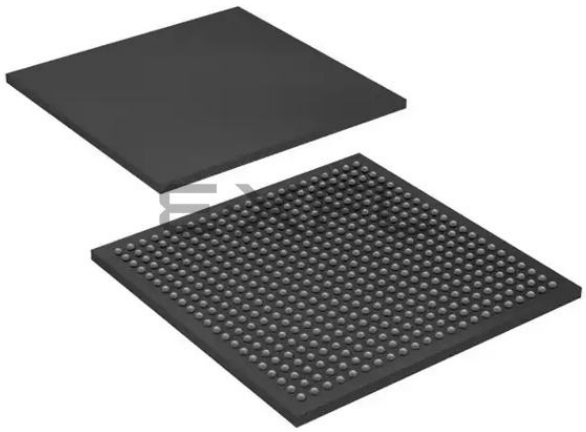


**Welcome to E-XFL.COM**

## 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 | 343 |
| Number of Gates | - |
| Voltage - Supply | 0.97V ~ 1.03V |
| Mounting Type | Surface Mount |
| Operating Temperature | 0°C ~ 85°C (TJ) |
| Package / Case | 484-BGA |
| Supplier Device Package | 484-FBGA (23x23) |
| Purchase URL | https://www.e-xfl.com/product-detail/intel/ep4ce15f23c9ln |

Figure 3–3 and Figure 3–4 show the address clock enable waveform during read and write cycles, respectively.

**Figure 3–3. Cyclone IV Devices Address Clock Enable During Read Cycle Waveform**

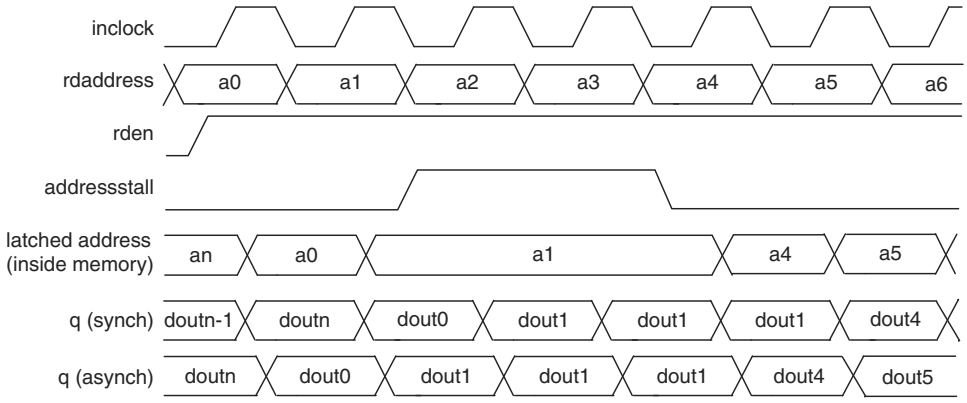
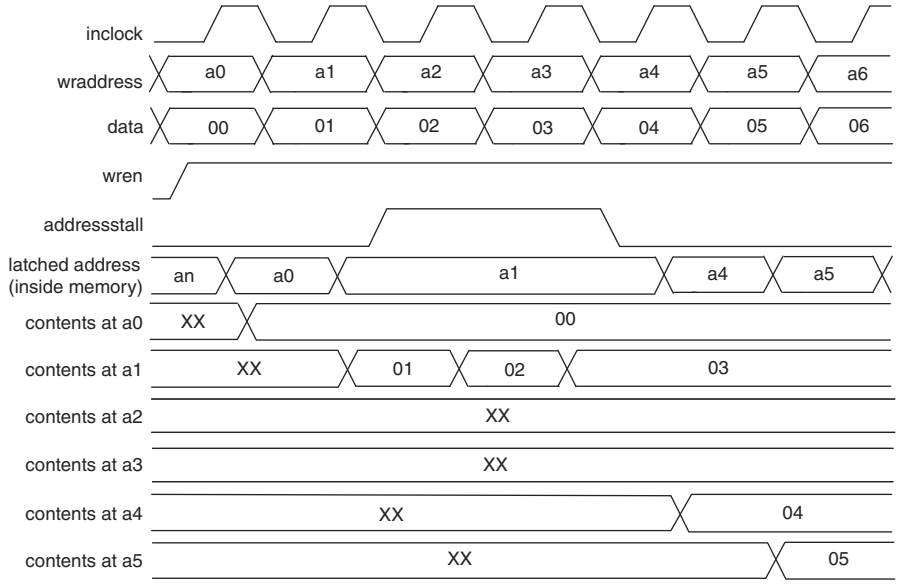



**Figure 3–4. Cyclone IV Devices Address Clock Enable During Write Cycle Waveform**




## Mixed-Width Support

M9K memory blocks support mixed data widths. When using simple dual-port, true dual-port, or FIFO modes, mixed width support allows you to read and write different data widths to an M9K memory block. For more information about the different widths supported per memory mode, refer to "Memory Modes" on page 3–7.

# Document Revision History

Table 4–3 lists the revision history for this chapter.

**Table 4–3. Document Revision History**

| Date | Version | Changes |
|---|---|---|
| February 2010 | 1.1 | Added Cyclone IV E devices in Table 4–1 for the Quartus II software version 9.1 SP1 release. |
| November 2009 | 1.0 | Initial release. |

If you do not use dedicated clock pins to feed the GCLKs, you can use them as general-purpose input pins to feed the logic array. However, when using them as general-purpose input pins, they do not have support for an I/O register and must use LE-based registers in place of an I/O register.

For more information about how to connect the clock and PLL pins, refer to the *Cyclone IV Device Family Pin Connection Guidelines*.

## Clock Control Block

The clock control block drives the GCLKs. Clock control blocks are located on each side of the device, close to the dedicated clock input pins. GCLKs are optimized for minimum clock skew and delay.

Table 5–4 lists the sources that can feed the clock control block, which in turn feeds the GCLKs.

