Supplement to

Logic and Computer Design Fundamentals

4th Edition[1]

VLSI PROGRAMMABLE LOGIC DEVICES: XILINX AND ALTERA FAMILIES

This reading supplement covers specific families of VLSI programmable logic devices from two manufacturers, Xilinx®, Inc. and Altera® Corporation. The device families chosen are those most likely to be used in beginning logic laboratories that use PLDs. This supplement is referenced at the end of Chapter 6.

Coverage of a VLSI PLD family is strongly recommended if the course has an associated laboratory component using the family. Coverage of one or both of the VLSI PLD families is recommended as a basic introduction to VLSI PLDs.

The advantage of using a PLD in the design of digital systems is that it can be programmed to incorporate complex logic functions within a single IC. But for larger or more complex functions, VLSI technology is appropriate. VLSI (Very Large Scale Integrated) refers to digital systems that contain thousands to millions of gates within a single IC chip.

In the last two decades, VLSI approaches have been developed for PLDs to handle designs that in the past were implemented by many small chips or with gate arrays having from 1,000 to millions of gates. The new approaches yield high-capacity programmable logic devices typically sharing the following properties:

1. substantial amounts of uncommitted combinational logic;
2. pre-implemented flip-flops and/or latches;
3. programmable interconnections between the combinational logic, flip-flops, and the chip input/outputs;
4. memories for storing information; and
5. a volatile or non-volatile programming technology.

---

[1]

Aside from these properties common to all VLSI PLDs, the devices differ from vendor to vendor. Field-Programmable Gate Arrays (FPGAs) has become a generic term referring to SRAM-based VLSI PLDs. To illustrate VLSI PLDs, we present an overview of two medium-density FPGA families that are most frequently used in basic undergraduate course laboratories.

XILINX® SPARTAN™-II FIELD-PROGRAMMABLE GATE ARRAYS[2]

Xilinx Spartan-II Field-Programmable Gate Arrays (FPGAs) include two major families: 1) The basic Spartan-II family and the enhanced Spartan-IIE family. There are many important differences in these families, including electrical characteristics and propagation delays. However, these parts with the same basic part numbers (except for the E) have similar architectures and logical structures. As a consequence, the discussion given here, while based on the basic Spartan-II family, applies as well to most of the features of the enhanced Spartan-IIE family as well. Significant differences will be pointed out in the discussion.

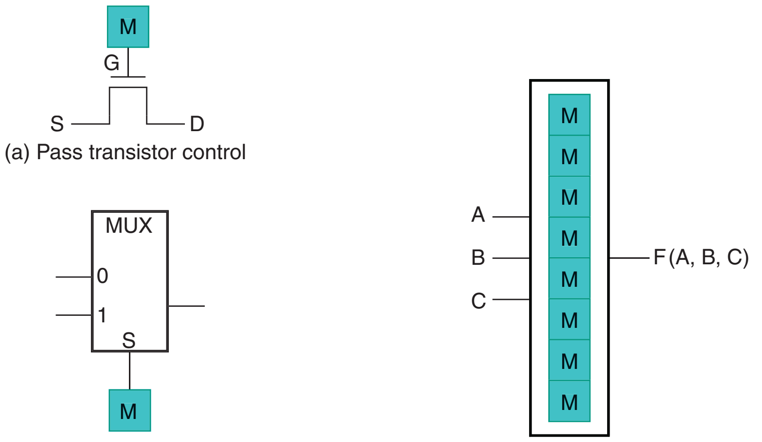
Xilinx Field-Programmable Gate Arrays (FPGAs) use SRAM technology to store the programming information. After power is applied to the circuit, the program data defining the logic configuration must be loaded into the FPGA SRAM. There are a number of different ways of loading the information, a process referred to as *configuration*. Once the programming information is loaded, the FPGA switches from the programming mode to the operational mode in which the logic is available for use. The logic remains until either the FPGA is reprogrammed or the power is turned off. The ability to reprogram the FPGA allows different logic to be implemented in a system by the same FPGA at different times, leading to the concept of *reconfigurable systems.*

Values loaded into SRAM bits during configuration control the logic implemented in a Xilinx FPGA. Three techniques, illustrated in Figure 1 (pass transistor control, multiplexer control, and lookup table implementation) are used to convert the stored 0's and 1's into logic. In addition, a portion of the SRAM bits reside in Block RAMs that are accessible to the user. Bits stored in a Block RAM during configuration permits the RAM to be used as a ROM. Otherwise, Block RAM can be can be configured to act to act as a RAM in a user's design.

Figure 1(a) shows an SRAM cell driving the gate (G) terminal of an n-channel MOS transistor. This transistor acts like a switch. If an H (logic 1) is applied by the SRAM cell, then the path between the two other terminals of the n-channel transistor is CLOSED, permitting a current to flow between the two connected wiring segments. If an L (Logic 0) is applied, the path between the two other terminals is OPEN, preventing current flow. When such a transistor is to make a bidirectional connection for the passage of a signal between two wiring segments, it is

---

[2] Xilinx is a registered trademark of Xilinx, Inc and Spartan is a trademark of Xilinx, Inc. This material in this section is intended for educational use only. For all other uses, consult the documentation available from the Xilinx, Inc. website listed in the References.



(a) Pass transistor control

(b) Multiplexer control

(c) Look up table implementation

FIGURE 1
SRAM Bit Use in Xilinx® FPGAs

called a *pass transistor*. A Xilinx FPGA typically contains thousands to millions of such transistors in its interconnection structure.

In Figure 1(b), an SRAM cell is attached to the select input *S* of a 2–to–1 multiplexer. If the SRAM cell contains a 0, then the value on the 0 input of the multiplexer is passed to the multiplexer output. If the SRAM cell contains a 1, then the value on the 1 input is passed to the multiplexer output. The structure is used to make selections between two signals. Sometimes there are two SRAM cells driving a 4–to–1 multiplexer. Finally, in cases where the data inputs to the 0 and 1 inputs are X and $\overline{X}$, respectively, the multiplexer symbol is replaced by an XOR gate with X applied to one input and the SRAM cell output applied to the other.

The final use of SRAM cells is to build a lookup table, as in Figure 1(c). In the figure, a lookup table for a three-variable function $F(A, B, C)$ is illustrated. The SRAM cells in the table store the actual truth table of the function, so each cell contains the value of function F for the corresponding minterm. The lookup table is functionally equivalent to a multiplexer with the SRAM bits applied to the data inputs and the input variables A, B, and C on the selection inputs. For example, if $(A, B, C) = 0\ 1\ 0$, the value in SRAM cell 2 (binary 010) appears on the output of the circuit. So the lookup table is conceptually a multiplexer implementation of combinational logic, (as discussed in Chapter 3), with the SRAM cells providing the data inputs.

