



FLEX 10K

Embedded Programmable Logic Family

July 1995, ver. 1

Data Sheet

Features...

Preliminary Information

- The industry's first embedded array programmable logic device family, providing system integration in a single device
 - Embedded array for implementing megafunctions, such as efficient memory and specialized logic functions
 - Logic array for general logic functions
- High density
 - 10,000 to 100,000 typical gates (see Table 1)
 - 720 to 5,392 registers
 - 6,144 to 24,576 RAM bits, all of which can be used without reducing logic capacity
- Flexible interconnect
 - FastTrack continuous routing structure for fast, predictable interconnect delays
 - Dedicated carry chain that implements arithmetic functions such as fast adders, counters, and comparators
 - Dedicated cascade chain that implements high-speed, high-fan-in logic functions
 - Tri-state emulation that implements internal tri-state nets
 - Up to 6 global Clock signals and 4 global Clear signals
- Powerful I/O pins
 - Individual tri-state control for each pin
 - Open-collector option on each I/O pin
 - Programmable output slew-rate control to reduce switching noise
 - Peripheral register for fast setup and Clock-to-output delay
- Fabricated on a 0.5-micron triple-layer metal SRAM process

Table 1. FLEX 10K Device Features

Feature	EPF10K10	EPF10K20	EPF10K30	EPF10K40	EPF10K50	EPF10K70	EPF10K100
Typical gates (logic & RAM)	10,000	20,000	30,000	40,000	50,000	70,000	100,000
Usable gates	7,000 to 31,000	15,000 to 63,000	22,000 to 69,000	29,000 to 93,000	36,000 to 116,000	46,000 to 118,000	62,000 to 158,000
Logic elements	576	1,152	1,728	2,304	2,880	3,744	4,992
Total RAM bits	6,144	12,288	12,288	16,384	20,480	18,432	24,576
Total registers	720	1,344	1,968	2,576	3,184	4,096	5,392
Maximum user I/O	150	198	248	278	310	358	406

...and More Features

- System-level features
 - In-circuit reconfigurability (ICR) via external configuration EPROM, intelligent controller, or Joint Test Action Group (JTAG) port
 - Full compliance with the Peripheral Component Interconnect (PCI) standard
 - Built-in JTAG boundary-scan test (BST) circuits compliant with IEEE Std 1149.1-1990
 - 3.3- or 5.0-V I/O pins on all devices in pin-grid array (PGA), ball-grid array (BGA), and 208-pin quad flat pack (QFP) packages
 - Able to bridge between 3.3-V and 5.0-V systems
 - Low power consumption (less than 1 mA in standby mode)
- Flexible package options
 - Available in a variety of packages with 84 to 560 pins (see Table 2)
 - Pin-compatibility with other FLEX 10K devices in the same package
- Software design support and automatic place-and-route provided by Altera's MAX+PLUS II development system for 486- and Pentium-based PCs and Sun SPARCstation, HP 9000 Series 700, and IBM RISC System/6000 workstations.

Table 2. FLEX 10K Package Options & I/O Count

Device	84-Pin PLCC	208-Pin QFP	240-Pin QFP	304-Pin QFP	356-Pin BGA	403-Pin PGA	504-Pin PGA	560-Pin BGA
EPF10K10	(1)	(1)	(1)		(1)			
EPF10K20	(1)	(1)	(1)		(1)			
EPF10K30		(1)	(1)		(1)	248		
EPF10K40		(1)	(1)		(1)			
EPF10K50			189	249	310	310		
EPF10K70			(1)	(1)			358	(1)
EPF10K100							406	(1)

Note:

(1) For I/O pin count and package availability, call Altera Marketing at (408) 894-7104.

General Description

Altera's FLEX 10K devices are the industry's first embedded array programmable logic devices (PLDs). Based on configurable CMOS SRAM elements, the Flexible Logic Element MatriX (FLEX) architecture incorporates all the features necessary to implement common gate array megafunctions. With up to 100,000 gates, the FLEX 10K family provides the density, speed, and features to integrate entire systems, including multiple 32-bit buses, into a single device. Table 3 shows FLEX 10K performance for typical applications, as well as the logic elements (LEs) and the embedded array blocks (EABs) required.

Application	LEs Used	EABs Used	-4 Speed Grade	-5 Speed Grade	Unit
16-bit loadable counter	16	0	79	70	MHz
16-bit accumulator	16	0	79	70	MHz
16-to-1 multiplexer, <i>Note (1)</i>	10	0	11.0	12.5	ns
4 × 4 multiplier, <i>Note (2)</i>	0	1	73	64	MHz
8 × 8 multiplier, <i>Note (2)</i>	28	4	23	20	MHz
256 × 8 RAM, <i>Note (2)</i>	0	1	50	44	MHz

Notes:

- (1) This application uses combinatorial inputs and outputs.
- (2) This application uses registered inputs and outputs.

The FLEX 10K architecture is similar to that of embedded gate arrays, the fastest-growing segment of the gate array market. As with standard gate arrays, embedded gate arrays implement general logic in a conventional "sea-of-gates" architecture. In addition, embedded gate arrays have dedicated die areas for implementing large, specialized functions. By embedding functions in silicon, embedded gate arrays provide reduced die area and increased speed compared to standard gate arrays. However, the embedded megafunctions typically cannot be customized, limiting the designer's options. In contrast, FLEX 10K devices are programmable, providing the designer full control over embedded megafunctions and general logic while facilitating iterative design changes during debugging.

Each FLEX 10K device contains an embedded array and a logic array. The embedded array is used to implement a variety of memory functions or complex logic functions, such as digital signal processing (DSP), microcontrollers, wide data-path manipulation, and data transformation. The logic array performs the same function as the sea-of-gates in the gate array: it is used to implement general logic, such as counters, adders, state machines, and multiplexers. The combination of embedded and logic arrays provides the high performance and high density of embedded gate arrays, enabling designers to implement an entire system on a single device.

FLEX 10K devices are configured at system power-up with data stored in an Altera serial Configuration EPROM device or provided by a system controller. Altera offers the EPC1 Configuration EPROM, which configures FLEX 10K devices via a serial data stream. Configuration data can also be downloaded from system RAM or from Altera's BitBlaster serial download cable. After a FLEX 10K device has been configured, it can be reconfigured in-circuit by resetting the device and loading new data. Because reconfiguration requires less than 200 ms, real-time changes can be made during system operation.



Go to the *EPC1 Configuration EPROM for FLEX Devices Data Sheet* for more information.

FLEX 10K devices are supported by Altera's MAX+PLUS II development system, a single, integrated package that offers schematic, text—including the Altera Hardware Description Language (AHDL)—and waveform design entry; compilation and logic synthesis; full simulation and worst-case timing analysis; and device configuration. MAX+PLUS II provides EDIF 2.0.0 and 3.0.0, VHDL, Verilog HDL, and other interfaces for additional design entry and simulation support from other industry-standard PC- and workstation-based EDA tools. MAX+PLUS II runs on 486- and Pentium-based PCs, and Sun SPARCstation, HP 9000 Series 700, and IBM RISC System/6000 workstations.



Go to the *MAX+PLUS II Programmable Logic Development System & Software Data Sheet* in the Altera **1995 Data Book** for more information.

Functional Description

Each FLEX 10K device contains an embedded array to implement memory and specialized logic functions, and a logic array to implement general logic.

The embedded array consists of a series of embedded array blocks (EABs). When implementing memory functions, each EAB provides 2,048 bits, which can be used to create a RAM, ROM, FIFO, or dual-port RAM. When implementing logic, each EAB can contribute 100 to 300 gates towards complex logic functions, such as multipliers, microcontrollers, and DSPs. EABs can be used independently, or multiple EABs can be combined to implement larger functions.

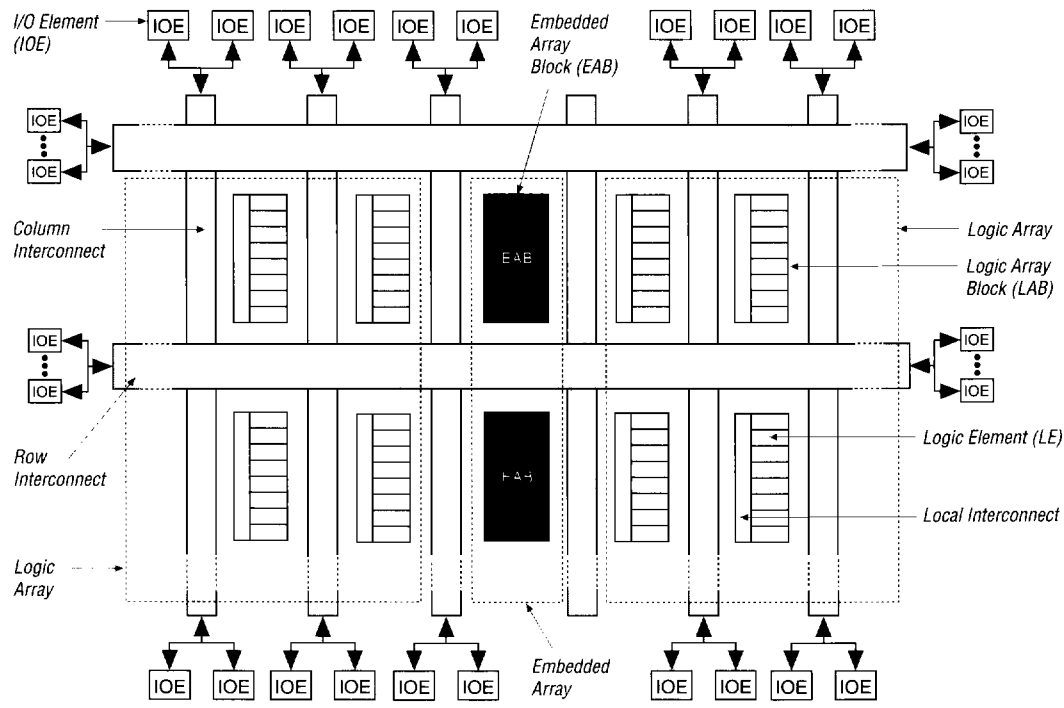
The logic array consists of Logic Array Blocks (LABs). Each LAB contains eight logic elements (LEs) and a local interconnect. An LE consists of a 4-input look-up table (LUT), a programmable flipflop, and dedicated signal paths for carry and cascade functions. The eight LEs can be used to create medium-sized blocks of logic—such as 8-bit counters, address decoders, or state machines—or combined across LABs to create larger logic blocks. Each LAB represents about 96 usable gates of logic.

Signal interconnections within FLEX 10K devices, and to and from device pins, are provided by the FastTrack Interconnect, a series of fast, continuous row and column channels that run the entire length and width of the device.

Each I/O pin is fed by an I/O element (IOE) located at the end of each row and column of the FastTrack Interconnect. Each IOE contains a bidirectional I/O buffer and a flipflop that can be used as either an output or input register to feed output, input, or bidirectional signals. When used with a dedicated Clock pin, these registers provide exceptional performance. As outputs, these registers provide Clock-to-output times of less than 8 ns; as inputs, they provide setup times of less than 10 ns and hold times of 0 ns. IOEs provide a variety of features, such as JTAG programming support, slew-rate control, tri-state buffers, and open-collector outputs.

Figure 1 shows a block diagram of the FLEX 10K architecture. Each group of LEs is combined into an LAB; LABs are arranged into rows and columns. Each row also contains a single EAB. The LABs and EABs are interconnected by the FastTrack Interconnect. At the end of each row and column of the FastTrack Interconnect are IOEs.

Figure 1. FLEX 10K Device Block Diagram



FLEX 10K devices provide six dedicated inputs that drive the control inputs of the flipflops to ensure the efficient distribution of high-speed, low-skew control signals. These signals use dedicated routing channels that provide shorter delays and lower skews than the FastTrack Interconnect. Four of the dedicated inputs drive four global signals. These four global signals can also be driven by internal logic, providing an ideal solution for a Clock divider or an internally generated asynchronous Clear that clears many registers in the device.

Embedded Array Block

The embedded array block (EAB) facilitates the implementation of common gate array megafunctions. The EAB is a flexible RAM with registers on the input and output ports. However, the size and flexibility of the EAB make it suitable for more than memory, including functions such as multipliers, vector scalars, and error correction circuits. These functions can be combined in applications such as digital filters and microcontrollers.

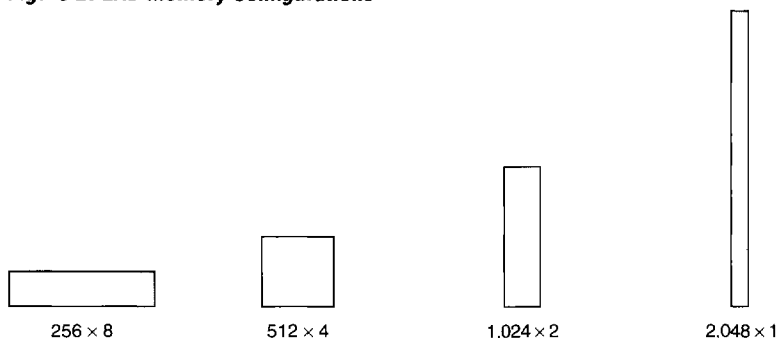
Logic functions are implemented by programming the EAB during configuration with a read-only pattern, creating a large LUT. In this LUT, combinatorial functions are implemented by looking up the results, rather than by computing them. This method of performing combinatorial functions is faster than using algorithms implemented in general logic. This performance advantage is further enhanced by the fast access times of EABs. The large capacity of EABs enable designers to implement complex functions in one logic level without the routing delays associated with linked LEs or FPGA RAM blocks. For example, a single EAB can fit a 4×4 multiplier with eight inputs and eight outputs.

The EAB has advantages over field-programmable gate arrays (FPGAs), which implement blocks of on-board RAM as arrays of small, distributed RAMs. These FPGA RAMs contain delays that are less predictable as the size of the RAM increases. In addition, because the small blocks of RAM must be connected together to make larger blocks, they are prone to routing problems. In contrast, EABs can be used to implement large, dedicated blocks of RAM that eliminate these timing and routing concerns. Dedicated EABs are easy to use and provide fast, predictable delays.