**Table 5–4. Clock Control Block Inputs**

| Input | Description |
|---|---|
| Dedicated clock inputs | Dedicated clock input pins can drive clocks or global signals, such as synchronous and asynchronous clears, presets, or clock enables onto given GCLKs. |
| Dual-purpose clock (DPCLK and CDPCLK) I/O input | DPCLK and CDPCLK I/O pins are bidirectional dual function pins that are used for high fan-out control signals, such as protocol signals, TRDY and IRDY signals for PCI, via the GCLK. Clock control blocks that have inputs driven by dual-purpose clock I/O pins are not able to drive PLL inputs. |
| PLL outputs | PLL counter outputs can drive the GCLK. |
| Internal logic | You can drive the GCLK through logic array routing to enable internal logic elements (LEs) to drive a high fan-out, low-skew signal path. Clock control blocks that have inputs driven by internal logic are not able to drive PLL inputs. |

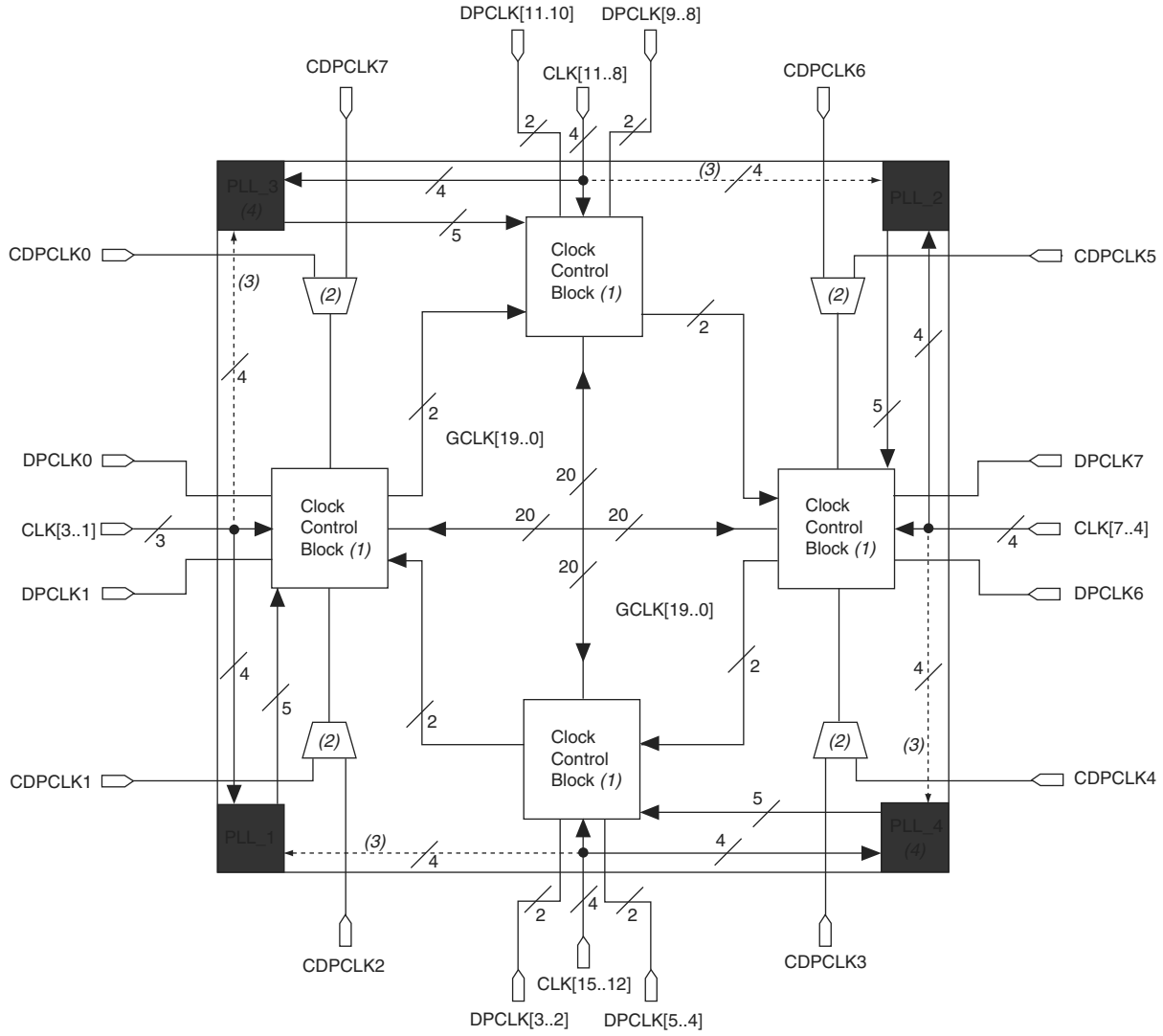
In Cyclone IV devices, dedicated clock input pins, PLL counter outputs, dual-purpose clock I/O inputs, and internal logic can all feed the clock control block for each GCLK. The output from the clock control block in turn feeds the corresponding GCLK. The GCLK can drive the PLL input if the clock control block inputs are outputs of another PLL or dedicated clock input pins. There are five or six clock control blocks on each side of the device periphery—depending on device density; providing up to 30 clock control blocks in each Cyclone IV GX device. The maximum number of clock control blocks per Cyclone IV E device is 20. For the clock control block locations, refer to Figure 5–2 on page 5–12, Figure 5–3 on page 5–13, and Figure 5–4 on page 5–14.

The clock control blocks on the left side of the Cyclone IV GX device do not support any clock inputs.

The control block has two functions:

- Dynamic GCLK clock source selection (not applicable for DPCLK, CDPCLK, and internal logic input)
- GCLK network power down (dynamic enable and disable)

**Figure 5–4. Clock Networks and Clock Control Block Locations in Cyclone IV E Devices**




**Notes to Figure 5–4:**

(1) There are five clock control blocks on each side.

(2) Only one of the corner CDPCLK pins in each corner can feed the clock control block at a time. You can use the other CDPCLK pins as general-purpose I/O (GPIO) pins.

(3) Dedicated clock pins can feed into this PLL. However, these paths are not fully compensated.

(4) PLL_3 and PLL_4 are not available in EP4CE6 and EP4CE10 devices.

The inputs to the clock control blocks on each side of the Cyclone IV GX device must be chosen from among the following clock sources:

- Four clock input pins
- Ten PLL counter outputs (five from each adjacent PLLs)
- Two, four, or six DPCLK pins from the top, bottom, and right sides of the device
- Five signals from internal logic

# 6. I/O Features in Cyclone IV Devices

CYIV-51006-2.7

This chapter describes the I/O and high speed I/O capabilities and features offered in Cyclone® IV devices.

The I/O capabilities of Cyclone IV devices are driven by the diversification of I/O standards in many low-cost applications, and the significant increase in required I/O performance. Altera's objective is to create a device that accommodates your key board design needs with ease and flexibility.

The I/O flexibility of Cyclone IV devices is increased from the previous generation low-cost FPGAs by allowing all I/O standards to be selected on all I/O banks. Improvements to on-chip termination (OCT) support and the addition of true differential buffers have eliminated the need for external resistors in many applications, such as display system interfaces.

High-speed differential I/O standards have become popular in high-speed interfaces because of their significant advantages over single-ended I/O standards. The Cyclone IV devices support LVDS, BLVDS, RSDS, mini-LVDS, and PPDS. The transceiver reference clocks and the existing general-purpose I/O (GPIO) clock input features also support the LVDS I/O standards.