Architecture

The Xilinx Spartan-II FPGA structure is shown in Figure 2. The logic within the FPGA is implemented in an array of programmable blocks of logic called configurable logic blocks (CLBs) and Block RAMs for storing information. The Block RAMs are SRAMs that are available for use in implementing RAM and ROM in designs. Inputs to and outputs from the array of blocks are handled by input/output blocks (IOBs) along the edges of the array. The CLBs and IOBs are interconnected by a variety of programmable interconnection structures. By using an array of programmable connection blocks referred to generically as switch matrices, connections to and from CLBs and IOBs can be programmed and wire segments can be interconnected to form paths from one block to another.

On the corners of the structure are four delay-locked loops (DLLs). These are used to align the edges of internal clock signals at the flip-flop inputs with the edges of an external clock. This approach creates a synchronous circuit that includes an FPGA and its environment or multiple FPGAs and their environment. In addition, the DLLs provide multiples of the external clock with different frequencies that are useful in more complex designs.

Logic

Most of the logic circuits in a Xilinx FPGA lie within the CLBs and the IOBs. Both of these structures are internally programmable and fairly complex. We will look in detail at the CLB and then sketch the main features of the IOB.

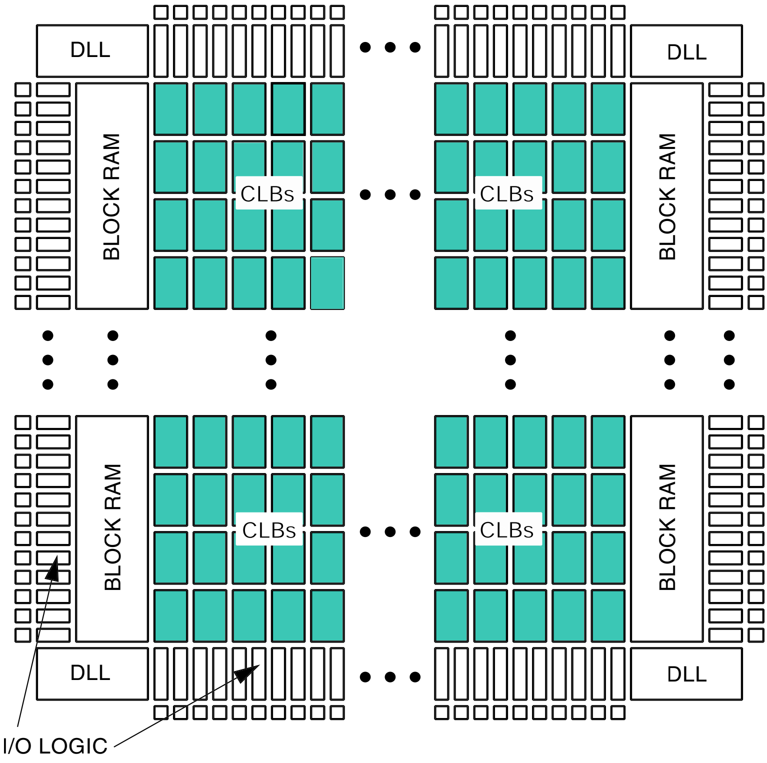


FIGURE 2
Xilinx Spartan-II FPGA Structure (Adapted with Permission of Xilinx, Inc.)

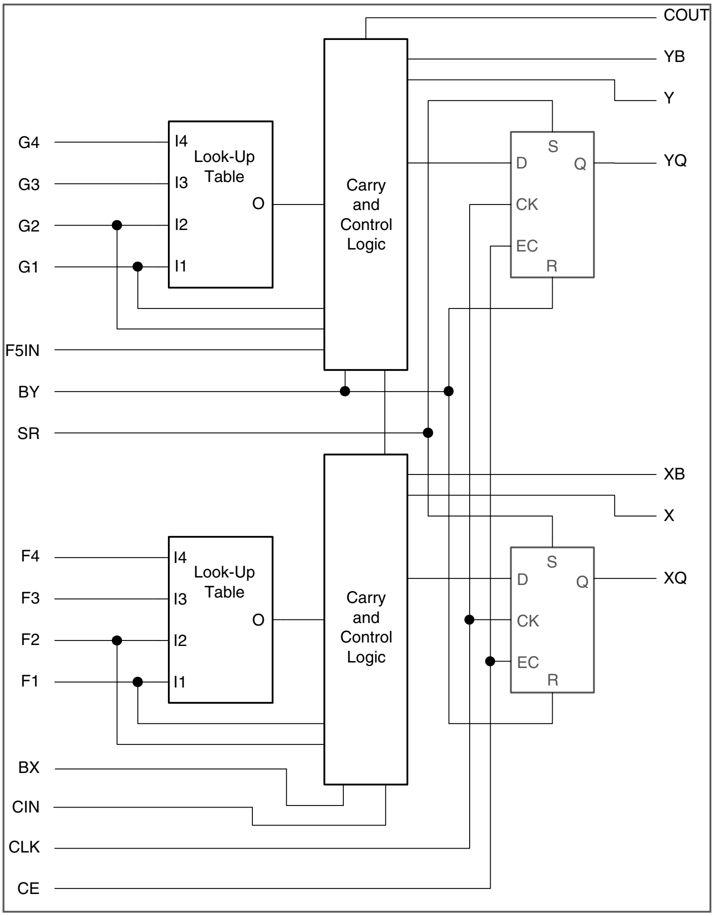


FIGURE 3
Simplified Diagram of a Xilinx Configurable Logic Block (CLB) (Adapted with Permission of Xilinx, Inc.)