The EAB can be used to implement synchronous RAM, which is easier to implement than the asynchronous RAM of typical FPGAs. A circuit using asynchronous RAM must generate the RAM's Write Enable (WE) signal, while ensuring that its data and address signals meet setup and hold time specifications relative to the WE signal. In contrast, the EAB's synchronous RAM generates its own WE signal and is self-timed with respect to the global Clock. A circuit using the EAB's RAM need only meet the setup and hold time specifications of the global Clock.

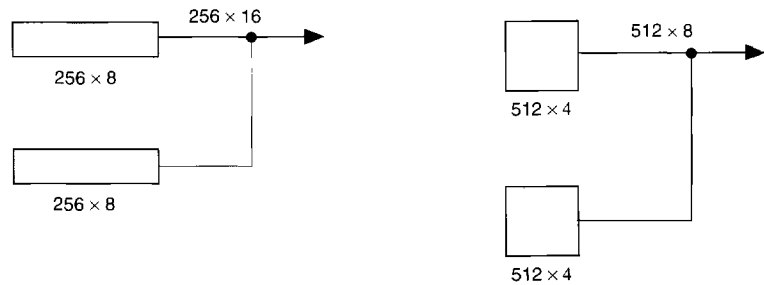
When used as RAM, each EAB can be configured as any of the following sizes: 256×8 , 512×4 , $1,024 \times 2$, or $2,048 \times 1$. See Figure 2.

Figure 2. EAB Memory Configurations



Larger RAMs are created by combining multiple EABs. For example, two 256×8 RAMs can be combined to form a 256×16 RAM; two 512×4 RAMs can be combined to form a 512×8 RAM. If necessary, all EABs in a device can be cascaded to form a single RAM. EABs can be cascaded to form RAMs of up to 2,048 words without impacting timing. Altera's MAX+PLUS II software automatically combines EABs to implement a designer's RAM specifications. See Figure 3.

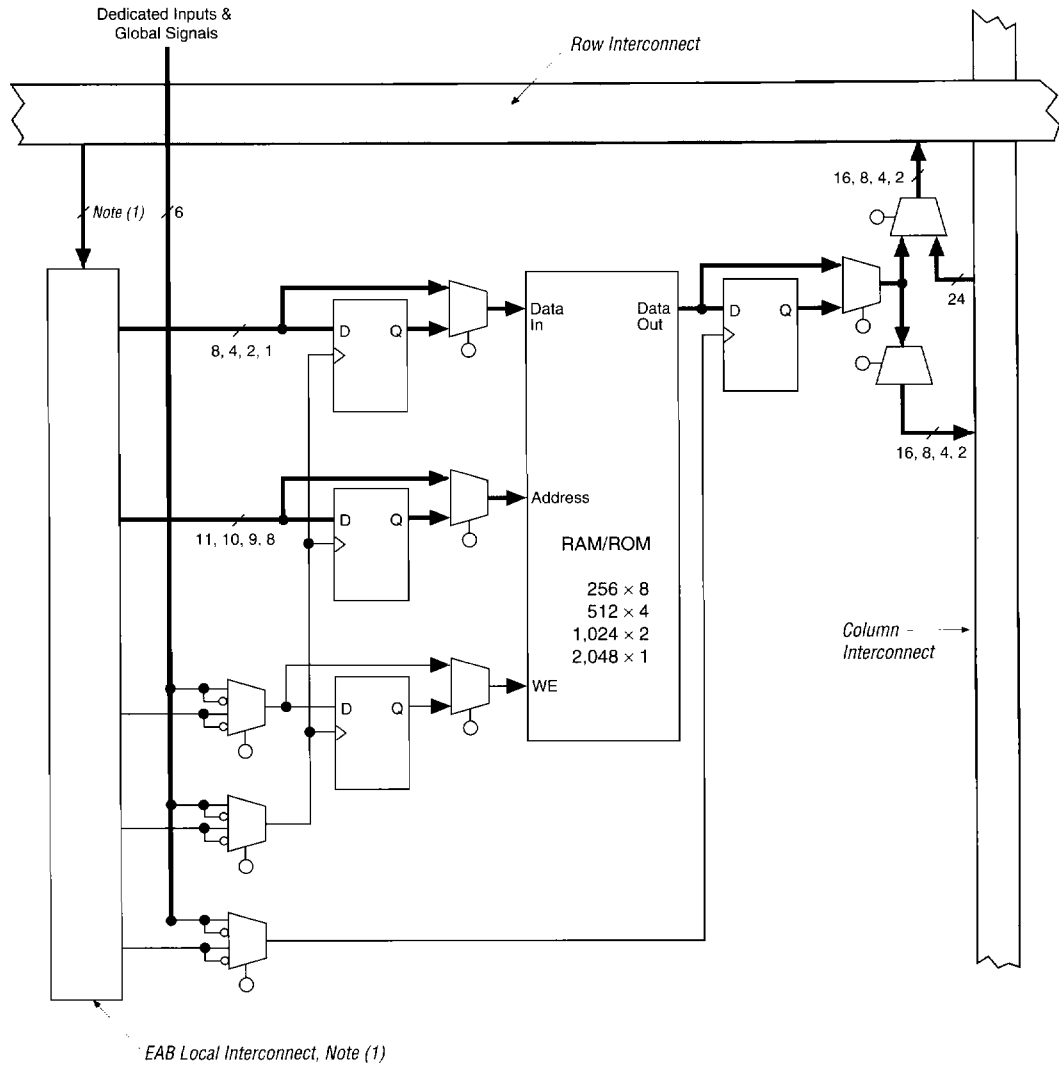
Figure 3. Examples of Combining EABs



The EAB provides flexible options for driving and controlling Clock signals. Different Clocks can be used for the EAB inputs and outputs. Registers can be independently inserted on the data input, EAB output, or the address and WE signals. The global signals and the EAB local interconnect can drive the WE signal. The global signals, dedicated Clock pins, and EAB local interconnect can drive the EAB Clock signals. Therefore, because the LEs drive the EAB local interconnect, the LEs can control the WE signal or the EAB Clock signals.

Each EAB is fed by a row interconnect, and can drive out to row and column interconnects. Each EAB output can drive either of two row channels and either of two column channels; the unused row channel can be driven by a column channel. This feature increases the routing resources available for EAB outputs. See Figure 4.

Figure 4. FLEX 10K Embedded Array Block (EAB)



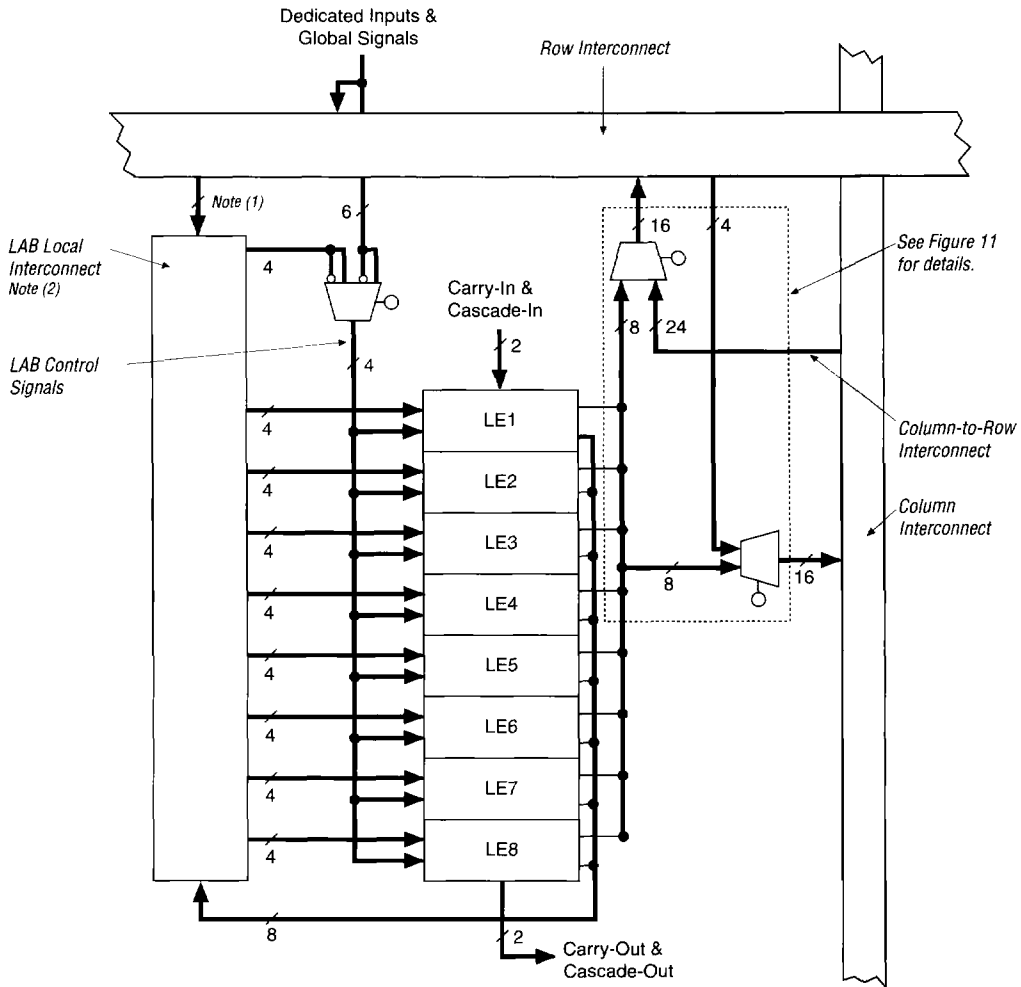
Note:

- (1) The EPF10K10, EPF10K20, EPF10K30, EPF10K40, and EPF10K50 devices have 22 EAB local interconnect channels; EPF10K70 and EPF10K100 devices have 26.

Logic Array Block

A Logic Array Block (LAB) consists of eight LEs, their associated carry and cascade chains, LAB control signals, and the LAB local interconnect. The LAB provides the coarse-grained structure to the FLEX 10K architecture, facilitating efficient routing with optimum device utilization and high performance. See Figure 5.

Figure 5. FLEX 10K LAB



Notes:

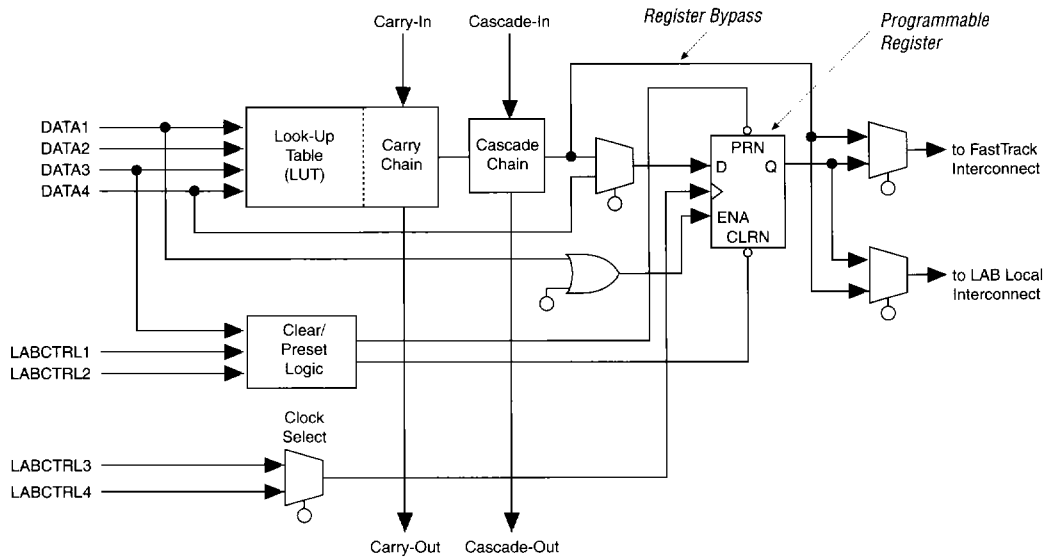
- (1) EPF10K10, EPF10K20, EPF10K30, EPF10K40, and EPF10K50 devices have 22 inputs to the LAB local interconnect channel from the row; EPF10K70 and EPF10K100 devices have 26.
- (2) EPF10K10, EPF10K20, EPF10K30, EPF10K40, and EPF10K50 devices have 30 LAB local interconnect channels; EPF10K70 and EPF10K100 devices have 34.

Each LAB provides four control signals with programmable inversion that can be used in all eight LEs. Two of these signals can be used as Clocks; the other two can be used for Clear/Presets. The LAB Clocks can be driven by the dedicated Clock input pins, global signals, I/O signals, or internal signals via the LAB local interconnect. The LAB Preset and Clear control signals can be driven by the global signals, I/O signals, or internal signals via the LAB local interconnect. The global control signals are typically used for global Clock, Clear, or Preset signals because they provide asynchronous control with very low skew across the device. If logic is required on a control signal, it can be generated in one or more LEs in any LAB and driven into the local interconnect of the target LAB. Additionally, the global control signals can be generated using LE outputs.

Logic Element

The logic element (LE), the smallest unit of logic in the FLEX 10K architecture, has a compact size that provides efficient logic utilization. Each LE contains a four-input LUT, which is a function generator that can quickly compute any function of four variables. In addition, each LE contains a programmable flipflop with a synchronous Enable, a carry chain, and a cascade chain. Each LE drives both the local and the FastTrack Interconnect. See Figure 6.

Figure 6. FLEX 10K Logic Element (LE)



The programmable flipflop in the LE can be configured for D, T, JK, or SR operation. The Clock, Clear, and Preset control signals on the flipflop can be driven by global signals, general-purpose I/O pins, or any internal logic. For combinatorial functions, the flipflop is bypassed and the output of the LUT drives the output of the LE.

The LE has two outputs that drive the interconnect; one drives the local interconnect and the other drives either the row or column FastTrack Interconnect. The two outputs can be controlled independently; for example, the LUT can drive one output while the register drives the other output. This feature, called register packing, can improve LE utilization because the register and the LUT can be used for unrelated functions.

The FLEX 10K architecture provides two types of dedicated high-speed data paths that connect adjacent LEs without using local interconnect paths: carry chains and cascade chains. The carry chain supports high-speed counters and adders; the cascade chain implements wide-input functions with minimum delay. Carry and cascade chains connect all LEs in an LAB and all LABs in the same row. Intensive use of carry and cascade chains can reduce routing flexibility. Therefore, the use of these chains should be limited to speed-critical portions of a design.

