



Welcome to [E-XFL.COM](https://www.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 | 1803 |
| Number of Logic Elements/Cells | 28848 |
| Total RAM Bits | 608256 |
| Number of I/O | 328 |
| Number of Gates | - |
| Voltage - Supply | 1.15V ~ 1.25V |
| 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/ep4ce30f23c7n |

Same-Port Read-During-Write Mode

This mode applies to a single-port RAM or the same port of a true dual-port RAM. In the same port read-during-write mode, there are two output choices: **New Data** mode (or flow-through) and **Old Data** mode. In **New Data** mode, new data is available on the rising edge of the same clock cycle on which it was written. In **Old Data** mode, the RAM outputs reflect the old data at that address before the write operation proceeds.

When using **New Data** mode together with *byteena*, you can control the output of the RAM. When *byteena* is high, the data written into the memory passes to the output (flow-through). When *byteena* is low, the masked-off data is not written into the memory and the old data in the memory appears on the outputs. Therefore, the output can be a combination of new and old data determined by *byteena*.

Figure 3-14 and Figure 3-15 show sample functional waveforms of same port read-during-write behavior with both **New Data** and **Old Data** modes, respectively.

Figure 3-14. Same Port Read-During Write: New Data Mode

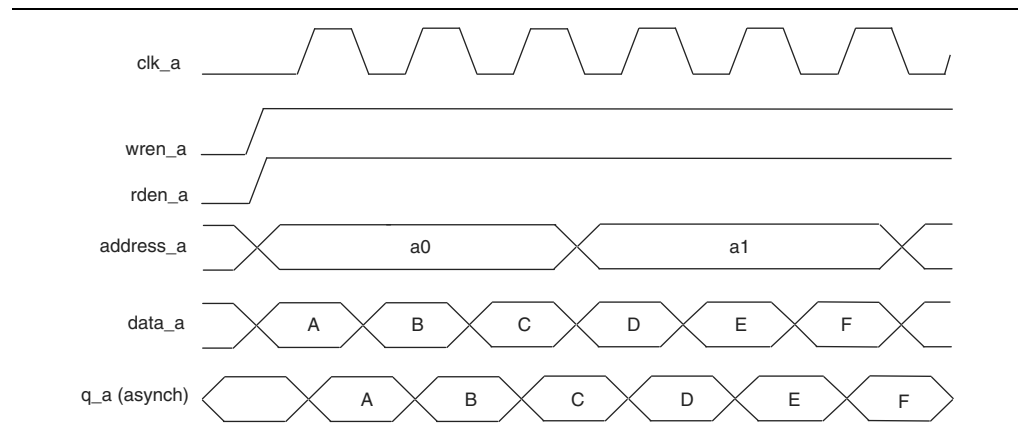
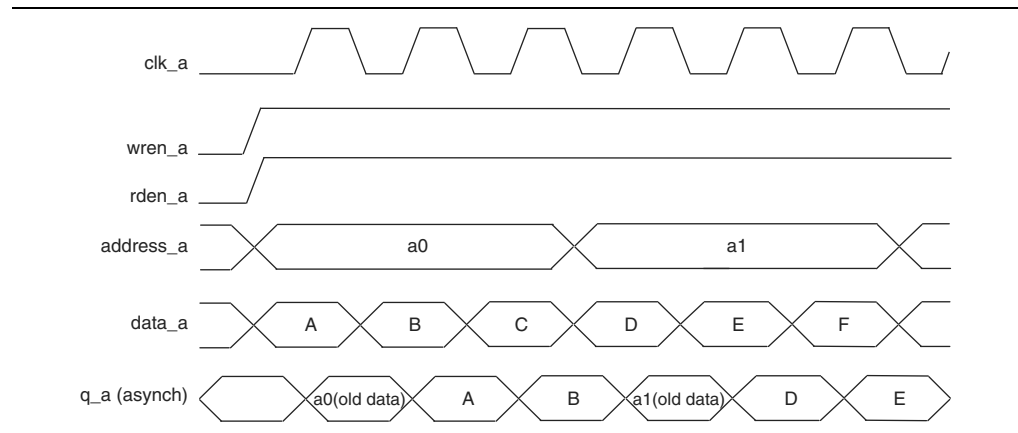


Figure 3-15. Same Port Read-During-Write: Old Data Mode



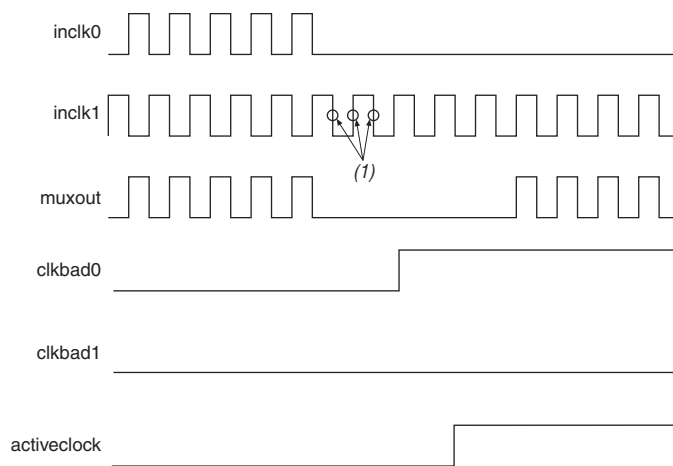
Mixed-Port Read-During-Write Mode

This mode applies to a RAM in simple or true dual-port mode, which has one port reading and the other port writing to the same address location with the same clock.

20%. This feature is useful when clock sources can originate from multiple cards on the backplane, requiring a system-controlled switchover between frequencies of operation. Choose the secondary clock frequency so the VCO operates in the recommended frequency range. Also, set the M, N, and C counters accordingly to keep the VCO operating frequency in the recommended range.

