



# Section I. Cyclone FPGA Family Data Sheet

This section provides designers with the data sheet specifications for Cyclone devices. The chapters contain feature definitions of the internal architecture, configuration and JTAG boundary-scan testing information, DC operating conditions, AC timing parameters, a reference to power consumption, and ordering information for Cyclone devices.

This section contains the following chapters:

- [Chapter 1. Introduction](#)
- [Chapter 2. Cyclone Architecture](#)
- [Chapter 3. Configuration & Testing](#)
- [Chapter 4. DC & Switching Characteristics](#)
- [Chapter 5. Reference & Ordering Information](#)

## Revision History

The table below shows the revision history for [Chapters 1](#) through [5](#).

Chapter(s)	Date / Version	Changes Made
1	October 2003 v1.2	Added 64-bit PCI support information.
	September 2003 v1.1	Updated LVDS data rates to 640 Mbps from 311 Mbps. Updated RSDS feature information.
	May 2003 v1.0	Added document to Cyclone Device Handbook.
2	October 2003 v1.2	Updated phase shift information. Added 64-bit PCI support information.
	September 2003 v1.1	Updated LVDS data rates to 640 Mbps from 311 Mbps.
	May 2003 v1.0	Added document to Cyclone Device Handbook.
3	May 2003 v1.0	Added document to Cyclone Device Handbook.

Chapter(s)	Date / Version	Changes Made
4	January 2004 v.1.3	Added extended-temperature grade device information. Updated <a href="#">Table 4–2</a> . Updated ICC0 information in <a href="#">Table 4–3</a> .
	October 2003 v.1.2	Added clock tree information in <a href="#">Table 4–19</a> . Finalized timing information for EP1C3 and EP1C12 devices. Updated timing information in <a href="#">Tables 4–25</a> through <a href="#">4–26</a> and <a href="#">Tables 4–30</a> through <a href="#">4–51</a> . Updated PLL specifications in <a href="#">Table 4–52</a> .
	July 2003 v1.1	Updated timing information. Timing finalized for EP1C6 and EP1C20 devices. Updated performance information. Added “ <a href="#">PLL Timing</a> ” section.
	May 2003 v1.0	Added document to Cyclone Device Handbook.
5	May 2003 v1.0	Added document to Cyclone Device Handbook.

## Introduction

The Cyclone™ field programmable gate array family is based on a 1.5-V, 0.13- $\mu$ m, all-layer copper SRAM process, with densities up to 20,060 logic elements (LEs) and up to 288 Kbits of RAM. With features like phase-locked loops (PLLs) for clocking and a dedicated double data rate (DDR) interface to meet DDR SDRAM and fast cycle RAM (FCRAM) memory requirements, Cyclone devices are a cost-effective solution for data-path applications. Cyclone devices support various I/O standards, including LVDS at data rates up to 640 megabits per second (Mbps), and 66- and 33-MHz, 64- and 32-bit peripheral component interconnect (PCI), for interfacing with and supporting ASSP and ASIC devices. Altera also offers new low-cost serial configuration devices to configure Cyclone devices.

The following shows the main sections in the Cyclone FPGA Family Data Sheet:

Section	Page
Features . . . . .	1-2
Functional Description . . . . .	2-1
Logic Array Blocks . . . . .	2-3
Logic Elements . . . . .	2-5
MultiTrack Interconnect . . . . .	2-12
Embedded Memory . . . . .	2-18
Global Clock Network & Phase-Locked Loops . . . . .	2-29
I/O Structure . . . . .	2-39
Power Sequencing & Hot Socketing . . . . .	2-55
IEEE Std. 1149.1 (JTAG) Boundary Scan Support . . . . .	3-1
SignalTap II Embedded Logic Analyzer . . . . .	3-5
Configuration . . . . .	3-5
Operating Conditions . . . . .	4-1
Power Consumption . . . . .	4-8
Timing Model . . . . .	4-9
Software . . . . .	5-1
Device Pin-Outs . . . . .	5-1
Ordering Information . . . . .	5-1

## Features

The Cyclone device family offers the following features:

- 2,910 to 20,060 LEs, see [Table 1-1](#)
- Up to 294,912 RAM bits (36,864 bytes)
- Supports configuration through low-cost serial configuration device
- Support for LVTTTL, LVCMOS, SSTL-2, and SSTL-3 I/O standards
- Support for 66- and 33-MHz, 64- and 32-bit PCI standard
- High-speed (640 Mbps) LVDS I/O support
- Low-speed (311 Mbps) LVDS I/O support
- 311-Mbps RSDS I/O support
- Up to two PLLs per device provide clock multiplication and phase shifting
- Up to eight global clock lines with six clock resources available per logic array block (LAB) row
- Support for external memory, including DDR SDRAM (133 MHz), FCRAM, and single data rate (SDR) SDRAM
- Support for multiple intellectual property (IP) cores, including Altera® MegaCore® functions and Altera Megafunctions Partners Program (AMPP<sup>SM</sup>) megafunctions.

**Table 1-1. Cyclone Device Features**

Feature	EP1C3	EP1C4	EP1C6	EP1C12	EP1C20
LEs	2,910	4,000	5,980	12,060	20,060
M4K RAM blocks (128 × 36 bits)	13	17	20	52	64
Total RAM bits	59,904	78,336	92,160	239,616	294,912
PLLs	1	2	2	2	2
Maximum user I/O pins (1)	104	301	185	249	301

**Note to Table 1-1:**

(1) This parameter includes global clock pins.

Cyclone devices are available in quad flat pack (QFP) and space-saving FineLine BGA® packages (see [Table 1-2](#) through [1-3](#)).

**Table 1-2. Cyclone Package Options & I/O Pin Counts**

Device	100-Pin TQFP (1)	144-Pin TQFP (1), (2)	240-Pin PQFP (1)	256-Pin FineLine BGA	324-Pin FineLine BGA	400-Pin FineLine BGA
EP1C3	65	104				
EP1C4					249	301
EP1C6		98	185	185		
EP1C12			173	185	249	
EP1C20					233	301

**Notes to Table 1-2:**

- (1) TQFP: thin quad flat pack.  
PQFP: plastic quad flat pack.
- (2) Cyclone devices support vertical migration within the same package (i.e., designers can migrate between the EP1C3 device in the 144-pin TQFP package and the EP1C6 device in the same package)

**Table 1-3. Cyclone QFP & FineLine BGA Package Sizes**

Dimension	100-Pin TQFP	144-Pin TQFP	240-Pin PQFP	256-Pin FineLine BGA	324-Pin FineLine BGA	400-Pin FineLine BGA
Pitch (mm)	0.5	0.5	0.5	1.0	1.0	1.0
Area (mm <sup>2</sup> )	256	484	1,024	289	361	441
Length × width (mm × mm)	16 × 16	22 × 22	34.6 × 34.6	17 × 17	19 × 19	21 × 21



### Functional Description

Cyclone devices contain a two-dimensional row- and column-based architecture to implement custom logic. Column and row interconnects of varying speeds provide signal interconnects between LABs and embedded memory blocks.

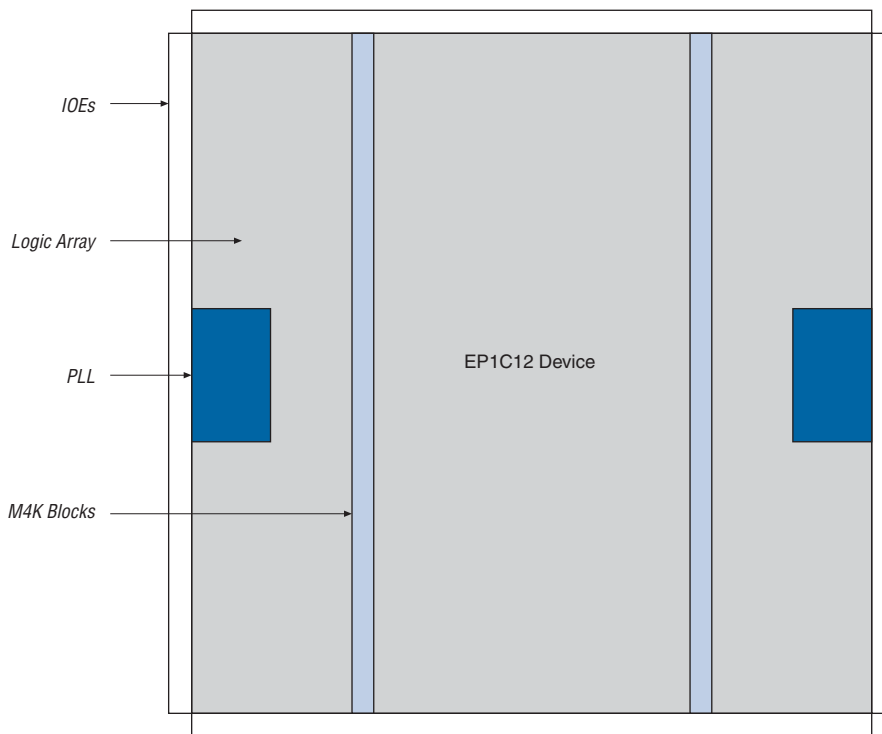
The logic array consists of LABs, with 10 LEs in each LAB. An LE is a small unit of logic providing efficient implementation of user logic functions. LABs are grouped into rows and columns across the device. Cyclone devices range between 2,910 to 20,060 LEs.

M4K RAM blocks are true dual-port memory blocks with 4K bits of memory plus parity (4,608 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 36-bits wide at up to 200 MHz. These blocks are grouped into columns across the device in between certain LABs. Cyclone devices offer between 60 to 288 Kbits of embedded RAM.

Each Cyclone device I/O pin is fed by an I/O element (IOE) located at the ends of LAB rows and columns around the periphery of the device. I/O pins support various single-ended and differential I/O standards, such as the 66- and 33-MHz, 64- and 32-bit PCI standard and the LVDS I/O standard at up to 640 Mbps. Each IOE contains a bidirectional I/O buffer and three registers for registering input, output, and output-enable signals. Dual-purpose DQS, DQ, and DM pins along with delay chains (used to phase-align DDR signals) provide interface support with external memory devices such as DDR SDRAM, and FCRAM devices at up to 133 MHz (266 Mbps).

Cyclone devices provide a global clock network and up to two PLLs. The global clock network consists of eight global clock lines that drive throughout the entire device. The global clock network can provide clocks for all resources within the device, such as IOEs, LEs, and memory blocks. The global clock lines can also be used for control signals. Cyclone PLLs provide general-purpose clocking with clock multiplication and phase shifting as well as external outputs for high-speed differential I/O support.

Figure 2–1 shows a diagram of the Cyclone EP1C12 device.

**Figure 2–1. Cyclone EP1C12 Device Block Diagram**

The number of M4K RAM blocks, PLLs, rows, and columns vary per device. [Table 2–1](#) lists the resources available in each Cyclone device.

**Table 2–1. Cyclone Device Resources**

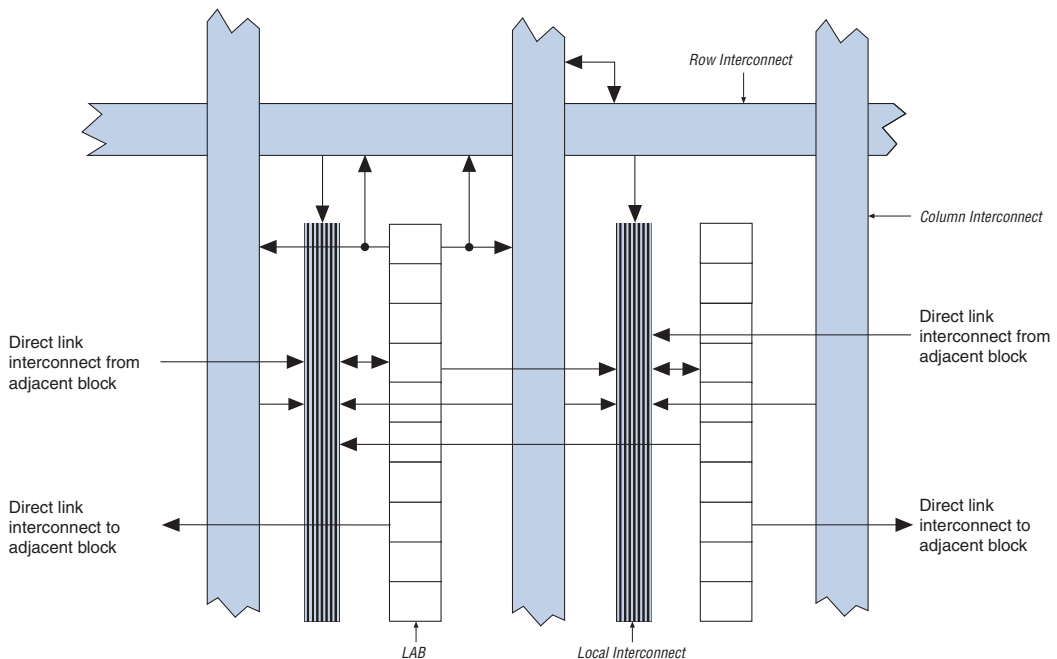
Device	M4K RAM		PLLs	LAB Columns	LAB Rows
	Columns	Blocks			
EP1C3	1	13	1	24	13
EP1C4	1	17	2	26	17
EP1C6	1	20	2	32	20
EP1C12	2	52	2	48	26
EP1C20	2	64	2	64	32



## Logic Array Blocks

Each LAB consists of 10 LEs, LE carry chains, LAB control signals, a local interconnect, look-up table (LUT) chain, and register chain connection lines. The local interconnect transfers signals between LEs in the same LAB. LUT chain connections transfer the output of one LE's LUT to the adjacent LE for fast sequential LUT connections within the same LAB. Register chain connections transfer the output of one LE's register to the adjacent LE's register within an LAB. The Quartus® II Compiler places associated logic within an LAB or adjacent LABs, allowing the use of local, LUT chain, and register chain connections for performance and area efficiency. Figure 2-2 details the Cyclone LAB structure.

Figure 2-2. Cyclone LAB Structure

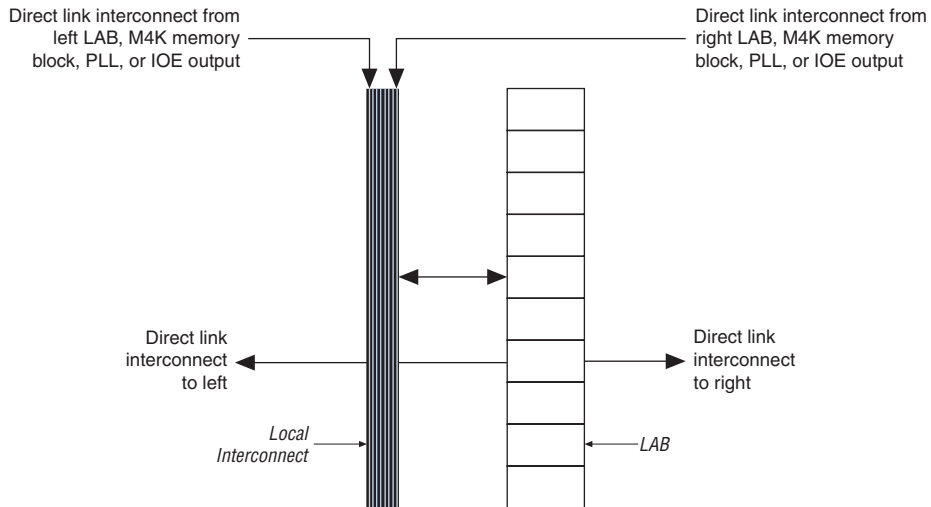


### LAB Interconnects

The LAB local interconnect can drive LEs within the same LAB. The LAB local interconnect is driven by column and row interconnects and LE outputs within the same LAB. Neighboring LABs, PLLs, and M4K RAM blocks from the left and right can also drive an LAB's local interconnect through the direct link connection. The direct link connection feature minimizes the use of row and column interconnects, providing higher

performance and flexibility. Each LE can drive 30 other LEs through fast local and direct link interconnects. Figure 2-3 shows the direct link connection.

**Figure 2-3. Direct Link Connection**



## LAB Control Signals

Each LAB contains dedicated logic for driving control signals to its LEs. The control signals include two clocks, two clock enables, two asynchronous clears, synchronous clear, asynchronous preset/load, synchronous load, and add/subtract control signals. This gives a maximum of 10 control signals at a time. Although synchronous load and clear signals are generally used when implementing counters, they can also be used with other functions.

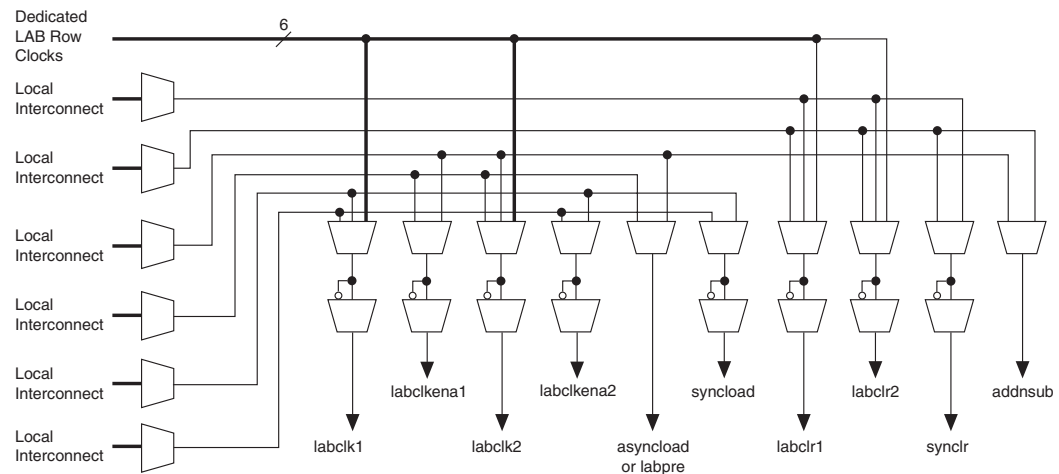
Each LAB can use two clocks and two clock enable signals. Each LAB's clock and clock enable signals are linked. For example, any LE in a particular LAB using the `labclk1` signal will also use `labckena1`. If the LAB uses both the rising and falling edges of a clock, it also uses both LAB-wide clock signals. De-asserting the clock enable signal will turn off the LAB-wide clock.

Each LAB can use two asynchronous clear signals and an asynchronous load/preset signal. The asynchronous load acts as a preset when the asynchronous load data input is tied high.

With the LAB-wide `addnsub` control signal, a single LE can implement a one-bit adder and subtractor. This saves LE resources and improves performance for logic functions such as DSP correlators and signed multipliers that alternate between addition and subtraction depending on data.

The LAB row clocks [5..0] and LAB local interconnect generate the LAB-wide control signals. The MultiTrack™ interconnect's inherent low skew allows clock and control signal distribution in addition to data. [Figure 2-4](#) shows the LAB control signal generation circuit.

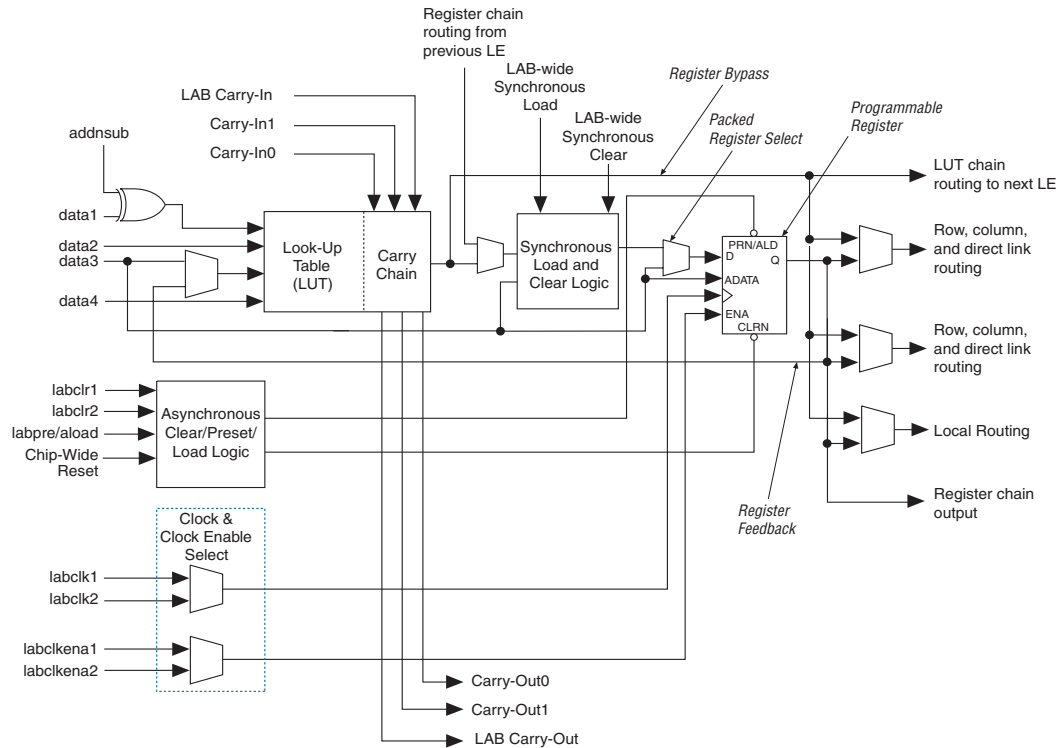
**Figure 2-4. LAB-Wide Control Signals**



## Logic Elements

The smallest unit of logic in the Cyclone architecture, the LE, is compact and provides advanced features with efficient logic utilization. Each LE contains a four-input LUT, which is a function generator that can implement any function of four variables. In addition, each LE contains a programmable register and carry chain with carry select capability. A single LE also supports dynamic single bit addition or subtraction mode selectable by an LAB-wide control signal. Each LE drives all types of interconnects: local, row, column, LUT chain, register chain, and direct link interconnects. See [Figure 2-5](#).

Figure 2-5. Cyclone LE



Each LE's programmable register can be configured for D, T, JK, or SR operation. Each register has data, true asynchronous load data, clock, clock enable, clear, and asynchronous load/preset inputs. Global signals, general-purpose I/O pins, or any internal logic can drive the register's clock and clear control signals. Either general-purpose I/O pins or internal logic can drive the clock enable, preset, asynchronous load, and asynchronous data. The asynchronous load data input comes from the data3 input of the LE. For combinatorial functions, the LUT output bypasses the register and drives directly to the LE outputs.

Each LE has three outputs that drive the local, row, and column routing resources. The LUT or register output can drive these three outputs independently. Two LE outputs drive column or row and direct link routing connections and one drives local interconnect resources. This allows the LUT to drive one output while the register drives another output. This feature, called register packing, improves device utilization because the device can use the register and the LUT for unrelated

functions. Another special packing mode allows the register output to feed back into the LUT of the same LE so that the register is packed with its own fan-out LUT. This provides another mechanism for improved fitting. The LE can also drive out registered and unregistered versions of the LUT output.

## LUT Chain & Register Chain

In addition to the three general routing outputs, the LEs within an LAB have LUT chain and register chain outputs. LUT chain connections allow LUTs within the same LAB to cascade together for wide input functions. Register chain outputs allow registers within the same LAB to cascade together. The register chain output allows an LAB to use LUTs for a single combinatorial function and the registers to be used for an unrelated shift register implementation. These resources speed up connections between LABs while saving local interconnect resources. “[MultiTrack Interconnect](#)” on page 2–12 for more information on LUT chain and register chain connections.

## addsub Signal

The LE's dynamic adder/subtractor feature saves logic resources by using one set of LEs to implement both an adder and a subtractor. This feature is controlled by the LAB-wide control signal `addsub`. The `addsub` signal sets the LAB to perform either  $A + B$  or  $A - B$ . The LUT computes addition; subtraction is computed by adding the two's complement of the intended subtractor. The LAB-wide signal converts to two's complement by inverting the B bits within the LAB and setting carry-in = 1 to add one to the least significant bit (LSB). The LSB of an adder/subtractor must be placed in the first LE of the LAB, where the LAB-wide `addsub` signal automatically sets the carry-in to 1. The Quartus II Compiler automatically places and uses the adder/subtractor feature when using adder/subtractor parameterized functions.

## LE Operating Modes

The Cyclone LE can operate in one of the following modes:

- Normal mode
- Dynamic arithmetic mode

Each mode uses LE resources differently. In each mode, eight available inputs to the LE—the four data inputs from the LAB local interconnect, `carry-in0` and `carry-in1` from the previous LE, the LAB carry-in from the previous carry-chain LAB, and the register chain connection—are directed to different destinations to implement the desired logic function. LAB-wide signals provide clock, asynchronous



### *Dynamic Arithmetic Mode*

The dynamic arithmetic mode is ideal for implementing adders, counters, accumulators, wide parity functions, and comparators. An LE in dynamic arithmetic mode uses four 2-input LUTs configurable as a dynamic adder/subtractor. The first two 2-input LUTs compute two summations based on a possible carry-in of 1 or 0; the other two LUTs generate carry outputs for the two chains of the carry select circuitry. As shown in [Figure 2-7](#), the LAB carry-in signal selects either the `carry-in0` or `carry-in1` chain. The selected chain's logic level in turn determines which parallel sum is generated as a combinatorial or registered output. For example, when implementing an adder, the sum output is the selection of two possible calculated sums:

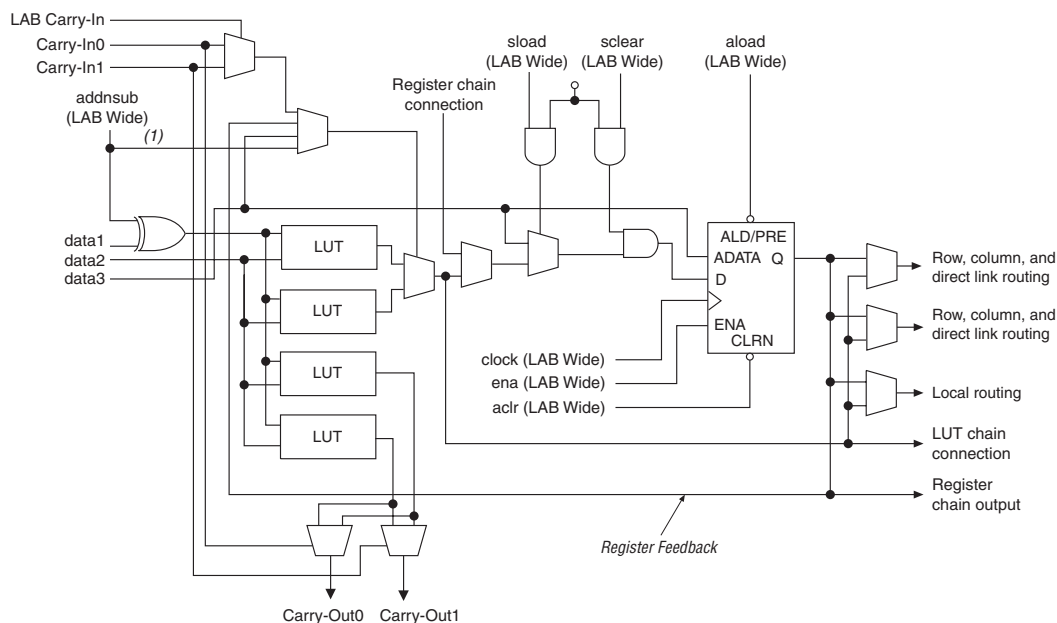
$$\text{data1} + \text{data2} + \text{carry-in0}$$

or

$$\text{data1} + \text{data2} + \text{carry-in1}$$

The other two LUTs use the `data1` and `data2` signals to generate two possible carry-out signals—one for a carry of 1 and the other for a carry of 0. The `carry-in0` signal acts as the carry select for the `carry-out0` output and `carry-in1` acts as the carry select for the `carry-out1` output. LEs in arithmetic mode can drive out registered and unregistered versions of the LUT output.

The dynamic arithmetic mode also offers clock enable, counter enable, synchronous up/down control, synchronous clear, synchronous load, and dynamic adder/subtractor options. The LAB local interconnect data inputs generate the counter enable and synchronous up/down control signals. The synchronous clear and synchronous load options are LAB-wide signals that affect all registers in the LAB. The Quartus II software automatically places any registers that are not used by the counter into other LABs. The `addnsub` LAB-wide signal controls whether the LE acts as an adder or subtractor.

**Figure 2-7. LE in Dynamic Arithmetic Mode****Note to Figure 2-7:**

(1) The addnsub signal is tied to the carry input for the first LE of a carry chain only.

### Carry-Select Chain

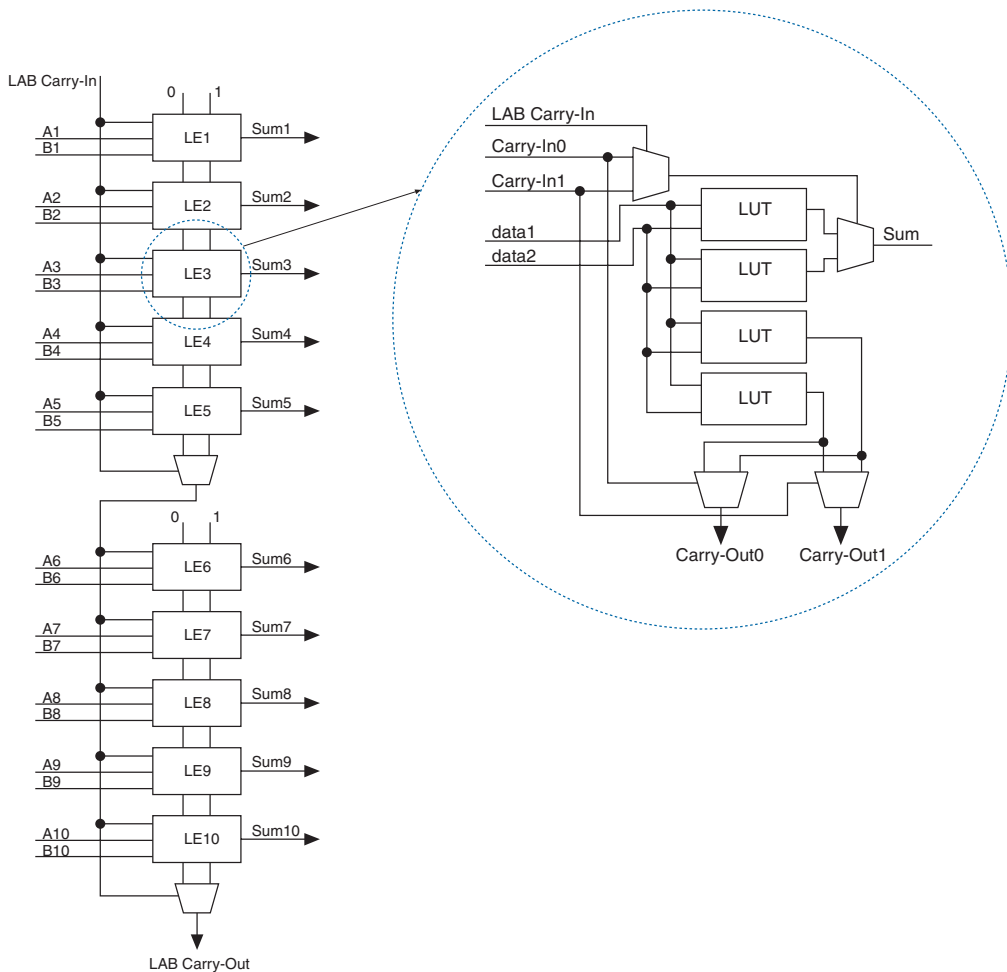
The carry-select chain provides a very fast carry-select function between LEs in dynamic arithmetic mode. The carry-select chain uses the redundant carry calculation to increase the speed of carry functions. The LE is configured to calculate outputs for a possible carry-in of 0 and carry-in of 1 in parallel. The carry-in0 and carry-in1 signals from a lower-order bit feed forward into the higher-order bit via the parallel carry chain and feed into both the LUT and the next portion of the carry chain. Carry-select chains can begin in any LE within an LAB.

The speed advantage of the carry-select chain is in the parallel pre-computation of carry chains. Since the LAB carry-in selects the precomputed carry chain, not every LE is in the critical path. Only the propagation delays between LAB carry-in generation (LE 5 and LE 10) are now part of the critical path. This feature allows the Cyclone architecture to implement high-speed counters, adders, multipliers, parity functions, and comparators of arbitrary width.



Figure 2–8 shows the carry-select circuitry in an LAB for a 10-bit full adder. One portion of the LUT generates the sum of two bits using the input signals and the appropriate carry-in bit; the sum is routed to the output of the LE. The register can be bypassed for simple adders or used for accumulator functions. Another portion of the LUT generates carry-out bits. An LAB-wide carry-in bit selects which chain is used for the addition of given inputs. The carry-in signal for each chain, `carry-in0` or `carry-in1`, selects the carry-out to carry forward to the carry-in signal of the next-higher-order bit. The final carry-out signal is routed to an LE, where it is fed to local, row, or column interconnects.

Figure 2–8. Carry Select Chain



The Quartus II Compiler automatically creates carry chain logic during design processing, or the designer can create it manually during design entry. Parameterized functions such as LPM functions automatically take advantage of carry chains for the appropriate functions.

The Quartus II Compiler creates carry chains longer than 10 LEs by linking LABs together automatically. For enhanced fitting, a long carry chain runs vertically allowing fast horizontal connections to M4K memory blocks. A carry chain can continue as far as a full column.

### *Clear & Preset Logic Control*

LAB-wide signals control the logic for the register's clear and preset signals. The LE directly supports an asynchronous clear and preset function. The register preset is achieved through the asynchronous load of a logic high. The direct asynchronous preset does not require a NOT-gate push-back technique. Cyclone devices support simultaneous preset/asynchronous load and clear signals. An asynchronous clear signal takes precedence if both signals are asserted simultaneously. Each LAB supports up to two clears and one preset signal.

In addition to the clear and preset ports, Cyclone devices provide a chip-wide reset pin (`DEV_CLRn`) that resets all registers in the device. An option set before compilation in the Quartus II software controls this pin. This chip-wide reset overrides all other control signals.

## **MultiTrack Interconnect**

In the Cyclone architecture, connections between LEs, M4K memory blocks, and device I/O pins are provided by the MultiTrack interconnect structure with DirectDrive™ technology. The MultiTrack interconnect consists of continuous, performance-optimized routing lines of different speeds used for inter- and intra-design block connectivity. The Quartus II Compiler automatically places critical design paths on faster interconnects to improve design performance.

DirectDrive technology is a deterministic routing technology that ensures identical routing resource usage for any function regardless of placement within the device. The MultiTrack interconnect and DirectDrive technology simplify the integration stage of block-based designing by eliminating the re-optimization cycles that typically follow design changes and additions.

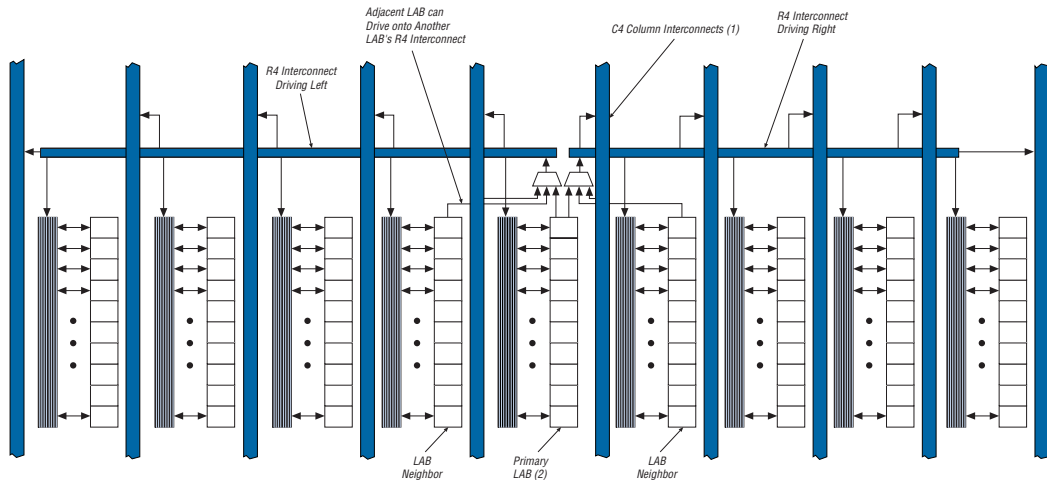
The MultiTrack interconnect consists of row and column interconnects that span fixed distances. A routing structure with fixed length resources for all devices allows predictable and repeatable performance when

migrating through different device densities. Dedicated row interconnects route signals to and from LABs, PLLs, and M4K memory blocks within the same row. These row resources include:

- Direct link interconnects between LABs and adjacent blocks
- R4 interconnects traversing four blocks to the right or left

The direct link interconnect allows an LAB or M4K memory block to drive into the local interconnect of its left and right neighbors. Only one side of a PLL block interfaces with direct link and row interconnects. The direct link interconnect provides fast communication between adjacent LABs and/or blocks without using row interconnect resources.

The R4 interconnects span four LABs, or two LABs and one M4K RAM block. These resources are used for fast row connections in a four-LAB region. Every LAB has its own set of R4 interconnects to drive either left or right. [Figure 2-9](#) shows R4 interconnect connections from an LAB. R4 interconnects can drive and be driven by M4K memory blocks, PLLs, and row IOEs. For LAB interfacing, a primary LAB or LAB neighbor can drive a given R4 interconnect. For R4 interconnects that drive to the right, the primary LAB and right neighbor can drive on to the interconnect. For R4 interconnects that drive to the left, the primary LAB and its left neighbor can drive on to the interconnect. R4 interconnects can drive other R4 interconnects to extend the range of LABs they can drive. R4 interconnects can also drive C4 interconnects for connections from one row to another.

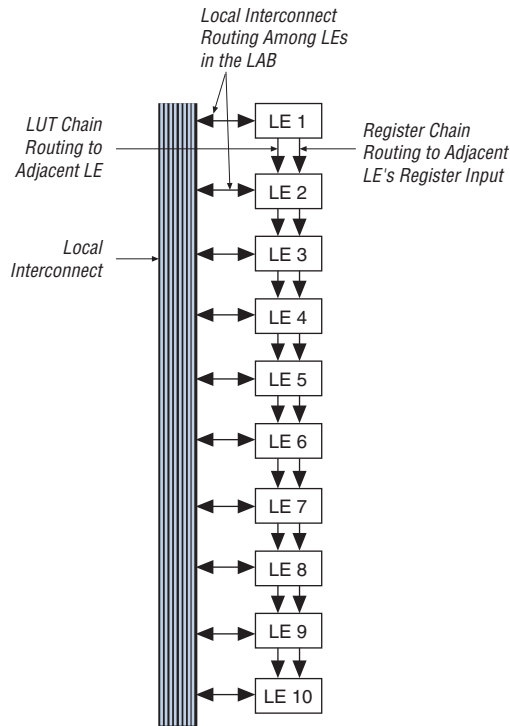
**Figure 2–9. R4 Interconnect Connections****Notes to Figure 2–9:**

- (1) C4 interconnects can drive R4 interconnects.
- (2) This pattern is repeated for every LAB in the LAB row.

The column interconnect operates similarly to the row interconnect. Each column of LABs is served by a dedicated column interconnect, which vertically routes signals to and from LABs, M4K memory blocks, and row and column IOEs. These column resources include:

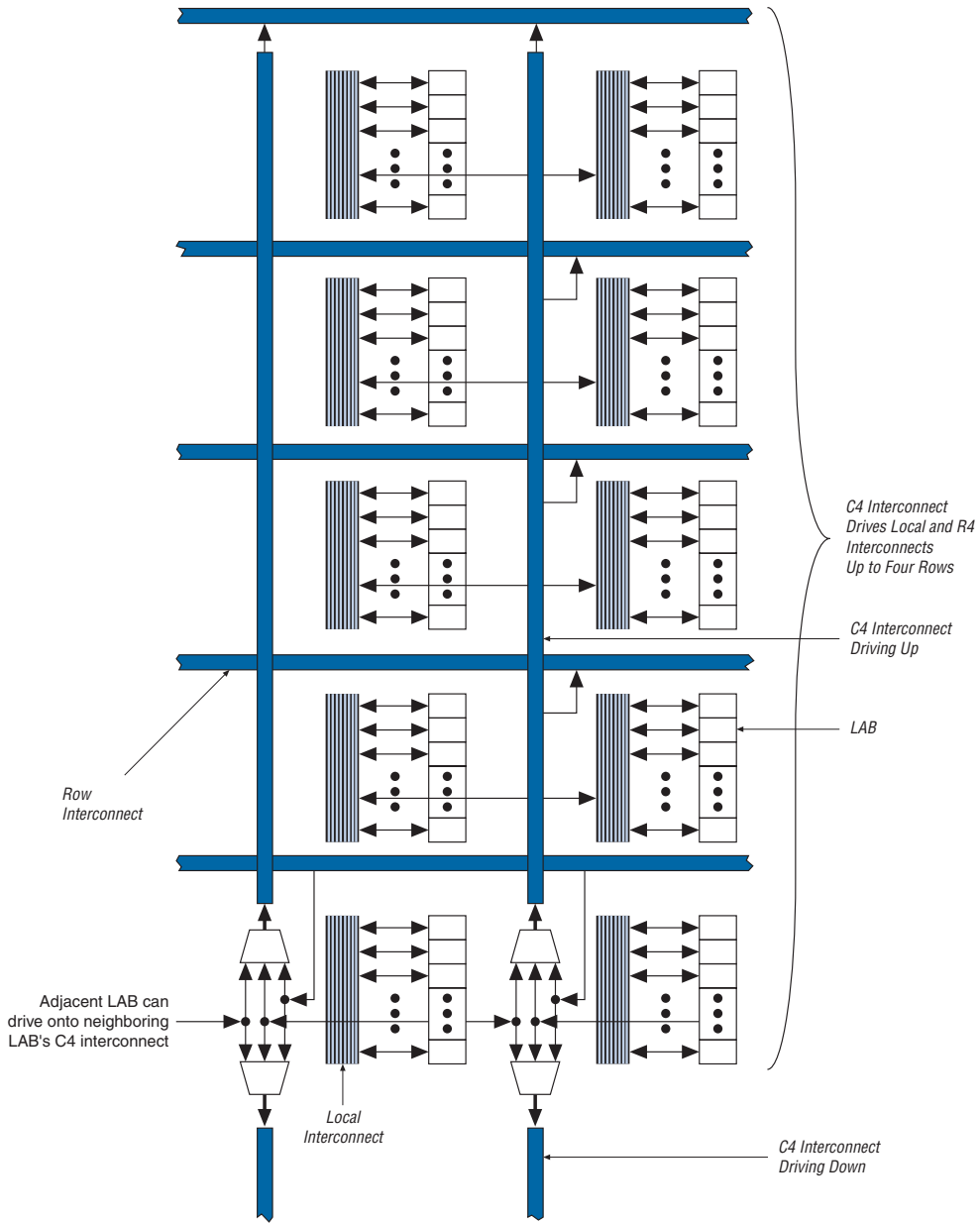
- LUT chain interconnects within an LAB
- Register chain interconnects within an LAB
- C4 interconnects traversing a distance of four blocks in an up and down direction

Cyclone devices include an enhanced interconnect structure within LABs for routing LE output to LE input connections faster using LUT chain connections and register chain connections. The LUT chain connection allows the combinatorial output of an LE to directly drive the fast input of the LE right below it, bypassing the local interconnect. These resources can be used as a high-speed connection for wide fan-in functions from LE 1 to LE 10 in the same LAB. The register chain connection allows the register output of one LE to connect directly to the register input of the next LE in the LAB for fast shift registers. The Quartus II Compiler automatically takes advantage of these resources to improve utilization and performance. Figure 2–10 shows the LUT chain and register chain interconnects.

**Figure 2–10. LUT Chain & Register Chain Interconnects**

The C4 interconnects span four LABs or M4K blocks up or down from a source LAB. Every LAB has its own set of C4 interconnects to drive either up or down. [Figure 2–11](#) shows the C4 interconnect connections from an LAB in a column. The C4 interconnects can drive and be driven by all types of architecture blocks, including PLLs, M4K memory blocks, and column and row IOEs. For LAB interconnection, a primary LAB or its LAB neighbor can drive a given C4 interconnect. C4 interconnects can drive each other to extend their range as well as drive row interconnects for column-to-column connections.

**Figure 2-11. C4 Interconnect Connections** *Note (1)*



**Note to Figure 2-11:**

- (1) Each C4 interconnect can drive either up or down four rows.

All embedded blocks communicate with the logic array similar to LAB-to-LAB interfaces. Each block (i.e., M4K memory or PLL) connects to row and column interconnects and has local interconnect regions driven by row and column interconnects. These blocks also have direct link interconnects for fast connections to and from a neighboring LAB.

Table 2–2 shows the Cyclone device's routing scheme.

**Table 2–2. Cyclone Device Routing Scheme**

Source	Destination										
	LUT Chain	Register Chain	Local Interconnect	Direct Link Interconnect	R4 Interconnect	C4 Interconnect	LE	M4K RAM Block	PLL	Column IOE	Row IOE
LUT Chain							✓				
Register Chain							✓				
Local Interconnect							✓	✓	✓	✓	✓
Direct Link Interconnect			✓								
R4 Interconnect			✓		✓	✓					
C4 Interconnect			✓		✓	✓					
LE	✓	✓	✓	✓	✓	✓					
M4K RAM Block			✓	✓	✓	✓					
PLL				✓	✓	✓					
Column IOE						✓					
Row IOE				✓	✓	✓					

## Embedded Memory

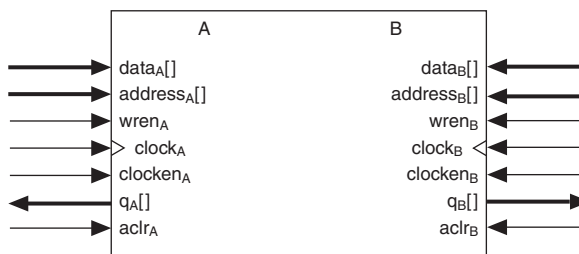
The Cyclone embedded memory consists of columns of M4K memory blocks. EP1C3 and EP1C6 devices have one column of M4K blocks, while EP1C12 and EP1C20 devices have two columns (see [Table 1–1 on page 1–2](#) for total RAM bits per density). Each M4K block can implement various types of memory with or without parity, including true dual-port, simple dual-port, and single-port RAM, ROM, and FIFO buffers. The M4K blocks support the following features:

- 4,608 RAM bits
- 200 MHz performance
- True dual-port memory
- Simple dual-port memory
- Single-port memory
- Byte enable
- Parity bits
- Shift register
- FIFO buffer
- ROM
- Mixed clock mode

### Memory Modes

The M4K memory blocks include input registers that synchronize writes and output registers to pipeline designs and improve system performance. M4K blocks offer a true dual-port mode to support any combination of two-port operations: two reads, two writes, or one read and one write at two different clock frequencies. [Figure 2–12](#) shows true dual-port memory.

**Figure 2–12. True Dual-Port Memory Configuration**



In addition to true dual-port memory, the M4K memory blocks support simple dual-port and single-port RAM. Simple dual-port memory supports a simultaneous read and write. Single-port memory supports non-simultaneous reads and writes. [Figure 2–13](#) shows these different M4K RAM memory port configurations.

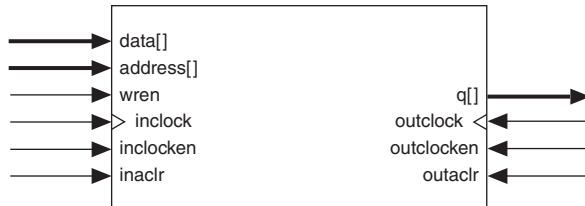


**Figure 2–13. Simple Dual-Port & Single-Port Memory Configurations**

**Simple Dual-Port Memory**



**Single-Port Memory (1)**



**Note to Figure 2–13:**

- (1) Two single-port memory blocks can be implemented in a single M4K block as long as each of the two independent block sizes is equal to or less than half of the M4K block size.

The memory blocks also enable mixed-width data ports for reading and writing to the RAM ports in dual-port RAM configuration. For example, the memory block can be written in  $\times 1$  mode at port A and read out in  $\times 16$  mode from port B.

The Cyclone memory architecture can implement fully synchronous RAM by registering both the input and output signals to the M4K RAM block. All M4K memory block inputs are registered, providing synchronous write cycles. In synchronous operation, the memory block generates its own self-timed strobe write enable (*wren*) signal derived from a global clock. In contrast, a circuit using asynchronous RAM must generate the RAM *wren* signal while ensuring its data and address signals meet setup and hold time specifications relative to the *wren* signal. The output registers can be bypassed. Pseudo-asynchronous reading is possible in the simple dual-port mode of M4K blocks by clocking the read enable and read address registers on the negative clock edge and bypassing the output registers.

When configured as RAM or ROM, the designer can use an initialization file to pre-load the memory contents.

Two single-port memory blocks can be implemented in a single M4K block as long as each of the two independent block sizes is equal to or less than half of the M4K block size.

The Quartus II software automatically implements larger memory by combining multiple M4K memory blocks. For example, two 256×16-bit RAM blocks can be combined to form a 256×32-bit RAM block. Memory performance does not degrade for memory blocks using the maximum number of words allowed. Logical memory blocks using less than the maximum number of words use physical blocks in parallel, eliminating any external control logic that would increase delays. To create a larger high-speed memory block, the Quartus II software automatically combines memory blocks with LE control logic.

### Parity Bit Support

The M4K blocks support a parity bit for each byte. The parity bit, along with internal LE logic, can implement parity checking for error detection to ensure data integrity. Designers can also use parity-size data words to store user-specified control bits. Byte enables are also available for data input masking during write operations.

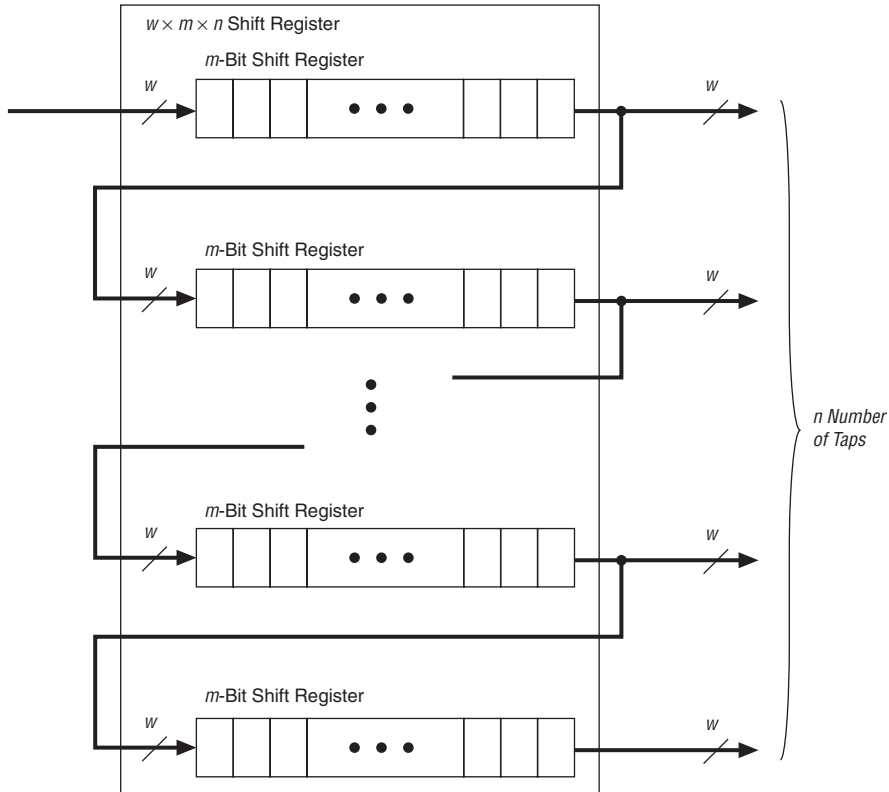
### Shift Register Support

The designer can configure M4K memory blocks to implement shift registers for DSP applications such as pseudo-random number generators, multi-channel filtering, auto-correlation, and cross-correlation functions. These and other DSP applications require local data storage, traditionally implemented with standard flip-flops, which can quickly consume many logic cells and routing resources for large shift registers. A more efficient alternative is to use embedded memory as a shift register block, which saves logic cell and routing resources and provides a more efficient implementation with the dedicated circuitry.