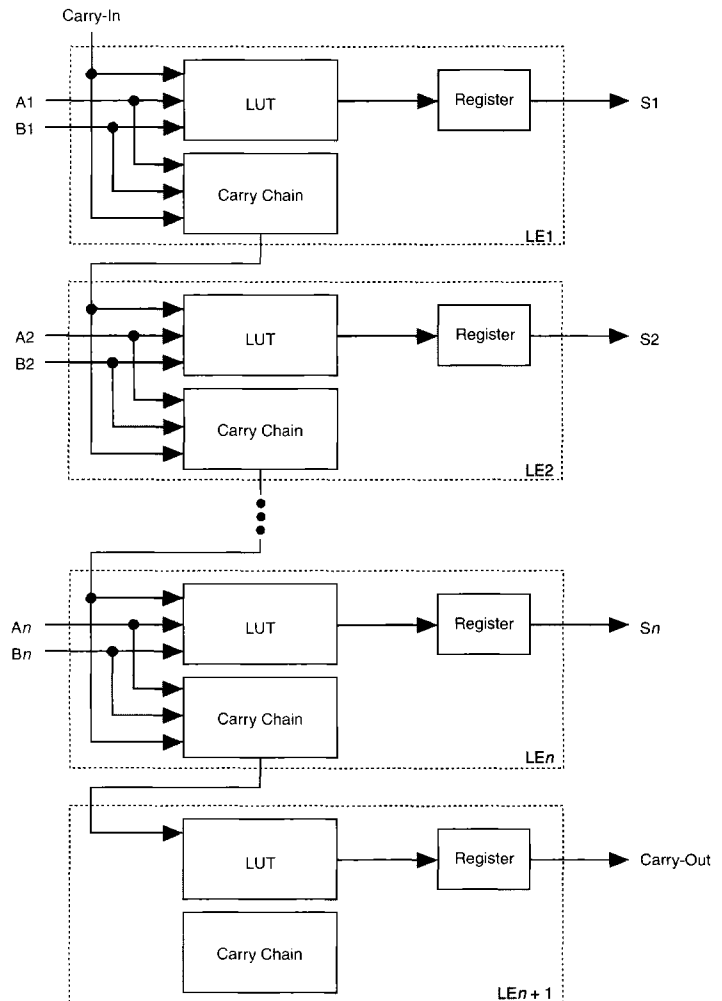
Carry Chain

The carry chain provides a very fast (less than 0.5 ns) carry-forward function between LEs. The carry-in signal from a lower-order bit moves forward into the higher-order bit via the carry chain, and feeds into both the LUT and the next portion of the carry chain. This feature allows the FLEX 10K architecture to implement high-speed counters, adders, and comparators of arbitrary width. Carry chain logic can be created automatically by the MAX+PLUS II Compiler during design processing, or manually by the designer during design entry.

Carry chains longer than eight bits are automatically implemented by linking LABs together. A long carry chain skips alternate LABs in a row, resulting in two carry chains: one in the even-numbered LABs and one in the odd-numbered LABs. The last LE of the first LAB in a row carries to the first LE of the third LAB in the row. The carry chain does not cross the EAB at the middle of the row. For example, in the EPF10K50, the carry chain stops at the eighteenth LAB and a new one begins at the nineteenth LAB.

Figure 7 shows how an n -bit full adder can be implemented in $n + 1$ LEs with the carry chain. One portion of the LUT generates the sum of two bits using the input signals and the carry-in signal; the sum is routed to the output of the LE. The register is typically bypassed for simple adders, but can be used for an accumulator function. Another portion of the LUT and the carry chain logic generate the carry-out signal, which is routed directly to the carry-in signal of the next-higher-order bit. The final carry-out signal is routed to an LE, where it can be used as a general-purpose signal.

Figure 7. Carry Chain Operation

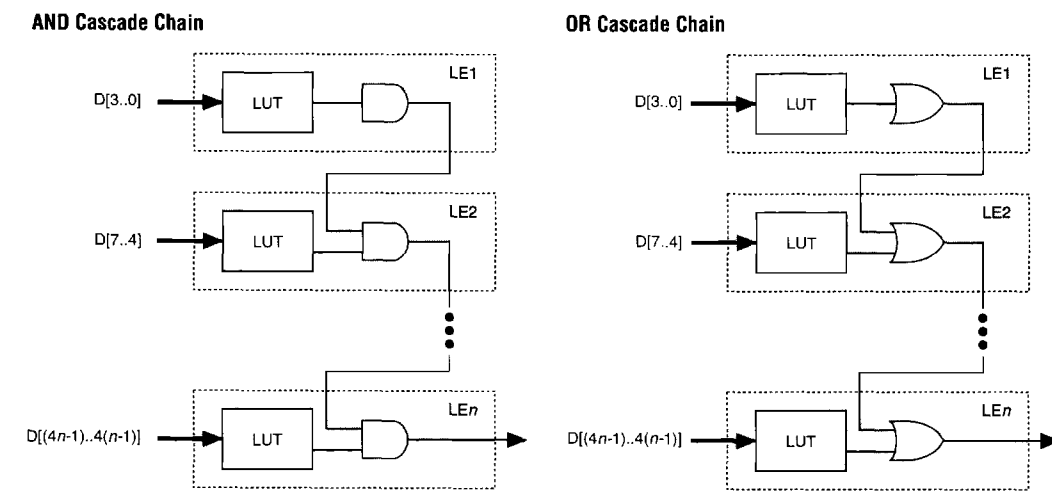


Cascade Chain

With the cascade chain, the FLEX 10K architecture can implement functions that have a very wide fan-in. Adjacent LUTs can be used to compute portions of the function in parallel; the cascade chain serially connects the intermediate values. The cascade chain can use a logical AND or logical OR (via De Morgan's inversion) to connect the outputs of adjacent LEs. Each additional LE provides four more inputs to the effective width of a function, with a delay of approximately 1.1 ns per LE. Cascade chain logic can be created automatically by the MAX+PLUS II Compiler during design processing, or manually by the designer during design entry.

Cascade chains longer than eight bits are automatically implemented by linking LABs together. For easier routing, a long cascade chain skips every other LAB in a row, resulting in two carry chains: one in the even-numbered LABs and one in the odd-numbered LABs. The last LE of the first LAB in a row cascades to the first LE of the third LAB. The cascade chain does not cross the center of the row. For example, in the EPF10K50, the cascade chain stops at the eighteenth LAB and a new one begins at the nineteenth LAB. This break is due to the EAB, which is in the middle of the row.

Figure 8 shows how the cascade function can connect adjacent LEs to form functions with a wide fan-in. These examples show functions of $4n$ variables implemented with n LEs. The LUT delay is approximately 3.0 ns; the cascade chain delay is 1.1 ns. With the cascade chain, 6.3 ns is needed to decode a 16-bit address.

Figure 8. Cascade Chain Operation

LE Operating Modes

The FLEX 10K LE can operate in one of the following four modes:

- Normal mode
- Arithmetic mode
- Up/Down Counter mode
- Clearable Counter mode

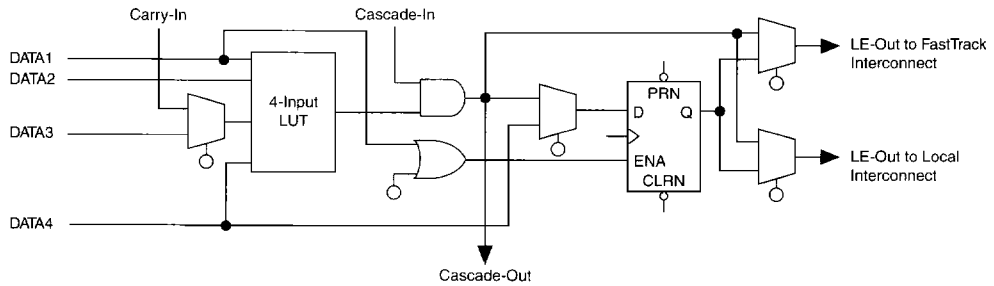
Each of these modes uses LE resources differently. In each mode, seven available inputs to the LE—the four data inputs from the LAB local interconnect, the feedback from the programmable register, and the carry-in and cascade-in from the previous LE—are directed to different destinations to implement the desired logic function. Three inputs to the LE provide Clock, Clear, and Preset control for the register. The MAX+PLUS II software automatically chooses the appropriate mode for each application. Design performance can be further enhanced by tailoring the design for the operating mode that supports the intended application.

The architecture provides a synchronous Clock Enable to the register in all modes. DATA1 can be set to synchronously enable the register, providing easy implementation of fully synchronous designs.

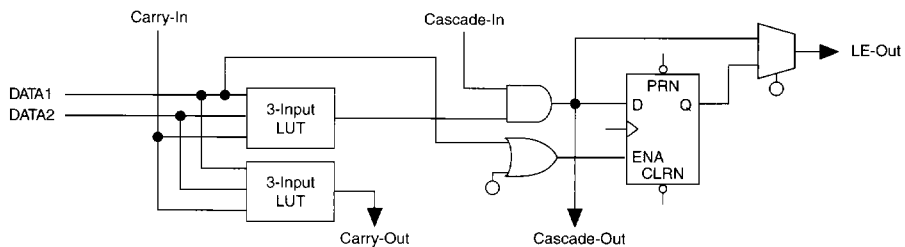
Figure 9 shows the LE operating modes.

Figure 9. FLEX 10K LE Operating Modes

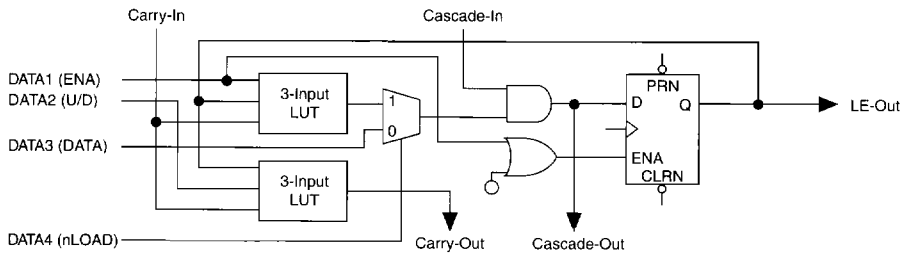
Normal Mode



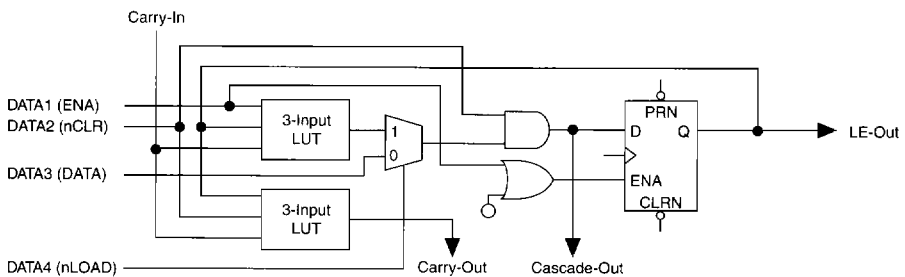
Arithmetic Mode



Up/Down Counter Mode



Clearable Counter Mode



Normal Mode

The Normal mode is suitable for general logic applications and wide decoding functions that can take advantage of a cascade chain. In Normal mode, four data inputs from the LAB local interconnect and the carry-in are inputs to a 4-input LUT. The MAX+PLUS II Compiler automatically selects the carry-in or the DATA3 signal as one of the inputs to the LUT. The LUT output can be combined with the cascade-in signal to form a cascade chain through the cascade-out signal. To support register packing, the LE has two outputs; one drives the local interconnect and the other drives the FastTrack Interconnect.

The LUT and the register in the LE can be used independently. Either the register or the LUT can be used to drive both the local interconnect and the FastTrack Interconnect at the same time. Alternatively, in a packed LE, the register can drive the FastTrack Interconnect while the LUT drives the local interconnect, or vice versa.

The DATA4 signal can drive the register directly, allowing the LUT to compute a function that is independent of the registered signal; a 3-input function can be computed in the LUT, and a fourth independent signal can be registered. Alternatively, a 4-input function can be generated, and one of the inputs to this function can be used to drive the register. The register in a packed LE can still use the Clock Enable, Clear and Preset signals in the LE.

Arithmetic Mode

The Arithmetic mode offers two 3-input LUTs that are ideal for implementing adders, accumulators, and comparators. One LUT computes a 3-input function; the other generates a carry output. As shown in Figure 9 on page 16, the first LUT uses the carry-in signal and two data inputs from the LAB local interconnect to generate a combinatorial or registered output. For example, in an adder, this output would be the sum of three signals: A, B, and carry-in. The second LUT uses the same three signals to generate a carry-out signal, thereby creating a carry chain. The Arithmetic mode also supports simultaneous use of the cascade chain.

Up/Down Counter Mode

The Up/Down Counter mode offers counter Enable, Clock Enable, synchronous up/down control, and data loading options. These control signals are generated by the data inputs from the LAB local interconnect, the carry-in signal, and output feedback from the programmable register. Two 3-input LUTs are used: one generates the counter data, the other generates the fast carry bit. A 2-to-1 multiplexer provides synchronous loading. Data can also be loaded asynchronously with the Clear and Preset register control signals, without using the LUT resources.

Clearable Counter Mode

The Clearable Counter mode is similar to the Up/Down Counter mode, but supports a synchronous Clear instead of the up/down control. The Clear function is substituted for the cascade-in signal in the Up/Down Counter mode. Two 3-input LUTs are used: one generates the counter data, the other generates the fast carry bit. Synchronous loading is provided by a 2-to-1 multiplexer. The output of this multiplexer is ANDed with a synchronous Clear signal.

Internal Tri-State Emulation

Internal tri-state emulation provides internal tri-stating without the limitations of a physical tri-state bus. In a physical tri-state bus, the tri-state buffers' Output Enable signals select which signal drives the bus. However, if multiple Output Enable signals are active, contending signals can be driven onto the bus. Conversely, if no Output Enable signals are active, the bus will float. Internal tri-state emulation resolves contending tri-state buffers to a low value and floating buses to a high value, thereby eliminating these problems. MAX+PLUS II automatically implements tri-state bus functionality with a multiplexer.