Figure 5-18 shows a waveform example of the switchover feature when using automatic loss of clock detection. Here, the `inclk0` signal remains low. After the `inclk0` signal remains low for approximately two clock cycles, the clock-sense circuitry drives the `clkbad0` signal high. Also, because the reference clock signal is not toggling, the switchover state machine controls the multiplexer through the `clksw` signal to switch to `inclk1`.

Figure 5-18. Automatic Switchover Upon Clock Loss Detection (1)



Note to Figure 5-18:

(1) Switchover is enabled on the falling edge of `inclk0` or `inclk1`, depending on which clock is available. In this figure, switchover is enabled on the falling edge of `inclk1`.

Manual Override

If you are using the automatic switchover, you must switch input clocks with the manual override feature with the `clkswitch` input.

Figure 5-19 shows an example of a waveform illustrating the switchover feature when controlled by `clkswitch`. In this case, both clock sources are functional and `inclk0` is selected as the reference clock. A low-to-high transition of the `clkswitch` signal starts the switchover sequence. The `clkswitch` signal must be high for at least three clock cycles (at least three of the longer clock period if `inclk0` and `inclk1` have different frequencies). On the falling edge of `inclk0`, the reference clock of the counter, `muxout`, is gated off to prevent any clock glitching. On the falling edge of `inclk1`, the reference clock multiplexer switches from `inclk0` to `inclk1` as the PLL reference, and the `activeclock` signal changes to indicate which clock is currently feeding the PLL.

Section II. I/O Interfaces

This section provides information about Cyclone® IV device family I/O features and high-speed differential and external memory interfaces.

This section includes the following chapters:

- Chapter 6, I/O Features in Cyclone IV Devices
- Chapter 7, External Memory Interfaces in Cyclone IV Devices

Revision History

Refer to each chapter for its own specific revision history. For information about when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the complete handbook.

- For the specific sustaining current for each V_{CCIO} voltage level driven through the resistor and for the overdrive current used to identify the next driven input level, refer to the *Cyclone IV Device Datasheet* chapter.

Programmable Pull-Up Resistor

Each Cyclone IV device I/O pin provides an optional programmable pull-up resistor while in user mode. If you enable this feature for an I/O pin, the pull-up resistor holds the output to the V_{CCIO} level of the output pin's bank.

- If you enable the programmable pull-up resistor, the device cannot use the bus-hold feature. Programmable pull-up resistors are not supported on the dedicated configuration, JTAG, and dedicated clock pins.
- When the optional `DEV_OE` signal drives low, all I/O pins remains tri-stated even with the programmable pull-up option enabled.

Programmable Delay

The Cyclone IV IOE includes programmable delays to ensure zero hold times, minimize setup times, increase clock-to-output times, and delay the clock input signal.

A path in which a pin directly drives a register may require a programmable delay to ensure zero hold time, whereas a path in which a pin drives a register through combinational logic may not require the delay. Programmable delays minimize setup time. The Quartus II Compiler can program these delays to automatically minimize setup time while providing a zero hold time. Programmable delays can increase the register-to-pin delays for output registers. Each dual-purpose clock input pin provides a programmable delay to the global clock networks.

Table 6–1 shows the programmable delays for Cyclone IV devices.

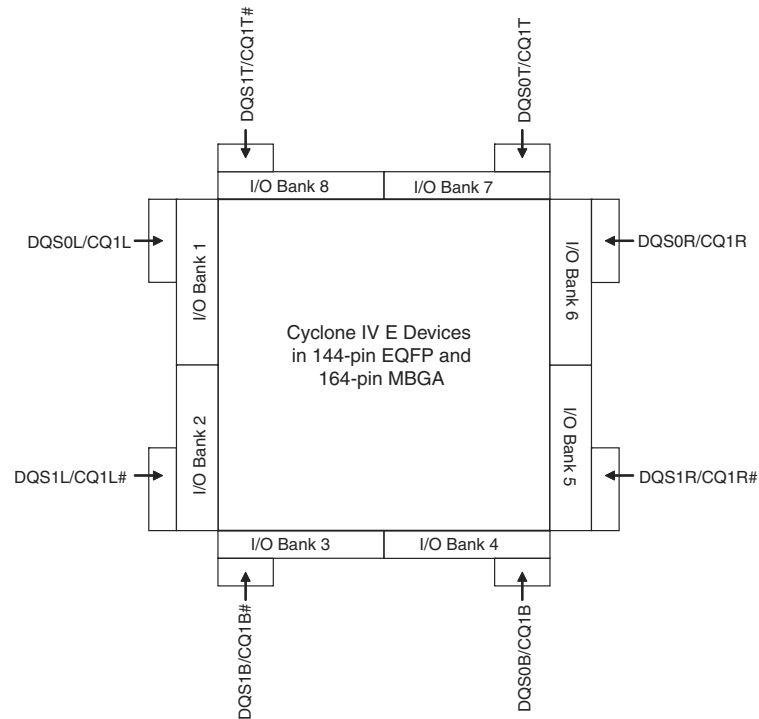
Table 6–1. Cyclone IV Devices Programmable Delay Chain

| Programmable Delay | Quartus II Logic Option |
|------------------------------------|---|
| Input pin-to-logic array delay | Input delay from pin to internal cells |
| Input pin-to-input register delay | Input delay from pin to input register |
| Output pin delay | Delay from output register to output pin |
| Dual-purpose clock input pin delay | Input delay from dual-purpose clock pin to fan-out destinations |

There are two paths in the IOE for an input to reach the logic array. Each of the two paths can have a different delay. This allows you to adjust delays from the pin to the internal logic element (LE) registers that reside in two different areas of the device. You must set the two combinational input delays with the input delay from pin to internal cells logic option in the Quartus II software for each path. If the pin uses the input register, one of the delays is disregarded and the delay is set with the input delay from pin to input register logic option in the Quartus II software.