CONFIGURABLE LOGIC BLOCKS (CLBS) A simplified diagram of a slice of a Xilinx Spartan-II CLB is shown in Figure 3. There are two slices in each CLB. Each slice has 15 inputs and 7 outputs. A slice contains two lookup tables (LUTs) that implement 4-input, 1-output combinational functions. One has G4, G3, G2, and G1, as inputs. The other has F4, F3, F2, and F1 as inputs. Each lookup table output enters a Carry and Control Logic block which has additional inputs. Each Carry and Control Logic block contains high-speed carry logic, an XOR gate, and an AND gate that in combination with the LUT implement one bit of arithmetic functions such as addition, counting, and multiplication. In association with this logic, each block has a carry-in and carry-out signal. The carry-in for the lower block and the carry out for the upper block are attached to other CLBs and the carry-out from the lower block is attached to the carry-in for the upper block. These carry signals and logic can also be used to cascade function generator outputs for implementation of logic functions with large numbers of inputs. Each Carry and Control Logic block also contains a number of multiplexers that select the outputs of the block from among the various inputs. Figure 4 shows three additional multiplexers that appear in each CLB. There is one F5 multiplexer in each slice. The two F5 multiplexer outputs in a CLB are the data inputs to the F6 multiplexer. By applying variables BX from each slice to the select inputs of the F5 multiplexers, two additional variables beyond those feeding the four LUTs are introduced. The output of an F5 multiplexer, can be any function of five variables or a restricted class of functions of up to nine variables. The output of the F6 multiplexer can be any function of six variables or a restricted class of functions up to 19 variables. The F5 and F6 multiplexers are included as a part of the Carry and Control Logic blocks in the two CLB slices.

Each CLB slice contains two storage elements that can be configured to be either edge-triggered D flip-flops or a level-sensitive latches. The D inputs are driven by the respective outputs of the Carry and Control Logic blocks. These outputs can be selected from a number of sources by multiplexers within the Carry and Control Logic blocks. In addition to shared Clock and Clock Enable signals, shared signals SR and BY are available for synchronous sets and resets. SR forces the storage element into the initialization state corresponding to the specified configuration and BY forces it into the opposite state if specified by the configuration. It is also possible to configure the storage element to provide asynchronous sets and resets using these signals. All of the storage element control inputs can be configured to be inverted or not.

There are seven outputs from a CLB slice, X, XB, XQ, Y, YB, YQ, and COUT. All of these outputs come from multiplexers in the Carry and Control Logic blocks fed by multiple sources. All of the outputs except for XQ and YQ are

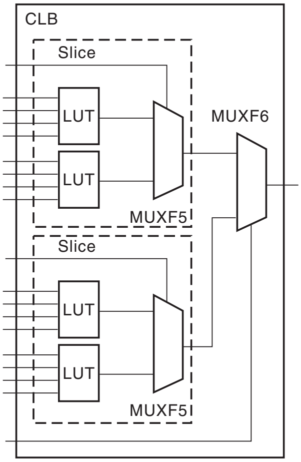


FIGURE 4
F5 and F6 Multiplexers (Adapted with Permission of Xilinx, Inc.)

combinational functions of the inputs. XQ and YQ are storage element outputs. F5 is an additional slice output, used internally to feed the other CLB slice input F5IN and is not a CLB output. Instead the value of the two F5 multiplexers can be configured to appear on X and Y outputs, and the value of the F6 output can be configured to appear on the Y output.

The LUTs can alternatively be used as RAM. With the utilization of write logic, each LUT implements 16 1-bit words of SRAM. Two LUTs within a slice can be combined to give a 16 2-bit word synchronous SRAM, a 32 1-bit word synchronous SRAM, or a 16 1-bit word dual-port synchronous SRAM. Alternatively, an LUT can form a 16-bit chain of flip-flops that can be used to shift data.

INPUT/OUTPUT BLOCKS (IOB) The Xilinx IOB is also programmable and offers the designer a number of choices. We will briefly sketch its primary features. To simplify the explanation, we consider the output and input portions of the IOB separately, as shown in Figure 5. A block labeled Electronic Interface Components lies between the single I/O pin and output and input lines. This block contains many features, a number of which are programmable, that govern electrical characteristics of the output and the sampling of the input value. The details of this block require some understanding of electrical and electronic circuits and are beyond the scope of our coverage.

The output portion of the IOB can provide output data O from the interior of the FPGA on the I/O pin. Alternatively, it can provide a stored value of output data O from a flip-flop with output Q. A 3-state driver on the output allows the I/O pin to be used as an input, an output, or an input/output. The control signal T for the three-state driver has the same alternative output capabilities as the output data signal O.

In the input portion of the IOB, the signal at the I/O pin enters an input buffer. The buffer output optionally passes through a programmable delay that provides a zero hold time on the input to insure that it can be properly captured. The output of the delay appears on input I to the FPGA interior and can be stored in a flip-flop with input IQ provided to the FPGA interior.

Block RAM

The Spartan II contains several Block RAMs organized in two columns along the left and right edges of the diagram in Figure 2. The number of Block RAMs per column ranges from 4 to 36 depending on the size of the FPGA. Each Block RAM contains $2^{12} = 4{,}096$ bits of storage and can be configured to have 2^m words of 2^n bits with $m + n = 12$ and $n \leq 4$. The Block RAMs are synchronous with a clock and can be configured to be ROM or RAM and to have single or dual ports. A single port RAM has just one set of control, address, and data inputs and one data output. A dual port RAM has two sets of control, address, data inputs and data outputs for reading and writing data. In either case, the Block RAM is synchronized with one or two clocks, and cannot be used directly as a combinational component.

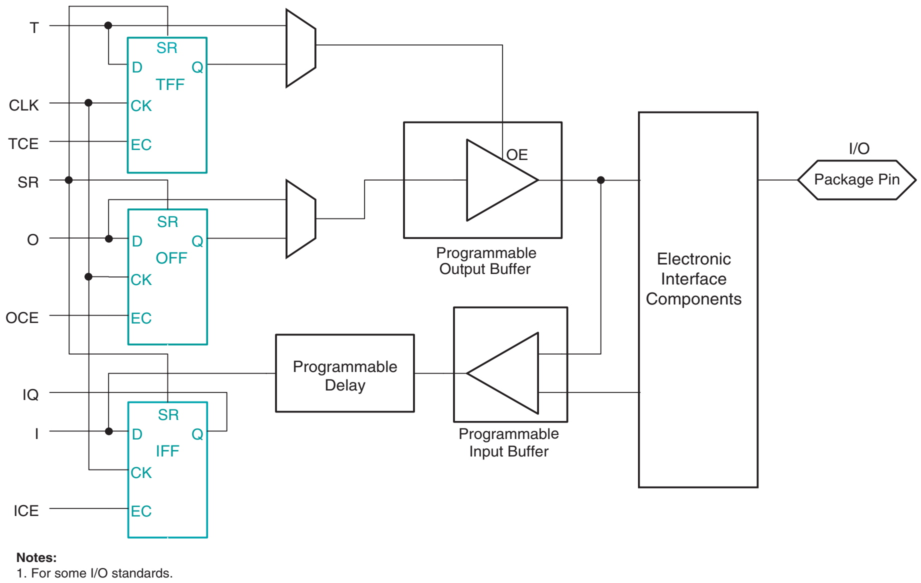


FIGURE 5
Sketch of Xilinx IOB Structure (Adapted with Permission of Xilinx, Inc.)

