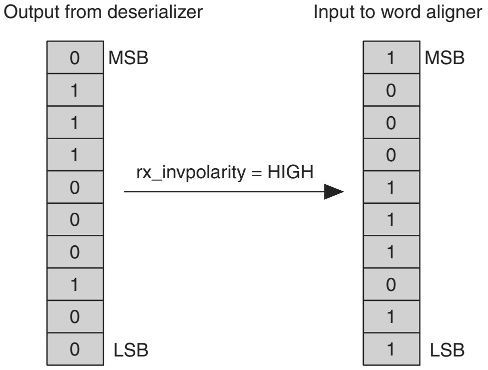
Equation 9–1.

$$\text{ror detection frequency} = \frac{80\text{ MH}}{2^n}$$

CRC calculation time depends on the device and the error detection clock frequency.

- Receiver polarity inversion—corrects accidental swapped positive and negative signals from the serial differential link during board layout. This feature works by inverting the polarity of every bit of the input data word to the word aligner, which has the same effect as swapping the positive and negative signals of the differential link. Inversion is dynamically controlled using `rx_invpolarity` port. Figure 1–19 shows the receiver polarity inversion feature.

Figure 1–19. Receiver Polarity Inversion



The generic receiver polarity inversion feature is different from the PCI Express (PIPE) 8B/10B polarity inversion feature. The generic receiver polarity inversion feature inverts the polarity of the data bits at the input of the word aligner and is not available in PCI Express (PIPE) mode. The PCI Express (PIPE) 8B/10B polarity inversion feature inverts the polarity of the data bits at the input of the 8B/10B decoder and is available only in PCI Express (PIPE) mode.

☞ The `rx_invpolarity` signal is dynamic and might cause initial disparity errors in an 8B/10B encoded link. The downstream system must be able to tolerate these disparity errors.

- Receiver bit reversal—by default, the Cyclone IV GX receiver assumes LSB to MSB transmission. If the link transmission order is MSB to LSB, the receiver forwards the incorrect reverse bit-ordered version of the parallel data to the FPGA fabric on the `rx_dataout` port. The receiver bit reversal feature is available to correct this situation. This feature is static in manual alignment and automatic

Clock Frequency Compensation

In GIGE mode, the rate match FIFO compensates up to ±100 ppm (200 ppm total) difference between the upstream transmitter and the local receiver reference clock. The GIGE protocol requires 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.

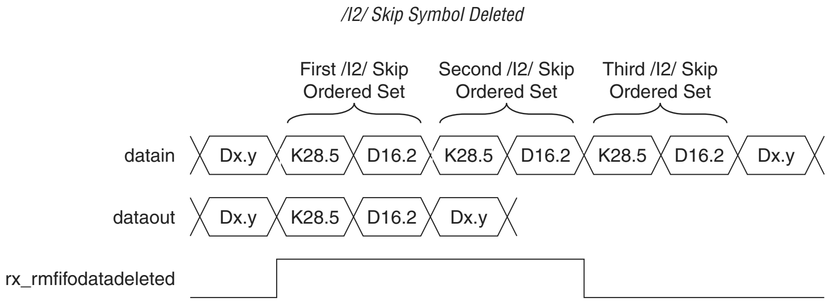
The rate match operation begins after the synchronization state machine in the word aligner indicates synchronization has been acquired by driving the `rx_syncstatus` signal high. The rate match FIFO deletes or inserts both symbols of the /I2/ ordered sets (/K28.5/ and /D16.2/) to prevent the rate match FIFO from overflowing or underflowing. It can insert or delete as many /I2/ ordered sets as necessary to perform the rate match operation.

☞ If you have the auto-negotiation 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 auto-negotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the auto-negotiation link to fail. For more information, refer to the Altera Knowledge Base Support Solution.

The status flags `rx_rmfifodatadeleted` and `rx_rmfifodatainserted` to indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric. These two flags are asserted for two clock cycles for each deleted and inserted /I2/ ordered set.

Figure 1–58 shows an example of rate match FIFO deletion where three symbols must be deleted. Because the rate match FIFO can only delete /I2/ ordered sets, it deletes two /I2/ ordered sets (four symbols deleted).