The size of a  $w \times m \times n$  shift register is determined by the input data width ( $w$ ), the length of the taps ( $m$ ), and the number of taps ( $n$ ). The size of a  $w \times m \times n$  shift register must be less than or equal to the maximum number of memory bits in the M4K block (4,608 bits). The total number of shift register outputs (number of taps  $n \times$  width  $w$ ) must be less than the maximum data width of the M4K RAM block ( $\times 36$ ). To create larger shift registers, multiple memory blocks are cascaded together.

Data is written into each address location at the falling edge of the clock and read from the address at the rising edge of the clock. The shift register mode logic automatically controls the positive and negative edge clocking to shift the data in one clock cycle. Figure 2-14 shows the M4K memory block in the shift register mode.

**Figure 2-14. Shift Register Memory Configuration**



### Memory Configuration Sizes

The memory address depths and output widths can be configured as  $4,096 \times 1$ ,  $2,048 \times 2$ ,  $1,024 \times 4$ ,  $512 \times 8$  (or  $512 \times 9$  bits),  $256 \times 16$  (or  $256 \times 18$  bits), and  $128 \times 32$  (or  $128 \times 36$  bits). The  $128 \times 32$ - or  $36$ -bit configuration

is not available in the true dual-port mode. Mixed-width configurations are also possible, allowing different read and write widths. Tables 2-3 and 2-4 summarize the possible M4K RAM block configurations.

**Table 2-3. M4K RAM Block Configurations (Simple Dual-Port)**

Read Port	Write Port								
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	128 × 32	512 × 9	256 × 18	128 × 36
4K × 1	✓	✓	✓	✓	✓	✓			
2K × 2	✓	✓	✓	✓	✓	✓			
1K × 4	✓	✓	✓	✓	✓	✓			
512 × 8	✓	✓	✓	✓	✓	✓			
256 × 16	✓	✓	✓	✓	✓	✓			
128 × 32	✓	✓	✓	✓	✓	✓			
512 × 9							✓	✓	✓
256 × 18							✓	✓	✓
128 × 36							✓	✓	✓

**Table 2-4. M4K RAM Block Configurations (True Dual-Port)**

Port A	Port B						
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	512 × 9	256 × 18
4K × 1	✓	✓	✓	✓	✓		
2K × 2	✓	✓	✓	✓	✓		
1K × 4	✓	✓	✓	✓	✓		
512 × 8	✓	✓	✓	✓	✓		
256 × 16	✓	✓	✓	✓	✓		
512 × 9						✓	✓
256 × 18						✓	✓

When the M4K RAM block is configured as a shift register block, the designer can create a shift register up to 4,608 bits ( $w \times m \times n$ ).

## Byte Enables

M4K blocks support byte writes when the write port has a data width of 16, 18, 32, or 36 bits. The byte enables allow the input data to be masked so the device can write to specific bytes. The unwritten bytes retain the previous written value. [Table 2-5](#) summarizes the byte selection.

<i>Table 2-5. Byte Enable for M4K Blocks</i> <i>Notes (1), (2)</i>		
<b>byteena[3..0]</b>	<b>datain × 18</b>	<b>datain × 36</b>
[0] = 1	[8..0]	[8..0]
[1] = 1	[17..9]	[17..9]
[2] = 1	–	[26..18]
[3] = 1	–	[35..27]

**Notes to [Table 2-5](#):**

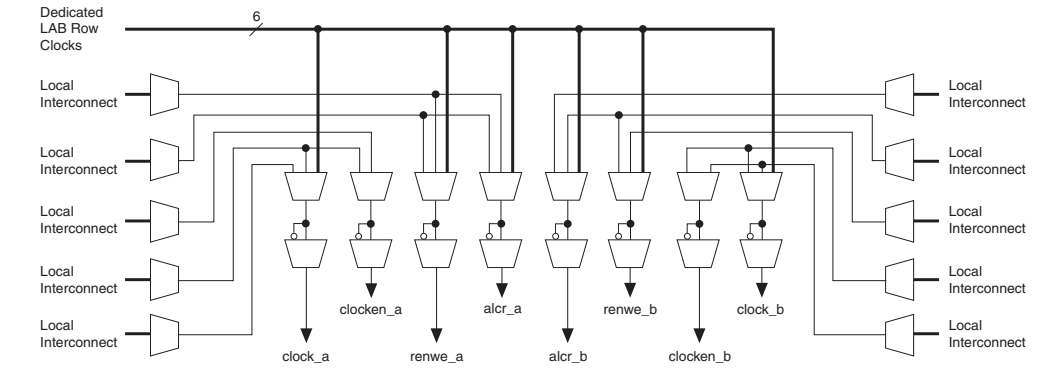
- (1) Any combination of byte enables is possible.
- (2) Byte enables can be used in the same manner with 8-bit words, i.e., in ×16 and ×32 modes.

## Control Signals & M4K Interface

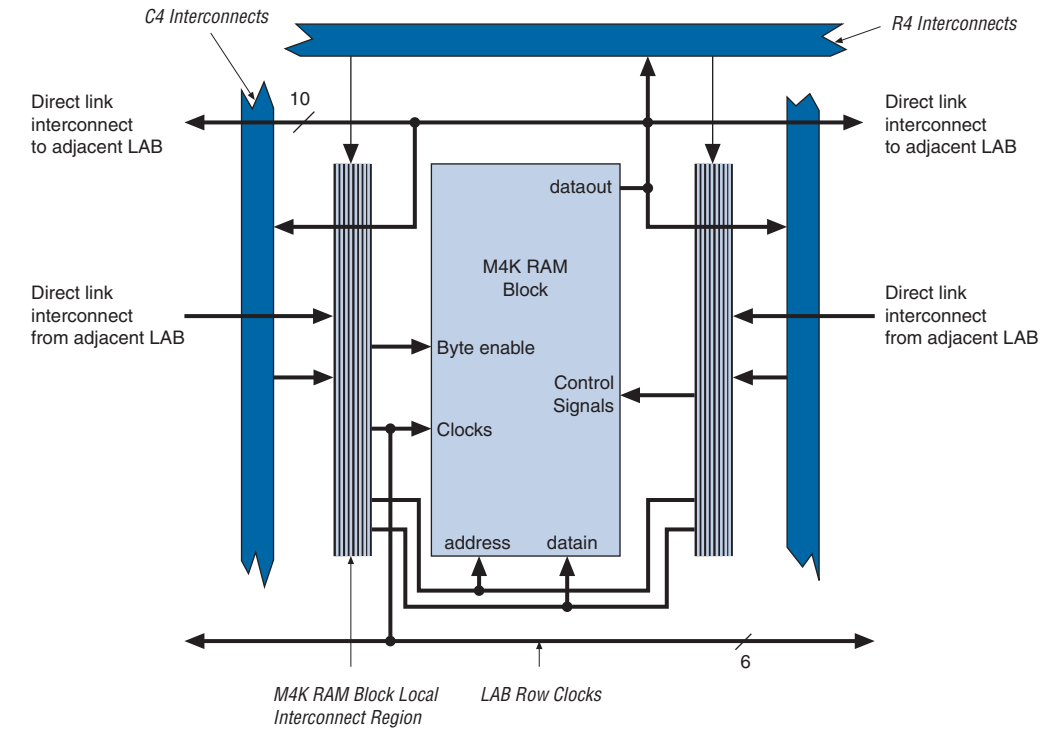
The M4K blocks allow for different clocks on their inputs and outputs. Either of the two clocks feeding the block can clock M4K block registers (*renwe*, *address*, *byte enable*, *datain*, and output registers). Only the output register can be bypassed. The six *labclk* signals or local interconnects can drive the control signals for the A and B ports of the M4K block. LEs can also control the *clock\_a*, *clock\_b*, *renwe\_a*, *renwe\_b*, *clr\_a*, *clr\_b*, *clocken\_a*, and *clocken\_b* signals, as shown in [Figure 2-15](#).

The R4, C4, and direct link interconnects from adjacent LABs drive the M4K block local interconnect. The M4K blocks can communicate with LABs on either the left or right side through these row resources or with LAB columns on either the right or left with the column resources. Up to 10 direct link input connections to the M4K block are possible from the left adjacent LABs and another 10 possible from the right adjacent LAB. M4K block outputs can also connect to left and right LABs through 10 direct link interconnects each. [Figure 2-16](#) shows the M4K block to logic array interface.

**Figure 2-15. M4K RAM Block Control Signals**



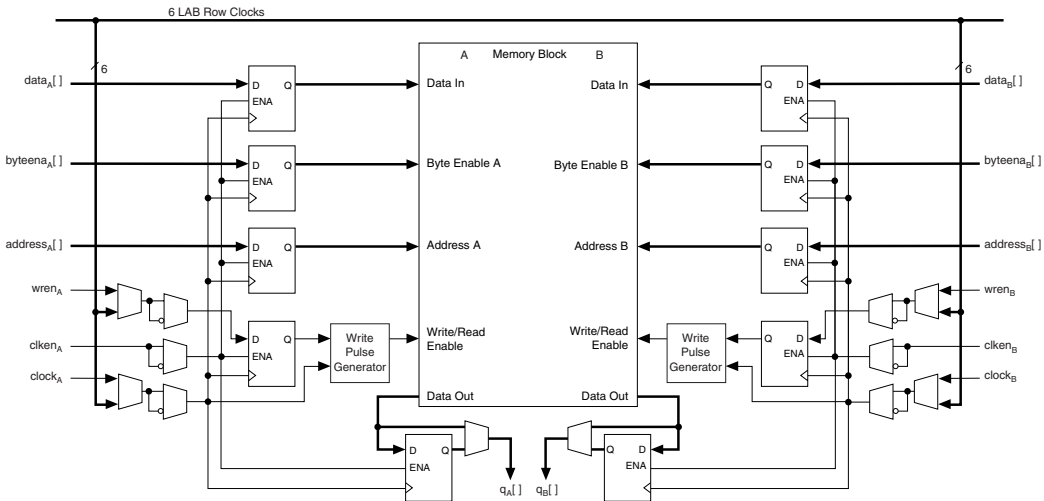
**Figure 2-16. M4K RAM Block LAB Row Interface**



## Independent Clock Mode

The M4K memory blocks implement independent clock mode for true dual-port memory. In this mode, a separate clock is available for each port (ports A and B). Clock A controls all registers on the port A side, while clock B controls all registers on the port B side. Each port, A and B, also supports independent clock enables and asynchronous clear signals for port A and B registers. Figure 2–17 shows an M4K memory block in independent clock mode.

**Figure 2–17. Independent Clock Mode** Note (1)



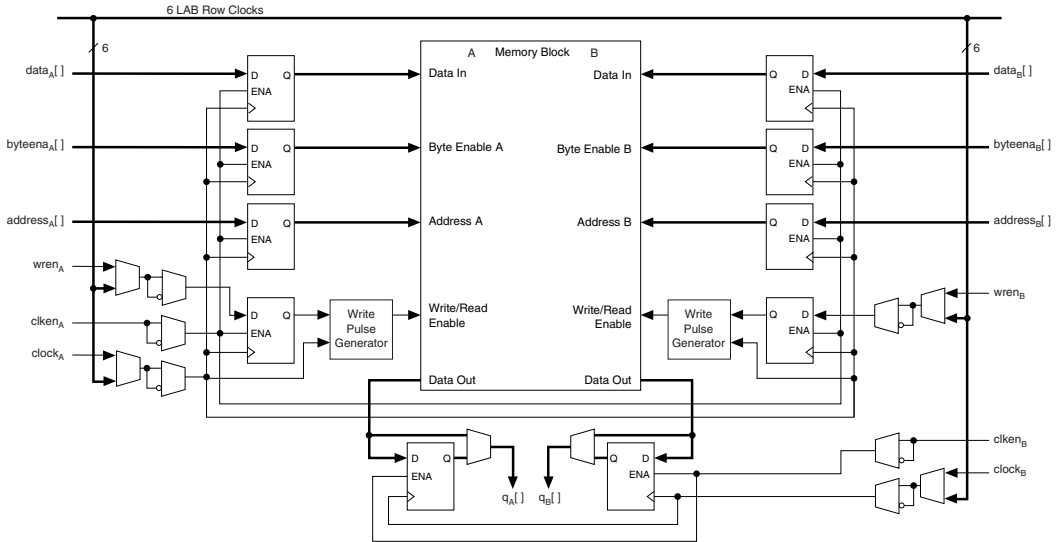
**Note to Figure 2–17:**

(1) All registers shown have asynchronous clear ports.

## Input/Output Clock Mode

Input/output clock mode can be implemented for both the true and simple dual-port memory modes. On each of the two ports, A or B, one clock controls all registers for inputs into the memory block: data input, wren, and address. The other clock controls the block’s data output registers. Each memory block port, A or B, also supports independent clock enables and asynchronous clear signals for input and output registers. Figures 2–18 and 2–19 show the memory block in input/output clock mode.

**Figure 2-18. Input/Output Clock Mode in True Dual-Port Mode** *Note (1)*

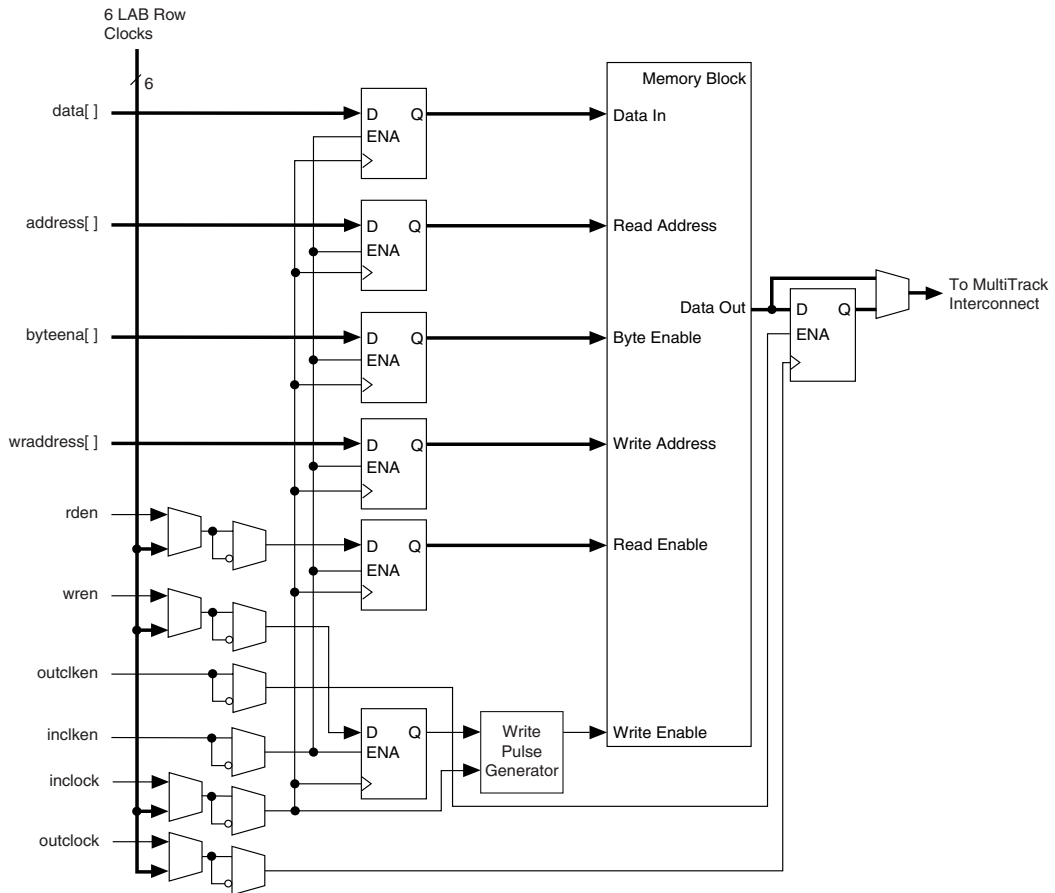


**Note to Figure 2-18:**

(1) All registers shown have asynchronous clear ports.



**Figure 2–19. Input/Output Clock Mode in Simple Dual-Port Mode** *Note (1)*

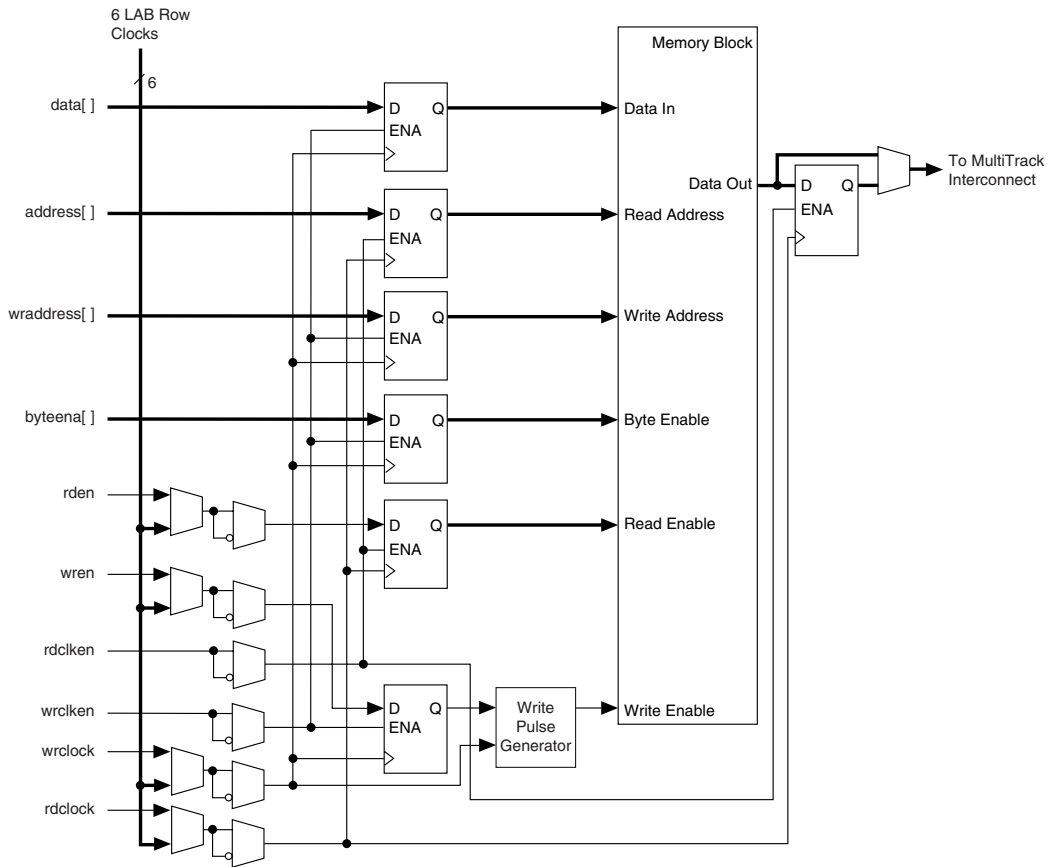


**Note to Figure 2–19:**

(1) All registers shown except the rden register have asynchronous clear ports.

### Read/Write Clock Mode

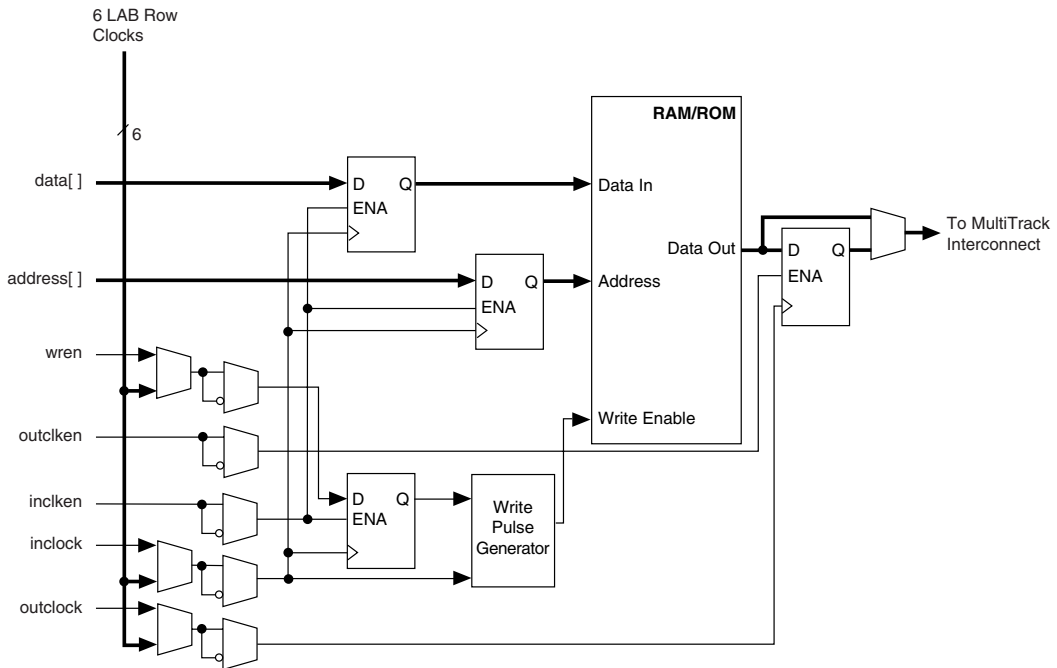
The M4K memory blocks implement read/write clock mode for simple dual-port memory. The designer can use up to two clocks in this mode. The write clock controls the block's data inputs, *waddress*, and *wren*. The read clock controls the data output, *rdaddress*, and *rden*. The memory blocks support independent clock enables for each clock and asynchronous clear signals for the read- and write-side registers. [Figure 2–20](#) shows a memory block in read/write clock mode.

**Figure 2–20. Read/Write Clock Mode in Simple Dual-Port Mode** *Note (1)***Note to Figure 2–20:**

(1) All registers shown except the rden register have asynchronous clear ports.

## Single-Port Mode

The M4K memory blocks also support single-port mode, used when simultaneous reads and writes are not required. See [Figure 2–21](#). A single M4K memory block can support up to two single-port mode RAM blocks if each RAM block is less than or equal to 2K bits in size.

**Figure 2–21. Single-Port Mode**

## Global Clock Network & Phase-Locked Loops

Cyclone devices provide a global clock network and up to two PLLs for a complete clock management solution.

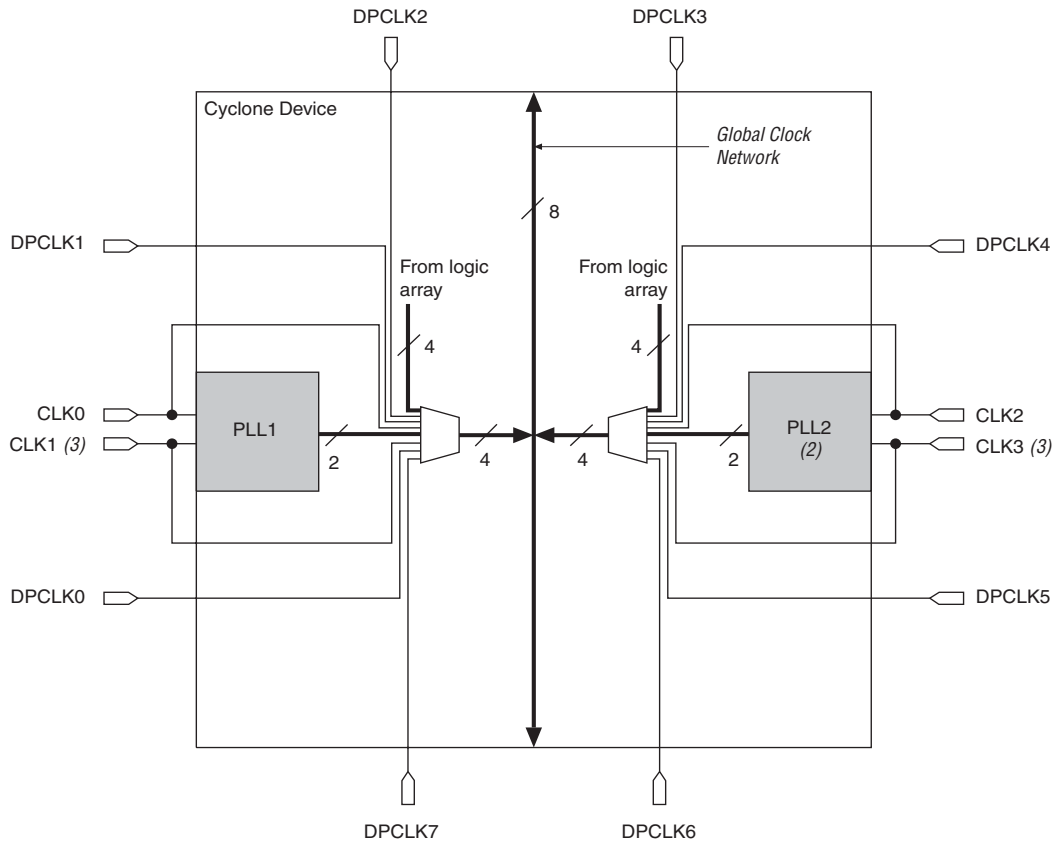
### Global Clock Network

There are four dedicated clock pins (CLK[3..0]), two pins on the left side and two pins on the right side) that drive the global clock network, as shown in [Figure 2–22](#). PLL outputs, logic array, and dual-purpose clock (DPCLK[7..0]) pins can also drive the global clock network.

The eight global clock lines in the global clock network drive throughout the entire device. The global clock network can provide clocks for all resources within the device — IOEs, LEs, and memory blocks. The global clock lines can also be used for control signals, such as clock enables and synchronous or asynchronous clears fed from the external pin, or DQS signals for DDR SDRAM or FCRAM interfaces. Internal logic can also drive the global clock network for internally generated global clocks and

asynchronous clears, clock enables, or other control signals with large fanout. Figure 2-22 shows the various sources that drive the global clock network.

**Figure 2-22. Global Clock Generation** *Note (1)*



**Notes to Figure 2-22:**

- (1) The EP1C3 device in the 100-pin TQFP package has five DPCLK pins (DPCLK2, DPCLK3, DPCLK4, DPCLK6, and DPCLK7).
- (2) EP1C3 devices only contain one PLL (PLL 1).
- (3) The EP1C3 device in the 100-pin TQFP package does not have dedicated clock pins CLK1 and CLK3.

## Dual-Purpose Clock Pins

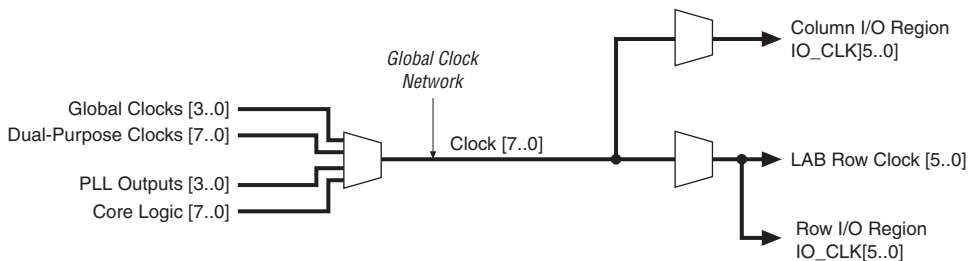
Each Cyclone device except the EP1C3 device has eight dual-purpose clock pins, DPCLK[7..0] (two on each I/O bank). EP1C3 devices have five DPCLK pins in the 100-pin TQFP package. These dual-purpose pins

can connect to the global clock network (see [Figure 2-22](#)) for high-fanout control signals such as clocks, asynchronous clears, presets, and clock enables, or protocol control signals such as TRDY and IRDY for PCI, or DQS signals for external memory interfaces.

## Combined Resources

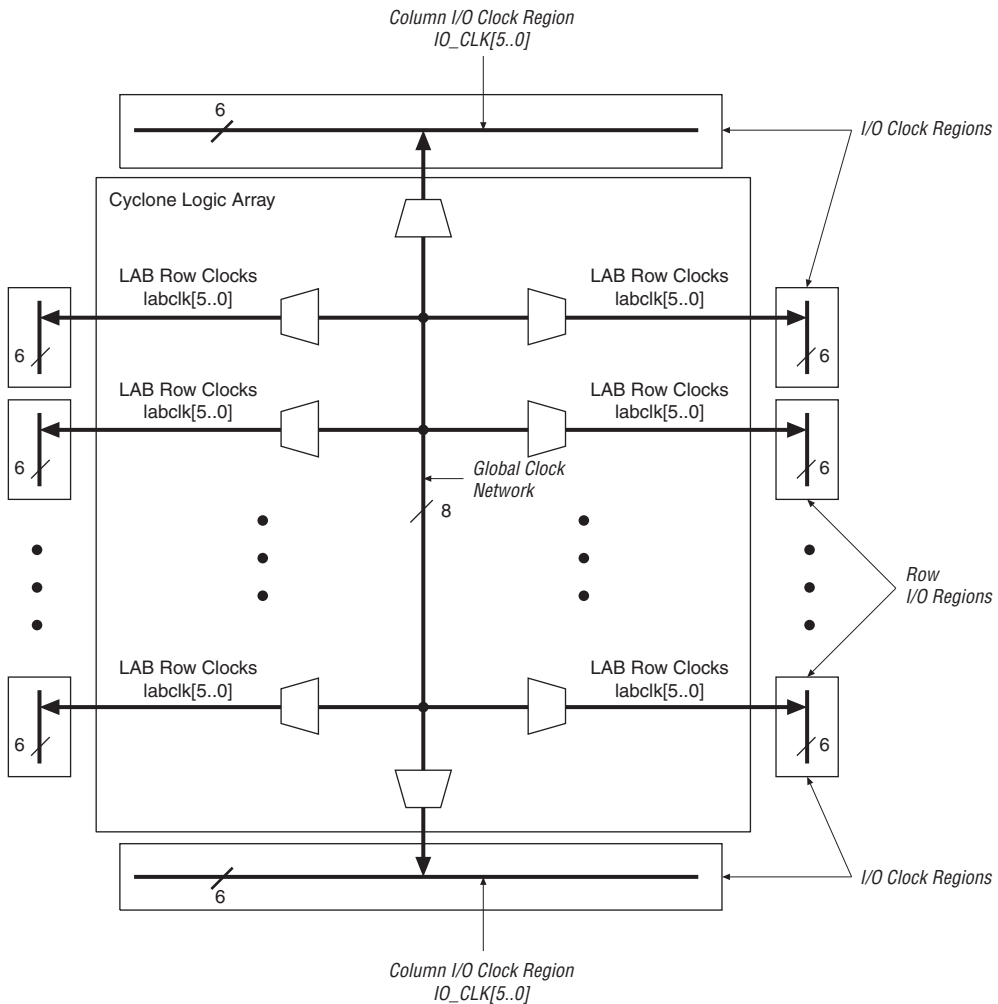
Each Cyclone device contains eight distinct dedicated clocking resources. The device uses multiplexers with these clocks to form six-bit buses to drive LAB row clocks, column IOE clocks, or row IOE clocks. See [Figure 2-23](#). Another multiplexer at the LAB level selects two of the six LAB row clocks to feed the LE registers within the LAB.

**Figure 2-23. Global Clock Network Multiplexers**



IOE clocks have row and column block regions. Six of the eight global clock resources feed to these row and column regions. [Figure 2-24](#) shows the I/O clock regions.

Figure 2–24. I/O Clock Regions



## PLLs

Cyclone PLLs provide general-purpose clocking with clock multiplication and phase shifting as well as outputs for differential I/O support. Cyclone devices contain two PLLs, except for the EP1C3 device, which contains one PLL.

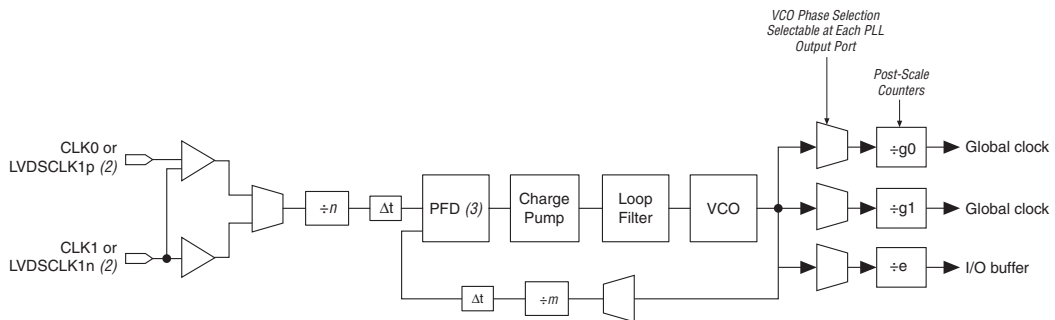
Table 2–6 shows the PLL features in Cyclone devices. Figure 2–25 shows a Cyclone PLL.

Feature	PLL Support
Clock multiplication and division	$m/(n \times \text{post-scale counter})$ (1)
Phase shift	Down to 125-ps increments (2), (3)
Programmable duty cycle	Yes
Number of internal clock outputs	2
Number of external clock outputs	One differential or one single-ended (4)

**Notes to Table 2–6:**

- (1) The  $m$  counter ranges from 2 to 32. The  $n$  counter and the post-scale counters range from 1 to 32.
- (2) The smallest phase shift is determined by the voltage-controlled oscillator (VCO) period divided by 8.
- (3) For degree increments, Cyclone devices can shift all output frequencies in increments of  $45^\circ$ . Smaller degree increments are possible depending on the frequency and divide parameters.
- (4) The EP1C3 device in the 100-pin TQFP package does not support external clock output. The EP1C6 device in the 144-pin TQFP package does not support external clock output from PLL2.

**Figure 2–25. Cyclone PLL** Note (1)

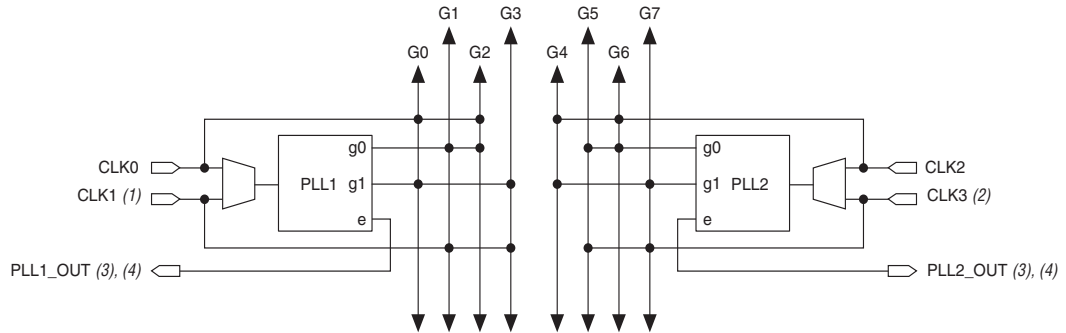


**Notes to Figure 2–25:**

- (1) The EP1C3 device in the 100-pin TQFP package does not support external outputs or LVDS inputs. The EP1C6 device in the 144-pin TQFP package does not support external output from PLL2.
- (2) LVDS input is supported via the secondary function of the dedicated clock pins. For PLL 1, the CLK0 pin's secondary function is LVDSCLK1P and the CLK1 pin's secondary function is LVDSCLK1N. For PLL 2, the CLK2 pin's secondary function is LVDSCLK2P and the CLK3 pin's secondary function is LVDSCLK2N.
- (3) PFD: phase frequency detector.

Figure 2–26 shows the PLL global clock connections.

**Figure 2–26. Cyclone PLL Global Clock Connections**



**Notes to Figure 2–26:**

- (1) PLL 1 supports one single-ended or LVDS input via pins CLK0 and CLK1.
- (2) PLL2 supports one single-ended or LVDS input via pins CLK2 and CLK3.
- (3) PLL1\_OUT and PLL2\_OUT support single-ended or LVDS output. If external output is not required, these pins are available as regular user I/O pins.
- (4) The EP1C3 device in the 100-pin TQFP package does not support external clock output. The EP1C6 device in the 144-pin TQFP package does not support external clock output from PLL2.

Table 2–7 shows the global clock network sources available in Cyclone devices.

Source		GCLK0	GCLK1	GCLK2	GCLK3	GCLK4	GCLK5	GCLK6	GCLK7
PLL Counter Output	PLL1 G0		✓	✓					
	PLL1 G1	✓			✓				
	PLL2 G0 (1)						✓	✓	
	PLL2 G1 (1)					✓			✓
Dedicated Clock Input Pins	CLK0	✓		✓					
	CLK1 (2)		✓		✓				
	CLK2					✓		✓	
	CLK3 (2)						✓		✓



**Table 2–7. Global Clock Network Sources (Part 2 of 2)**

Source		GCLK0	GCLK1	GCLK2	GCLK3	GCLK4	GCLK5	GCLK6	GCLK7
Dual-Purpose Clock Pins	DPCLK0 (3)				✓				
	DPCLK1 (3)			✓					
	DPCLK2	✓							
	DPCLK3					✓			
	DPCLK4							✓	
	DPCLK5 (3)								✓
	DPCLK6						✓		
	DPCLK7		✓						

**Notes to Table 2–7:**

- (1) EP1C3 devices only have one PLL (PLL 1).
- (2) EP1C3 devices in the 100-pin TQFP package do not have dedicated clock pins CLK1 and CLK3.
- (3) EP1C3 devices in the 100-pin TQFP package do not have the DPCLK0, DPCLK1, or DPCLK5 pins.

## Clock Multiplication & Division

Cyclone PLLs provide clock synthesis for PLL output ports using  $m/(n \times \text{post scale counter})$  scaling factors. The input clock is divided by a pre-scale divider,  $n$ , and is then multiplied by the  $m$  feedback factor. The control loop drives the VCO to match  $f_{IN} \times (m/n)$ . Each output port has a unique post-scale counter to divide down the high-frequency VCO. For multiple PLL outputs with different frequencies, the VCO is set to the least-common multiple of the output frequencies that meets its frequency specifications. Then, the post-scale dividers scale down the output frequency for each output port. For example, if the output frequencies required from one PLL are 33 and 66 MHz, the VCO is set to 330 MHz (the least-common multiple in the VCO's range).

Each PLL has one pre-scale divider,  $n$ , that can range in value from 1 to 32. Each PLL also has one multiply divider,  $m$ , that can range in value from 2 to 32. Global clock outputs have two post scale G dividers for global clock outputs, and external clock outputs have an E divider for external clock output, both ranging from 1 to 32. The Quartus II software automatically chooses the appropriate scaling factors according to the input frequency, multiplication, and division values entered.

## External Clock Inputs

Each PLL supports single-ended or differential inputs for source-synchronous receivers or for general-purpose use. The dedicated clock pins (CLK[3..0]) feed the PLL inputs. These dual-purpose pins can also act as LVDS input pins. See [Figure 2-25](#).

[Table 2-8](#) shows the I/O standards supported by PLL input and output pins.

I/O Standard	CLK Input	EXTCLK Output
3.3-V LVTTTL/LVCMOS	✓	✓
2.5-V LVTTTL/LVCMOS	✓	✓
1.8-V LVTTTL/LVCMOS	✓	✓
1.5-V LVCMOS	✓	✓
3.3-V PCI	✓	✓
LVDS	✓	✓
SSTL-2 class I	✓	✓
SSTL-2 class II	✓	✓
SSTL-3 class I	✓	✓
SSTL-3 class II	✓	✓
Differential SSTL-2		✓

For more information on LVDS I/O support, see “LVDS I/O Pins” on [page 2-54](#).

## External Clock Outputs

Each PLL supports one differential or one single-ended output for source-synchronous transmitters or for general-purpose external clocks. If the PLL does not use these PLL\_OUT pins, the pins are available for use as general-purpose I/O pins. The PLL\_OUT pins support all I/O standards shown in [Table 2-8](#).

The external clock outputs do not have their own V<sub>CC</sub> and ground voltage supplies. Therefore, to minimize jitter, do not place switching I/O pins next to these output pins. The EP1C3 device in the 100-pin TQFP package

does not have dedicated clock output pins. The EP1C6 device in the 144-pin TQFP package only supports dedicated clock outputs from PLL 1.

## Clock Feedback

Cyclone PLLs have three modes for multiplication and/or phase shifting:

- Zero delay buffer mode—The external clock output pin is phase-aligned with the clock input pin for zero delay.
- Normal mode—If the design uses an internal PLL clock output, the normal mode compensates for the internal clock delay from the input clock pin to the IOE registers. The external clock output pin is phase shifted with respect to the clock input pin if connected in this mode. The designer defines which internal clock output from the PLL should be phase-aligned to compensate for internal clock delay.
- No compensation mode—In this mode, the PLL will not compensate for any clock networks.

## Phase Shifting

Cyclone PLLs have an advanced clock shift capability that enables programmable phase shifts. Designers can enter a phase shift (in degrees or time units) for each PLL clock output port or for all outputs together in one shift. Designers can perform phase shifting in time units with a resolution range of 125 to 250 ps. The finest resolution equals one eighth of the VCO period. The VCO period is a function of the frequency input and the multiplication and division factors. Each clock output counter can choose a different phase of the VCO period from up to eight taps. Designers can use this clock output counter along with an initial setting on the post-scale counter to achieve a phase-shift range for the entire period of the output clock. The phase tap feedback to the m counter can shift all outputs to a single phase. The Quartus II software automatically sets the phase taps and counter settings according to the phase shift entered.

## Lock Detect Signal

The lock output indicates that there is a stable clock output signal in phase with the reference clock. Without any additional circuitry, the lock signal may toggle as the PLL begins tracking the reference clock. Therefore, the designer may need to gate the lock signal for use as a system-control signal. For correct operation of the lock circuit below  $-20\text{ C}$ ,  $f_{\text{IN}/N} > 200\text{ MHz}$ .

## Programmable Duty Cycle

The programmable duty cycle allows PLLs to generate clock outputs with a variable duty cycle. This feature is supported on each PLL post-scale counter (g0, g1, e). The duty cycle setting is achieved by a low- and high-time count setting for the post-scale dividers. The Quartus II software uses the frequency input and the required multiply or divide rate to determine the duty cycle choices.

## Control Signals

There are three control signals for clearing and enabling PLLs and their outputs. The designer can use these signals to control PLL resynchronization and the ability to gate PLL output clocks for low-power applications.

The `pllenable` signal enables and disables PLLs. When the `pllenable` signal is low, the clock output ports are driven by ground and all the PLLs go out of lock. When the `pllenable` signal goes high again, the PLLs relock and resynchronize to the input clocks. An input pin or LE output can drive the `pllenable` signal.

The `areset` signals are reset/resynchronization inputs for each PLL. Cyclone devices can drive these input signals from input pins or from LEs. When `areset` is driven high, the PLL counters will reset, clearing the PLL output and placing the PLL out of lock. When driven low again, the PLL will resynchronize to its input as it relocks.

The `pfdena` signals control the phase frequency detector (PFD) output with a programmable gate. If you disable the PFD, the VCO will operate at its last set value of control voltage and frequency with some drift, and the system will continue running when the PLL goes out of lock or the input clock disables. By maintaining the last locked frequency, the system has time to store its current settings before shutting down. The designer can either use their own control signal or gated locked status signals to trigger the `pfdena` signal.



For more information on Cyclone PLLs, see [Chapter 6, Using PLLs in Cyclone Devices](#).

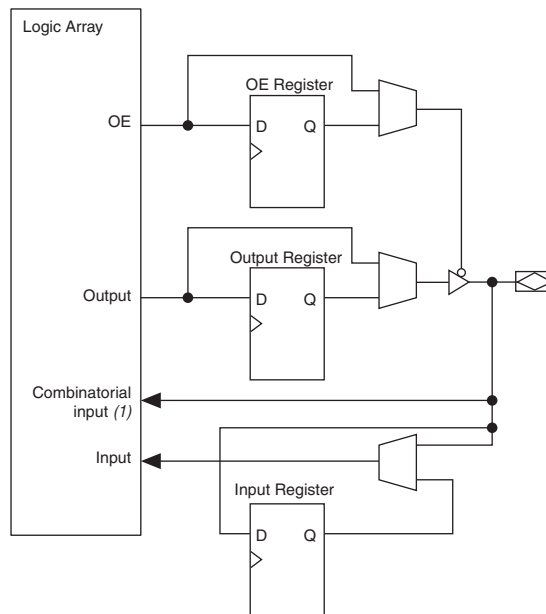
## I/O Structure

IOEs support many features, including:

- Differential and single-ended I/O standards
- 3.3-V, 64- and 32-bit, 66- and 33-MHz PCI compliance
- Joint Test Action Group (JTAG) boundary-scan test (BST) support
- Output drive strength control
- Weak pull-up resistors during configuration
- Slew-rate control
- Tri-state buffers
- Bus-hold circuitry
- Programmable pull-up resistors in user mode
- Programmable input and output delays
- Open-drain outputs
- DQ and DQS I/O pins

Cyclone device IOEs contain a bidirectional I/O buffer and three registers for complete embedded bidirectional single data rate transfer.

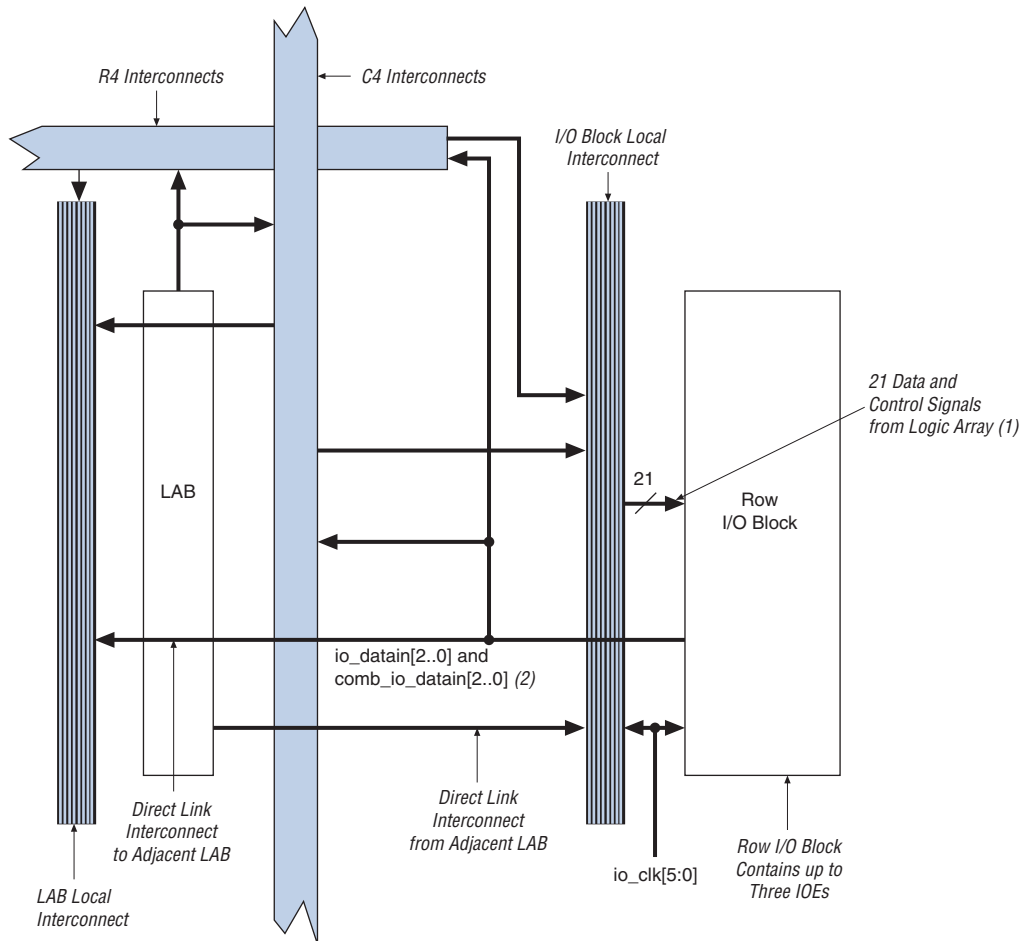
Figure 2–27 shows the Cyclone IOE structure. The IOE contains one input register, one output register, and one output enable register. The designer can use the input registers for fast setup times and output registers for fast clock-to-output times. Additionally, the designer can use the output enable (OE) register for fast clock-to-output enable timing. The Quartus II software automatically duplicates a single OE register that controls multiple output or bidirectional pins. IOEs can be used as input, output, or bidirectional pins.

**Figure 2–27. Cyclone IOE Structure****Note to Figure 2–27:**

- (1) There are two paths available for combinatorial inputs to the logic array. Each path contains a unique programmable delay chain.

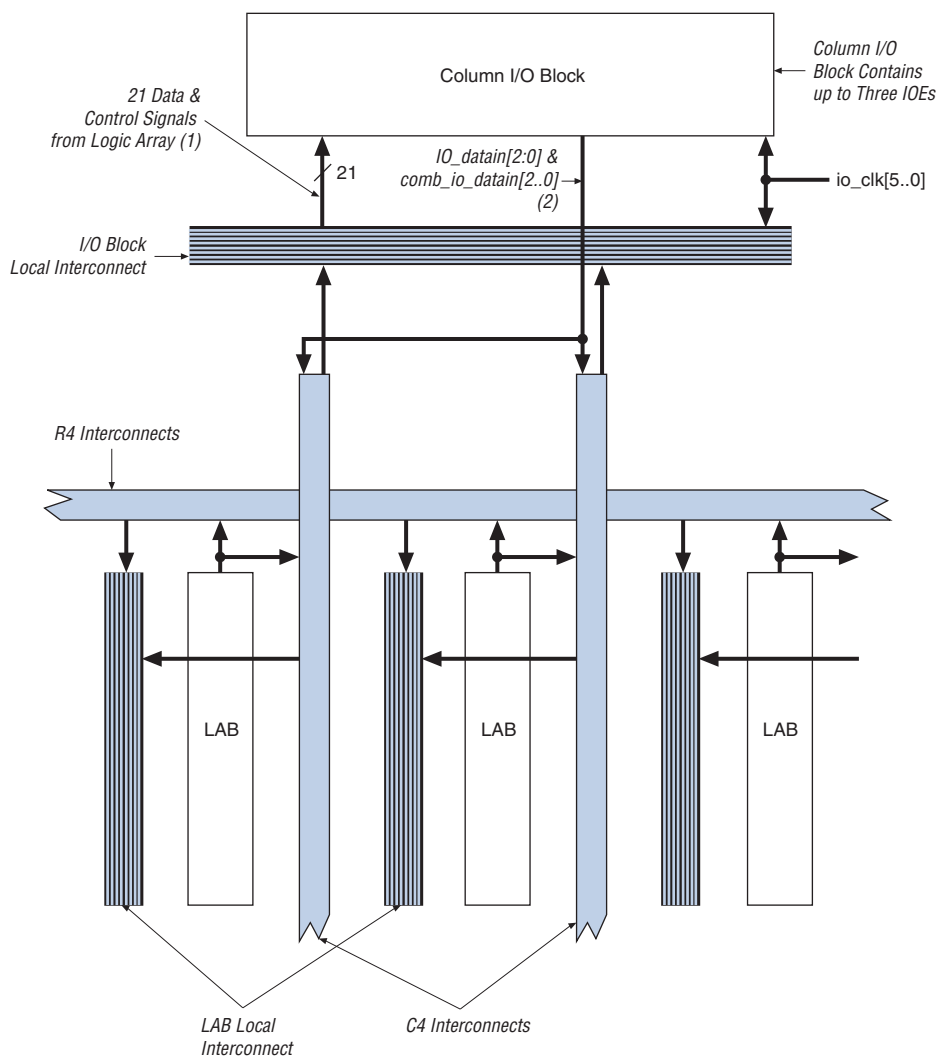
The IOEs are located in I/O blocks around the periphery of the Cyclone device. There are up to three IOEs per row I/O block and up to three IOEs per column I/O block (column I/O blocks span two columns). The row I/O blocks drive row, column, or direct link interconnects. The column I/O blocks drive column interconnects. [Figure 2–28](#) shows how a row I/O block connects to the logic array. [Figure 2–29](#) shows how a column I/O block connects to the logic array.

Figure 2–28. Row I/O Block Connection to the Interconnect

**Notes to Figure 2–28:**

- (1) The 21 data and control signals consist of three data out lines,  $io\_dataout[2..0]$ , three output enables,  $io\_coe[2..0]$ , three input clock enables,  $io\_cce\_in[2..0]$ , three output clock enables,  $io\_cce\_out[2..0]$ , three clocks,  $io\_clk[2..0]$ , three asynchronous clear signals,  $io\_cac1r[2..0]$ , and three synchronous clear signals,  $io\_csc1r[2..0]$ .
- (2) Each of the three IOEs in the row I/O block can have one  $io\_datain$  input (combinatorial or registered) and one  $comb\_io\_datain$  (combinatorial) input.