Interconnections

There are four major types of interconnections in the Spartan-II FPGA: 1) local routing, 2) general-purpose routing, 3) global routing, and 4) VersaRing routing. Local routing in the vicinity of a single CLB is shown in Figure 6. There are connections within the CLB that connect the internal lookup tables (LUTs) and fast direct connections to the CLBs to the left and right. Local routing connections lie in both directions between each CLB and the General Routing Matrix (GRM) to its upper left. A GRM provides connections between its CLB and the general-purpose routing segments and between general-purpose routing segments attached to the GRM. The general routing matrix is a switch matrix similar to that shown in Figure 7(a). Where four segments meet, there are six pass transistors—one vertical, one horizontal, and four on the diagonals. Each pass transistor is represented by a green line. The connection between two segments is CLOSED for a 1 stored in the SRAM cell driving the gate of the transistor. The connection between two segments is OPEN for a 0 stored in the SRAM cell. Several connections are shown in Figure 7(b). Note that at point 1 all four segments are joined together by closing three transistors. In this case, all six transistors could be closed to make a connection with less electrical resistance. At point 2, two distinct signal paths pass through a single set of pass transistors. Wiring segments from the matrix inputs and outputs extend across adjacent wiring channels. SRAM-controlled pass transistors lie at selected intersections between these segments and perpendicular wiring segments

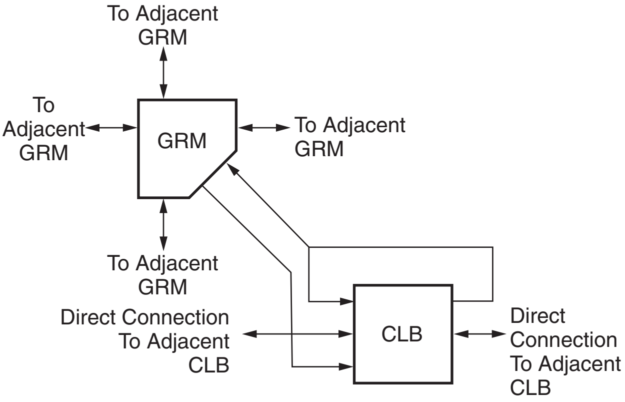
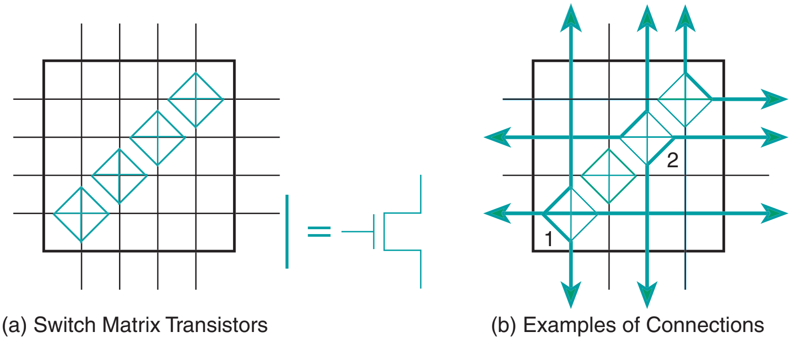


FIGURE 6
Spartan-II Local Routing (Adapted with Permission of Xilinx®, Inc.)

in the channels. The GRM connects to wiring segments in both horizontal and vertical channels lying between the CLBs. Some of the segments are very long, spanning the entire length or width of the array. Other segments span a single CLB or six CLBs. In addition to the wiring segments connected together by GRMs, there are global routing interconnections for clocks and, from each CLB, two 3-state buffers as inputs to some of the horizontal lines.

Design Methodology

The overall structure of the interconnections, CLBs, and IOBs, and Block RAM is clearly quite complicated. A designer having to deal with hundreds of CLBs and IOBs and thousands of interconnection points in such an FPGA would have a very difficult job. As a consequence, CAD tools are provided that take a design in the form of a schematic or HDL description, automatically partition the design into



(a) Switch Matrix Transistors
(b) Examples of Connections

FIGURE 7
Example of Xilinx Switch Matrix (Adapted with Permission of Xilinx, Inc.)

pieces that fit within a CLB, place the pieces into specific CLBs and route the connections between the CLBs, and to and from IOBs. The end result of this process is thousands of bits of programming information that can be loaded into the FPGA to implement the desired logic.

ALTERA® FLEX 10K® EMBEDDED PROGRAMMABLE LOGIC DEVICES[3]

The Altera Flex 10K Embedded Programmable Logic Devices (EPLDs) includes two major families: 1) The basic 10K family and the enhanced 10KE family. There are many important differences in these families, including electrical characteristics and propagation delays. However, these parts with the same basic part numbers (except for the E) have similar architectures and logical structures. As a consequence, the discussion given here, while based on the basic 10K family, applies as well to most of the features of the enhanced 10KE family as well. Significant differences will be pointed out in the discussion.

Altera Flex 10K EPLDs use SRAM technology to store the programming information. After power is applied to the circuit, the program data defining the logic configuration must be loaded into the EPLD SRAM. There are a number of different ways of loading the information, a process referred to as *configuration*. Once the programming information is loaded, the EPLD switches from the programming mode to the operational mode in which the logic is available for use. The logic remains until either the EPLD is reprogrammed or the power is turned off. The ability to reprogram the EPLD allows different logic to be implemented in a system by the same EPLD at different times, leading to the concept of *reconfigurable systems*.

Values loaded into SRAM bits during configuration control the logic implemented in an Altera EPLD. Three techniques, illustrated in Figure 8 (multiplexer control, gate control, and lookup table implementation) are used to convert the stored 0's and 1's into logic. In addition, a portion of the SRAM bits reside in the embedded array blocks (EABs) and can be used as stored information that is fixed or dynamic during normal EPLD operation. An embedded array block with fixed contents is used as a ROM, which can implement complex combinational logic, and with variable contents is used as a RAM.

In Figure 8(a), an SRAM cell represented by a small circle is attached to the select input S of a 2–to–1 multiplexer. If the SRAM cell contains a 0, then the value on the I0 input of the multiplexer is passed to the multiplexer output. If the SRAM cell contains a 1, then the value on the I1 input is passed to the multiplexer output. The structure is used to make selections between two signals. Sometimes there are two SRAM cells driving a 4–to–1 multiplexer.