The Quartus® II software completes the solution with powerful pin planning features that allow you to plan and optimize I/O system designs even before the design files are available.

This chapter includes the following sections:


- "Cyclone IV I/O Elements" on page 6–2
- "I/O Element Features" on page 6–3
- "OCT Support" on page 6–6
- "I/O Standards" on page 6–11
- "Termination Scheme for I/O Standards" on page 6–13
- "I/O Banks" on page 6–16

Table 8–25 lists the contents of previous state register 1 and previous state register 2 in the status register. The status register bit in Table 8–25 shows the bit positions in a 3-bit register. The previous state register 1 and previous state register 2 have the same bit definitions. The previous state register 1 reflects the current application configuration and the previous state register 2 reflects the previous application configuration.

**Table 8–25. Remote System Upgrade Previous State Register 1 and Previous State Register 2 Contents in Status Register**

| Status Register Bit | Definition | Description |
|---|---|---|
| 30 | nCONFIG source | One-hot, active-high field that describes the reconfiguration source that caused the Cyclone IV device to leave the previous application configuration. If there is a tie, the higher bit order indicates precedence. For example, if nCONFIG and remote system upgrade nCONFIG reach the reconfiguration state machine at the same time, the nCONFIG precedes the remote system upgrade nCONFIG. |
| 29 | CRC error source | |
| 28 | nSTATUS source | |
| 27 | User watchdog timer source | |
| 26 | Remote system upgrade nCONFIG source | |
| 25:24 | Master state machine current state | The state of the master state machine during reconfiguration causes the Cyclone IV device to leave the previous application configuration. |
| 23:0 | Boot address | The address used by the configuration scheme to load the previous application configuration. |

If a capture is inappropriately done while capturing a previous state before the system has entered remote update application configuration for the first time, a value outputs from the shift register to indicate that the capture is incorrectly called.

## Remote System Upgrade State Machine

The remote system upgrade control and update registers have identical bit definitions, but serve different roles (Table 8–22 on page 8–75). While both registers can only be updated when the device is loaded with a factory configuration image, the update register writes are controlled by the user logic, and the control register writes are controlled by the remote system upgrade state machine.

In factory configurations, the user logic should send the option bits (Cd_early and Osc_int), the configuration address, and watchdog timer settings for the next application configuration bit to the update register. When the logic array configuration reset (RU_nCONFIG) goes high, the remote system upgrade state machine updates the control register with the contents of the update register and starts system reconfiguration from the new application page.

☞ To ensure the successful reconfiguration between the pages, assert the RU_nCONFIG signal for a minimum of 250 ns. This is equivalent to strobing the reconfig input of the ALTREMOTE_UPDATE megafunction high for a minimum of 250 ns.

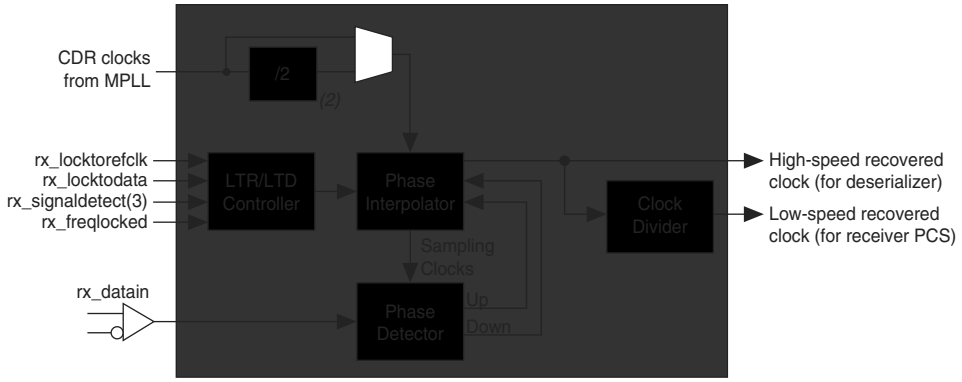
If there is an error or reconfiguration trigger condition, the remote system upgrade state machine directs the system to load a factory or application configuration (based on mode and error condition) by setting the control register accordingly.

Table 8–26 lists the contents of the control register after such an event occurs for all possible error or trigger conditions.

## Clock Data Recovery

Each receiver channel has an independent CDR unit to recover the clock from the incoming serial data stream. The high-speed recovered clock is used to clock the deserializer for serial-to-parallel conversion of the received input data, and low-speed recovered clock to clock the receiver PCS blocks. Figure 1–15 illustrates the CDR unit block diagram.

**Figure 1–15. CDR Unit Block Diagram**




**Notes to Figure 1–15:**

(1) Optional RX local divider for CDR clocks from multipurpose PLL is only available in each CDR unit for EP4CGX30 (F484 package), EP4CGX50, and EP4CGX75 devices. This block is used with the transceiver dynamic reconfiguration feature. For more information, refer to the *Cyclone IV Dynamic Reconfiguration* chapter and *AN 609: Implementing Dynamic Reconfiguration in Cyclone IV GX Devices*.

(2) CDR state transition in automatic lock mode is not dependent on `rx_signaldetect` signal, except when configured in PCI Express (PIPE) mode only.

Each CDR unit gets the reference clock from one of the two multipurpose phase-locked loops (PLLs) adjacent to the transceiver block. The CDR works by tracking the incoming data with a phase detector and finding the optimum sampling clock phase from the phase interpolator unit. The CDR operations are controlled by the LTR/LTD controller block, where the CDR may operate in the following states:

- Lock-to-reference (LTR) state—phase detector disabled and CDR ignores incoming data
- Lock-to-data (LTD) state—phase detector enabled and CDR tracks incoming data to find the optimum sampling clock phase

State transitions are supported with automatic lock mode and manual lock mode.

### Automatic Lock Mode

Upon receiver power-up and reset cycle, the CDR is put into LTR state. Transition to the LTD state is performed automatically when both of the following conditions are met:

- Signal detection circuitry indicates the presence of valid signal levels at the receiver input buffer. This condition is valid for PCI Express (PIPE) mode only. CDR transitions are not dependent on signal detection circuitry in other modes.
- The recovered clock is within the configured part per million (ppm) frequency threshold setting with respect to the CDR clocks from multipurpose PLL.

Table 1–4 lists the synchronization state machine parameters for the word aligner in this mode.

**Table 1–4. Synchronization State Machine Parameters**

| Parameter | Allowed Values |
|---|---|
| Number of erroneous code groups received to lose synchronization | 1–64 |
| Number of continuous good code groups received to reduce the error count by one | 1–256 |

After deassertion of the rx_digitalreset signal in automatic synchronization state machine mode, the word aligner starts looking for the synchronization code groups, word alignment pattern or its complement in the received data stream. When the programmed number of valid synchronization code groups or ordered sets are received, the rx_syncstatus signal is driven high to indicate that synchronization is acquired. The rx_syncstatus signal is constantly driven high until the programmed number of erroneous code groups are received without receiving intermediate good groups; after which the rx_syncstatus signal is driven low. The word aligner indicates loss of synchronization (rx_syncstatus signal remains low) until the programmed number of valid synchronization code groups are received again.