Figure 2–29. Column I/O Block Connection to the Interconnect

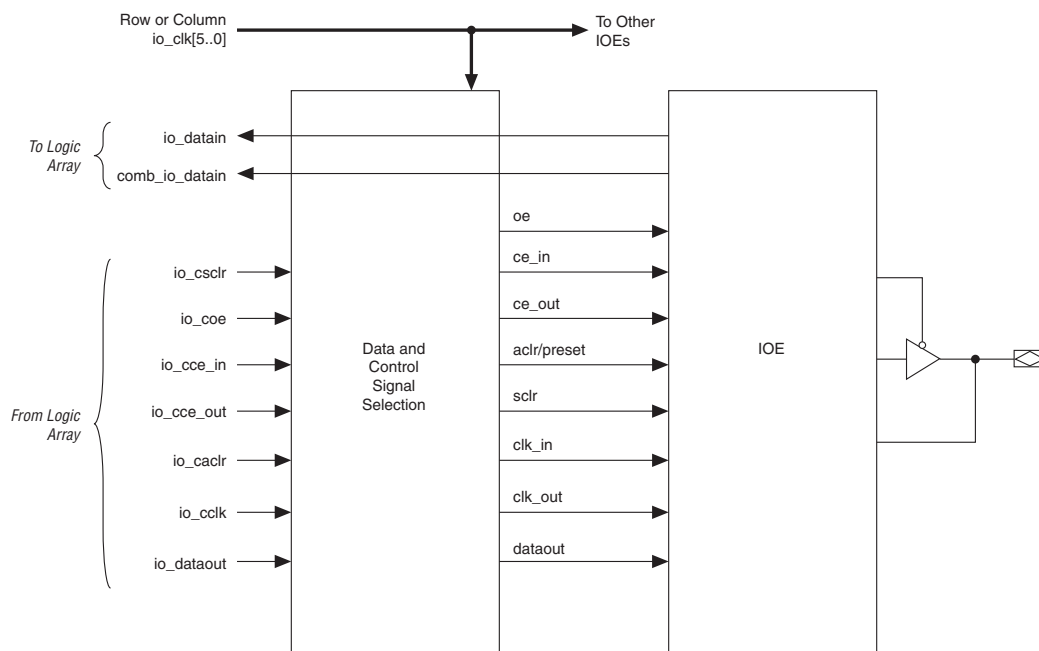
**Notes to Figure 2–29:**

- (1) The 21 data and control signals consist of three data out lines, `io_dataout[2..0]`, three output enables, `io_coe[2..0]`, three input clock enables, `io_cce_in[2..0]`, three output clock enables, `io_cce_out[2..0]`, three clocks, `io_clk[2..0]`, three asynchronous clear signals, `io_caclr[2..0]`, and three synchronous clear signals, `io_csclr[2..0]`.
- (2) Each of the three IOEs in the column I/O block can have one `io_datain` input (combinatorial or registered) and one `comb_io_datain` (combinatorial) input.

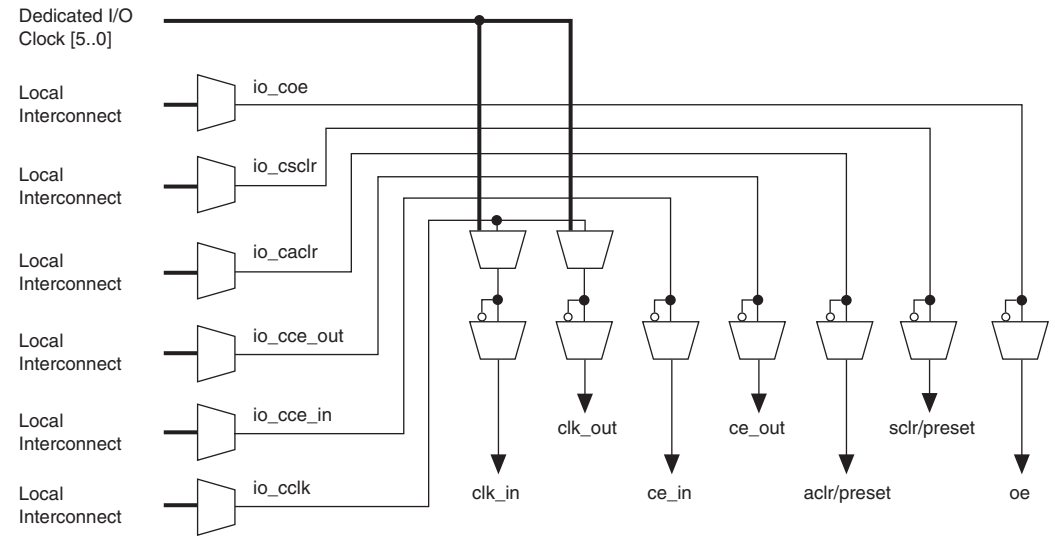


The pin's datain signals can drive the logic array. The logic array drives the control and data signals, providing a flexible routing resource. The row or column IOE clocks, `io_clk[5..0]`, provide a dedicated routing resource for low-skew, high-speed clocks. The global clock network generates the IOE clocks that feed the row or column I/O regions (see “Global Clock Network & Phase-Locked Loops” on page 2-29). Figure 2-30 illustrates the signal paths through the I/O block.

**Figure 2-30. Signal Path through the I/O Block**

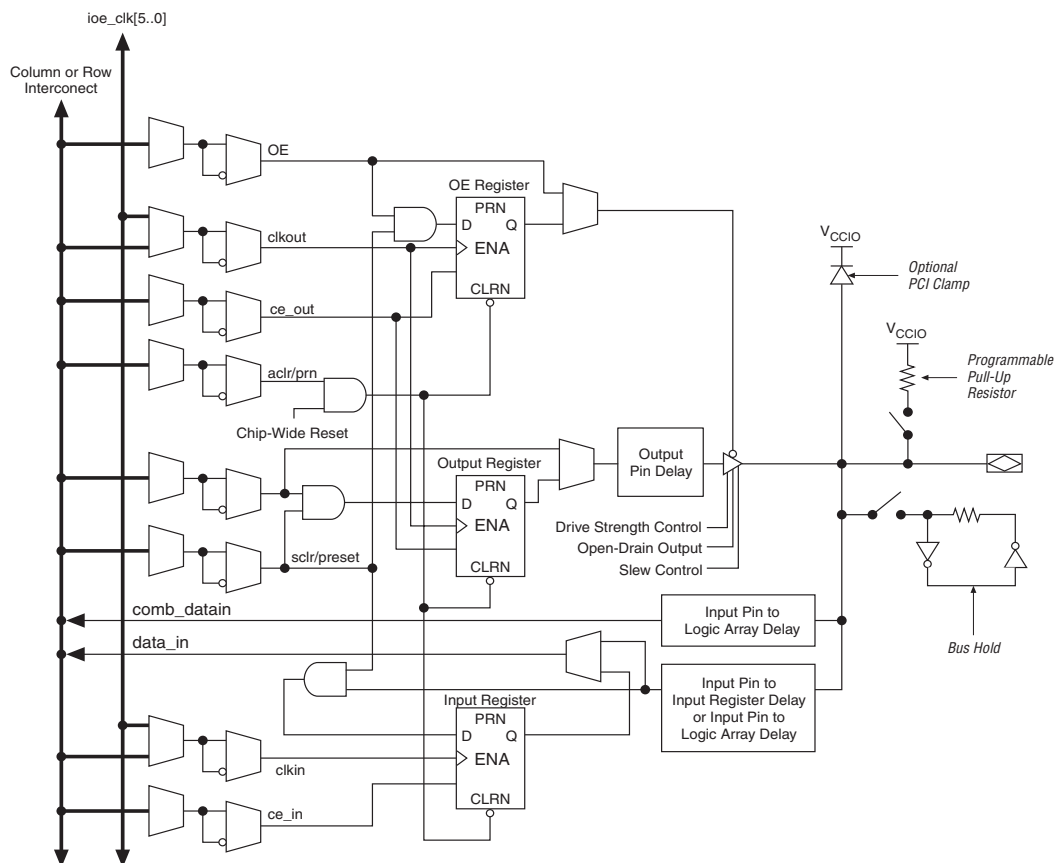


Each IOE contains its own control signal selection for the following control signals: `oe`, `ce_in`, `ce_out`, `aclr/preset`, `sclr/preset`, `clk_in`, and `clk_out`. Figure 2-31 illustrates the control signal selection.

**Figure 2–31. Control Signal Selection per IOE**

In normal bidirectional operation, the designer can use the input register for input data requiring fast setup times. The input register can have its own clock input and clock enable separate from the OE and output registers. The output register can be used for data requiring fast clock-to-output performance. The OE register is available for fast clock-to-output enable timing. The OE and output register share the same clock source and the same clock enable source from the local interconnect in the associated LAB, dedicated I/O clocks, or the column and row interconnects. [Figure 2–32](#) shows the IOE in bidirectional configuration.

**Figure 2–32. Cyclone IOE in Bidirectional I/O Configuration**



The Cyclone device IOE includes programmable delays to ensure zero hold times, minimize setup times, or increase clock to output times.

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 combinatorial logic may not require the delay. Programmable delays decrease input-pin-to-logic-array and IOE input register delays. 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. Table 2–9 shows the programmable delays for Cyclone devices.

<b>Programmable Delays</b>	<b>Quartus II Logic Option</b>
Input pin to logic array delay	Decrease input delay to internal cells
Input pin to input register delay	Decrease input delay to input registers
Output pin delay	Increase delay to output pin

There are two paths in the IOE for a combinatorial input to reach the logic array. Each of the two paths can have a different delay. This allows the designer to adjust delays from the pin to internal LE registers that reside in two different areas of the device. The designer sets the two combinatorial input delays by selecting different delays for two different paths under the **Decrease input delay to internal cells** logic option in the Quartus II software. When the input signal requires two different delays for the combinatorial input, the input register in the IOE is no longer available.

The IOE registers in Cyclone devices share the same source for clear or preset. The designer can program preset or clear for each individual IOE. The designer can also program the registers to power up high or low after configuration is complete. If programmed to power up low, an asynchronous clear can control the registers. If programmed to power up high, an asynchronous preset can control the registers. This feature prevents the inadvertent activation of another device's active-low input upon power up. If one register in an IOE uses a preset or clear signal then all registers in the IOE must use that same signal if they require preset or clear. Additionally a synchronous reset signal is available to the designer for the IOE registers.

## External RAM Interfacing

Cyclone devices support DDR SDRAM and FCRAM interfaces at up to 133 MHz through dedicated circuitry.

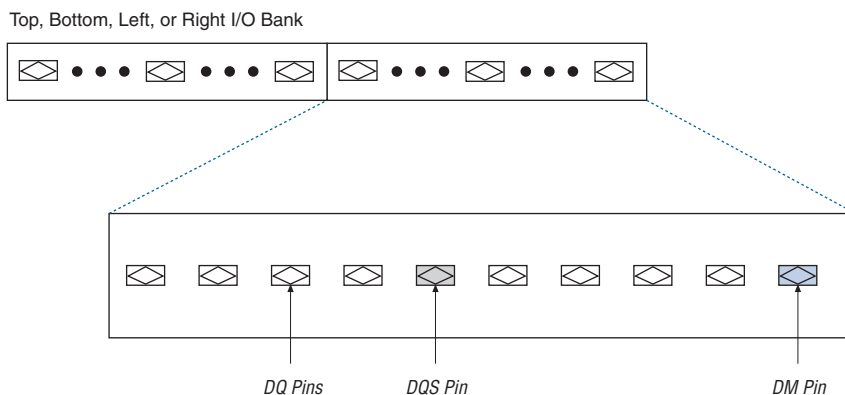
## DDR SDRAM & FCRAM

Cyclone devices have dedicated circuitry for interfacing with DDR SDRAM. All I/O banks support DDR SDRAM and FCRAM I/O pins. However, the configuration input pins in bank 1 must operate at 2.5 V because the SSTL-2  $V_{CCIO}$  level is 2.5 V. Additionally, the configuration

output pins ( $nSTATUS$  and  $CONF\_DONE$ ) and all the JTAG pins in I/O bank 3 must operate at 2.5 V because the  $V_{CCIO}$  level of SSTL-2 is 2.5 V. I/O banks 1, 2, 3, and 4 support DQS signals with DQ bus modes of  $\times 8$ .

For  $\times 8$  mode, there are up to eight groups of programmable DQS and DQ pins, I/O banks 1, 2, 3, and 4 each have two groups in the 324-pin and 400-pin FineLine BGA packages. Each group consists of one DQS pin, a set of eight DQ pins, and one DM pin (see Figure 2–33). Each DQS pin drives the set of eight DQ pins within that group.

**Figure 2–33. Cyclone Device DQ & DQS Groups in  $\times 8$  Mode** *Note (1)*



**Note to Figure 2–33:**

- (1) Each DQ group consists of one DQS pin, eight DQ pins, and one DM pin.

Table 2–10 shows the number of DQ pin groups per device.

Device	Package	Number of $\times 8$ DQ Pin Groups	Total DQ Pin Count
EP1C3	100-pin TQFP (1)	3	24
	144-pin TQFP	4	32
EP1C4	324-pin FineLine BGA	8	64
	400-pin FineLine BGA	8	64

<b>Device</b>	<b>Package</b>	<b>Number of × 8 DQ Pin Groups</b>	<b>Total DQ Pin Count</b>
EP1C6	144-pin TQFP	4	32
	240-pin PQFP	4	32
	256-pin FineLine BGA	4	32
EP1C12	240-pin PQFP	4	32
	256-pin FineLine BGA	4	32
	324-pin FineLine BGA	8	64
EP1C20	324-pin FineLine BGA	8	64
	400-pin FineLine BGA	8	64

**Note to Table 2–10:**

- (1) EP1C3 devices in the 100-pin TQFP package do not have any DQ pin groups in I/O bank 1.

A programmable delay chain on each DQS pin allows for either a 90° phase shift (for DDR SDRAM), or a 72° phase shift (for FCRAM) which automatically center-aligns input DQS synchronization signals within the data window of their corresponding DQ data signals. The phase-shifted DQS signals drive the global clock network. This global DQS signal clocks DQ signals on internal LE registers.

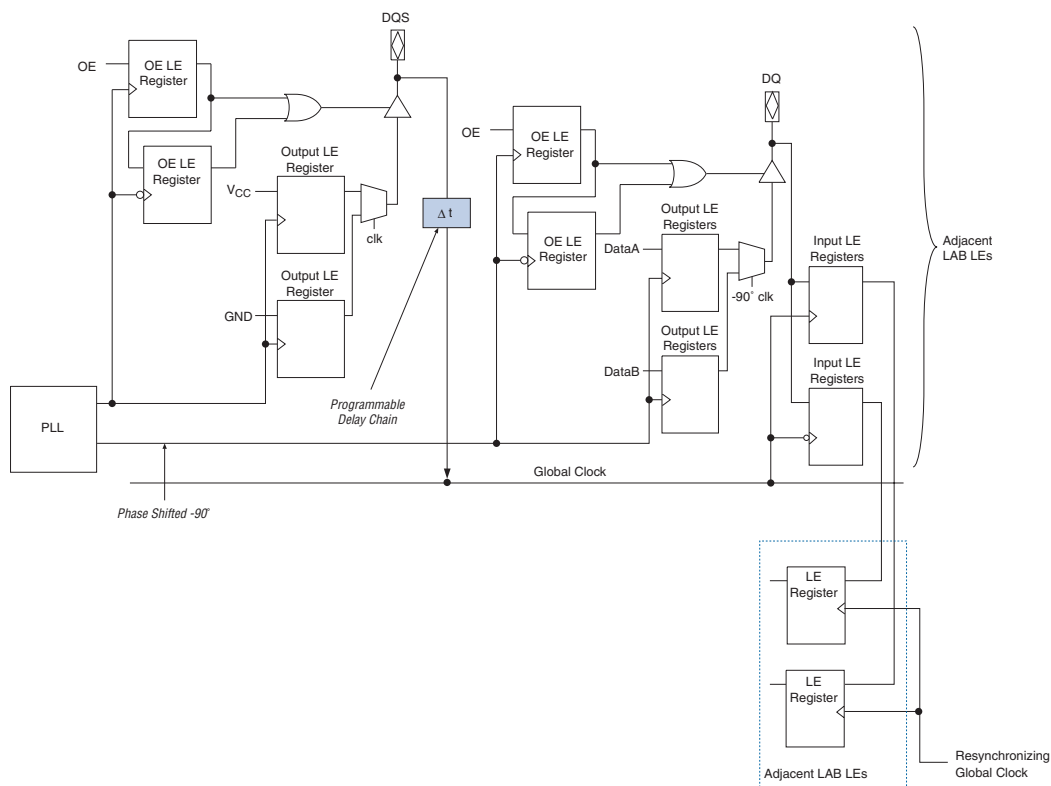
These DQS delay elements combine with the PLL's clocking and phase shift ability to provide a complete hardware solution for interfacing to high-speed memory.

The clock phase shift allows the PLL to clock the DQ output enable and output paths. The designer should use the following guidelines to meet 133 MHz performance for DDR SDRAM and FCRAM interfaces:

- The DQS signal must be in the middle of the DQ group it clocks
- Resynchronize the incoming data to the logic array clock using successive LE registers or FIFO buffers
- LE registers must be placed in the LAB adjacent to the DQ I/O pin column it is fed by

Figure 2–34 illustrates DDR SDRAM and FCRAM interfacing from the I/O through the dedicated circuitry to the logic array.

Figure 2-34. DDR SDRAM &amp; FCRAM Interfacing



## Programmable Drive Strength

The output buffer for each Cyclone device I/O pin has a programmable drive strength control for certain I/O standards. The LVTTTL and LVCMOS standards have several levels of drive strength that the designer can control. SSTL-3 class I and II, and SSTL-2 class I and II support a minimum setting, the lowest drive strength that guarantees the  $I_{OH}/I_{OL}$

of the standard. Using minimum settings provides signal slew rate control to reduce system noise and signal overshoot. Table 2–11 shows the possible settings for the I/O standards with drive strength control.

**Table 2–11. Programmable Drive Strength**

I/O Standard	I <sub>OH</sub> /I <sub>OL</sub> Current Strength Setting (mA)
LVTTTL (3.3 V)	4
	8
	12
	16
	24
LVCMOS (3.3 V)	2
	4
	8
	12
LVTTTL (2.5 V)	2
	8
	12
	16
LVTTTL (1.8 V)	2
	8
	12
LVCMOS (1.5 V)	2
	4
	8

## Open-Drain Output

Cyclone devices provide an optional open-drain (equivalent to an open-collector) output for each I/O pin. This open-drain output enables the device to provide system-level control signals (e.g., interrupt and write-enable signals) that can be asserted by any of several devices.

## Slew-Rate Control

The output buffer for each Cyclone device I/O pin has a programmable output slew-rate control that can be configured for low noise or high-speed performance. A faster slew rate provides high-speed transitions for high-performance systems. However, these fast transitions may introduce noise transients into the system. A slow slew rate reduces



system noise, but adds a nominal delay to rising and falling edges. Each I/O pin has an individual slew-rate control, allowing the designer to specify the slew rate on a pin-by-pin basis. The slew-rate control affects both the rising and falling edges.

## Bus Hold

Each Cyclone device I/O pin provides an optional bus-hold feature. The bus-hold circuitry can hold the signal on an I/O pin at its last-driven state. Since the bus-hold feature holds the last-driven state of the pin until the next input signal is present, an external pull-up or pull-down resistor is not necessary to hold a signal level when the bus is tri-stated.

The bus-hold circuitry also pulls undriven pins away from the input threshold voltage where noise can cause unintended high-frequency switching. The designer can select this feature individually for each I/O pin. The bus-hold output will drive no higher than  $V_{CCIO}$  to prevent overdriving signals. If the bus-hold feature is enabled, the device cannot use the programmable pull-up option. Disable the bus-hold feature when the I/O pin is configured for differential signals.

The bus-hold circuitry uses a resistor with a nominal resistance (RBH) of approximately 7 k $\Omega$  to pull the signal level to the last-driven state. [Table 4-15 on page 4-6](#) gives the specific sustaining current for each  $V_{CCIO}$  voltage level driven through this resistor and overdrive current used to identify the next-driven input level.

The bus-hold circuitry is only active after configuration. When going into user mode, the bus-hold circuit captures the value on the pin present at the end of configuration.

## Programmable Pull-Up Resistor

Each Cyclone device I/O pin provides an optional programmable pull-up resistor during user mode. If the designer enables this feature for an I/O pin, the pull-up resistor (typically 25 k $\Omega$ ) holds the output to the  $V_{CCIO}$  level of the output pin's bank.

## Advanced I/O Standard Support

Cyclone device IOEs support the following I/O standards:

- 3.3-V LVTTTL/LVCMOS
- 2.5-V LVTTTL/LVCMOS
- 1.8-V LVTTTL/LVCMOS
- 1.5-V LVCMOS
- 3.3-V PCI
- LVDS
- SSTL-2 class I and II
- SSTL-3 class I and II
- Differential SSTL-2 class II (on output clocks only)

Table 2–12 describes the I/O standards supported by Cyclone devices.

I/O Standard	Type	Input Reference Voltage ( $V_{REF}$ ) (V)	Output Supply Voltage ( $V_{CCIO}$ ) (V)	Board Termination Voltage ( $V_{TT}$ ) (V)
3.3-V LVTTTL/LVCMOS	Single-ended	N/A	3.3	N/A
2.5-V LVTTTL/LVCMOS	Single-ended	N/A	2.5	N/A
1.8-V LVTTTL/LVCMOS	Single-ended	N/A	1.8	N/A
1.5-V LVCMOS	Single-ended	N/A	1.5	N/A
3.3-V PCI (1)	Single-ended	N/A	3.3	N/A
LVDS (2)	Differential	N/A	2.5	N/A
SSTL-2 class I and II	Voltage-referenced	1.25	2.5	1.25
SSTL-3 class I and II	Voltage-referenced	1.5	3.3	1.5
Differential SSTL-2 (3)	Differential	1.25	2.5	1.25

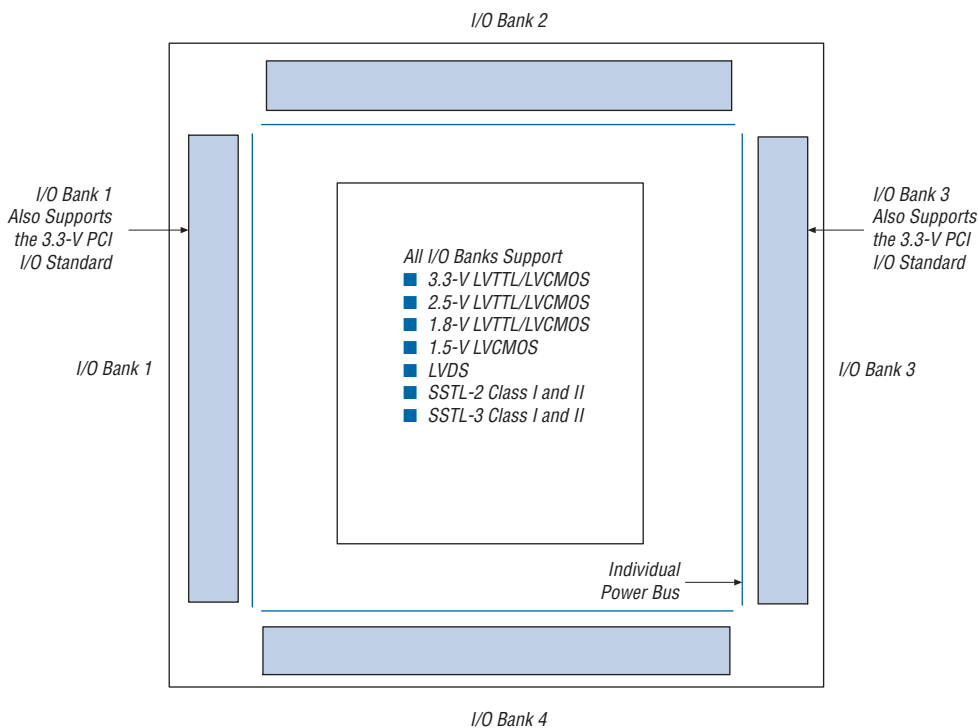
### Notes to Table 2–12:

- (1) EP1C3 devices support PCI by using the LVTTTL 16-mA I/O standard and drive strength assignments in the Quartus II software. The device requires an external diode for PCI compliance.
- (2) EP1C3 devices in the 100-pin TQFP package do not support the LVDS I/O standard.
- (3) This I/O standard is only available on output clock pins (PLL\_OUT pins).

Cyclone devices contain four I/O banks, as shown in Figure 2–35. I/O banks 1 and 3 support all the I/O standards listed in Table 2–12. I/O banks 2 and 4 support all the I/O standards listed in Table 2–12 except the 3.3-V PCI standard. I/O banks 2 and 4 contain dual-purpose DQS, DQ, and DM pins to support a DDR SDRAM or FCRAM interface. I/O bank 1 can also support a DDR SDRAM or FCRAM interface, however, the

configuration input pins in I/O bank 1 must operate at 2.5 V. I/O bank 3 can also support a DDR SDRAM or FCRAM interface, however, all the JTAG pins in I/O bank 3 must operate at 2.5 V.

**Figure 2–35. Cyclone I/O Banks** Notes (1), (2)



**Notes to Figure 2–35:**

- (1) Figure 2–35 is a top view of the silicon die.
- (2) Figure 2–35 is a graphic representation only. Refer to the pin list and the Quartus II software for exact pin locations.

Each I/O bank has its own  $V_{CCIO}$  pins. A single device can support 1.5-V, 1.8-V, 2.5-V, and 3.3-V interfaces; each individual bank can support a different standard with different I/O voltages. Each bank also has dual-purpose  $V_{REF}$  pins to support any one of the voltage-referenced standards (e.g., SSTL-3) independently. If an I/O bank does not use voltage-referenced standards, the  $V_{REF}$  pins are available as user I/O pins.

Each I/O bank can support multiple standards with the same  $V_{CCIO}$  for input and output pins. For example, when  $V_{CCIO}$  is 3.3-V, a bank can support LVTTTL, LVCMOS, 3.3-V PCI, and SSTL-3 for inputs and outputs.

## LVDS I/O Pins

A subset of pins in all four I/O banks supports LVDS interfacing. These dual-purpose LVDS pins require an external-resistor network at the transmitter channels in addition to 100- $\Omega$  termination resistors on receiver channels. These pins do not contain dedicated serialization or deserialization circuitry; therefore, internal logic performs serialization and deserialization functions.

Table 2–13 shows the total number of supported LVDS channels per device density.

Device	Pin Count	Number of LVDS Channels
EP1C3	100	(1)
	144	34
EP1C4	324	103
	400	129
EP1C6	144	29
	240	72
	256	72
EP1C12	240	66
	256	72
	324	103
EP1C20	324	95
	400	129

**Note to Table 2–13:**

- (1) EP1C3 devices in the 100-pin TQFP package do not support the LVDS I/O standard.

## MultiVolt I/O Interface

The Cyclone architecture supports the MultiVolt I/O interface feature, which allows Cyclone devices in all packages to interface with systems of different supply voltages. The devices have one set of  $V_{CC}$  pins for internal operation and input buffers ( $V_{CCINT}$ ), and four sets for I/O output drivers ( $V_{CCIO}$ ).

The Cyclone  $V_{CCINT}$  pins must always be connected to a 1.5-V power supply. If the  $V_{CCINT}$  level is 1.5 V, then input pins are 1.5-V, 1.8-V, 2.5-V, and 3.3-V tolerant. The  $V_{CCIO}$  pins can be connected to either a 1.5-V, 1.8-V,

2.5-V, or 3.3-V power supply, depending on the output requirements. The output levels are compatible with systems of the same voltage as the power supply (i.e., when  $V_{CCIO}$  pins are connected to a 1.5-V power supply, the output levels are compatible with 1.5-V systems). When  $V_{CCIO}$  pins are connected to a 3.3-V power supply, the output high is 3.3-V and is compatible with 3.3-V or 5.0-V systems. Table 2–14 summarizes Cyclone MultiVolt I/O support.

**Table 2–14. Cyclone MultiVolt I/O Support** *Note (1)*

$V_{CCIO}$ (V)	Input Signal					Output Signal				
	1.5 V	1.8 V	2.5 V	3.3 V	5.0 V	1.5 V	1.8 V	2.5 V	3.3 V	5.0 V
1.5	✓	✓	✓ (2)	✓ (2)		✓				
1.8		✓	✓	✓		✓ (3)	✓			
2.5			✓	✓		✓ (5)	✓ (5)	✓		
3.3			✓ (4)	✓	✓ (6)	✓ (7)	✓ (7)	✓ (7)	✓	✓ (8)

**Notes to Table 2–14:**

- (1) The PCI clamping diode must be disabled to drive an input with voltages higher than  $V_{CCIO}$ .
- (2) When  $V_{CCIO} = 1.5\text{-V}$  and a 2.5-V or 3.3-V input signal feeds an input pin, higher pin leakage current is expected.
- (3) When  $V_{CCIO} = 1.8\text{-V}$ , a Cyclone device can drive a 1.5-V device with 1.8-V tolerant inputs.
- (4) When  $V_{CCIO} = 3.3\text{-V}$  and a 2.5-V input signal feeds an input pin, the  $V_{CCIO}$  supply current will be slightly larger than expected.
- (5) When  $V_{CCIO} = 2.5\text{-V}$ , a Cyclone device can drive a 1.5-V or 1.8-V device with 2.5-V tolerant inputs.
- (6) Cyclone devices can be 5.0-V tolerant with the use of an external resistor and the internal PCI clamp diode.
- (7) When  $V_{CCIO} = 3.3\text{-V}$ , a Cyclone device can drive a 1.5-V, 1.8-V, or 2.5-V device with 3.3-V tolerant inputs.
- (8) When  $V_{CCIO} = 3.3\text{-V}$ , a Cyclone device can drive a device with 5.0-V LVTTTL inputs but not 5.0-V LVCMOS inputs.

## Power Sequencing & Hot Socketing

Because Cyclone devices can be used in a mixed-voltage environment, they have been designed specifically to tolerate any possible power-up sequence. Therefore, the  $V_{CCIO}$  and  $V_{CCINT}$  power supplies may be powered in any order.

Signals can be driven into Cyclone devices before and during power up without damaging the device. In addition, Cyclone devices do not drive out during power up. Once operating conditions are reached and the device is configured, Cyclone devices operate as specified by the user.



## IEEE Std. 1149.1 (JTAG) Boundary Scan Support

All Cyclone devices provide JTAG BST circuitry that complies with the IEEE Std. 1149.1a-1990 specification. JTAG boundary-scan testing can be performed either before or after, but not during configuration. Cyclone devices can also use the JTAG port for configuration together with either the Quartus® II software or hardware using either Jam Files (.jam) or Jam Byte-Code Files (.jbc).

Cyclone devices support reconfiguring the I/O standard settings on the IOE through the JTAG BST chain. The JTAG chain can update the I/O standard for all input and output pins any time before or during user mode. Designers can use this ability for JTAG testing before configuration when some of the Cyclone pins drive or receive from other devices on the board using voltage-referenced standards. Since the Cyclone device might not be configured before JTAG testing, the I/O pins might not be configured for appropriate electrical standards for chip-to-chip communication. Programming those I/O standards via JTAG allows designers to fully test I/O connection to other devices.

The JTAG pins support 1.5-V/1.8-V or 2.5-V/3.3-V I/O standards. The TDO pin voltage is determined by the V<sub>CCIO</sub> of the bank where it resides. The bank V<sub>CCIO</sub> selects whether the JTAG inputs are 1.5-V, 1.8-V, 2.5-V, or 3.3-V compatible.

Cyclone devices also use the JTAG port to monitor the operation of the device with the SignalTap® II embedded logic analyzer. Cyclone devices support the JTAG instructions shown in [Table 3-1](#).

**Table 3-1. Cyclone JTAG Instructions (Part 1 of 2)**

JTAG Instruction	Instruction Code	Description
SAMPLE/PRELOAD	00 0000 0101	Allows a snapshot of signals at the device pins to be captured and examined during normal device operation, and permits an initial data pattern to be output at the device pins. Also used by the SignalTap II embedded logic analyzer.
EXTEST (1)	00 0000 0000	Allows the external circuitry and board-level interconnects to be tested by forcing a test pattern at the output pins and capturing test results at the input pins.
BYPASS	11 1111 1111	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation.

<b>Table 3–1. Cyclone JTAG Instructions (Part 2 of 2)</b>		
<b>JTAG Instruction</b>	<b>Instruction Code</b>	<b>Description</b>
USERCODE	00 0000 0111	Selects the 32-bit USERCODE register and places it between the TDI and TDO pins, allowing the USERCODE to be serially shifted out of TDO.
IDCODE	00 0000 0110	Selects the IDCODE register and places it between TDI and TDO, allowing the IDCODE to be serially shifted out of TDO.
HIGHZ (1)	00 0000 1011	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation, while tri-stating all of the I/O pins.
CLAMP (1)	00 0000 1010	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation while holding I/O pins to a state defined by the data in the boundary-scan register.
ICR instructions		Used when configuring a Cyclone device via the JTAG port with a MasterBlaster™ or ByteBlasterMV™ download cable, or when using a Jam File or Jam Byte-Code File via an embedded processor.
PULSE_NCONFIG	00 0000 0001	Emulates pulsing the nCONFIG pin low to trigger reconfiguration even though the physical pin is unaffected.
CONFIG_IO	00 0000 1101	Allows configuration of I/O standards through the JTAG chain for JTAG testing. Can be executed before, after, or during configuration. Stops configuration if executed during configuration. Once issued, the CONFIG_IO instruction will hold nSTATUS low to reset the configuration device. nSTATUS is held low until the device is reconfigured.
SignalTap II instructions		Monitors internal device operation with the SignalTap II embedded logic analyzer.

**Note to Table 3–1:**

- (1) Bus hold and weak pull-up resistor features override the high-impedance state of HIGHZ, CLAMP, and EXTEST.



The Cyclone device instruction register length is 10 bits and the USERCODE register length is 32 bits. Tables 3–2 and 3–3 show the boundary-scan register length and device IDCODE information for Cyclone devices.

Device	Boundary-Scan Register Length
EP1C3	339
EP1C4	930
EP1C6	582
EP1C12	774
EP1C20	930

Device	IDCODE (32 bits) (1)			
	Version (4 Bits)	Part Number (16 Bits)	Manufacturer Identity (11 Bits)	LSB (1 Bit) (2)
EP1C3	0000	0010 0000 1000 0001	000 0110 1110	1
EP1C4	0000	0010 0000 1000 0101	000 0110 1110	1
EP1C6	0000	0010 0000 1000 0010	000 0110 1110	1
EP1C12	0000	0010 0000 1000 0011	000 0110 1110	1
EP1C20	0000	0010 0000 1000 0100	000 0110 1110	1

**Notes to Table 3–3:**

- (1) The most significant bit (MSB) is on the left.
- (2) The IDCODE's least significant bit (LSB) is always 1.

Figure 3-1 shows the timing requirements for the JTAG signals.

**Figure 3-1. Cyclone JTAG Waveforms**

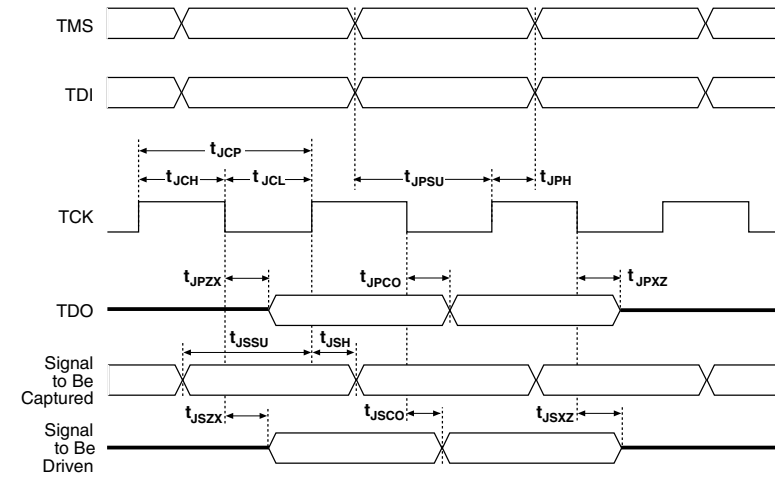


Table 3-4 shows the JTAG timing parameters and values for Cyclone devices.

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 to 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
$t_{JSH}$	Capture register hold time	45		ns
$t_{JSCO}$	Update register clock to output		35	ns
$t_{JSZX}$	Update register high impedance to valid output		35	ns
$t_{JSXZ}$	Update register valid output to high impedance		35	ns



For more information on JTAG, see the following documents:

- *AN 39: IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*
- *Jam Programming & Test Language Specification*

## SignalTap II Embedded Logic Analyzer

Cyclone devices feature the SignalTap II embedded logic analyzer, which monitors design operation over a period of time through the IEEE Std. 1149.1 (JTAG) circuitry. A designer can analyze internal logic at speed without bringing internal signals to the I/O pins. This feature is particularly important for advanced packages, such as FineLine BGA packages, because it can be difficult to add a connection to a pin during the debugging process after a board is designed and manufactured.

## Configuration

The logic, circuitry, and interconnects in the Cyclone architecture are configured with CMOS SRAM elements. Cyclone devices are reconfigurable and are 100% tested prior to shipment. As a result, the designer does not have to generate test vectors for fault coverage purposes, and can instead focus on simulation and design verification. In addition, the designer does not need to manage inventories of different ASIC designs. Cyclone devices can be configured on the board for the specific functionality required.

Cyclone devices are configured at system power-up with data stored in an Altera configuration device or provided by a system controller. The Cyclone device's optimized interface allows the device to act as controller in an active serial configuration scheme with the new low-cost serial configuration device. Cyclone devices can be configured in under 120 ms using serial data at 20 MHz. The serial configuration device can be programmed via the ByteBlaster II download cable, the Altera Programming Unit (APU), or third-party programmers.

In addition to the new low-cost serial configuration device, Altera offers in-system programmability (ISP)-capable configuration devices that can configure Cyclone devices via a serial data stream. The interface also enables microprocessors to treat Cyclone devices as memory and configure them by writing to a virtual memory location, making reconfiguration easy. After a Cyclone device has been configured, it can be reconfigured in-circuit by resetting the device and loading new data. Real-time changes can be made during system operation, enabling innovative reconfigurable computing applications.

## Operating Modes

The Cyclone architecture uses SRAM configuration elements that require configuration data to be loaded each time the circuit powers up. The process of physically loading the SRAM data into the device is called configuration. During initialization, which occurs immediately after configuration, the device resets registers, enables I/O pins, and begins to operate as a logic device. Together, the configuration and initialization processes are called command mode. Normal device operation is called user mode.

SRAM configuration elements allow Cyclone devices to be reconfigured in-circuit by loading new configuration data into the device. With real-time reconfiguration, the device is forced into command mode with a device pin. The configuration process loads different configuration data, reinitializes the device, and resumes user-mode operation. Designers can perform in-field upgrades by distributing new configuration files either within the system or remotely.

A built-in weak pull-up resistor pulls all user I/O pins to  $V_{CCIO}$  before and during device configuration.

The configuration pins support 1.5-V/1.8-V or 2.5-V/3.3-V I/O standards. The voltage level of the configuration output pins is determined by the  $V_{CCIO}$  of the bank where the pins reside. The bank  $V_{CCIO}$  selects whether the configuration inputs are 1.5-V, 1.8-V, 2.5-V, or 3.3-V compatible.

## Configuration Schemes

Designers can load the configuration data for a Cyclone device with one of three configuration schemes (see [Table 3–5](#)), chosen on the basis of the target application. Designers can use a configuration device, intelligent controller, or the JTAG port to configure a Cyclone device. A low-cost configuration device can automatically configure a Cyclone device at system power-up.

Multiple Cyclone devices can be configured in any of the three configuration schemes by connecting the configuration enable (nCE) and configuration enable output (nCEO) pins on each device.

<b>Configuration Scheme</b>	<b>Data Source</b>
Active serial	Low-cost serial configuration device
Passive serial (PS)	Enhanced or EPC2 configuration device, MasterBlaster or ByteBlasterMV download cable, or serial data source
JTAG	MasterBlaster or ByteBlasterMV download cable or a microprocessor with a Jam or JBC file



## Operating Conditions

Cyclone devices are offered in both commercial, industrial, and extended temperature grades. However, industrial-grade and extended-temperature-grade devices may have limited speed-grade availability.

Tables 4–1 through 4–16 provide information on absolute maximum ratings, recommended operating conditions, DC operating conditions, and capacitance for Cyclone devices.

**Table 4–1. Cyclone Device Absolute Maximum Ratings** *Notes (1), (2)*

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCINT}$	Supply voltage	With respect to ground (3)	–0.5	2.4	V
$V_{CCIO}$			–0.5	4.6	V
$V_I$	DC input voltage		–0.5	4.6	V
$I_{OUT}$	DC output current, per pin		–25	25	mA
$T_{STG}$	Storage temperature	No bias	–65	150	°C
$T_{AMB}$	Ambient temperature	Under bias	–65	135	°C
$T_J$	Junction temperature	BGA packages under bias		135	°C

**Table 4–2. Cyclone Device Recommended Operating Conditions (Part 1 of 2)**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCINT}$	Supply voltage for internal logic and input buffers	(4)	1.425	1.575	V
$V_{CCIO}$	Supply voltage for output buffers, 3.3-V operation	(4)	3.00	3.60	V
	Supply voltage for output buffers, 2.5-V operation	(4)	2.375	2.625	V
	Supply voltage for output buffers, 1.8-V operation	(4)	1.71	1.89	V
	Supply voltage for output buffers, 1.5-V operation	(4)	1.4	1.6	V
$V_I$	Input voltage	(3), (5)	–0.5	4.1	V

**Table 4–2. Cyclone Device Recommended Operating Conditions (Part 2 of 2)**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_O$	Output voltage		0	$V_{CCIO}$	V
$T_J$	Operating junction temperature	For commercial use	0	85	° C
		For industrial use	–40	100	° C
		For extended-temperature use	–40	125	° C
$t_R$	Input rise time			40	ns
$t_F$	Input fall time			40	ns

**Table 4–3. Cyclone Device DC Operating Conditions** *Note (6)*

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$I_I$	Input pin leakage current	$V_I = V_{CCIOmax}$ to 0 V (8)	–10		10	μA
$I_{OZ}$	Tri-stated I/O pin leakage current	$V_O = V_{CCIOmax}$ to 0 V (8)	–10		10	μA
$I_{CC0}$	$V_{CC}$ supply current (standby) (All M4K blocks in power-down mode) (7)	EP1C3		4		mA
		EP1C4		6		mA
		EP1C6		6		mA
		EP1C12		8		mA
		EP1C20		12		mA
$R_{CONF}$	Value of I/O pin pull-up resistor before and during configuration	$V_{CCIO} = 3.0$ V (9)	20		50	kΩ
		$V_{CCIO} = 2.375$ V (9)	30		80	kΩ
		$V_{CCIO} = 1.71$ V (9)	60		150	kΩ

**Table 4–4. LVTTTL Specifications**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCIO}$	Output supply voltage		3.0	3.6	V
$V_{IH}$	High-level input voltage		1.7	4.1	V
$V_{IL}$	Low-level input voltage		–0.5	0.7	V
$V_{OH}$	High-level output voltage	$I_{OH} = -4$ to $-24$ mA (10)	2.4		V
$V_{OL}$	Low-level output voltage	$I_{OL} = 4$ to $24$ mA (10)		0.45	V



**Table 4–5. LVCMOS Specifications**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCIO}$	Output supply voltage		3.0	3.6	V
$V_{IH}$	High-level input voltage		1.7	4.1	V
$V_{IL}$	Low-level input voltage		-0.5	0.7	V
$V_{OH}$	High-level output voltage	$V_{CCIO} = 3.0$ , $I_{OH} = -0.1$ mA	$V_{CCIO} - 0.2$		V
$V_{OL}$	Low-level output voltage	$V_{CCIO} = 3.0$ , $I_{OL} = 0.1$ mA		0.2	V

**Table 4–6. 2.5-V I/O Specifications**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCIO}$	Output supply voltage		2.375	2.625	V
$V_{IH}$	High-level input voltage		1.7	4.1	V
$V_{IL}$	Low-level input voltage		-0.5	0.7	V
$V_{OH}$	High-level output voltage	$I_{OH} = -0.1$ mA	2.1		V
		$I_{OH} = -1$ mA	2.0		V
		$I_{OH} = -2$ to $-16$ mA (10)	1.7		V
$V_{OL}$	Low-level output voltage	$I_{OL} = 0.1$ mA		0.2	V
		$I_{OL} = 1$ mA		0.4	V
		$I_{OL} = 2$ to $16$ mA (10)		0.7	V

**Table 4–7. 1.8-V I/O Specifications**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCIO}$	Output supply voltage		1.65	1.95	V
$V_{IH}$	High-level input voltage		$0.65 \times V_{CCIO}$	2.25	V
$V_{IL}$	Low-level input voltage		-0.3	$0.35 \times V_{CCIO}$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -2$ to $-8$ mA (10)	$V_{CCIO} - 0.45$		V
$V_{OL}$	Low-level output voltage	$I_{OL} = 2$ to $8$ mA (10)		0.45	V

**Table 4–8. 1.5-V I/O Specifications**

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CCIO}$	Output supply voltage		1.4	1.6	V
$V_{IH}$	High-level input voltage		$0.65 \times V_{CCIO}$	$V_{CCIO} + 0.3$	V
$V_{IL}$	Low-level input voltage		-0.3	$0.35 \times V_{CCIO}$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -2 \text{ mA}$ (10)	$0.75 \times V_{CCIO}$		V
$V_{OL}$	Low-level output voltage	$I_{OL} = 2 \text{ mA}$ (10)		$0.25 \times V_{CCIO}$	V

**Table 4–9. 2.5-V LVDS I/O Specifications** Note (11)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{CCIO}$	I/O supply voltage		2.375	2.5	2.625	V
$V_{OD}$	Differential output voltage	$R_L = 100 \Omega$	250		550	mV
$\Delta V_{OD}$	Change in $V_{OD}$ between high and low	$R_L = 100 \Omega$			50	mV
$V_{OS}$	Output offset voltage	$R_L = 100 \Omega$	1.125	1.25	1.375	V
$\Delta V_{OS}$	Change in $V_{OS}$ between high and low	$R_L = 100 \Omega$			50	mV
$V_{TH}$	Differential input threshold	$V_{CM} = 1.2 \text{ V}$	-100		100	mV
$V_{IN}$	Receiver input voltage range		0.0		2.4	V
$R_L$	Receiver differential input resistor		90	100	110	$\Omega$

**Table 4–10. 3.3-V PCI Specifications**

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{CCIO}$	Output supply voltage		3.0	3.3	3.6	V
$V_{IH}$	High-level input voltage		$0.5 \times V_{CCIO}$		$V_{CCIO} + 0.5$	V
$V_{IL}$	Low-level input voltage		-0.5		$0.3 \times V_{CCIO}$	V
$V_{OH}$	High-level output voltage	$I_{OUT} = -500 \mu\text{A}$	$0.9 \times V_{CCIO}$			V
$V_{OL}$	Low-level output voltage	$I_{OUT} = 1,500 \mu\text{A}$			$0.1 \times V_{CCIO}$	V

**Table 4–11. SSTL-2 Class I Specifications**

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{CCIO}$	Output supply voltage		2.375	2.5	2.625	V
$V_{TT}$	Termination voltage		$V_{REF} - 0.04$	$V_{REF}$	$V_{REF} + 0.04$	V
$V_{REF}$	Reference voltage		1.15	1.25	1.35	V
$V_{IH}$	High-level input voltage		$V_{REF} + 0.18$		3.0	V
$V_{IL}$	Low-level input voltage		-0.3		$V_{REF} - 0.18$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -8.1 \text{ mA}$ (10)	$V_{TT} + 0.57$			V
$V_{OL}$	Low-level output voltage	$I_{OL} = 8.1 \text{ mA}$ (10)			$V_{TT} - 0.57$	V

**Table 4–12. SSTL-2 Class II Specifications**

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{CCIO}$	Output supply voltage		2.3	2.5	2.7	V
$V_{TT}$	Termination voltage		$V_{REF} - 0.04$	$V_{REF}$	$V_{REF} + 0.04$	V
$V_{REF}$	Reference voltage		1.15	1.25	1.35	V
$V_{IH}$	High-level input voltage		$V_{REF} + 0.18$		$V_{CCIO} + 0.3$	V
$V_{IL}$	Low-level input voltage		-0.3		$V_{REF} - 0.18$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -16.4 \text{ mA}$ (10)	$V_{TT} + 0.76$			V
$V_{OL}$	Low-level output voltage	$I_{OL} = 16.4 \text{ mA}$ (10)			$V_{TT} - 0.76$	V

**Table 4–13. SSTL-3 Class I Specifications**

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{CCIO}$	Output supply voltage		3.0	3.3	3.6	V
$V_{TT}$	Termination voltage		$V_{REF} - 0.05$	$V_{REF}$	$V_{REF} + 0.05$	V
$V_{REF}$	Reference voltage		1.3	1.5	1.7	V
$V_{IH}$	High-level input voltage		$V_{REF} + 0.2$		$V_{CCIO} + 0.3$	V
$V_{IL}$	Low-level input voltage		-0.3		$V_{REF} - 0.2$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -8 \text{ mA}$ (10)	$V_{TT} + 0.6$			V
$V_{OL}$	Low-level output voltage	$I_{OL} = 8 \text{ mA}$ (10)			$V_{TT} - 0.6$	V

**Table 4–14. SSTL-3 Class II Specifications**

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{CCIO}$	Output supply voltage		3.0	3.3	3.6	V
$V_{TT}$	Termination voltage		$V_{REF} - 0.05$	$V_{REF}$	$V_{REF} + 0.05$	V
$V_{REF}$	Reference voltage		1.3	1.5	1.7	V
$V_{IH}$	High-level input voltage		$V_{REF} + 0.2$		$V_{CCIO} + 0.3$	V
$V_{IL}$	Low-level input voltage		-0.3		$V_{REF} - 0.2$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -16 \text{ mA}$ (10)	$V_{TT} + 0.8$			V
$V_{OL}$	Low-level output voltage	$I_{OL} = 16 \text{ mA}$ (10)			$V_{TT} - 0.8$	V

**Table 4–15. Bus Hold Parameters**

Parameter	Conditions	$V_{CCIO}$ Level								Unit
		1.5 V		1.8 V		2.5 V		3.3 V		
		Min	Max	Min	Max	Min	Max	Min	Max	
Low sustaining current	$V_{IN} > V_{IL}$ (maximum)			30		50		70		$\mu\text{A}$
High sustaining current	$V_{IN} < V_{IH}$ (minimum)			-30		-50		-70		$\mu\text{A}$
Low overdrive current	$0 \text{ V} < V_{IN} < V_{CCIO}$				200		300		500	$\mu\text{A}$
High overdrive current	$0 \text{ V} < V_{IN} < V_{CCIO}$				-200		-300		-500	$\mu\text{A}$

**Table 4–16. Cyclone Device Capacitance** *Note (12)*

Symbol	Parameter	Typical	Unit
$C_{IO}$	Input capacitance for user I/O pin	4.0	pF
$C_{LVDS}$	Input capacitance for dual-purpose LVDS/user I/O pin	4.7	pF
$C_{VREF}$	Input capacitance for dual-purpose $V_{REF}$ /user I/O pin.	12.0	pF
$C_{DPCLK}$	Input capacitance for dual-purpose $DPCLK$ /user I/O pin.	4.4	pF
$C_{CLK}$	Input capacitance for CLK pin.	4.7	pF

**Notes to Tables 4–1 through 4–16:**

- (1) See the *Operating Requirements for Altera Devices Data Sheet*.
- (2) Conditions beyond those listed in [Table 4–1](#) may cause permanent damage to a device. Additionally, device operation at the absolute maximum ratings for extended periods of time may have adverse effects on the device.
- (3) Minimum DC input is  $-0.5$  V. During transitions, the inputs may undershoot to  $-0.5$  V or overshoot to  $4.6$  V for input currents less than  $100$  mA and periods shorter than  $20$  ns.
- (4) Maximum  $V_{CC}$  rise time is  $100$  ms, and  $V_{CC}$  must rise monotonically.
- (5) All pins, including dedicated inputs, clock, I/O, and JTAG pins, may be driven before  $V_{CCINT}$  and  $V_{CCIO}$  are powered.
- (6) Typical values are for  $T_A = 25^\circ$  C,  $V_{CCINT} = 1.5$  V, and  $V_{CCIO} = 1.5$  V,  $1.8$  V,  $2.5$  V, and  $3.3$  V.
- (7)  $V_I =$  ground, no load, no toggling inputs.
- (8) This value is specified for normal device operation. The value may vary during power-up. This applies for all  $V_{CCIO}$  settings ( $3.3$ ,  $2.5$ ,  $1.8$ , and  $1.5$  V).
- (9) Pin pull-up resistance values will lower if an external source drives the pin higher than  $V_{CCIO}$ .
- (10) Drive strength is programmable according to values in [Table 4–14](#).
- (11) The Cyclone LVDS interface requires a resistor network outside of the transmitter channels.
- (12) Capacitance is sample-tested only. Capacitance is measured using time-domain reflections (TDR). Measurement accuracy is within  $\pm 0.5$  pF.