---

[3] Altera and Flex 10K are registered trademarks of Altera Corporation. FastTrack is a trademark of Altera Corporation. This material in this section is intended for educational use only. For all other uses, consult the documentation available from the Altera Corporation website listed in the References.

In Figure 8(b), an OR gate with one input attached to an SRAM cell is shown. If the SRAM cell contains a 0, then the signal on the other input to the OR gate is pass through to the gate output. If the SRAM cell contains a 1, then the output to the OR gate is a fixed 1. This is one form of enabling circuit, as introduced in Chapter 3. An AND gate with one input attached to an SRAM cell is also used.

The final specialized use of SRAM cells is to build a lookup table, as in Figure 8(c). In the figure, a lookup table for a three-variable function *F(A, B, C)* is illustrated (The actual lookup tables in a Flex 10K EPLD implement four-variable functions). The SRAM cells in the table store the actual truth table of the function, so each cell contains the value of function *F* for the corresponding minterm. The lookup table is functionally equivalent to a multiplexer with the SRAM bits applied to the data inputs and the input variables *A*, *B*, and C on the selection inputs. For example, if (*A*, *B*, *C*) = 0 1 0, the value in SRAM cell 2 (binary 010) appears on the output of the circuit. So the lookup table is actually a multiplexer implementation of combinational logic, as discussed in Chapter 3, with the SRAM cells providing the data inputs.

Architecture

The Altera® Flex 10K™ EPLD structure is shown in Figure 9. The logic within the EPLD is implemented in an array of Logic Array Blocks (LABs) and SRAMs are implemented within Embedded Array Blocks (EABs). An EAB also contains logic that permits it to be treated as a ROM, a memory composed of latches, or a memory composed of flip-flops. Inputs to and outputs from the EPLD are handled by Input/Output Elements (IOEs) along the edges of the EPLD. The LABs and IOEs are interconnected using horizontal rows of fixed wires and vertical columns of fixed wires. A wire are referred to as a *channel*. There are programmable connections between the channels and the LABs, the channels and EABs, and the

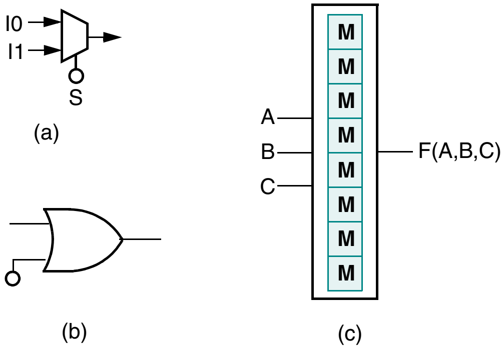


FIGURE 8
SRAM Bit Use in Altera® EPLDs

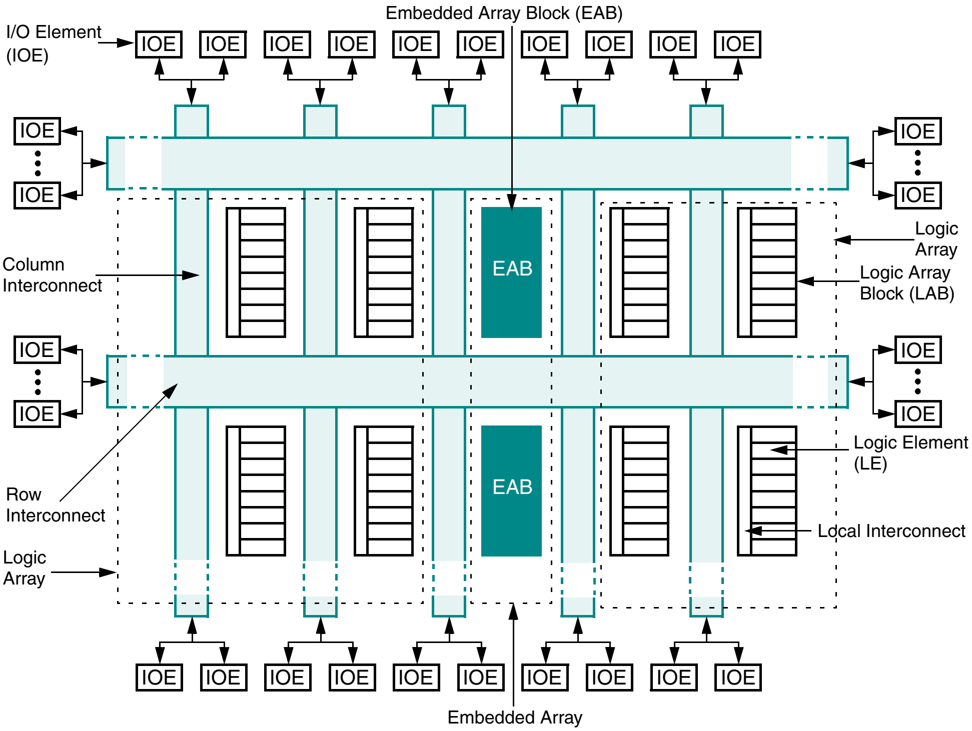


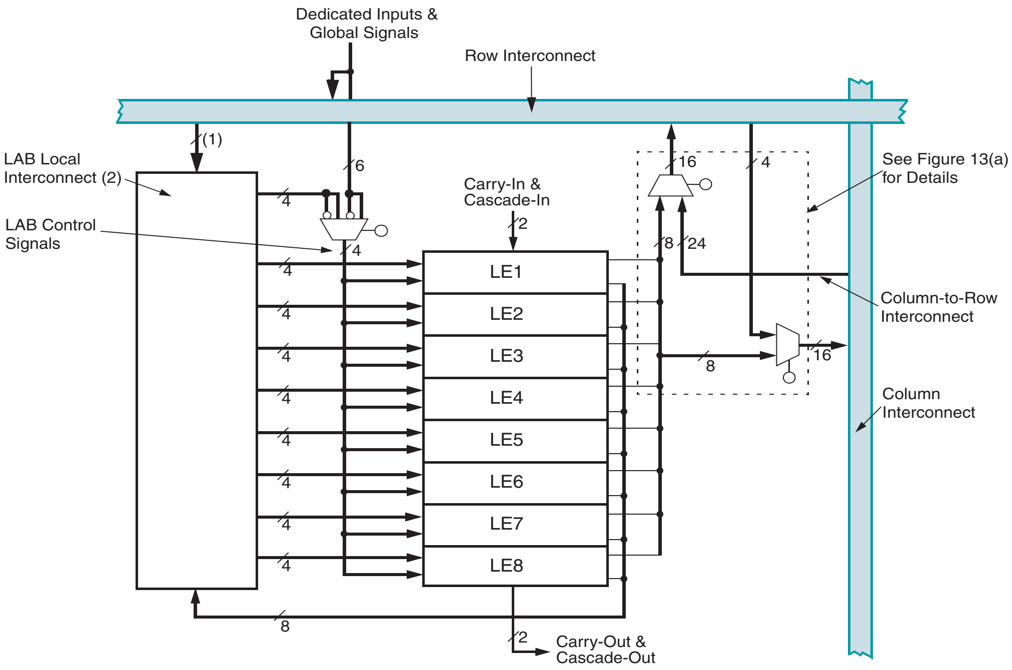
FIGURE 9
Altera Flex 10K EPLD Structure (Reprinted with permission of Altera Corporation. © 2004 Altera Corporation)