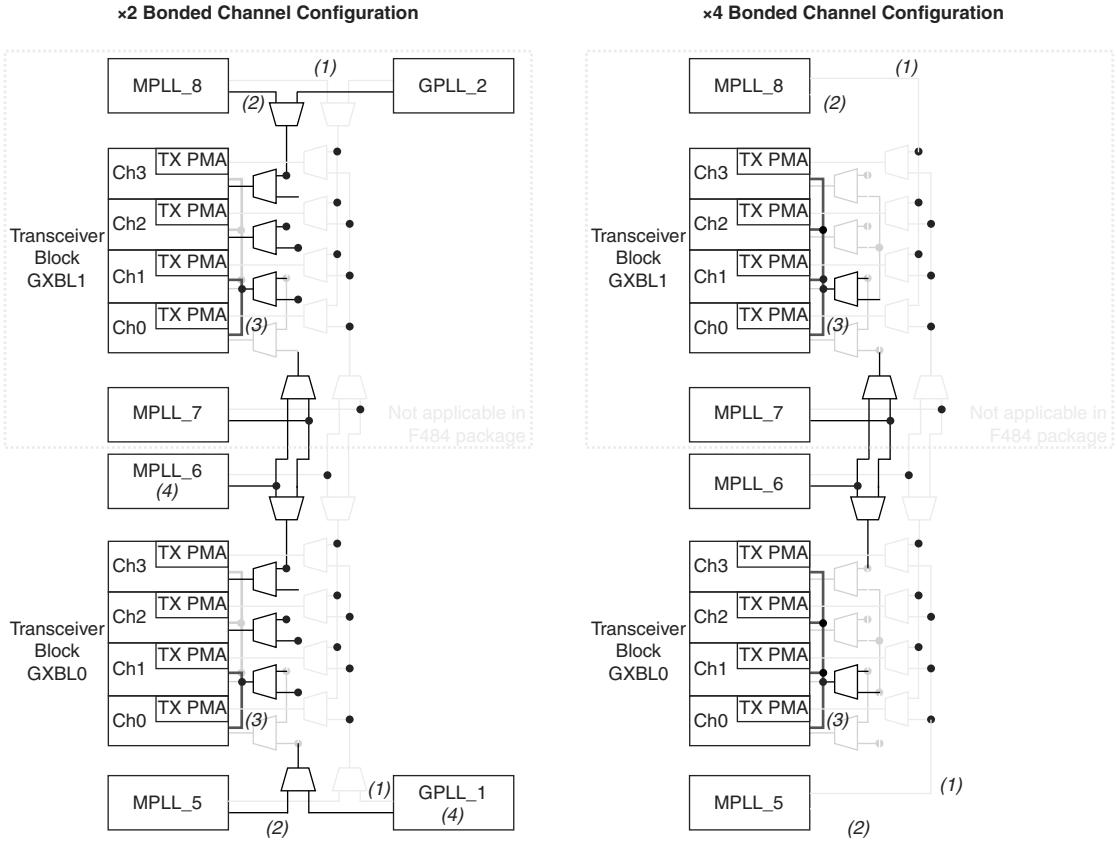
In addition to restoring word boundaries, the word aligner supports the following features:

- Programmable run length violation detection—detects consecutive 1s or 0s in the data stream, and asserts run length violation signal (rx_rlv) when a preset run length threshold (maximum number of consecutive 1s or 0s) is detected. The rx_rlv signal in each channel is clocked by its parallel recovered clock and is asserted for a minimum of two recovered clock cycles to ensure that the FPGA fabric clock can latch the rx_rlv signal reliably because the FPGA fabric clock might have phase differences, ppm differences (in asynchronous systems), or both, with the recovered clock. Table 1–5 lists the run length violation circuit detection capabilities.

**Table 1–5. Run Length Violation Circuit Detection Capabilities**

| Supported Data Width | Detector Range | | Increment Step Settings |
|---|---|---|---|
| | Minimum | Maximum | |
| 8-bit | 4 | 128 | 4 |
| 10-bit | 5 | 160 | 5 |

**Figure 1–37. Clock Distribution in Bonded (×2 and ×4) Channel Configuration for Transceivers in F484 and Larger Packages**




**Notes to Figure 1–37:**

(1) High-speed clock.
(2) Low-speed clock.
(3) Bonded common low-speed clock path.
(4) These PLLs have restricted clock driving capability and may not reach all connected channels. For details, refer to Table 1–10.

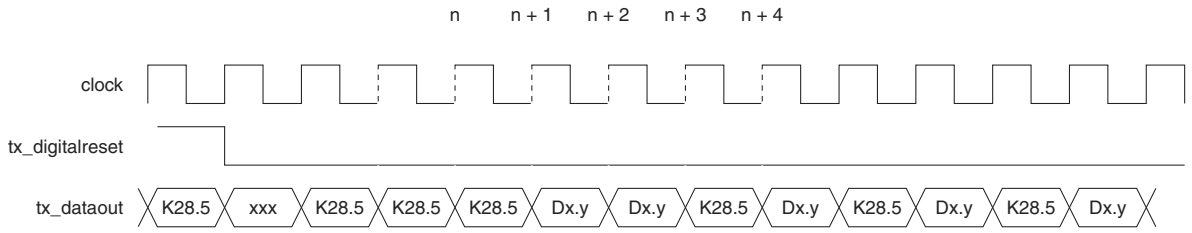
The channel datapath clocking is similar between bonded channels in ×2 and ×4 configurations.

Figure 1–38 shows the datapath clocking in Transmitter Only operation for ×2 and ×4 bonded configurations. In these configurations, each bonded channel selects the high-speed clock from one the supported PLLs. The high-speed clock in each bonded channel feeds the respective serializer for parallel to serial operation. The common bonded low-speed clock feeds to each bonded channel that is used for the following blocks in each transmitter PCS channel:

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

Figure 1–57 shows an example of even numbers of /Dx.y/ between the last automatically sent /K28.5/ and the first user-sent /K28.5/. The first user-sent /K28.5/ code group received at an odd code group boundary in cycle $n + 3$ takes the receiver synchronization state machine in Loss-of-Sync state. The first synchronization ordered-set /K28.5/Dx.y/ in cycles $n + 3$ and $n + 4$ is discounted and three additional ordered sets are required for successful synchronization.

