2. Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet

CF52002-2.4

Features

- Enhanced configuration devices include EPC4, EPC8, and EPC16 devices
- Single-chip configuration solution for Altera® Stratix® series, Cyclone® series, APEX™ II, APEX 20K (including APEX 20K, APEX 20KC, and APEX 20KE), Mercury™, ACEX® 1K, and FLEX® 10K (FLEX 10KE and FLEX 10KA) devices
- Contains 4-, 8-, and 16-Mbit flash memories for configuration data storage
  - On-chip decompression feature almost doubles the effective configuration density
- Standard flash die and a controller die combined into single stacked chip package
- External flash interface supports parallel programming of flash and external processor access to unused portions of memory
  - Flash memory block/sector protection capability via external flash interface
  - Supported in EPC16 and EPC4 devices
- Page mode support for remote and local reconfiguration with up to eight configurations for the entire system
  - Compatible with Stratix series Remote System Configuration feature
- Supports byte-wide configuration mode fast passive parallel (FPP); 8-bit data output per DCLK cycle
- Supports true n-bit concurrent configuration (n = 1, 2, 4, and 8) of Altera FPGAs
- Pin-selectable 2-ms or 100-ms power-on reset (POR) time
- Configuration clock supports programmable input source and frequency synthesis
  - Multiple configuration clock sources supported (internal oscillator and external clock input pin)
  - External clock source with frequencies up to 100 MHz
  - Internal oscillator defaults to 10 MHz; Programmable for higher frequencies of 33, 50, and 66 MHz
  - Clock synthesis supported via user programmable divide counter
- Available in the 100-pin plastic quad flat pack (PQFP) and the 88-pin Ultra FineLine BGA® packages
  - Vertical migration between all devices supported in the 100-pin PQFP package
- Supply voltage of 3.3 V (core and I/O)

- Hardware compliant with IEEE Std. 1532 in-system programmability (ISP) specification
- Supports ISP via Jam Standard Test and Programming Language (STAPL)
- Supports Joint Test Action Group (JTAG) boundary scan
- *n*INIT_CONF pin allows private JTAG instruction to initiate FPGA configuration
- Internal pull-up resistor on *n*INIT_CONF always enabled
- User programmable weak internal pull-up resistors on *n*CS and OE pins
- Internal weak pull-up resistors on external flash interface address and control lines, bus hold on data lines
- Standby mode with reduced power consumption

For more information on FPGA configuration schemes and advanced features, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

Functional Description

The Altera enhanced configuration device is a single-device, high-speed, advanced configuration solution for very high-density FPGAs. The core of an enhanced configuration device is divided into two major blocks, a configuration controller and a flash memory. The flash memory is used to store configuration data for systems made up of one or more Altera FPGAs. Unused portions of the flash memory can be used to store processor code or data that can be accessed via the external flash interface after FPGA configuration is complete.

Altera has introduced additional enhanced configuration devices. For details, please refer to the Process Change Notification *PCN0506: Addition of Intel Flash Memory As Source For EPC4, EPC8 & EPC16 Enhanced Configuration Devices* and *Using Intel Flash Memory Based EPC4, EPC8 and EPC16* white paper.

EPC devices support three different types of flash memory. Table 2–1 shows the supported flash memory for all EPC devices.

Table 2–1. Enhanced Configuration Devices Flash Memory (Part 1 of 2)

Device	Flash Memory
EPC16	Intel Flash *(1)*
	Sharp Flash
EPC8	Intel Flash *(1)*
	Sharp Flash

Table 2–1. Enhanced Configuration Devices Flash Memory (Part 2 of 2)

Device	Flash Memory
EPC4	Intel Flash *(1)*
	Micron Flash

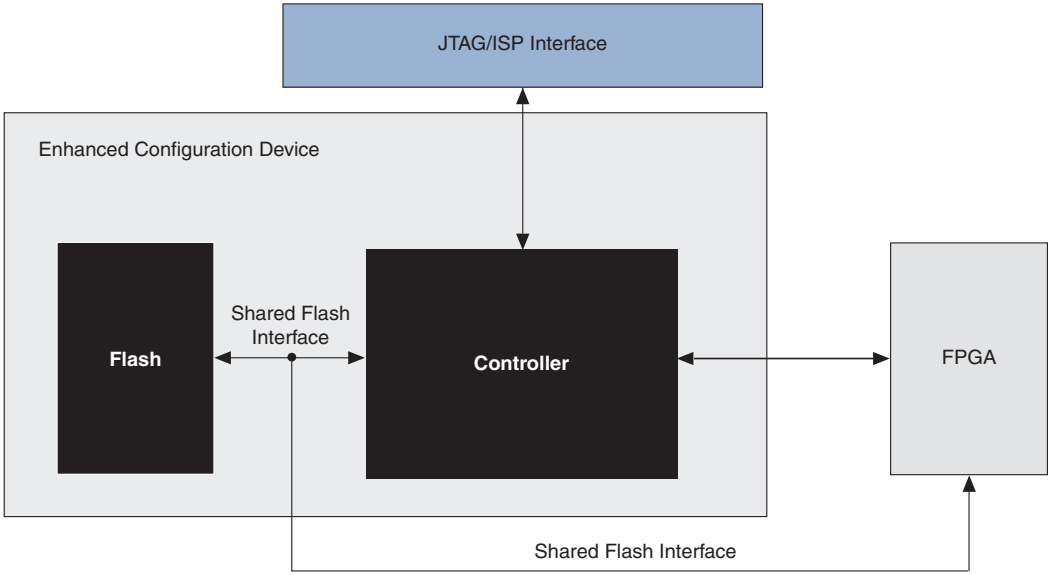
Note to Table 2–1:
(1) Refer to Process Change Notice *PCN0506: Addition of Intel Flash Memory As Source for EPC4, EPC8 & EPC16 Enhanced Configuration Devices*.

The external flash interface is currently supported in the EPC16 and EPC4 devices. For information on using this feature in the EPC8 device, contact Altera Applications.

The enhanced configuration device has a 3.3-V core and I/O interface. The controller chip is a synchronous system that implements the various interfaces and features. Figure 2–1 shows a block diagram of the enhanced configuration device. The controller chip features three separate interfaces:

- A configuration interface between the controller and the Altera FPGA(s)
- A JTAG interface on the controller that enables in-system programmability (ISP) of the flash memory
- An external flash interface that the controller shares with an external processor, or FPGA implementing a Nios® embedded processor (interface available after ISP and configuration)

Figure 2–1. Enhanced Configuration Device Block Diagram



The enhanced configuration device features multiple configuration schemes. In addition to supporting the traditional passive serial (PS) configuration scheme for a single device or a serial device chain, the enhanced configuration device features concurrent configuration and parallel configuration. With the concurrent configuration scheme, up to eight PS device chains can be configured simultaneously. In the FPP configuration scheme, 8-bits of data are clocked into the FPGA each cycle. These schemes offer significantly reduced configuration times over traditional schemes.

Furthermore, the enhanced configuration device features a dynamic configuration or page mode feature. This feature allows you to dynamically reconfigure all the FPGAs in your system with new images stored in the configuration memory. Up to eight different system configurations or pages can be stored in memory and selected using the `PGM[2..0]` pins. Your system can be dynamically reconfigured by selecting one of the eight pages and initiating a reconfiguration cycle.

This page mode feature combined with the external flash interface allows remote and local updates of system configuration data. The enhanced configuration devices are compatible with the Stratix Remote System Configuration feature.

For more information on Stratix Remote System Configuration, refer to the *Using Remote System Configuration with Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook*.

Other user programmable features include:

- Real-time decompression of configuration data
- Programmable configuration clock (DCLK)
- Flash ISP
- Programmable power-on-reset delay (PORSEL)

FPGA Configuration

FPGA configuration is managed by the configuration controller chip. This process includes reading configuration data from the flash memory, decompressing it if necessary, transmitting configuration data via the appropriate DATA[] pins, and handling error conditions.

After POR, the controller determines the user-defined configuration options by reading its option bits from the flash memory. These options include the configuration scheme, configuration clock speed, decompression, and configuration page settings. The option bits are stored at flash address location 0x8000 (word address) and occupy 512-bits or 32-words of memory. These options bits are read using the internal flash interface and the default 10 MHz internal oscillator.

After obtaining the configuration settings, it checks if the FPGA is ready to accept configuration data by monitoring the *n*STATUS and CONF_DONE lines. When the FPGA is ready (*n*STATUS is high and CONF_DONE is low), the controller begins data transfer using the DCLK and DATA[] output pins. The controller selects the configuration page to be transmitted to the FPGA(s) by sampling its PGM[2..0] pins after POR or reset.

The function of the configuration unit is to transmit decompressed data to the FPGA, depending on the configuration scheme. The enhanced configuration device supports four concurrent configuration modes, with $n = 1, 2, 4$, or 8 (where n is the number of bits that are sent per DCLK cycle on the DATA[n] lines). The value $n = 1$ corresponds to the traditional PS configuration scheme. The values $n = 2, 4$, and 8 correspond to concurrent configuration of 2, 4, or 8 different PS configuration chains, respectively. Additionally, the FPGA can be configured in FPP mode, where eight bits of DATA are clocked into the FPGA per DCLK cycle. Depending on the configuration bus width (n), the circuit shifts uncompressed configuration data to the valid DATA[n] pins. Unused DATA[] pins drive low.