channels and the IOEs. In addition, there are programmable connections between the row interconnect channels and the column interconnect channels.

Logic

The logic circuits in the Altera Flex 10K EPLD lie within the LABs and the IOEs. Both of these structures are internally programmable and fairly complex. We will look in detail at the LAB and then sketch the main features of the IOE.

LOGIC ARRAY BLOCK (LAB) A diagram of a LAB appears in Figure 10. The core of the LAB consists of eight logic elements (LEs), each of which contains an identical block of implementation logic. In order to interconnect the LEs to each other, LAB local interconnect is provided. In addition, there is a set of four signals that enter all of the LEs which are used for controlling the LE storage element. These four signals can be driven externally or from the LAB local interconnect. The outputs of the LEs can be attached to row interconnect above and the column interconnect to the right of LAB. Connections can also be made within the LAB from column interconnect to row interconnect and from row interconnect to column interconnect. Finally, there are a Carry-in and a Carry-out for implementing arith-



Notes:
(1) EPF10K10, EPF10K10A, EPF10K20, EPF10K30, EPF10K30A, EPF10K40, EPF10K50, and EPF10K50V devices have 22 inputs to the LAB local interconnect channel from the row; EPF10K70, EPF10K100, EPF10K100A, EPF10K130V, and EPF10K250A devices have 26.
(2) EPF10K10, EPF10K10A, EPF10K20, EPF10K30, EPF10K30A, EPF10K40, EPF10K50, and EPF10K50V devices have 30 LAB local interconnect channels; EPF10K70, EPF10K100, EPF10K100A, EPF10K130V, and EPF10K250A devices have 34.

FIGURE 10
Diagram of an Altera Flex 10K Logic Array Block (LAB) (Reprinted with permission of Altera Corporation. © 2004 Altera Corporation)

metic functions and a Cascade-in and Cascade-out for implementing a limited class of functions with large numbers of inputs.

LOGIC ELEMENT (LE) A diagram of the Flex 10K LE appears in Figure 11. A lookup table (LUT) is provided for implementing a 4-input, 1-output combinational function. The lookup table is attached to Carry Chain block which also has a Carry-in from the LE above it and a Carry-out to the LE below it. The combination of the LUT and the Carry Chain block provides two 3-input functions that implement the sum and carry functions of a full-adder. In the Cascade Chain block, the LUT output is ANDed with Cascade-in.

Each LE contains a storage element that can be configured to be either an edge-triggered D flip-flop or a level-sensitive latch. The D input is driven by a multiplexer that can be programmed to select between the output of the Cascade Chain block and the LE input data, permitting the storage element to be used with or independently of the LE combinational logic. The clock for the storage element