## Power Consumption

Designers can use the Altera web power calculator to estimate the device power.

Cyclone devices require a certain amount of power-up current to successfully power up because of the nature of the leading-edge process on which they are fabricated. Table 4–17 shows the maximum power-up current required to power up a Cyclone device.

Device	Maximum Power-Up Current Requirement	Unit
EP1C3	300	mA
EP1C4 (1)	400	mA
EP1C6 (2)	500	mA
EP1C12	900	mA
EP1C20	1,200	mA

**Notes to Table 4–17:**

- (1) The EP1C4 maximum power-up current is an estimated specification and may change.
- (2) The EP1C6 maximum power-up current is for all EP1C6 devices except for those with lot codes listed in the *Cyclone FPGA Family Errata Sheet*.

Designers should select power supplies and regulators that can supply this amount of current when designing with Cyclone devices. This specification is for commercial operating conditions. Measurements were performed with an isolated Cyclone device on the board. Decoupling capacitors were not used in this measurement. To factor in the current for decoupling capacitors, sum up the current for each capacitor using the following equation:

$$I = C (dV/dt)$$

The exact amount of current that will be consumed varies according to the process, temperature, and power ramp rate. If the power supply or regulator can supply more current than required, the Cyclone device may consume more current than the maximum current specified in Table 4–17. However, the device does not require any more current to successfully power up than what is listed in Table 4–17.

The duration of the  $I_{CCINT}$  power-up requirement depends on the  $V_{CCINT}$  voltage supply rise time. The power-up current consumption drops when the  $V_{CCINT}$  supply reaches approximately 0.75 V. For example, if the  $V_{CCINT}$  rise time has a linear rise of 15 ms, the current consumption spike will drop by 7.5 ms.

Typically, the user-mode current during device operation is lower than the power-up current in [Table 4-17](#). Altera recommends using the Cyclone Power Calculator, available on the Altera web site, to estimate the user-mode  $I_{CCINT}$  consumption and then select power supplies or regulators based on the higher value.

## Timing Model

The DirectDrive technology and MultiTrack interconnect ensure predictable performance, accurate simulation, and accurate timing analysis across all Cyclone device densities and speed grades. This section describes and specifies the performance, internal, external, and PLL timing specifications.

All specifications are representative of worst-case supply voltage and junction temperature conditions.

### Preliminary & Final Timing

Timing models can have either preliminary or final status. The Quartus® II software issues an informational message during the design compilation if the timing models are preliminary. [Table 4-18](#) shows the status of the Cyclone device timing models.

Preliminary status means the timing model is subject to change. Initially, timing numbers are created using simulation results, process data, and other known parameters. These tests are used to make the preliminary numbers as close to the actual timing parameters as possible.

Final timing numbers are based on actual device operation and testing. These numbers reflect the actual performance of the device under worst-case voltage and junction temperature conditions.

**Table 4–18. Cyclone Device Timing Model Status**

Device	Preliminary	Final
EP1C3		✓
EP1C4	✓	
EP1C6		✓
EP1C12		✓
EP1C20		✓

## Performance

The maximum internal logic array clock tree frequency is limited to the specifications shown in [Table 4–19](#).

**Table 4–19. Clock Tree Maximum Performance Specification**

Parameter	Definition	-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Clock tree $f_{MAX}$	Maximum frequency that the clock tree can support for clocking registered logic			405			320			275	MHz



Table 4–20 shows the Cyclone device performance for some common designs. All performance values were obtained with the Quartus II software compilation of library of parameterized modules (LPM) functions or megafunctions. These performance values are based on EP1C6 devices in 144-pin TQFP packages.

Resource Used	Design Size & Function	Mode	Resources Used			Performance		
			LEs	M4K Memory Bits	M4K Memory Blocks	-6 Speed Grade (MHz)	-7 Speed Grade (MHz)	-8 Speed Grade (MHz)
LE	16-to-1 multiplexer	-	21	-	-	405.00	320.00	275.00
	32-to-1 multiplexer	-	44	-	-	317.36	284.98	260.15
	16-bit counter	-	16	-	-	405.00	320.00	275.00
	64-bit counter (1)	-	66	-	-	208.99	181.98	160.75
M4K memory block	RAM 128 × 36 bit	Single port	-	4,608	1	256.00	222.67	197.01
	RAM 128 × 36 bit	Simple dual-port mode	-	4,608	1	255.95	222.67	196.97
	RAM 256 × 18 bit	True dual-port mode	-	4,608	1	255.95	222.67	196.97
	FIFO 128 × 36 bit	-	40	4,608	1	256.02	222.67	197.01
	Shift register 9 × 4 × 128	Shift register	11	4,536	1	255.95	222.67	196.97

Note to Table 4–20:

(1) The performance numbers for this function are from an EP1C6 device in a 240-pin PQFP package.

## Internal Timing Parameters

Internal timing parameters are specified on a speed grade basis independent of device density. Tables 4–21 through 4–24 describe the Cyclone device internal timing microparameters for LEs, IOEs, M4K memory structures, and MultiTrack interconnects.

Symbol	Parameter
$t_{SU}$	LE register setup time before clock
$t_H$	LE register hold time after clock

**Table 4–21. LE Internal Timing Microparameter Descriptions (Part 2 of 2)**

Symbol	Parameter
$t_{CO}$	LE register clock-to-output delay
$t_{LUT}$	LE combinatorial LUT delay for data-in to data-out
$t_{CLR}$	Minimum clear pulse width
$t_{PRE}$	Minimum preset pulse width
$t_{CLKHL}$	Minimum clock high or low time

**Table 4–22. IOE Internal Timing Microparameter Descriptions**

Symbol	Parameter
$t_{SU}$	IOE input and output register setup time before clock
$t_H$	IOE input and output register hold time after clock
$t_{CO}$	IOE input and output register clock-to-output delay
$t_{PIN2COMBOUT\_R}$	Row input pin to IOE combinatorial output
$t_{PIN2COMBOUT\_C}$	Column input pin to IOE combinatorial output
$t_{COMBIN2PIN\_R}$	Row IOE data input to combinatorial output pin
$t_{COMBIN2PIN\_C}$	Column IOE data input to combinatorial output pin
$t_{CLR}$	Minimum clear pulse width
$t_{PRE}$	Minimum preset pulse width
$t_{CLKHL}$	Minimum clock high or low time

**Table 4–23. M4K Block Internal Timing Microparameter Descriptions**

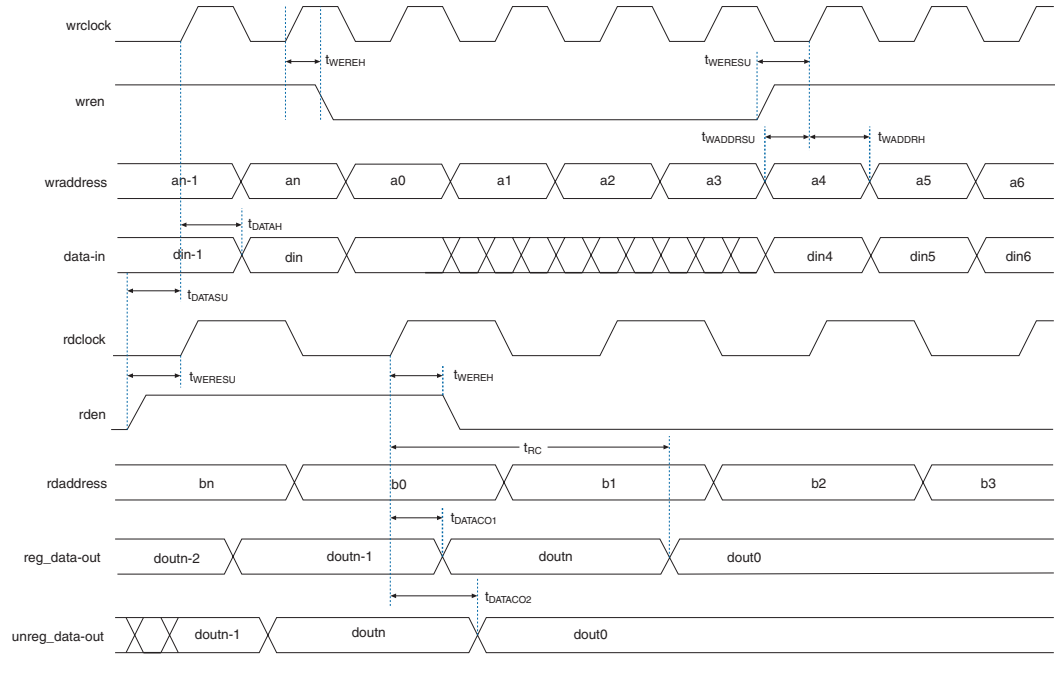
Symbol	Parameter
$t_{M4KRC}$	Synchronous read cycle time
$t_{M4KWC}$	Synchronous write cycle time
$t_{M4KWRESU}$	Write or read enable setup time before clock
$t_{M4KWEREH}$	Write or read enable hold time after clock
$t_{M4KBESU}$	Byte enable setup time before clock
$t_{M4KBEH}$	Byte enable hold time after clock
$t_{M4KDATAASU}$	A port data setup time before clock
$t_{M4KDATAAH}$	A port data hold time after clock
$t_{M4KADDRASU}$	A port address setup time before clock
$t_{M4KADDRAH}$	A port address hold time after clock
$t_{M4KDATABSU}$	B port data setup time before clock
$t_{M4KDATA BH}$	B port data hold time after clock
$t_{M4KADDRBSU}$	B port address setup time before clock
$t_{M4KADDRBH}$	B port address hold time after clock
$t_{M4KDATA CO1}$	Clock-to-output delay when using output registers
$t_{M4KDATA CO2}$	Clock-to-output delay without output registers
$t_{M4KCLKHL}$	Minimum clock high or low time
$t_{M4KCLR}$	Minimum clear pulse width

**Table 4–24. Routing Delay Internal Timing Microparameter Descriptions**

Symbol	Parameter
$t_{R4}$	Delay for an R4 line with average loading; covers a distance of four LAB columns
$t_{C4}$	Delay for an C4 line with average loading; covers a distance of four LAB rows
$t_{LOCAL}$	Local interconnect delay

Figure 4–1 shows the memory waveforms for the M4K timing parameters shown in Table 4–23.

**Figure 4–1. Dual-Port RAM Timing Microparameter Waveform**



Internal timing parameters are specified on a speed grade basis independent of device density. Tables 4–25 through 4–28 show the internal timing microparameters for LEs, IOEs, TriMatrix memory structures, DSP blocks, and MultiTrack interconnects.

**Table 4–25. LE Internal Timing Microparameters (Part 1 of 2)**

Symbol	-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	
$t_{SU}$	29		33		37		ps
$t_H$	12		13		15		ps
$t_{CO}$		173		198		224	ps
$t_{LUT}$		454		522		590	ps

**Table 4–25. LE Internal Timing Microparameters (Part 2 of 2)**

Symbol	-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	
$t_{CLR}$	129		148		167		ps
$t_{PRE}$	129		148		167		ps
$t_{CLKHL}$	107		123		139		ps

**Table 4–26. IOE Internal Timing Microparameters**

Symbol	-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	
$t_{SU}$	348		400		452		ps
$t_H$	0		0		0		ps
$t_{CO}$		511		587		664	ps
$t_{PIN2COMBOUT\_R}$		1,130		1,299		1,469	ps
$t_{PIN2COMBOUT\_C}$		1,135		1,305		1,475	ps
$t_{COMBIN2PIN\_R}$		2,627		3,021		3,415	ps
$t_{COMBIN2PIN\_C}$		2,615		3,007		3,399	ps
$t_{CLR}$	280		322		364		ps
$t_{PRE}$	280		322		364		ps
$t_{CLKHL}$	95		109		123		ps

**Table 4–27. M4K Block Internal Timing Microparameters (Part 1 of 2)**

Symbol	-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	
$t_{M4KRC}$		4,379		5,035		5,691	ps
$t_{M4KWC}$		2,910		3,346		3,783	ps
$t_{M4KWRESU}$	72		82		93		ps
$t_{M4KWEREH}$	43		49		55		ps
$t_{M4KBESU}$	72		82		93		ps
$t_{M4KBEH}$	43		49		55		ps
$t_{M4KDATAASU}$	72		82		93		ps
$t_{M4KDATAAH}$	43		49		55		ps

**Table 4–27. M4K Block Internal Timing Microparameters (Part 2 of 2)**

Symbol	-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	
$t_{M4KADDRASU}$	72		82		93		ps
$t_{M4KADDRAH}$	43		49		55		ps
$t_{M4KDATABSU}$	72		82		93		ps
$t_{M4KDATABH}$	43		49		55		ps
$t_{M4KADDRBSU}$	72		82		93		ps
$t_{M4KADDRBH}$	43		49		55		ps
$t_{M4KDATAO1}$		621		714		807	ps
$t_{M4KDATAO2}$		4,351		5,003		5,656	ps
$t_{M4KCLKHL}$	105		120		136		ps
$t_{M4KCLR}$	286		328		371		ps

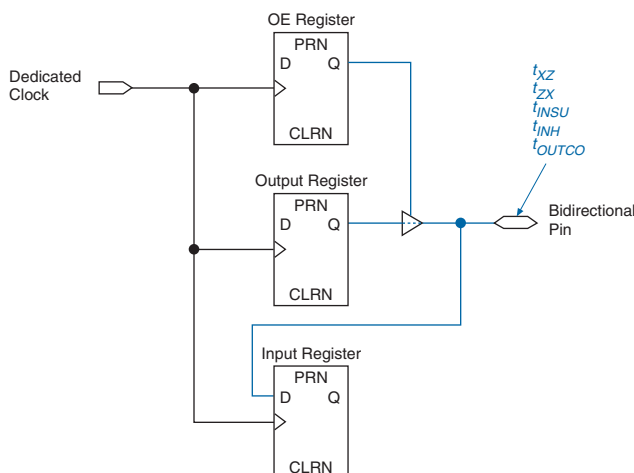
**Table 4–28. Routing Delay Internal Timing Microparameters**

Symbol	-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	
$t_{R4}$		261		300		339	ps
$t_{C4}$		338		388		439	ps
$t_{LOCAL}$		244		281		318	ps

## External Timing Parameters

External timing parameters are specified by device density and speed grade. [Figure 4–2](#) shows the timing model for bidirectional IOE pin timing. All registers are within the IOE.

Figure 4–2. External Timing in Cyclone Devices



All external I/O timing parameters shown are for 3.3-V LVTTTL I/O standard with the maximum current strength and fast slew rate. For external I/O timing using standards other than LVTTTL or for different current strengths, use the I/O standard input and output delay adders in Tables 4–40 through 4–44.

Table 4–29 shows the external I/O timing parameters when using global clock networks.

Symbol	Parameter	Conditions
$t_{INSU}$	Setup time for input or bidirectional pin using IOE input register with global clock fed by CLK pin	
$t_{INH}$	Hold time for input or bidirectional pin using IOE input register with global clock fed by CLK pin	
$t_{OUTCO}$	Clock-to-output delay output or bidirectional pin using IOE output register with global clock fed by CLK pin	$C_{LOAD} = 10 \text{ pF}$
$t_{XZ}$	Synchronous column IOE output enable register to output pin disable delay using global clock fed by CLK pin	$C_{LOAD} = 10 \text{ pF}$
$t_{ZX}$	Synchronous column IOE output enable register to output pin enable delay using global clock fed by CLK pin	$C_{LOAD} = 10 \text{ pF}$
$t_{INSUPLL}$	Setup time for input or bidirectional pin using IOE input register with global clock fed by Enhanced PLL with default phase setting	

**Table 4–29. Cyclone Global Clock External I/O Timing Parameters** Notes (1), (2) (Part 2 of 2)

Symbol	Parameter	Conditions
$t_{INHPLL}$	Hold time for input or bidirectional pin using IOE input register with global clock fed by enhanced PLL with default phase setting	
$t_{OUTCOPLL}$	Clock-to-output delay output or bidirectional pin using IOE output register with global clock enhanced PLL with default phase setting	$C_{LOAD} = 10\text{ pF}$
$t_{XZPLL}$	Synchronous column IOE output enable register to output pin disable delay using global clock fed by enhanced PLL with default phase setting	$C_{LOAD} = 10\text{ pF}$
$t_{ZXPLL}$	Synchronous column IOE output enable register to output pin enable delay using global clock fed by enhanced PLL with default phase setting	$C_{LOAD} = 10\text{ pF}$

**Notes to Table 4–29:**

- (1) These timing parameters are sample-tested only.
- (2) These timing parameters are for IOE pins using a 3.3-V LVTTTL, 24-mA setting. Designers should use the Quartus II software to verify the external timing for any pin.

Tables 4–30 through 4–31 show the external timing parameters on column and row pins for EP1C3 devices.

**Table 4–30. EP1C3 Column Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	3.085		3.547		4.009		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	4.073	2.000	4.682	2.000	5.295	ns
$t_{XZ}$		4.035		4.638		5.245	ns
$t_{ZX}$		4.035		4.638		5.245	ns
$t_{INSUPLL}$	1.795		2.063		2.332		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	2.306	0.500	2.651	0.500	2.998	ns
$t_{XZPLL}$		2.268		2.607		2.948	ns
$t_{ZXPLL}$		2.268		2.607		2.948	ns



**Table 4–31. EP1C3 Row Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	3.157		3.630		4.103		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.984	2.000	4.580	2.000	5.180	ns
$t_{XZ}$		3.905		4.489		5.077	ns
$t_{ZX}$		3.905		4.489		5.077	ns
$t_{INSUPLL}$	1.867		2.146		2.426		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	2.217	0.500	2.549	0.500	2.883	ns
$t_{XZPLL}$		2.138		2.458		2.780	ns
$t_{ZXPLL}$		2.138		2.458		2.780	ns

Tables 4–32 through 4–33 show the external timing parameters on column and row pins for EP1C4 devices.

**Table 4–32. EP1C4 Column Pin Global Clock External I/O Timing Parameters** *Note (1)*

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.671		3.071		3.470		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.937	2.000	4.526	2.000	5.119	ns
$t_{XZ}$		3.899		4.482		5.069	ns
$t_{ZX}$		3.899		4.482		5.069	ns
$t_{INSUPLL}$	1.471		1.690		1.910		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	2.080	0.500	2.392	0.500	2.705	ns
$t_{XZPLL}$		2.042		2.348		2.655	ns
$t_{ZXPLL}$		2.042		2.348		2.655	ns

**Table 4–33. EP1C4 Row Pin Global Clock External I/O Timing Parameters** *Note (1)*

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.800		3.220		3.639		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.791	2.000	4.358	2.000	4.929	ns
$t_{XZ}$		3.712		4.267		4.826	ns
$t_{ZX}$		3.712		4.267		4.826	ns
$t_{INSUPLL}$	1.600		1.839		2.079		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	1.934	0.500	2.224	0.500	2.515	ns
$t_{XZPLL}$		1.855		2.133		2.412	ns
$t_{ZXPLL}$		1.855		2.133		2.412	ns

*Note to Tables 4–32 and 4–33:*

(1) Contact Altera Applications for EP1C4 device timing parameters.

Tables 4–34 through 4–35 show the external timing parameters on column and row pins for EP1C6 devices.

**Table 4–34. EP1C6 Column Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.691		3.094		3.496		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.917	2.000	4.503	2.000	5.093	ns
$t_{XZ}$		3.879		4.459		5.043	ns
$t_{ZX}$		3.879		4.459		5.043	ns
$t_{INSUPLL}$	1.513		1.739		1.964		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	2.038	0.500	2.343	0.500	2.651	ns
$t_{XZPLL}$		2.000		2.299		2.601	ns
$t_{ZXPLL}$		2.000		2.299		2.601	ns

**Table 4–35. EP1C6 Row Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.774		3.190		3.605		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.817	2.000	4.388	2.000	4.963	ns
$t_{XZ}$		3.738		4.297		4.860	ns
$t_{ZX}$		3.738		4.297		4.860	ns
$t_{INSUPLL}$	1.596		1.835		2.073		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	1.938	0.500	2.228	0.500	2.521	ns
$t_{XZPLL}$		1.859		2.137		2.418	ns
$t_{ZXPLL}$		1.859		2.137		2.418	ns

Tables 4–36 through 4–37 show the external timing parameters on column and row pins for EP1C12 devices.

**Table 4–36. EP1C12 Column Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.510		2.885		3.259		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.798	2.000	4.367	2.000	4.940	ns
$t_{XZ}$		3.760		4.323		4.890	ns
$t_{ZX}$		3.760		4.323		4.890	ns
$t_{INSUPLL}$	1.588		1.824		2.061		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	1.663	0.500	1.913	0.500	2.164	ns
$t_{XZPLL}$		1.625		1.869		2.114	ns
$t_{ZXPLL}$		1.625		1.869		2.114	ns

**Table 4–37. EP1C12 Row Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.620		3.012		3.404		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.671	2.000	4.221	2.000	4.774	ns
$t_{XZ}$		3.592		4.130		4.671	ns
$t_{ZX}$		3.592		4.130		4.671	ns
$t_{INSUPLL}$	1.698		1.951		2.206		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	1.536	0.500	1.767	0.500	1.998	ns
$t_{XZPLL}$		1.457		1.676		1.895	ns
$t_{ZXPLL}$		1.457		1.676		1.895	ns

Tables 4–38 through 4–39 show the external timing parameters on column and row pins for EP1C20 devices.

**Table 4–38. EP1C20 Column Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.288		2.630		2.971		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.870	2.000	4.450	2.000	5.033	ns
$t_{XZ}$		3.832		4.406		4.983	ns
$t_{ZX}$		3.832		4.406		4.983	ns
$t_{INSUPLL}$	1.288		1.480		1.671		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	1.813	0.500	2.085	0.500	2.359	ns
$t_{XZPLL}$		1.775		2.041		2.309	ns
$t_{ZXPLL}$		1.775		2.041		2.309	ns

**Table 4–39. EP1C20 Row Pin Global Clock External I/O Timing Parameters**

Symbol	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
$t_{INSU}$	2.417		2.779		3.140		ns
$t_{INH}$	0.000		0.000		0.000		ns
$t_{OUTCO}$	2.000	3.724	2.000	4.282	2.000	4.843	ns
$t_{XZ}$		3.645		4.191		4.740	ns
$t_{ZX}$		3.645		4.191		4.740	ns
$t_{INSUPLL}$	1.417		1.629		1.840		ns
$t_{INHPLL}$	0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	0.500	1.667	0.500	1.917	0.500	2.169	ns
$t_{XZPLL}$		1.588		1.826		2.066	ns
$t_{ZXPLL}$		1.588		1.826		2.066	ns

### External I/O Delay Parameters

External I/O delay timing parameters for I/O standard input and output adders and programmable input and output delays are specified by speed grade independent of device density.

Tables 4–40 through 4–45 show the adder delays associated with column and row I/O pins for all packages. If an I/O standard is selected other than LVTTTL 24 mA with a fast slew rate, add the selected delay to the external  $t_{CO}$  and  $t_{SU}$  I/O parameters shown in Tables 4–25 through 4–28.

**Table 4–40. Cyclone I/O Standard Column Pin Input Delay Adders (Part 1 of 2)**

I/O Standard	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
LVC MOS		0		0		0	ps
3.3-V LVTTTL		0		0		0	ps
2.5-V LVTTTL		27		31		35	ps
1.8-V LVTTTL		182		209		236	ps
1.5-V LVTTTL		278		319		361	ps
SSTL-3 class I		–250		–288		–325	ps
SSTL-3 class II		–250		–288		–325	ps
SSTL-2 class I		–278		–320		–362	ps

**Table 4–40. Cyclone I/O Standard Column Pin Input Delay Adders (Part 2 of 2)**

I/O Standard	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
SSTL-2 class II		–278		–320		–362	ps
LVDS		–261		–301		–340	ps

**Table 4–41. Cyclone I/O Standard Row Pin Input Delay Adders**

I/O Standard	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
LVC MOS		0		0		0	ps
3.3-V LVTTTL		0		0		0	ps
2.5-V LVTTTL		27		31		35	ps
1.8-V LVTTTL		182		209		236	ps
1.5-V LVTTTL		278		319		361	ps
3.3-V PCI (1)		0		0		0	ps
SSTL-3 class I		–250		–288		–325	ps
SSTL-3 class II		–250		–288		–325	ps
SSTL-2 class I		–278		–320		–362	ps
SSTL-2 class II		–278		–320		–362	ps
LVDS		–261		–301		–340	ps

**Table 4–42. Cyclone I/O Standard Output Delay Adders for Fast Slew Rate on Column Pins (Part 1 of 2)**

Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
LVC MOS	2 mA		993		1,142		1,291	ps
	4 mA		504		579		655	ps
	8 mA		138		158		179	ps
	12 mA		0		0		0	ps
3.3-V LVTTTL	4 mA		993		1,142		1,291	ps
	8 mA		646		742		839	ps
	12 mA		135		155		175	ps
	16 mA		174		200		226	ps
	24 mA		0		0		0	ps

**Table 4–42. Cyclone I/O Standard Output Delay Adders for Fast Slew Rate on Column Pins (Part 2 of 2)**

Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
2.5-V LVTTTL	2 mA		1,322		1,520		1,718	ps
	8 mA		332		381		431	ps
	12 mA		338		388		439	ps
	16 mA		198		227		257	ps
1.8-V LVTTTL	2 mA		997		1,146		1,296	ps
	8 mA		785		902		1,020	ps
	12 mA		785		902		1,020	ps
1.5-V LVTTTL	2 mA		3,281		3,773		4,265	ps
	4 mA		1,601		1,841		2,081	ps
	8 mA		1,285		1,477		1,670	ps
SSTL-3 class I			583		670		758	ps
SSTL-3 class II			182		209		236	ps
SSTL-2 class I			508		584		660	ps
SSTL-2 class II			235		270		305	ps
LVDS			-5		-6		-7	ps

**Table 4–43. Cyclone I/O Standard Output Delay Adders for Fast Slew Rate on Row Pins (Part 1 of 2)**

Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		993		1,142		1,291	ps
	4 mA		504		579		655	ps
	8 mA		138		158		179	ps
	12 mA		0		0		0	ps
3.3-V LVTTTL	4 mA		993		1,142		1,291	ps
	8 mA		646		742		839	ps
	12 mA		135		155		175	ps
	16 mA		174		200		226	ps
	24 mA		0		0		0	ps
2.5-V LVTTTL	2 mA		1,322		1,520		1,718	ps
	8 mA		332		381		431	ps
	12 mA		338		388		439	ps
	16 mA		198		227		257	ps

**Table 4–43. Cyclone I/O Standard Output Delay Adders for Fast Slew Rate on Row Pins (Part 2 of 2)**

Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
1.8-V LVTTTL	2 mA		2,283		2,625		2,968	ps
	8 mA		997		1,146		1,296	ps
	12 mA		785		902		1,020	ps
1.5-V LVTTTL	2 mA		3,281		3,773		4,265	ps
	4 mA		1,601		1,841		2,081	ps
	8 mA		1,285		1,477		1,670	ps
3.3-V PCI (1)			116		133		150	ps
SSTL-3 class I			583		670		758	ps
SSTL-3 class II			182		209		236	ps
SSTL-2 class I			508		584		660	ps
SSTL-2 class II			235		270		305	ps
LVDS			-5		-6		-7	ps

**Table 4–44. Cyclone I/O Standard Output Delay Adders for Slow Slew Rate on Column Pins (Part 1 of 2)**

I/O Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		2,793		3,212		3,631	ps
	4 mA		2,304		2,649		2,995	ps
	8 mA		1,938		2,228		2,519	ps
	12 mA		1,800		2,070		2,340	ps
3.3-V LVTTTL	4 mA		2,824		3,247		3,671	ps
	8 mA		2,477		2,847		3,219	ps
	12 mA		1,966		2,260		2,555	ps
	16 mA		2,005		2,305		2,606	ps
	24 mA		1,831		2,105		2,380	ps
2.5-V LVTTTL	2 mA		3,740		4,300		4,861	ps
	8 mA		2,750		3,161		3,574	ps
	12 mA		2,756		3,168		3,582	ps
	16 mA		2,616		3,007		3,400	ps
1.8-V LVTTTL	2 mA		6,499		7,473		8,448	ps
	8 mA		5,213		5,994		6,776	ps
	12 mA		5,001		5,750		6,500	ps



**Table 4–44. Cyclone I/O Standard Output Delay Adders for Slow Slew Rate on Column Pins (Part 2 of 2)**

I/O Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
1.5-V LVTTTL	2 mA		7,782		8,949		10,116	ps
	4 mA		6,102		7,017		7,932	ps
	8 mA		5,786		6,653		7,521	ps
SSTL-3 class I			2,383		2,740		3,098	ps
SSTL-3 class II			1,982		2,279		2,576	ps
SSTL-2 class I			2,958		3,401		3,845	ps
SSTL-2 class II			2,685		3,087		3,490	ps
LVDS			1,795		2,064		2,333	ps

**Table 4–45. Cyclone I/O Standard Output Delay Adders for Slow Slew Rate on Row Pins (Part 1 of 2)**

I/O Standard		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		2,793		3,212		3,631	ps
	4 mA		2,304		2,649		2,995	ps
	8 mA		1,938		2,228		2,519	ps
	12 mA		1,800		2,070		2,340	ps
3.3-V LVTTTL	4 mA		2,824		3,247		3,671	ps
	8 mA		2,477		2,847		3,219	ps
	12 mA		1,966		2,260		2,555	ps
	16 mA		2,005		2,305		2,606	ps
	24 mA		1,831		2,105		2,380	ps
2.5-V LVTTTL	2 mA		3,740		4,300		4,861	ps
	8 mA		2,750		3,161		3,574	ps
	12 mA		2,756		3,168		3,582	ps
	16 mA		2,616		3,007		3,400	ps
1.8-V LVTTTL	2 mA		6,499		7,473		8,448	ps
	8 mA		5,213		5,994		6,776	ps
	12 mA		5,001		5,750		6,500	ps
1.5-V LVTTTL	2 mA		7,782		8,949		10,116	ps
	4 mA		6,102		7,017		7,932	ps
	8 mA		5,786		6,653		7,521	ps
3.3-V PCI			1,916		2,203		2,490	ps

**Table 4–45. Cyclone I/O Standard Output Delay Adders for Slow Slew Rate on Row Pins (Part 2 of 2)**

I/O Standard	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
SSTL-3 class I		2,383		2,740		3,098	ps
SSTL-3 class II		1,982		2,279		2,576	ps
SSTL-2 class I		2,958		3,401		3,845	ps
SSTL-2 class II		2,685		3,087		3,490	ps
LVDS		1,795		2,064		2,333	ps

Note to Tables 4–40 through 4–45:

- (1) EP1C3 devices do not support the PCI I/O standard.

Tables 4–46 through 4–47 show the adder delays for the IOE programmable delays. These delays are controlled with the Quartus II software options listed in the Parameter column.

**Table 4–46. Cyclone IOE Programmable Delays on Column Pins**

Parameter	Setting	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
Decrease input delay to internal cells	Off		3,057		3,515		3,974	ps
	Small		2,639		3,034		3,430	ps
	Medium		2,212		2,543		2,875	ps
	Large		155		178		201	ps
	On		155		178		201	ps
Decrease input delay to input register	Off		3,057		3,515		3,974	ps
	On		0		0		0	ps
Increase delay to output pin	Off		0		0		0	ps
	On		552		634		717	ps

**Table 4–47. Cyclone IOE Programmable Delays on Row Pins**

Parameter	Setting	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	
Decrease input delay to internal cells	Off		3,057		3,515		3,974	ps
	Small		2,639		3,034		3,430	ps
	Medium		2,212		2,543		2,875	ps
	Large		154		177		200	ps
	On		154		177		200	ps
Decrease input delay to input register	Off		3,057		3,515		3,974	ps
	On		0		0		0	ps
Increase delay to output pin	Off		0		0		0	ps
	On		556		639		722	ps

## Maximum Input & Output Clock Rates

Tables 4–48 and 4–49 show the maximum input clock rate for column and row pins in Cyclone devices.

**Table 4–48. Cyclone Maximum Input Clock Rate for Column Pins**

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTTL	464	428	387	MHz
2.5 V	392	302	207	MHz
1.8 V	387	311	252	MHz
1.5 V	387	320	243	MHz
LVC MOS	405	374	333	MHz
SSTL-3 class I	405	356	293	MHz
SSTL-3 class II	414	365	302	MHz
SSTL-2 class I	464	428	396	MHz
SSTL-2 class II	473	432	396	MHz
LVDS	567	549	531	MHz

**Table 4–49. Cyclone Maximum Input Clock Rate for Row Pins**

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTTL	464	428	387	MHz
2.5 V	392	302	207	MHz
1.8 V	387	311	252	MHz
1.5 V	387	320	243	MHz
LVCMOS	405	374	333	MHz
SSTL-3 class I	405	356	293	MHz
SSTL-3 class II	414	365	302	MHz
SSTL-2 class I	464	428	396	MHz
SSTL-2 class II	473	432	396	MHz
3.3-V PCI (1)	464	428	387	MHz
LVDS	567	549	531	MHz

Note to Tables 4–48 through 4–49:

- (1) EP1C3 devices do not support the PCI I/O standard. These parameters are only available on row I/O pins.

Tables 4–50 and 4–51 show the maximum output clock rate for column and row pins in Cyclone devices.

**Table 4–50. Cyclone Maximum Output Clock Rate for Column Pins**

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTTL	296	285	273	MHz
2.5 V	381	366	349	MHz
1.8 V	286	277	267	MHz
1.5 V	219	208	195	MHz
LVCMOS	367	356	343	MHz
SSTL-3 class I	169	166	162	MHz
SSTL-3 class II	160	151	146	MHz
SSTL-2 class I	160	151	142	MHz
SSTL-2 class II	131	123	115	MHz
LVDS	328	303	275	MHz

**Table 4–51. Cyclone Maximum Output Clock Rate for Row Pins**

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTTL	296	285	273	MHz
2.5 V	381	366	349	MHz
1.8 V	286	277	267	MHz
1.5 V	219	208	195	MHz
LVC MOS	367	356	343	MHz
SSTL-3 class I	169	166	162	MHz
SSTL-3 class II	160	151	146	MHz
SSTL-2 class I	160	151	142	MHz
SSTL-2 class II	131	123	115	MHz
3.3-V PCI (1)	66	66	66	MHz
LVDS	328	303	275	MHz

Note to Tables 4–50 through 4–51:

- (1) EP1C3 devices do not support the PCI I/O standard. These parameters are only available on row I/O pins.

## PLL Timing

Table 4–52 describes the Cyclone FPGA PLL specifications.

**Table 4–52. Cyclone PLL Specifications Note (1) (Part 1 of 2)**

Symbol	Parameter	Min	Max	Unit
$f_{IN}$	Input frequency (-6 speed grade)	15.625	464	MHz
	Input frequency (-7 speed grade)	15.625	428	MHz
	Input frequency (-8 speed grade)	15.625	387	MHz
$f_{IN}$ DUTY	Input clock duty cycle	40.00	60	%
$t_{IN}$ JITTER	Input clock period jitter		± 200	ps
$f_{OUT\_EXT}$ (external PLL clock output)	PLL output frequency (-6 speed grade)	15.625	320	MHz
	PLL output frequency (-7 speed grade)	15.625	320	MHz
	PLL output frequency (-8 speed grade)	15.625	275	MHz

**Table 4–52. Cyclone PLL Specifications** *Note (1) (Part 2 of 2)*

Symbol	Parameter	Min	Max	Unit
$f_{OUT}$ (to global clock)	PLL output frequency (-6 speed grade)	15.625	405	MHz
	PLL output frequency (-7 speed grade)	15.625	320	MHz
	PLL output frequency (-8 speed grade)	15.625	275	MHz
$t_{OUT DUTY}$	Duty cycle for external clock output (when set to 50%)	45.00	55	%
$t_{JITTER}$ (2)	Period jitter for external clock output		$\pm 300$ (3)	ps
$t_{LOCK}$ (4)	Time required to lock from end of device configuration	10.00	100	$\mu$ s
$f_{VCO}$	PLL internal VCO operating range	500.00	1,000	MHz
M	Counter values	2 to 32		integer
N, G0, G1, E	Counter values	1	32	integer

**Notes to Table 4–52:**

- (1) These numbers are preliminary and pending silicon characterization.
- (2) The  $t_{JITTER}$  specification for the PLL[2..1]\_OUT pins are dependent on the I/O pins in its  $V_{CCIO}$  bank, how many of them are switching outputs, how much they toggle, and whether or not they use programmable current strength or slow slew rate.
- (3)  $f_{OUT} \geq 100$  MHz. When the PLL external clock output frequency ( $f_{OUT}$ ) is smaller than 100 MHz, the jitter specification is 60 mUI.
- (4)  $f_{IN/N}$  must be greater than 200 MHz to ensure correct lock circuit operation below  $-20$  C.

## Software

Cyclone devices are supported by the Altera Quartus<sup>®</sup> II design software, which provides a comprehensive environment for system-on-a-programmable-chip (SOPC) design. The Quartus II software includes HDL and schematic design entry, compilation and logic synthesis, full simulation and advanced timing analysis, SignalTap II logic analysis, and device configuration. See the Design Software Selector Guide for more details on the Quartus II software features.

The Quartus II software supports the Windows 2000/NT/98, Sun Solaris, Linux Red Hat v7.1 and HP-UX operating systems. It also supports seamless integration with industry-leading EDA tools through the NativeLink<sup>®</sup> interface.

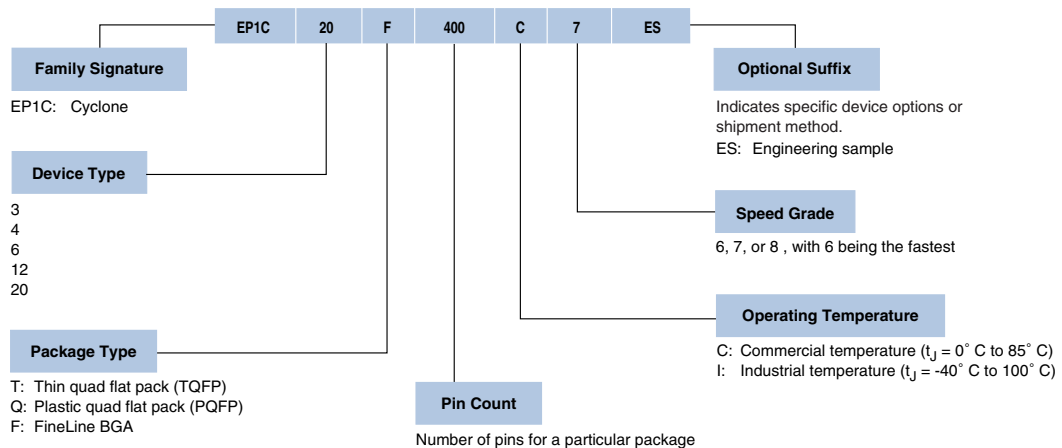
## Device Pin-Outs

Device pin-outs for Cyclone devices are available on the Altera web site ([www.altera.com](http://www.altera.com)) and in the *Cyclone FPGA Device Handbook*.

## Ordering Information

Figure 5–1 describes the ordering codes for Cyclone devices. For more information on a specific package, refer to [Chapter 6, Package Information for Cyclone Devices](#).

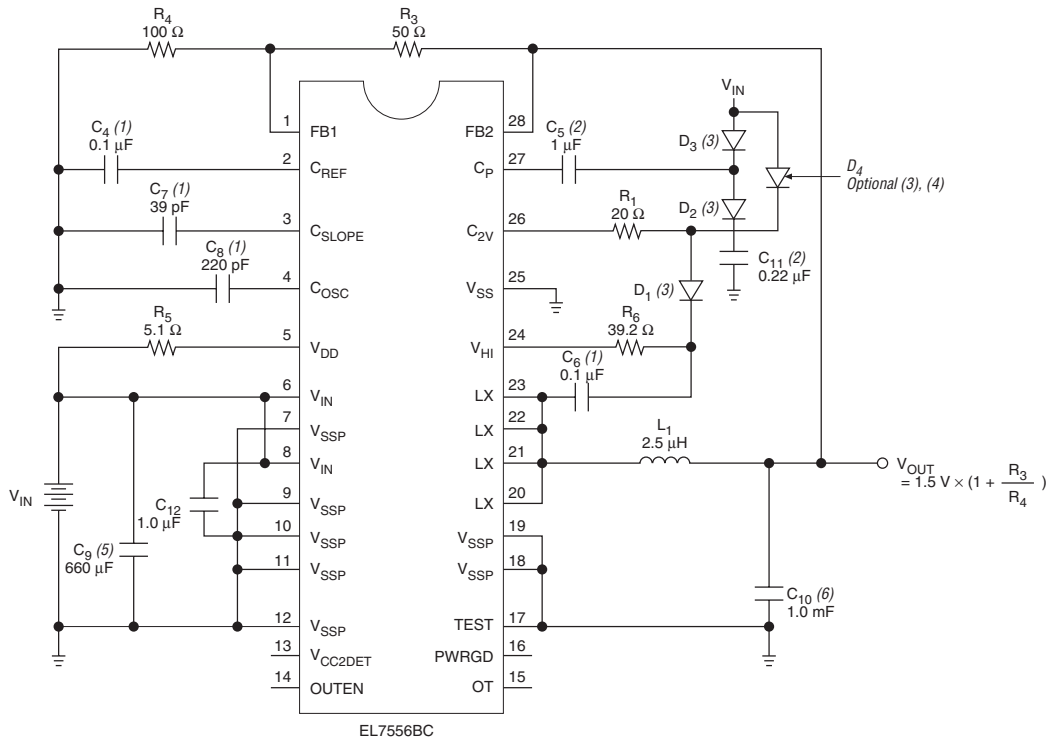
**Figure 5–1. Cyclone Device Packaging Ordering Information**







**Figure 12–15. EL7556BC: 5.0-V-to-1.5-V/6-A Synchronous Switching Regulator**

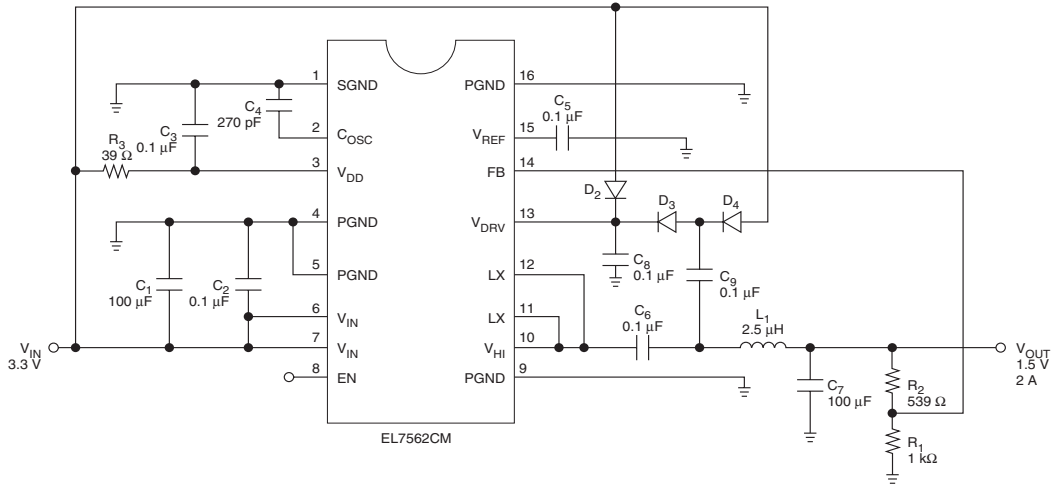


**Notes to Figures 12–13 – 12–15:**

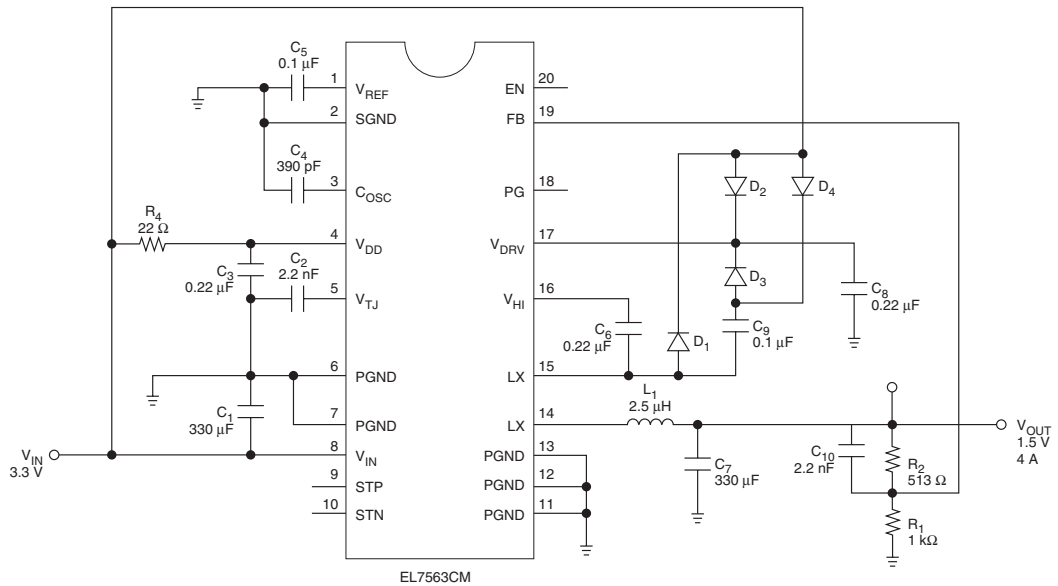
- (1) These capacitors are ceramic capacitors.
- (2) These capacitors are ceramic or tantalum capacitor.
- (3) These are BAT54S fast diodes.
- (4) D4 is only required for EL7556ACM.
- (5) This is a Sprague 293D337X96R3 2X330μF capacitor.
- (6) This is a Sprague 293D337X96R3 3X330μF capacitor.

Figures 12-16 and 12-17 show the switching regulator that converts 3.3 V to 1.5 V with different output currents.

**Figure 12-16. EL7562CM: 3.3-V to 1.5-V/2-A Synchronous Switching Regulator**



**Figure 12-17. EL7563CM: 3.3-V to 1.5-V/4-A Synchronous Switching Regulator**



## 1.5-V Regulator Application Examples

The following sections show the process used to select a voltage regulator for three sample designs. The regulator selection is based on the amount of power that the Cyclone device consumes. There are 14 variables to consider when selecting a voltage regulator. The following variables apply to Cyclone device power consumption:

- $f_{MAX}$
- Output and bidirectional pins
- Average toggle rate for I/O pins ( $to_{GIO}$ )
- Average toggle rate for logic elements (LEs) ( $to_{LC}$ )
- User-mode  $I_{CC}$  consumption
- Maximum power-up  $I_{CCINT}$  requirement
- Utilization
- $V_{CCIO}$  supply level
- $V_{CCINT}$  supply level

The following variables apply to the voltage regulator:

- Output voltage precision requirement
- Supply voltage on the board
- Voltage supply output current
- Variance of board supply
- Efficiency

Different designs have different power consumptions based on the variables listed. Once you calculate the Cyclone device's power consumption, you must consider how much current the Cyclone device needs. You can use the Cyclone power calculator (available at [www.altera.com](http://www.altera.com)) or the PowerGauge™ tool in the Quartus II software to determine the current needs. Also check the maximum power-up current requirement listed in the Power Consumption section of the Cyclone FPGA Family Data Sheet because the power-up current requirement may exceed the user-mode current consumption for a specific design.

Once you determine the minimum current the Cyclone device requires, you must select a voltage regulator that can generate the desired output current with the voltage and current supply that is available on the board using the variables listed in this section. An example is shown to illustrate the voltage regulator selection process.

## Synchronous Switching Regulator Example

This example shows a worst-case scenario for power consumption where the design uses all the LEs and RAM. Table 12–7 shows the design requirements for 1.5-V design using a Cyclone EP1C12 FPGA.

Design Requirement	Value
Output voltage precision requirement	±5%
Supply voltages available on the board	3.3 V
Voltage supply output current available for this section ( $I_{IN, DC(MAX)}$ )	2 A
Variance of board supply ( $V_{IN}$ )	±5%
$f_{MAX}$	150 MHz
Average $to_{gIO}$	12.5%
Average $to_{gLC}$	12.5%
Utilization	100%
Output and bidirectional pins	125
$V_{CCIO}$ supply level	3.3 V
$V_{CCINT}$ supply level	1.5 V
Efficiency	≥90%

Table 12–8 uses the checklist on page 12–8 to help select the appropriate voltage regulator.

Output voltage requirements	$V_{OUT} = 1.5\text{ V}$
Supply voltages	$V_{IN}$ OR $V_{CC} = 3.3\text{ V}$
Supply variance from Linear Technology data sheet	Supply variance = ±5%
Estimated $I_{CCINT}$ Use Cyclone Power Calculator	$I_{CCINT} = 620\text{ mA}$
Estimated $I_{CCIO}$ if regulator powers $V_{CCIO}$ Use Cyclone Power Calculator (not applicable in this example because $V_{CCIO} = 3.3\text{ V}$ )	$I_{CCIO} = \text{N/A}$
Total user-mode current consumption $I_{CC} = I_{CCINT} + I_{CCIO}$	$I_{CC} = 620\text{ mA}$

**Table 12–8. Voltage Regulator Selection Process for EP1C12F324C Design (Part 2 of 2)**

EP1C12 maximum power-up current requirement See Power Consumption section of the Cyclone FPGA Family Data Sheet for other densities	$I_{PUC(MAX)} = 900 \text{ mA}$
Maximum output current required Compare $I_{CC}$ with $I_{PUC(MAX)}$	$I_{OUT(MAX)} = 900 \text{ mA}$
Voltage regulator selection See <i>Linear Technology LTC 1649 data sheet</i> See <i>Intersil (Elantec) EL7562C data sheet</i>	LTC1649 $I_{OUT(MAX)} = 15 \text{ A}$ EL7562C $I_{OUT(MAX)} = 2 \text{ A}$
<b>LTC1649</b>	
Nominal efficiency ( $\eta$ )	Nominal efficiency ( $\eta$ ) = > 90%
Line and load regulation Line regulation + load regulation = $(0.17 \text{ mV} + 7 \text{ mV}) / 1.5 \text{ V} \times 100\%$	Line and Load Regulation = $0.478\% < 5\%$
Minimum input voltage ( $V_{IN(MIN)}$ ) $(V_{IN(MIN)}) = V_{IN}(1 - \Delta V_{IN}) = 3.3V(1 - 0.05)$	$(V_{IN(MIN)}) = 3.135 \text{ V}$
Maximum input current $I_{IN, DC(MAX)} = (V_{OUT} \times I_{OUT(MAX)}) / (\eta \times V_{IN(MIN)})$	$I_{IN, DC(MAX)} = 478 \text{ mA} < 2 \text{ A}$
<b>EL7562C</b>	
Nominal efficiency ( $\eta$ )	Nominal efficiency ( $\eta$ ) = > 95%
Line and load regulation Line regulation + load regulation = $(0.17 \text{ mV} + 7 \text{ mV}) / 1.5 \text{ V} \times 100\%$	Line and Load Regulation = $0.5\% < 5\%$
Minimum input voltage ( $V_{IN(MIN)}$ ) $(V_{IN(MIN)}) = V_{IN}(1 - \Delta V_{IN}) = 3.3V(1 - 0.05)$	$(V_{IN(MIN)}) = 3.135 \text{ V}$
Maximum input current $I_{IN, DC(MAX)} = (V_{OUT} \times I_{OUT(MAX)}) / (\eta \times V_{IN(MIN)})$	$I_{IN, DC(MAX)} = 453 \text{ mA} < 2 \text{ A}$

## Board Layout

Laying out a printed circuit board (PCB) properly is extremely important in high-frequency ( $\geq 100 \text{ kHz}$ ) switching regulator designs. A poor PCB layout results in increased EMI and ground bounce, which affects the reliability of the voltage regulator by obscuring important voltage and current feedback signals. Altera recommends using Gerber files — pre-designed layout files— supplied by the regulator vendor for your board layout.