Clear & Preset Logic Control

Logic for the programmable register's Clear and Preset functions is controlled by the DATA3, LABCTRL1, and LABCTRL2 inputs to the LE. The Clear and Preset control structure of the LE is used to asynchronously load signals into a register. The register can be set up so that LABCTRL1 implements an asynchronous load. The data to be loaded is driven to DATA3; when LABCTRL1 is asserted, DATA3 is loaded into the register.

During compilation, the MAX+PLUS II Compiler automatically selects the best control signal implementation. Since the Clear and Preset functions are active-low, the Compiler automatically assigns a logic high to an unused Clear or Preset.

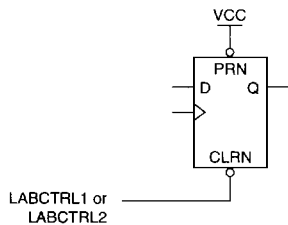
The Clear and Preset logic is implemented in one of the following five modes, chosen during design entry:

- Asynchronous Clear
- Asynchronous Preset
- Asynchronous Clear and Preset
- Asynchronous Load with Clear
- Asynchronous Load without Clear

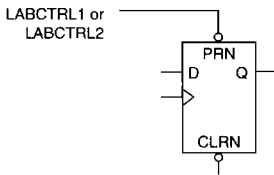
In addition to the five Clear and Preset modes, FLEX 10K devices provide a device-wide Clear pin that can reset all registers in the device. This pin is set during design entry. In any of the Clear and Preset modes, the device-wide Clear overrides all other signals. See Figure 10.

Figure 10. LE Clear & Preset Modes

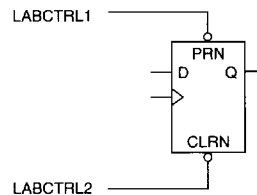
Asynchronous Clear



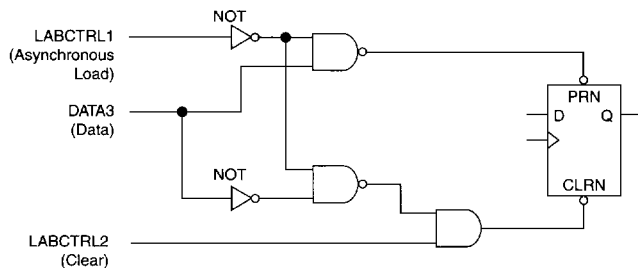
Asynchronous Preset



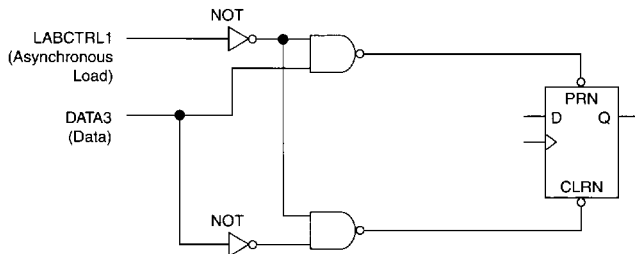
Asynchronous Preset & Clear



Asynchronous Load with Clear



Asynchronous Load without Clear



Asynchronous Clear

The flip-flop can be cleared by either LABCTRL1 or LABCTRL2. In this mode, the Preset signal is tied to VCC to deactivate it.

Asynchronous Preset

An asynchronous Preset is implemented as either an asynchronous load, or with an asynchronous Clear. If DATA3 is tied to VCC, asserting LABCTRL1 asynchronously loads a one into the register. Alternatively, MAX+PLUS II can provide Preset control by using the Clear and inverting the input and output of the register. Inversion control is available for the inputs to both LEs and IOEs. Therefore, if a register is preset by only one of the two LABCTRL signals, the DATA3 input is not needed and can be used for one of the LE operating modes.

Asynchronous Clear & Preset

When implementing asynchronous Clear and Preset, LABCTRL1 controls the Preset and LABCTRL2 controls the Clear. DATA3 is tied to VCC, therefore, asserting LABCTRL1 asynchronously loads a one into the register, effectively presetting the register. Asserting LABCTRL2 clears the register.

Asynchronous Load with Clear

When implementing an asynchronous load with the Clear, LABCTRL1 implements the asynchronous load of DATA3 by controlling the register Preset and Clear. LABCTRL2 implements the Clear by controlling the register Clear; LABCTRL2 does not have to feed the Preset circuits.

Asynchronous Load without Clear

When implementing an asynchronous load without the Clear, LABCTRL1 implements the asynchronous load of DATA3 by controlling the register Preset and Clear.

FastTrack Interconnect

In the FLEX 10K architecture, connections between LEs and device I/O pins are provided by the FastTrack Interconnect, a series of continuous horizontal and vertical routing channels that traverse the device. This global routing structure provides predictable performance, even in complex designs. In contrast, the segmented routing in FPGAs requires switch matrices to connect a variable number of routing paths, increasing the delays between logic resources and reducing performance.

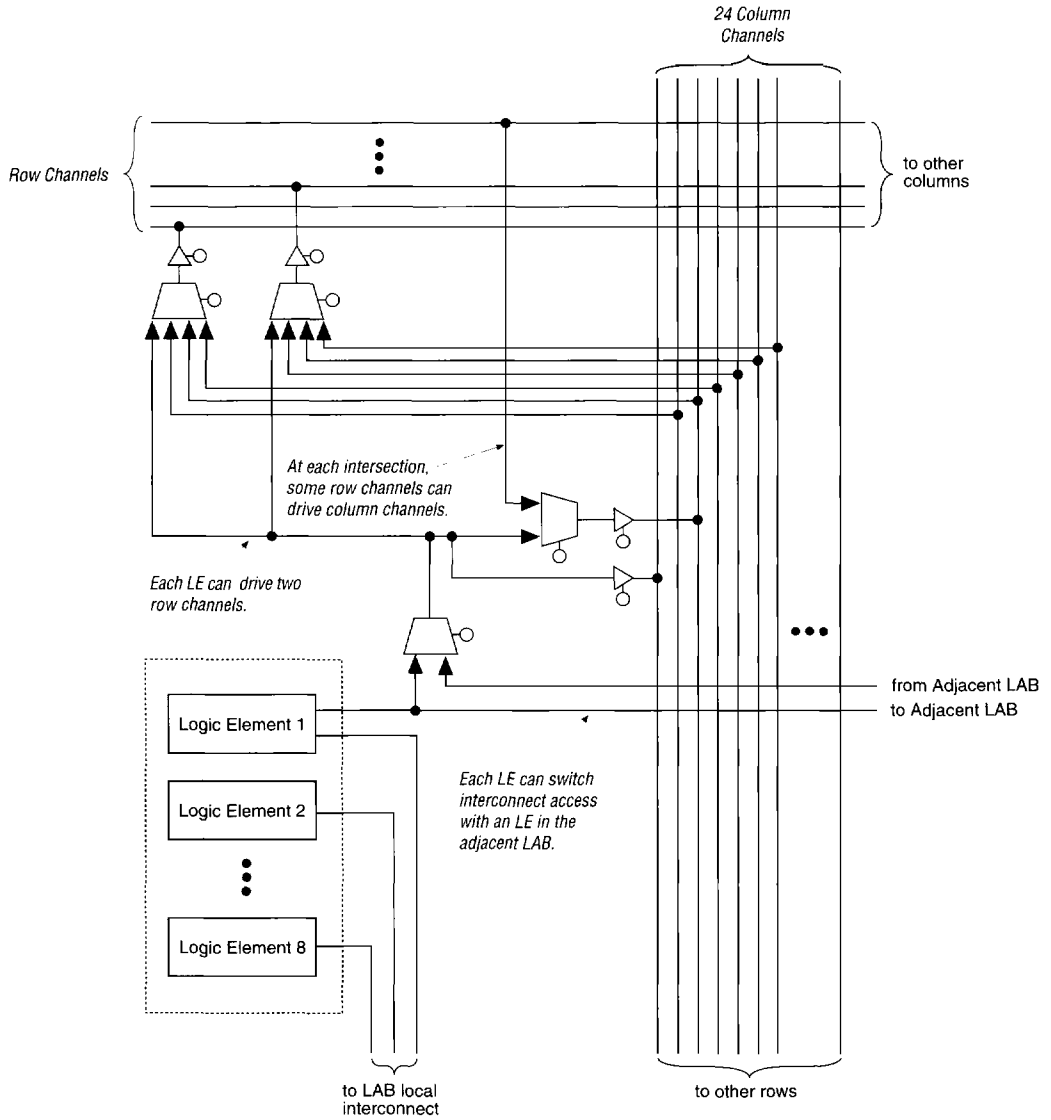
The FastTrack Interconnect consists of column and row interconnect channels that span the entire device. Each row of LABs is served by a dedicated row interconnect. The row interconnect can drive I/O pins and feed other LABs in the device. The column interconnect routes signals between rows and can drive I/O pins.

A row channel can be driven by an LE or by one of three column channels. These four signals feed dual 4-to-1 multiplexers that connect to two specific row channels. These multiplexers, which are connected to each LE, allow column channels to drive row channels even when all eight LEs in a LAB drive a row interconnect.

Each column of LABs is served by a dedicated column interconnect. The column interconnect can then drive I/O pins or drive another row's interconnect to route the signals to other LABs in the device. A signal from the column interconnect, which can be either the output of an LE or an input from an I/O pin, must be routed to the row interconnect before it can enter an LAB or EAB. Each row channel that is driven by an IOE or EAB can drive one specific column channel.

Access to row and column channels can be switched between LEs in adjacent pairs of LABs. For example, an LE in one LAB can drive the row and column channels normally driven by a particular LE in the adjacent LAB in the same row, and vice versa. This routing flexibility enables routing resources to be used more efficiently. See Figure 11.

Figure 11. LAB Connections to Row & Column Interconnect



For improved routability, the row interconnect is comprised of a combination of full-length and half-length channels. The full-length channels connect to all LABs in a row; the half-length channels connect to the LABs in one-half of the row. The EAB can be driven by the half-channels in the left half of the row and by the full channels. In addition to providing a predictable, row-wide interconnect, this architecture provides increased routing resources because two neighboring LABs can be connected using a half-row channel, thereby saving the other half of the channel for the other half of the row.

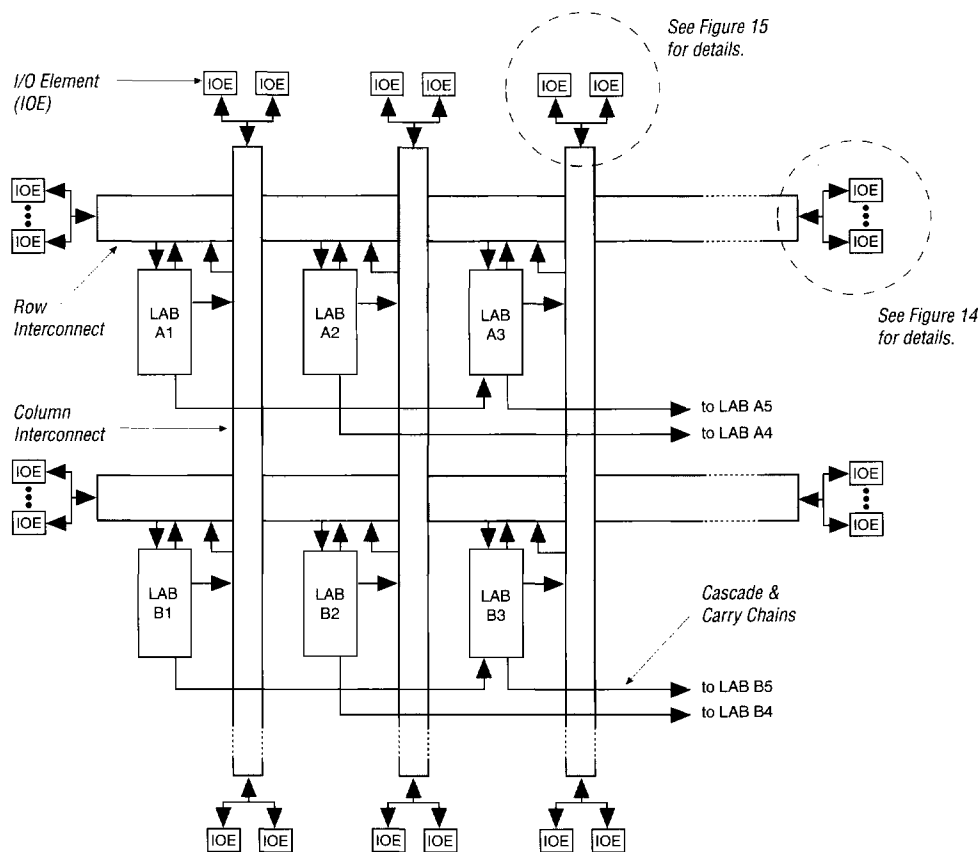
Table 4 summarizes the FastTrack Interconnect resources available in each FLEX 10K device.

Device	Rows	Channels per Row	Columns	Channels per Column
EPF10K10	3	144	24	24
EPF10K20	6	144	24	24
EPF10K30	6	216	36	24
EPF10K40	8	216	36	24
EPF10K50	10	216	36	24
EPF10K70	9	312	52	24
EPF10K100	12	312	52	24

In addition to general-purpose I/O pins, FLEX 10K devices have six dedicated input pins that provide low-skew signal distribution across the device. These six inputs may be used for global Clock, Clear, Preset, and peripheral Output Enable and Clock Enable control signals. These signals are available as control signals for all LABs and IOEs in the device. The dedicated inputs can also be used as general-purpose data inputs because they can feed the local interconnect of each LAB in the device. However, the use of dedicated inputs as data inputs can introduce additional delay into the control signal network.

Figure 12 shows the interconnection of adjacent LABs and EABs, with row, column, and local interconnects, as well as the associated cascade and carry chains. The LABs are labelled according to their location, with a letter representing their row and a number representing their column. For example, LAB B3 is in row B, column 3.