Figure 7-6 shows the location and numbering of the DQS, DQ, or CQ# pins in I/O banks of the Cyclone IV E device in the 144-pin EQFP and 164-pin MBGA packages.

Figure 7-6. DQS, CQ, or CQ# Pins for Cyclone IV E Devices in the 144-Pin EQFP and 164-pin MBGA Packages

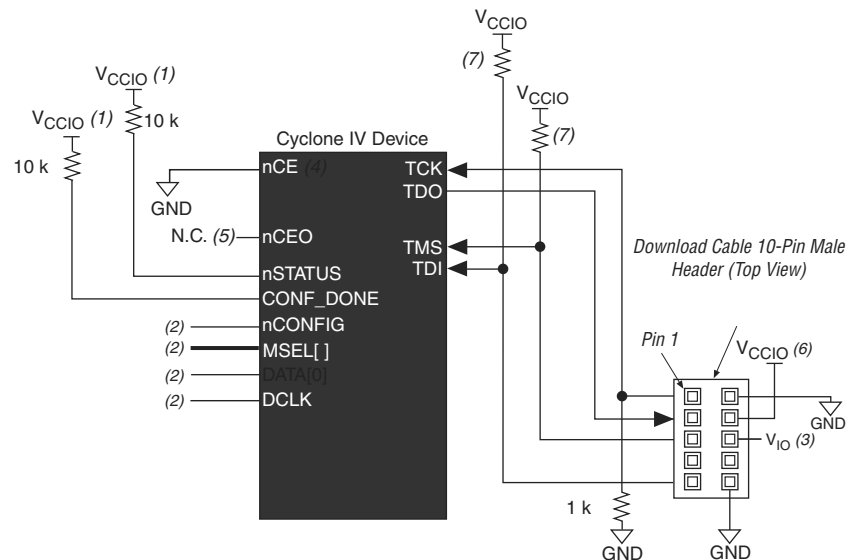


In Cyclone IV devices, the $\times 9$ mode uses the same DQ and DQS pins as the $\times 8$ mode, and one additional DQ pin that serves as a regular I/O pin in the $\times 8$ mode. The $\times 18$ mode uses the same DQ and DQS pins as $\times 16$ mode, with two additional DQ pins that serve as regular I/O pins in the $\times 16$ mode. Similarly, the $\times 36$ mode uses the same DQ and DQS pins as the $\times 32$ mode, with four additional DQ pins that serve as regular I/O pins in the $\times 32$ mode. When not used as DQ or DQS pins, the memory interface pins are available as regular I/O pins.

Optional Parity, DM, and Error Correction Coding Pins

Cyclone IV devices support parity in $\times 9$, $\times 18$, and $\times 36$ modes. One parity bit is available per eight bits of data pins. You can use any of the DQ pins for parity in Cyclone IV devices because the parity pins are treated and configured similarly to DQ pins.

DM pins are only required when writing to DDR2 and DDR SDRAM devices. QDR II SRAM devices use the BWS# signal to select the byte to be written into memory. A low signal on the DM or BWS# pin indicates the write is valid. Driving the DM or BWS# pin high causes the memory to mask the DQ signals. Each group of DQS and DQ signals has one DM pin. Similar to the DQ output signals, the DM signals are clocked by the -90° shifted clock.

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 k Ω to 10 k Ω .

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.



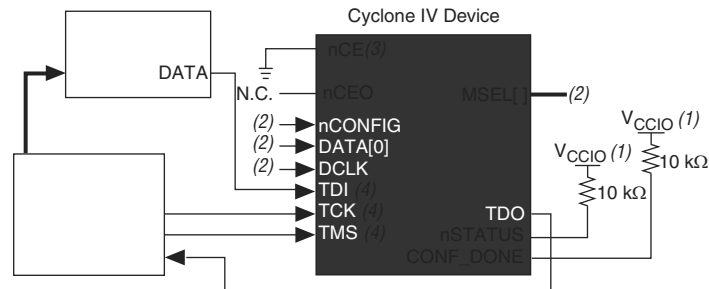
-  JTAG configuration allows an unlimited number of Cyclone IV devices to be cascaded in a JTAG chain.
-  For more information about configuring multiple Altera devices in the same configuration chain, refer to the *Configuring Mixed Altera FPGA Chains* chapter in volume 2 of the *Configuration Handbook*.

Figure 8-27 shows JTAG configuration with a Cyclone IV device and a microprocessor.

Figure 8-27. JTAG Configuration of a Single Device Using a Microprocessor




Notes to Figure 8-27:

- (1) You must connect the pull-up resistor to a supply that provides an acceptable input signal for all devices in the chain.
- (2) Connect the nCONFIG and MSEL pins to support a non-JTAG configuration scheme. If you only use a 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) You must connect the nCE pin to GND or driven low for successful JTAG configuration.
- (4) All I/O inputs must maintain a maximum AC voltage of 4.1 V. Signals driving into TDI, TMS, and TCK must fit the maximum overshoot outlined in Equation 8-1 on page 8-5.

Configuring Cyclone IV Devices with Jam STAPL

Jam™ STAPL, JEDEC standard JESD-71, is a standard file format for in-system programmability (ISP) purposes. Jam STAPL supports programming or configuration of programmable devices and testing of electronic systems, using the IEEE 1149.1 JTAG interface. Jam STAPL is a freely licensed open standard. The Jam Player provides an interface for manipulating the IEEE Std. 1149.1 JTAG TAP state machine.