If you cannot use the supplied layout files, contact the voltage regulator vendor for help on re-designing the board to fit your design requirements while maintaining the proper functionality.

Altera recommends that you use separate layers for signals, the ground plane, and voltage supply planes. You can support separate layers by using multi-layer PCBs, assuming you are using two signal layers.

Figure 12–18 shows how to use regulators to generate 1.5-V and 2.5-V power supplies if the system needs two power supply systems. One regulator is used for each power supply.

**Figure 12–18. Two Regulator Solution for Systems that Require 5.0-V, 2.5-V & 1.5-V Supply Levels**

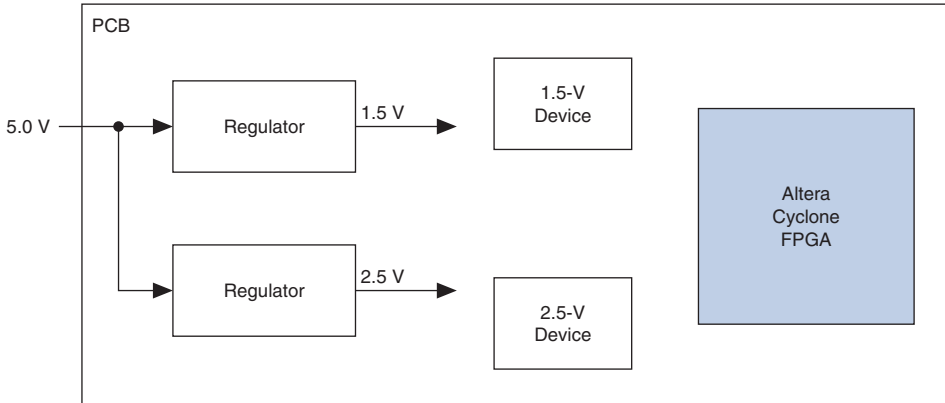
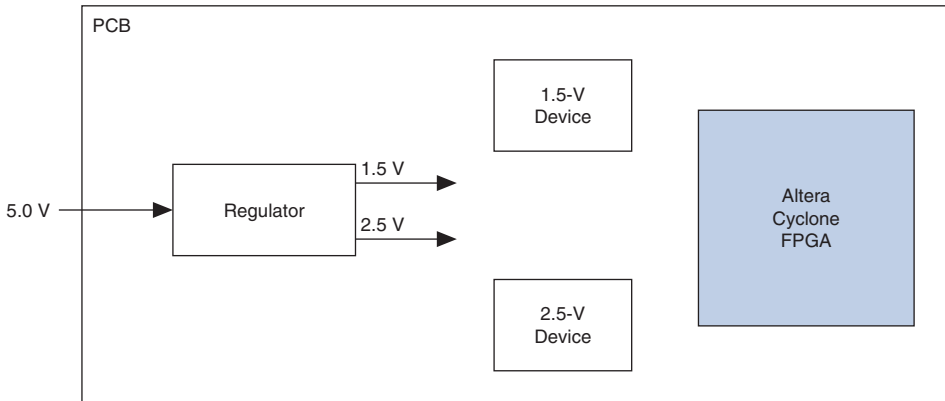


Figure 12–19 shows how to use a single regulator to generate two different power supplies (1.5-V and 2.5-V). The use of a single regulator to generate 1.5-V and 2.5-V supplies from the 5.0-V power supply can minimize the board size and thus save cost.

**Figure 12–19. Single Regulator Solution for Systems that Require 5.0-V, 2.5-V & 1.5-V Supply Levels**



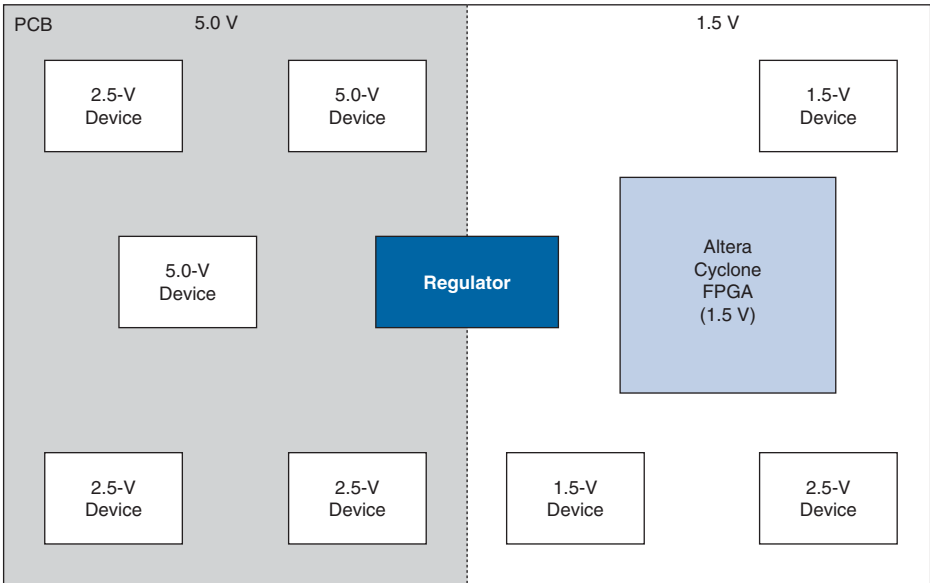
### Split-Plane Method

The split-plane design method reduces the number of planes required by placing two power supply planes in one plane (see Figure 12-20). For example, the layout for this method can be structured as follows:

- One 2.5-V plane, covering the entire board
- One plane split between 5.0-V and 1.5-V

This technique assumes that the majority of devices are 2.5-V. To support MultiVolt I/O, Altera devices must have access to 1.5-V and 2.5-V planes.

Figure 12-20. Split Board Layout for 2.5-V Systems With 5.0-V & 1.5-V Devices



### Conclusion

With the proliferation of multiple voltage levels in systems, it is important to design a voltage system that can support a low-power device like Cyclone devices. Designers must consider key elements of the PCB, such as power supplies, regulators, power consumption, and board layout when successfully designing a system that incorporates the low-voltage Cyclone family of devices.

## References

Linear Technology Corporation. *Application Note 35 (Step-Down Switching Regulators)*. Milpitas: Linear Technology Corporation, 1989.

Linear Technology Corporation. *LT1573 Data Sheet (Low Dropout Regulator Driver)*. Milpitas: Linear Technology Corporation, 1997.

Linear Technology Corporation. *LT1083/LT1084/LT1085 Data Sheet (7.5 A, 5 A, 3 A Low Dropout Positive Adjustable Regulators)*. Milpitas: Linear Technology Corporation, 1994.

Linear Technology Corporation. *LTC1649 Data Sheet (3.3V Input High Power Step-Down Switching Regulator Controller)*. Milpitas: Linear Technology Corporation, 1998.

Linear Technology Corporation. *LTC1775 Data Sheet (High Power No Rsense Current Mode Synchronous Step-Down Switching Regulator)*. Milpitas: Linear Technology Corporation, 1999.

Intersil Corporation. *EL7551C Data Sheet (Monolithic 1 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.

Intersil Corporation. *EL7564C Data Sheet (Monolithic 4 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.

Intersil Corporation. *EL7556BC Data Sheet (Integrated Adjustable 6 Amp Synchronous Switcher)*. Milpitas: Intersil Corporation, 2001.

Intersil Corporation. *EL7562C Data Sheet (Monolithic 2 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.

Intersil Corporation. *EL7563C Data Sheet (Monolithic 4 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.



This section provides information for all of the supported configuration schemes for Cyclone devices. The last chapter provides information on EPCS1 and EPCS4 serial configuration devices.

This section contains the following chapters:

- [Chapter 13. Configuring Cyclone FPGAs](#)
- [Chapter 14. Serial Configuration Devices \(EPCS1 & EPCS4\) Data Sheet](#)

## Revision History

The table below shows the revision history for [Chapter 13](#) and [14](#).

Chapter(s)	Date / Version	Changes Made
13	July 2003 v1.1	Updated <b>.rbf</b> sizes. Minor updates throughout the document.
	May 2003 v1.0	Added document to Cyclone Device Handbook.
14	October 2003 v1.2	Added Serial Configuration Device Memory Access section. Updated timing information in <a href="#">Tables 14–10</a> and <a href="#">14–11</a> .
	July 2003 v1.1	Minor updates.
	May 2003 v1.0	Added document to Cyclone Device Handbook.



## Introduction

You can configure Cyclone™ FPGAs using one of several configuration schemes, including the new active serial (AS) configuration scheme. This new scheme is used with the new, low cost serial configuration devices. Passive serial (PS) and Joint Test Action Group (JTAG)-based configuration schemes are also supported by Cyclone FPGAs. Additionally, Cyclone FPGAs can receive a compressed configuration bit stream and decompress this data in real-time, reducing storage requirements and configuration time.

This chapter describes how to configure Cyclone devices using each of the three supported configuration schemes.

## Device Configuration Overview

Cyclone FPGAs use SRAM cells to store configuration data. Since SRAM memory is volatile, configuration data must be downloaded to Cyclone FPGAs each time the device powers up. You can download configuration data to Cyclone FPGAs using the AS, PS, or JTAG interfaces. See [Table 13-1](#).

<i>Table 13-1. Cyclone FPGA Configuration Schemes</i>	
<b>Configuration Scheme</b>	<b>Description</b>
Active serial (AS) configuration	Configuration using: <ul style="list-style-type: none"> <li>• Serial configuration devices (EPCS1 or EPCS4)</li> </ul>
Passive serial (PS) configuration	Configuration using: <ul style="list-style-type: none"> <li>• Enhanced configuration devices (EPC4, EPC8, and EPC16)</li> <li>• EPC2, EPC1 configuration devices</li> <li>• Intelligent host (microprocessor)</li> <li>• Download cable</li> </ul>
JTAG-based configuration	Configuration via JTAG pins using: <ul style="list-style-type: none"> <li>• Download cable</li> <li>• Intelligent host (microprocessor)</li> <li>• Jam™ Standard Test and Programming Language (STAPL)</li> </ul>

You can select a Cyclone FPGA configuration scheme by driving its MSEL1 and MSEL0 pins either high (1) or low (0), as shown in [Table 13-2](#). If your application only requires a single configuration mode, the MSEL

pins can be connected to  $V_{CC}$  (the I/O bank's  $V_{CCIO}$  voltage where the MSEL pin resides) or to ground. If your application requires more than one configuration mode, the MSEL pins can be switched after the FPGA has been configured successfully. Toggling these pins during user mode will not affect the device operation. However, the MSEL pins must be valid before initiating reconfiguration.

**Table 13–2. Selecting Cyclone Configuration Schemes**

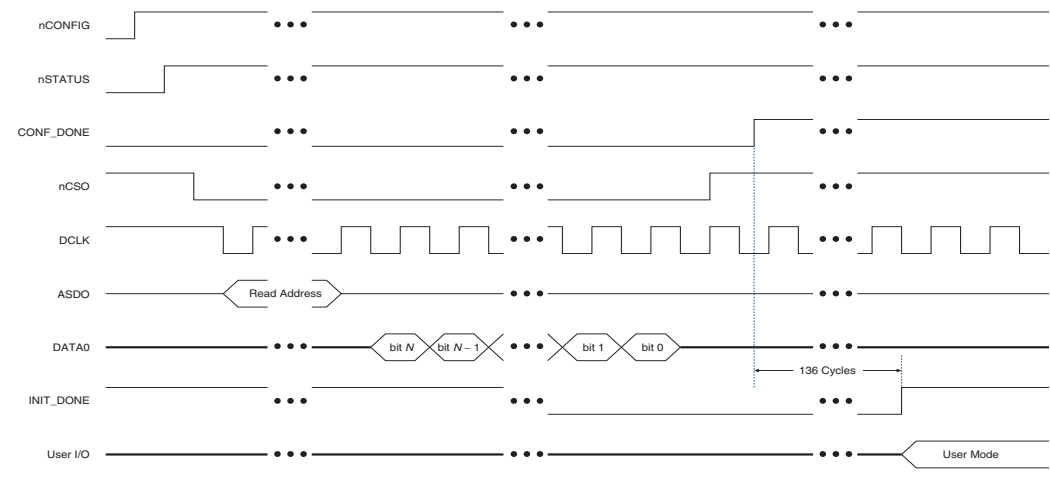
MSEL1	MSEL0	Configuration Scheme
0	0	AS
0	1	PS
0	0 or 1 (1)	JTAG-based (2)

**Notes to Table 13–2:**

- (1) Do not leave MSEL pins floating. Connect them to a low- or high-logic level. These pins support the non-JTAG configuration scheme used in production. If your design only uses JTAG configuration, you should connect the MSEL0 pin to  $V_{CC}$ .
- (2) JTAG-based configuration takes precedence over other schemes, which means that MSEL pin settings are ignored.

After configuration, Cyclone FPGAs will initialize registers and I/O pins, then enter user mode and function as per the user design. [Figure 13–1](#) shows an AS configuration waveform.

**Figure 13–1. AS Configuration Waveform**



You can configure Cyclone FPGAs using the 3.3-V, 2.5-V, 1.8-V, or 1.5-V LVTTTL I/O standard on configuration and JTAG input pins. These devices do not feature a VCCSEL pin; therefore, you should connect the VCCIO pins of the I/O banks containing configuration or JTAG pins according to the I/O standard specifications.

Table 13–3 summarizes the approximate uncompressed configuration file size for each Cyclone FPGA. To calculate the amount of storage space required for multi-device configurations, add the file size of each device together.

<b>Table 13–3. Cyclone Raw Binary File (.rbf) Sizes</b> <i>Note (1)</i>		
<b>Device</b>	<b>Data Size (Bits)</b>	<b>Data Size (Bytes)</b>
EP1C3	627,376	78,422
EP1C4	925,000	115,625
EP1C6	1,167,216	145,902
EP1C12	2,326,528	290,816
EP1C20	3,559,608	444,951

*Note to Table 13–3:*

(1) These values are preliminary.

You should only use the numbers in Table 13–3 to estimate the configuration file size before design compilation. Different file formats, such as .hex or .tff files, will have different file sizes. For any specific version of the Quartus® II software, any design targeted for the same device has the same uncompressed configuration file size. If compression is used, the file size can vary after each compilation.

## Data Compression

Cyclone FPGAs are the first FPGAs to support decompression of configuration data. This feature allows you to store compressed configuration data in configuration devices or other memory, and transmit this compressed bit stream to Cyclone FPGAs. During configuration, the Cyclone FPGA decompresses the bit stream in real time and programs its SRAM cells.

Cyclone FPGAs support compression in the AS and PS configuration schemes. Compression is not supported for JTAG-based configuration.



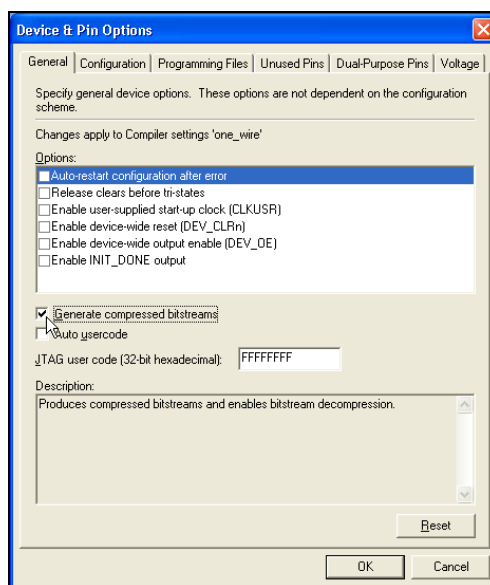
Preliminary data indicates that compression reduces configuration bit stream size by 35 to 60%.

When you enable compression, the Quartus II software generates configuration files with compressed configuration data. This compression reduces the storage requirements in the configuration device or flash, and decreases the time needed to transmit the bit stream to the Cyclone FPGA.

There are two methods to enable compression for Cyclone bitstreams: before design compilation (in the Compiler Settings menu) and after design compilation (in the Convert Programming Files window).

To enable compression in the project's compiler settings, select **Device** under the Assignments menu to bring up the settings window. After selecting your Cyclone device open the **Device & Pin Options** window, and in the **General** settings tab enable the check box for **Generate compressed bitstreams** (as shown in [Figure 13–2](#)).

**Figure 13–2. Enabling Compression for Cyclone Bitstreams in Compiler Settings**

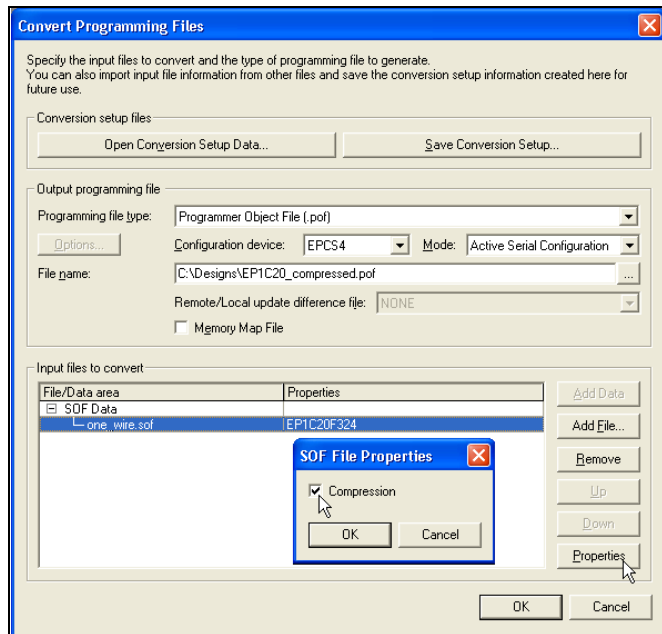


Compression can also be enabled when creating programming files from the **Convert Programming Files** window. See [Figure 13–3](#).

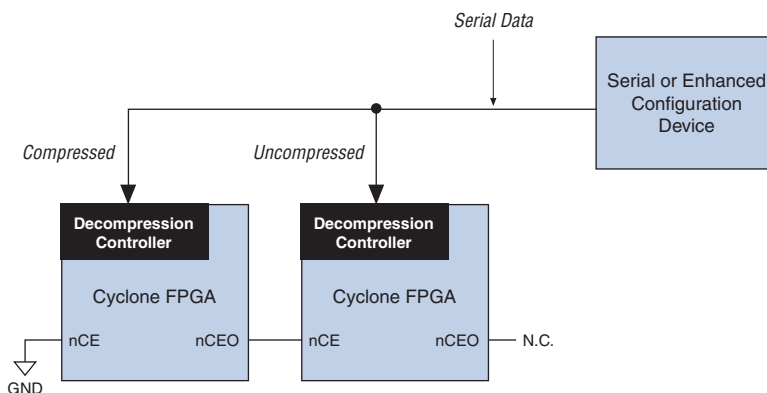
1. Click **Convert Programming Files** (File menu).

2. Select the Programming File type (POF, SRAM HEXOUT, RBF, or TTF).
3. For POF output files, select a configuration device.
4. Select **Add File** and add a Cyclone SOF file(s).
5. Select the name of the file you added to the **SOF Data** area and click on **Properties**.
6. Check the **Compression** checkbox.

**Figure 13–3. Enabling Compression for Cyclone Bitstreams in Convert Programming Files**



When multiple Cyclone devices are cascaded, the compression feature can be selectively enabled for each device in the chain. [Figure 13–4](#) depicts a chain of two Cyclone FPGAs. The first Cyclone FPGA has compression enabled and therefore receives a compressed bit stream from the configuration device. The second Cyclone FPGA has the compression feature disabled and receives uncompressed data.

**Figure 13–4. Compressed & Uncompressed Configuration Data in the Same Programming File** *Note (1)***Note to Figure 13–4:**

- (1) The first device in the chain should be set up in AS configuration mode ( $MSEL[1..0] = "00"$ ). The remaining devices in the chain must be set up in PS configuration mode ( $MSEL[1..0] = "01"$ ).

You can generate programming files for this setup from the **Convert Programming Files** window (File menu) in the Quartus II software.

The decompression feature supported by Cyclone FPGAs is separate from the decompression feature in enhanced configuration devices (EPC16, EPC8, and EPC4). The data compression feature in the enhanced configuration devices allows them to store compressed data and decompress the bit stream before transmitting to the target devices. When using Cyclone FPGAs with enhanced configuration devices, Altera recommends using compression on one of the devices, not both (preferably the Cyclone FPGA since transmitting compressed data reduces configuration time).

## Configuration Schemes

This section describes the various configuration schemes you can use to configure Cyclone FPGAs. Descriptions include an overview of the protocol, pin connections, and timing information. The schemes discussed are:

- AS configuration (serial configuration devices)
- PS configuration
- JTAG-based configuration



## Active Serial Configuration (Serial Configuration Devices)

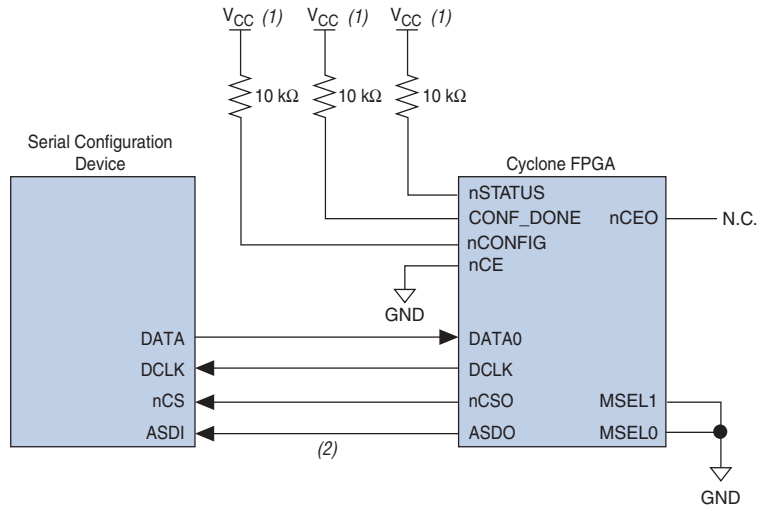
In the AS configuration scheme, Cyclone FPGAs are configured using the new serial configuration devices. These configuration devices are low cost devices with non-volatile memory that feature a simple four-pin interface and a small form factor. These features make serial configuration devices an ideal solution for configuring the low-cost Cyclone FPGAs.



For more information on serial configuration devices, see [Chapter 14, Serial Configuration Devices \(EPCS1 & EPCS4\) Data Sheet](#).

Serial configuration devices provide a serial interface to access configuration data. During device configuration, Cyclone FPGAs read configuration data via the serial interface, decompress data if necessary, and configure their SRAM cells. This scheme is referred to as an AS configuration scheme because the FPGA controls the configuration interface. This scheme is in contrast to the PS configuration scheme where the configuration device controls the interface.

Serial configuration devices have a four-pin interface: serial clock input (DCLK), serial data output (DATA), AS data input (ASDI), and an active-low chip select ( $\bar{n}CS$ ). This four-pin interface connects to Cyclone FPGA pins, as shown in [Figure 13–5](#).

**Figure 13–5. AS Configuration of a Single Cyclone FPGA****Notes to Figure 13–5:**

- (1) Connect the pull-up resistors to a 3.3-V supply.  
 (2) Cyclone FPGAs use the ASDO to ASDI path to control the configuration device.

Connecting the MSEL[1..0] pins to 00 selects the AS configuration scheme. The Cyclone chip enable signal, nCE, must also be connected to ground or driven low for successful configuration.

During system power up, both the Cyclone FPGA and serial configuration device enter a power-on reset (POR) period. As soon as the Cyclone FPGA enters POR, it drives nSTATUS low to indicate it is busy and drives CONF\_DONE low to indicate that it has not been configured. After POR, which typically lasts 100 ms, the Cyclone FPGA releases nSTATUS and enters configuration mode when this signal is pulled high by the external 10-kΩ resistor. Once the FPGA successfully exits POR, all user I/O pins are tri-stated. Cyclone devices have weak pull-up resistors on the user I/O pins which are on before and during configuration.



The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in [Chapter 4, DC & Switching Characteristics](#).

The serial clock (DCLK) generated by the Cyclone FPGA controls the entire configuration cycle (see [Figure 13–1 on page 13–2](#)) and this clock signal provides the timing for the serial interface. Cyclone FPGAs use an internal oscillator to generate DCLK. [Table 13–4](#) shows the active serial DCLK output frequencies.

Minimum	Typical	Maximum	Units
14	17	20	MHz

The serial configuration device latches input/control signals on the rising edge of DCLK and drives out configuration data on the falling edge. Cyclone FPGAs drive out control signals on the falling edge of DCLK and latch configuration data on the rising edge of DCLK.

In configuration mode, the Cyclone FPGA enables the serial configuration device by driving its `nCS0` output pin low that is connected to the chip select (`nCS`) pin of the configuration device. The Cyclone FPGA's serial clock (DCLK) and serial data output (ASDO) pins are used to read configuration data. The configuration device provides data on its serial data output (DATA) pin that is connected to the `DATA0` input on Cyclone FPGAs.

After all configuration bits are received by the Cyclone FPGA, it releases the open-drain `CONF_DONE` pin allowing the external 10-k $\Omega$  resistor to pull this signal to a high level. Initialization begins only after the `CONF_DONE` line reaches a high level.

You can select the clock used for initialization by using the **User Supplied Start-Up Clock** option in the Quartus II software. The Quartus II software uses the 10-MHz (typical) internal oscillator (separate from the AS internal oscillator) by default to initialize the Cyclone FPGA. When you enable the **User Supplied Start-Up Clock** option, the software uses the `CLKUSR` pin as the initialization clock. Supplying a clock on the `CLKUSR` pin will not affect the configuration process. After all configuration data is accepted and the `CONF_DONE` signal goes high, Cyclone devices require 136 clock cycles to initialize properly.

An optional `INIT_DONE` pin is available. This pin signals the end of initialization and the start of user mode with a low-to-high transition. The **Enable INIT\_DONE output** option is available in the Quartus II software. If the `INIT_DONE` pin is used, it will be high due to an external 10-K $\Omega$  pull-up resistor when `nCONFIG` is low and during the beginning of configuration. Once the option bit to enable `INIT_DONE` is programmed into the device (during the first frame of configuration

data), the `INIT_DONE` pin will go low. When initialization is complete, the `INIT_DONE` pin will be released and pulled high. This low-to-high transition signals that the FPGA has entered user mode. In user mode, the user I/O pins will no longer have weak pull-ups and will function as assigned in your design.

If an error occurs during configuration, the Cyclone FPGA asserts the `nSTATUS` signal low indicating a data frame error, and the `CONF_DONE` signal will stay low. With the **Auto-Restart Configuration on Frame Error** option enabled in the Quartus II software, the Cyclone FPGA resets the configuration device by pulsing `nCSO`, releases `nSTATUS` after a reset time-out period (about 30  $\mu$ s), and retries configuration. If this option is turned off, the system must monitor `nSTATUS` for errors and then pulse `nCONFIG` low for at least 40  $\mu$ s to restart configuration. After successful configuration, the `CONF_DONE` signal is tri-stated by the target device and then pulled high by the pull-up resistor.

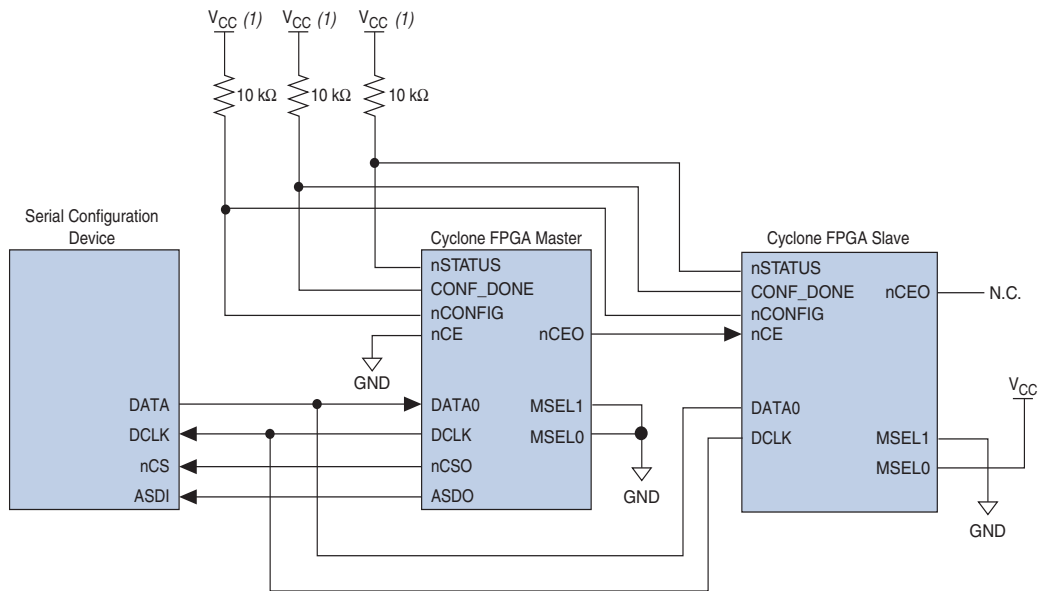
All AS configuration pins, `DATA0`, `DCLK`, `nCSO`, and `ASDO`, have weak internal pull-up resistors. These pull-up resistors are always active.

When the Cyclone FPGA is in user mode, you can initiate reconfiguration by pulling the `nCONFIG` pin low. The `nCONFIG` pin should be low for at least 40  $\mu$ s. When `nCONFIG` is pulled low, the FPGA also pulls `nSTATUS` and `CONF_DONE` low and all I/O pins are tri-stated. Once `nCONFIG` returns to a logic high level and `nSTATUS` is released by the Cyclone FPGA, reconfiguration begins.

### *Configuring Multiple Devices (Cascading)*

You can configure multiple Cyclone FPGAs using a single serial configuration device. You can cascade multiple Cyclone FPGAs using the chip-enable (`nCE`) and chip-enable-out (`nCEO`) pins. The first device in the chain must have its `nCE` pin connected to ground. You must connect its `nCEO` pin to the `nCE` pin of the next device in the chain. When the first device captures all of its configuration data from the bit stream, it drives the `nCEO` pin low enabling the next device in the chain. You must leave the `nCEO` pin of the last device unconnected. The `nCONFIG`, `nSTATUS`, `CONF_DONE`, `DCLK`, and `DATA0` pins of each device in the chain are connected (see [Figure 13–6](#)).

This first Cyclone FPGA in the chain is the configuration master and controls configuration of the entire chain. You must connect its `MSEL` pins to select the AS configuration scheme. The remaining Cyclone FPGAs are configuration slaves and you must connect their `MSEL` pins to select the PS configuration scheme. [Figure 13–6](#) shows the pin connections for this setup.

**Figure 13–6. Configuring Multiple Devices Using a Serial Configuration Device (AS)****Note to Figure 13–6:**

(1) Connect the pull-up resistors to a 3.3-V supply.

As shown in Figure 13–6, the nSTATUS and CONF\_DONE pins on all target FPGAs are connected together with external pull-up resistors. These pins are open-drain bidirectional pins on the FPGAs. When the first device asserts nCEO (after receiving all of its configuration data), it releases its CONF\_DONE pin. But the subsequent devices in the chain keep this shared CONF\_DONE line low until they have received their configuration data. When all target FPGAs in the chain have received their configuration data and have released CONF\_DONE, the pull-up resistor drives a high level on this line and all devices simultaneously enter initialization mode. If an error occurs at any point during configuration, the nSTATUS line is driven low by the failing FPGA. If you enable the **Auto Restart Configuration on Frame Error** option, reconfiguration of the entire chain begins after a reset time-out period (a maximum of 40 μs). If the option is turned off, see



While you can cascade Cyclone FPGAs, serial configuration devices cannot be cascaded or chained together.

If the configuration bit stream size exceeds the capacity of a serial configuration device, you must select a larger configuration device and/or enable the compression feature. While configuring multiple devices, the size of the bit stream is the sum of the individual devices' configuration bit streams.

### *Configuring Multiple Devices with the Same Data*

Certain applications require the configuration of multiple Cyclone devices with the same design through a configuration bit stream or SOF file. In active serial chains, this can be implemented by storing two copies of the SOF file in the serial configuration device. The first copy would configure the master Cyclone device, and the second copy would configure all remaining slave devices concurrently. The setup is similar to [Figure 13–6](#) where the master is setup in AS mode ( $MSEL=00$ ), and the slave devices are setup in PS mode ( $MSEL=01$ ).

To configure four identical Cyclone devices with the same SOF file, you could setup the chain similar to the example shown in [Figure 13–6](#), except connect the three slave devices for concurrent configuration. The  $nCEO$  pin from the master device drives the  $nCE$  input pins on all three slave devices, and the  $DATA$  and  $DCLK$  pins connect in parallel to all four devices. During the first configuration cycle, the master device reads its configuration data from the serial configuration device while holding  $nCEO$  high. After completing its configuration cycle, the master drives  $nCE$  low and transmits the second copy of the configuration data to all three slave devices, configuring them simultaneously.

### *Estimating Active Serial Configuration Time*

Active serial configuration time is dominated by the time it takes to transfer data from the serial configuration device to the Cyclone FPGA. This serial interface is clocked by the Cyclone  $DCLK$  output (generated from an internal oscillator). As listed in [Table 13–4](#), the  $DCLK$  minimum frequency is 14 MHz (71 ns). Therefore, the maximum configuration time estimate for an EP1C3 device (0.628 MBits of uncompressed data) is:

$$(0.628 \text{ MBits} \times 71 \text{ ns}) = 47 \text{ ms.}$$

The typical configuration time is 33 ms.

Enabling compression reduces the amount of configuration data that is transmitted to the Cyclone device, reducing configuration time. On average, compression reduces configuration time by 50%.

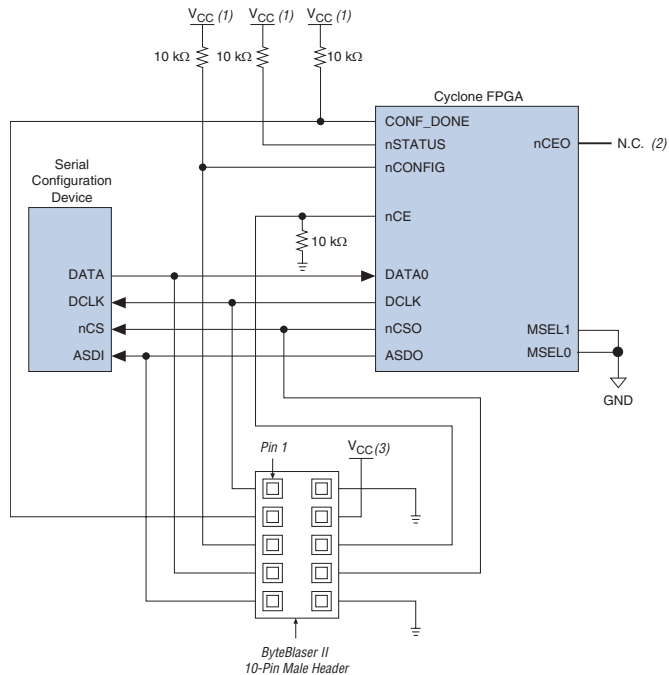
### *Programming Serial Configuration Devices*

Serial configuration devices are non-volatile, flash-memory-based devices. You can program these devices in-system using the ByteBlaster™ II download cable. Alternatively, you can program them using the Altera Programming Unit (APU) or supported third-party programmers.

You can perform in-system programming of serial configuration devices via the AS programming interface. During in-system programming, the download cable disables FPGA access to the AS interface by driving the nCE pin high. Cyclone FPGAs are also held in reset by a low level on nCONFIG. After programming is complete, the download cable releases nCE and nCONFIG, allowing the pull-down and pull-up resistor to drive GND and VCC, respectively. [Figure 13–7](#) shows the download cable connections to the serial configuration device.



For more information on the ByteBlaster II cable, see the *ByteBlaster II Download Cable Data Sheet*.

**Figure 13–7. In-System Programming of Serial Configuration Devices****Notes to Figure 13–7:**

- (1) Connect these pull-up resistors to 3.3-V supply.
- (2) The nCEO pin is left unconnected.
- (3) Power up the ByteBlaster II cable's  $V_{CC}$  with a 3.3-V supply.

You can program serial configuration devices by using the Quartus II software with the APU and the appropriate configuration device programming adapter. All serial configuration devices are offered in an eight-pin small outline integrated circuit (SOIC) package and can be programmed using the PLMSEPC-8 adapter.

In production environments, serial configuration devices can be programmed using multiple methods. Altera programming hardware (APU) or other third-party programming hardware can be used to program blank serial configuration devices before they are mounted onto PCBs. Alternatively, you can use an on-board microprocessor to program the serial configuration device in-system using C-based software drivers provided by Altera.



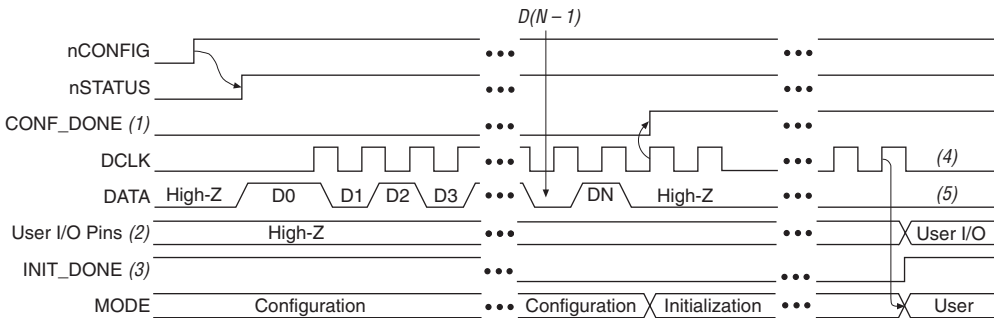


For more information on programming serial configuration devices, see the Cyclone Literature web page and the *Serial Configuration Devices (EPCS1 & EPCS4) Data Sheet*.

## Passive Serial Configuration

Cyclone FPGAs also feature the PS configuration scheme supported by all Altera FPGAs. In the PS scheme, an external host (configuration device, embedded processor, or host PC) controls configuration. Configuration data is clocked into the target Cyclone FPGAs via the DATA0 pin at each rising edge of DCLK. The configuration waveforms for this scheme are shown in [Figure 13-8](#).

**Figure 13-8. PS Configuration Cycle Waveform**



### Notes to [Figure 13-8](#):

- (1) During initial power up and configuration, CONF\_DONE is low. After configuration, CONF\_DONE goes high to indicate successful configuration. If the device is reconfigured, CONF\_DONE goes low after nCONFIG is driven low.
- (2) User I/O pins are tri-stated during configuration. Cyclone FPGAs also have a weak pull-up resistor on I/O pins during configuration. After initialization, the user I/O pins perform the function assigned in the user's design.
- (3) When used, the optional INIT\_DONE signal is high when nCONFIG is low before configuration and during the first 136 clock cycles of configuration.
- (4) In user mode, DCLK should be driven high or low when using the PS configuration scheme. When using the AS configuration scheme, DCLK is a Cyclone output pin and should not be driven externally.
- (5) In user mode, DATA0 should be driven high or low.

### PS Configuration using Configuration Device

In the PS configuration device scheme, nCONFIG is usually tied to V<sub>CC</sub> (when using EPC16, EPC8, EPC4, or EPC2 devices, you can connect nCONFIG to nINIT\_CONF). Upon device power-up, the target Cyclone FPGA senses the low-to-high transition on nCONFIG and initiates configuration. The target device then drives the open-drain CONF\_DONE pin low, which in-turn drives the configuration device's nCS pin low. When exiting POR, both the target and configuration device release the open-drain nSTATUS pin (typically Cyclone POR lasts 100 ms).

Before configuration begins, the configuration device goes through a POR delay of up to 100 ms (maximum) to allow the power supply to stabilize. You must power the Cyclone FPGA before or during the POR time of the enhanced configuration device. During POR, the configuration device drives its OE pin low. This low signal delays configuration because the OE pin is connected to the target device's nSTATUS pin. When the target and configuration devices complete POR, they both release the nSTATUS to OE line, which is then pulled high by a pull-up resistor.

When configuring multiple devices, configuration does not begin until all devices release their OE or nSTATUS pins. When all devices are ready, the configuration device clocks out DATA and DCLK to the target devices using an internal oscillator.

After successful configuration, the Cyclone FPGA starts initialization using the 10-MHz internal oscillator as the reference clock. The CONF\_DONE pin is released by the target device and then pulled high by a pull-up resistor. When initialization is complete, the target Cyclone FPGA enters user mode.

If an error occurs during configuration, the target device drives its nSTATUS pin low, resetting itself internally and resetting the configuration device. If you turn on the **Auto-Restart Configuration on Frame Error** option, the device reconfigures automatically if an error occurs. To set this option, select **Compiler Settings** (Processing menu), and click on the **Chips & Devices** tab. Select **Device & Pin Options**, and click on the **Configuration** tab.


If the **Auto-Restart Configuration on Frame Error** option is turned off, the external system (configuration device or microprocessor) must monitor nSTATUS for errors and then pulse nCONFIG low to restart configuration. The external system can pulse nCONFIG if it is under system control rather than tied to V<sub>CC</sub>. When configuration is complete, the target device releases CONF\_DONE, which disables the configuration device by driving nCS high. The configuration device drives DCLK low before and after configuration.

In addition, if the configuration device sends all of its data and then detects that CONF\_DONE has not gone high, it recognizes that the target device has not configured successfully. (For CONF\_DONE to reach a high state, enhanced configuration devices wait for 64 DCLK cycles after the last configuration bit. EPC2 devices wait for 16 DCLK cycles.) In this case, the configuration device pulses its OE pin low for a few microseconds, driving the target device's nSTATUS pin low. If the **Auto-Restart Configuration on Frame Error** option is set in the Quartus II software, the

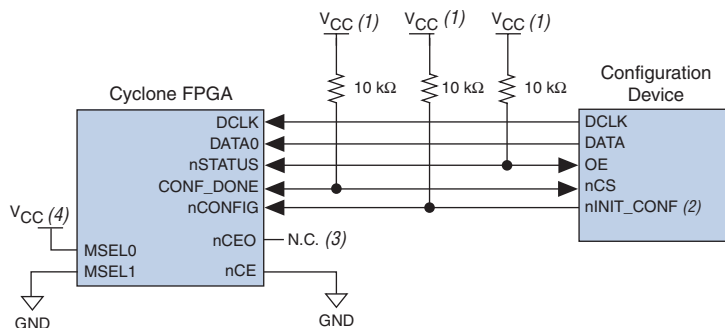
target device resets and then releases its `nSTATUS` pin after a reset time-out period. When `nSTATUS` returns high, the configuration device reconfigures the target device.

You should not pull `CONF_DONE` low to delay initialization. Instead, use the Quartus II software's **User-Supplied Start-Up Clock** option to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain initialize together since their `CONF_DONE` pins are tied together. For more information on this option, see ["Device Options"](#) on page 13–45.

`CONF_DONE` goes high during the first few clock cycles of initialization. Hence when using the `CLKUSR` feature you would not see the `CONF_DONE` signal high until you start clocking `CLKUSR`. However, the device does retain configuration data and waits for these initialization clocks to release `CONF_DONE` and go into user mode.

 When using internal pull-up resistors on configuration devices, power the supply voltage on the Cyclone FPGA I/O pins ( $V_{CCIO}$ ) to 3.3-V. EPC2, EPC4, EPC8, and EPC16 devices support 3.3-V operation but not 2.5-V operation. Therefore, you must set the  $V_{CCIO}$  voltages for the banks where programming pins reside to 3.3 V.

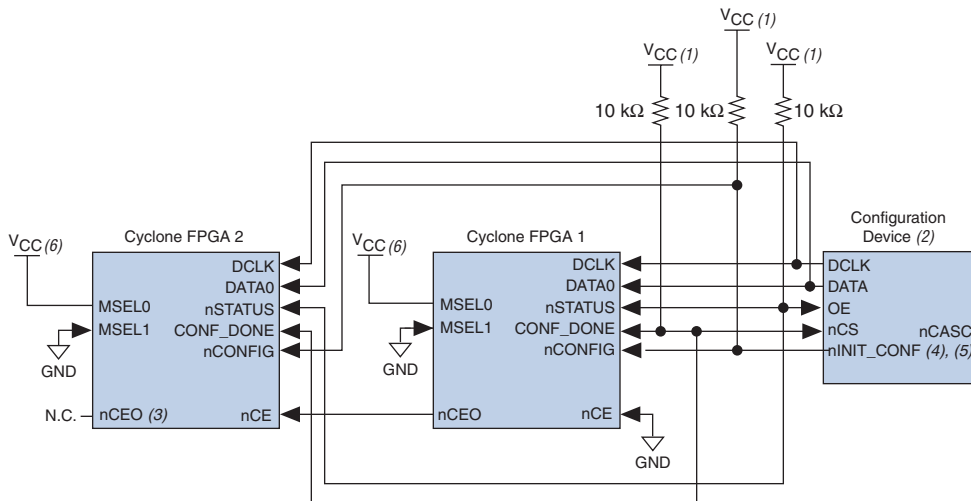
[Figure 13–9](#) shows how to configure one Cyclone FPGA with one configuration device.

**Figure 13–9. Single Device Configuration Circuit****Notes to Figure 13–9:**

- (1) The pull-up resistor should be connected to the same supply voltage as the configuration device. This pull-up resistor is 10 k $\Omega$ . The EPC16, EPC8, EPC4, and EPC2 devices' OE and nCS pins have internal, user-configurable pull-up resistors. If you use internal pull-up resistors, do not use external pull-up resistors on these pins.
- (2) The nINIT\_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices and has an internal pull-up resistor that is always active. If nINIT\_CONF is not used, nCONFIG must be pulled to V<sub>CC</sub> through a resistor.
- (3) The nCEO pin is left unconnected for the last device in the chain.
- (4) Connect MSEL0 to the V<sub>CC</sub> supply voltage of the I/O bank it resides in.

**Configuring Multiple Cyclone FPGAs**

You can use a single configuration device to configure multiple Cyclone FPGAs. In this setup, the nCEO pin of the first device is connected to the nCE pin of the second device in the chain. If there are additional devices, connect the nCE pin of the next device to the nCEO pin of the previous device. You should leave the nCEO pin on the last device in the chain unconnected. To configure properly, all of the target device CONF\_DONE and nSTATUS pins must be tied together. Figure 13–10 shows an example of configuring multiple Cyclone FPGAs using a single configuration device.

**Figure 13–10. Configuring Multiple Cyclone FPGAs with a Single Configuration Device****Notes to Figure 13–10:**

- (1) The pull-up resistor should be connected to the same supply voltage as the configuration device. The EPC16, EPC8, EPC4, and EPC2 devices' OE and nCS pins have internal, user-configurable pull-up resistors. If you use internal pull-up resistors, do not use external pull-up resistors on these pins.
- (2) EPC16, EPC8, and EPC4 configuration devices cannot be cascaded.
- (3) The nCEO pin is left unconnected for the last device in the chain.
- (4) The nINIT\_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices. If nINIT\_CONF is not used, nCONFIG must be pulled to  $V_{CC}$  through a resistor.
- (5) The nINIT\_CONF pin has an internal pull-up resistor that is always active in EPC16, EPC8, EPC4, and EPC2 devices. These devices do not need an external pull-up resistor on the nINIT\_CONF pin.
- (6) Connect MSEL0 to the  $V_{CC}$  supply voltage of the I/O bank it resides in.

When performing multi-device PS configuration, you must generate the configuration device programming file (.sof) from each project. Then you must combine multiple .sof files using the Quartus II software through the **Convert Programming Files** dialog box.



For more information on how to create Programmer Object Files (.pof) for enhanced configuration devices, see *AN 218: Using Enhanced Configuration Devices*. For a description of the various configuration and programming files, see “[Device Configuration Files](#)” on page 13–53.

After the first Cyclone FPGA completes configuration during multi-device configuration, its nCEO pin activates the second device's nCE pin, prompting the second device to begin configuration. Because all device CONF\_DONE pins are tied together, all devices initialize and enter user mode at the same time.

In addition, all `nSTATUS` pins are tied together; therefore, if any device (including the configuration device) detects an error, configuration stops for the entire chain. Also, if the configuration device does not detect `CONF_DONE` going high at the end of configuration, it resets the chain by pulsing its `OE` pin low for a few microseconds. For `CONF_DONE` to reach a high state, enhanced configuration devices wait for 64 `DCLK` cycles after the last configuration bit. EPC2 devices wait for 16 `DCLK` cycles.

If the **Auto-Restart Configuration on Frame Error** option is turned on in the Quartus II software, the Cyclone FPGA releases its `nSTATUS` pins after a reset time-out period (about 30 micro-seconds). When the `nSTATUS` pins are released and pulled high, the configuration device reconfigures the chain. If the **Auto-Restart Configuration on Frame Error** option is not turned on, the devices drive `nSTATUS` low until they are reset with a low pulse on `nCONFIG`.

You can also cascade several EPC2 configuration devices to configure multiple Cyclone FPGAs. When all data from the first configuration device is sent, it drives `nCASC` low, which in turn drives `nCS` on the subsequent EPC2 device. Because a configuration device requires less than one clock cycle to activate a subsequent configuration device, the data stream is uninterrupted. You cannot cascade EPC16, EPC8, and EPC4 configuration devices.

### *Programming Configuration Devices*

Enhanced configuration devices (EPC4, EPC8, and EPC16 devices) and EPC2 devices support in-system programming via JTAG. You can program these configuration devices using the Quartus II software and a download cable (e.g., ByteBlaster II, MasterBlaster™, or ByteBlasterMV™ cables).

You can also program configuration devices using the Quartus II software, the APU, and the appropriate configuration device programming adapter. [Table 13–5](#) shows which programming adapter to use with each configuration device.

<i>Table 13–5. Programming Adapters (Part 1 of 2)</i>		
Device	Package	Adapter
EPC16	88-pin Ultra FineLine BGA® 100-pin PQFP	PLMU EPC-88 PLMQEPC-100
EPC8	100-pin PQFP	PLMQEPC-100
EPC4	100-pin PQFP	PLMQEPC-100

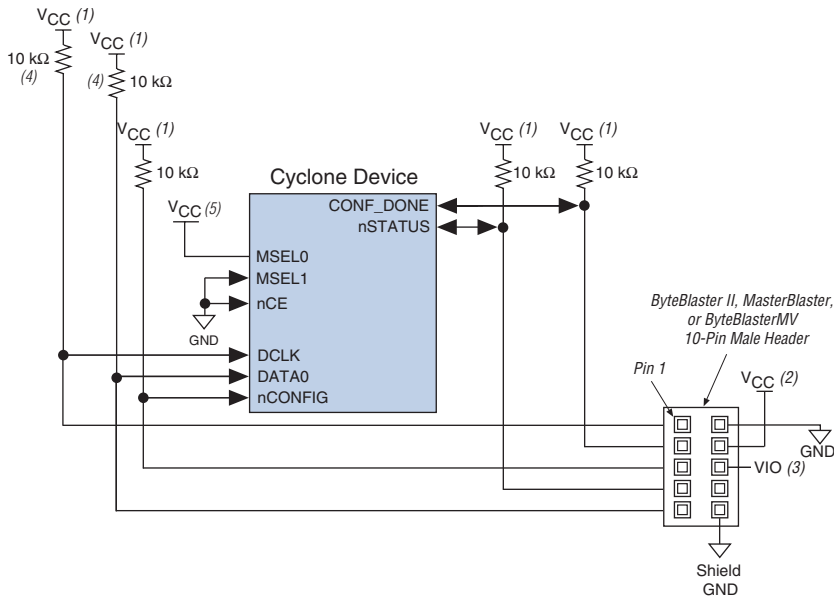
**Table 13–5. Programming Adapters (Part 2 of 2)**

Device	Package	Adapter
EPC2	20-pin J-Lead 32-pin TQFP	PLMJ1213 PLMT1213
EPC1	8-pin DIP 20-pin J-Lead	PLMJ1213 PLMJ1213

*PS Configuration Using a Download Cable*

Using a download cable in PS configuration, an intelligent host (e.g., your PC) transfers data from a storage device (e.g., your hard drive) to the Cyclone FPGA through a ByteBlaster II, MasterBlaster, or ByteBlasterMV cable. To initiate configuration in this scheme, the download cable generates a low-to-high transition on the nCONFIG pin. The programming hardware then sends the configuration data one bit at a time on the device's DATA0 pin. The data is clocked into the target device using DCLK until the CONF\_DONE goes high.