In addition to transmitting configuration data to the FPGAs, the configuration circuit is also responsible for pausing configuration whenever there is insufficient data available for transmission. This occurs when the flash read bandwidth is lower than the configuration write bandwidth. Configuration is paused by stopping the DCLK to the FPGA, when waiting for data to be read from the flash or for data to be decompressed. This technique is called "Pausing DCLK."

The enhanced configuration device flash-memories feature a 90-ns access time (approximately 10 MHz). Hence, the flash read bandwidth is limited to about 160 megabits per second (Mbps) (16-bit flash data bus, DQ[], at 10 MHz). However, the configuration speeds supported by Altera FPGAs are much higher and translate to high configuration write bandwidths. For instance, 100-MHz Stratix FPP configuration requires data at the rate of 800 Mbps (8-bit DATA[] bus at 100 MHz). This is much higher than the 160 Mbps the flash memory can support, and is the limiting factor for configuration time. Compression increases the effective flash-read bandwidth since the same amount of configuration data takes up less space in the flash memory after compression. Since Stratix configuration data compression ratios are approximately two, the effective read bandwidth doubles to about 320 Mbps.

Finally, the configuration controller also manages errors during configuration. A CONF_DONE error occurs when the FPGA does not de-assert its CONF_DONE signal within 64 DCLK cycles after the last bit of configuration data is transmitted. When a CONF_DONE error is detected, the controller pulses the OE line low, which pulls *n*STATUS low and triggers another configuration cycle.

A cyclic redundancy check (CRC) error occurs when the FPGA detects corruption in the configuration data. This corruption could be a result of noise coupling on the board such as poor signal integrity on the configuration signals. When this error is signaled by the FPGA (by driving the *n*STATUS line low), the controller stops configuration. If the **Auto-Restart Configuration After Error** option is enabled in the FPGA, it releases its *n*STATUS signal after a reset time-out period and the controller attempts to reconfigure the FPGA.

After the FPGA configuration process is complete, the controller drives DCLK low and the DATA[] pins high. Additionally, the controller tri-states its internal interface to the flash memory, enables the weak internal pull-ups on the flash address and control lines, and enables bus-keep circuits on flash data lines.

The following sections briefly describe the different configuration schemes supported by the enhanced configuration device: FPP, PS, and concurrent configuration.

For detailed information on using these schemes to configure your Altera FPGA, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

Configuration Signals

Table 2–2 lists the configuration signal connections between the enhanced configuration device and Altera FPGAs.

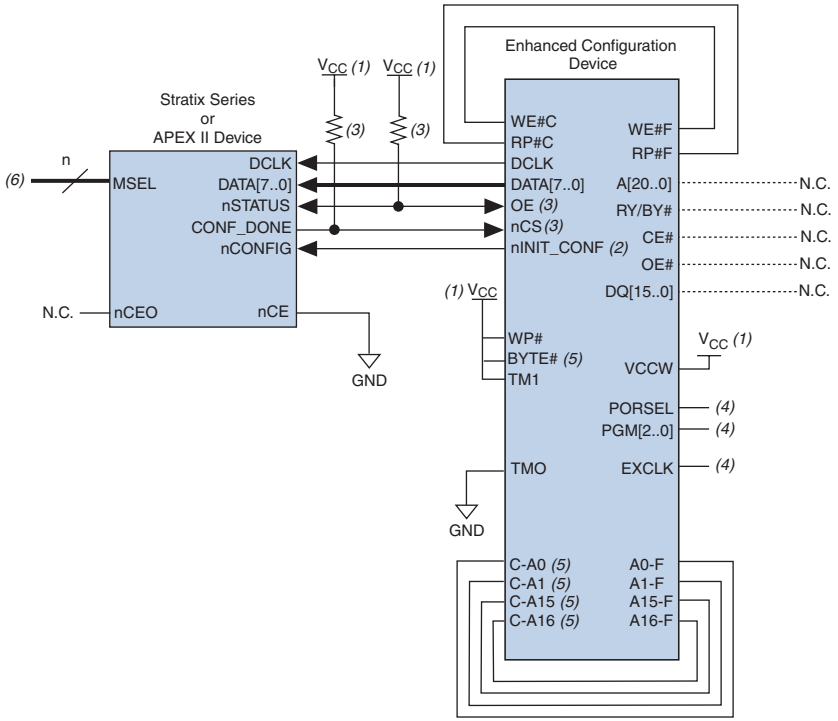
Table 2–2. Configuration Signals

Enhanced Configuration Device Pin	Altera FPGA Pin	Description
DATA[]	DATA[]	Configuration data transmitted from the configuration device to the FPGA, which is latched on the rising edge of DCLK.
DCLK	DCLK	Configuration device generated clock used by the FPGA to latch configuration data provided on the DATA[] pins.
*n*INIT_CONF	*n*CONFIG	Open-drain output from the configuration device that is used to initiate FPGA reconfiguration using the initiate configuration (INIT_CONF) JTAG instruction. This connection is not needed if the INIT_CONF JTAG instruction is not needed. If *n*INIT_CONF is not connected to *n*CONFIG, *n*CONFIG must be tied to V_{CC} either directly or through a pull-up resistor.
OE	*n*STATUS	Open-drain bidirectional configuration status signal, which is driven low by either device during POR and to signal an error during configuration. Low pulse on OE resets the enhanced configuration device controller.
*n*CS	CONF_DONE	Configuration done output signal driven by the FPGA.

Fast Passive Parallel Configuration

Stratix series and APEX II devices can be configured using the enhanced configuration device in FPP mode. In this mode, the enhanced configuration device sends a byte of data on the DATA[7..0] pins, which connect to the DATA[7..0] input pins of the FPGA, per DCLK cycle. Stratix series and APEX II FPGAs receive byte-wide configuration data per DCLK cycle. Figure 2–2 shows the enhanced configuration device in FPP configuration mode. In this figure, the external flash interface is not used and hence most flash pins are left unconnected (with the few noted exceptions). For specific details on configuration interface connections including pull-up resistor values, supply voltages, and MSEL pin settings, refer to the appropriate FPGA family chapter in the Configuration Handbook.

Figure 2–2. FPP Configuration



Notes to Figure 2–2:

(1) The V_{CC} should be connected to the same supply voltage as the configuration device.

(2) The *n*INIT_CONF pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active. This means an external pull-up resistor is not required on the *n*INIT_CONF/*n*CONFIG line. The *n*INIT_CONF pin does not need to be connected if its functionality is not used. If *n*INIT_CONF is not used, *n*CONFIG must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' OE and *n*CS pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus® II software. To turn off the internal pull-up resistors, check the **Disable *n*CS and OE pull-ups on configuration device** option when generating programming files.

(4) For PORSEL, PGM[], and EXCLK pin connections, refer to Table 2–8.

(5) In the 100-pin PQFP package, you must externally connect the following pins: C-A0 to F-A0, C-A1 to F-A1, C-A15 to F-A15, C-A16 to F-A16, and BYTE# to V_{CC}. Additionally, you must make the following pin connections in both 100-pin PQFP and 88-pin Ultra FineLine BGA packages: C-RP# to F-RP#, C-WE# to F-WE#, TM1 to V_{CC}, TM0 to GND, and WP# to V_{CC}.

(6) Connect the FPGA MSEL[] input pins to select the FPP configuration mode. For details, refer to the appropriate FPGA family chapter in the Configuration Handbook.

Multiple FPGAs can be configured using a single enhanced configuration device in FPP mode. In this mode, multiple Stratix series and/or APEX II FPGAs are cascaded together in a daisy chain.

After the first FPGA completes configuration, its *n*CEO pin asserts to activate the second FPGA's *n*CE pin, which prompts the second device to start capturing configuration data. In this setup, the FPGAs CONF_DONE pins are tied together, and hence all devices initialize and enter user mode simultaneously. If the enhanced configuration device or one of the FPGAs detects an error, configuration stops (and simultaneously restarts) for the whole chain because the *n*STATUS pins are tied together.

While Altera FPGAs can be cascaded in a configuration chain, the enhanced configuration devices cannot be cascaded to configure larger devices/chains.

For configuration schematics and more information on multi-device FPP configuration, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

Passive Serial Configuration

Stratix series, Cyclone series, APEX II, APEX 20KC, APEX 20KE, APEX 20K, and FLEX 10K devices can be configured using enhanced configuration devices in the PS mode. This mode is similar to the FPP mode, with the exception that only one bit of data (DATA[0]) is transmitted to the FPGA per DCLK cycle. The remaining DATA[7..1] output pins are unused in this mode and drive low.

The configuration schematic for PS configuration of a single FPGA or single serial chain is identical to the FPP schematic (with the exception that only DATA[0] output from the enhanced configuration device connects to the FPGA DATA0 input pin; remaining DATA[7..1] pins are left floating).