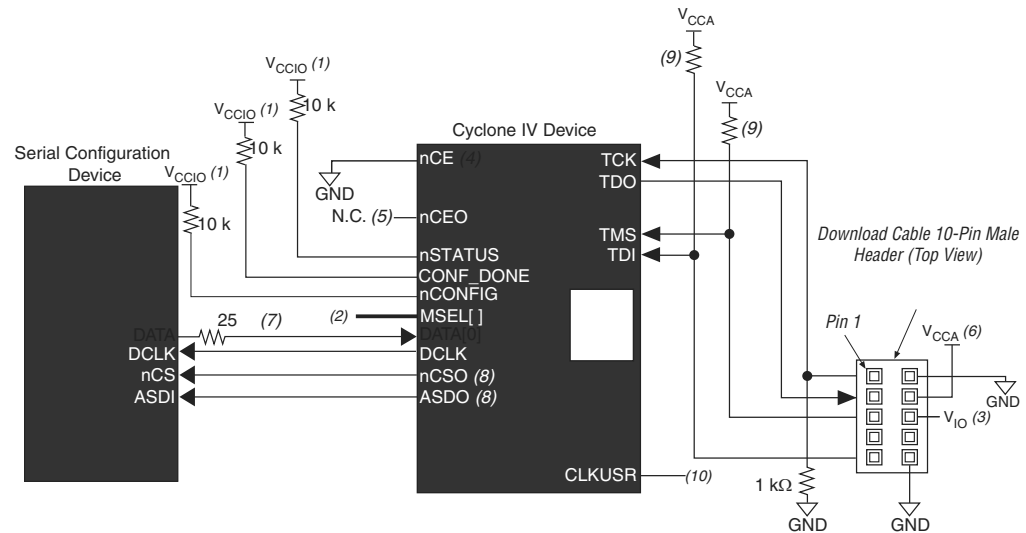
-  For more information about JTAG and Jam STAPL in embedded environments, refer to *AN 425: Using Command-Line Jam STAPL Solution for Device Programming*. To download the Jam Player, visit the Altera website (www.altera.com).

Configuring Cyclone IV Devices with the JRunner Software Driver

The JRunner software driver allows you to configure Cyclone IV devices through the ByteBlaster II or ByteBlasterMV cables in JTAG mode. The supported programming input file is in .rbf format. The JRunner software driver also requires a Chain Description File (.cdf) generated by the Quartus II software. The JRunner software driver is targeted for embedded JTAG configuration. The source code is developed for the Windows NT operating system (OS). You can customize the code to make it run on your embedded platform.

If you configure a master device with an SFL design, the master device enters user mode even though the slave devices in the multiple device chain are not being configured. The master device enters user mode with a SFL design even though the CONF_DONE signal is externally held low by the other slave devices in chain. Figure 8-29 shows the JTAG configuration of a single Cyclone IV device with a SFL design.

Figure 8-29. Programming Serial Configuration Devices In-System Using the JTAG Interface



Notes to Figure 8-29:

- (1) Connect the pull-up resistors to the V_{CCIO} supply of the bank in which the pin resides.
- (2) The MSEL pin settings vary for different configuration voltage standards and POR time. To connect MSEL for AS configuration schemes, refer to Table 8-3 on page 8-8, 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.
- (3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. The V_{IO} must match the V_{CCA} of the device. For this value, refer to the *MasterBlaster Serial/USB Communications Cable User Guide*. When using the ByteBlasterMV download cable, this pin is a no connect. When using USB-Blaster, ByteBlaster II, and EthernetBlaster cables, this pin is connected to nCE when it is used for AS programming, otherwise it is a no connect.
- (4) You must connect the nCE pin 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, USB-Blaster, or ByteBlasterMV cable with a 2.5- V V_{CCA} supply. Third-party programmers must switch to 2.5 V. Pin 4 of the header is a V_{CC} power supply for the MasterBlaster cable. The MasterBlaster cable can receive power from either 5.0- or 3.3-V circuit boards, DC power supply, or 5.0 V from the USB cable. For this value, refer to the *MasterBlaster Serial/USB Communications Cable User Guide*.
- (7) Connect the series resistor at the near end of the serial configuration device.
- (8) These pins are dual-purpose I/O pins. The nCSO pin functions as FLASH_nCE pin in AP mode. The ASDO pin functions as DATA[1] pin in AP and FPP modes.
- (9) Resistor value can vary from 1 k Ω to 10 k Ω .
- (10) Only Cyclone IV GX devices have an option to select CLKUSR (40 MHz maximum) as the external clock source for DCLK.

ISP of the Configuration Device

In the second stage, the SFL design in the master device allows you to write the configuration data for the device chain into the serial configuration device with the Cyclone IV device JTAG interface. The JTAG interface sends the programming data for the serial configuration device to the Cyclone IV device first. The Cyclone IV device then uses the ASMI pins to send the data to the serial configuration device.

Table 8-20. Dedicated Configuration Pins on the Cyclone IV Device (Part 3 of 4)