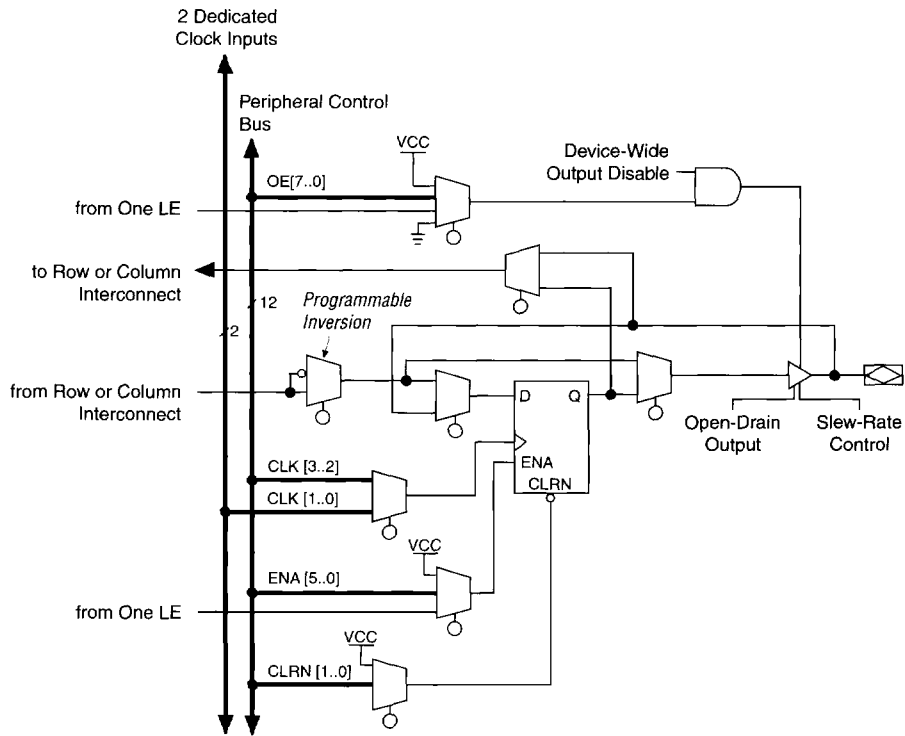
Figure 12. FLEX 10K Device Interconnect Resources



I/O Element

An IOE contains a bidirectional I/O buffer and a register that can be used either as an input register for external data that requires a fast setup time, or as an output register for data that requires fast Clock-to-output performance. I/O elements can be used as input, output, or bidirectional pins. The MAX+PLUS II Compiler uses the programmable inversion option to automatically invert signals from the row and column interconnect where appropriate. Figure 13 shows the IOE block diagram.

Figure 13. FLEX 10K I/O Element (IOE)



The output buffer in each IOE has an adjustable output slew rate that can be configured for low-noise or high-speed performance. A slower slew rate reduces system noise and adds a maximum delay of 4.5 ns. The fast slew rate should be used for speed-critical outputs in systems that are adequately protected against noise. Designers can specify the slew rate on a pin-by-pin basis during design entry or assign a default slew rate to all pins on a device-wide basis. Each pin can also be specified to be open-collector on a pin-by-pin basis.

Each IOE selects the Clock, Clear, Clock Enable, and Output Enable controls from a network of I/O control signals called the peripheral control bus. The peripheral control bus uses high-speed drivers to minimize signal skew across devices; it provides up to 12 peripheral control signals that can be allocated as follows:

- Up to 8 Output Enable signals
- Up to 6 Clock Enable signals
- Up to 2 Clock signals
- Up to 2 Clear signals

If more than 6 Clock Enable or 8 Output Enable signals are required, each IOE on the device can be controlled by Clock Enable and Output Enable signals driven by a specific LE. In addition to the 2 Clock signals available on the peripheral control bus, each IOE can use one of 2 dedicated Clock pins. Each peripheral control signal can be driven by any of the dedicated input pins or the first LE of each LAB in a particular row.

Table 5 lists the sources for each peripheral control signal, shows how the 8 Output Enable signals, 6 Clock Enable signals, 2 Clock signals, and 2 Clear signals share the 12 peripheral control signals, and shows which rows can drive the global signals.

Peripheral Control Signal	EPF10K50	EPF10K100
OE0	Row A	Row A
OE1	Row B	Row C
OE2	Row C	Row E
OE3	Row D	Row G
OE4	Row E	Row I
OE5	Row J	Row K
ENA0, CLK0, or GLOBAL0	Row A	Row B
ENA1, OE6, or GLOBAL1	Row F	Row D
ENA2 or CLR0	Row G	Row F
ENA3, OE7, or GLOBAL2	Row H	Row H
ENA4 or CLR1	Row I	Row J
ENA5, CLK1, or GLOBAL3	Row J	Row L

Note:

- (1) Call Altera Applications for peripheral control bus data for other FLEX 10K devices.

Signals on the peripheral control bus can also drive the four global signals, referred to as GLOBAL0 through GLOBAL3 in Table 5. The internally generated signal can drive the global signal, providing the same low-skew, low-delay characteristics for an internally generated signal as for a signal driven by an input. This feature is ideal for internally generated Clear or Clock signals with high fan-out.

A device-wide output disable pin is an active-low pin that can be used to tri-state all pins on the device. This option can be set in the design file.

Row-to-IOE Connections

When an IOE is used as an input signal, it can drive two separate row channels. The signal is accessible by all LEs within that row. When an IOE is used as an output, the signal is driven by a multiplexer that selects a signal from the row channels. Eight IOEs connect to each side of each row channel. See Figure 14.

Figure 14. FLEX 10K Row-to-IOE Connections

The values for m and n are provided in Table 6.

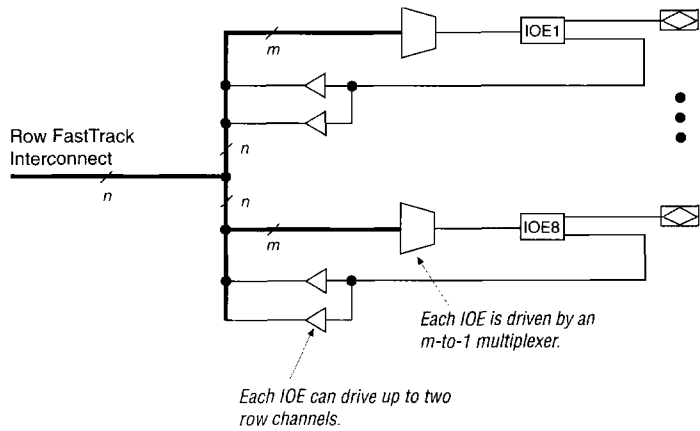


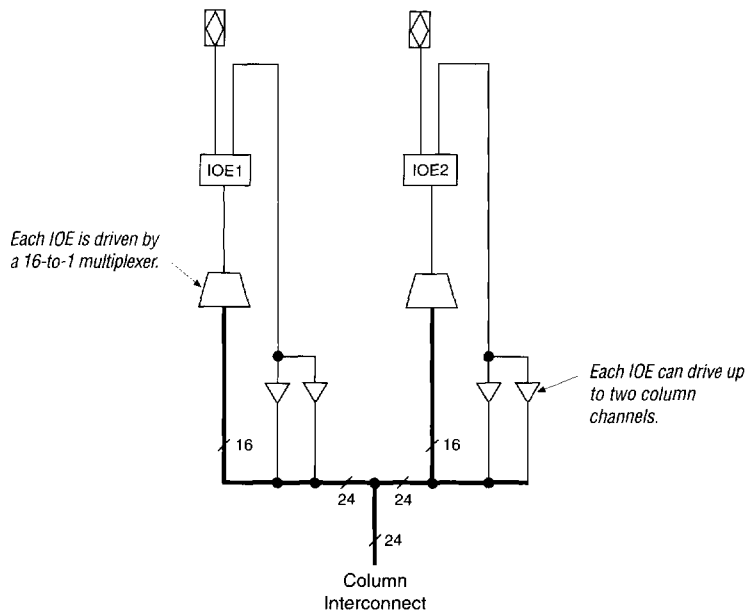
Table 6 lists the FLEX 10K row-to-IOE interconnect resources.

Table 6. FLEX 10K Row-to-IOE Interconnect Resources		
Device	Channels per Row (n)	Row Channel per Pin (m)
EPF10K10	144	18
EPF10K20	144	18
EPF10K30	216	27
EPF10K40	216	27
EPF10K50	216	27
EPF10K70	312	39
EPF10K100	312	39

Column-to-IOE Connections

When an IOE is used as an input, it can drive up to two separate column channels. When an IOE is used as an output, the signal is driven by a multiplexer that selects a signal from the column channels. Two IOEs connect to each side of the column channels. Each IOE can be connected to 16 of the 24 column channels via a 16-to-1 multiplexer. The set of 16 column channels that each IOE can access is different for each IOE. See Figure 15.

Figure 15. FLEX 10K Column-to-IOE Connections



3.3- or 5.0-V I/O Pin Operation

Some members of the FLEX 10K family can be set for 3.3-V or 5.0-V I/O pin operation. These devices have one set of V_{CC} pins for internal operation and input buffers (V_{CCINT}), and another set for I/O output drivers (V_{CCIO}).

The V_{CCINT} pins must always be connected to a 5.0-V power supply. With a 5.0-V V_{CCINT} level, input voltages are at TTL levels and are therefore compatible with 3.3-V and 5.0-V inputs. The V_{CCIO} pins can be connected to either a 3.3-V or 5.0-V power supply, depending on the output requirements. When the V_{CCIO} pins are connected to a 5.0-V power supply, the output levels are compatible with 5.0-V systems. When the V_{CCIO} pins are connected to a 3.3-V power supply, the output high is at 3.3 V and is therefore compatible with 3.3-V or 5.0-V systems. Devices operating with V_{CCIO} levels lower than 4.75 V incur a nominally greater timing delay of t_{OD2} instead of t_{OD1} .

JTAG Operation

All FLEX 10K devices provide JTAG BST circuits that comply with the IEEE Std 1149.1-1990 specification. All FLEX 10K devices can also be configured using the JTAG PROGRAM instruction.



Go to *Application Note 39 (JTAG Boundary-Scan Testing in Altera Devices)* for more information.

Timing Model

The continuous, high-performance FastTrack Interconnect routing resources ensure predictable performance and accurate simulation analysis. This predictable performance contrasts with that of FPGAs, which use a segmented connection scheme and hence have unpredictable performance.

Device performance can be estimated by following the signal path from a source, through the interconnect, to the destination. For example, the registered performance between two LEs on the same row could be calculated by adding the following parameters:

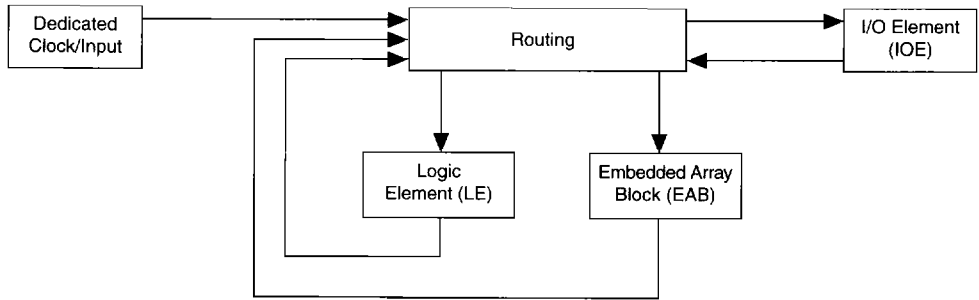
- LE register Clock-to-output delay (t_{CO})
- Routing delay ($t_{SAMEROW}$)
- LE look-up table delay (t_{LUT})
- LE register setup time (t_{SU})

The routing delay depends on the placement of the source and destination LEs. A more complex registered path may involve multiple combinatorial LEs between the source and destination LEs.

Timing simulation and delay prediction are available with the MAX+PLUS II Simulator and Timing Analyzer, or with industry-standard EDA tools. The Simulator offers both pre-synthesis functional simulation to evaluate logic design accuracy and post-synthesis timing simulation with 0.1-ns resolution. The Timing Analyzer provides point-to-point timing delay information, setup and hold time analysis, and system-level performance analysis.

Figure 16 shows the overall timing model, which maps the possible routing paths to and from the various elements of the FLEX 10K device.

Figure 16. FLEX 10K Timing Model



Figures 17 through 19 show the delays that correspond to various paths and functions within the LE, IOE, and EAB timing models.