For configuration schematics and more information on multi-device PS configuration, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

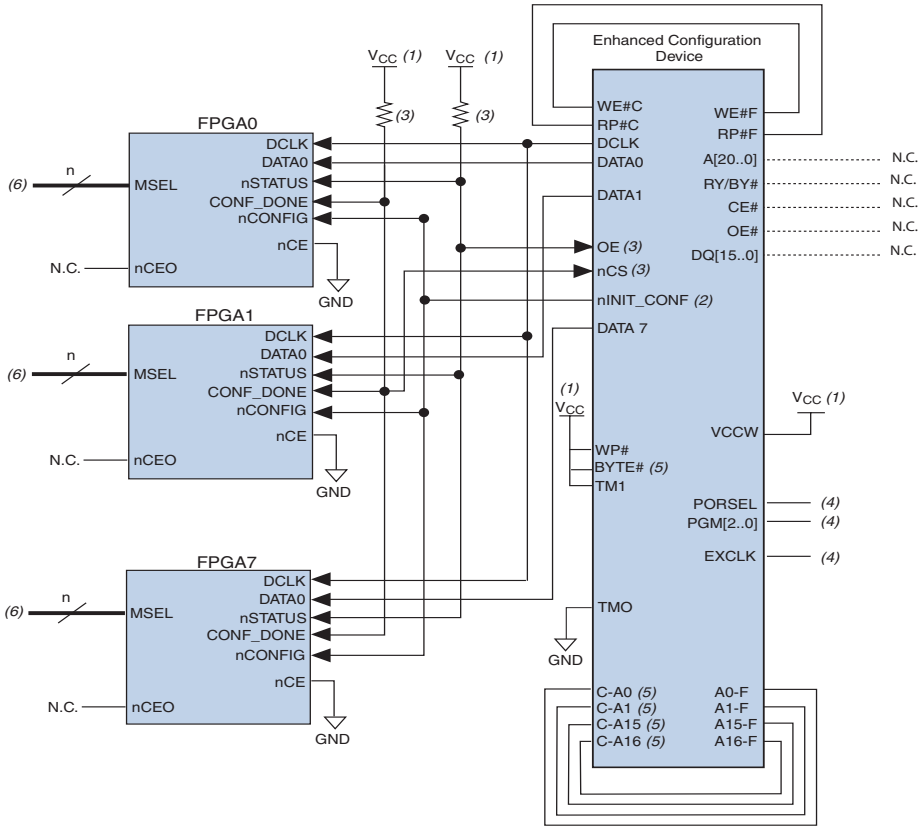
Concurrent Configuration

The enhanced configuration device supports concurrent configuration of multiple FPGAs (or FPGA chains) in PS mode. Concurrent configuration is when the enhanced configuration device simultaneously outputs n bits of configuration data on the DATA[*n*-1..0] pins (*n* = 1, 2, 4, or 8), and each DATA[] line serially configures a different FPGA (chain). The number of concurrent serial chains is user-defined via the Quartus II software and can be any number between 1 and 8. For example, three concurrent chains you can select the 4-bit PS mode, and connect the least significant DATA bits to the FPGAs or FPGA chains. Leave the most significant DATA bit (DATA[3]) unconnected. Similarly, for 5-, 6- or 7-bit concurrent chains you can select the 8-bit PS mode.

Figure 2–3 shows the schematic for configuring multiple FPGAs concurrently in the PS mode using an enhanced configuration device.

For specific details on configuration interface connections including pull-up resistor values, supply voltages, and MSEL pin settings, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

Figure 2–3. Concurrent Configuration of Multiple FPGAs in PS Mode (n = 8)



Notes to Figure 2–3:

(1) Connect V_{CC} to the same supply voltage as the configuration device.

(2) The *n*INIT_CONF pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active. This means an external pull-up resistor is not required on the *n*INIT_CONF/*n*CONFIG line. The *n*INIT_CONF pin does not need to be connected if its functionality is not used. If *n*INIT_CONF is not used, *n*CONFIG must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' OE and *n*CS pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

(4) For PORSEL, PGM[], and EXCLK pin connections, refer to Table 2–8.

(5) In the 100-pin PQFP package, you must externally connect the following pins: C-A0 to F-A0, C-A1 to F-A1, C-A15 to F-A15, C-A16 to F-A16, and BYTE# to V_{CC}. Additionally, you must make the following pin connections in both 100-pin PQFP and 88-pin Ultra FineLine BGA packages: C-RP# to F-RP#, C-WE# to F-WE#, TM1 to V_{CC}, TM0 to GND, and WP# to V_{CC}.

(6) Connect the FPGA MSEL[] input pins to select the PS configuration mode. For details, refer to the appropriate FPGA family chapter in the Configuration Handbook.

Table 2–3 summarizes the concurrent PS configuration modes supported in the enhanced configuration device.

Table 2–3. Enhanced Configuration Devices in PS Mode

Mode Name	Mode (*n* =) *(1)*	Used Outputs	Unused Outputs
Passive serial mode	1	DATA0	DATA[7..1] drive low
Multi-device passive serial mode	2	DATA[1..0]	DATA[7..2] drive low
Multi-device passive serial mode	4	DATA[3..0]	DATA[7..4] drive low
Multi-device passive serial mode	8	DATA[7..0]	-

Note to Table 2–3:
(1) This is the number of valid DATA outputs for each configuration mode.

For configuration schematics and more information on concurrent configurations, refer to the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook,* Volume 2, or refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

External Flash Interface

The enhanced configuration devices support external FPGA or processor access to its flash memory. The unused portions of the flash memory can be used by the external device to store code or data. This interface can also be used in systems that implement remote configuration capabilities. Configuration data within a particular configuration page can be updated via the external flash interface and the system could be reconfigured with the new FPGA image. This interface is also useful to store Nios boot code and/or application code.

For more information on the Stratix remote configuration feature, refer to the *Using Remote System Configuration with Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook*.

The address, data, and control ports of the flash memory are internally connected to the enhanced configuration device controller and to external device pins. An external source can drive these external device pins to access the flash memory when the flash interface is available.

This external flash interface is a shared bus interface with the configuration controller chip. The configuration controller is the primary bus master. Since there is no bus arbitration support, the external device can only access the flash interface when the controller has tri-stated its

internal interface to the flash. Simultaneous access by the controller and the external device will cause contention, and result in configuration and programming failures.

Since the internal flash interface is directly connected to the external flash interface pins, controller flash access cycles will toggle the external flash interface pins. The external device must be able to tri-state its flash interface during these times and ignore transitions on the flash interface pins.

The external flash interface signals cannot be shared between multiple enhanced configuration devices because this causes contention during in-system programming and configuration. During these times, the controller chips inside the enhanced configuration devices are actively accessing flash memory. Therefore, enhanced configuration devices do not support shared flash bus interfaces.

The enhanced configuration device controller chip accesses flash memory during:

- FPGA configuration—reading configuration data from flash
- JTAG-based flash programming—storing configuration data in flash
- At POR—reading option bits from flash

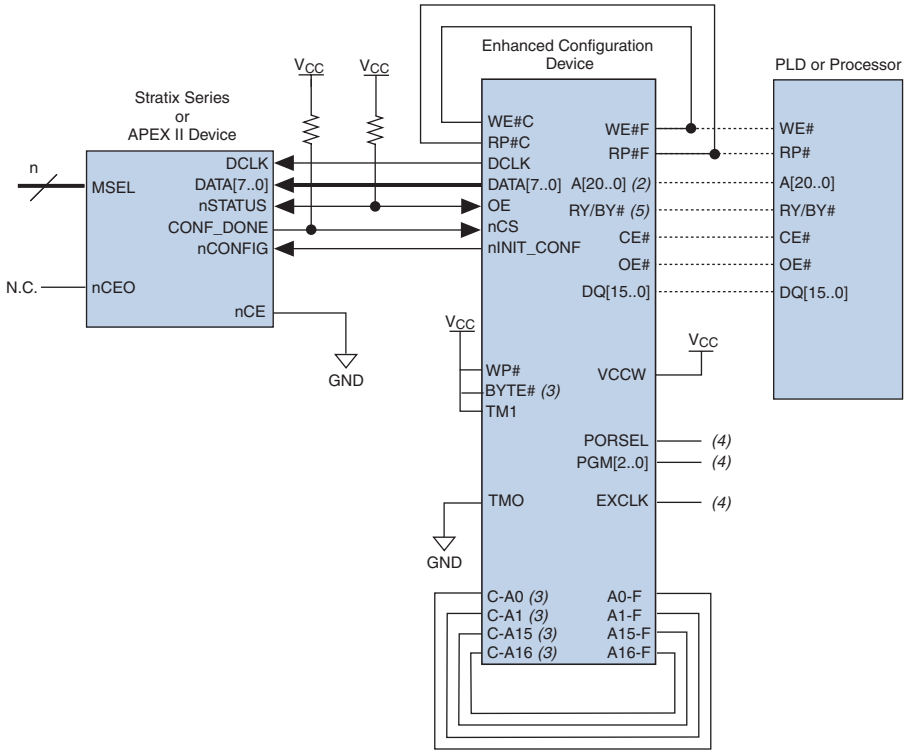
During these times, the external FPGA/processor must tri-state its interface to the flash memory. After configuration and programming, the enhanced configuration device's controller tri-states the internal interface and goes into an idle mode. To interrupt a configuration cycle in order to access the flash via the external flash interface, the external device can hold the FPGA's *n*CONFIG input low. This keeps the configuration device in reset by holding the *n*STATUS-OE line low, allowing external flash access.

For further details on the software support for the external flash interface feature, refer to the *Using Altera Enhanced Configuration Devices* chapter in in the *Configuration Handbook,* Volume 2. For details on flash commands, timing, memory organization, and write protection features, refer to the following documents:

- For Micron flash-based EPC4, refer to the *Micron Flash Memory MT28F400B3 Data Sheet* at **www.micron.com**.
- For Sharp flash-based EPC16, refer to the *Sharp LHF16J06 Data Sheet Flash Memory Used in EPC16 Devices* at **www.sharpsma.com**.
- For the *Intel Advanced Boot Block Flash Memory (B3) 28F008/800B3, 28F016/160B3, 28F320B3, 28F640B3 Datasheet*, visit **www.intel.com**.