When using programming hardware for the Cyclone FPGA, turning on the **Auto-Restart Configuration on Frame Error** option does not affect the configuration cycle because the Quartus II software must restart configuration when an error occurs. [Figure 13–11](#) shows the PS configuration setup for the Cyclone FPGA using a ByteBlaster II, MasterBlaster, or ByteBlasterMV cable.

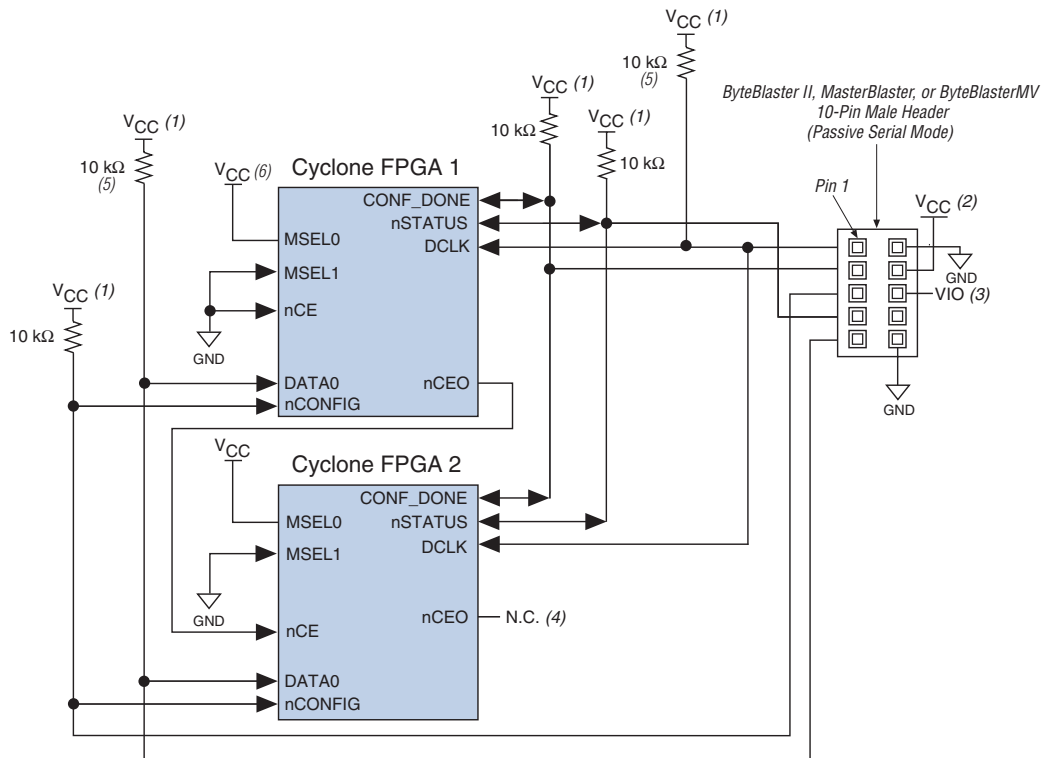
**Figure 13–11. PS Configuration Circuit with ByteBlaster II, MasterBlaster, or ByteBlasterMV Cable****Notes to Figure 13–11:**

- (1) You should connect the pull-up resistor to the same supply voltage as the MasterBlaster (V<sub>IO</sub> pin) or ByteBlasterMV cable.
- (2) Power supply voltage: V<sub>CC</sub> = 3.3-V for the ByteBlaster II, MasterBlaster, and ByteBlasterMV cable.
- (3) Pin 6 of the header is a V<sub>IO</sub> reference voltage for the MasterBlaster output driver. V<sub>IO</sub> should match the device's V<sub>CCIO</sub>. This pin is a no-connect pin for the ByteBlasterMV header.
- (4) The pull-up resistors on DATA0 and DCLK are only needed if the download cable is the only configuration scheme used on your board. This is to ensure that DATA0 and DCLK are not left floating after configuration. For example, if you are also using a configuration device, the pull-up resistors on DATA0 and DCLK are not needed.
- (5) Connect MSEL0 to the V<sub>CC</sub> supply voltage of the I/O bank it resides in.

You can use the download cable to configure multiple Cyclone FPGAs by connecting each device's nCEO pin to the subsequent device's nCE pin. All other configuration pins are connected to each device in the chain.

Because all CONF\_DONE pins are tied together, all devices in the chain initialize and enter user mode at the same time. In addition, because the nSTATUS pins are tied together, the entire chain halts configuration if any device detects an error. In this situation, the Quartus II software must restart configuration; the **Auto-Restart Configuration on Frame Error** option does not affect the configuration cycle. Figure 13–12 shows how to configure multiple Cyclone FPGAs with a ByteBlaster II, MasterBlaster, or ByteBlasterMV cable.



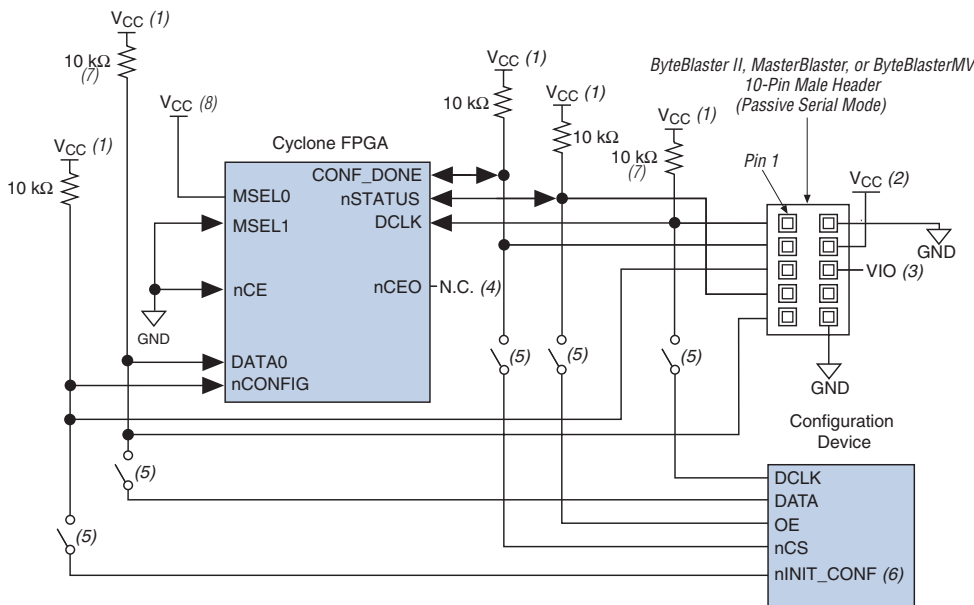
**Figure 13–12. Multi-Device PS Configuration with a ByteBlaster II, MasterBlaster, or ByteBlasterMV Cable****Notes to Figure 13–12:**

- (1) You should connect the pull-up resistor to the same supply voltage as the MasterBlaster (V<sub>IO</sub> pin) or ByteBlasterMV cable.
- (2) Power supply voltage: V<sub>CC</sub> = 3.3-V for the ByteBlaster II, MasterBlaster, and ByteBlasterMV cable.
- (3) V<sub>IO</sub> is a reference voltage for the MasterBlaster output driver. V<sub>IO</sub> should match the device's V<sub>CCIO</sub>. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.
- (4) The nCEO pin is left unconnected for the last device in the chain.
- (5) The pull-up resistors on DATA0 and DCLK are only needed if the download cable is the only configuration scheme used on your board. This is to ensure that DATA0 and DCLK are not left floating after configuration. For example, if you are also using a configuration device, the pull-up resistors on DATA0 and DCLK are not needed.
- (6) Connect MSEL0 to the V<sub>CC</sub> supply voltage of the I/O bank it resides in.

If you are using a ByteBlaster II, MasterBlaster, or ByteBlasterMV cable to configure device(s) on a board that also is populated with configuration devices, you should electrically isolate the configuration devices from the target device(s) and cable. One way to isolate the configuration devices is to add logic, such as a multiplexer, that can select between the configuration devices and the cable. The multiplexer allows bidirectional transfers on the nSTATUS and CONF\_DONE signals. Another option is to

add switches to the five common signals (CONF\_DONE, nSTATUS, DCLK, nCONFIG, and DATA0) between the cable and the configuration devices. The last option is to remove the configuration devices from the board when configuring with the cable. Figure 13–13 shows a combination of a configuration device and a ByteBlaster II, MasterBlaster, or ByteBlasterMV cable to configure a Cyclone FPGA.

**Figure 13–13. Configuring with a Combined PS & Configuration Device Scheme**



**Notes to Figure 13–13:**

- (1) You should connect the pull-up resistor to the same supply voltage as the configuration device.
- (2) Power supply voltage:  $V_{CC} = 3.3\text{-V}$  for the ByteBlaster II, MasterBlaster, and ByteBlasterMV cable.
- (3) Pin 6 of the header is a  $V_{IO}$  reference voltage for the MasterBlaster output driver.  $V_{IO}$  should match the target device's  $V_{CCI/O}$ . This is a no-connect pin for the ByteBlasterMV header.
- (4) The nCEO pin is left unconnected.
- (5) You should not attempt configuration with a ByteBlaster II, MasterBlaster, or ByteBlasterMV cable while a configuration device is connected to a Cyclone FPGA. Instead, you should either remove the configuration device from its socket when using the download cable or place a switch on the five common signals between the download cable and the configuration device. Remove the ByteBlaster II, MasterBlaster, or ByteBlasterMV cable when configuring with a configuration device.
- (6) If nINIT\_CONF is not used, nCONFIG must be pulled to  $V_{CC}$  either directly or through a resistor.
- (7) The pull-up resistors on DATA0 and DCLK are only needed if the download cable is the only configuration scheme used on your board. This is to ensure that DATA0 and DCLK are not left floating after configuration. For example, if you are also using a configuration device, the pull-up resistors on DATA0 and DCLK are not needed.
- (8) Connect MSEL0 to the  $V_{CC}$  supply voltage of the I/O bank it resides in.



For more information on how to use the ByteBlaster II, MasterBlaster, or ByteBlasterMV cables, see the following documents:

- *ByteBlaster II Parallel Port Download Cable Data Sheet*
- *MasterBlaster Serial/USB Communications Cable Data Sheet*
- *ByteBlasterMV Parallel Port Download Cable Data Sheet*

### *PS Configuration from a Microprocessor*

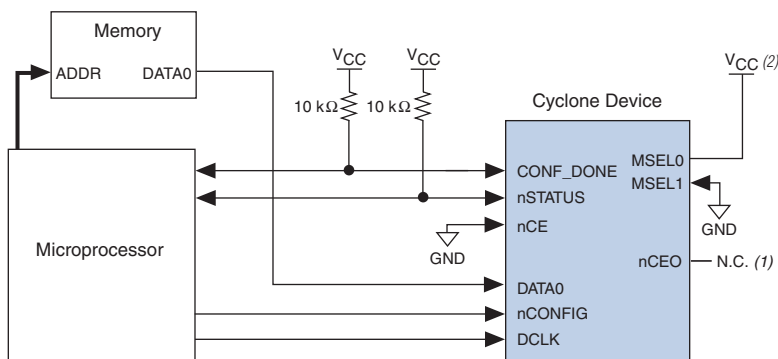
In PS configuration with a microprocessor, a microprocessor transfers data from a storage device to the target Cyclone FPGA. To initiate configuration in this scheme, the microprocessor must generate a low-to-high transition on the `nCONFIG` pin and the target device must release `nSTATUS`. The microprocessor then places the configuration data one bit at a time on the `DATA0` pin of the Cyclone FPGA. The least significant bit (LSB) of each data byte must be presented first. Data is clocked continuously into the target device using `DCLK` until the `CONF_DONE` signal goes high.

The Cyclone FPGA starts initialization using the internal oscillator after all configuration data is transferred. The device's `CONF_DONE` pin goes high to show successful configuration and the start of initialization. Driving `DCLK` to the device after configuration does not affect device operation.

Since the PS configuration scheme is a synchronous scheme, the configuration clock speed must be below the specified maximum frequency to ensure successful configuration. Maximum `DCLK` frequency supported by Cyclone FPGAs is 100 MHz (See [Table 13–6 on page 13–27](#)). No maximum `DCLK` period (i.e., minimum `DCLK` frequency) exists. You can pause configuration by halting `DCLK` for an indefinite amount of time.

If the target device detects an error during configuration, it drives its `nSTATUS` pin low to alert the microprocessor. The microprocessor can then pulse `nCONFIG` low to restart the configuration process. Alternatively, if the **Auto-Restart Configuration on Frame Error** option is turned on in the Quartus II software, the target device releases `nSTATUS` after a reset time-out period. After `nSTATUS` is released, the microprocessor can reconfigure the target device without needing to pulse `nCONFIG` low.

The microprocessor can also monitor the `CONF_DONE` and `INIT_DONE` pins to ensure successful configuration and initialization. If the microprocessor sends all data but `CONF_DONE` and `INIT_DONE` has not gone high, it must reconfigure the target device. [Figure 13–14](#) shows the circuit for PS configuration with a microprocessor.

**Figure 13–14. PS Configuration Circuit with a Microprocessor****Notes to Figure 13–14:**

- (1) The nCEO pin is left unconnected.
- (2) Connect MSEL0 to the V<sub>CC</sub> supply voltage of the I/O bank it resides in.

**Configuring Cyclone FPGAs with the MicroBlaster Software**

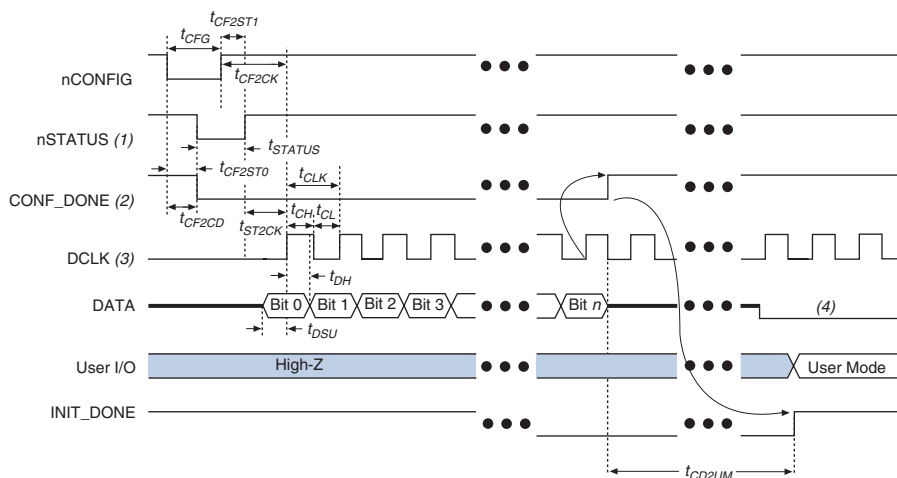
The MicroBlaster™ software driver allows you to configure Altera FPGAs, including Cyclone FPGAs, through the ByteBlaster II or ByteBlasterMV cable in PS mode. The MicroBlaster software driver supports a Raw Binary File (.rbf) programming input file and is targeted for embedded PS configuration. The source code is developed for the Windows NT operating system, although you can customize it to run on other operating systems. For more information on the MicroBlaster software driver, see the *Configuring the MicroBlaster Passive Serial Software Driver White Paper* and source files on the Altera web site at [www.altera.com](http://www.altera.com).

**Passive Serial Timing**

For successful configuration using the PS scheme, several timing parameters such as setup, hold, and maximum clock frequency must be satisfied. The enhanced configuration and EPC2 devices are designed to meet these interface timing specifications. If you use a microprocessor or another intelligent host to control the PS interface, ensure that you meet these timing requirements.

Figure 13–15 shows the PS timing waveform for Cyclone FPGAs.

Figure 13–15. PS Timing Waveform for Cyclone FPGAs



Notes to Figure 13–15:

- (1) Upon power-up, the Cyclone FPGA holds nSTATUS low for about 100 ms.
- (2) Upon power-up and before configuration, CONF\_DONE is low.
- (3) In user mode, DCLK should be driven high or low when using the PS configuration scheme. When using the AS configuration scheme, DCLK is a Cyclone output pin and should not be driven externally.
- (4) DATA should not be left floating after configuration. It should be driven high or low, whichever is more convenient.

Table 13–6 contains the PS timing information for Cyclone FPGAs.

Symbol	Parameter	Min	Max	Units
$t_{CF2CD}$	nCONFIG low to CONF_DONE low		800	ns
$t_{CF2ST0}$	nCONFIG low to nSTATUS low		800	ns
$t_{CF2ST1}$	nCONFIG high to nSTATUS high		40 (4)	$\mu$ s
$t_{CFG}$	nCONFIG low pulse width (2)	40		$\mu$ s
$t_{STATUS}$	nSTATUS low pulse width	10	40 (4)	$\mu$ s
$t_{CF2CK}$	nCONFIG high to first rising edge on DCLK	40		$\mu$ s
$t_{ST2CK}$	nSTATUS high to first rising edge on DCLK	1		$\mu$ s
$t_{DSU}$	Data setup time before rising edge on DCLK	7		ns
$t_{DH}$	Data hold time after rising edge on DCLK	0		ns

**Table 13–6. PS Timing Parameters for Cyclone Devices** *Note (1) (Part 2 of 2)*

Symbol	Parameter	Min	Max	Units
$t_{CH}$	DCLK high time	4		ns
$t_{CL}$	DCLK low time	4		ns
$t_{CLK}$	DCLK period	10		ns
$f_{MAX}$	DCLK maximum frequency		100	MHz
$t_{CD2UM}$	CONF_DONE high to user mode (3)	6	20	$\mu$ s

**Notes to Table 13–6:**

- (1) This information is preliminary.
- (2) This value applies only if the internal oscillator is selected as the clock source for device initialization. If the clock source is CLKUSR, multiply the clock period by 270 to obtain this value. CLKUSR must be running during this period to reset the device.
- (3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for device initialization. If the clock source is CLKUSR, multiply the clock period by 140 to obtain this value.
- (4) You can obtain this value if you do not delay configuration by extending the nSTATUS low-pulse width.

## JTAG-Based Configuration

JTAG has developed a specification for boundary-scan testing. This boundary-scan test (BST) architecture offers the capability to efficiently test components on printed circuit boards (PCBs) with tight lead spacing. The BST architecture can test pin connections without using physical test probes and capture functional data while a device is operating normally. You can also use the JTAG circuitry to shift configuration data into Cyclone FPGAs. The Quartus II software automatically generates .sof files that can be used for JTAG configuration.



For more information on JTAG boundary-scan testing, see *AN 39: IEEE 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*.

A device operating in JTAG mode uses four required pins, TDI, TDO, TMS, and TCK. Cyclone FPGAs do not support the optional TRST pin. The three JTAG input pins, TCK, TDI, and TMS, have weak internal pull-up resistors. All user I/O pins are tri-stated during JTAG configuration.

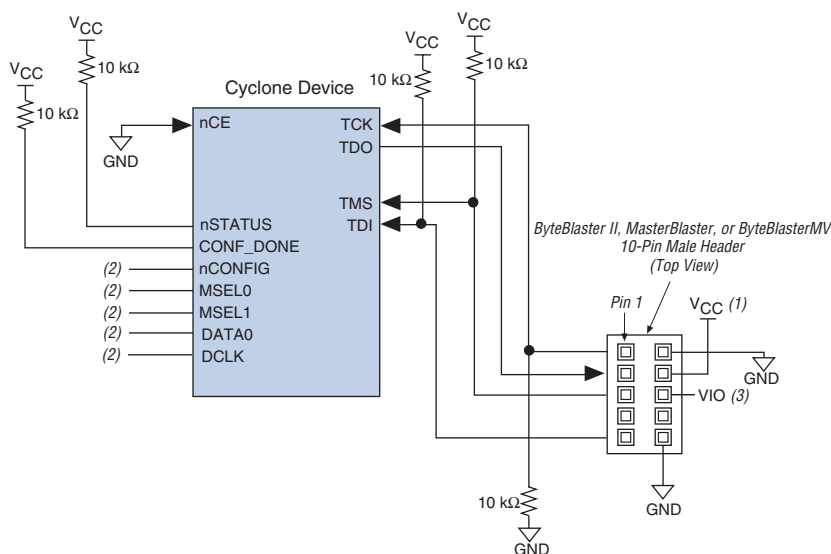
Cyclone is designed such that JTAG instructions have precedence over any device operating modes. So JTAG configuration can take place without waiting for other configuration to complete (e.g., configuration with serial or enhanced configuration devices). If you attempt JTAG configuration in Cyclone FPGAs during non-JTAG configuration, non-JTAG configuration will be terminated and JTAG configuration will be initiated.

Table 13–7 shows each JTAG pin’s function.

<b>Pin</b>	<b>Description</b>	<b>Function</b>
TDI	Test data input	Serial input pin for instructions as well as test and programming data. Data is shifted in on the rising edge of TCK.
TDO	Test data output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of TCK. The pin is tri-stated if data is not being shifted out of the device.
TMS	Test mode select	Input pin that provides the control signal to determine the transitions of the Test Access Port (TAP) controller state machine. Transitions within the state machine occur on the rising edge of TCK. Therefore, TMS must be set up before the rising edge of TCK. TMS is evaluated on the rising edge of TCK.
TCK	Test clock input	The clock input to the BST circuitry. Some operations occur at the rising edge, while others occur at the falling edge.

### *JTAG Configuration Using a Download Cable*

During JTAG configuration, data is downloaded to the device on the board through a ByteBlaster II, ByteBlasterMV, or MasterBlaster download cable. Configuring devices through a cable is similar to programming devices in-system. See Figure 13–16 for pin connection information.

**Figure 13–16. JTAG Configuration of Single Cyclone FPGA****Notes to Figure 13–16:**

- (1) You should connect the pull-up resistor to the same supply voltage as the download cable.
- (2) You should connect the nCONFIG, MSEL0, and MSEL1 pins to support a non-JTAG configuration scheme. If you only use JTAG configuration, connect nCONFIG to  $V_{CC}$ , and MSEL0 and MSEL1 to ground. Pull DATA0 and DCLK to high or low.
- (3)  $V_{IO}$  is a reference voltage for the MasterBlaster output driver.  $V_{IO}$  should match the device's  $V_{CCIO}$ . Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

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 upon completion. The software checks the state of CONF\_DONE through the JTAG port. 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 bit stream is transmitted serially via the JTAG TDI port, the TCK port is clocked an additional 134 cycles to perform device initialization.



If  $V_{CCIO}$  is tied to 3.3-V, both the I/O pins and the JTAG TDO port drive at 3.3-V levels.



Cyclone FPGAs have dedicated JTAG pins. Not only can you perform JTAG testing on Cyclone FPGAs before and after, but also during configuration. While other device families do not support JTAG testing during configuration, Cyclone FPGAs support the `BYPASS`, `IDCODE`, and `SAMPLE` instructions during configuration without interrupting configuration. All other JTAG instructions may only be issued by first interrupting configuration and reprogramming I/O pins using the `CONFIG_IO` instruction.

The `CONFIG_IO` instruction allows I/O buffers to be configured via the JTAG port, and when issued, interrupts configuration. This instruction allows you to perform board-level testing prior to configuring the Cyclone FPGA or waiting for a configuration device to complete configuration. Once configuration has been interrupted and JTAG testing is complete, the part must be reconfigured via JTAG (`PULSE_CONFIG` instruction) or by pulsing `nCONFIG` low.

The chip-wide reset and output enable pins on Cyclone FPGAs do not affect JTAG boundary-scan or programming operations. Toggling these pins does not affect JTAG operations (other than the usual boundary-scan operation).

When designing a board for JTAG configuration of Cyclone FPGAs, you should consider the regular configuration pins. [Table 13–8](#) shows how you should connect these pins during JTAG configuration.

**Table 13–8. JTAG Termination of Unused Pins (Part 1 of 2)**

Signal	Description
<code>nCE</code>	Drive all Cyclone devices in the chain low by connecting <code>nCE</code> to ground, pulling it down via a resistor, or driving it low by some control circuitry. For devices in a multi-device PS and AS configuration chains, connect the <code>nCE</code> pins to ground during JTAG configuration or configure them via JTAG in the same order as the configuration chain.
<code>nCEO</code>	For all Cyclone devices in a chain, the <code>nCEO</code> pin can be left floating or connected to the <code>nCE</code> pin of the previous device.
<code>nSTATUS</code>	Pulled to $V_{CC}$ through a 10-k $\Omega$ resistor. When configuring multiple devices in the same JTAG chain, pull up each <code>nSTATUS</code> pin to $V_{CC}$ individually. (1)
<code>CONF_DONE</code>	Pulled to $V_{CC}$ through a 10-k $\Omega$ resistor. When configuring multiple devices in the same JTAG chain, pull up each <code>CONF_DONE</code> pin to $V_{CC}$ individually. (1)
<code>nCONFIG</code>	Driven high by connecting to $V_{CC}$ , pulling up through a resistor, or driving it high by some control circuitry.

**Table 13–8. JTAG Termination of Unused Pins (Part 2 of 2)**

Signal	Description
MSEL0 , MSEL1	Do not leave these pins floating. These pins support whichever non-JTAG configuration is used in production. If only JTAG configuration is used, you should tie these pins to ground.
DCLK	Do not leave these pins floating. Drive low or high, whichever is more convenient.
DATA0	Do not leave these pins floating. Drive low or high, whichever is more convenient.

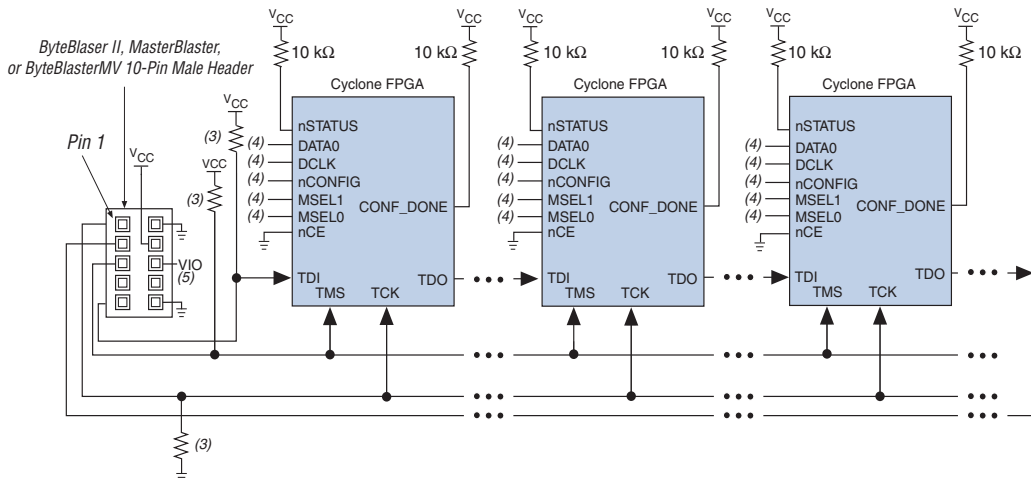
**Note to Table 13–8:**

- (1) nSTATUS going low in the middle of JTAG configuration indicates that an error has occurred; CONF\_DONE going high at the end of JTAG configuration indicates successful configuration.

*JTAG Configuration of Multiple Devices*

When programming a JTAG device chain, one JTAG-compatible header, such as the ByteBlaster II header, is connected to several devices. The number of devices in the JTAG chain is limited only by the drive capacity of the download cable. However, when four or more devices are connected in a JTAG chain, Altera recommends buffering the TCK, TDI, and TMS pins with an on-board buffer.

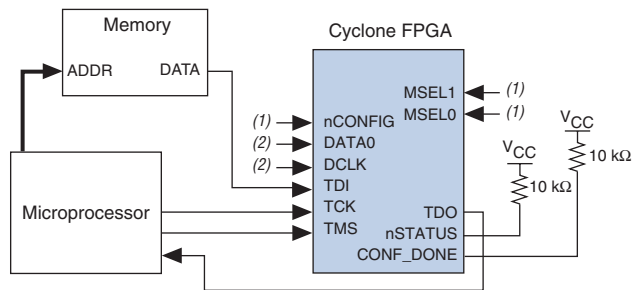
JTAG-chain device configuration is ideal when the system contains multiple devices, or when testing your system using JTAG BST circuitry. [Figure 13–17](#) shows multi-device JTAG configuration.

**Figure 13–17. Multi-Device JTAG Configuration****Notes to Figure 13–17:**

- (1) Cyclone, APEX™ II, APEX 20K, Mercury™, ACEX® 1K, and FLEX® 10K devices can be placed within the same JTAG chain for device programming and configuration.
- (2) For more information on all configuration pins connected in this mode, refer to [Table 13–7 on page 13–29](#).
- (3) These pull-up/pull-down resistors are 10 kΩ.
- (4) Connect the nCONFIG, MSEL0, and MSEL1 pins to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect nCONFIG to V<sub>CC</sub>, and MSEL0 and MSEL1 to ground. Pull DATA0 and DCLK to either high or low.
- (5) V<sub>IO</sub> is a reference voltage for the MasterBlaster output driver. V<sub>IO</sub> should match the device's V<sub>CCIO</sub>. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

Connect the nCE pin to ground or drive it low during JTAG configuration. In multi-device PS and AS configuration chains, connect the first device's nCE pin to ground and connect the nCEO pin to the nCE pin of the next device in the chain. The last device's nCE input comes from the previous device, while its nCEO pin is left floating. After the first device completes configuration in a multi-device configuration chain, it's nCEO pin drives low to activate the second device's nCE pin, which prompts the second device to begin configuration. Therefore, if these devices are also in a JTAG chain, you should make sure the nCE pins are connected to ground during JTAG configuration or that the devices are configured via JTAG in the same order as the configuration chain. As long as the devices are configured in the same order as the multi-device configuration chain, the nCEO pin of the previous device will drive the nCE pin of the next device low when it has successfully been configured.

Figure 13–18 shows the JTAG configuration of a Cyclone FPGA with a microprocessor.

**Figure 13–18. JTAG Configuration of Cyclone FPGAs with a Microprocessor****Notes to Figure 13–18:**

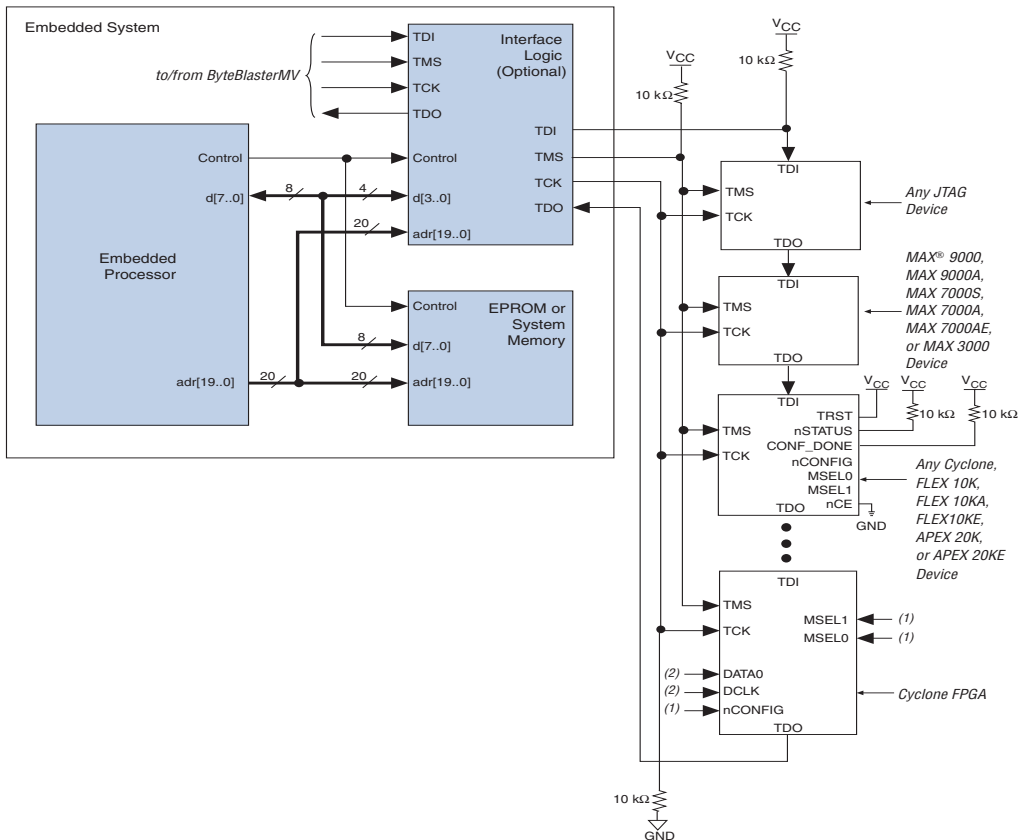
- (1) Connect the nCONFIG, MSEL1, and MSEL0 pins to support a non-JTAG configuration scheme. If your design only uses JTAG configuration, connect the nCONFIG pin to  $V_{CC}$  and the MSEL1 and MSEL0 pins to ground.
- (2) Pull DATA0 and DCLK to either high or low.

**Connecting the JTAG Chain to the Embedded Processor**

There are two ways to connect the JTAG chain to the embedded processor. The most straightforward method is to connect the embedded processor directly to the JTAG chain. In this method, four of the processor pins are dedicated to the JTAG interface, saving board space but reducing the number of available embedded processor pins.

Figure 13–19 illustrates the second method, which is to connect the JTAG chain to an existing bus through an interface programmable logic device (PLD). In this method, the JTAG chain becomes an address on the existing bus. The processor then reads from or writes to the address representing the JTAG chain.

Figure 13–19. Embedded System Block Diagram



Notes to Figure 13–19:

- (1) Connect the nCONFIG, MSEL1, and MSEL0 pins to support a non-JTAG configuration scheme. If your design only uses JTAG configuration, connect the nCONFIG pin to V<sub>CC</sub> and the MSEL1 and MSEL0 pins to ground.
- (2) Pull DATA0 and DCLK to either high or low.

Configuring Cyclone FPGAs with JRunner

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

other platforms. For more information on the JRunner software driver, see *JRunner Software Driver: An Embedded Solution to the JTAG Configuration* and the source files on the Altera web site.

### *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.

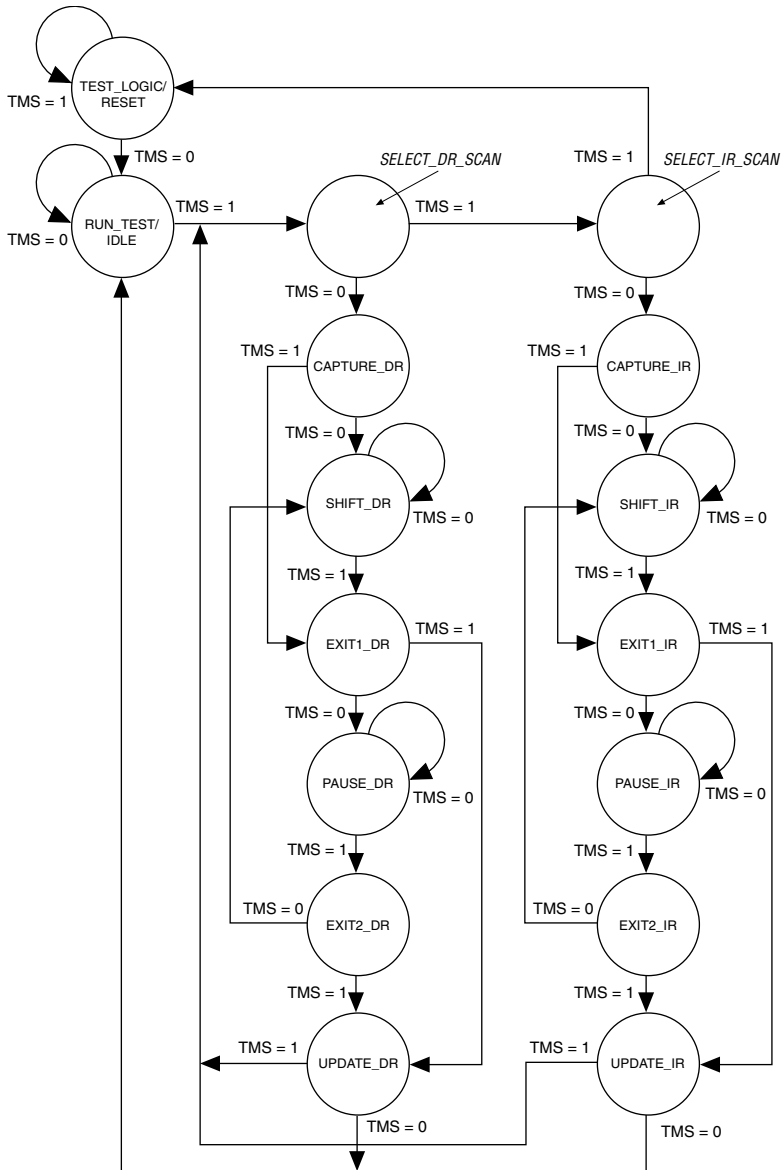


Both JTAG connection methods should include space for the MasterBlaster or ByteBlasterMV header connection. The header is useful during prototyping because it allows you to verify or modify the Cyclone FPGA's contents. During production, you can remove the header to save cost.

### **Program Flow**

The Jam Player provides an interface for manipulating the IEEE Std. 1149.1 JTAG TAP state machine. The TAP controller is a 16-state, state machine that is clocked on the rising edge of TCK, and uses the TMS pin to control JTAG operation in a device. [Figure 13–20](#) shows the flow of an IEEE Std. 1149.1 TAP controller state machine.

Figure 13–20. JTAG TAP Controller State Machine



While the Jam Player provides a driver that manipulates the TAP controller, the Jam Byte-Code File (**.jbc**) provides the high-level intelligence needed to program a given device. All Jam instructions that

send JTAG data to the device involve moving the TAP controller through either the data register leg or the instruction register leg of the state machine. For example, loading a JTAG instruction involves moving the TAP controller to the `SHIFT_IR` state and shifting the instruction into the instruction register through the TDI pin. Next, the TAP controller is moved to the `RUN_TEST/IDLE` state where a delay is implemented to allow the instruction time to be latched. This process is identical for data register scans, except that the data register leg of the state machine is traversed.

The high-level Jam instructions are the `DRSCAN` instruction for scanning the JTAG data register, the `IRSCAN` instruction for scanning the instruction register, and the `WAIT` command that causes the state machine to sit idle for a specified period of time. Each leg of the TAP controller is scanned repeatedly, according to instructions in the `.jbc` file, until all of the target devices are programmed.

Figure 13–21 illustrates the functional behavior of the Jam Player when it parses the `.jbc` file. When the Jam Player encounters a `DRSCAN`, `IRSCAN`, or `WAIT` instruction, it generates the proper data on TCK, TMS, and TDI to complete the instruction. The flow diagram shows branches for the `DRSCAN`, `IRSCAN`, and `WAIT` instructions. Although the Jam Player supports other instructions, they are omitted from the flow diagram for simplicity.



Figure 13–21. Jam Player Flow Diagram (Part 1 of 2)

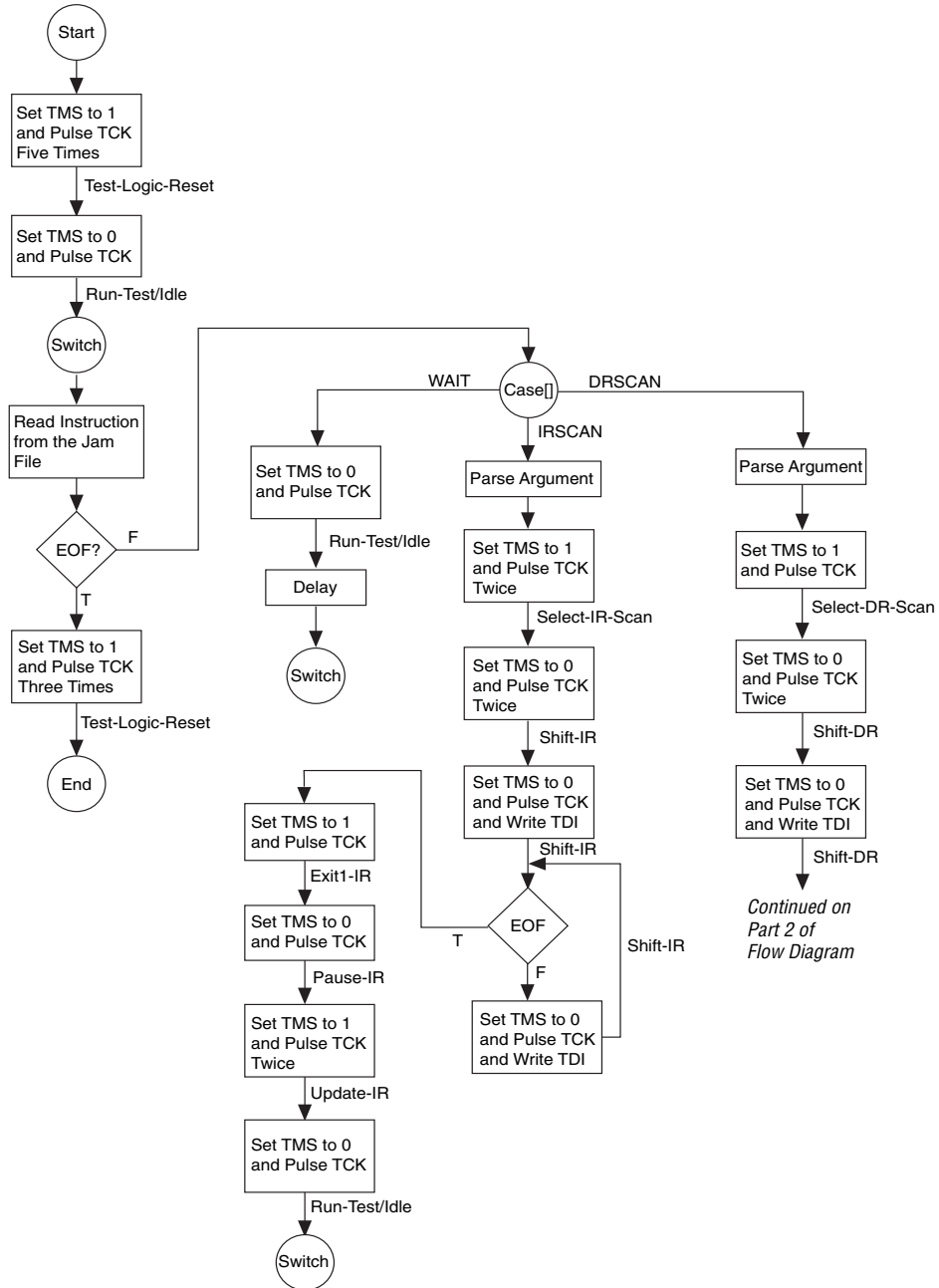
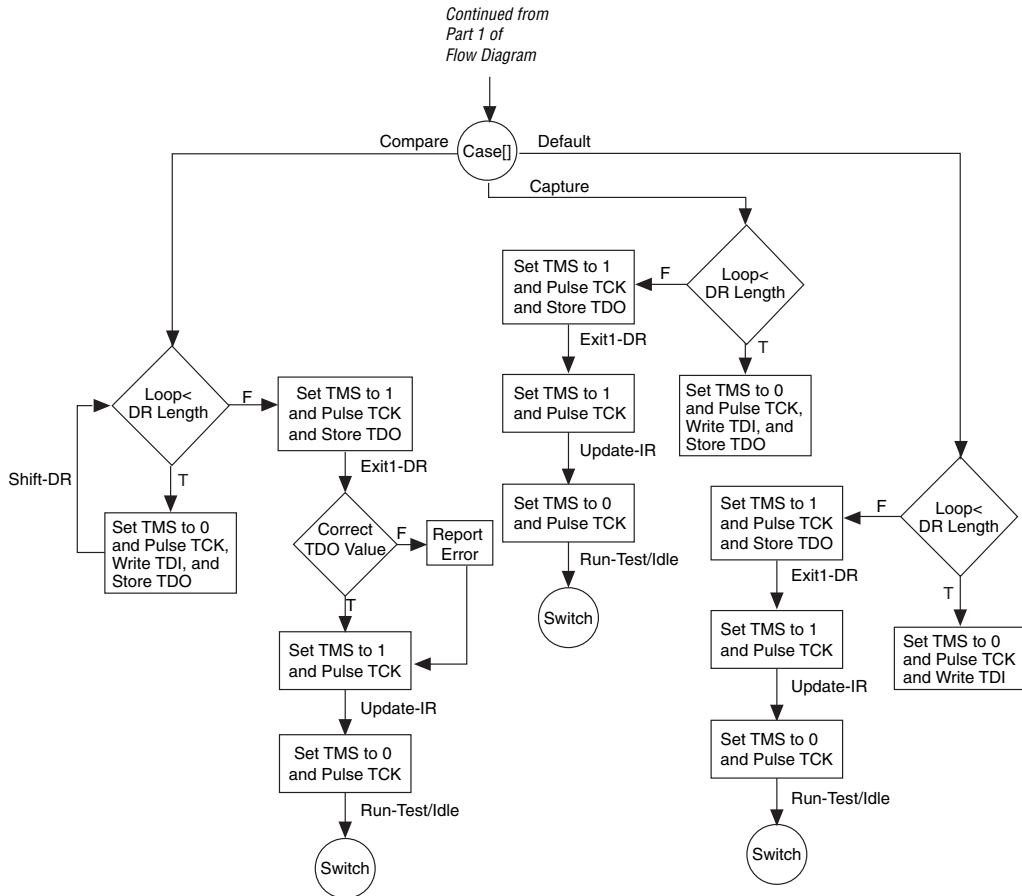


Figure 13–22. Jam Player Flow Diagram (Part 2 of 2)



Execution of a Jam program starts at the beginning of the program. The program flow is controlled using GOTO, CALL/RETURN, and FOR/NEXT structures. The GOTO and CALL statements refer to labels that are symbolic names for program statements located elsewhere in the Jam program. The language itself enforces almost no constraints on the organizational structure or control flow of a program.



The Jam language does not support linking multiple Jam programs together or including the contents of another file into a Jam program.

### Jam Instructions

Each Jam statement begins with one of the instruction names listed in [Table 13–9](#). The instruction names, including the names of the optional instructions, are reserved keywords that you cannot use as variable or label identifiers in a Jam program.

BOOLEAN	INTEGER	PREIR
CALL	IRSCAN	PRINT
CRC	IRSTOP	PUSH
DRSCAN	LET	RETURN
DRSTOP	NEXT	STATE
EXIT	NOTE	WAIT
EXPORT	POP	VECTOR (1)
FOR	POSTDR	VMAP (1)
GOTO	POSTIR	–
IF	PREDR	–

**Note to Table 13–9:**

- (1) This instruction name is an optional language extension.

[Table 13–10](#) shows the state names that are reserved keywords in the Jam language. These keywords correspond to the state names specified in the IEEE Std. 1149.1 JTAG specification.

IEEE Std. 1149.1 JTAG State Names	Jam Reserved State Names
Test-Logic-Reset	RESET
Run-Test-Idle	IDLE
Select-DR-Scan	DRSELECT
Capture-DR	DRCAPTURE
Shift-DR	DRSHIFT
Exit1-DR	DREXIT1
Pause-DR	DRPAUSE
Exit2-DR	DREXIT2
Update-DR	DRUPDATE
Select-IR-Scan	IRSELECT
Capture-IR	IRCAPTURE

**Table 13–10. Reserved Keywords (Part 2 of 2)**

IEEE Std. 1149.1 JTAG State Names	Jam Reserved State Names
Shift-IR	IRSHIFT
Exit1-IR	IREXIT1
Pause-IR	IRPAUSE
Exit2-IR	IREXIT2
Update-IR	IRUPDATE

**Example Jam File that Reads the IDCODE**

The following illustrates the flexibility and utility of the Jam STAPL. The example code reads the IDCODE out of a single device in a JTAG chain.



The array variable, `I_IDCODE`, is initialized with the IDCODE instruction bits ordered the LSB first (on the left) to most significant bit (MSB) (on the right). This order is important because the array field in the IRSCAN instruction is always interpreted and sent, MSB to LSB.

**Example Jam File Reading IDCODE**

```

BOOLEAN read_data[32];
BOOLEAN I_IDCODE[10] = BIN 1001101000; `assumed
BOOLEAN ONES_DATA[32] = HEX FFFFFFFF;
INTEGER i;
`Set up stop state for IRSCAN
IRSTOP IRPAUSE;
`Initialize device
STATE RESET;
IRSCAN 10, I_IDCODE[0..9]; `LOAD IDCODE INSTRUCTION
STATE IDLE;
WAIT 5 USEC, 3 CYCLES;
DRSCAN 32, ONES_DATA[0..31], CAPTURE read_data[0..31];
`CAPTURE IDCODE
PRINT "IDCODE:";
FOR i=0 to 31;
PRINT read_data[i];
NEXT i;
EXIT 0;

```

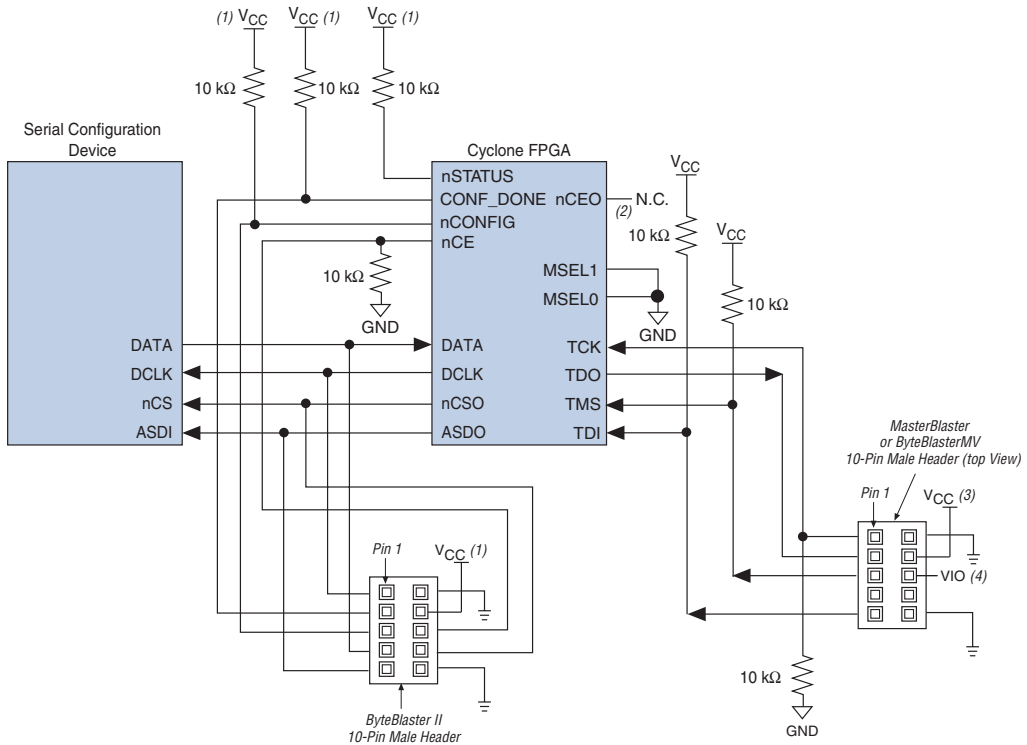
## Combining Configuration Schemes

This section shows you how to configure Cyclone FPGAs using multiple configuration schemes on the same board.

### Active Serial & JTAG

You can combine the AS configuration scheme with JTAG-based configuration. Set the `MSEL[1..0]` pins to 00 in this setup, as shown in [Figure 13–23](#). This setup uses two 10-pin download cable headers on the board. The first header programs the serial configuration device in-system via the AS programming interface, and the second header configures the Cyclone FPGA directly via the JTAG interface.

If you try configuring the device using both schemes simultaneously, JTAG configuration takes precedence and AS configuration will be terminated.