Figure 1–58. Example of Rate Match FIFO Deletion in GIGE Mode



Clock Frequency Compensation

In Serial RapidIO mode, the rate match FIFO compensates up to ±100 ppm (200 ppm total) difference between the upstream transmitter and the local receiver reference clock.

Rate matcher is an optional block available for selection in Serial RapidIO mode. However, this block is not fully compliant to the SRIO specification. When enabled in the ALTGX MegaWizard Plug-In Manager, the default settings are:

- control pattern 1 = K28.5 with positive disparity
- skip pattern 1 = K29.7 with positive disparity
- control pattern 2 = K28.5 with negative disparity
- skip pattern 2 = K29.7 with negative disparity

When enabled, the rate match FIFO operation begins after the link is synchronized (indicated by assertion of `rx_syncstatus` from the word aligner). When the rate matcher receives either of the two 10-bit control patterns followed by the respective 10-bit skip pattern, it inserts or deletes the 10-bit skip pattern as necessary to avoid the rate match FIFO from overflowing or under-running. The rate match FIFO can delete/insert a maximum of one skip pattern from a cluster.

☞ The rate match FIFO may perform multiple insertion or deletion if the ppm difference is more than the allowable 200 ppm range. Ensure that the ppm difference in your system is less than 200 ppm.

XAUI Mode

XAUI mode provides the bonded (×4) transceiver channel datapath configuration for XAUI protocol implementation. The Cyclone IV GX transceivers configured in XAUI mode provides the following functions:

- XGMII-to-PCS code conversion at transmitter datapath
- PCS-to-XGMII code conversion at receiver datapath
- channel deskewing of four lanes
- 8B/10B encoding and decoding
- IEEE P802.3ae-compliant synchronization state machine
- clock rate compensation

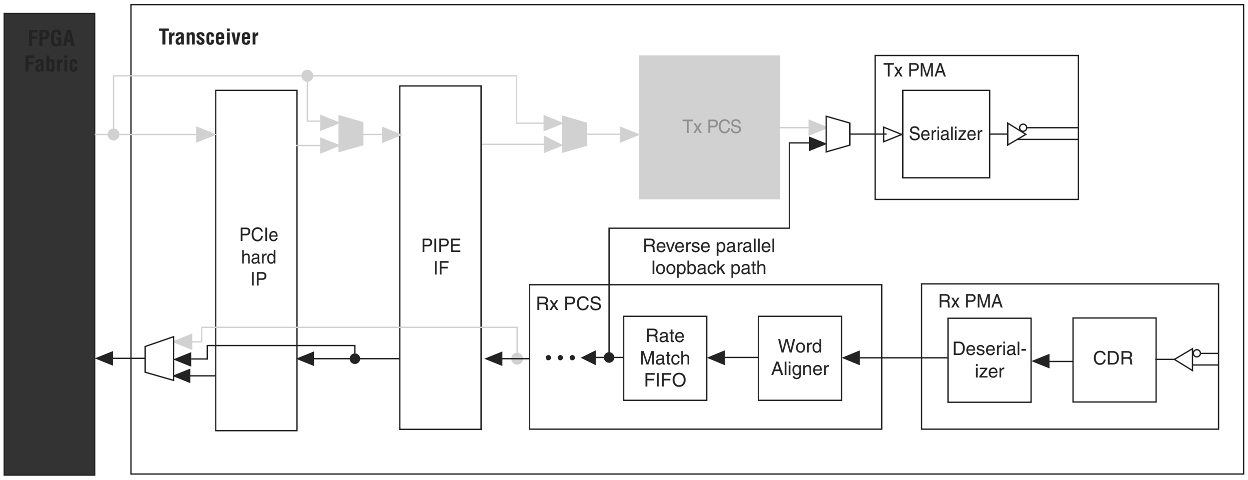
The XAUI is a self-managed interface to transparently extend the physical reach of the XGMII between the reconciliation sublayer and the PHY layer in the 10 Gbps LAN as shown in Figure 1–62. The XAUI interface consists of four lanes, each running at 3.125 Gbps with 8B/10B encoded data for a total of actual 10 Gbps data throughput. At the transmit side of the XAUI interface, the data and control characters are

Reverse Parallel Loopback

The reverse parallel loopback option is only available for PIPE mode. In this mode, the received serial data passes through the receiver CDR, deserializer, word aligner, and rate match FIFO before looping back to the transmitter serializer and transmitted out through the TX buffer, as shown in Figure 1–70. The received data is also available to the FPGA fabric. This loopback mode is compliant with version 2.00 of the *PHY Interface for the PCI Express Architecture* specification.