Figure 2–4 shows an FPP configuration schematic with the external flash interface in use.

Figure 2–4. FPP Configuration with External Flash Interface *Note (1)*



Notes to Figure 2–4:

(1) For external flash interface support in EPC8 enhanced configuration device, contact Altera Applications.

(2) Pin A20 in EPC16 devices, pins A20 and A19 in EPC8 devices, and pins A20, A19, and A18 in EPC4 devices should be left floating. These pins should not be connected to any signal; they are no-connect pins.

(3) In the 100-pin PQFP package, you must externally connect the following pins: C-A0 to F-A0, C-A1 to F-A1, C-A15 to F-A15, C-A16 to F-A16, and BYTE # to V_{CC}. Additionally, you must make the following pin connections in both 100-pin PQFP and 88-pin Ultra FineLine BGA packages: C-RP# to F-RP#, C-WE# to F-WE#, TM1 to V_{CC}, TM0 to GND, and WP# to V_{CC}.

(4) For PORSEL, PGM[], and EXCLK pin connections, refer to Table 2–8.

(5) RY/BY# pin is only available for Sharp flash-based EPC8 and EPC16.

Protecting Intel Flash in EPC16 Devices

Altera recommends the following steps to protect the Intel flash in EPC16 devices:

1. Sequence the V_{CCW} to less than 1 V during the flash V_{CC} power transitions.
2. Isolate the V_{CCW} from the V_{CC} supply on board.

☞ The lock bit protection is only supported using Sharp flash.

Dynamic Configuration (Page Mode)

The dynamic configuration (or page mode) feature allows the enhanced configuration device to store up to eight different sets of designs for all the FPGAs in your system. You can then choose which page (set of configuration files) the enhanced configuration device should use for FPGA configuration.

Dynamic configuration or the page mode feature enables you to store a minimum of two pages: a factory default or fail-safe configuration, and an application configuration. The fail-safe configuration page could be programmed during system production, while the application configuration page could support remote or local updates. These remote updates could add or enhance system features and performance. However, with remote update capabilities comes the risk of possible corruption of configuration data. In the event of such a corruption, the system could automatically switch to the fail-safe configuration and avoid system downtime.

The enhanced configuration device page mode feature works with the Stratix Remote System Configuration feature, to enable intelligent remote updates to your systems.

For more information on remotely updating Stratix FPGAs, refer to *Using Remote System Configuration with Stratix & Stratix GX Devices* in the *Stratix Device Handbook*.

The three `PGM[2..0]` input pins control which page is used for configuration, and these pins are sampled at the start of each configuration cycle when `OE` goes high. The page mode selection allows you to dynamically reconfigure the functionality of your FPGA(s) by switching the `PGM[2..0]` pins and asserting *n*`CONFIG`. Page 0 is defined as the default page and the `PGM[2]` pin is the most significant bit (MSB).

The PGM[2..0] input pins must not be left floating on your board, regardless of whether this feature is used or not. When this feature is not used, connect the PGM[2..0] pins to GND to select the default page 000.

The enhanced configuration device pages are dynamically sized regions in memory. The start address and length of each page is programmed into the option-bit space of the flash memory during initial programming. All subsequent configuration cycles will sample the PGM[] pins and use the option-bit information to jump to the start of the corresponding configuration page. Each page must have configuration files for all FPGAs in your system that are connected to that enhanced configuration device.

For example, if your system requires three configuration pages and includes two FPGAs, each page will store two SRAM Object Files (**.sof**) for a total of six SOFs in the configuration device.

Furthermore, all enhanced configuration device configuration schemes (PS, FPP, and concurrent PS) are supported with the page-mode feature. The number of pages and/or devices that can be configured using a single enhanced configuration device is only limited by the size of the flash memory.

For detailed information on the page-mode feature implementation and programming file generation steps using Quartus II software, refer to the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook*, Volume 2.

Real-Time Decompression

Enhanced configuration devices support on-chip real time decompression of configuration data. FPGA configuration data is compressed by the Quartus II software and stored in the enhanced configuration device. During configuration, the decompression engine inside the enhanced configuration device will decompress or expand configuration data. This feature increases the effective-configuration density of the enhanced configuration device up to 7, 15, or 30 Mbits in the EPC4, EPC8, and EPC16, respectively.

The enhanced configuration device also supports a parallel 8-bit data bus to the FPGA to reduce configuration time. However, in some cases, the FPGA data-transfer time is limited by the flash-read bandwidth. For example, when configuring an APEX II device in FPP (byte-wide data per cycle) mode at a configuration speed of 66 MHz, the FPGA write bandwidth is equal to 8 bits × 66 MHz = 528 Mbps. The flash read interface, however, is limited to approximately 10 MHz (since the flash

access time is ~90 ns). This translates to a flash-read bandwidth of 16 bits × 10 MHz = 160 Mbps. Hence, the configuration time is limited by the flash-read time.

When configuration data is compressed, the amount of data that needs to be read out of the flash is reduced by about 50%. If 16 bits of compressed data yields 30 bits of uncompressed data, the flash-read bandwidth increases to 30 bits × 10 MHz = 300 Mbps, reducing overall configuration time.

You can enable the controller's decompression feature in the Quartus II software, **Configuration Device Options** window by turning on **Compression Mode**.

The decompression feature supported in the enhanced configuration devices is different from the decompression feature supported by the Stratix II FPGAs and the Cyclone series. When configuring Stratix II FPGAs or the Cyclone series using enhanced configuration devices, Altera recommends enabling decompression in Stratix II FPGAS or the Cyclone series only for faster configuration.

The compression algorithm used in Altera devices is optimized for FPGA configuration bitstreams. Since FPGAs have several layers of routing structures (for high performance and easy routability), large amounts of resources go unused. These unused routing and logic resources as well as un-initialized memory structures result in a large number of configuration RAM bits in the disabled state. Altera's proprietary compression algorithm takes advantage of such bitstream qualities.

The general guideline for effectiveness of compression is the higher the device logic/routing utilization, the lower the compression ratio (where the compression ratio is defined as the original bitstream size divided by the compressed bitstream size).

For Stratix designs, based on a suite of designs with varying amounts of logic utilization, the minimum compression ratio was observed to be 1.9 or a ~47% size reduction for these designs. Table 2–4 shows sample

compression ratios from a suite of Stratix designs. These numbers serve as a guideline (not a specification) to help you allocate sufficient configuration memory to store compressed bitstreams.

Table 2–4. Stratix Compression Ratios Note (1)

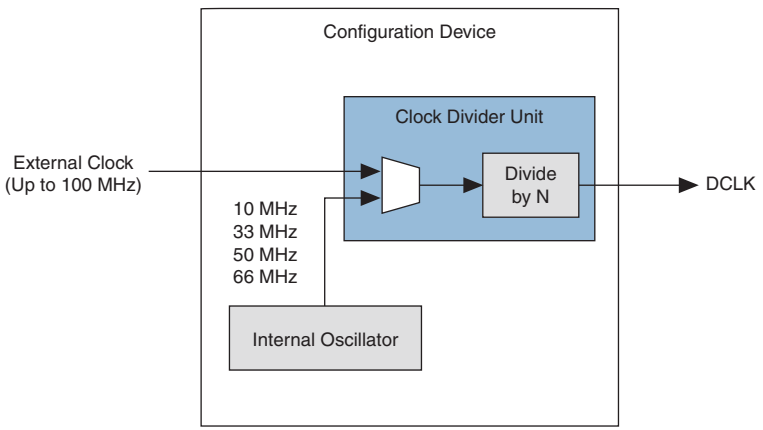
	Minimum	Average
Logic Utilization	98%	64%
Compression Ratio	1.9	2.3
% Size Reduction	47%	57%

Note to Table 2–4:
(1) These numbers are preliminary. They are intended to serve as a guideline, not a specification.

Programmable Configuration Clock

The configuration clock (DCLK) speed is user programmable. One of two clock sources can be used to synthesize the configuration clock; a programmable oscillator or an external clock input pin (EXCLK). The configuration clock frequency can be further synthesized using the clock divider circuitry. This clock can be divided by the N counter to generate your DCLK output. The N divider supports all integer dividers between 1 and 16, as well as a 1.5 divider and a 2.5 divider. The duty cycle for all clock divisions other than non-integer divisions is 50% (for the non-integer dividers, the duty cycle will not be 50%). See Figure 2–5 for a block diagram of the clock divider unit.

Figure 2–5. Clock Divider Unit



The DCLK frequency is limited by the maximum DCLK frequency the FPGA supports.

The maximum DCLK input frequency supported by the FPGA is specified in the appropriate FPGA family chapter in the *Configuration Handbook*.

The controller chip features a programmable oscillator that can output four different frequencies. The various settings generate clock outputs at frequencies as high as 10, 33, 50, and 66 MHz, as shown in Table 2–5.

Table 2–5. Internal Oscillator Frequencies

Frequency Setting	Min (MHz)	Typ (MHz)	Max (MHz)
10	6.4	8.0	10.0
33	21.0	26.5	33.0
50	32.0	40.0	50.0
66	42.0	53.0	66.0

Clock source, oscillator frequency, and clock divider (N) settings can be made in the Quartus II software, by accessing the **Configuration Device Options** inside the **Device Settings** window or the **Convert Programming Files** window. The same window can be used to select between the internal oscillator and the external clock (EXCLK) input pin as your configuration clock source. The default setting selects the internal oscillator at the 10 MHz setting as the clock source, with a divide factor of 1.