**Figure 13–23. Combining AS & JTAG Configuration****Notes to Figure 13–23:**

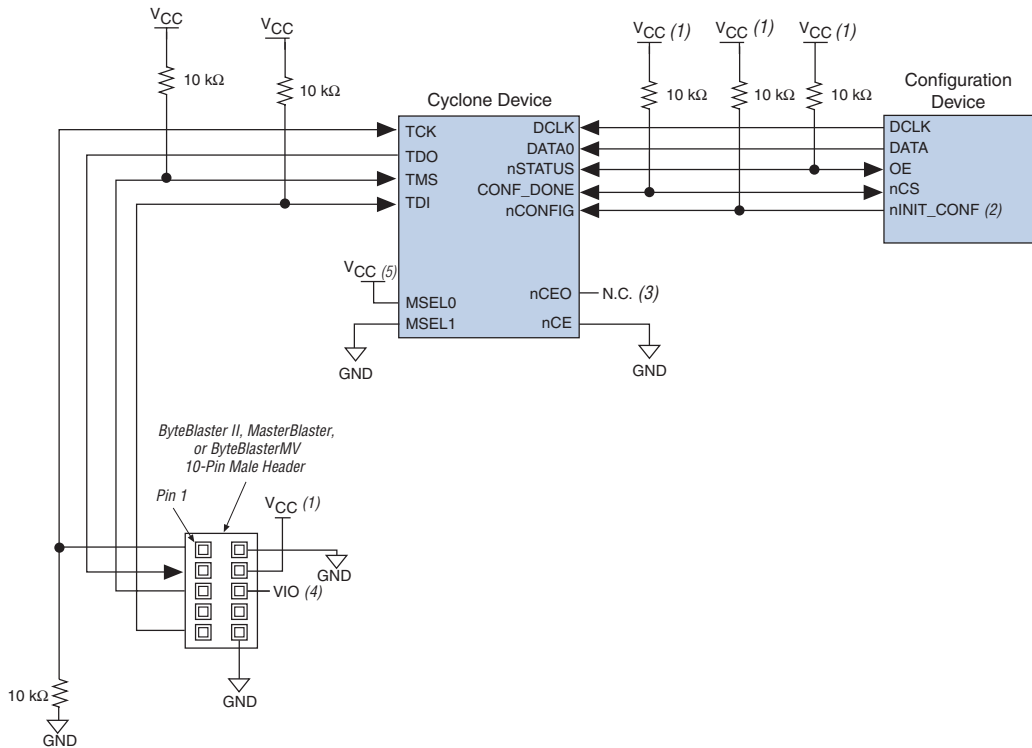
- (1) Connect these pull-up resistors to 3.3 V.
- (2) The nCEO pin is left unconnected.
- (3) You should connect the pull-up resistor to the same supply voltage as the download cable.
- (4)  $V_{IO}$  is a reference voltage for the MasterBlaster output driver.  $V_{IO}$  should match the device's  $V_{CCIO}$ . Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

## Passive Serial & JTAG

The PS- and JTAG-based configuration are also supported on the same board. Set the MSEL[1..0] pins to 01 in this setup. [Figure 13–24](#) shows the pin connections required for configuring Cyclone FPGAs using PS and JTAG interfaces on the same board. The JTAG chain only connects to the Cyclone FPGA in [Figure 13–24](#), but could also connect to the configuration device for in-system programming of that device.

If you try configuring the device using both schemes simultaneously, JTAG configuration takes precedence and PS configuration will be terminated.

Figure 13–24. Combining PS &amp; JTAG Configuration



## Notes to Figure 13–24:

- (1) The pull-up resistor should be connected to the same supply voltage as the configuration device. The EPC16, EPC8, EPC4, and EPC2 devices' OE and nCS pins have internal, user-configurable pull-up resistors. If you use internal pull-up resistors, do not use external pull-up resistors on these pins.
- (2) The nINIT\_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices. If nINIT\_CONF is not used, nCONFIG must be pulled to V<sub>CC</sub> through a resistor.
- (3) The nCEO pin is left unconnected for the last device in the chain.
- (4) V<sub>IO</sub> is a reference voltage for the MasterBlaster output driver. V<sub>IO</sub> should match the device's V<sub>CCIO</sub>. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.
- (5) Connect MSEL0 to the V<sub>CC</sub> supply voltage of the I/O bank it resides in.

## Device Options

You can set Cyclone FPGA options in Altera's Quartus II development software using the **Device & Pin Options** dialog box. Select **Compiler Settings** (Processing menu), then click on the **Chips & Devices** tab. Figure 13–25 shows the **Device & Pin Options** dialog box.

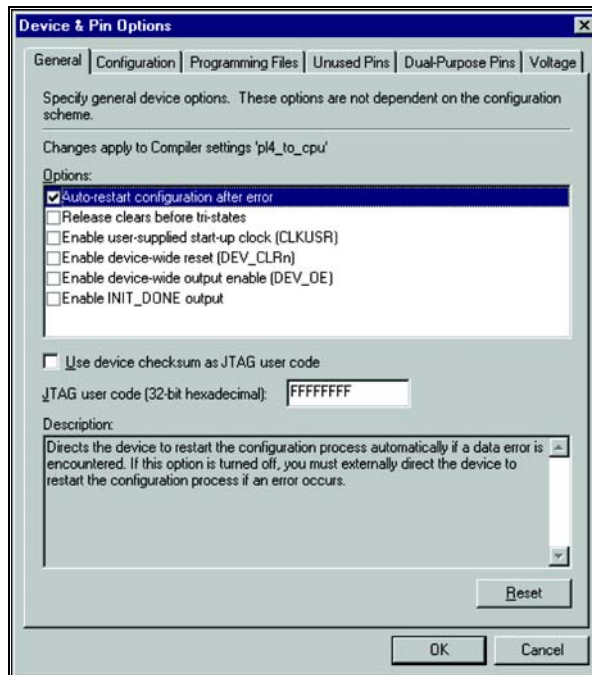
**Figure 13–25. Configuration Options Dialog Box**



Table 13–11 summarizes each of these options.

<b>Table 13–11. Cyclone Configuration Option Bits (Part 1 of 2)</b>			
<b>Device Option</b>	<b>Option Usage</b>	<b>Default Configuration (Option Off)</b>	<b>Modified Configuration (Option On)</b>
Auto-restart configuration on frame error	If a data error occurs during configuration, you can choose how to restart configuration.	The configuration process stops until you direct the device to restart configuration. The <code>nSTATUS</code> pin is driven low when an error occurs. When <code>nCONFIG</code> is pulled low and then high, the device begins to reconfigure.	<p>The configuration process restarts automatically. The <code>nSTATUS</code> pin drives low and releases. The <code>nSTATUS</code> pin is then pulled to <math>V_{CC}</math> by the pull-up resistor, indicating that configuration can restart.</p> <p>In the configuration device scheme, if the target device's <code>nSTATUS</code> pin is tied to the configuration device's <code>OE</code> pin, the <code>nSTATUS</code> reset pulse resets the configuration device automatically. The configuration device then releases its <code>OE</code> pin (which is pulled high) and reconfiguration begins.</p> <p>If an error occurs during passive configuration, the device can be reconfigured without the system having to pulse <code>nCONFIG</code>. After <code>nSTATUS</code> goes high, reconfiguration can begin.</p>
Release clears before tri-states	During configuration, the device I/O pins are tri-stated. During initialization, you choose the order for releasing the tri-states and clearing the registers.	The device releases the tri-states on its I/O pins before releasing the clear signal on its registers.	The device releases the clear signals on its registers before releasing the tri-states. You can use this option to allow the design to operate before it drives out, so all outputs do not start up low.
Enable chip-wide reset	Enables a single pin to reset all device registers.	Chip-wide reset is not enabled. The <code>DEV_CLRn</code> pin is available as a user I/O pin.	Chip-wide reset is enabled for all registers in the device. All registers are cleared when the <code>DEV_CLRn</code> pin is driven low.

**Table 13–11. Cyclone Configuration Option Bits (Part 2 of 2)**

Device Option	Option Usage	Default Configuration (Option Off)	Modified Configuration (Option On)
Enable chip-wide output enable	Enables a single pin to control all device tri-states.	Chip-wide output enable is not enabled. The DEV_OE pin is available as a user I/O pin.	Chip-wide output enable is enabled for all device tri-states. After configuration, all user I/O pins are tri-stated when DEV_OE is low.
Enable INIT_DONE output	Enables a pin to drive out a signal when the initialization process is complete and the device has entered user mode.	The INIT_DONE signal is not available. The INIT_DONE pin is available as a user I/O pin.	The INIT_DONE signal is available on the open-drain INIT_DONE pin. This pin drives low during configuration. After initialization, it is released and pulled high externally. The INIT_DONE pin must be connected to a 10-k $\Omega$ pull-up resistor. If the INIT_DONE output is used, the INIT_DONE pin cannot be used as a user I/O pin.
Data Compression	Enables Cyclone FPGAs to receive compressed configuration bit stream in Active and PS configuration schemes.	The Quartus II software generates uncompressed programming files and Cyclone FPGAs do not decompress data.	The Quartus II software generates compressed programming files and Cyclone FPGAs decompress the bit stream during configuration.

## Device Configuration Pins

Tables 13–12 through 13–14 describe the connections and functionality of all the configuration related pins on the Cyclone device. Table 13–12 describes the dedicated configuration pins. These pins are required to be connected properly on your board for successful configuration. Some of these pins may not be required for your configuration schemes.

**Table 13–12. Dedicated Cyclone Device Configuration Pins (Part 1 of 3)**

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
MSEL1 MSEL0	–	All	Input	Two-bit configuration input that set the Cyclone device configuration scheme (see Table 13–2). Use these pins to select the Cyclone configuration schemes for the appropriate connections. These pins must remain at a valid state during power-up before nCONFIG is pulled low to initiate a reconfiguration and during configuration.
nCONFIG	–	All	Input	Configuration control input. Pulling this pin low during user-mode causes the FPGA to lose its configuration data, enter a reset state, and tri-state all I/O pins. Returning this pin to a logic high will initiate a reconfiguration. If the configuration scheme uses an enhanced configuration device or EPC2 device, the nCONFIG pin can be tied directly to V <sub>CC</sub> or to the configuration device's nINIT_CONF pin.

**Table 13–12. Dedicated Cyclone Device Configuration Pins (Part 2 of 3)**

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nSTATUS	–	All	Bidirectional open-drain	<p>The device drives nSTATUS low immediately after power-up and releases it within 5 <math>\mu</math>s. (When using a configuration device, the configuration device holds nSTATUS low for up to 200 ms.)</p> <p>Status output. If an error occurs during configuration, nSTATUS is pulled low by the target device.</p> <p>Status input. If an external source drives the nSTATUS pin low during configuration or initialization, the target device enters an error state. Driving nSTATUS low after configuration and initialization does not affect the configured device.</p> <p>If the design uses a configuration device, driving nSTATUS low causes the configuration device to attempt to configure the FPGA, but since the FPGA ignores transitions on nSTATUS in user-mode, the FPGA will not reconfigure. To initiate a reconfiguration, nCONFIG must be pulled low. The enhanced configuration devices' and EPC2 devices' OE and nCS pins have optional internal programmable pull-up resistors. If the design uses internal pull-up resistors, do not use external 10-k<math>\Omega</math> pull-up resistors on these pins.</p>
CONF_DONE	–	All	Bidirectional open-drain	<p>Status output. The target device drives the CONF_DONE pin low before and during configuration. Once all configuration data is received without error and the initialization clock cycle starts, the target device releases CONF_DONE.</p> <p>Status input. After all data is received and CONF_DONE goes high, the target device initializes and enters user mode.</p> <p>Driving CONF_DONE low after configuration and initialization does not affect the configured device. The enhanced configuration devices' and EPC2 devices' OE and nCS pins have optional internal programmable pull-up resistors. If the design uses internal pull-up resistors, do not use external 10-k<math>\Omega</math> pull-up resistors on these pins.</p>

**Table 13–12. Dedicated Cyclone Device Configuration Pins (Part 3 of 3)**

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
DCLK	–	PS AS	Input (PS) Output (AS)	In PS configuration, the clock input clocks data from an external source into the target device. Data is latched into the FPGA on the rising edge of DCLK. In AS configuration, DCLK is an output from the Cyclone FPGA that provides timing for the configuration interface. After configuration, the logic levels on this pin do not affect the Cyclone FPGA.
ASDO	I/O in PS mode, N/A in AS mode	AS	Output	Control signal from the Cyclone FPGA to the serial configuration device in AS mode used to read out configuration data.
nCSO	I/O in PS mode, N/A in AS mode	AS	Output	Output control signal from the Cyclone FPGA to the serial configuration device in AS mode that enables the configuration device.
nCE	–	All	Input	Active-low chip enable. The nCE pin activates the device with a low signal to allow configuration. The nCE pin must be held low during configuration, initialization, and user mode. In single device configuration, tie the nCE pin low. In multi-device configuration, the first device's nCE pin is tied low while its nCEO pin is connected to nCE of the next device in the chain. Hold the nCE pin low for programming the FPGA via JTAG.
nCEO	–	All	Output	Output that drives low when device configuration is complete. In single device configuration, this pin is left floating. In multi-device configuration, this pin feeds the next device's nCE pin. The nCEO of the last device in the chain is left floating.
DATA0	–	All	Input	Data input. In serial configuration mode, bit-wide configuration data is presented to the target device on the DATA0 pin. Toggling DATA0 after configuration does not affect the configured device.

Table 13–13 describes the optional configuration pins. If these optional configuration pins are not enabled in the Quartus II software, they are available as general-purpose user I/O pins. Therefore during configuration, these pins function as user I/O pins and are tri-stated with weak pull-ups.

**Table 13–13. Optional Cyclone Device Configuration Pins**

Pin Name	User Mode	Pin Type	Description
CLKUSR	N/A if option is on, I/O if option is off	Input	Optional user-supplied clock input. Synchronizes the initialization of one or more devices. This pin is enabled by turning on the <b>Enable user-supplied start-up clock (CLKUSR)</b> option in the Quartus II software.
INIT_DONE	N/A if option is on, I/O if option is off	Output open-drain	Status pin. Can be used to indicate when the device has initialized and is in user mode. The INIT_DONE pin must be pulled to V <sub>CC</sub> with a 10-kΩ resistor. The INIT_DONE pin drives low during configuration. Before and after configuration, the INIT_DONE pin is released and is pulled to V <sub>CC</sub> by an external pull-up resistor. Because INIT_DONE is tri-stated before configuration, it is pulled high by the external pull-up resistor. Thus, the monitoring circuitry must be able to detect a low-to-high transition. This pin is enabled by turning on the <b>Enable INIT_DONE output</b> option in the Quartus II software.
DEV_OE	N/A if the option is on, I/O if the option is off.	Input	Optional pin that allows the user to override all tri-states on the device. When this pin is driven low, all I/O pins are tri-stated; when this pin is driven high, all I/O pins behave as programmed. This pin is enabled by turning on the <b>Enable device-wide output enable (DEV_OE)</b> option in the Quartus II software.
DEV_CLRn	N/A if the option is on, I/O if the option is off.	Input	Optional pin that allows you to override all clears on all device registers. When this pin is driven low, all registers are cleared; when this pin is driven high, all registers behave as programmed. This pin is enabled by turning on the <b>Enable device-wide reset (DEV_CLRn)</b> option in the Quartus II software.

Table 13–14 describes the dedicated JTAG pins. JTAG pins must be kept stable before and during configuration to prevent accidental loading of JTAG instructions.

<b>Pin Name</b>	<b>User Mode</b>	<b>Pin Type</b>	<b>Description</b>
TDI	N/A	Input	Serial input pin for instructions as well as test and programming data. Data is shifted in on the rising edge of TCK. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V <sub>CC</sub> .
TDO	N/A	Output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of TCK. The pin is tri-stated if data is not being shifted out of the device. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by leaving this pin unconnected.
TMS	N/A	Input	Input pin that provides the control signal to determine the transitions of the TAP controller state machine. Transitions within the state machine occur on the rising edge of TCK. Therefore, TMS must be set up before the rising edge of TCK. TMS is evaluated on the rising edge of TCK. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V <sub>CC</sub> .
TCK	N/A	Input	The clock input to the BST circuitry. Some operations occur at the rising edge, while others occur at the falling edge. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to ground.

## Device Configuration Files

The Quartus II software can create one or more configuration and programming files to support the configuration schemes discussed in this chapter. This section describes these files.

### SRAM Object File (.sof)

You should use an **.sof** during PS and JTAG configuration when the data is downloaded directly from the ByteBlaster II, MasterBlaster, or ByteBlasterMV download cables. For Cyclone FPGAs, the Quartus II Compiler's Assembler module automatically creates the **.sof** file for each device in your design. The Quartus II software controls the configuration sequence and automatically inserts the appropriate headers into the configuration data stream. All other configuration files are created from the **.sof**.

## Programmer Object File (.pof)

A **.pof** is used by the Altera programming hardware to program a configuration device, including serial configuration devices and enhanced configuration devices. A **.pof** is automatically generated when a Cyclone project is compiled for the configuration device selected in the **Configuration** dialog box.

## Raw Binary File (.rbf)

The **.rbf** is a binary file (e.g., one byte of **.rbf** data is eight configured bits 10000101 (85 Hex)) containing the configuration data. Store data so that the LSB of each data byte is loaded first. A mass storage device can store the converted image. The microprocessor can then read data from the binary file and load it into device. You can also use the microprocessor to perform real-time conversion during configuration. In the PS configuration scheme, the data is shifted in serially, LSB first.

## Hexadecimal (Intel-Format) File (.hex)

A **.hex** file is an ASCII file in the Intel hexadecimal format. Third-party programmers use this file to program Altera's serial configuration devices. Microprocessors can also use the **.hex** file to store and transmit configuration data using the PS configuration scheme.

## Tabular Text File (.tff)

The **.tff** file is a tabular ASCII file that provides a comma-separated version of the configuration data for the bit-wide PS configuration scheme. In some applications, the storage device containing the configuration data is neither dedicated to nor connected directly to the target device. For example, a configuration device can also contain executable code for a system (e.g., BIOS routines) and other data. The **.tff** allows you to include the configuration data as part of the microprocessor's source code using the `include` or `source` commands. The microprocessor can access this data from a configuration device or mass-storage device and load it into the target device. A **.tff** can be imported into nearly any assembly language or high-level language compiler.

## Jam File (.jam)

A **.jam** file is an ASCII text file in the Jam device programming language that stores device programming information. These files are used to program, verify, and blank-check one or more devices in the Quartus II Programmer or in an embedded processor-type environment.



## Jam Byte-Code File (.jbc)

A **.jbc** file is a binary version of a Jam file in a byte-code representation. The **.jbc** file stores device programming information used to program, verify, and blank-check one or more devices.

## Configuration Reliability

The Cyclone architecture is designed to minimize the effects of power supply and data noise in a system, and to ensure that the configuration data is not corrupted during configuration or normal user-mode operation. A number of circuit design features ensure the highest possible level of reliability from this SRAM technology.

Cyclic redundancy code (CRC) circuitry validates each data frame (i.e., sequence of data bits) as it is loaded into the target device. If the CRC generated by the device does not match the data stored in the data stream, the configuration process is halted, and the **nSTATUS** pin is pulled and held low to indicate an error condition. CRC circuitry ensures that noisy systems will not cause errors that yield an incorrect or incomplete configuration.

The Cyclone FPGA architecture also provides a very high level of reliability in low-voltage brown-out conditions. Cyclone FPGA SRAM blocks require a certain  $V_{CC}$  level to maintain accurate data. This voltage threshold is significantly lower than the voltage required to activate the device's POR circuitry. Therefore, the target device stops operating if the  $V_{CC}$  starts to fail, and indicates an operation error by pulling and holding the **nSTATUS** pin low. You must then reconfigure the device before it can resume operation as a logic device. In active configuration schemes in which **nCONFIG** is tied to  $V_{CC}$ , reconfiguration begins as soon as  $V_{CC}$  returns to an acceptable level. The low pulse on **nSTATUS** resets the configuration device by driving **OE** low. In passive configuration schemes, the host system starts the reconfiguration process.

These device features ensure that Cyclone FPGAs have the highest possible reliability in a wide variety of environments, and provide the same high level of system reliability that exists in other Altera PLDs.

## Board Layout Tips

Even though the **DCLK** signal (used in PS and AS configuration schemes) is fairly low-frequency, it drives edge-triggered pins on the Cyclone FPGA. Therefore, any overshoot, undershoot, ringing, or other noise can affect configuration. When designing the board, lay out the **DCLK** trace using the same techniques as laying out a clock line, including appropriate buffering. If more than five devices are used, Altera recommends using buffers to split the fan-out on the **DCLK** signal.



### Features

The serial configuration devices provide the following features:

- 1- and 4-Mbit flash memory devices that serially configure Cyclone™ FPGAs using the active serial (AS) configuration scheme
- Easy-to-use four-pin interface
- Low cost, low pin count and non-volatile memory
- Low current during configuration and near-zero standby mode current
- 3.3-V operation
- Available in 8-pin small outline integrated circuit (SOIC) package
- Enables the Nios® processor to access unused flash memory through AS memory interface
- Re-programmable memory with more than 100,000 erase/program cycles
- Write protection support for memory sectors using status register bits
- In-system programming support with SRunner software driver
- Programming support with ByteBlaster™ II download cable
- Additional programming support with the Altera® Programming Unit (APU) and programming hardware from BP Microsystems, System General, and other vendors
- Software design support with the Altera Quartus® II development system for Windows-based PCs as well as Sun SPARC station and HP 9000 Series 700/800
- Delivered with the memory array erased (all the bits set to 1)



Whenever the term “serial configuration device(s)” is used in this document, it refers to Altera EPCS1 and EPCS4 devices.

### Functional Description

With SRAM-based devices such as Cyclone FPGAs, configuration data must be reloaded each time the device powers up, the system initializes, or when new configuration data is needed. Serial configuration devices are flash memory devices with a serial interface that can store

configuration data for a Cyclone device and reload the data to the device upon power-up or reconfiguration. [Table 14–1](#) lists the serial configuration devices.

Device	Memory Size (Bits)
EPCS1	1,048,576
EPCS4	4,194,304

[Table 14–2](#) lists the serial configuration device used with each Cyclone FPGA and the configuration file size.

Cyclone Device	Raw Binary Configuration File Size (Bits) (1)	Serial Configuration Device	
		EPCS1	EPCS4
EP1C3	627,376	✓	✓
EP1C4	924,512	✓	✓
EP1C6	1,167,216	✓ (2)	✓
EP1C12	2,326,528		✓
EP1C20	3,559,608		✓

**Note to [Table 14–2](#):**

- (1) These are preliminary, uncompressed file sizes.
- (2) The EP1C6 device's programming file fits in an EPCS1 device with compression turned on.

With the new data-decompression feature in the Cyclone FPGA family, designers can use smaller serial configuration devices to configure larger Cyclone FPGAs.

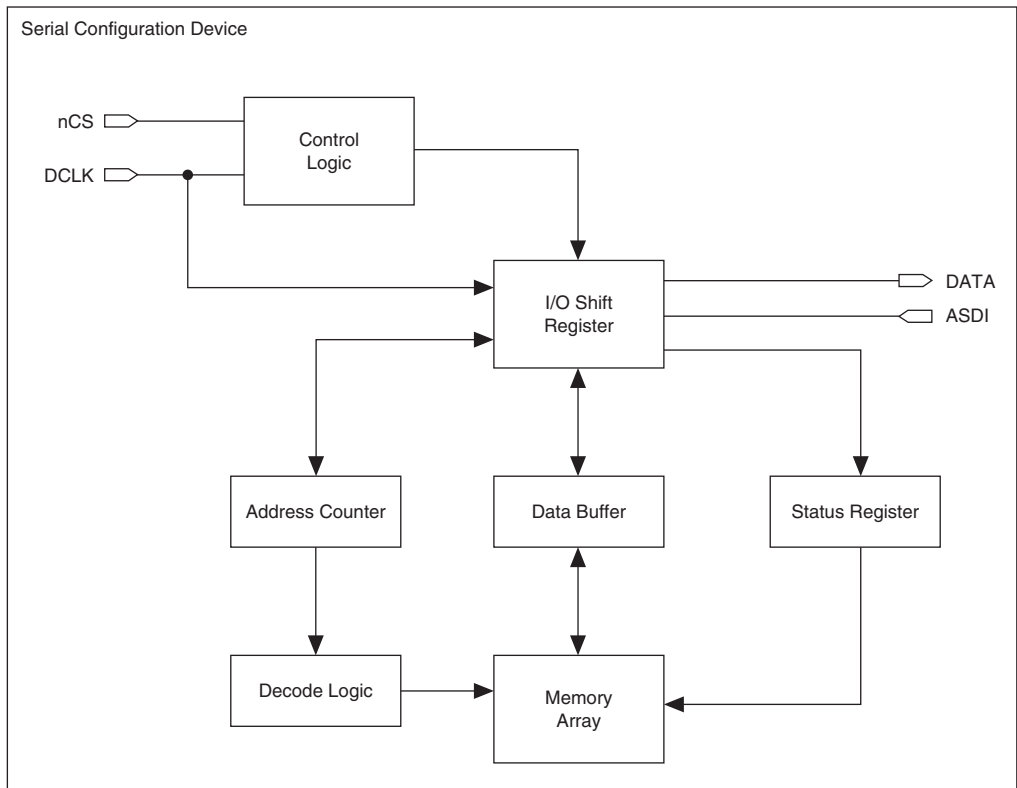


Serial configuration devices cannot be cascaded.



See [Chapter 13, Configuring Cyclone FPGAs](#) for more information regarding the Cyclone FPGA decompression feature in AS mode.

The serial configuration devices are designed to configure Cyclone FPGAs and cannot configure other existing Altera device families. [Figure 14–1](#) shows the serial configuration device block diagram.

**Figure 14–1. Serial Configuration Device Block Diagram**

## Accessing Memory in Serial Configuration Devices

A designer can access the unused memory locations of the serial configuration device to store or retrieve data through the Nios processor and SOPC Builder. SOPC Builder is an Altera tool for creating bus-based (especially microprocessor-based) systems in Altera devices. SOPC Builder assembles library components like processors and memories into custom microprocessor systems.

SOPC Builder includes the active serial memory interface (ASMI) peripheral, an interface core specifically designed to work with the serial configuration device. Using this core, a designer can create a system with a Nios embedded processor that allows software access to any memory location within the serial configuration device.

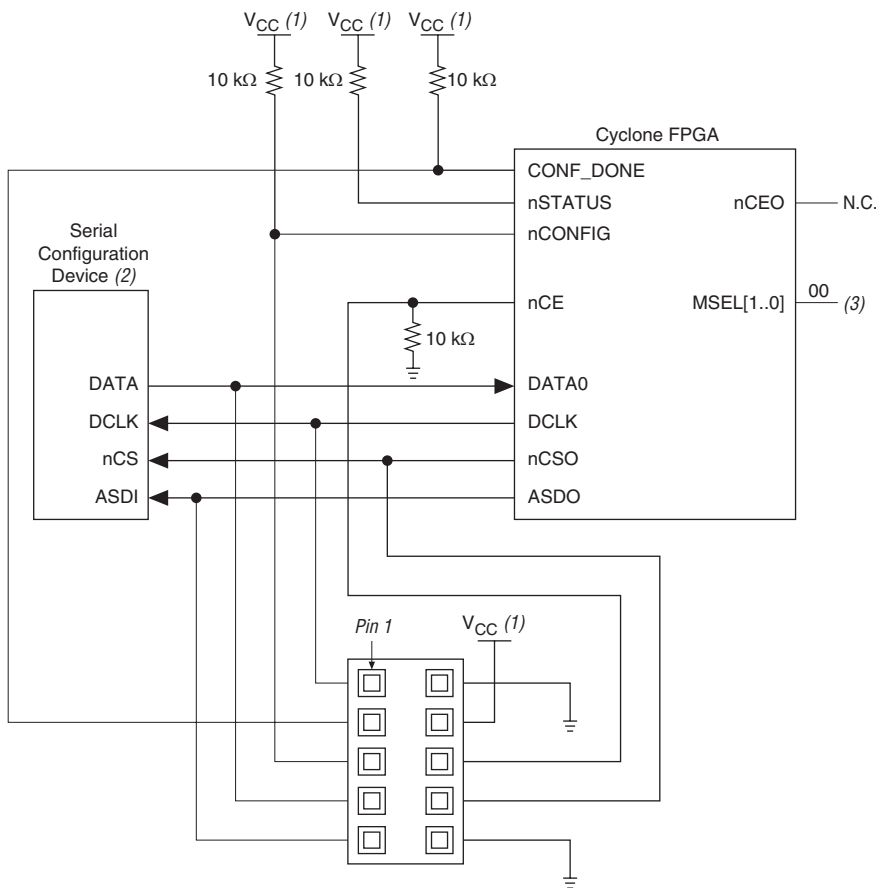


For more information on accessing memory within the serial configuration device, see the *Active Serial Memory Interface Data Sheet*.

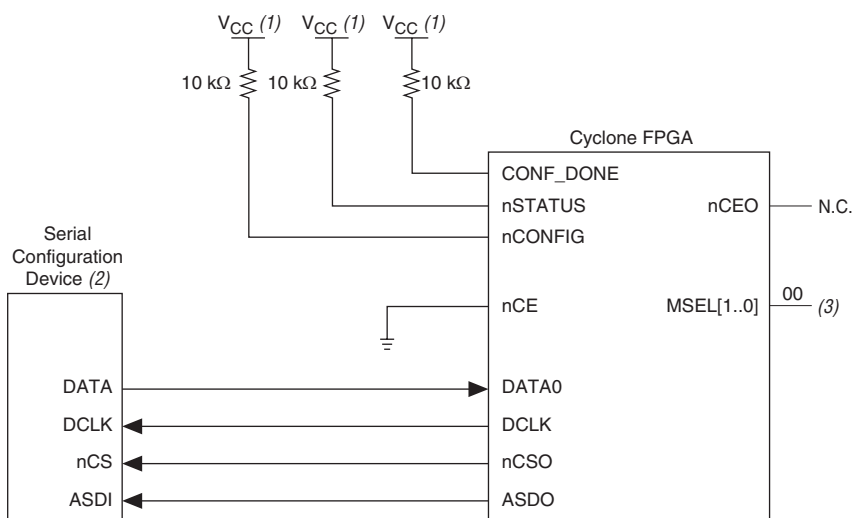
## Cyclone FPGA Configuration

Cyclone FPGAs can be configured with a serial configuration device through the AS configuration mode. There are four signals on the serial configuration device that interface directly with the Cyclone device's control signals. The serial configuration device signals DATA, DCLK, ASDI, and nCS interface with DATA0, DCLK, ASDO, and nCSO control signals on a Cyclone FPGA, respectively. [Figure 14–2](#) shows a serial configuration device programmed via a download cable which configures a Cyclone FPGA in AS mode. [Figure 14–3](#) shows a serial configuration device programmed using the APU or a third-party programmer configuring a Cyclone FPGA in AS configuration mode.

**Figure 14–2. Cyclone Configuration in AS Mode (Serial Configuration Device Programmed Using Download Cable)**



**Figure 14–3. Cyclone Configuration in AS Mode (Serial Configuration Device Programmed by APU or Third-Party Programmer)**



**Notes to Figures 14–2 and 14–3:**

- (1)  $V_{CC} = 3.3\text{-V}$ .
- (2) Serial configuration devices cannot be cascaded.
- (3) Set MSEL0 to 0 and MSEL1 to 0 for AS configuration mode.

The Cyclone FPGA acts as the configuration master in the configuration flow and provides the clock to the serial configuration device. The Cyclone device enables the serial configuration device by pulling the nCS signal low via the nCSO signal (See Figures 14–2 and 14–3). Subsequently, the Cyclone FPGA sends the instructions and addresses to the serial configuration device via the ASDO signal. The serial configuration device responds to the instructions by sending the configuration data to the Cyclone FPGA's DATA0 pin on the falling edge of DCLK. The data is latched into the Cyclone device on the DCLK signal's rising edge.

The Cyclone FPGA controls the nSTATUS and CONF\_DONE pins during configuration in AS mode. If the CONF\_DONE signal does not go high at the end of configuration or if the signal goes high too early, the Cyclone FPGA will pulse its nSTATUS pin low to start reconfiguration. Upon successful configuration, the Cyclone FPGA releases the CONF\_DONE pin, allowing the external 10-kΩ resistor to pull this signal high. Initialization begins after the CONF\_DONE goes high and completes within 136 clock cycles. After initialization, the Cyclone FPGA enters user mode.

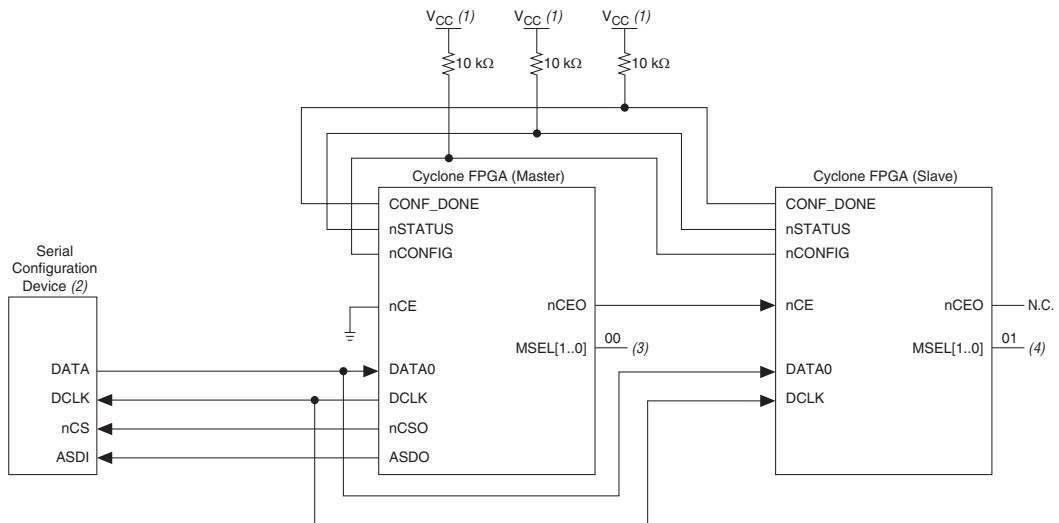


For more information on configuring Cyclone FPGAs in AS mode or other configuration modes, see [Chapter 13, Configuring Cyclone FPGAs](#).



Multiple Cyclone devices can be configured by a single EPCS device. However, serial configuration devices cannot be cascaded. Check [Table 14-1](#) to ensure the programming file size of the cascaded Cyclone FPGAs does not exceed the capacity of a serial configuration device. [Figure 14-4](#) shows the AS configuration scheme with multiple Cyclone FPGAs in the chain. In AS configuration mode, all the devices in the chain must be Cyclone devices. The first Cyclone device is the configuration master and has its MSEL[1..0] pins set to AS mode. The following Cyclone devices are configuration slave devices and have the MSEL[1..0] pins set to PS mode.

**Figure 14-4. Multiple Devices in AS Mode**



**Notes to Figure 14-4:**

- (1)  $V_{CC} = 3.3\text{-V}$ .
- (2) Serial configuration devices cannot be cascaded.
- (3) Set MSEL0 to 0 and MSEL1 to 0 to select AS mode in the Cyclone device.
- (4) Set MSEL0 to 0 and MSEL1 to 1 to select PS mode in the Cyclone device.

## Serial Configuration Device Memory Access

This section describes the serial configuration device's memory array organization and operation codes. Timing specifications for the memory are provided in the ["Timing Information"](#) section.

## Memory Array Organization

Table 14–3 provides details on the memory array organization in EPCS4 and EPCS1 devices.

Details	EPCS4	EPCS1
Bytes (bits)	524,888 bytes (4 Mbits)	131, 072 bytes (1 Mbit)
Number of sectors	8	4
Bytes (bits) per sector	65,536 bytes (512 Kbits)	32,768 bytes (256 Kbits)
Pages per sector	256	128
Total number of pages	2,048	512
Bytes per page	256 bytes	256 bytes

Tables 14–4 and 14–5 show the address range for each sector in the EPCS4 and EPCS1 devices, respectively.

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
7	H'70000	H'7FFFF
6	H'60000	H'6FFFF
5	H'50000	H'5FFFF
4	H'40000	H'4FFFF
3	H'30000	H'3FFFF
2	H'20000	H'2FFFF
1	H'10000	H'1FFFF
0	H'00000	H'0FFFF

**Table 14–5. Address Range for Sectors in EPCS1 Devices**

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
3	H'18000	H'1FFFF
2	H'10000	H'17FFF
1	H'08000	H'0FFFF
0	H'00000	H'07FFF

## Operation Codes

This section describes the operations that can be used to access the memory in serial configuration devices. The DATA, DCLK, ASDI, and nCS signals access to the memory in serial configuration devices. All serial configuration device operation codes, addresses and data are shifted in and out of the device serially, with the most significant bit (MSB) first.

The device samples the active serial data input on the first rising edge of the DCLK after the active low chip select (nCS) input signal is driven low. Shift the operation code (MSB first) serially into the serial configuration device through the active serial data input pin. Each operation code bit is latched into the serial configuration device on the rising edge of the DCLK.

Different operations require a different sequence of inputs. While executing an operation, you must shift in the desired operation code, followed by the address bytes, data bytes, both, or neither. The device must drive nCS high after the last bit of the operation sequence is shifted in. [Table 14–6](#) shows the operation sequence for every operation supported by the serial configuration devices.

For the read byte, read status, and read silicon ID operations, the shifted-in operation sequence is followed by data shifted out on the DATA pin. You can drive the nCS pin high after any bit of the data-out sequence is shifted out.

For the write byte, erase bulk, erase sector, write enable, write disable, and write status operations, drive the nCS pin high exactly at a byte boundary (drive the nCS pin high a multiple of eight clock pulses after the nCS pin was driven low). Otherwise, the operation is rejected and will not be executed.

All attempts to access the memory contents while a write or erase cycle is in progress will not be granted, and the write or erase cycle will continue unaffected.

**Table 14–6. Operation Codes for Serial Configuration Devices**

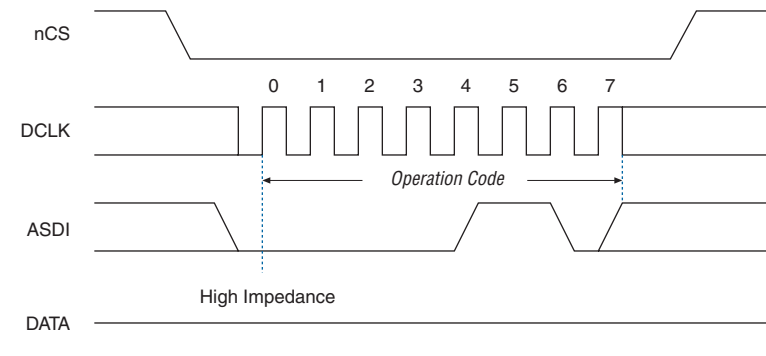
Operation	Operation Code (1)	Address Bytes	Dummy Bytes	Data Bytes	DCLK f <sub>MAX</sub> (MHz)
Write enable	0000 0110	0	0	0	25
Write disable	0000 0100	0	0	0	25
Read status	0000 0101	0	0	1 to infinite (2)	25
Read bytes	0000 0011	3	0	1 to infinite (2)	20
Read silicon ID	1010 1011	0	3	1 to infinite (2)	25
Write status	0000 0001	0	0	1	25
Write bytes	0000 0010	3	0	1 to 256 (3)	25
Erase bulk	1100 0111	0	0	0	25
Erase sector	1101 1000	3	0	0	25

**Notes to Table 14–6:**

- (1) The MSB is listed first and the LSB is listed last.
- (2) The status register, data or silicon ID are read out at least once on the DATA pin and will continuously be read out until nCS is driven high
- (3) Write bytes operation requires at least one data byte on the DATA pin. If more than 256 bytes are sent to the device, only the last 256 bytes are written to the memory.

### Write Enable Operation

The write enable operation code is b'0000 0110, and the most significant bit is listed first. The write enable operation sets the write enable latch bit, which is bit 1 in the status register. Always set the write enable latch bit before write bytes, write status, erase bulk, and erase sector operations. Figure 14–5 shows the timing diagram for the write enable operation. Figures 14–7 and 14–8 show the status register bit definitions.

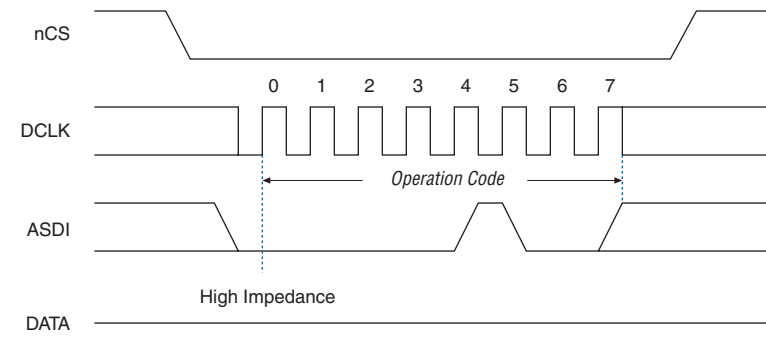
**Figure 14–5. Write Enable Operation Timing Diagram**

### Write Disable Operation

The write disable operation code is  $b'0000\ 0100$ , with the MSB listed first. The write disable operation resets the write enable latch bit, which is bit 1 in the status register. To prevent the memory from being written unintentionally, the write enable latch bit is automatically reset when implementing the write disable operation as well as under the following conditions:

- Power up
- Write bytes operation completion
- Write status operation completion
- Erase bulk operation completion
- Erase sector operation completion

Figure 14–6 shows the timing diagram for the write disable operation.

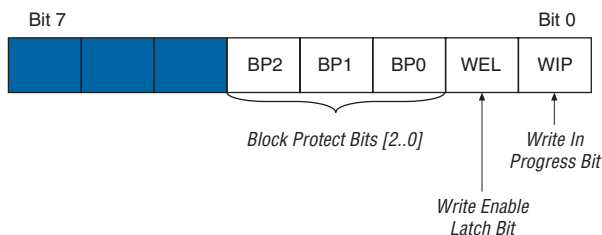
**Figure 14–6. Write Disable Operation Timing Diagram**

### Read Status Operation

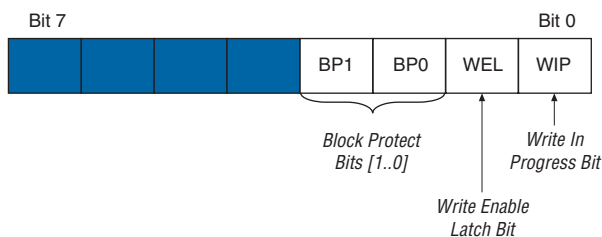
The read status operation code is  $b'0000\ 0101$ , with the MSB listed first. You can use the read status operation to read the status register.

Figures 14–7 and 14–8 show the status bits in the status register of both serial configuration devices.

**Figure 14–7. EPCS4 Status Register Status Bits**



**Figure 14–8. EPCS1 Status Register Status Bits**



Setting the write in progress bit to 1 indicates that the serial configuration device is busy with a write or erase cycle. Resetting the write in progress bit to 0 means no write or erase cycle is in progress.

Resetting the write enable latch bit to 0 indicates that no write or erase cycle will be accepted. Set the write enable latch bit to 1 before every write bytes, write status, erase bulk, and erase sector operation.

The non-volatile block protect bits determine the area of the memory protected from being written or erased unintentionally. Tables 14–7 and 14–8 show the protected area in both serial configuration devices with reference to the block protect bits. The erase bulk operation is only

available when all the block protect bits are 0. When any of the block protect bits are set to one, the relevant area is protected from being written by write bytes operations or erased by erase sector operations.

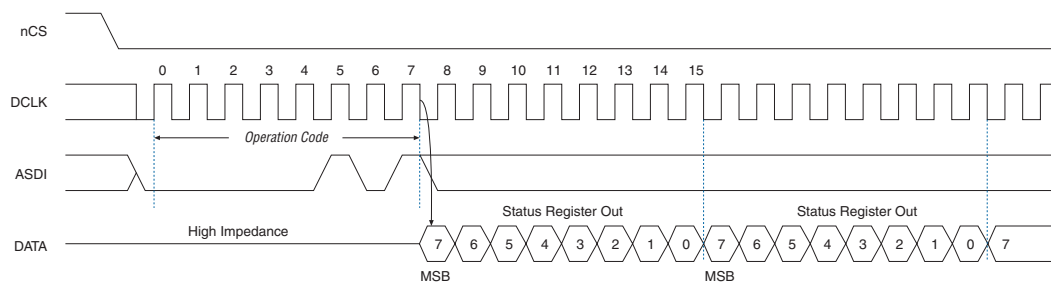
**Table 14–7. Block Protection Bits in EPCS4 Devices**

Status Register Content			Memory Content	
BP2 Bit	BP1 Bit	BPO Bit	Protected Area	Unprotected Area
0	0	0	None	All eight sectors: 0 to 7
0	0	1	Sector 7	Seven sectors: 0 to 6
0	1	0	Sectors 6 and 7	Six sectors: 0 to 5
0	1	1	Four sectors: 4 to 7	Four sectors: 0 to 3
1	0	0	All sectors	None
1	0	1	All sectors	None
1	1	0	All sectors	None
1	1	1	All sectors	None

**Table 14–8. Block Protection Bits in EPCS1**

Status Register Content		Memory Content	
BP1 Bit	BPO Bit	Protected Area	Unprotected Area
0	0	None	All four sectors: 0 to 3
0	1	Sector 3	Three sectors: 0 to 2
1	0	Two sectors: 2 and 3	Two sectors: 0 and 1
1	1	All sectors	None

The status register can be read at any time, even while a write or erase cycle is in progress. When one of these cycles is in progress, you can check the write in progress bit (bit 0 of the status register) before sending a new operation to the device. The device can also read the status register continuously, as shown in [Figure 14–9](#).

**Figure 14–9. Read Status Operation Timing Diagram**

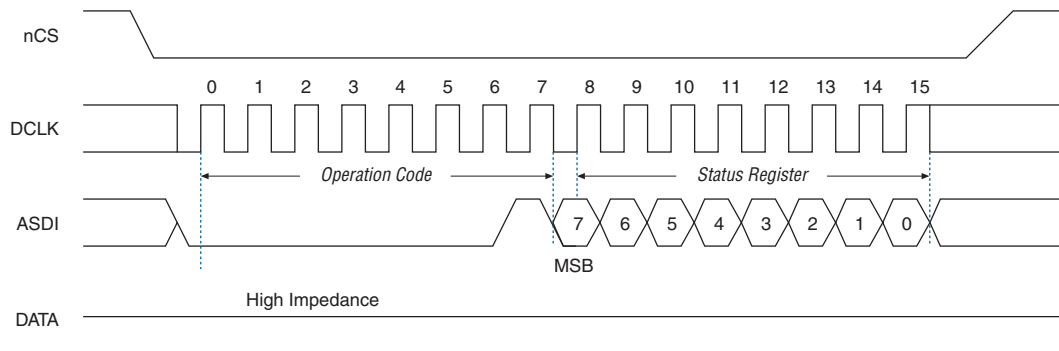
### Write Status Operation

The write status operation code is  $b'0000\ 0001$ , with the MSB listed first. Use the write status operation to set the status register block protection bits. The write status operation has no effect on the other bits. Therefore, designers can implement this operation to protect certain memory sectors, as defined in [Tables 14–7](#) and [14–8](#). After setting the block protect bits, the protected memory sectors are treated as read-only memory. Designers must execute the write enable operation before the write status operation so the device sets the status register's write enable latch bit to 1.

The write status operation is implemented by driving nCS low, followed by shifting in the write status operation code and one data byte for the status register on the ASDI pin. [Figure 14–10](#) shows the timing diagram for the write status operation. nCS must be driven high after the eighth bit of the data byte has been latched in, otherwise, the write status operation is not executed.

Immediately after nCS is driven high, the device initiates the self-timed write status cycle. The self-timed write status cycle usually takes 5 ms for both serial configuration devices and is guaranteed to be less than 15 ms (see  $t_{WS}$  in [Table 14–10](#)). Designers must account for this delay to ensure that the status register is written with desired block protect bits. Alternatively, you can check the write in progress bit in the status register by executing the read status operation while the self-timed write status cycle is in progress. The write in progress bit is 1 during the self-timed write status cycle, and is 0 when it is complete.



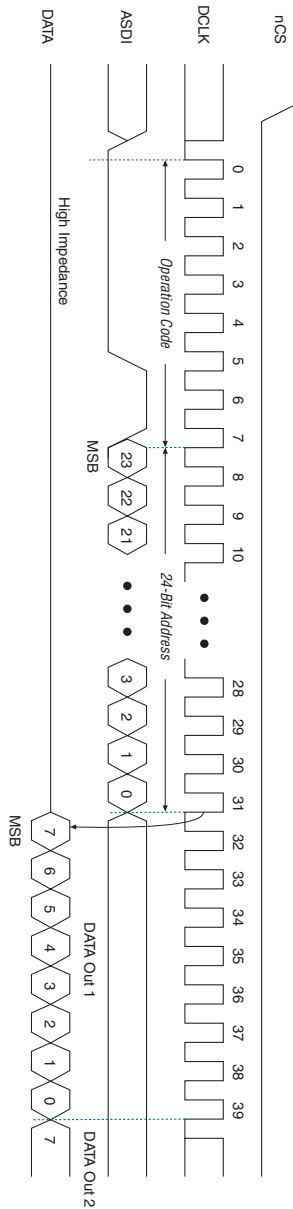
**Figure 14–10. Write Status Operation Timing Diagram**

### Read Bytes Operation

The read bytes operation code is `b'0000 0011`, with the MSB listed first. To read the memory contents of the serial configuration device, the device is first selected by driving **nCS** low. Then, the read bytes operation code is shifted-in followed by a 3-byte address (`A[23..0]`). Each address bit must be latched-in on the rising edge of **DCLK**. After the address is latched in, the memory contents of the specified address are shifted out serially on the **DATA** pin, beginning with the MSB. Each data bit is shifted out on the falling edge of **DCLK**. The maximum **DCLK** frequency during the read bytes operation is 20 MHz. [Figure 14–11](#) shows the timing diagram for read bytes operation.

The first byte addressed can be at any location. The device automatically increments the address to the next higher address after shifting out each byte of data. Therefore, the device can read the whole memory with a single read bytes operation. When the device reaches the highest address, the address counter restarts at `0x000000`, allowing the memory contents to be read out indefinitely until the read bytes operation is terminated by driving **nCS** high. The device can drive **nCS** high any time after data is shifted out. If the read bytes operation is shifted in while a write or erase cycle is in progress, the operation will not be executed. Additionally, it will not have any effect on the write or erase cycle in progress.

Figure 14–11. Read Bytes Operation Timing Diagram



**Note to Figure 14–11:**

- (1) Address bits A[23 . . 19] are don't care bits in the EPCS4 device. Address bits A[23 . . 17] are don't care bits in the EPCS1 device.

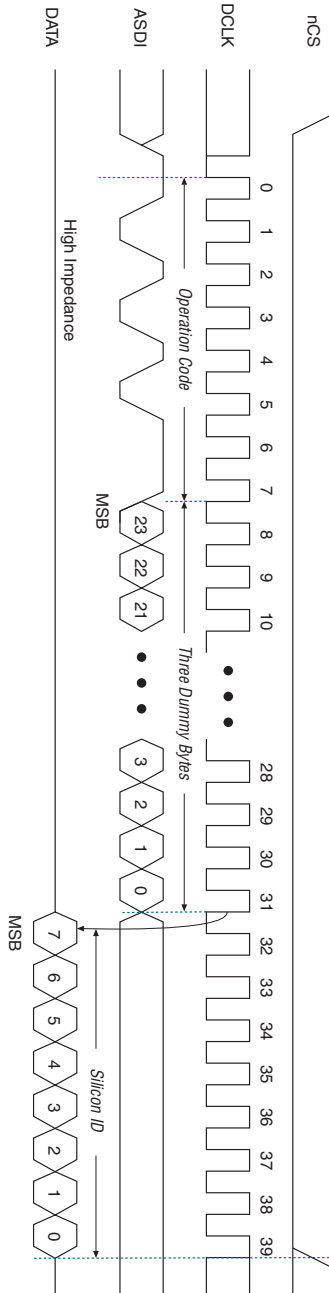
*Read Silicon ID Operation*

The read silicon ID operation code is  $b'1010\ 1011$ , with the MSB listed first. This operation reads the serial configuration device's 8-bit silicon ID from the DATA output pin. If this operation is shifted in during an erase or write cycle, it will be ignored and have no effect on the cycle that is in progress. [Table 14-9](#) shows the EPCS1 and EPCS4 device silicon IDs.

<b>Table 14-9. Serial Configuration Device Silicon ID</b>	
<b>Serial Configuration Device</b>	<b>Silicon ID (Binary Value)</b>
EPCS1	$b'0001\ 0000$
EPCS4	$b'0001\ 0010$

The device implements the read silicon ID operation by driving  $nCS$  low then shifting in the read silicon ID operation code followed by three dummy bytes on  $ASDI$ . The serial configuration device's 8-bit silicon ID is then shifted out on the DATA pin on the falling edge of  $DCLK$ , as shown in [Figure 14-12](#). The device can terminate the read silicon ID operation by driving  $nCS$  high after the silicon ID has been read at least once. Sending additional clock cycles on  $DCLK$  while  $nCS$  is driven low can cause the silicon ID to be shifted out repeatedly.