To enable the reverse parallel loopback mode, assert the `tx_detectrxloopback` port in P0 power state.

Figure 1–70. PIPE Reverse Parallel Loopback Path (1)



Note to Figure 1–70:

(1) Grayed-Out Blocks are Not Active in this mode.

Serial Loopback

The serial loopback option is available for all functional modes except PIPE mode. In this mode, the data from the FPGA fabric passes through the transmitter channel and looped back to the receiver channel, bypassing the receiver buffer, as shown in Figure 1–71. The received data is available to the FPGA logic for verification. The receiver input buffer is not active in this mode. With this option, you can check the operation of all enabled PCS and PMA functional blocks in the transmitter and receiver channels.

The transmitter channel sends the data to both the serial output port and the receiver channel. The differential output voltage on the serial ports is based on the selected V_{OD} settings. The data is looped back to the receiver CDR and is retimed through different clock domains. You must provide an alignment pattern for the word aligner to enable the receiver channel to retrieve the byte boundary.

Control and Status Signals for Channel Reconfiguration

The various control and status signals involved in the Channel Reconfiguration mode are as follows. Refer to “Dynamic Reconfiguration Controller Port List” on page 3–4 for the descriptions of the control and status signals.

The following are the input control signals:

- `logical_channel_address[n..0]`
- `reset_reconfig_address`
- `reconfig_reset`
- `reconfig_mode_sel[2..0]`
- `write_all`

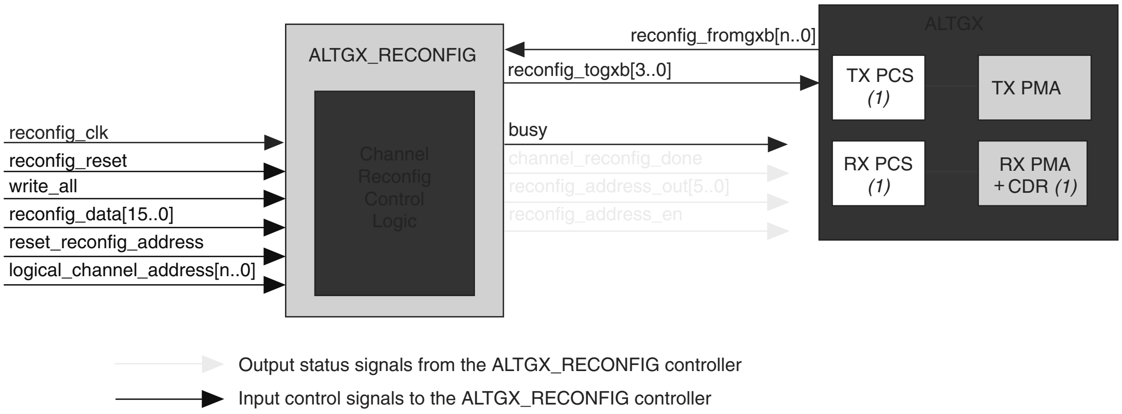
The following are output status signals:

- `reconfig_address_en`
- `reconfig_address_out[5..0]`
- `channel_reconfig_done`
- `busy`

The ALTGX_RECONFIG connection to the ALTGX instances when set in channel reconfiguration mode are as follows. For the port information, refer to “Dynamic Reconfiguration Controller Port List” on page 3–4.

Figure 3–10 shows the connection for channel reconfiguration mode.

Figure 3–10. ALTGX and ALTGX_RECONFIG Connection for Channel Reconfiguration Mode



Note to Figure 3–10:

(1) This block can be reconfigured in channel reconfiguration mode.