For more information on making the configuration clock source, frequency, and divider settings, refer to the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook,* Volume 2.

Flash In-System Programming (ISP)

The flash memory inside enhanced configuration devices can be programmed in-system via the JTAG interface and the external flash interface. JTAG-based programming is facilitated by the configuration controller in the enhanced configuration device. External flash interface programming requires an external processor or FPGA to control the flash.

The enhanced configuration device flash memory supports 100,000 erase cycles.

JTAG-based Programming

The IEEE Std. 1149.1 JTAG Boundary Scan is implemented in enhanced configuration devices to facilitate the testing of its interconnection and functionality. Enhanced configuration devices also support the ISP mode. The enhanced configuration device is compliant with the IEEE Std. 1532 draft 2.0 specification.

The JTAG unit of the configuration controller communicates directly with the flash memory. The controller processes the ISP instructions and performs the necessary flash operations. The enhanced configuration devices support a maximum JTAG TCK frequency of 10 MHz.

During JTAG-based ISP, the external flash interface is not available. Before the JTAG interface programs the flash memory, an optional JTAG instruction (PENDCFG) can be used to assert the FPGA's nCONFIG pin (via the nINIT_CONF pin). This will keep the FPGA in reset and terminate any internal flash access. This function prevents contention on the flash pins when both JTAG ISP and an external FPGA/processor try to access the flash simultaneously. The nINIT_CONF pin is released when the Initiate Configuration (nINIT_CONF) JTAG instruction is updated. As a result, the FPGA is configured with the new configuration data stored in flash.

This function can be added to your programming file in the Quartus II software by enabling the **Initiate configuration after programming** option in the **Programmer options** window (Options menu).

Programming via External Flash Interface

This method allows parallel programming of the flash memory (using the 16-bit data bus). An external processor or FPGA acts as the flash controller and has access to programming data (via a communication link such as UART, Ethernet, and PCI). In addition to the program, erase, and verify operations, the external flash interface supports block/sector protection instructions.

For information on protection commands, areas, and lock bits, refer to the appropriate flash data sheets.

- For Micron flash-based EPC4, refer to the *Micron Flash Memory MT28F400B3 Data Sheet* at **www.micron.com**.
- For Sharp flash-based EPC16, refer to the *Sharp LHF16J06 Data Sheet Flash Memory Used in EPC16 Devices* at **www.sharpsma.com**.
- For the *Intel Advanced Boot Block Flash Memory (B3) 28F008/800B3, 28F016/160B3, 28F320B3, 28F640B3 Datasheet*, visit **www.intel.com**.

External flash interface programming is only allowed when the configuration controller has relinquished flash access (by tri-stating its internal interface). If the controller has not relinquished flash access (during configuration or JTAG-based ISP), you must hold the controller in reset before initiating external programming. The controller can be reset by holding the FPGA *n*CONFIG line at a logic low level. This keeps the controller in reset by holding the *n*STATUS-OE line low, allowing external flash access.

If initial programming of the enhanced configuration device is done in-system via the external flash interface, the controller must be kept in reset by driving the FPGA *n*CONFIG line low to prevent contention on the flash interface.

Pin Description

Tables 2–6 through 2–8 describe the enhanced configuration device pins. These tables include configuration interface pins, external flash interface pins, JTAG interface pins, and other pins.

Table 2–6. Configuration Interface Pins

Pin Name	Pin Type	Description
DATA[7..0]	Output	This is the configuration data output bus. DATA changes on each falling edge of DCLK. DATA is latched into the FPGA on the rising edge of DCLK.
DCLK	Output	The DCLK output pin from the enhanced configuration device serves as the FPGA configuration clock. DATA is latched by the FPGA on the rising edge of DCLK.
*n*CS	Input	The *n*CS pin is an input to the enhanced configuration device and is connected to the FPGA's CONF_DONE signal for error detection after all configuration data is transmitted to the FPGA. The FPGA will always drive *n*CS and OE low when *n*CONFIG is asserted. This pin contains a programmable internal weak pull-up resistor that can be disabled/enabled in the Quartus II software through the **Disable *n*CS and OE pull-ups on configuration device** option.

Table 2–6. Configuration Interface Pins

Pin Name	Pin Type	Description
*n*INIT_CONF	Open-Drain Output	The *n*INIT_CONF pin can be connected to the *n*CONFIG pin on the FPGA to initiate configuration from the enhanced configuration device via a private JTAG instruction. This pin contains an internal weak pull-up resistor that is always active. The INIT_CONF pin does not need to be connected if its functionality is not used. If *n*INIT_CONF is not used, *n*CONFIG must be pulled to V_{CC} either directly or through a pull-up resistor.
OE	Open-Drain Bidirectional	This pin is driven low when POR is not complete. A user-selectable 2-ms or 100-ms counter holds off the release of OE during initial power up to permit voltage levels to stabilize. POR time can be extended by externally holding OE low. OE is connected to the FPGA *n*STATUS signal. After the enhanced configuration device controller releases OE, it waits for the *n*STATUS-OE line to go high before starting the FPGA configuration process. This pin contains a programmable internal weak pull-up resistor that can be disabled/enabled in the Quartus II software through the **Disable *n*CS and OE pull-ups on configuration device** option.

Table 2–7. External Flash Interface Pins (Part 1 of 3)

Pin Name	Pin Type	Description
A[20..0]	Input	These pins are the address input to the flash memory for read and write operations. The addresses are internally latched during a write cycle. When the external flash interface is not used, leave these pins floating (with the few exceptions noted below). These flash address, data, and control pins are internally connected to the configuration controller. In the 100-pin PQFP package, four address pins (A0, A1, A15, A16) are not internally connected to the controller. These loop-back connections must be made on the board between the C-A[] and F-A[] pins even when not using the external flash interface. All other address pins are connected internal to the package. All address pins are connected internally in the 88-pin Ultra FineLine BGA package. Pin A20 in EPC16 devices, pins A20 and A19 in EPC8 devices, and pins A20, A19, and A18 in EPC4 devices are no-connects. These pins should be left floating on the board.
DQ[15..0]	Bidirectional	This is the flash data bus interface between the flash memory and the controller. The controller or an external source drives DQ[15..0] during the flash command and the data write bus cycles. During the data read cycle, the flash memory drives the DQ[15..0] to the controller or external device. Leave these pins floating on the board when the external flash interface is not used.
CE#	Input	Active low flash input pin that activates the flash memory when asserted. When it is high, it deselects the device and reduces power consumption to standby levels. This flash input pin is internally connected to the controller. Leave this pin floating on the board when the external flash interface is not used.
RP# *(1)*	Input	Active low flash input pin that resets the flash when asserted. When high, it enables normal operation. When low, it inhibits write operation to the flash memory, which provides data protection during power transitions. This flash input is not internally connected to the controller. Hence, an external loop-back connection between C-RP# and F-RP# must be made on the board even when you are not using the external flash interface. When using the external flash interface, connect the external device to the RP# pin with the loop back. Tri-state RP# at all times when the flash is not in use.

Table 2–7. External Flash Interface Pins (Part 2 of 3)

Pin Name	Pin Type	Description
OE#	Input	Active-low flash-control input that is asserted by the controller or external device during flash read cycles. When asserted, it enables the drivers of the flash output pins. Leave this pin floating on the board when the external flash interface is not used.
WE# *(1)*	Input	Active-low flash-write strobe asserted by the controller or external device during flash write cycles. When asserted, it controls writes to the flash memory. In the flash memory, addresses and data are latched on the rising edge of the `WE#` pulse. This flash input is not internally connected to the controller. Hence, an external loop-back connection between `C-WE#` and `F-WE#` must be made on the board even when you are not using the external flash interface. When using the external flash interface, connect the external device to the `WE#` pin with the loop back.
WP#	Input	This pin is usually tied to V_{CC} or ground on the board. The controller does not drive this pin because it could cause contention. Connection to V_{CC} is recommended for faster block erase/programming times and to allow programming of the flash-bottom boot block, which is required when programming the device using the Quartus II software. This pin should be connected to V_{CC} even when the external flash interface is not used.
VCCW	Supply	Block erase, full-chip erase, word write, or lock-bit configuration power supply. Connect this pin to the 3.3-V V_{CC} supply, even when you are not using the external flash interface.
RY/BY#	Output	Flash asserts this pin when a write or erase operation is complete. This pin is not connected to the controller. RY/BY# is only available in Sharp flash-based EPC8 and EPC16.*(2)* Leave this pin floating when the external flash interface is not used.

Table 2–7. External Flash Interface Pins (Part 3 of 3)

Pin Name	Pin Type	Description
BYTE#	Input	This is a flash byte-enable pin and is only available for enhanced configuration devices in the 100-pin PQFP package. This pin must be connected to V_{CC} on the board even when you are not using the external flash interface (the controller uses the flash in 16-bit mode).

Note to Table 2–7:

(1) These pins can be driven to 12 V during production testing of the flash memory. Since the controller cannot tolerate the 12-V level, connections from the controller to these pins are not made internal to the package. Instead they are available as two separate pins. You must connect the two pins at the board level (for example, on the printed circuit board (PCB), connect the C-WE# pin from controller to F-WE# pin from the flash memory).