Figure 17. FLEX 10K LE Timing Model

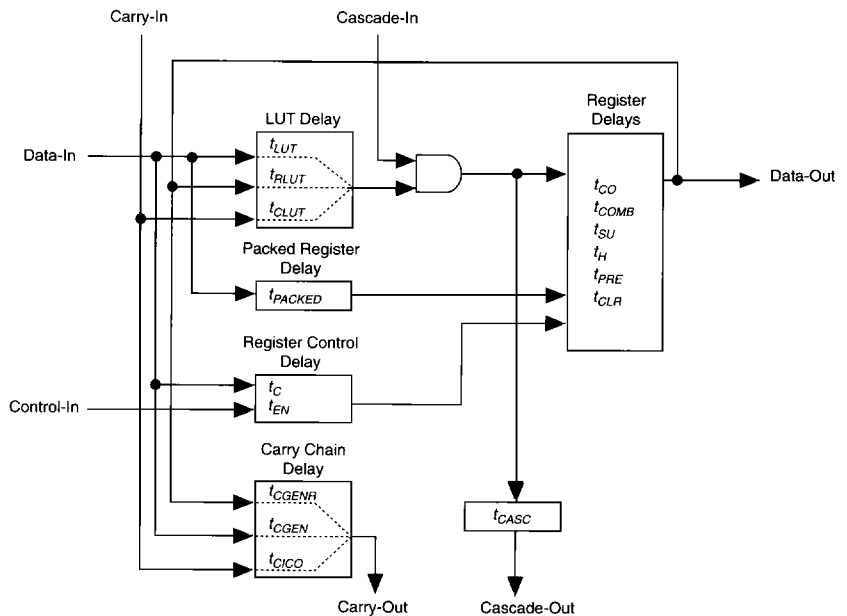


Figure 18. FLEX 10K IOE Timing Model

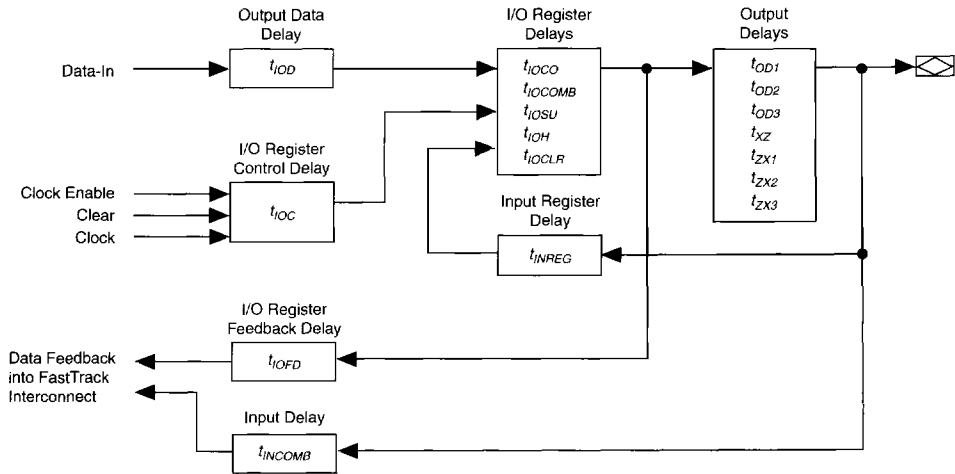
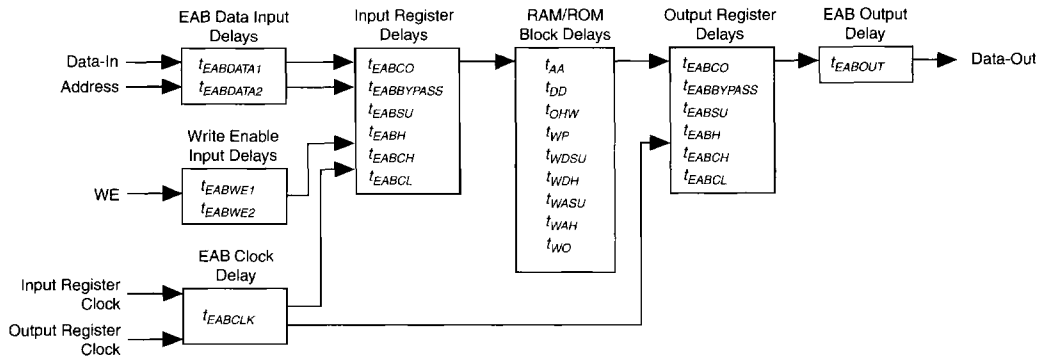


Figure 19. FLEX 10K EAB Timing Model



Tables 7 through 11 describe the FLEX 10K internal timing parameters. These internal timing parameters are expressed as worst-case values. Using hand calculations, these parameters can be used to estimate design performance. However, before committing designs to silicon, actual worst-case performance should be modeled using timing simulation and timing analysis. Tables 12 and 13 describe FLEX 10K external timing parameters.

Symbol	Parameter	Conditions
t_{LUT}	LUT delay for data-in	
t_{CLUT}	LUT delay for carry-in	
t_{RLUT}	LUT delay for LE register feedback	
t_{PACKED}	Data-in to packed register delay	
t_{EN}	LE register enable time	
t_{CICO}	Carry-in to carry-out delay	
t_{CGEN}	Data-in to carry-out delay	
t_{CGENR}	LE register feedback to carry-out delay	
t_{CASC}	Cascade-in to cascade-out delay	
t_C	LE register control signal delay	
t_{CO}	LE register clock-to-output delay	
t_{COMB}	Combinatorial delay	
t_{SU}	LE register setup time before clock	
t_H	LE register hold time before clock	
t_{PRE}	LE register preset delay	
t_{CLR}	LE register clear delay	

Table 8. IOE Timing Microparameters *Note (1)*

Symbol	Parameter	Conditions
t_{IOD}	IOE data delay	
t_{IOC}	IOE register control signal delay	
t_{IOCO}	IOE register clock-to-output delay	
t_{IOCOMB}	IOE combinatorial delay	
t_{IOSU}	IOE register data setup time before clock	
t_{IOH}	IOE register data hold time after clock	
t_{IOCLR}	IOE register clear time	
t_{OD1}	Output buffer and pad delay, Slow slew rate = off, $V_{CCIO} = 5.0$ V	C1 = 35 pF <i>Note (2)</i>
t_{OD2}	Output buffer and pad delay, Slow slew rate = off, $V_{CCIO} = 3.3$ V	C1 = 35 pF <i>Note (3)</i>
t_{OD3}	Output buffer and pad delay, Slow slew rate = on	C1 = 35 pF <i>Note (4)</i>
t_{XZ}	IOE output buffer disable delay	
t_{ZX1}	IOE output buffer enable delay, Slow slew rate = off, $V_{CCIO} = 5.0$ V	C1 = 35 pF <i>Note (2)</i>
t_{ZX2}	IOE output buffer enable delay, Slow slew rate = off, $V_{CCIO} = 3.3$ V	C1 = 35 pF <i>Note (3)</i>
t_{ZX3}	IOE output buffer enable delay, Slow slew rate = off	C1 = 35 pF <i>Note (4)</i>
t_{INREG}	IOE input pad and buffer to IOE register delay	
t_{IOFD}	IOE register feedback delay	
t_{INCOMB}	IOE input pad and buffer to FastTrack Interconnect delay	

Table 9. EAB Timing Microparameters Note (1)

Symbol	Parameter	Conditions
$t_{EABDATA1}$	Data or address delay to EAB for combinatorial input	
$t_{EABDATA2}$	Data or address delay to EAB for registered input	
t_{EABWE1}	Write enable delay to EAB for combinatorial input	
t_{EABWE2}	Write enable delay to EAB for registered input	
t_{EABCLK}	EAB register clock delay	
t_{EABCO}	EAB register clock-to-output delay	
$t_{EABBYPASS}$	Bypass register delay	
t_{EABSU}	EAB register setup time before clock	
t_{EABH}	EAB register hold time after clock	
t_{EABCH}	Clock high time	
t_{EABCL}	Clock low time	
t_{AA}	Address access delay	
t_{WP}	Write pulse width	
t_{WDSU}	Data setup time before falling edge of write pulse	Note (5)
t_{WDH}	Data hold time after falling edge of write pulse	Note (5)
t_{WASU}	Address setup time before rising edge of write pulse	Note (5)
t_{WAH}	Address hold time after falling edge of write pulse	Note (5)
t_{WO}	Write enable to data output valid delay	
t_{OHV}	Data-out hold from WE change	
t_{DD}	Data-in to data-out valid delay	
t_{EABOUT}	Data-out delay	

Table 10. EAB Timing Macroparameters (Part 1 of 2) Note (6)

Symbol	Parameter	Equation	Conditions
t_{EABAA}	EAB address access delay	$t_{EABDATA1} + t_{EABBYPASS} + t_{AA} + t_{EABBYPASS} + t_{EABOUT}$	
$t_{EABRCCOMB}$	EAB asynchronous read cycle time	$t_{EABDATA1} + t_{EABBYPASS} + t_{AA} + t_{EABBYPASS} + t_{EABOUT}$	
$t_{EABRCREG}$	EAB synchronous read cycle time	$t_{EABCO} + t_{AA} + t_{EABSU}$	
t_{EABWP}	EAB write pulse width	t_{WP}	
$t_{EABWCCOMB}$	EAB asynchronous write cycle time	$t_{WASU} + t_{WP} + t_{WAH}$	
$t_{EABWCREG}$	EAB synchronous write cycle time	$t_{EABCO} + t_{WO} + t_{EABSU}$	

Table 10. EAB Timing Macroparameters (Part 2 of 2) Note (6)

Symbol	Parameter	Equation	Conditions
t_{EABDD}	EAB data-in to data-out valid delay (when in read during write mode)	$t_{EABDATA1} + t_{EABBYPASS} + t_{DD} + t_{EABBYPASS} + t_{EABOUT}$	
$t_{EABDATAO}$	EAB clock-to-output delay when using output registers	$t_{EABCLK} + t_{EABCO} + t_{EABOUT}$	
$t_{EABDATASU}$	EAB data/address setup time before clock when using input register	$t_{EABSU} + t_{EABDATA2} - t_{EABCLK}$	
$t_{EABDATAH}$	EAB data/address hold time after clock when using input register	$t_{EABH} + t_{EABCLK} - t_{EABDATA2}$	
$t_{EABWESU}$	EAB \overline{WE} setup time before clock when using input register	$t_{EABSU} + t_{EABWE2} - t_{EABCLK}$	
t_{EABWEH}	EAB \overline{WE} hold time after clock when using input register	$t_{EABH} + t_{EABCLK} - t_{EABWE2}$	
$t_{EABWDSU}$	EAB data setup time before falling edge of write pulse when not using input registers	$t_{WDSU} + (t_{EABDATA1} + t_{EABBYPASS}) - (t_{EABWE1} + t_{EABBYPASS})$	
t_{EABWDH}	EAB data hold time after falling edge of write pulse when not using input registers	$t_{WDH} + (t_{EABWE1} + t_{EABBYPASS}) - (t_{EABDATA1} + t_{EABBYPASS})$	
$t_{EABWASU}$	EAB address setup time before rising edge of write pulse when not using input registers	$t_{WASU} + (t_{EABDATA1} + t_{EABBYPASS}) - (t_{EABWE1} + t_{EABBYPASS})$	
t_{EABWAH}	EAB address hold time after falling edge of write pulse when not using input registers	$t_{WAH} + (t_{EABWE1} + t_{EABBYPASS}) - (t_{EABDATA1} + t_{EABBYPASS})$	
t_{EABWO}	EAB write enable to data output valid delay	$t_{EABWE1} + t_{EABBYPASS} + t_{WO} + t_{EABBYPASS} + t_{EABOUT}$	
t_{EABOHV}	EAB data-out hold from \overline{WE} change	$t_{EABWE1} + t_{EABBYPASS} + t_{OHV} + t_{EABBYPASS} + t_{EABOUT}$	

Symbol	Parameter	Conditions
t_{SAMELAB}	Routing delay for an LE driving another LE in the same LAB	
t_{SAMEROW}	Routing delay for a row IOE, LE, or EAB driving a row IOE, LE, or EAB in the same row	<i>Note (7)</i>
$t_{\text{SAMECOLUMN}}$	Routing delay for an LE driving an IOE in the same column	<i>Note (7)</i>
t_{DIFFROW}	Routing delay for a column IOE, LE, or EAB driving an LE or EAB in a different row	<i>Note (7)</i>
t_{TWOROWS}	Routing delay for a row IOE or EAB driving an LE or EAB in a different row	<i>Note (7)</i>
t_{LEPERIPH}	Routing delay for an LE driving a control signal of an IOE via the peripheral control bus	<i>Note (7)</i>
t_{LEGLOBAL}	Routing delay for an LE driving a control signal of an LE or an EAB via a global control signal	<i>Note (7)</i>
t_{LABCARRY}	Routing delay for the carry-out signal of an LE driving the carry-in signal of a different LE in a different LAB	
t_{LABCASC}	Routing delay for the cascade-out signal of an LE driving the cascade-in signal of a different LE in a different LAB	
t_{DIN2IOE}	Delay from dedicated input pin to IOE control input	<i>Note (7)</i>
t_{DIN2LE}	Delay from dedicated input pin to LE control input	<i>Note (7)</i>
t_{DCLK2IOE}	Delay from dedicated clock pin to IOE clock	<i>Note (7)</i>
t_{DCLK2LE}	Delay from dedicated clock pin to LE clock	<i>Note (7)</i>

Symbol	Parameter	Conditions
t_{DRR}	Register-to-register delay via 4 LEs, 3 row interconnects, and 4 local interconnects	<i>Note (9)</i>

Symbol	Parameter	Conditions
t_{INSU}	Setup time with global clock at IOE register	
t_{INH}	Hold time with global clock at IOE register	
t_{OUTCO}	Clock-to-output delay with global clock at IOE register	

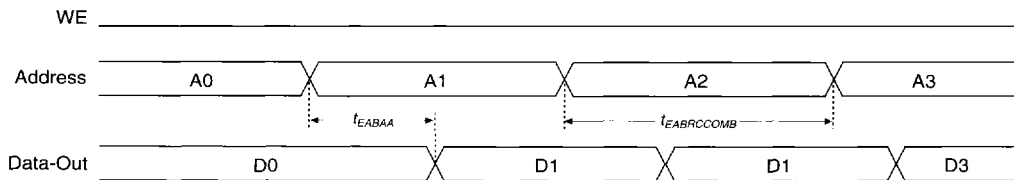
Notes to tables:

- (1) Microparameters are timing delays contributed by individual architectural elements. These parameters cannot be measured explicitly.
- (2) Operating conditions: $V_{CCIO} = 5.0\text{ V} \pm 5\%$ for commercial use.
 $V_{CCIO} = 5.0\text{ V} \pm 10\%$ for industrial or military use.
- (3) Operating conditions: $V_{CCIO} = 3.3\text{ V} \pm 5\%$ for commercial use.
 $V_{CCIO} = 3.3\text{ V} \pm 10\%$ for industrial or military use.
- (4) Operating conditions: $V_{CCIO} = 3.3\text{ V}$ or 5.0 V .
- (5) Because the RAM in the EAB is self-timed, this parameter can be ignored when the WE signal is registered.
- (6) EAB macroparameters are internal parameters that can simplify predicting the behavior of an EAB at its boundary; these parameters are calculated by summing selected microparameters.
- (7) These parameters are worst-case values for typical applications. Post-compilation timing simulation and timing analysis are required to determine actual worst-case performance.
- (8) External reference timing parameters are factory-tested, guaranteed worst-case values. A representative subset of signal paths is tested to approximate typical device applications.
- (9) Contact Altera Applications at (800) 800-EPLD for test circuit specifications and test conditions.
- (10) These external parameters are not factory-tested, however they are guaranteed.

Figures 20 and 21 show the asynchronous and synchronous timing waveforms, respectively, for the EAB macroparameters in Table 10.