| Pin Name | User Mode | Configuration Scheme | Pin Type | Description |
|-----------------------------|-----------|--------------------------------|---|---|
| DCLK ⁽¹⁾ | N/A | PS, FPP, AS, AP ⁽²⁾ | Input (PS, FPP) ⁽²⁾ | In PS and FPP configuration, DCLK is the clock input used to clock data from an external source into the target Cyclone IV device. Data is latched into the device on the rising edge of DCLK. |
| | I/O | | Output (AS, AP) | In AS mode, DCLK is an output from the Cyclone IV device that provides timing for the configuration interface. It has an internal pull-up resistor (typically 25 kΩ) that is always active. In AP mode, DCLK is an output from the Cyclone IV E device that provides timing for the configuration interface. ⁽²⁾ In AS or AP configuration schemes, this pin is driven into an inactive state after configuration completes. Alternatively, in active schemes, you can use this pin as a user I/O during user mode. In PS or FPP schemes that use a control host, you must drive DCLK either high or low, whichever is more convenient. In passive schemes, you cannot use DCLK as a user I/O in user mode. Toggling this pin after configuration does not affect the configured device. |
| DATA[0] ⁽¹⁾ | I/O | PS, FPP, AS, AP ⁽²⁾ | Input (PS, FPP, AS). Bidirectional (AP) ⁽²⁾ | Data input. In serial configuration modes, bit-wide configuration data is presented to the target Cyclone IV device on the DATA[0] pin. In AS mode, DATA[0] has an internal pull-up resistor that is always active. After AS configuration, DATA[0] is a dedicated input pin with optional user control. After PS or FPP configuration, DATA[0] is available as a user I/O pin. The state of this pin depends on the Dual-Purpose Pin settings. After AP configuration, DATA[0] is a dedicated bidirectional pin with optional user control. ⁽²⁾ |
| DATA[1]/ASDO ⁽¹⁾ | I/O | FPP, AS, AP ⁽²⁾ | Input (FPP). Output (AS). Bidirectional (AP) ⁽²⁾ | The DATA[1] pin functions as the ASDO pin in AS mode. Data input in non-AS mode. Control signal from the Cyclone IV device to the serial configuration device in AS mode used to read out configuration data. In AS mode, DATA[1] has an internal pull-up resistor that is always active. After AS configuration, DATA[1] is a dedicated output pin with optional user control. In a PS configuration scheme, DATA[1] functions as a user I/O pin during configuration, which means it is tri-stated. After FPP configuration, DATA[1] is available as a user I/O pin and the state of this pin depends on the Dual-Purpose Pin settings. In an AP configuration scheme, for Cyclone IV E devices only, the byte-wide or word-wide configuration data is presented to the target Cyclone IV E device on DATA[7..0] or DATA[15..0], respectively. After AP configuration, DATA[1] is a dedicated bidirectional pin with optional user control. ⁽²⁾ |

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.

$$\text{Error detection frequency} = \frac{80 \text{ MHz}}{2^n}$$

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

Table 9-6 lists the estimated time for each CRC calculation with minimum and maximum clock frequencies for Cyclone IV devices.

Table 9-6. CRC Calculation Time

| Device | | Minimum Time (ms) ⁽¹⁾ | Maximum Time (s) ⁽²⁾ |
|---------------|-------------------------|----------------------------------|---------------------------------|
| Cyclone IV E | EP4CE6 ⁽³⁾ | 5 | 2.29 |
| | EP4CE10 ⁽³⁾ | 5 | 2.29 |
| | EP4CE15 ⁽³⁾ | 7 | 3.17 |
| | EP4CE22 ⁽³⁾ | 9 | 4.51 |
| | EP4CE30 ⁽³⁾ | 15 | 7.48 |
| | EP4CE40 ⁽³⁾ | 15 | 7.48 |
| | EP4CE55 ⁽³⁾ | 23 | 11.77 |
| | EP4CE75 ⁽³⁾ | 31 | 15.81 |
| | EP4CE115 ⁽³⁾ | 45 | 22.67 |
| Cyclone IV GX | EP4CGX15 | 6 | 2.93 |
| | EP4CGX22 | 12 | 5.95 |
| | EP4CGX30 | 12 | 5.95 |
| | | 34 ⁽⁴⁾ | 17.34 ⁽⁴⁾ |
| | EP4CGX50 | 34 | 17.34 |
| | EP4CGX75 | 34 | 17.34 |
| | EP4CGX110 | 62 | 31.27 |
| | EP4CGX150 | 62 | 31.27 |

Notes to Table 9-6:

- (1) The minimum time corresponds to the maximum error detection clock frequency and may vary with different processes, voltages, and temperatures (PVT).
- (2) The maximum time corresponds to the minimum error detection clock frequency and may vary with different PVT.
- (3) Only applicable for device with 1.2-V core voltage
- (4) Only applicable for the F484 device package.

Software Support

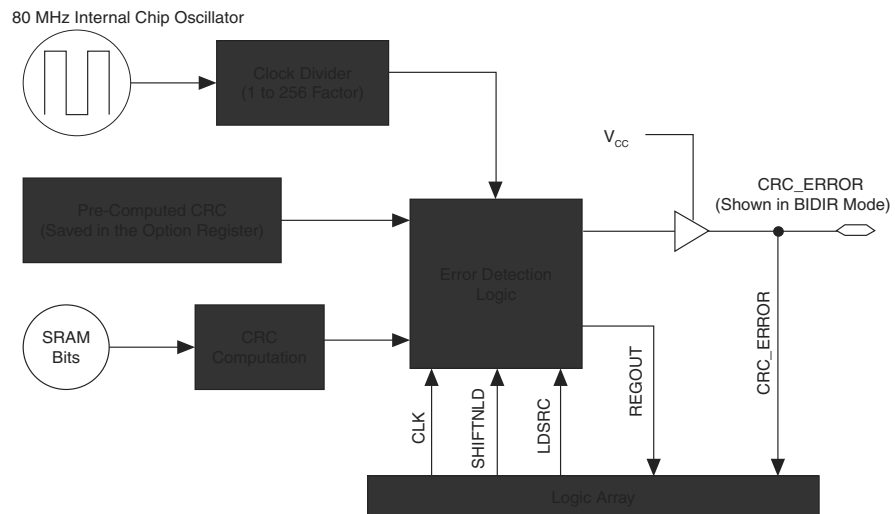
Enabling the CRC error detection feature in the Quartus II software generates the CRC_ERROR output to the optional dual purpose CRC_ERROR pin.


To enable the error detection feature using CRC, perform the following steps:

1. Open the Quartus II software and load a project using Cyclone IV devices.
2. On the Assignments menu, click **Settings**. The **Settings** dialog box appears.
3. In the Category list, select **Device**. The **Device** page appears.
4. Click **Device and Pin Options**. The **Device and Pin Options** dialog box appears as shown in Figure 9-2.
5. In the **Device and Pin Options** dialog box, click the **Error Detection CRC** tab.
6. Turn on **Enable error detection CRC**.
7. In the **Divide error check frequency by** box, enter a valid divisor as documented in Table 9-5 on page 9-5.

Figure 9-3 shows the error detection block diagram in FPGA devices and shows the interface that the WYSIWYG atom enables in your design.