(2) For details, please refer to Process Change Notification *PCN0506: Addition of Intel Flash Memory As Source For EPC4, EPC8 & EPC16 Enhanced Configuration Devices* and *Using Intel Flash Memory Based EPC4, EPC8 and EPC16* white paper.

Table 2–8. JTAG Interface Pins and Other Required Controller Pins (Part 1 of 2)

Pin Name	Pin Type	Description
TDI	Input	This is the JTAG data input pin. Connect this pin to V_{CC} if the JTAG circuitry is not used.
TDO	Output	This is the JTAG data output pin. Do not connect this pin if the JTAG circuitry is not used (leave floating).
TCK	Input	This is the JTAG clock pin. Connect this pin to GND if the JTAG circuitry is not used.
TMS	Input	This is the JTAG mode select pin. Connect this pin to V_{CC} if the JTAG circuitry is not used.
PGM[2..0]	Input	These three input pins select one of the eight pages of configuration data to configure the FPGA(s) in the system. Connect these pins on the board to select the page specified in the Quartus II software when generating the enhanced configuration device POF. PGM[2] is the MSB. The default selection is page 0; PGM[2..0]=000. These pins must not be left floating.
EXCLK	Input	Optional external clock input pin that can be used to generate the configuration clock (DCLK). When an external clock source is not used, connect this pin to a valid logic level (high or low) to prevent a floating-input buffer.

Table 2–8. JTAG Interface Pins and Other Required Controller Pins (Part 2 of 2)

Pin Name	Pin Type	Description
PORSEL	Input	This pin selects a 2-ms or 100-ms POR counter delay during power up. When PORSEL is low, POR time is 100-ms. When PORSEL is high, POR time is 2 ms. This pin must be connected to a valid logic level.
TM0	Input	For normal operation, this test pin must be connected to GND.
TM1	Input	For normal operating, this test pin must be connected to V_{CC}.

Power-On Reset

The POR circuit keeps the system in reset until power-supply voltage levels have stabilized. The POR time consists of the V_{CC} ramp time and a user-programmable POR delay counter. When the supply is stable and the POR counter expires, the POR circuit releases the OE pin. The POR time can be further extended by an external device by driving the OE pin low.

Do not execute JTAG or ISP instructions until POR is complete.

The enhanced configuration device supports a programmable POR delay setting. You can set the POR delay to the default 100-ms setting or reduce the POR delay to 2 ms for systems that require fast power-up. The PORSEL input pin controls this POR delay; a logic-high level selects the 2-ms delay, while a logic-low level selects the 100-ms delay.

The enhanced configuration device can enter reset under the following conditions:

- The POR reset starts at initial power-up during V_{CC} ramp-up or if V_{CC} drops below the minimum operating condition anytime after V_{CC} has stabilized
- The FPGA initiates reconfiguration by driving *n*STATUS low, which occurs if the FPGA detects a CRC error or if the FPGA's *n*CONFIG input pin is asserted
- The controller detects a configuration error and asserts OE to initiate re-configuration of the Altera FPGA (for example when CONF_DONE stays low after all configuration data has been transmitted)

Power Sequencing

Altera requires that you power-up the FPGA's V_{CCINT} supply before the enhanced configuration device's POR expires.

Power up needs to be controlled so that the enhanced configuration device's OE signal goes high after the CONF_DONE signal is pulled low. If the EEPC device exits POR before the FPGA is powered up, the CONF_DONE signal will be high since the pull-up resistor is holding this signal high. When the enhanced configuration device exits POR, OE is released and pulled high by a pull-up resistor. Since the enhanced configuration device samples the *n*CS signal on the rising edge of OE, it detects a high level on CONF_DONE and enters an idle mode. DATA and DCLK outputs will not toggle in this state and configuration will not begin. The enhanced configuration device will only exit this mode if it is powered down and then powered up correctly.

To ensure the enhanced configuration device enters configuration mode properly, you need to ensure that the FPGA completes power-up before the enhanced configuration device exits POR.

The pin-selectable POR time feature is useful for ensuring this power-up sequence. The enhanced configuration device has two POR settings, 2 ms when PORSEL is set to a high level and 100 ms when PORSEL is set to a low level. For more margin, the 100-ms setting can be selected to allow the FPGA to power-up before configuration is attempted.

Alternatively, a power-monitoring circuit or a power-good signal can be used to keep the FPGA's *n*CONFIG pin asserted low until both supplies have stabilized. This ensures the correct power up sequence for successful configuration.

Programming & Configuration File Support

The Quartus II development software provides programming support for the enhanced configuration device and automatically generates the POF files for the EPC4, EPC8, and EPC16 devices. In a multi-device project, the software can combine the SOF files for multiple Stratix series, Cyclone series, APEX II, APEX 20K, Mercury, ACEX 1K, and FLEX 10K FPGAs into one programming file for the enhanced configuration device.

Refer to the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook*, Volume 2, or refer to the *Software Settings* section in the *Configuration Handbook* for details on generating programming files.

Enhanced configuration devices can be programmed in-system through its industry-standard 4-pin JTAG interface. The ISP feature in the enhanced configuration device provides ease in prototyping and updating FPGA functionality.

After programming an enhanced configuration device in-system, FPGA configuration can be initiated by including the enhanced configuration device's JTAG INIT_CONF instruction (Table 2–9).

The ISP circuitry in the enhanced configuration device is compliant with the IEEE Std. 1532 specification. The IEEE Std. 1532 is a standard that allows concurrent ISP between devices from multiple vendors.

Table 2–9. Enhanced Configuration Device JTAG Instructions (Part 1 of 2) *Note (1)*

JTAG Instruction	OPCODE	Description
SAMPLE/PRELOAD	00 0101 0101	Allows a snapshot of the state of the enhanced configuration device pins to be captured and examined during normal device operation and permits an initial data pattern output at the device pins.
EXTEST	00 0000 0000	Allows the external circuitry and board-level interconnections to be tested by forcing a test pattern at the output pins and capturing results at the input pins.
BYPASS	11 1111 1111	Places the 1-bit bypass register between the TDI and the TDO pins, which allow the BST data to pass synchronously through a selected device to adjacent devices during normal device operation.
IDCODE	00 0101 1001	Selects the device IDCODE register and places it between TDI and TDO, allowing the device IDCODE to be serially shifted out to TDO. The device IDCODE for all enhanced configuration devices is the same and shown below: 0100A0DDh
USERCODE	00 0111 1001	Selects the USERCODE register and places it between TDI and TDO, allowing the USERCODE to be serially shifted out the TDO. The 32-bit USERCODE is a programmable user-defined pattern.
INIT_CONF	00 0110 0001	This function initiates the FPGA re-configuration process by pulsing the *n*INIT_CONF pin low, which is connected to the FPGA(s) *n*CONFIG pin(s). After this instruction is updated, the *n*INIT_CONF pin is pulsed low when the JTAG state machine enters Run-Test/Idle state. The *n*INIT_CONF pin is then released and *n*CONFIG is pulled high by the resistor after the JTAG state machine goes out of Run-Test/Idle state. The FPGA configuration starts after *n*CONFIG goes high. As a result, the FPGA is configured with the new configuration data stored in flash via ISP. This function can be added to your programming file (POF, JAM, JBC) in the Quartus II software by enabling the **Initiate configuration after programming** option in the **Programmer options** window (Options menu).

Table 2–9. Enhanced Configuration Device JTAG Instructions (Part 2 of 2) *Note (1)*

JTAG Instruction	OPCODE	Description
PENDCFG	00 0110 0101	This optional function can be used to hold the *n*INIT_CONF pin low during JTAG-based ISP of the enhanced configuration device. This feature is useful when the external flash interface is controlled by an external FPGA/processor. This function prevents contention on the flash pins when both the controller and external device try to access the flash simultaneously. Before the enhanced configuration device's controller can access the flash memory, the external FPGA/processor needs to tri-state its interface to flash.This can be ensured by resetting the FPGA using the *n*INIT_CONF, which drives the *n*CONFIG pin and keeps the external FPGA/processor in the "reset" state. The *n*INIT_CONF pin is released when the Initiate Configuration (INIT_CONF) JTAG instruction is issued.

Note to Table 2–9:

(1) Enhanced configuration device instruction register length is 10 and boundary scan length is 174.

For more information on the enhanced configuration device JTAG support, refer to the BSDL files provided at the Altera web site.

Enhanced configuration devices can also be programmed by third-party flash programmers or on-board processors using the external flash interface. Programming files (POF) can be converted to an Intel HEX format file (**.hexout**) using the Quartus II **Convert Programming Files** utility, for use with the programmers or processors.

You can also program the enhanced configuration devices using the Quartus II software, the Altera Programming Unit (APU), and the appropriate configuration device programming adapter. Table 2–10 shows which programming adapter to use with each enhanced configuration device.

Table 2–10. Programming Adapters

Device	Package	Adapter
EPC16	88-pin Ultra FineLine BGA	PLMUEPC-88
	100-pin PQFP	PLMQEPC-100
EPC8	100-pin PQFP	PLMQEPC-100
EPC4	100-pin PQFP	PLMQEPC-100

IEEE Std. 1149.1 (JTAG) Boundary-Scan

The enhanced configuration device provides JTAG BST circuitry that complies with the IEEE Std. 1149.1-1990 specification. JTAG boundary-scan testing can be performed before or after configuration, but not during configuration.