**Figure 1–57. Example of Reset Condition in GIGE Mode**




## Running Disparity Preservation with Idle Ordered Set

During idle ordered sets transmission in GIGE mode, the transmitter ensures a negative running disparity at the end of an idle ordered set. Any /Dx.y/, except for /D21.5/ (part of /C1/ ordered set) or /D2.2/ (part of /C2/ ordered set) following a /K28.5/ is automatically replaced with either of the following:

- A /D5.6/ (/I1/ ordered set) if the running disparity before /K28.5/ is positive
- A /D16.2/ (/I2/ ordered set) if the running disparity before /K28.5/ is negative

## Lane Synchronization

In GIGE mode, the word aligner is configured in automatic synchronization state machine mode that complies with the IEEE P802.3ae standard. A synchronization ordered set is a /K28.5/ code group followed by an odd number of valid /Dx.y/ code groups. Table 1–19 lists the synchronization state machine parameters that implements the GbE-compliant synchronization.

**Table 1–19. Synchronization State Machine Parameters (1)**

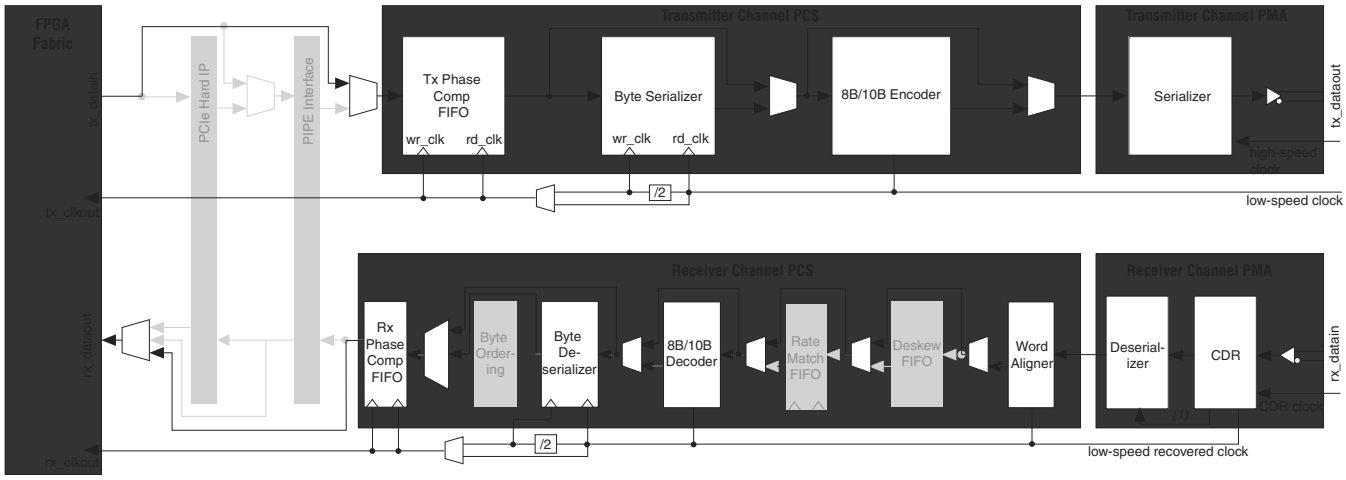
| Parameter | Value |
|---|---|
| Number of valid synchronization ordered sets received to achieve synchronization | 3 |
| Number of erroneous code groups received to lose synchronization | 4 |
| Number of continuous good code groups received to reduce the error count by one | 4 |

**Note to Table 1–19:**

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

Figure 1–66 shows the transceiver channel datapath and clocking when configured in deterministic latency mode.

**Figure 1–66. Transceiver Channel Datapath and Clocking when Configured in Deterministic Latency Mode**

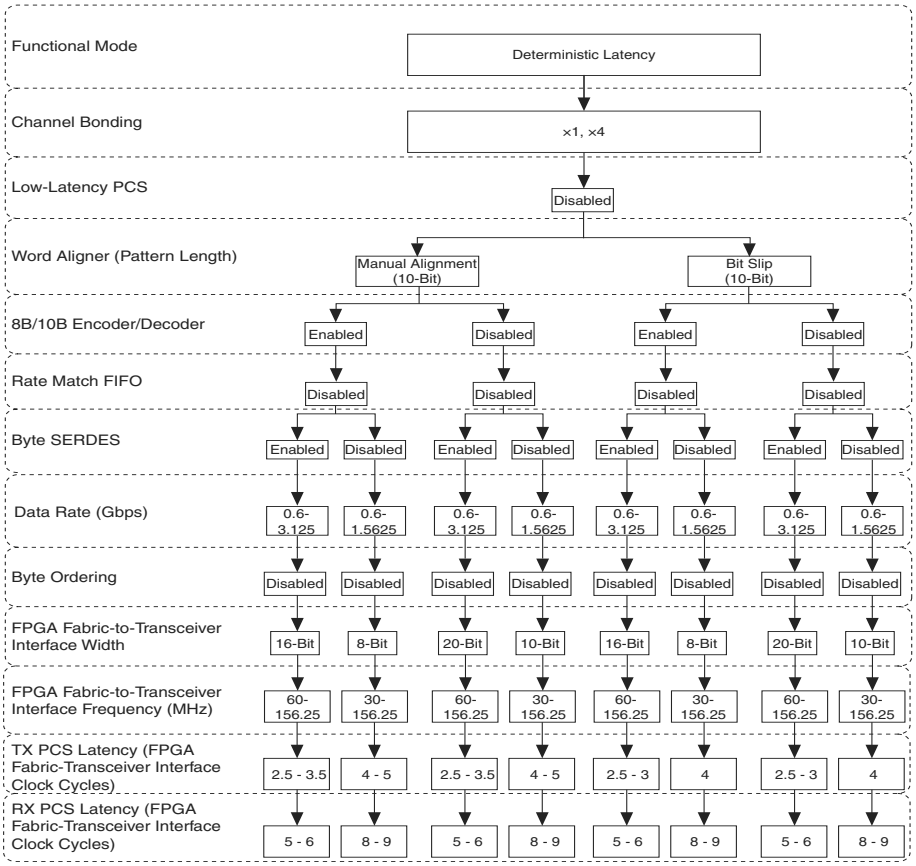



**Note to Figure 1–66:**

(1) High-speed recovered clock.

Figure 1–67 shows the transceiver configuration in Deterministic Latency mode.