Figure 9-3. Error Detection Block Diagram



 The user logic is affected by the soft error failure, so reading out the 32-bit CRC signature through the `regout` should not be relied upon to detect a soft error. You should rely on the `CRC_ERROR` output signal itself, because this `CRC_ERROR` output signal cannot be affected by a soft error.

To enable the `cycloneiv_crcblock` WYSIWYG atom, you must name the atom for each Cyclone IV device accordingly.

Example 9-1 shows an example of how to define the input and output ports of a WYSIWYG atom in a Cyclone IV device.

Example 9-1. Error Detection Block Diagram

```
cycloneiv_crcblock<crcblock_name>
(
  .clk(<clock source>),
  .shiftnld(<shiftnld source>),
  .ldsrc(<ldsrc source>),
  .crcerror(<crcerror out destination>),
  .regout(<output destination>),
);
```

In configuration with rate match FIFO, the transmitter datapath clocking is identical to Transmitter Only operation as shown in Figure 1-38. In each bonded receiver channel, the CDR unit recovers the clock from serial received data and generates the high- and low-speed recovered clock for each bonded channel. The high-speed recovered clock feeds the channel's deserializer, and low-speed recovered clock is forwarded to receiver PCS. The individual low-speed recovered clock feeds to the following blocks in the receiver PCS:

- word aligner
- write clock of rate match FIFO

The common bonded low-speed clock that is used in all bonded transmitter PCS datapaths feeds the following blocks in each bonded receiver PCS:

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

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

FPGA Fabric-Transceiver Interface Clocking

The FPGA fabric-transceiver interface clocks consists of clock signals from the FPGA fabric to the transceiver blocks, and from the transceiver blocks to the FPGA fabric. These clock resources use the global clock networks (GCLK) in the FPGA core.

 For information about the GCLK resources in the Cyclone IV GX devices, refer to *Clock Networks and PLLs in Cyclone IV Devices* chapter.

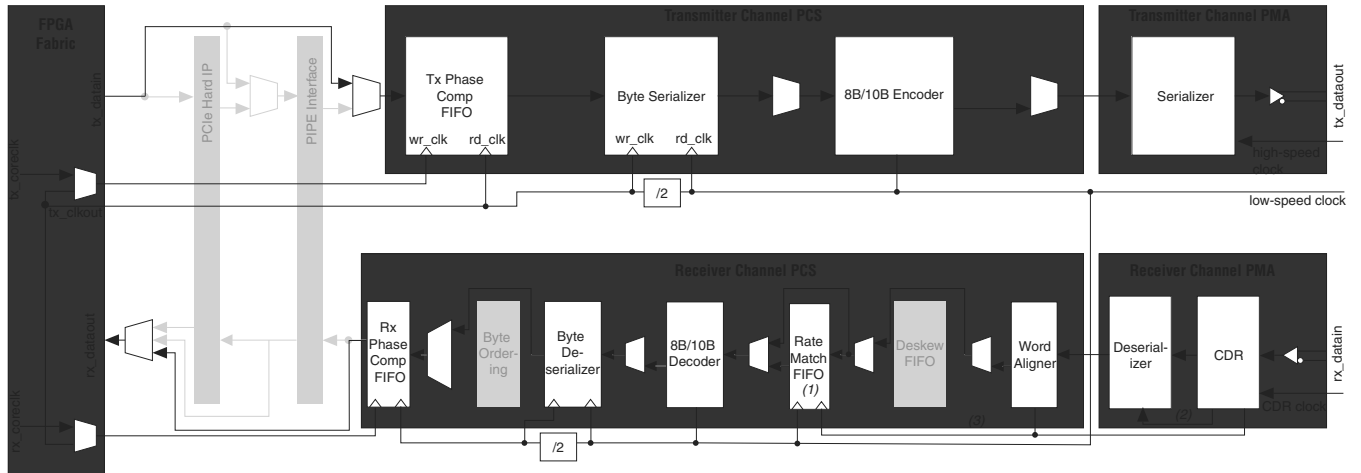
Table 1-11 lists the FPGA fabric-transceiver interface clocks.

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

| Clock Name | Clock Description | Interface Direction |
|----------------------------------|---|----------------------------|
| tx_clkout | Phase compensation FIFO clock | Transceiver to FPGA fabric |
| rx_clkout | Phase compensation FIFO clock | Transceiver to FPGA fabric |
| coreclkout | Phase compensation FIFO clock | Transceiver to FPGA fabric |
| fixed_clk | 125MHz receiver detect clock in PIPE mode | FPGA fabric to transceiver |
| reconfig_clk ^{(1), (2)} | Transceiver dynamic reconfiguration and offset cancellation clock | FPGA fabric to transceiver |

Figure 1-60 shows the transceiver channel datapath and clocking when configured in Serial RapidIO mode.

Figure 1-60. Transceiver Channel Datapath and Clocking when Configured in Serial RapidIO Mode



Notes to Figure 1-60:

- (1) Optional rate match FIFO.
- (2) High-speed recovered clock.
- (3) Low-speed recovered clock.