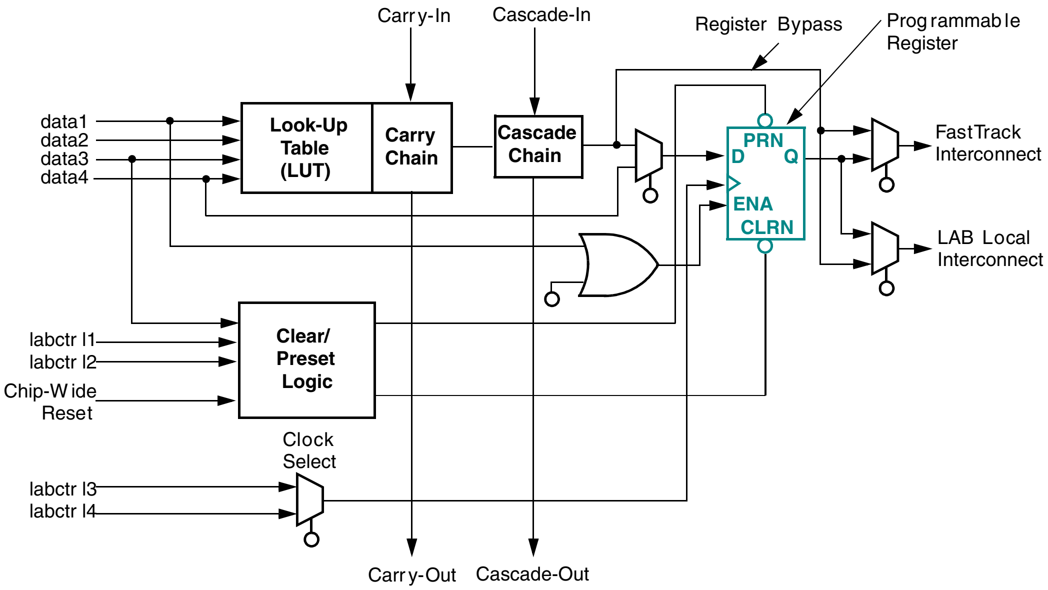


FIGURE 11

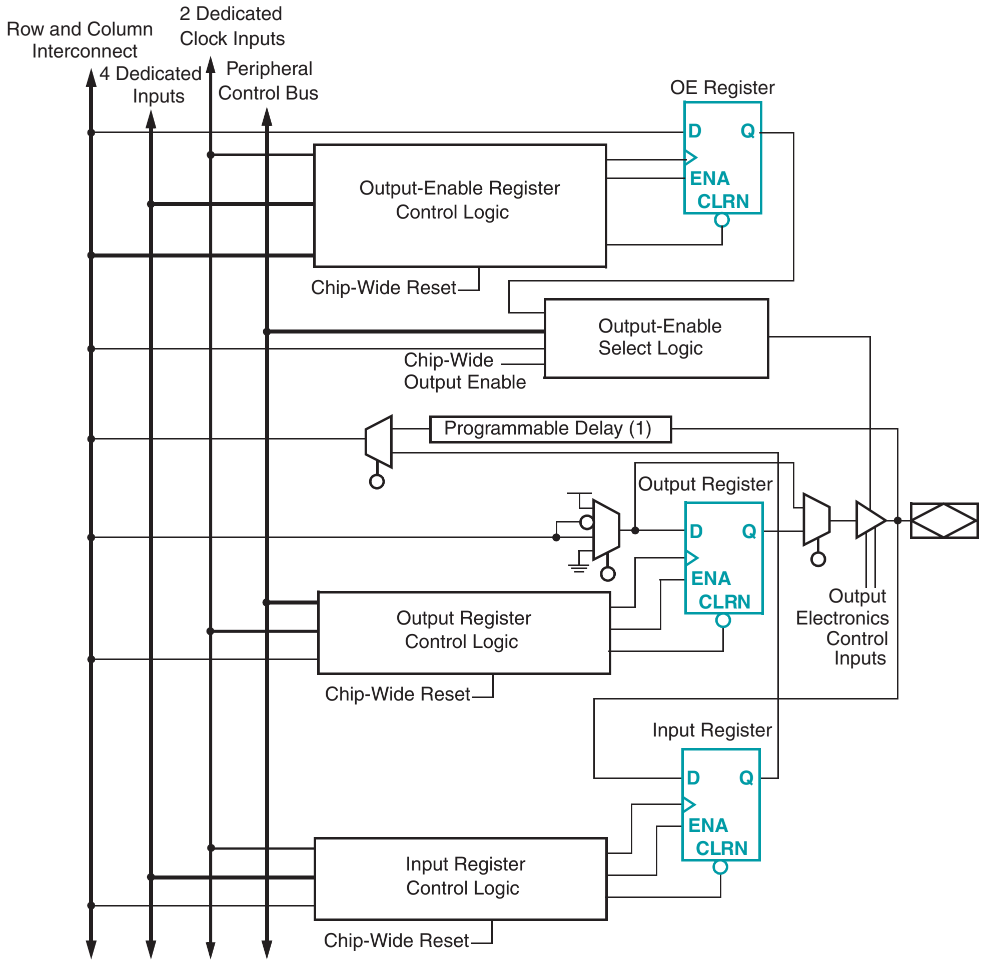
Diagram of an Altera Flex 10K Logic Element (LE) (Reprinted with permission of Altera Corporation. © 2004 Altera Corporation)

can be selected by a programmed multiplexer from two control inputs. The storage element also has Clear and Preset inputs that are control by Clear/Preset logic having a mixture of data and control inputs.

Two of the outputs from the LE attach to the row and column interconnect channels (called FastTrack™ interconnect) and to the local interconnect respectively. Two outputs are used to allow independent selection using programmed multiplexers from two sources. One source, an output of the Cascade Chain, is combinational and the other, the output of the storage element, is sequential. The other two outputs of the LE are Carry-out and Cascade-out.

INPUT/OUTPUT ELEMENT (IOE) The Flex 10K Input/Output Element (IOE) shown in Figure 12 has its I/O pin driven by a three-state output buffer and permits the signal on the pin to be used as an input. This permits the pin to be configured as an output, an input, or a bidirectional connection. There are three primary internal signals associated with I/Os: 1) the input signal from the I/O pin, 2) the output signal that goes to the 3-state buffer, and 3) the output enable signal for the 3-state buffer. Each of these signals may be optionally driven by a local storage register or may be driven by signals from elsewhere within the EPLD. The storage registers each have a clock, a load enable input, and a direct clear input.

The combinational logic within an IOE selects 1) whether or not each of these three internal signals is driven by a local storage register or by a signal from elsewhere and 2) selects the sources for the signals that control the local storage registers. A simplified version of this logic appears in Figure 12. The inputs to this



Note:

(1) All FLEX 10KE devices (except the EPF10K50E and EPF10K200E devices) have a programmable input delay buffer on the input path.

FIGURE 12

Simplified Diagram of an Altera Flex 10K Input/Output Element (IOE) (Adapted with permission of Altera Corporation. © 2004 Altera Corporation)

logic includes data signals, control signals, and SRAM-configuration bits. Beginning at the input signal from the I/O pin, there are two paths to the row or column interconnect selected by a multiplexer controlled by an SRAM bit. One of the multiplexer inputs is the input signal itself. For some parts in the 10K family and most of the parts within the 10KE family, there is a programmable delay available in this path that is used to delay the input so that the hold time with respect to the clock is zero. The other multiplexer input is the output of the Input register. In addition, there is control logic for selection of the input register clock, enable, and clear signals. The inputs to this logic come from row and column interconnect, four dedicated inputs, two dedicated clock lines, and a constant 1. The selection process is controlled by SRAM bits. In addition, the result of the selection for the clear is ANDed with the Chip-wide Reset to produce the clear output signal from this logic.

Beginning at the input to the 3-state buffer, two multiplexers controlled by SRAM bits are used to select whether or not the output signal is stored in the output register and to select one of four signals to act as the output and the input to the output register. There four signals are a constant 1, a constant 0, and true and inverted versions of a signal from the row and column interconnect. In addition, there is control logic for selection of the output register clock, enable, and clear signals. The inputs to this logic come from the row or column interconnect, the peripheral control bus, two dedicated clock lines, and a constant 1. The selection process is controlled by SRAM bits. In addition, the result of the selection for the clear is ANDed with the Chip-wide Reset to produce the clear output signal from this logic. Finally, for the output control enable input to the 3-state buffer, there is selection logic controlled by SRAM bits for selecting from the row or column interconnect, the peripheral control bus, or the output of the OE register. The input of the OE register can come only from the row or column interconnect. Also, the Chip-wide Reset is ANDed with the result of the selection for the clear to produce the clear output signal from this logic. In addition, there is control logic for selection of the output register clock, enable, and clear signals. The inputs to this logic come from the row or column interconnect, four dedicated inputs, two dedicated clock lines, and a constant 1. The selection process is controlled by SRAM bits.