Figure 14–12. Read Silicon ID Operation Timing Diagram



### *Write Bytes Operation*

The write bytes operation code is  $b'0000\ 0010$ , with the MSB listed first. The write bytes operation allows bytes to be written to the memory. The write enable operation must be executed prior to the write bytes operation to set the write enable latch bit in the status register to 1.

The write bytes operation is implemented by driving  $nCS$  low, followed by the write bytes operation code, three address bytes and a minimum one data byte on ASDI. If the eight least significant address bits ( $A[7..0]$ ) are not all 0, all sent data that goes beyond the end of the current page is not written into the next page. Instead, this data is written at the start address of the same page (from the address whose eight LSBs are all 0). Drive  $nCS$  low during the entire write bytes operation sequence as shown in [Figure 14-13](#).

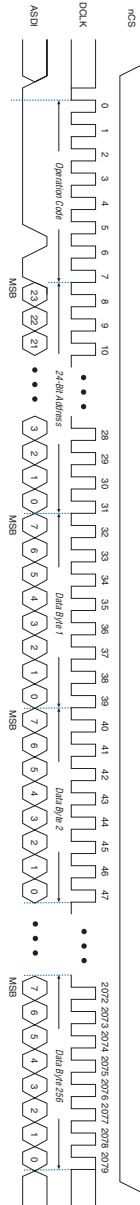
If more than 256 data bytes are shifted into the serial configuration device with a write bytes operation, the previously latched data is discarded and the last 256 bytes are written to the page. However, if less than 256 data bytes are shifted into the serial configuration device, they are guaranteed to be written at the specified addresses and the other bytes of the same page are unaffected.

If the design must write more than 256 data bytes to the memory, it needs more than one page of memory. Send the write enable and write bytes operation codes followed by three new targeted address bytes and 256 data bytes before a new page is written.

$nCS$  must be driven high after the eighth bit of the last data byte has been latched in. Otherwise, the device will not execute the write bytes operation. The write enable latch bit in the status register is reset to 0 before the completion of each write bytes operation. Therefore, the write enable operation must be carried out before the next write bytes operation.

The device initiates the self-timed write cycle immediately after  $nCS$  is driven high. The self-timed write cycle usually takes 1.5 ms for EPCS4 devices and 2 ms for EPCS1 devices and is guaranteed to be less than 5 ms (see  $t_{WB}$  in [Table 14-10](#)). Therefore, the designer must account for this amount of delay before another page of memory is written. Alternatively, the designer can check the status register's write in progress bit by executing the read status operation while the self-timed write cycle is in progress. The write in progress bit is set to 1 during the self-timed write cycle, and is 0 when it is complete.

Figure 14–13. Write Bytes Operation Timing Diagram



**Note to Figure 14–13:**

- (1) Address bits A[23..19] are don't care bits in the EPCS4 device. Address bits A[23..17] are don't care bits in the EPCS1 device.

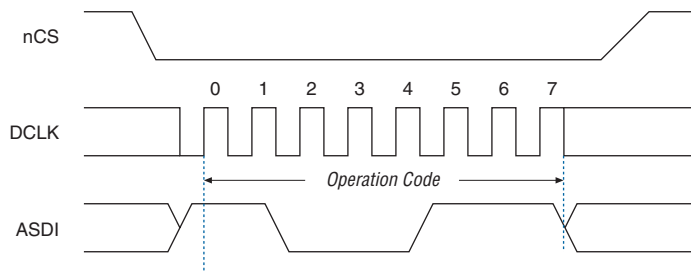
### Erase Bulk Operation

The erase bulk operation code is  $b'1100\ 0111$ , with the MSB listed first. The erase bulk operation sets all memory bits to 1 or  $0xFF$ . Similar to the write bytes operation, the write enable operation must be executed prior to the erase bulk operation so that the write enable latch bit in the status register is set to 1.

Designers implement the erase bulk operation by driving  $nCS$  low and then shifting in the erase bulk operation code on the ASDI pin.  $nCS$  must be driven high after the eighth bit of the erase bulk operation code has been latched in. Figure 14–14 shows the timing diagram.

The device initiates the self-timed erase bulk cycle immediately after  $nCS$  is driven high. The self-timed erase bulk cycle usually takes 5 s for EPCS4 devices (guaranteed to be less than 10 s) or 3 s for EPCS1 devices (guaranteed to be less than 6 s). See  $t_{EB}$  in Table 14–10. Designers must account for this delay before accessing the memory contents. Alternatively, designers can check the write in progress bit in the status register by executing the read status operation while the self-timed erase cycle is in progress. The write in progress bit is 1 during the self-timed erase cycle and is 0 when it is complete. The write enable latch bit in the status register is reset to 0 before the erase cycle is complete.

**Figure 14–14. Erase Bulk Operation Timing Diagram**



### Erase Sector Operation

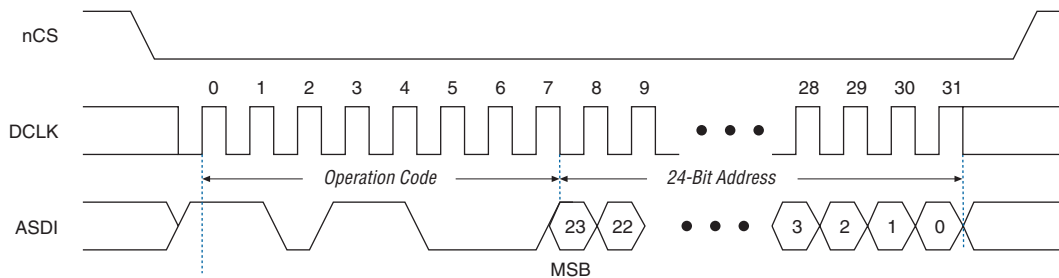
The erase sector operation code is  $b'1101\ 1000$ , with the MSB listed first. The erase sector operation allows the user to erase a certain sector in the serial configuration device by setting all bits inside the sector to 1 or  $0xFF$ . This operation is useful for users who access the unused sectors as general purpose memory in their applications.

The write enable operation must be executed prior to the erase sector operation so that the write enable latch bit in the status register is set to 1.

The erase sector operation is implemented by first driving  $nCS$  low, then shifting in the erase sector operation code and the three address bytes of the chosen sector on the ASDI pin. The three address bytes for the erase sector operation can be any address inside the specified sector. (See [Tables 14-4](#) and [14-5](#) for sector address range information.) Drive  $nCS$  high after the eighth bit of the erase sector operation code has been latched in. [Figure 14-15](#) shows the timing diagram.

Immediately after the device drives  $nCS$  high, the self-timed erase sector cycle is initiated. The self-timed erase sector cycle usually takes 2 s for EPCS1 and EPCS4 devices and is guaranteed to be less than 3 s for both serial configuration devices. You must account for this amount of delay before the memory contents can be accessed. Alternatively, you can check the write in progress bit in the status register by executing the read status operation while the erase cycle is in progress. The write in progress bit is 1 during the self-timed erase cycle and is 0 when it is complete. The write enable latch bit in the status register is reset to 0 before the erase cycle is complete.

**Figure 14-15. Erase Sector Operation Timing Diagram**



## Power & Operation

This section describes the power modes, power-on reset (POR) delay, error detection, and initial programming state of serial configuration devices.

### Power Mode

Serial configuration devices support active power and standby power modes. When  $nCS$  is low, the device is enabled and is in active power mode. The Cyclone FPGA is configured while in active power mode. When  $nCS$  is high, the device is disabled but could remain in active power mode until all internal cycles have completed (such as write or erase operations). The serial configuration device then goes into stand-by



power mode. The  $I_{CC1}$  parameter specifies the  $V_{CC}$  supply current when the device is in active power mode and the  $I_{CC0}$  parameter specifies the current when the device is in stand-by power mode (see Table 14-16).

## Power-On Reset

During initial power-up, a POR delay occurs to ensure the system voltage levels have stabilized. During AS configuration, the Cyclone FPGA controls the configuration and has a longer POR delay than the serial configuration device. Therefore, the POR delay is governed by the Cyclone FPGA (typically 100 ms).

## Error Detection

During AS configuration with the serial configuration device, the Cyclone FPGA monitors the configuration status through the  $nSTATUS$  and  $CONF\_DONE$  pins. If an error condition occurs ( $nSTATUS$  drives low) or if the  $CONF\_DONE$  pin does not go high, the Cyclone FPGA will initiate reconfiguration by pulsing the  $nSTATUS$  and  $nCSO$  signals, which controls the chip select pin on the serial configuration device ( $nCS$ ).

After an error, configuration automatically restarts if the *Auto-Restart Upon Frame Error* option is turned on in the Quartus II software. If the option is turned off, the system must monitor the  $nSTATUS$  signal for errors and then pulse the  $nCONFIG$  signal low to restart configuration.

## Timing Information

Figure 14-16 shows the timing waveform for write operation to the serial configuration device.

Figure 14-16. Write Operation Timing

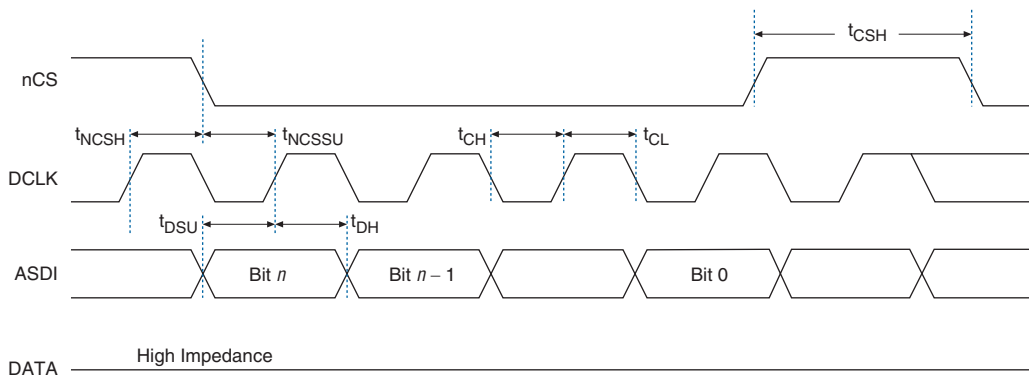


Table 14–10 defines the serial configuration device timing parameters for write operation.

Symbol	Parameter	Min	Max	Unit
$f_{WCLK}$	Write clock frequency (from Cyclone FPGA, ByteBlaster II cable or embedded processor) for write enable, write disable, read status, read silicon ID, write bytes, erase bulk, and erase sector operations		25	MHz
$t_{CH}$	DCLK high time	20		ns
$t_{CL}$	DCLK low time	20		ns
$t_{NCSSU}$	Chip select ( $nCS$ ) setup time	10		ns
$t_{NCSH}$	Chip select ( $nCS$ ) hold time	10		ns
$t_{DSU}$	Data (ASDI) in setup time before rising edge on DCLK	5		ns
$t_{DH}$	Data (ASDI) hold time after rising edge on DCLK	5		ns
$t_{CSH}$	Chip select high time	100		ns
$t_{WB\_EPCS1}$ (1)	Write bytes cycle time for EPCS1 devices	2	5	ms
$t_{WB\_EPCS4}$ (1)	Write bytes cycle time for EPCS4 devices	1.5	5	ms
$t_{WS}$ (1)	Write status cycle time	5	15	ms
$t_{EB\_EPCS1}$ (1)	Erase bulk cycle time for EPCS1 devices	3	6	s
$t_{EB\_EPCS4}$ (1)	Erase bulk cycle time for EPCS4 devices	5	10	s
$t_{ES}$ (1)	Erase sector cycle time	2	3	s

Note to Table 14–10:

(1) These parameters are not shown in Figure 14–16.

Figure 14–17 shows the timing waveform for the serial configuration device's read operation.

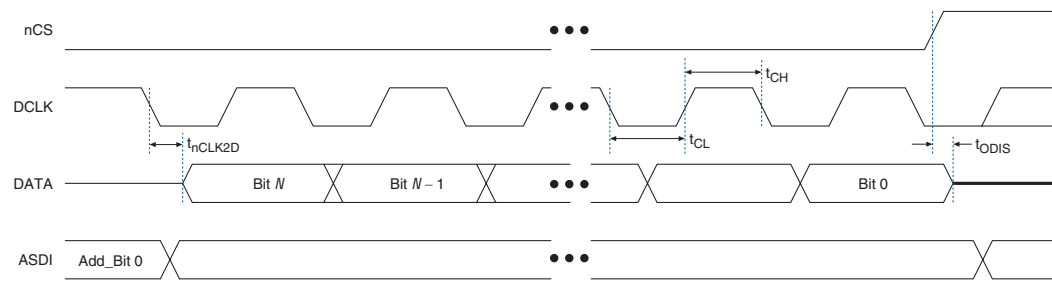
**Figure 14–17. Read Operation Timing**

Table 14–11 defines the serial configuration device timing parameters for read operation.

Symbol	Parameter	Min	Max	Unit
$f_{RCLK}$	Read clock frequency (from Cyclone FPGA or embedded processor) for read bytes operation		20	MHz
$t_{CH}$	DCLK high time	25		ns
$t_{CL}$	DCLK low time	25		ns
$t_{ODIS}$	Output disable time after read		15	ns
$t_{nCLK2D}$	Clock falling edge to data		15	ns

Figure 14–18 shows the timing waveform for Cyclone FPGA AS configuration scheme using a serial configuration device.

**Figure 14–18. AS Configuration Timing**

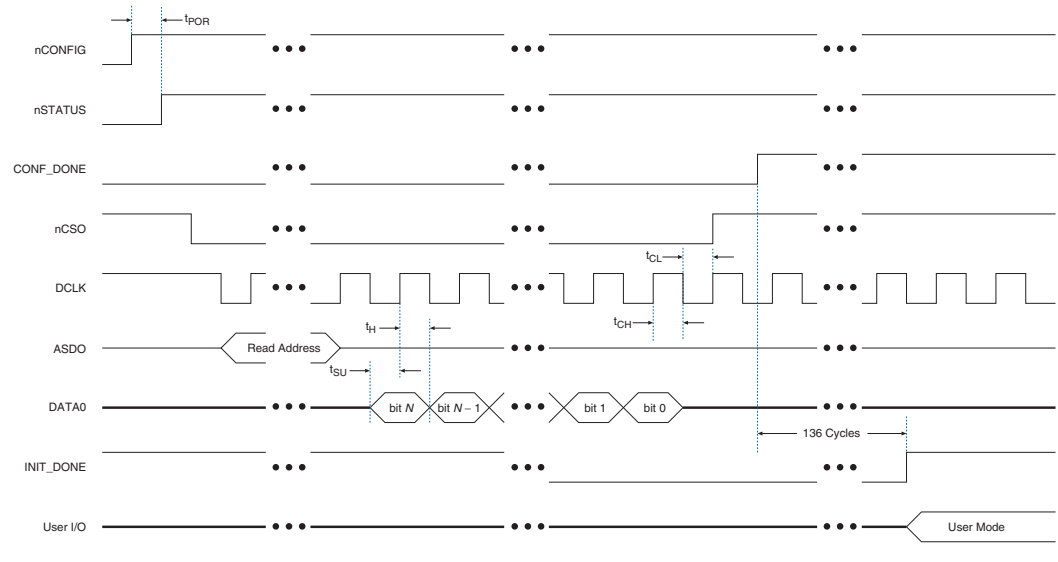


Table 14–12 shows the timing parameters for AS configuration mode.

**Table 14–12. Timing Parameters for AS Configuration**

Symbol	Parameter	Min	Typ	Max	Unit
$f_{CLK}$	DCLK frequency (from Cyclone FPGA)		15	20	MHz
$t_{CH}$	DCLK high time	25			ns
$t_{CL}$	DCLK low time	25			ns
$t_H$	Data hold time after rising edge on DCLK	0			ns
$t_{SU}$	Data set up time before rising edge on DCLK	5			ns
$t_{POR}$	POR delay			100	ms

## Programming & Configuration File Support

The Quartus II design software provides programming support for serial configuration devices. After selecting the serial configuration device, the Quartus II software automatically generates the Programmer Object File (.pof) to program the device. The software allows users to select the appropriate serial configuration device density that most efficiently stores the configuration data for a selected Cyclone FPGA.

The serial configuration device can be programmed in-system by an external microprocessor using SRRunner. SRRunner is a software driver developed for embedded serial configuration device programming that designers can customize to fit in different embedded systems. The SRRunner can read a Raw Programming Data file (.rpd) and write to the serial configuration devices. The programming time is comparable to the Quartus II software programming time.



For more information about SRRunner, see the *SRRunner: An Embedded Solution for Serial Configuration Device Programming White Paper* and the source code on the Altera web site ([www.altera.com](http://www.altera.com)).

Serial configuration devices can be programmed using the APU with the appropriate programming adapter (PLMSEPC-8) via the Quartus II software or the ByteBlaster II download cable via the Quartus II software. In addition, many third-party programmers, such as BP Microsystems and System General, offer programming hardware that supports serial configuration devices.

During in-system programming of a serial configuration device via the ByteBlaster II download cable, the cable pulls nCONFIG low to reset the Cyclone device and overrides the 10-k $\Omega$  pull-down resistor on the Cyclone device's nCE pin (see [Figure 14-2](#)). The download cable then uses the four interface pins (DATA, nCS, ASDI, and DCLK) to program the serial configuration device. Once the programming is complete, the download cable releases the serial configuration device's four interface pins and the Cyclone device's nCE pin, and pulses nCONFIG to start configuration.



For more information on programming and configuration support, see the following documents:

- *Altera Programming Hardware Data Sheet*
- *Programming Hardware Manufacturers*
- *ByteBlaster II Parallel Port Download Cable Data Sheet*

## Operating Conditions

[Tables 14-13](#) through [14-17](#) provide information on absolute maximum ratings, recommended operating conditions, DC operating conditions, and capacitance for serial configuration devices.

<i>Table 14-13. Absolute Maximum Ratings</i> <i>Note (1) (Part 1 of 2)</i>					
Symbol	Parameter	Condition	Min	Max	Unit
V <sub>CC</sub>	Supply voltage	With respect to ground	-0.6	4.0	V
V <sub>I</sub>	DC input voltage	With respect to ground	-0.6	4.0	V

**Table 14–13. Absolute Maximum Ratings** *Note (1) (Part 2 of 2)*

Symbol	Parameter	Condition	Min	Max	Unit
$I_{MAX}$	DC $V_{CC}$ or GND current			15	mA
$I_{OUT}$	DC output current per pin		-25	25	mA
$P_D$	Power dissipation			54	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 14–14. Recommended Operating Conditions**

Symbol	Parameter	Conditions	Min	Max	Unit
$V_{CC}$	Supply voltage	(2)	3.0	3.6	V
$V_I$	Input voltage	Respect to GND	-0.3	$0.3 + V_{CC}$	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			5	ns
$t_F$	Input fall time			5	ns

**Table 14–15. DC Operating Conditions**

Symbol	Parameter	Conditions	Min	Max	Unit
$V_{IH}$	High-level input voltage		$0.7 \times V_{CC}$	$V_{CC} + 0.4$	V
$V_{IL}$	Low-level input voltage		-0.5	$0.3 \times V_{CC}$	V
$V_{OH}$	High-level output voltage	$I_{OH} = -100 \mu A$ (3)	$V_{CC} - 0.2$		V
$V_{OL}$	Low-level output voltage	$I_{OL} = 1.6 \text{ mA}$ (3)		0.4	V
$I_I$	Input leakage current	$V_I = V_{CC}$ or GND	-10	10	$\mu A$
$I_{OZ}$	Tri-state output off-state current	$V_O = V_{CC}$ or GND	-10	10	$\mu A$

**Table 14–16.  $I_{CC}$  Supply Current**

Symbol	Parameter	Conditions	Min	Max	Unit
$I_{CC0}$	$V_{CC}$ supply current (standby)			50	$\mu A$
$I_{CC1}$	$V_{CC}$ supply current (during active power mode)		5	14	mA

**Table 14–17. Capacitance** Note (4)

Symbol	Parameter	Conditions	Min	Max	Unit
$C_{IN}$	Input pin capacitance	$V_{IN} = 0\text{ V}$		6	$\mu\text{F}$
$C_{OUT}$	Output pin capacitance	$V_{OUT} = 0\text{ V}$		8	$\mu\text{F}$

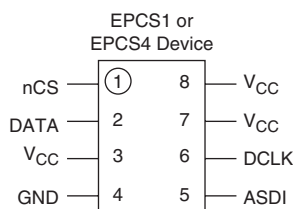
Notes to Table 14–13 through 14–17:

- (1) See the *Operating Requirements for Altera Devices Data Sheet*.
- (2) Maximum  $V_{CC}$  rise time is 100 ms.
- (3) The  $I_{OH}$  parameter refers to high-level TTL or CMOS output current; the  $I_{OL}$  parameter refers to low-level TTL or CMOS output current.
- (4) Capacitance is sample-tested only at  $T_A = 25\text{ }^\circ\text{C}$  and at a 20-MHz frequency.

## Pin Information

As shown in Figure 14–19, the serial configuration device is an 8-pin device. The control pins on the serial configuration device are: serial data output (DATA), active serial data input (ASDI), serial clock (DCLK), and chip select (nCS). Table 14–18 shows the serial configuration device's pin descriptions.

Figure 14–19 shows the Altera serial configuration device 8-pin SOIC package and its pin-out diagram.

**Figure 14–19. Altera Serial Configuration Device Package Pin-Out Diagram****Table 14–18. Serial Configuration Device Pin Description (Part 1 of 2)**

Pin Name	Pin Number	Pin Type	Description
DATA	2	Output	The DATA output signal transfers data serially out of the serial configuration device to the Cyclone FPGA during read/configuration operation. During a read/configuration operations, the serial configuration device is enabled by pulling nCS low. The DATA signal transitions on the falling edge of DCLK.
ASDI	5	Input	The AS data input signal is used to transfer data serially into the serial configuration device. It receives the data that should be programmed into the serial configuration device. Data is latched in the rising edge of DCLK.

**Table 14–18. Serial Configuration Device Pin Description (Part 2 of 2)**

Pin Name	Pin Number	Pin Type	Description
nCS	1	Input	The active low chip select input signal toggles at the beginning and end of a valid instruction. When this signal is high, the device is deselected and the DATA pin is tri-stated. When this signal is low, it enables the device and puts the device in an active mode. After power up, the serial configuration device requires a falling edge on the nCS signal before beginning any operation.
DCLK	6	Input	DCLK is provided by the Cyclone FPGA. This signal provides the timing of the serial interface. The data presented on ASDI is latched to the serial configuration device, at the rising edge of DCLK. Data on the DATA pin changes after the falling edge of DCLK and is latched into the Cyclone FPGA on the rising edge.
VCC	3, 7, 8	Power	Power pins connect to 3.3 V.
GND	4	Ground	Ground pin.

## Package

All serial configuration devices are available in 8-pin plastic SOIC package.



For more information on Altera device packaging, see [Chapter 6, Package Information for Cyclone Devices](#), including mechanical drawing and specifications for this package.

## Ordering Code

[Table 14–19](#) shows the ordering codes for serial configuration devices.

**Table 14–19. Serial Configuration Device Ordering Codes**

Device	Ordering Code
EPCS1	EPCS1SI8
EPCS4	EPCS4SI8



## 1.5-V Devices

- Board Layout 12-21
- Choose a Regulator Type 12-9
- Designing with 12-1
- Linear Voltage Regulators 12-4
- Maximum Output Current 12-8
- Power Sequencing & Hot Socketing 12-1
- Regulator Application Examples 12-19
- Regulator Circuits 12-10
- Selecting Voltage Regulators 12-8
- Split-Plane Method 12-23
- Switching Voltage Regulators 12-6
- Synchronous Switching Regulator Example 12-20
- Using MultiVolt I/O Pins 12-2
- Voltage Divider Network 12-10
- Voltage Regulators 12-3

## A

### Architecture

- addsub Signal 2-7
- Bus Hold 2-51
- Byte Enables 2-23
- Carry-Select Chain 2-10
- Clear & Preset Logic Control 2-12
- Clock
  - Clock Feedback 2-37
  - Clock Mode
    - Independent 2-25
    - Input/Output 2-25
    - Read/Write 2-27
  - Clock Multiplication & Division 2-35
  - Dual-Purpose Clock Pins 2-30
  - External Clock Inputs 2-36
  - External Clock Outputs 2-36
  - Global Clock Network 2-29
  - Global Clock Network & Phase-Locked Loops 2-29
  - Maximum Input & Output Clock Rates 4-29

- Phase Shifting 2-37
- Combined Resources 2-31
- Configuration
  - 3-5
  - Schemes 3-6
  - Testing 3-1
- Control Signals 2-38
- Control Signals & M4K Interface 2-23
- Cyclone Architecture 2-1
- DC & Switching Characteristics 4-1
- Device Pin-Outs 5-1
- Dynamic Arithmetic Mode 2-9
- Embedded Memory 2-18
- Functional Description 2-1
- I/O Standard
  - Advanced I/O Standard Support 2-52
  - External I/O Delay Parameters 4-23
  - I/O Structure 2-39
  - LVDS I/O Pins 2-54
  - MultiVolt I/O Interface 2-54
- IEEE Std. 1149.1 (JTAG) Boundary Scan Support 3-1
- LAB
  - Control Signals 2-4
  - Interconnects 2-3
- Lock Detect Signal 2-37
- Logic Array Blocks 2-3
- Logic Elements
  - 2-5
  - Operating Modes 2-7
- LUT Chain & Register Chain 2-7
- Memory
  - Configuration Sizes 2-21
  - DDR SDRAM & FCRAM 2-46
  - External RAM Interfacing 2-46
  - Modes 2-18
- MultiTrack Interconnect 2-12
- Normal Mode 2-8
- Open-Drain Output 2-50
- Operating Conditions 4-1
- Operating Modes 3-6

- Ordering Information 5-1
- Parity Bit Support 2-20
- PLLs 2-32
- Power Consumption 4-8
- Power Sequencing & Hot Socketing 2-55
- Programmable Drive Strength 2-49
- Programmable Duty Cycle 2-38
- Programmable Pull-Up Resistor 2-51
- Reference & Ordering Information 5-1
- Shift Register Support 2-20
- SignalTap II Embedded Logic Analyzer 3-5
- Single-Port Mode 2-28
- Slew-Rate Control 2-50
- Software 5-1
- Timing
  - External Timing Parameters 4-16
  - Internal Timing Parameters 4-11
  - Model 4-9
  - Preliminary & Final 4-9
- STAPL 13-36
- JTAG Configuration of Multiple Devices 13-32
- JTAG Configuration Using a Download Cable 13-29
- JTAG-Based Configuration 13-28
- Passive Serial 13-15
- Passive Serial & JTAG 13-44
- Program Flow 13-36
- Programming Configuration Devices 13-20
- Programming Serial Configuration Devices 13-13
- PS Configuration from a Microprocessor 13-25
- PS Configuration Using a Download Cable 13-21
- PS Configuration using Configuration Device 13-15
- Quartus II Software
  - Board Layout Tips 13-55
  - Configuration Reliability 13-55
  - Hexadecimal (Intel-Format) File (.hex) 13-54
  - Jam Byte-Code File (.jbc) 13-55
  - Jam File (.jam) 13-54
  - Programmer Object File (.pof) 13-54
  - Raw Binary File (.rbf) 13-54
  - SRAM Object File (.sof) 13-53
  - Tabular Text File (.tff) 13-54
- Schemes 13-6
- Timing
  - Passive Serial Timing 13-26

## C

### Configuration

- Active Serial & JTAG 13-43
- Active Serial Configuration (Serial Configuration Devices) 13-7
- Combining Configuration Schemes 13-43
- Configuring Cyclone FPGAs 13-1
- FPGAs with JRunner 13-35
- FPGAs with the MicroBlaster Software 13-26
- Configuring Multiple Cyclone FPGAs 13-18
- Configuring Multiple Devices (Cascading) 13-10
- Connecting the JTAG Chain to the Embedded Processor 13-34
- Data Compression 13-3
- Device Configuration
  - Overview 13-1
- Device Configuration Files 13-53
- Device Options 13-45
- Jam
  - Example Jam File that Reads the IDCODE 13-42
  - Instructions 13-41

## D

- Device Configuration Pins 13-49

## I

### I/O Standards

- 11-1
- 1.5-V
  - LVC MOS Normal & Wide Voltage Ranges (EIA/JEDEC Standard JESD8-11) 8-5
- 1.8-V

- LVC MOS Normal & Wide Voltage Ranges (EIA/JEDEC Standard EIA/JESD8-7) 8-4
- LVTTL Normal & Wide Voltage Ranges (EIA/JEDEC Standard EIA/JESD8-7) 8-4
- 2.5-V
  - LVC MOS Normal & Wide Voltage Ranges (EIA/JEDEC Standard EIA/JESD8-5) 8-4
  - LVTTL Normal & Wide Voltage Ranges (EIA/JEDEC Standard EIA/JESD8-5) 8-3
- 3.3-V
  - (PCI Special Interest Group (SIG) PCI Local Bus Specification Revision 2.2) 8-5
  - LVC MOS (EIA/JEDEC Standard JESD8-B) 8-3
  - LVTTL (EIA/JEDEC Standard JESD8-B) 8-2
- Bidirectional Pads 8-15
- Cyclone I/O Banks 8-9
- DC Guidelines 8-17
- Differential I/O Standard Termination 8-14
- Differential Pad Placement Guidelines 8-14
- Differential SSTL-2 - EIA/JEDEC Standard JESD8-9A 8-9
- Hot Socketing 8-13
- I/O Termination 8-13
- Input Pads 8-15
- LVDS (ANSI/TIA/EIA Standard ANSI/TIA/EIA-644) 8-8
- Output Pads 8-15
- Pad Placement & DC Guidelines 8-14
- Programmable Current Drive Strength 8-12
- Quartus II Software
  - Assigning Pins 8-19
  - Auto Placement & Verification of Selectable I/O Standards 8-21
  - Compiler Settings 8-18
  - Device & Pin Options 8-18
  - I/O Banks in the Floorplan View 8-20
  - Programmable Drive Strength Settings 8-20
  - Software Support 8-18

- SSTL-2 Class I & II (EIA/JEDEC Standard JESD8-9A) 8-7
- SSTL-3 Class I & II (EIA/JEDEC Standard JESD8-8) 8-7
- Supported I/O Standards 8-2
- Using Selectable I/O Standards in Cyclone Devices 8-1
- Voltage-Referenced I/O Standard Termination 8-14
- Voltages
  - 5.0-V Device Compatibility 11-3
  - Devices Can Be Driven before Power-Up 11-6
  - Hot-Socketing 11-6
  - I/O Pins Remain Tri-States during Power-Up 11-6
  - MultiVolt I/O Operation 11-2
  - Power-On Reset 11-7
  - Power-Up Sequence 11-7
  - Signal Pins Do Not Drive the VCCIO or VCCINT Power Supplies 11-6
- VREF Pad Placement Guidelines 8-14

## L

### LVDS

- Clock Domains 9-3
- Cyclone I/O Banks 9-2
- Cyclone I/O Interface 9-3
- Cyclone Receiver & Transmitter Termination 9-15
- Implementing LVDS in Cyclone Devices 9-1
- Quartus II Software
  - Board Design Considerations 9-26
  - Capturing Serial Data on Cyclone LVDS Inputs 9-18
  - Design Guidelines 9-25
  - Differential Pad Placement Guidelines 9-25
  - Implementing Cyclone LVDS I/O Pins in the Quartus II Software 9-16
  - Receiver Circuit 9-19
  - Transmitter Circuit 9-17
  - Transmitting Serial Data on Cyclone LVDS Outputs 9-16
- Receiver & Transmitter 9-4

Timing in Cyclone Devices 9–10

## M

### Memory

Bidirectional Double Data Rate 10–3  
 DDR Memory Support 10–4  
 Double Data Rate  
   Input 10–1  
   Output 10–2  
 Implementing Double Data Rate I/O Signal-  
 ing in Cyclone Devices 10–1

## P

### PLL

altpll  
   Compilation Report 6–31  
   Input Ports 6–22  
   MegaWizard Customization 6–23  
   MegaWizard Page Description 6–25  
   Output Ports 6–23  
   Simulation 6–37  
   Timing Analysis 6–33  
 areset 6–12  
 Board Layout 6–17  
 Clock  
   Combined Sources 6–41  
   Dedicated Clock Input Pins 6–40  
   Dual-Purpose Clock I/O Pins 6–40  
   External Clock Output 6–11  
   Feedback Modes 6–13  
   Global Clock Network 6–38  
   Multiplication & Division 6–8  
 Control Signals 6–12  
 Cyclone PLL Blocks 6–2  
 Hardware Features 6–8  
 Hardware Overview 6–1  
 Jitter Considerations 6–19  
 Modes  
   No Compensation 6–15  
   Normal 6–13

Zero Delay Buffer 6–14  
 Partitioned VCCA Island within VCCINT  
   Plane 6–17  
 pfdena 6–12  
 Phase Shifting 6–9  
 Pins 6–16  
 Pins & Clock Network Connections 6–6  
 pllana 6–12  
 Programmable Duty Cycle 6–10  
 Quartus II altpll Megafunction 6–20  
 Separate VCCA Power Plane 6–17  
 Software Support 6–20  
 Specifications 6–20  
 Thick VCCA Traces 6–18  
 Using PLLs in Cyclone Devices 6–1  
 VCCA & GNDA 6–17

## S

### Serial Configuration Devices

(EPCS1 & EPCS4) Data Sheet 14–1  
 Accessing Memory 14–3  
 Cyclone FPGA Configuration 14–4  
 Error Detection 14–23  
 Features 14–1  
 Functional Description 14–1  
 Operating Conditions 14–27  
 Ordering Code 14–30  
 Package 14–30  
 Pin Description 14–29  
 Power & Operation 14–22  
 Power Mode 14–22  
 Power-On Reset 14–23  
 Programming & Configuration File  
   Support 14–26

**Software Overview** 6–4

## U

**Using Cyclone Devices in Multiple-Voltage  
 Systems** 11–1

**Results For:** EP1C3T100

4 part numbers found and 0 obsolete part numbers found

Cyclone Device Family (1.5V)					
<a href="#">Cyclone Datasheet</a> <a href="#">Part Number Format</a>			<a href="#">Cyclone Literature</a> <a href="#">Buying Altera Devices</a>		
Part Number	Device	Pin & Package	Temperature	Speeds	Options
EP1C3T100C6 EP1C3T100C7 EP1C3T100C8	EP1C3	<a href="#">100 pin TQFP</a>	Commercial ( 0 to 85°C)	6, 7, 8	None
EP1C3T100I7	EP1C3	<a href="#">100 pin TQFP</a>	Industrial ( -40 to 100°C)	7	None

[Advanced](#)   [Help](#)

[Please Give Us Feedback](#)  
[Sign Up for E-mail Updates](#)

Choose Your Device

[FPGA vs ASIC](#)

[Calculator](#)

[Device Family](#)

[Overview](#)

[Performance](#)

[Advantages](#)

FPGAs

- [Stratix II](#)
- [Stratix](#)
- [Cyclone II](#)
- [Cyclone](#)
- [Stratix GX](#)
- [APEX II](#)
- [APEX 20K](#)
- [Mercury](#)
- [FLEX 10K](#)
- [ACEX 1K](#)
- [FLEX 6000](#)

CPLDs

- [MAX II](#)
- [MAX 3000A](#)
- [MAX 7000](#)

Structured ASICs

- [About HardCopy](#)
- [HardCopy Stratix](#)
- [HardCopy APEX 20K](#)

[Home](#) > [Products](#) > [Devices](#) > Industrial



[Print This Page](#)

[E-mail This Page](#)

## Industrial Temperature Device Support

Altera is committed to supporting the industrial device needs of its customers. The industrial temperature range guarantees a device to be fully functional from -40°C to +100°C (junction) [-40°C to +105°C (junction) for MAX devices]. These devices go through extensive product characterization and reliability stresses to ensure functionality under extreme temperature variations.

Altera offers a wide range of industrial devices in all device families. Table 1 shows the products currently offered in the industrial temperature range. For additional details regarding industrial temperature offerings not shown below, contact your [local sales office or distributor](#).

*Table 1. Industrial Temperature Device Offerings (Click on the product name for more details.)*

Supply Voltage	Device								
	MAX <sup>®</sup> II	Stratix <sup>™</sup>	Cyclone <sup>™</sup>	APEX <sup>™</sup>	Mercury <sup>™</sup>	FLEX <sup>®</sup>	ACEX <sup>®</sup>	MAX	Configuration Devices
1.5 V		<a href="#">Stratix</a>	<a href="#">Cyclone</a>	<a href="#">APEX II</a>					
1.8 V	<a href="#">MAX IIG</a>			<a href="#">APEX 20KC</a> <a href="#">APEX 20KE</a>	<a href="#">Mercury</a>				
2.5 V	<a href="#">MAX II</a>			<a href="#">APEX 20K</a>		<a href="#">FLEX 10KE</a>	<a href="#">ACEX 1K</a>	<a href="#">MAX 7000B</a>	

Configuration Devices

[Enhanced Configuration](#)

[Serial Configuration](#)

Embedded Processors

[About Excalibur](#)

[Excalibur Devices](#)

Market-Specific Offerings

[Lead-Free](#)

[Extended Temperature](#)

[Industrial](#)

Mature Devices

[FLEX 8000](#)

[MAX 9000](#)

[Classic](#)

<b>3.3 V</b>	<a href="#">MAX II</a>					<a href="#">FLEX 10KA</a> <a href="#">FLEX 6000</a>	<a href="#">MAX 7000A</a> <a href="#">MAX 3000A</a>	<a href="#">440 Kbit</a> <a href="#">1 Mbit</a> <a href="#">2 Mbit</a> <a href="#">4 Mbit</a> <a href="#">8 Mbit</a> <a href="#">16 Mbit</a>
<b>5.0 V</b>						<a href="#">FLEX 10K®</a> <a href="#">FLEX 8000</a>	<a href="#">MAX 7000</a> <a href="#">MAX 7000S</a> <a href="#">MAX 9000</a> <a href="#">Classic</a>	<a href="#">65 Kbit</a> <a href="#">200 Kbit</a> <a href="#">440 Kbit</a> <a href="#">1 Mbit</a> <a href="#">2 Mbit</a>

[Back to Top](#)

*Table 2. MAX II Industrial Support (Coming Soon!)*

Device	Package (1)	Speed Grade
<b>EPM240</b> <b>EPM240G</b>	100-pin TQFP	-5
<b>EPM570</b> <b>EPM570G</b>	100-pin TQFP 144-pin TQFP 256-pin FineLine BGA®	-5
<b>EPM1270</b> <b>EPM1270G</b>	144-pin TQFP 256-pin FineLine BGA	-5
<b>EPM2210</b> <b>EPM2210G</b>	256-pin FineLine BGA 324-pin FineLine BGA	-5

[Back to Top](#)

*Table 3. Stratix Industrial Support*

Device	Package (1)	Speed Grade
<b>EP1S10</b>	484-pin FineLine BGA 672-pin FineLine BGA 780-pin FineLine BGA	-6 -7 -6

<b>EP1S20</b>	484-pin FineLine BGA	-6
	672-pin FineLine BGA	-7
	780-pin FineLine BGA	-6
<b>EP1S25</b>	672-pin FineLine BGA	-7
	780-pin FineLine BGA	-6
	1,020-pin FineLine BGA	-6
<b>EP1S30</b>	1,020-pin FineLine BGA	-6
<b>EP1S40</b>	1,020-pin FineLine BGA	-6
<b>EP1S60</b>	1,020-pin FineLine BGA	-6
<b>EP1S80</b>	1,020-pin FineLine BGA	-7

[Back to Top](#)

*Table 4. Cyclone Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP1C3</b>	100-pin TQFP 144-pin TQFP	-7
<b>EP1C4</b>	324-pin FineLine BGA 400-pin FineLine BGA	-7
<b>EP1C6</b>	144-pin TQFP 240-pin PQFP 256-pin FineLine BGA	-7
<b>EP1C12</b>	240-pin PQFP 256-pin FineLine BGA 324-pin FineLine BGA	-7
<b>EP1C20</b>	324-pin FineLine BGA 400-pin FineLine BGA	-7

[Back to Top](#)

*Table 5. APEX II Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP2A15</b>	672-pin FineLine BGA	-8



<b>EP2A25</b>	672-pin FineLine BGA 724-pin BGA	-8
<b>EP2A40</b>	724-pin BGA 1,020-pin FineLine BGA	-8

[Back to Top](#)

*Table 6. APEX 20KC Industrial Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP20K200C</b>	484-pin FineLine BGA	-8
<b>EP20K400C</b>	652-pin BGA 672-pin FineLine BGA	-8
<b>EP20K600C</b>	652-pin BGA 672-pin FineLine BGA	-8
<b>EP20K1000C</b>	1,020-pin FineLine BGA	-8

[Back to Top](#)

*Table 7. APEX 20KE Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP20K30E</b>	144-pin FineLine BGA	-2X
<b>EP20K60E</b>	324-pin FineLine BGA	-2X
<b>EP20K60E</b>	144-pin FineLine BGA 324-pin FineLine BGA 208-pin PQFP	-2X
<b>EP20K100E</b>	144-pin FineLine BGA 324-pin FineLine BGA 356-pin BGA 240-pin PQFP	-2X
<b>EP20K160E</b>	484-pin FineLine BGA	-2X
<b>EP20K200E</b>	240-pin PQFP	-2

<b>EP20K200E</b>	484-pin FineLine BGA 672-pin FineLine BGA 356-pin BGA 240-pin PQFP	-2X
<b>EP20K300E</b>	672-pin FineLine BGA 652-pin BGA 240-pin PQFP	-2X
<b>EP20K400E</b>	652-pin BGA 672-pin FineLine BGA	-2X
<b>EP20K600E</b>	652-pin BGA 672-pin FineLine BGA	-2

[Back to Top](#)

**Table 8. APEX 20K Industrial Temperature Support**

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP20K100</b>	208-pin PQFP 240-pin PQFP	-2V
<b>EP20K100</b>	208-pin PQFP 240-pin PQFP	-2
<b>EP20K200</b>	240-pin PQFP 484-pin FineLine BGA	-2V
<b>EP20K200</b>	240-pin PQFP	-2
<b>EP20K400</b>	652-pin BGA	-2
<b>EP20K400</b>	672-pin FineLine BGA	-2V

[Back to Top](#)

**Table 9. Mercury Industrial Temperature Support**

<b>Device</b>	<b>Package(1)</b>	<b>Speed Grade</b>
<b>EP1M120</b>	484-pin FineLine BGA	-6
<b>EP1M350</b>	780-pin FineLine BGA	-6

[Back to Top](#)

*Table 10. FLEX 10KE Industrial Temperature Support*

Device	Package (1)	Speed Grade
<b>EPF10K30E</b>	144-pin TQFP 208-pin PQFP 256-pin FineLine BGA	-2
<b>EPF10K50E</b>	144-pin TQFP, 240-pin PQFP 256-pin FineLine BGA	-2
<b>EPF10K50S</b>	208-pin PQFP 484-pin FineLine BGA	-2
<b>EPF10K100E</b>	208-pin PQFP 256-pin FineLine BGA 484-pin FineLine BGA	-2
<b>EPF10K130E</b>	240-pin PQFP 356-pin BGA 484-pin FineLine BGA	-2
<b>EPF10K200E</b>	600-pin BGA	-2
<b>EPF10K200S</b>	356-pin BGA 672-pin FineLine BGA	-2

[Back to Top](#)

*Table 11. FLEX 10KA Industrial Temperature Support*

Device	Package (1)	Speed Grade
<b>EPF10K10A</b>	144-pin TQFP 208-pin PQFP	-3
<b>EPF10K30A</b>	256-pin FineLine BGA	-2
	144-pin TQFP 208-pin PQFP 356-pin BGA	-3

<b>EPF10K50V</b>	240-pin PQFP	-2
	240-pin PQFP 356-pin BGA	-3
	240-pin PQFP 600-pin BGA	-4
<b>EPF10K100A</b>	240-pin PQFP 356-pin BGA	-3

[Back to Top](#)

*Table 12. FLEX 6000 Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPF6010A</b>	100-pin TQFP	-2
<b>EPF6016</b>	144-pin TQFP 208-pin PQFP	-2
	144-pin TQFP	-3
<b>EPF6016A</b>	144-pin TQFP 208-pin PQFP	-3
<b>EPF6024A</b>	208-pin PQFP 256-pin BGA 256-pin FineLine BGA	-2

[Back to Top](#)

*Table 13. FLEX 10K Industrial Temperature*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPF10K10</b>	84-pin PLCC 144-pin TQFP 208-pin PQFP	-4
<b>EPF10K20</b>	144-pin TQFP 208-pin PQFP 240-pin PQFP	-4

<b>EPF10K30</b>	208-pin PQFP 240-pin PQFP	-4
<b>EPF10K50</b>	240-pin PQFP	-4

[Back to Top](#)

*Table 14. FLEX 8000 Industrial Temperature*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPF81188A</b>	208-pin PQFP	-4
<b>EPF81188A</b>	208-pin PQFP	-3
<b>EPF81188A</b>	240-pin RQFP	-4
<b>EPF81500A</b>	240-pin RQFP	-3
<b>EPF8282A</b>	84-pin PLCC	-4
<b>EPF8282A</b>	100-pin TQFP	-3
<b>EPF8452A</b>	84-pin PLCC	-4
<b>EPF8452A</b>	160-pin RQFP	-3
<b>EPF8636A</b>	84-pin PLCC	-4
<b>EPF8820A</b>	208-pin RQFP	-4

[Back to Top](#)

*Table 15. ACEX 1K Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP1K10</b>	100-pin TQFP 256-pin FineLine BGA	-2
<b>EP1K30</b>	144-pin TQFP 256-pin FineLine BGA	-2
<b>EP1K50</b>	144-pin TQFP, 208-pin PQFP 256-pin FineLine BGA	-2

<b>EP1K100</b>	208-pin PQFP 256-pin FineLine BGA 484-pin FineLine BGA	-2
----------------	--	----

[Back to Top](#)

*Table 16. MAX 7000B Industrial Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPM7032B</b>	44-pin TQFP	-5
<b>EPM7064B</b>	44-pin TQFP 100-pin TQFP	-5
<b>EPM7128B</b>	100-pin FineLine BGA 100-pin TQFP 256-pin FineLine BGA	-7
<b>EPM7256B</b>	100-pin TQFP 208-pin PQFP 256-pin FineLine BGA	-7

[Back to Top](#)

*Table 17. MAX 7000A Industrial Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPM7032AE</b>	44-pin TQFP	-7
<b>EPM7064AE</b>	44-pin PLCC 44-pin TQFP 100-pin TQFP	-7
<b>EPM7128AE</b>	100-pin FineLine BGA 100-pin TQFP 144-pin TQFP	-7
<b>EPM7128A</b>	100-pin TQFP 144-pin TQFP	-10

<b>EPM7256AE</b>	100-pin FineLine BGA 100-pin TQFP 144-pin TQFP 208-pin PQFP 256-pin FineLine BGA	-7
<b>EPM7256A</b>	144-pin TQFP 208-pin PQFP 256-pin BGA	-10
<b>EPM7512AE</b>	208-pin PQFP 256-pin FineLine BGA 256-pin BGA	-10

[Back to Top](#)

**Table 18. MAX 3000A Industrial Temperature Support**

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPM3032A</b>	44-pin PLCC 44-pin TQFP	-10
<b>EPM3064A</b>	44-pin PLCC 44-pin TQFP 100-pin TQFP	-10
<b>EPM3128A</b>	100-pin TQFP 144-pin TQFP 256-pin FineLine BGA	-10
<b>EPM3256A</b>	144-pin TQFP 208-pin PQFP 256-pin FineLine BGA	-10
<b>EPM3512A</b>	208-pin PQFP 256-pin FineLine BGA	-10

[Back to Top](#)

**Table 19. MAX 7000 Industrial Temperature Support**

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
---------------	--------------------	--------------------

<b>EPM7032</b>	44-pin PLCC 44-pin PQFP	-12
	44-pin PLCC 44-pin PQFP 44-pin TQFP	-15
<b>EPM7064</b>	44-pin PLCC, 68-pin PLCC 84-pin PLCC 100-pin PQFP	-15
<b>EPM7096</b>	68-pin PLCC 84-pin PLCC 100-pin PQFP	-15
<b>EPM7128E</b>	100-pin PQFP	-15
	84-pin PLCC 100-pin PQFP	-20
<b>EPM7160E</b>	100-pin PQFP 160-pin PQFP	-15
	84-pin PLCC 160-pin PQFP	-20
<b>EPM7192E</b>	160-pin BGA	-20
<b>EPM7256E</b>	192-pin PGA 208-pin RQFP	-20

[Back to Top](#)

*Table 20. MAX 7000S Industrial Temperature Support*

Device	Package (1)	Speed Grade
<b>EPM7032S</b>	44-pin PLCC 44-pin TQFP	-7
<b>EPM7064S</b>	44-pin PLCC 44-pin TQFP 84-pin PLCC 100-pin TQFP	-7



<b>EPM7128S</b>	84-pin PLCC 100-pin PQFP 100-pin TQFP 160-pin PQFP	-10
<b>EPM7160S</b>	84-pin PLCC 100-pin TQFP 160-pin PQFP	-10
<b>EPM7192S</b>	160-pin PQFP	-10
<b>EPM7256S</b>	208-pin RQFP	-10

[Back to Top](#)

*Table 21. MAX 9000 Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EPM9320</b>	84-pin PLCC 208-pin RQFP	-20
<b>EPM9320A</b>	84-pin PLCC 208-pin RQFP	-10
<b>EPM9560</b>	208-pin RQFP 240-pin RQFP 304-pin RQFP	-20
<b>EPM9560A</b>	208-pin RQFP 240-pin RQFP	-10

[Back to Top](#)

*Table 22. Classic Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>	<b>Speed Grade</b>
<b>EP610</b>	24-pin CerDip 24-pin PDIP	-30
<b>EP610I</b>	28-pin PLCC	-12

<b>EP910</b>	40-pin CerDIP 40-pin PDIP 44-pin PLCC	- 35
<b>EP910I</b>	40-pin CerDIP	-15
	44-pin PLCC	-12
<b>EP1810</b>	68-pin PLCC	-25, -45

[Back to Top](#)

*Table 23. Configuration Device Industrial Temperature Support*

<b>Device</b>	<b>Package (1)</b>
<b>EPC1064 (65 Kbit)</b>	8-pin PDIP 20-pin PLCC
<b>EPC1213 (200 Kbit)</b>	8-pin PDIP 20-pin PLCC
<b>EPC1441 (440 Kbit)</b>	8-pin PDIP 20-pin PLCC 32-pin TQFP
<b>EPCS1 (1 Mbit)</b>	8-pin SOIC
<b>EPC1 (1 Mbit)</b>	8-pin PDIP
	20-pin PLCC
<b>EPC2 (2 Mbit)</b>	20-pin PLCC
	32-pin TQFP
<b>EPCS4 (4 Mbit)</b>	8-pin SOIC
<b>EPC4 (4 Mbit)</b>	100-pin PQFP
<b>EPC8 (8 Mbit)</b>	100-pin PQFP
<b>EPC16 (16 Mbit)</b>	100-pin PQFP

**Notes:**

1. BGA: ball-grid array  
PDIP: plastic dual in-line package

[Please Give Us Feedback](#)

[Sign Up for E-mail](#)

[Updates](#)



PLCC: plastic J-lead chip carrier  
PQFP: plastic quad flat pack  
RQFP: power quad flat pack  
SOIC: small-outline integrated circuit  
TQFP: thin quad flat pack

[Back to Top](#)

## Related Links

- [Altera® Device Literature](#)
- [MAX II Devices](#)
- [Stratix Devices](#)
- [Cyclone Devices](#)
- [APEX II Devices](#)
- [Mercury Devices](#)
- [APEX 20K Devices](#)
- [FLEX 10K Devices](#)
- [FLEX 6000 Devices](#)
- [MAX 7000, MAX 7000A & MAX 7000B Devices](#)
- [MAX 9000 Devices](#)
- [Classic Devices](#)
- [Configuration Devices](#)

[Rate This Page](#)

---

[Home](#) | [Products](#) | [Support](#) | [System Solutions](#) | [Technology Center](#) | [Education & Events](#) | [Corporate](#) | [Buy On-Line Devices](#) | [Design Software](#) | [Intellectual Property](#) | [Design Services](#) | [Dev. Kits/Cables](#) | [Literature](#)

[Contact Us](#) | [New User](#) | [Site Map](#) | [Privacy](#) | [Legal Notice](#)

Copyright © 1995-2005 Altera Corporation, 101 Innovation Drive, San Jose, California 95134, USA