Figure 20. EAB Asynchronous Timing Waveforms

EAB Asynchronous Read



EAB Asynchronous Write with Read during Write

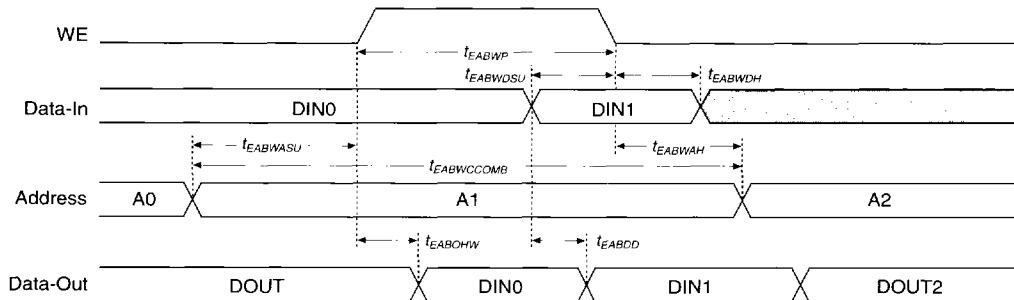
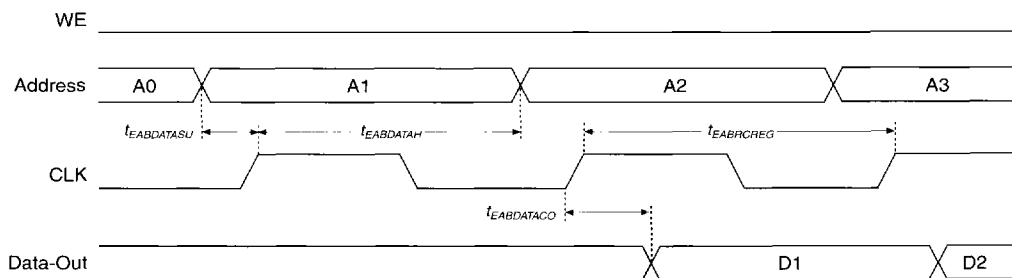
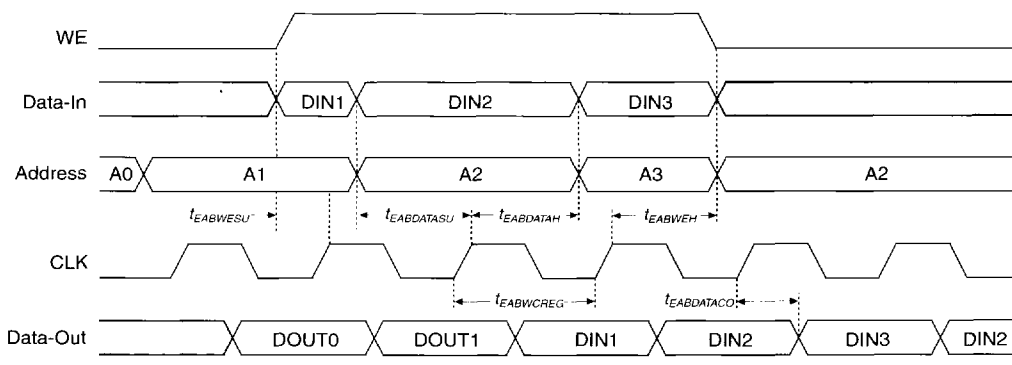


Figure 21. EAB Synchronous Timing Waveforms

EAB Synchronous Read



EAB Synchronous Write with Read during Write

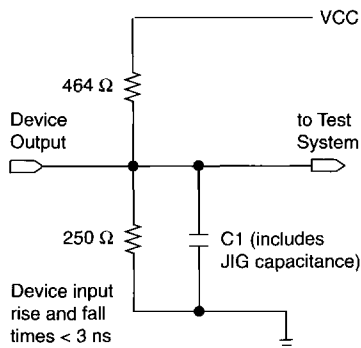


Generic Testing

Each FLEX 10K device is functionally tested and guaranteed. Complete testing of each configurable SRAM bit and all logic functionality ensures 100% configuration yield. AC test measurements for FLEX 10K devices are made under conditions equivalent to those shown in Figure 22. Multiple test patterns can be used to configure devices during all stages of the production flow.

Figure 22. FLEX 10K AC Test Conditions

Power supply transients can affect AC measurements. Simultaneous transitions of multiple outputs should be avoided for accurate measurement. Threshold tests must not be performed under AC conditions. Large-amplitude, fast-ground-current transients normally occur as the device outputs discharge the load capacitances. When these transients flow through the parasitic inductance between the device ground pin and the test system ground, significant reductions in observable noise immunity can result.



MAX+PLUS II Development System

FLEX 10K devices are supported by Altera's MAX+PLUS II development system. Designs can be entered as logic schematics with the Graphic Editor; as state machines, truth tables, conditional logic, and Boolean equations with the Altera Hardware Description Language (AHDL), VHDL, or Verilog HDL; or as waveforms with the Waveform Editor. Logic synthesis and minimization automatically optimize the logic of a design. MAX+PLUS II also provides automatic design partitioning into multiple devices from the same family. Design verification and timing analysis are performed with the built-in Simulator and Timing Analyzer. Errors in a design can be automatically located and highlighted in the original design files.

The MAX+PLUS II software runs on 486- and Pentium-based PCs, as well as Sun SPARCstation, HP 9000 Series 700, and IBM RISC System/6000 workstations. MAX+PLUS II provides an EDIF netlist interface for additional design entry and simulation support with popular EDA tools from Cadence, Intergraph, Logic Modeling, Mentor Graphics, Synopsys, Viewlogic, and others. MAX+PLUS II also exports Verilog HDL and VHDL netlist files for use with other industry-standard design verification tools.

Altera's FLEX 10K-compatible programming hardware includes the Altera Logic Programmer card, the Master Programming Unit (MPU), and various device adapters. The MPU is used to program one or more EPC1 Configuration EPROMs, which configure the FLEX 10K device on system power-up. The MPU supports continuity checking to ensure adequate electrical contact between the adapter and the device.

Configuration EPROM device adapters are shipped with the FLEX Download Cable, which facilitates the configuration of FLEX 10K devices in-circuit with Altera programming hardware and the MAX+PLUS II Programmer. FLEX 10K devices can also be configured in-circuit via a standard RS-232 serial port from a workstation or PC using the BitBlaster serial download cable.



Go to *Altera Programming Hardware* in the **1995 Data Book** for more information on programming hardware. Go to the *BitBlaster Serial Download Cable Data Sheet* in the **1995 Data Book** for more information on the BitBlaster. Go to the *MAX+PLUS II Programmable Logic Development System & Software Data Sheet* in the **1995 Data Book** for more information on the MAX+PLUS II development system.

FLEX 10K Device Absolute Maximum Ratings Note (1)

Symbol	Parameter	Conditions	Min	Max	Unit
V_{CC}	Supply voltage	With respect to GND	-2.0	7.0	V
V_I	DC input voltage	Note (2)	-2.0	7.0	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	Under bias		150	°C

FLEX 10K Device Recommended Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
V_{CCINT}	Supply voltage for internal logic and input buffers	Note (3)	4.75 (4.50)	5.25 (5.50)	V
V_{CCIO}	Supply voltage for output buffers, 5.0-V operation	Note (3)	4.75 (4.50)	5.25 (5.50)	V
	Supply voltage for output buffers, 3.3-V operation		3.00	3.60	V
V_I	Input voltage		0	V_{CCINT}	V
V_O	Output voltage		0	V_{CCIO}	V
T_A	Operating temperature	For commercial use	0	70	°C
T_A	Operating temperature	For industrial use	-40	85	°C
T_C	Case temperature	For military use	-55	125	°C
t_R	Input rise time			40	ns
t_F	Input fall time			40	ns

FLEX 10K Device DC Operating Conditions Notes (4), (5)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IH}	High-level input voltage		2.0		$V_{CCINT} + 0.3$	V
V_{IL}	Low-level input voltage		-0.3		0.8	V
V_{OH}	5.0-V high-level TTL output voltage	$I_{OH} = -4$ mA DC, $V_{CCIO} = 4.75$ V	2.4			V
	3.3-V high-level TTL output voltage	$I_{OH} = -4$ mA DC, $V_{CCIO} = 3.00$ V	2.4			V
V_{OL}	5.0-V low-level TTL output voltage	$I_{OL} = 12$ mA DC, $V_{CCIO} = 4.75$ V			0.45	V
	3.3-V low-level TTL output voltage	$I_{OL} = 12$ mA DC, $V_{CCIO} = 3.00$ V			0.45	V
I_I	Input pin leakage current	$V_I = V_{CC}$ or GND	-10		10	μA
I_{OZ}	Tri-stated I/O pin leakage current	$V_O = V_{CC}$ or GND	-40		40	μA
I_{CC0}	V_{CC} supply current (standby)	$V_I =$ GND, No load		500		μA

FLEX 10K Device Capacitance Note (6)

Symbol	Parameter	Conditions	Min	Max	Unit
C _{IN}	Input capacitance	V _{IN} = 0 V, f = 1.0 MHz		10	pF
C _{INCLK}	Input capacitance on dedicated clock pin	V _{IN} = 0 V, f = 1.0 MHz		15	pF
C _{OUT}	Output capacitance	V _{OUT} = 0 V, f = 1.0 MHz		10	pF

Notes to tables:

- (1) See *Operating Requirements for Altera Devices* in the *Altera 1995 Data Book*.
- (2) Minimum DC input is -0.3 V. During transitions, the inputs may undershoot to -2.0 V or overshoot to 7.0 V for periods shorter than 20 ns under no-load conditions.
- (3) Numbers in parentheses are for military- and industrial-temperature-range versions.
- (4) Typical values are for T_A = 25° C and V_{CC} = 5.0 V.
- (5) Operating conditions: V_{CCINT} = 5 V ± 5%, T_A = 0° C to 70° C for commercial use.
V_{CCINT} = 5 V ± 10%, T_A = -40° C to 85° C for industrial use.
V_{CCINT} = 5 V ± 10%, T_A = -55° C to 125° C for military use.
- (6) Capacitance is sample-tested only.

Figure 23 shows the typical output drive characteristics of FLEX 10K devices with 5.0-V V_{CCIO} . The output driver is compatible with the *PCI Local Bus Specification*, version 2.0.

Figure 23. Output Drive Characteristics for Devices with 5.0-V V_{CCIO}

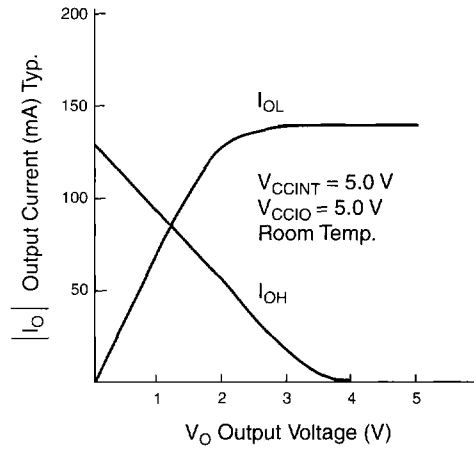
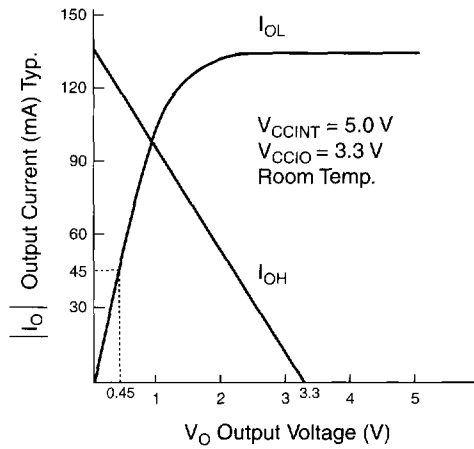


Figure 24 shows the typical output drive characteristics of FLEX 10K devices with 3.3-V V_{CCIO} . The output driver is compatible with the *PCI Local Bus Specification*, version 2.0.

Figure 24. Output Drive Characteristics for Devices with 3.3-V V_{CCIO}



FLEX 10K Internal Timing Parameters

LE Timing Microparameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
t_{LUT}	2.2		2.5		ns
t_{CLUT}		1.2		1.4	ns
t_{RLUT}		2.1		2.4	ns
t_{PACKED}		1.0		1.1	ns
t_{EN}		1.0		1.1	ns
t_{CICO}		0.4		0.5	ns
t_{CGEN}		1.8		2.0	ns
t_{CGENR}		0.2		0.3	ns
t_{CASC}		1.1		1.2	ns
t_C		1.6		1.9	ns
t_{CO}		0.3		0.3	ns
t_{CCOMB}		0.8		0.9	ns
t_{SU}	3.2		3.7		ns
t_H	0.2		0.3	0.3	ns
t_{PRE}		1.7		1.9	ns
t_{CLR}		1.2		1.4	ns

IOE Timing Microparameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
t_{IOD}		0.4		0.5	ns
t_{IOC}		0.2		0.3	ns
t_{IOCO}		0.3		0.3	ns
t_{IOCOMB}		0.3		0.3	ns
t_{IOSU}	3.5		4.0		ns
t_{IOH}	0.2		0.3		ns
t_{IOCLR}		1.6		1.8	ns
t_{OD1}		2.6		3.0	ns
t_{OD2}		3.2		3.6	ns
t_{OD3}		6.5		7.3	ns
t_{XZ}		3.5		4.0	ns
t_{ZX1}		3.5		4.0	ns
t_{ZX2}		4.1		4.6	ns
t_{ZX3}		7.4		8.4	ns
t_{INREG}		7.7		8.8	ns
t_{IOFD}		1.8		2.1	ns
t_{INCOMB}		2.1		2.4	ns

EAB Timing Microparameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
$t_{EABDATA1}$		6.4		7.3	ns
$t_{EABDATA2}$		10.0		11.4	ns
t_{EABWE1}		0.3		0.4	ns
t_{EABWE2}		8.4		9.5	ns
t_{EABCLK}		1.7		2.0	ns
t_{EABCO}		1.7		2.0	ns
$t_{EABBYPASS}$		0.3		0.4	ns
t_{EABSU}	1.9		2.1		ns
t_{EABH}	5.8		6.6		ns
t_{EABCH}	8.2		9.4		ns
t_{EABCL}	4.4		5.0		ns
t_{AA}		10.2		11.5	ns
t_{WP}	11.0		12.5		ns
t_{WDSU}	4.9		5.6		ns
t_{WDH}	4.5		5.1		ns
t_{WASU}	0.3		0.4		ns
t_{WAH}	4.8		5.5		ns
t_{WO}		16.3		18.5	ns
t_{OHV}	0.3		0.4		ns
t_{DD}		11.8		13.4	ns
t_{EABOUT}		4.8		5.4	ns