**Figure 1–67. Transceiver Configuration in Deterministic Latency Mode**




Both CPRI and OBSAI protocols define the serial interface connecting the base station component (specifically channel cards) and remote radio heads (specifically radio frequency cards) in a radio base station system with fiber optics. The protocols require the accuracy of round trip delay measurement for single-hop and multi-hop connections to be within ± 16.276 ns. The Cyclone IV GX transceivers support the following CPRI and OBSAI line rates using Deterministic Latency mode:

- CPRI —614.4 Mbps, 1.2288 Gbps, 2.4576 Gbps, and 3.072 Gbps
- OBSAI—768 Mbps, 1.536 Gbps, and 3.072 Gbps

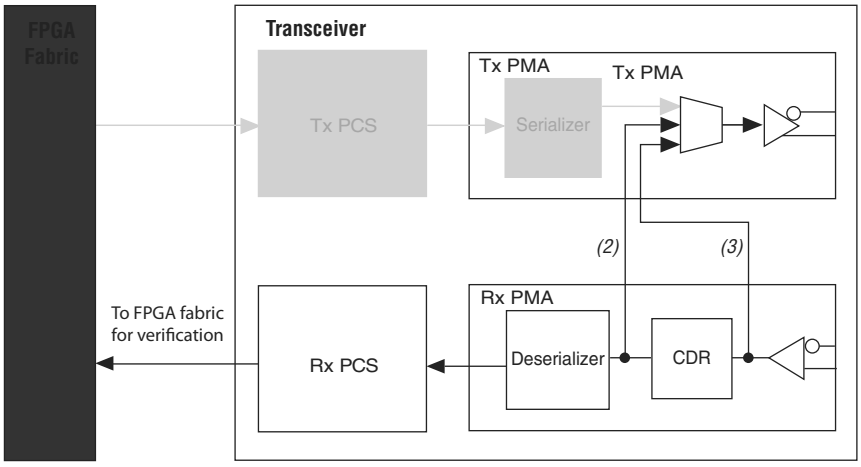
For more information about deterministic latency implementation, refer to *AN 610: Implementing Deterministic Latency for CPRI and OBSAI Protocols in Stratix IV, HardCopy IV, Arria II GX, and Cyclone IV Devices*.

## Registered Mode Phase Compensation FIFO

In Deterministic Latency mode, the RX phase compensation FIFO is set to registered mode while the TX phase compensation FIFO supports optional registered mode. When set into registered mode, the phase compensation FIFO acts as a register and eliminates the latency uncertainty through the FIFOs.

Figure 1–72 shows the two paths in reverse serial loopback mode.

**Figure 1–72. Reverse Serial Loopback (1)**




**Notes to Figure 1–72:**

(1) Grayed-Out Blocks are Not Active in this mode.
(2) Post-CDR reverse serial loopback path.
(3) Pre-CDR reverse serial loopback path.

# Self Test Modes

Each transceiver channel in the Cyclone IV GX device contains modules for pattern generator and verifier. Using these built-in features, you can verify the functionality of the functional blocks in the transceiver channel without requiring user logic. The self test functionality is provided as an optional mechanism for debugging transceiver channels.

There are three types of supported pattern generators and verifiers:

- Built-in self test (BIST) incremental data generator and verifier—test the complete transmitter PCS and receiver PCS datapaths for bit errors with parallel loopback before the PMA blocks.
- Pseudo-random binary sequence (PRBS) generator and verifier—the PRBS generator and verifier interface with the serializer and deserializer in the PMA blocks. The advantage of using a PRBS data stream is that the randomness yields an environment that stresses the transmission medium. In the data stream, you can observe both random jitter and deterministic jitter using a time interval analyzer, bit error rate tester, or oscilloscope.
- High frequency and low frequency pattern generator—the high frequency patterns generate alternate ones and zeros and the low frequency patterns generate five ones and five zeroes. These patterns do not have a corresponding verifier.

The self-test features are only supported in Basic mode.

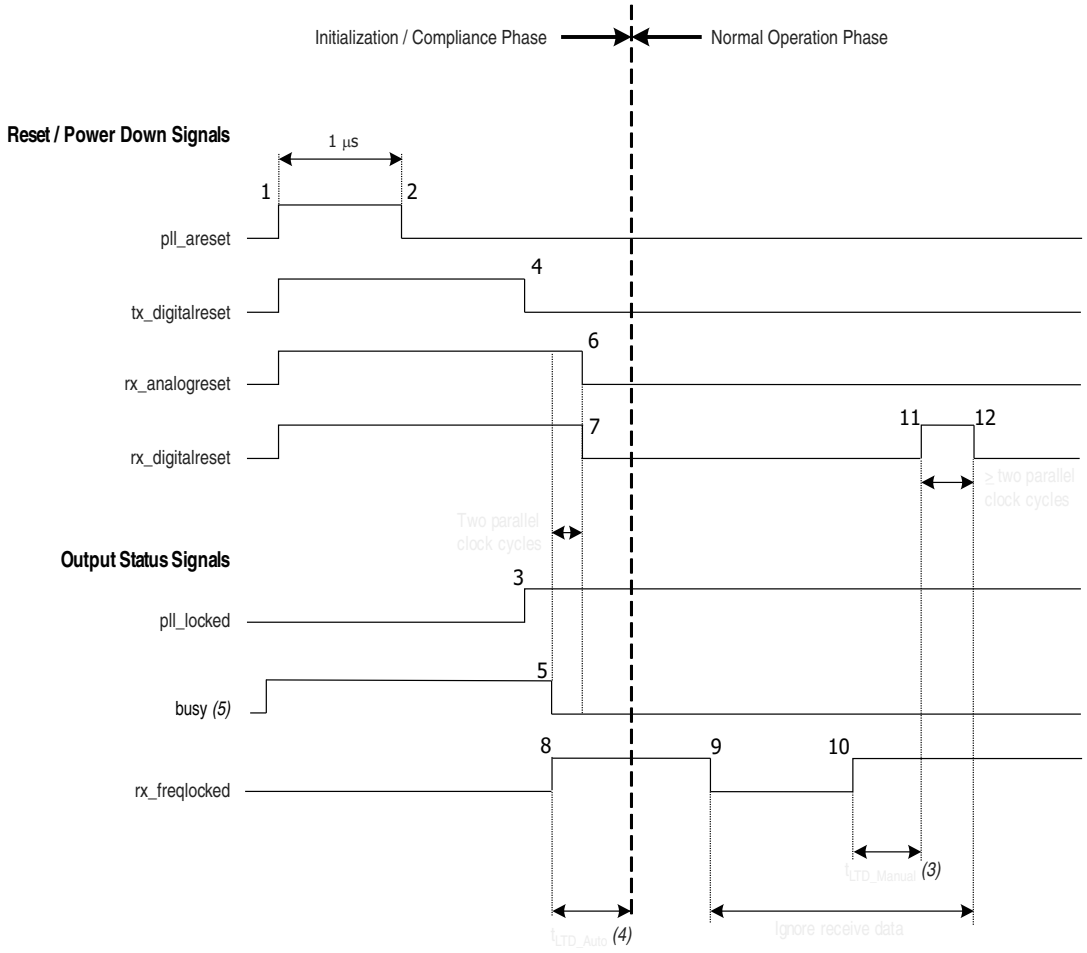
# PCIe Functional Mode

You can configure PCIe functional mode with or without the receiver clock rate compensation FIFO in the Cyclone IV GX device. The reset sequence remains the same whether or not you use the receiver clock rate compensation FIFO.