Embedded Array Block

The Flex 10K parts contain several Embedded Array Blocks (EABs). Each EAB in the 10K parts contains $2^{11} = 2{,}048$ bits of storage and in the 10KE parts contains $2^{12} = 4{,}096$ bits of storage. The 10K EABs can be configured to have 2^m words of 2^n bits with $m + n = 11$ and $0 \le n \le 3$ and the 10KE parts can be configured to have 2^m words of 2^n bits with $m + n = 12$ and $1 \le n \le 4$. The EABs can be configured as a number of useful functional blocks for system design. The simplest is as a ROM, effectively using the EAB as a large LUT. A second configuration is as a RAM which has a single port consisting of control, address, and data inputs and one data output. A dual-port RAM configuration, which has two sets of control, address, data inputs and data outputs for reading and writing data, is available in the 10KE family. Data inputs and data outputs for a port can be merged into a single bidirectional data input/output if desired. The clock signals, read and write enable signals, and use of registers on inputs and outputs is flexible and programmable.

FastTrack™ Interconnections

The row interconnection channels are divided into two groups, full-length channels and half-length channels. A full length channel can be connected to all LABs in a

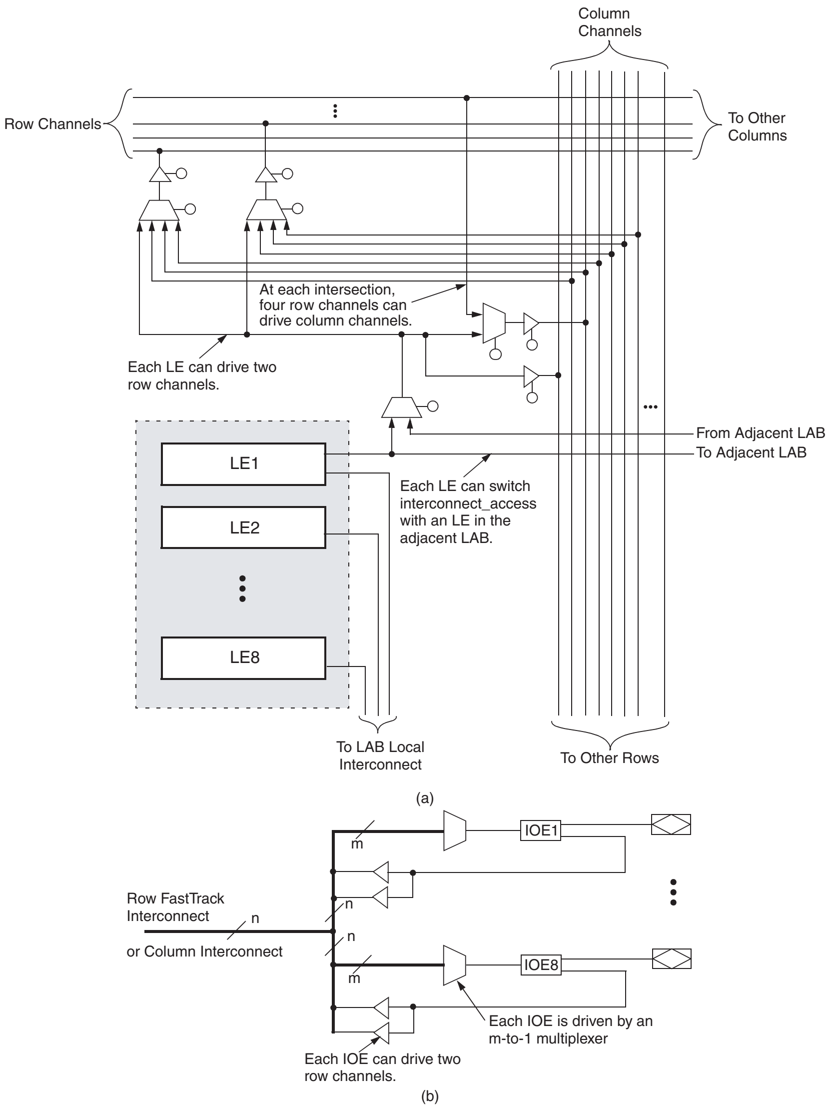


FIGURE 13
Diagrams of Altera Flex 10K Interconnect (Reprinted with permission of Altera Corporation. © 2004 Altera Corporation)

row. A half-length channel can be connected to all LABs in the left half or right half of the array. Two diagrams showing the connections associated with the row and column interconnection structure are given in Figure 13.

The programmable connections between an LE, an interconnect row and an interconnect column all lie within a LAB are shown in Figure 13(a). The LE output can be programmed to from zero to two row channels and to from zero to two column channels. In addition, for flexibility in routing signals on the channels, the same connections for an LE can be made from the adjacent LAB. If the connections are to be made for the corresponding LE to channels, then due to the limits of the structure shown, the connection access can only be switched and not mixed between the two LABs. The connections to the row channels for the LE are shared with connections from column channels to row channels as shown. Likewise, one of the connections from the LE to the column channels is shared with a connection from a row channel. In each LAB, this overall structure appears eight times, once for each of the eight LEs. The inputs to the LAB from the row channels are shown in Figure 13. There are from 22 to 26 such signals entering the LAB Local interconnect depending on the size of the particular FLEX 10K part.

The programmable connections between an interconnect row and IOEs are shown in Figure 13(b). An IOE can drive two separate row channels as shown. Each IOE is driven by a multiplexer that selects from a subset of the row channels. The structure for connections between an interconnect column and IOEs is similar except that there are only two IOEs at each end of a column.

Design Methodology

The overall structure of the interconnections, LABs, and LEs, IOEs, and EAB is complicated. A designer having to deal with 10's of LABs, hundreds of IOEs, and thousands of interconnection points in an EPLD has a very difficult job. As a consequence, CAD tools are provided that take a design in the form of a schematic or HDL description, automatically partition the design into pieces that fit within the LEs of a LAB, place the pieces into specific LEs and LABs and route the connections between the LABs. The end result of this process is thousands of bits of programming information that can be loaded into the EPLD to implement the desired logic.

REFERENCES

1. Xilinx, Inc., Spartan-II 2.5 V FPGA Family Complete Data Sheet, DS001, September 3, 2003 (http://direct.xilinx.com/bvdocs/publications/ds001.pdf).
2. Xilinx, Inc., Spartan-IIE 1.8 V FPGA Family Complete Data Sheet, DS077, July 9, 2003 (http://direct.xilinx.com/bvdocs/publications/ds077.pdf).
3. Altera Corporation, FLEX 10K Embedded Programmable Logic Device Family Data Sheet, January 2003, version 4.2 (http://www.altera.com/literature/ds/dsf10k.pdf).
4. Altera Corporation, FLEX 10KE Embedded Programmable Logic Device Data Sheet, January 2003, version 2.5 (http://www.altera.com/literature/ds/dsf10ke.pdf).