Table 1-28. PIPE Interface Ports in ALTGX Megafunction for Cyclone IV GX⁽¹⁾ (Part 1 of 2)


| Port Name | Input/Output | Clock Domain | Description |
|-------------------------|--------------|---------------------|---|
| fixedclk | Input | Clock signal | 125-MHz clock for receiver detect and offset cancellation only in PIPE mode. |
| tx_detectrxloop | Input | Asynchronous signal | Receiver detect or reverse parallel loopback control. <ul style="list-style-type: none"> ■ A high level in the P1 power state and <code>tx_forceelecidle</code> signal asserted begins the receiver detection operation to determine if there is a valid receiver downstream. This signal must be deasserted when the <code>pipephydonestatus</code> signal indicates receiver detect completion. ■ A high level in the P0 power state with the <code>tx_forceelecidle</code> signal deasserted dynamically configures the channel to support reverse parallel loopback mode. |
| tx_forcedisp compliance | Input | Asynchronous signal | Force the 8B/10B encoder to encode with negative running disparity. <ul style="list-style-type: none"> ■ Assert only when transmitting the first byte of the PIPE-compliance pattern to force the 8B/10B encoder with a negative running disparity. |
| pipe8b10binvpolarity | Input | Asynchronous signal | Invert the polarity of every bit of the 10-bit input to the 8B/10B decoder |
| powerdn | Input | Asynchronous signal | PIPE power state control. <ul style="list-style-type: none"> ■ Signal is 2 bits wide and is encoded as follows: <ul style="list-style-type: none"> ■ 2'b00: P0 (Normal operation) ■ 2'b01: P0s (Low recovery time latency, low power state) ■ 2'b10: P1 (Longer recovery time latency, lower power state) ■ 2'b11: P2 (Lowest power state) |
| pipedatavalid | Output | N/A | Valid data and control on the <code>rx_dataout</code> and <code>rx_ctrlldetect</code> ports indicator. |
| pipephydone status | Output | Asynchronous signal | PHY function completion indicator. <ul style="list-style-type: none"> ■ Asserted for one clock cycle to communicate completion of several PHY functions, such as power state transition and receiver detection. |
| pipeelecidle | Output | Asynchronous signal | Electrical idle detected or inferred at the receiver indicator. <ul style="list-style-type: none"> ■ When electrical idle inference is used, this signal is driven high when it infers an electrical idle condition ■ When electrical idle inference is not used, the <code>rx_signaldetect</code> signal is inverted and driven on this port. |

Table 2–2 lists the power-down signals available for each transceiver block.

Table 2–2. Transceiver Block Power-Down Signals

| Signal | Description |
|---------------|--|
| pll_areset | Resets the transceiver PLL. The <code>pll_areset</code> signal is asserted in two conditions: <ul style="list-style-type: none"> ■ During reset sequence, the signal is asserted to reset the transceiver PLL. This signal is controlled by the user. ■ After the transceiver PLL is reconfigured, the signal is asserted high by the <code>ALTPLL_RECONFIG</code> controller. This signal is not controlled by the user. |
| gxb_powerdown | Powers down the entire transceiver block. When this signal is asserted, this signal powers down the PCS and PMA in all the transceiver channels. This signal operates independently from the other reset signals. This signal is common to the transceiver block. |
| pll_locked | A status signal. Indicates the status of the transmitter multipurpose PLLs or general purpose PLLs. <ul style="list-style-type: none"> ■ A high level—indicates the multipurpose PLL or general purpose PLL is locked to the incoming reference clock frequency. |
| rx_freqlocked | A status signal. Indicates the status of the receiver CDR lock mode. <ul style="list-style-type: none"> ■ A high level—the receiver is in lock-to-data mode. ■ A low level—the receiver CDR is in lock-to-reference mode. |
| busy | A status signal. An output from the <code>ALTGX_RECONFIG</code> block indicates the status of the dynamic reconfiguration controller. This signal remains low for the first <code>reconfig_clk</code> clock cycle after power up. It then gets asserted from the second <code>reconfig_clk</code> clock cycle. Assertion on this signal indicates that the offset cancellation process is being executed on the receiver buffer as well as the receiver CDR. When this signal is deasserted, it indicates that offset cancellation is complete. This busy signal is also used to indicate the dynamic reconfiguration duration such as in analog reconfiguration mode and channel reconfiguration mode. |

 For more information about offset cancellation, refer to the *Cyclone IV Dynamic Reconfiguration* chapter.

 If none of the channels is instantiated in a transceiver block, the Quartus® II software automatically powers down the entire transceiver block.

Blocks Affected by the Reset and Power-Down Signals

Table 2–3 lists the blocks that are affected by specific reset and power-down signals.

Table 2–3. Blocks Affected by Reset and Power-Down Signals (Part 1 of 2)

| Transceiver Block | rx_digitalreset | rx_analogreset | tx_digitalreset | pll_areset | gxb_powerdown |
|--|-----------------|----------------|-----------------|------------|---------------|
| multipurpose PLLs and general purpose PLLs | — | — | — | ✓ | — |
| Transmitter Phase Compensation FIFO | — | — | ✓ | — | ✓ |
| Byte Serializer | — | — | ✓ | — | ✓ |
| 8B/10B Encoder | — | — | ✓ | — | ✓ |

Table 1-4. Recommended Operating Conditions for Cyclone IV GX Devices (Part 2 of 2)

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|----------------|---|--|------------|-----|------------|------|
| V_{CCA_GXB} | Transceiver PMA and auxiliary power supply | — | 2.375 | 2.5 | 2.625 | V |
| V_{CCL_GXB} | Transceiver PMA and auxiliary power supply | — | 1.16 | 1.2 | 1.24 | V |
| V_I | DC input voltage | — | -0.5 | — | 3.6 | V |
| V_O | DC output voltage | — | 0 | — | V_{CCIO} | V |
| T_J | Operating junction temperature | For commercial use | 0 | — | 85 | °C |
| | | For industrial use | -40 | — | 100 | °C |
| t_{RAMP} | Power supply ramp time | Standard power-on reset (POR) ⁽⁷⁾ | 50 μ s | — | 50 ms | — |
| | | Fast POR ⁽⁸⁾ | 50 μ s | — | 3 ms | — |
| I_{Diode} | Magnitude of DC current across PCI-clamp diode when enabled | — | — | — | 10 | mA |

Notes to Table 1-4:

- All V_{CCA} pins must be powered to 2.5 V (even when PLLs are not used) and must be powered up and powered down at the same time.
- You must connect V_{CCD_PLL} to V_{CCINT} through a decoupling capacitor and ferrite bead.
- Power supplies must rise monotonically.
- V_{CCIO} for all I/O banks must be powered up during device operation. Configuration pins are powered up by V_{CCIO} of I/O Banks 3, 8, and 9 where I/O Banks 3 and 9 only support V_{CCIO} of 1.5, 1.8, 2.5, 3.0, and 3.3 V. For fast passive parallel (FPP) configuration mode, the V_{CCIO} level of I/O Bank 8 must be powered up to 1.5, 1.8, 2.5, 3.0, and 3.3 V.
- You must set V_{CC_CLKIN} to 2.5 V if you use $CLKIN$ as a high-speed serial interface (HSSI) $refclk$ or as a $DIFFCLK$ input.
- The $CLKIN$ pins in I/O Banks 3B and 8B can support single-ended I/O standard when the pins are used to clock left PLLs in non-transceiver applications.
- The POR time for Standard POR ranges between 50 and 200 ms. V_{CCINT} , V_{CCA} , and V_{CCIO} of I/O Banks 3, 8, and 9 must reach the recommended operating range within 50 ms.
- The POR time for Fast POR ranges between 3 and 9 ms. V_{CCINT} , V_{CCA} , and V_{CCIO} of I/O Banks 3, 8, and 9 must reach the recommended operating range within 3 ms.

ESD Performance

This section lists the electrostatic discharge (ESD) voltages using the human body model (HBM) and charged device model (CDM) for Cyclone IV devices general purpose I/Os (GPIOs) and high-speed serial interface (HSSI) I/Os. Table 1-5 lists the ESD for Cyclone IV devices GPIOs and HSSI I/Os.

Table 1-5. ESD for Cyclone IV Devices GPIOs and HSSI I/Os

| Symbol | Parameter | Passing Voltage | Unit |
|--------------|--|-----------------|------|
| V_{ESDHBM} | ESD voltage using the HBM (GPIOs) ⁽¹⁾ | ± 2000 | V |
| | ESD using the HBM (HSSI I/Os) ⁽²⁾ | ± 1000 | V |
| V_{ESDCDM} | ESD using the CDM (GPIOs) | ± 500 | V |
| | ESD using the CDM (HSSI I/Os) ⁽²⁾ | ± 250 | V |

Notes to Table 1-5:

- The passing voltage for EP4CGX15 and EP4CGX30 row I/Os is ± 1000 V.
- This value is applicable only to Cyclone IV GX devices.

Table 1–20. Differential I/O Standard Specifications for Cyclone IV Devices ⁽¹⁾ (Part 2 of 2)

| I/O Standard | V _{CCIO} (V) | | | V _{ID} (mV) | | V _{ICM} (V) ⁽²⁾ | | | V _{OD} (mV) ⁽³⁾ | | | V _{OS} (V) ⁽³⁾ | | | |
|---|-----------------------|-----|-------|----------------------|-----|-------------------------------------|---|------|-------------------------------------|-----|-----|------------------------------------|------|-------|---|
| | Min | Typ | Max | Min | Max | Min | Condition | Max | Min | Typ | Max | Min | Typ | Max | |
| LVDS (Column I/Os) | 2.375 | 2.5 | 2.625 | 100 | — | 0.05 | $D_{MAX} \leq 500 \text{ Mbps}$ | 1.80 | 247 | — | 600 | 1.125 | 1.25 | 1.375 | |
| | | | | | | 0.55 | $500 \text{ Mbps} \leq D_{MAX} \leq 700 \text{ Mbps}$ | 1.80 | | | | | | | |
| | | | | | | 1.05 | $D_{MAX} > 700 \text{ Mbps}$ | 1.55 | | | | | | | |
| BLVDS (Row I/Os) ⁽⁴⁾ | 2.375 | 2.5 | 2.625 | 100 | — | — | — | — | — | — | — | — | — | — | — |
| BLVDS (Column I/Os) ⁽⁴⁾ | 2.375 | 2.5 | 2.625 | 100 | — | — | — | — | — | — | — | — | — | — | — |
| mini-LVDS (Row I/Os) ⁽⁵⁾ | 2.375 | 2.5 | 2.625 | — | — | — | — | — | 300 | — | 600 | 1.0 | 1.2 | 1.4 | |
| mini-LVDS (Column I/Os) ⁽⁵⁾ | 2.375 | 2.5 | 2.625 | — | — | — | — | — | 300 | — | 600 | 1.0 | 1.2 | 1.4 | |
| RSDS [®] (Row I/Os) ⁽⁵⁾ | 2.375 | 2.5 | 2.625 | — | — | — | — | — | 100 | 200 | 600 | 0.5 | 1.2 | 1.5 | |
| RSDS (Column I/Os) ⁽⁵⁾ | 2.375 | 2.5 | 2.625 | — | — | — | — | — | 100 | 200 | 600 | 0.5 | 1.2 | 1.5 | |
| PPDS (Row I/Os) ⁽⁵⁾ | 2.375 | 2.5 | 2.625 | — | — | — | — | — | 100 | 200 | 600 | 0.5 | 1.2 | 1.4 | |
| PPDS (Column I/Os) ⁽⁵⁾ | 2.375 | 2.5 | 2.625 | — | — | — | — | — | 100 | 200 | 600 | 0.5 | 1.2 | 1.4 | |

Notes to Table 1–20:

- (1) For an explanation of terms used in Table 1–20, refer to “Glossary” on page 1–37.
- (2) V_{IN} range: $0 \text{ V} \leq V_{IN} \leq 1.85 \text{ V}$.
- (3) R_L range: $90 \leq R_L \leq 110 \Omega$.
- (4) There are no fixed V_{IN}, V_{OD}, and V_{OS} specifications for BLVDS. They depend on the system topology.
- (5) The Mini-LVDS, RSDS, and PPDS standards are only supported at the output pins.
- (6) The LVPECL I/O standard is only supported on dedicated clock input pins. This I/O standard is not supported for output pins.