## PCIe Reset Sequence

The PCIe protocol consists of an initialization/compliance phase and a normal operation phase. The reset sequences for these two phases are described based on the timing diagram in Figure 2–10.

**Figure 2–10. Reset Sequence of PCIe Functional Mode (1), (2)**




**Notes to Figure 2–10:**

(1) This timing diagram is drawn based on the PCIe Gen 1 ×1 mode.

(2) For bonded PCIe Gen 1 ×2 and ×4 modes, there will be additional `rx_freqlocked[n]` signal. n=number of channels.

(3) For $t_{LTD\_Manual}$ duration, refer to the *Cyclone IV Device Datasheet* chapter.

(4) For $t_{LTD\_Auto}$ duration, refer to the *Cyclone IV Device Datasheet* chapter.

(5) The `busy` signal is asserted and deasserted only during initial power up when offset cancellation occurs. In subsequent reset sequences, the `busy` signal is asserted and deasserted only if there is a read or write operation to the ALTGX_RECONFIG megafunction.

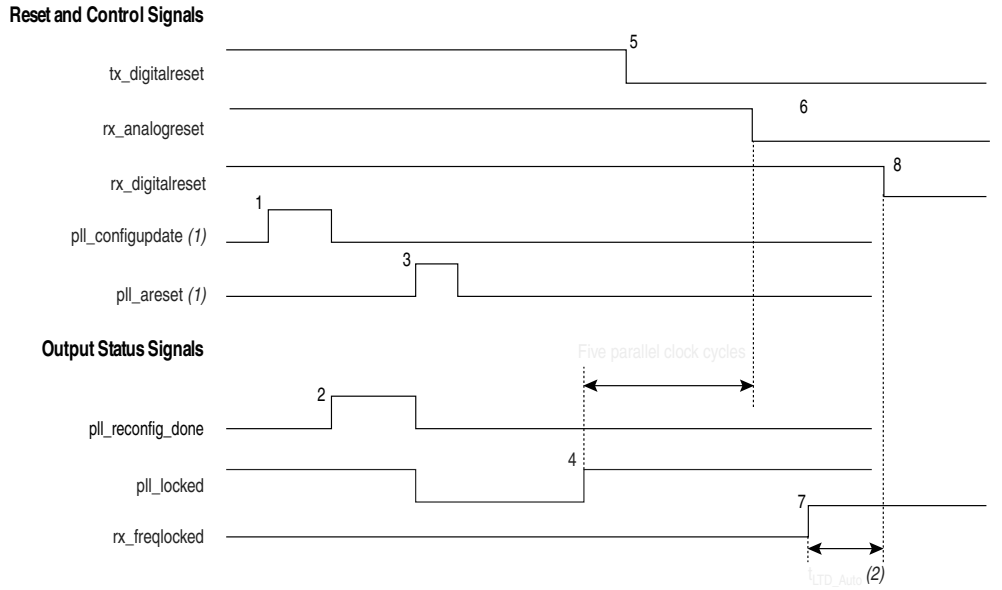
# Dynamic Reconfiguration Reset Sequences

When using dynamic reconfiguration in data rate divisions in PLL reconfiguration or channel reconfiguration mode, use the following reset sequences.

## Reset Sequence in PLL Reconfiguration Mode

Use the example reset sequence shown in Figure 2–11 when you use the PLL dynamic reconfiguration controller to change the data rate of the transceiver channel. In this example, PLL dynamic reconfiguration is used to dynamically reconfigure the data rate of the transceiver channel configured in Basic ×1 mode with the receiver CDR in automatic lock mode.

**Figure 2–11. Reset Sequence When Using the PLL Dynamic Reconfiguration Controller to Change the Data Rate of the Transceiver Channel**




**Notes to Figure 2–11:**

(1) The `pll_configupdate` and `pll_areset` signals are driven by the ALTPLL_RECONFIG megafunction. For more information, refer to *AN 609: Implementing Dynamic Reconfiguration in Cyclone IV GX Devices* and the *Cyclone IV Dynamic Reconfiguration* chapter.

(2) For $t_{LTD\_Auto}$ duration, refer to the *Cyclone IV Device Datasheet* chapter.

As shown in Figure 2–11, perform the following reset procedure when using the PLL dynamic reconfiguration controller to change the configuration of the PLLs in the transmitter channel:

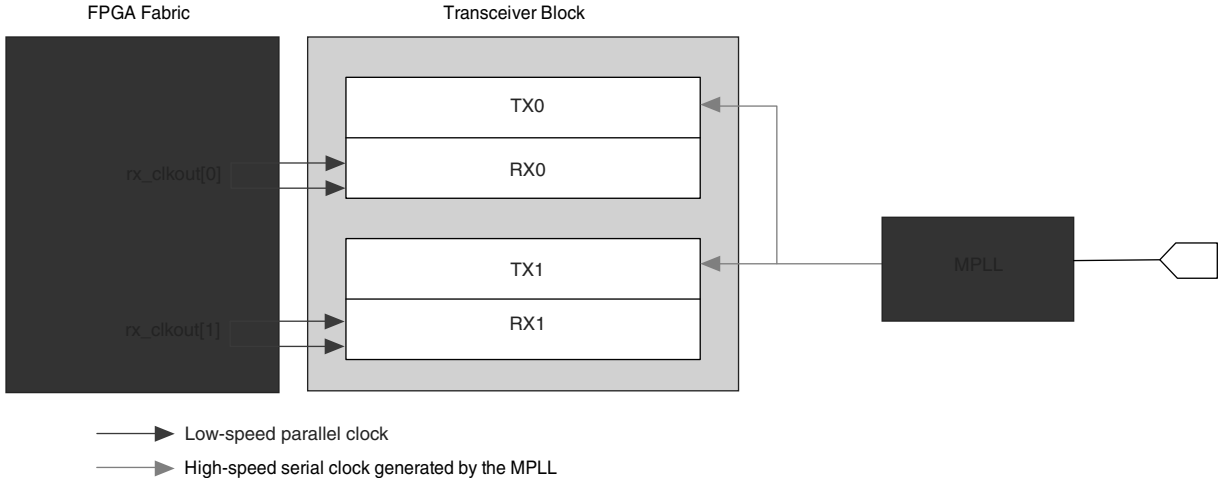
1. Assert the `tx_digitalreset`, `rx_analogreset`, and `rx_digitalreset` signals. The `pll_configupdate` signal is asserted (marker 1) by the ALTPLL_RECONFIG megafunction after the final data bit is sent out. The `pll_reconfig_done` signal is asserted (marker 2) to inform the ALTPLL_RECONFIG megafunction that the scan chain process is completed. The ALTPLL_RECONFIG megafunction then asserts the `pll_areset` signal (marker 3) to reset the transceiver PLL.