Figure 2–6 shows the timing requirements for the JTAG signals.

Figure 2–6. JTAG Timing Waveforms

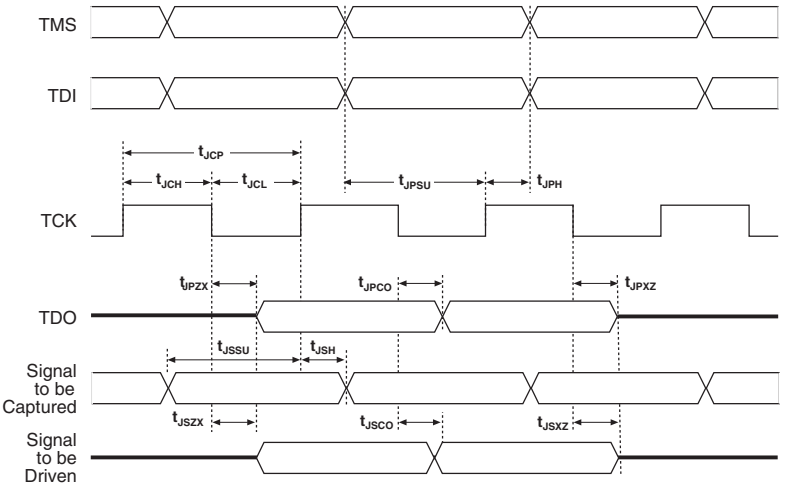


Table 2–11 shows the timing parameters and values for the enhanced configuration device.

Table 2–11. JTAG Timing Parameters & Values (Part 1 of 2)

Symbol	Parameter	Min	Max	Unit
t_{JCP}	TCK clock period	100		ns
t_{JCH}	TCK clock high time	50		ns
t_{JCL}	TCK clock low time	50		ns
t_{JPSU}	JTAG port setup time	20		ns
t_{JPH}	JTAG port hold time	45		ns
t_{JPCO}	JTAG port clock output		25	ns
t_{JPZX}	JTAG port high impedance to valid output		25	ns
t_{JPXZ}	JTAG port valid output to high impedance		25	ns
t_{JSSU}	Capture register setup time	20		ns

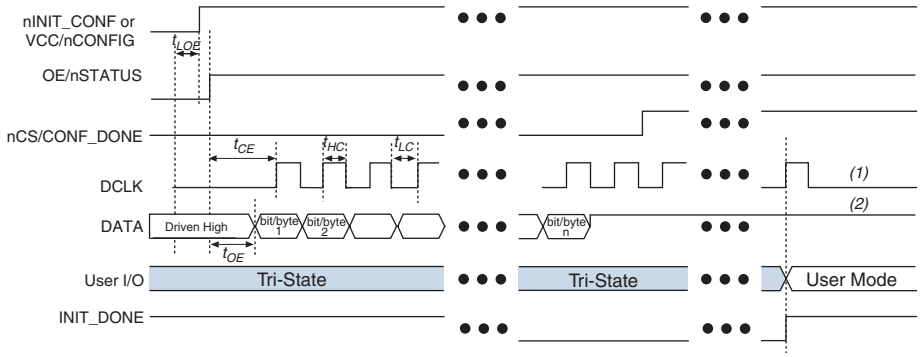
Table 2–11. JTAG Timing Parameters & Values (Part 2 of 2)

Symbol	Parameter	Min	Max	Unit
t_{JSH}	Capture register hold time	45		ns
t_{JSCO}	Update register clock to output		25	ns
t_{JSZX}	Update register high-impedance to valid output		25	ns
t_{JSXZ}	Update register valid output to high impedance		25	ns

Timing Information

Figure 2–7 shows the configuration timing waveform when using an enhanced configuration device.

Figure 2–7. Configuration Timing Waveform Using an Enhanced Configuration Device



Notes to Figure 2–7:
(1) The enhanced configuration device will drive DCLK low after configuration.
(2) The enhanced configuration device will drive DATA[] high after configuration.

Table 2–12 defines the timing parameters when using the enhanced configuration devices.

For flash memory (external flash interface) timing information, please refer to the appropriate flash data sheet on the Altera web site at **www.altera.com**.

- For Micron flash-based EPC4, refer to the *Micron MT28F400B3 Data Sheet Flash Memory Used in EPC4 Devices*
- For Sharp flash-based EPC16, refer to the *Sharp LHF16J06 Data Sheet Flash Memory Used in EPC16 Devices*
- For Intel flash-based EPC4 and EPC16, refer to Intel Flash 28F016B3

Table 2–12. Enhanced Configuration Device Configuration Parameters

Symbol	Parameter	Condition	Min	Typ	Max	Unit
f_{DCLK}	DCLK frequency	40% duty cycle			66.7	MHz
t_{DCLK}	DCLK period		15			ns
t_{HC}	DCLK duty cycle high time	40% duty cycle	6			ns
t_{LC}	DCLK duty cycle low time	40% duty cycle	6			ns
t_{CE}	OE to first DCLK delay		40			ns
t_{OE}	OE to first DATA available		40			ns
t_{OH}	DCLK rising edge to DATA change		*(1)*			ns
t_{CF} *(2)*	OE assert to DCLK disable delay		277			ns
t_{DF} *(2)*	OE assert to DATA disable delay		277			ns
t_{RE} *(3)*	DCLK rising edge to OE		60			ns
t_{LOE}	OE assert time to assure reset		60			ns
f_{ECLK}	EXCLK input frequency	40% duty cycle			100	MHz
t_{ECLK}	EXCLK input period		10			ns
t_{ECLKH}	EXCLK input duty cycle high time	40% duty cycle	4			ns
t_{ECLKL}	EXCLK input duty cycle low time	40% duty cycle	4			ns
t_{ECLKR}	EXCLK input rise time	100 MHz			3	ns
t_{ECLKF}	EXCLK input fall time	100 MHz			3	ns
t_{POR} *(4)*	POR time	2 ms	1	2	3	ms
		100 ms	70	100	120	ms

Notes to Table 2–12:

(1) To calculate t_{OH}, use the following equation: t_{OH} = 0.5 (DCLK period) - 2.5 ns.

(2) This parameter is used for CRC error detection by the FPGA.

(3) This parameter is used for CONF_DONE error detection by the enhanced configuration device.

(4) The FPGA V_{CCINT} ramp time should be less than 1-ms for 2-ms POR, and it should be less than 70 ms for 100-ms POR.

Operating Conditions

Tables 2–13 through 2–17 provide information on absolute maximum ratings, recommended operating conditions, DC operating conditions, supply current values, and pin capacitance data for the enhanced configuration devices.

Table 2–13. Enhanced Configuration Device Absolute Maximum Rating

Symbol	Parameter	Condition	Min	Max	Unit
V_{CC}	Supply voltage	With respect to ground	-0.2	4.6	V
V_I	DC input voltage	With respect to ground	-0.5	3.6	V
I_{MAX}	DC V_{CC} or ground current			100	mA
I_{OUT}	DC output current, per pin		-25	25	mA
P_D	Power dissipation			360	mW
T_{STG}	Storage temperature	No bias	-65	150	C
T_{AMB}	Ambient temperature	Under bias	-65	135	C
T_J	Junction temperature	Under bias		135	C

Table 2–14. Enhanced Configuration Device Recommended Operating Conditions

Symbol	Parameter	Condition	Min	Max	Unit
V_{CC}	Supplies voltage for 3.3-V operation		3.0	3.6	V
V_I	Input voltage	With respect to ground	–0.3	V_{CC} + 0.3	V
V_O	Output voltage		0	V_{CC}	V
T_A	Operating temperature	For commercial use	0	70	C
		For industrial use	–40	85	C
T_R	Input rise time			20	ns
T_F	Input fall time			20	ns

Table 2–15. Enhanced Configuration Device DC Operating Conditions (Part 1 of 2)

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{CC}	Supplies voltage to core		3.0	3.3	3.6	V
V_{IH}	High-level input voltage		2.0		V_{CC} + 0.3	V
V_{IL}	Low-level input voltage				0.8	V

Table 2–15. Enhanced Configuration Device DC Operating Conditions (Part 2 of 2)

Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{OH}	3.3-V mode high-level TTL output voltage	I_{OH} = –4 mA	2.4			V
	3.3-V mode high-level CMOS output voltage	I_{OH} = –0.1 mA	V_{CC} – 0.2			V
V_{OL}	Low-level output voltage TTL	I_{OL} = –4 mA DC			0.45	V
	Low-level output voltage CMOS	I_{OL} = –0.1 mA DC			0.2	V
I_I	Input leakage current	V_I = V_{CC} or ground	–10		10	µA
I_{OZ}	Tri-state output off-state current	V_O = V_{CC} or ground	–10		10	µA
R_{CONF}	Configuration pins	Internal pull up (OE, *n*CS, *n*INIT, CONF)		6		kΩ

Table 2–16. Enhanced Configuration Device I_{CC} Supply Current Values

Symbol	Parameter	Condition	Min	Typ	Max	Unit
I_{CC0}	Current (standby)			50	150	µA
I_{CC1}	V_{CC} supply current (during configuration)			60	90	mA
I_{CCW}	V_{CCW} supply current			*(1)*	*(1)*	

Note to Table 2–16:
(1) For V_{CCW} supply current information, refer to the appropriate flash memory data sheet at **www.altera.com**.