EAB Timing Macroparameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
t_{EABAA}		22.0		25.0	ns
$t_{EABRCCOMB}$		22.0		25.0	ns
$t_{EABRCREG}$		13.8		15.6	ns
t_{EABWP}	11.0		12.5		ns
$t_{EABWCCOMB}$		16.1		18.3	ns
$t_{EABWCREG}$		19.9		22.6	ns
t_{EABDD}		23.6		26.9	ns
$t_{EABDATA CO}$		8.2		9.4	ns
$t_{EABDATA SU}$	10.2		11.6		ns
$t_{EABDATA H}$	0.0		0.0		ns
$t_{EABWESU}$	8.5		9.7		ns
t_{EABWEH}	0.0		0.0		ns
$t_{EABWDSU}$	11.0		12.5		ns
t_{EABWDH}	0.0		0.0		ns
$t_{EABWASU}$	6.4		7.3		ns
t_{EABWAH}	0.0		0.0		ns
t_{EABWO}		22.0		25.0	ns
t_{EABOHV}	6.1		6.9		ns

Routing Timing Microparameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
$t_{SAMELAB}$		0.4		0.5	
$t_{SAMEROW}$		4.7		5.4	
$t_{SAMECOLUMN}$		7.2		8.1	
$t_{DIFFROW}$		11.9		13.5	
$t_{TROWROWS}$		16.6		18.9	
$t_{LEPERIPH}$		5.8		6.6	
$t_{LEGLOBAL}$		11.4		13.0	
$t_{LABCARRY}$		2.0		2.3	
$t_{LABCASC}$		2.3		2.6	
$t_{DIN2IOE}$		9.4		10.6	
t_{DIN2LE}		6.9		7.9	
$t_{DCLK2IOE}$		4.0		4.5	
$t_{DCLK2LE}$		6.5		7.4	

External Timing Parameters

External Reference Timing Parameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
t_{DDR}		23.8		27.0	ns

External Timing Parameters Note (1)					
Symbol	-4 Speed Grade		-5 Speed Grade		Unit
	Min	Max	Min	Max	
t_{INSU}	9.1		10.3		ns
t_{INH}	0.0		0.0		ns
t_{OUTCO}		7.4		8.3	ns

Note to tables:

- (1) All timing parameters are described in "Timing Model" on pages 30 through 37 of this data sheet.

Calculating Supply Current

The V_{CC} supply current for FLEX 10K devices, I_{CC} , can be calculated with the following equation:

$$I_{CC} = I_{CC\text{STANDBY}} + I_{CC\text{OUTPUT}} + I_{CC\text{ACTIVE}}$$

Typical $I_{CC\text{STANDBY}}$ values are shown as I_{CC0} in the "FLEX 10K Device DC Operating Conditions" table on page 43 of this data sheet. The $I_{CC\text{OUTPUT}}$ value, which depends on the device output load characteristics and switching frequency, can be calculated using the guidelines given in *Operating Requirements for Altera Devices* in the **1995 Data Book**. The $I_{CC\text{ACTIVE}}$ value depends on the switching frequency and the application logic. This value is calculated based on the amount of current that each LE typically consumes.

The following equation shows the general formula for calculating $I_{CC\text{ACTIVE}}$:

$$I_{CC\text{ACTIVE}} = 90 \times F \times N \times 0.125 \times \frac{\mu\text{A}}{\text{MHz} \times \text{LE}}$$

In this equation, F is the maximum operating frequency in MHz; N is the number of LEs used in the device; the constant, 0.125, is based on a 16-bit counter in which 2 out of 16 (12.5%) of the output bits switch on each Clock edge.


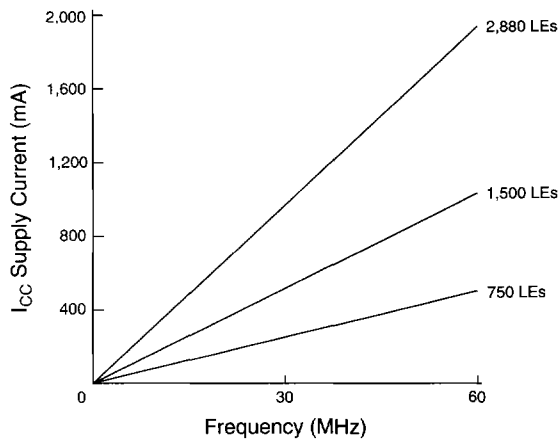
 Relative to the rest of the device, the embedded array consumes a small amount of power. Therefore, the embedded array can be ignored when calculating supply current.

Figure 25 shows the relationship between I_{CC} and operating frequency for several LE utilization values.

Figure 25. $I_{CC\text{ACTIVE}}$ vs. Operating Frequency



Configuration & Operation

The FLEX 10K architecture supports several configuration schemes to load a design into the device(s) on the circuit board. This data sheet summarizes the device operating modes and available device configuration schemes.

Operating Modes

The FLEX 10K architecture uses SRAM configuration elements that requires configuration data to be loaded every time the circuit powers up. The process of physically loading the SRAM configuration 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 FLEX 10K devices to be reconfigured in-circuit by loading new configuration data into the device. Real-time reconfiguration is performed by forcing the device into command mode with a device pin, loading different configuration data, reinitializing the device, and resuming user-mode operation. The entire reconfiguration process requires less than 200 ms and can be used to dynamically reconfigure an entire system. In-field upgrades can be performed by distributing new configuration files.

Configuration Schemes

The configuration data for a FLEX 10K device can be loaded with one of four passive configuration modes (see Table 14), chosen on the basis of the target application. An EPC1 Configuration EPROM, intelligent controller, or Joint Test Action Group (JTAG) port can be used to control the configuration of a FLEX 10K device, allowing automatic configuration on system power-up.

Multiple FLEX 10K devices can be configured in any of the four configuration modes by connecting the Configuration Enable (nCE) and Configuration Enable Output (nCEO) pins on each device.

Table 14 shows the source of data for each configuration mode.

Configuration Mode	Data Source
Passive serial (PS)	EPC1 Configuration EPROM, BitBlaster, serial data source
Passive parallel asynchronous (PPA)	Parallel data source
Passive parallel synchronous (PPS)	Parallel data source
JTAG	JTAG controller

Device Pin-Outs

Table 15 shows the pin names and numbers for the 403-pin pin-grid array (PGA) package.

Pin Name	403-Pin PGA
MSEL0 (2)	AN1
MSEL1 (2)	AR1
nSTATUS (2)	AU37
nCONFIG (2)	AU1
DCLK (2)	E1

Table 15. FLEX 10K 403-Pin PGA Package Pin-Outs (Part 2 of 2) Note (1)

Pin Name	403-Pin PGA
CONF_DONE (2)	C37
INIT_DONE	R35
nCE (2)	G1
nCEO (2)	E37
nWS (3)	E31
nRS (3)	A33
nCS (3)	A35
CS (3)	C33
DATA7 (3)	C9
DATA6 (3)	A7
DATA5 (3)	E9
DATA4 (3)	C7
DATA3 (3)	A5
DATA2 (3)	E7
DATA1 (3)	C5
DATA0 (3)	C1
TDI (2)	J1
TDO (2)	G37
TCLK (2)	A37
TMS (2)	AN37
nTRST (2)	AR37
Dedicated inputs	A17, A21, AU17, AU21
Dedicated Clock pins	A19, AU19
VCC _{INT} (5.0 V)	AD34, AE5, AL5, AM6, AM20, AN25, AN29, AP4, AT16, AT36, B2, D14, E25, F22, K36, T2, T32, V6
VCC _{IO} (5.0 V or 3.3 V)	AB36, AG5, AG33, AH2, AM18, AM32, AN11, AN27, AP24, AT22, B22, D34, E11, E27, F16, L5, L33, P4, T6, T36, V32
GND _{IO}	AB32, AD4, AM22, AN5, AN19, AN21, AN33, AP34, AT10, AT 28, B10, B28, D24, E5, E19, E33, F6, F20, K2, W5, W33, Y6
GND _{INT}	AA33, AB2, AB6, AH36, AM16, AN17, AP14, AT2, B16, B36, D4, E21, F18, F32, G33, P34, U5, Y32
Total user I/O pins	310

Notes:

- (1) All pins not listed are user I/O pins.
- (2) This is a dedicated pin; it is not available as a user I/O pin.
- (3) This pin can be used as a user I/O pin after configuration.

Package Outlines

Figure 26 shows the 403-pin PGA package outline for the EPF10K50. Package outline dimensions are shown in the following formats:

min. inches (min. millimeters)

max. inches (max. millimeters)

or:

nominal inches ± tolerance
(nominal millimeters ± tolerance)

or:

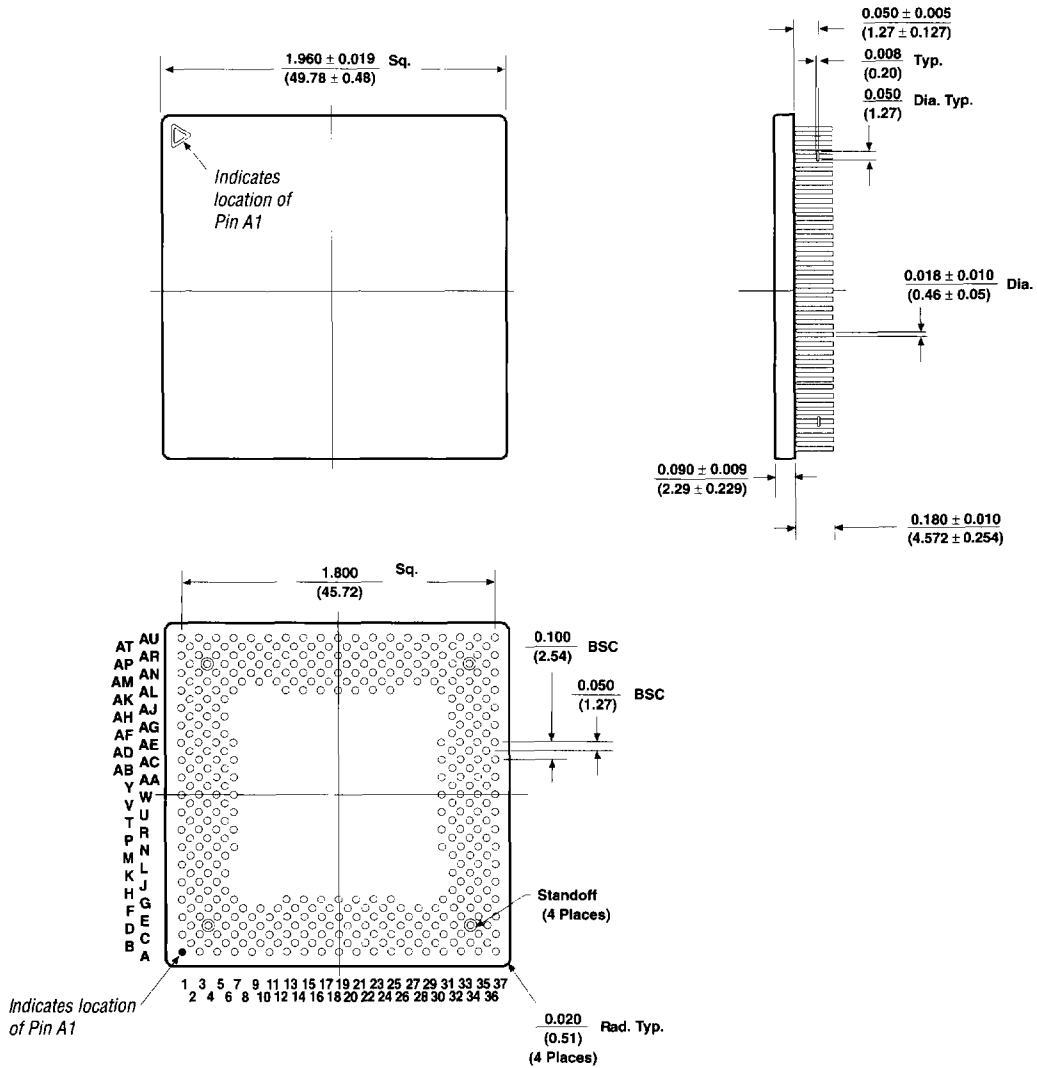
inches BSC, Min., Max., Ref., Typ., R, Dia., Sq.
(millimeters)



Go to *Altera Device Package Outlines* in the **1995 Data Book** for package outlines of 84-, 160-, 208-, 240-, and 304-pin packages. Go to *Ordering Information* in the **1995 Data Book** for information on device package ordering codes.

Figure 26. 403-Pin Pin-Grid Array (PGA)

Controlling measurement is in inches. Millimeter measurements, shown in parenthesis, are for reference only.





2610 Orchard Parkway
San Jose, CA 95134-2020
(408) 894-7000
Applications Hotline:
(800) 800-EPLD
Customer Marketing:
(408) 894-7104
Literature Services:
(408) 894-7144

Altera, MAX, MAX+PLUS, FLEX, and FLEX Ability are registered trademarks of Altera Corporation. The following are trademarks of Altera Corporation: MAX+PLUS II, BitBlaster, FastTrack, AHDL, FLEX 10K, EPF10K10, EPF10K20, EPF10K30, EPF10K40, EPF10K50, EPF10K70, EPF10K100, and EPC1. Altera acknowledges the trademarks of other organizations for their respective products or services mentioned in this document, specifically: Verilog and Verilog-XL are registered trademarks of Cadence Design Systems, Inc. Data I/O is a registered trademark of Data I/O Corporation. HP is a registered trademark of Hewlett-Packard Company. IBM and AT are registered trademarks, and IBM PC-AT and PS/2 are trademarks of International Business Machines Corporation. Pentium is a registered trademark of Intel Corporation. Mentor Graphics is a registered trademark of Mentor Graphics Corporation. SPARCstation is a trademark of SPARC International, Inc. and is licensed exclusively to Sun Microsystems, Inc. Sun Workstation is a registered trademark, and Sun is a trademark of Sun Microsystems, Inc. Synopsys is a registered trademark of Synopsys, Inc. Viewlogic is a registered trademark of Viewlogic Systems, Inc. Altera products marketed under trademarks are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

U.S. and European patents pending.

Copyright © 1995 Altera Corporation. All rights reserved.



I.S. EN ISO 9001