#### Option 3: Use the Respective Channel Receiver Core Clocks

- Enable this option if you want the individual channel's `rx_clkout` signal to provide the read clock to its respective Receive Phase Compensation FIFO.
- This option is typically enabled when the channel is reconfigured from a Basic or Protocol configuration with or without rate matching to another Basic or Protocol configuration with or without rate matching.

Figure 3–15 shows the respective `rx_clkout` of each channel clocking the respective receiver channels of a transceiver block.

**Figure 3–15. Option 3 for Receiver Core Clocking (Channel Reconfiguration Mode)**




## PLL Reconfiguration Mode

Cyclone IV GX device support the PLL reconfiguration support through the ALTPLL_RECONFIG MegaWizard. You can use this mode to reconfigure the multipurpose PLL or general purpose PLL used to clock the transceiver channel without affecting the remaining blocks of the channel. When you reconfigure the multipurpose PLL or general purpose PLL of a transceiver block to run at a different data rate, all the transceiver channels listening to this multipurpose PLL or general purpose PLL also get reconfigured to the new data rate. Channel settings are not affected. When you reconfigure the multipurpose PLL or general purpose PLL to support a different data rate, you must ensure that the functional mode of the transceiver channel supports the reconfigured data rate.

The PLL reconfiguration mode can be enabled by selecting the **Enable PLL Reconfiguration** option in the ALTGX MegaWizard under **Reconfiguration Setting** tab. For multipurpose PLL or general purpose PLL reconfiguration, **.mif** files are required to dynamically reconfigure the PLL setting in order to change the output frequency of the transceiver PLL to support different data rates.

Table 3–7 lists the ALTGX megafunction ports for PLL Reconfiguration mode.

**Table 3–7. ALTGX Megafunction Port List for PLL Reconfiguration Mode**

| Port Name (1) | Input/ Output | Description | Comments |
|---|---|---|---|
| pll_areset [n..0] | Input | Resets the transceiver PLL. The pll_areset are asserted in two conditions:<br>■ Used to reset the transceiver PLL during the reset sequence. During reset sequence, this signal is user controlled.<br>■ After the transceiver PLL is reconfigured, this signal is asserted high by the ALTPLL_RECONFIG controller. At this time, this signal is not user controlled. | You must connect the pll_areset port of ALTGX to the pll_areset port of the ALTPLL_RECONFIG megafunction.<br>The ALTPLL_RECONFIG controller asserts the pll_areset port at the next rising clock edge after the pll_reconfig_done signal from the ALTGX megafunction goes high. After the pll_reconfig_done signal goes high, the transceiver PLL is reset. When the PLL reconfiguration is completed, this reset is performed automatically by the ALTPLL_RECONFIG megafunction and is not user controlled. |
| pll_scandata [n..0] | Input | Receives the scan data input from the ALTPLL_RECONFIG megafunction. | The reconfigurable transceiver PLL received the scan data input through this port for the dynamically reconfigurable bits from the ALTPLL_RECONFIG controller. |
| pll_scanclk [n..0] | Input | Drives the scanclk port on the reconfigurable transceiver PLL. | Connect the pll_scanclk port of the ALTGX megafunction to the ALTPLL_RECONFIG scanclk port. |
| pll_scanclkena [n..0] | Input | Acts as a clock enable for the scanclk port on the reconfigurable transceiver PLL. | Connect the pll_scanclkena port of the ALTGX megafunction to the ALTPLL_RECONFIG scanclk port. |
| pll_configupdate [n..0] | Input | Drives the configupdate port on the reconfigurable transceiver PLL. | This port is connected to the pll_configupdate port from the ALTPLL_RECONFIG controller. After the final data bit is sent out, the ALTPLL_RECONFIG controller asserts this signal. |
| pll_reconfig_done[n..0] | Output | This signal is asserted to indicate the reconfiguration process is done. | Connect the pll_reconfig_done port to the pll_scandone port on the ALTPLL_RECONFIG controller. The transceiver PLL scandone output signal drives this port and determines when the PLL is reconfigured. |
| pll_scandataout [n..0] | Output | This port scan out the current configuration of the transceiver PLL. | Connect the pll_scandataout port to the pll_scandataout port of the ALTPLL_RECONFIG controller. This port reads the current configuration of the transceiver PLL and send it to the ALTPLL_RECONFIG megafunction. |

**Note to Table 3–7:**

(1) *<n>* = (number of transceiver PLLs configured in the ALTGX MegaWizard) - 1.

For more information about the ALTPLL_RECONFIG megafunction port list, description and usage, refer to the *Phase-Locked Loop Reconfiguration (ALTPL_RECONFIG) Megafunction User Guide*.

This chapter provides additional information about the document and Altera.

## About this Handbook

This handbook provides comprehensive information about the Altera® Cyclone® IV family of devices.

## How to Contact Altera

To locate the most up-to-date information about Altera products, refer to the following table.

| Contact (1) | Contact Method | Address |
|---|---|---|
| Technical support | Website | www.altera.com/support |
| Technical training | Website | www.altera.com/training |
| | Email | custrain@altera.com |
| Product literature | Website | www.altera.com/literature |
| Nontechnical support (general) | Email | nacomp@altera.com |
| (software licensing) | Email | authorization@altera.com |

**Note to Table:**

(1) You can also contact your local Altera sales office or sales representative.

## Typographic Conventions

The following table shows the typographic conventions this document uses.

| Visual Cue | Meaning |
|---|---|
| **Bold Type with Initial Capital Letters** | Indicate command names, dialog box titles, dialog box options, and other GUI labels. For example, **Save As** dialog box. For GUI elements, capitalization matches the GUI. |
| **bold type** | Indicates directory names, project names, disk drive names, file names, file name extensions, software utility names, and GUI labels. For example, **\qdesigns** directory, **D:** drive, and **chiptrip.gdf** file. |
| *Italic Type with Initial Capital Letters* | Indicate document titles. For example, *Stratix IV Design Guidelines*. |
| *italic type* | Indicates variables. For example, *n* + 1.<br>Variable names are enclosed in angle brackets (< >). For example, *<file name>* and *<project name>***.pof** file. |
| Initial Capital Letters | Indicate keyboard keys and menu names. For example, the Delete key and the Options menu. |
| "Subheading Title" | Quotation marks indicate references to sections in a document and titles of Quartus II Help topics. For example, "Typographic Conventions." |