Table 2–17. Enhanced Configuration Device Capacitance

Symbol	Parameter	Condition	Min	Max	Unit
CIN	Input pin capacitance			10	pF
COUT	Output pin capacitance			10	pF

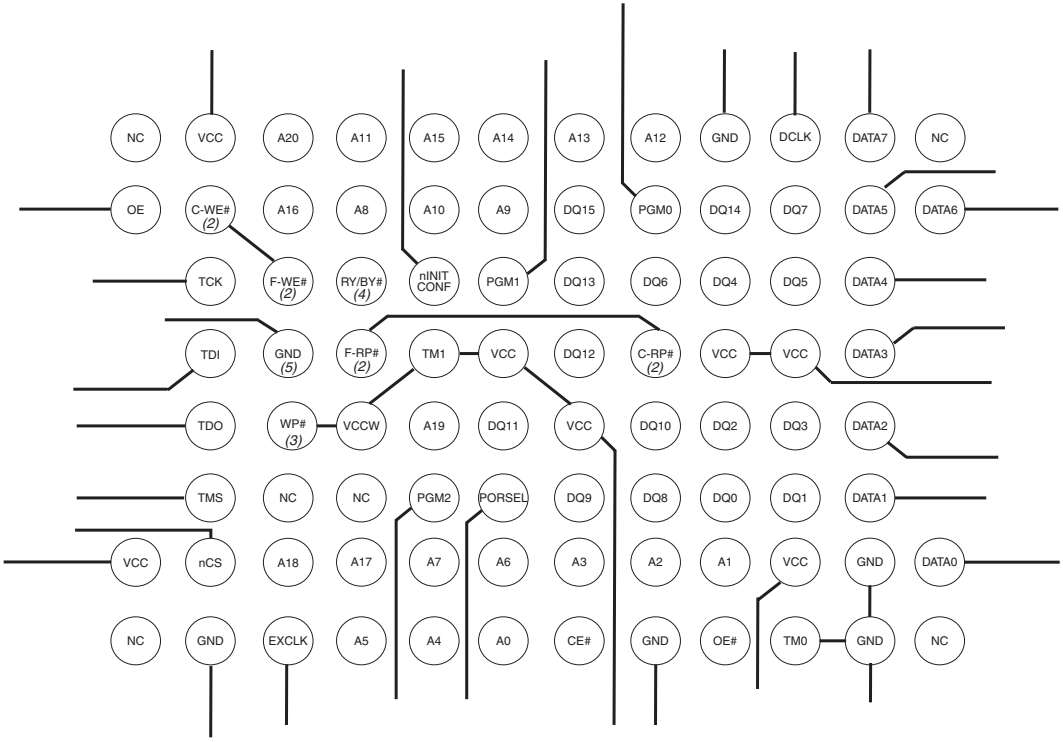
Package

The EPC16 enhanced configuration device is available in both the 88-pin Ultra FineLine BGA package and the 100-pin PQFP package. The Ultra FineLine BGA package, which is based on 0.8-mm ball pitch, maximizes board space efficiency. A board can be laid out for this package using a single PCB layer. The EPC8 and EPC4 devices are available in the 100-pin PQFP package.

Enhanced configuration devices support vertical migration in the 100-pin PQFP package.

Figure 2–8 shows the PCB routing for the 88-pin Ultra FineLine BGA package. The Gerber file for this layout is on the Altera web site.

Figure 2–8. PCB Routing for 88-Pin Ultra FineLine BGA Package *Note (1)*



Notes to Figure 2–8:

(1) If the external flash interface feature is not used, then the flash pins should be left unconnected since they are internally connected to the controller unit. The only pins that need external connections are `WP#`, `WE#`, and `RP#`. If the flash is being used as an external memory source, then the flash pins should be connected as outlined in the pin descriptions section.

(2) `F-RP#` and `F-WE#` are pins on the flash die. `C-RP#` and `C-WE#` are pins on the controller die. `C-WE#` and `F-WE#` should be connected together on the PCB. `F-RP#` and `C-RP#` should also be connected together on the PCB.

(3) `WP#` (write protection pin) should be connected to a high level (3.3 V) to be able to program the flash bottom boot block, which is required when programming the device using the Quartus II software.

(4) RY/BY# is only available in Sharp flash-based enhanced configuration devices.

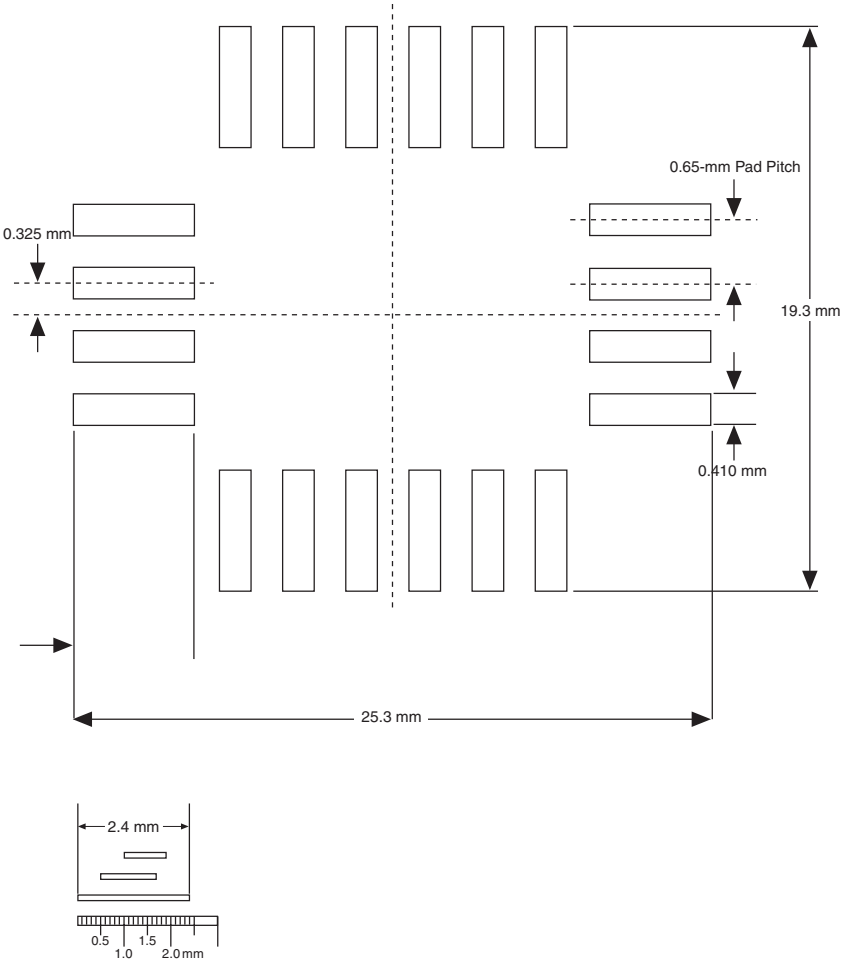
(5) Pin D3 is a NC pin for Intel Flash-based EPC16.

Package Layout Recommendation

Sharp flash-based EPC16 and EPC8 enhanced configuration devices in the 100-pin PQFP packages have different package dimensions than other Altera 100-pin PQFP devices (including the Micron flash-based EPC4,

Intel flash-based EPC16, EPC8 and EPC4). Figure 2–9 shows the 100-pin PQFP PCB footprint specifications for enhanced configuration devices that allows for vertical migration between all devices. These footprint dimensions are based on vendor-supplied package outline diagrams.

Figure 2–9. Enhanced Configuration Device PCB Footprint Specifications for 100-Pin PQFP Packages *Notes (1), (2)*



Notes to Figure 2–9:
(1) Used 0.5-mm increase for front and back of nominal foot length.
(2) Used 0.3-mm increase to maximum foot width.

For package outline drawings, refer to the *Altera Device Package Information Data Sheet*.

Device Pin-Outs

For pin-out information, see the Altera web site at **www.altera.com**.

Ordering Codes

Table 2–18 shows the ordering codes for EPC4, EPC8, and EPC16 enhanced configuration devices.

Table 2–18. Enhanced Configuration Device Ordering Codes

Device	Package	Temperature	Ordering Code
EPC4	100-pin PQFP	Commercial	EPC4QC100
EPC4	100-pin PQFP	Industrial	EPC4QI100
EPC8	100-pin PQFP	Commercial	EPC8QC100
EPC8	100-pin PQFP	Industrial	EPC8QI100
EPC16	100-pin PQFP	Commercial	EPC16QC100
EPC16	100-pin PQFP	Industrial	EPC16QI100
EPC16	88-pin UBGA	Commercial	EPC16UC88

Document Revision History

Table 2–19 shows the revision history for this document.

Table 2–19. Document Revision History

Date & Document Version	Changes Made	Summary of Changes
May 2007 v2.4	Added "Protecting Intel Flash in EPC16 Devices" section.	Added procedure about protecting Intel flash in EPC16 devices.
April 2007 v2.3	Added document revision history.	
October 2005 v2.2	Made changes to content.	
July 2004 v2.0	• Added Stratix II and Cyclone II device information throughout chapter. • Updated VCCW connection in Figures 2–2, 2–3, and 2–4. • Updated *Note (1)* of Figures 2–2, 2–3, and 2–4. • Updated *Note (4)* of Table 2–12. • Updated unit of ICC0 in Table 2–16. • Added ICCW to Table 2–16.	
September 2003 v1.0	Initial Release.	