Integrating Product-Term Logic in APEX 20K Devices

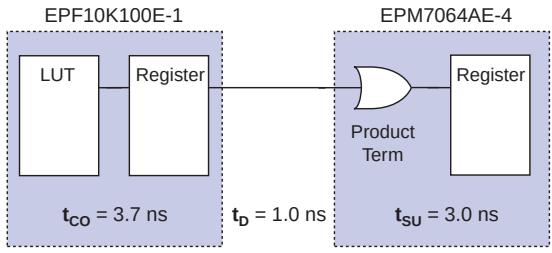
April 1999, ver. 1.0 Application Note 112

Introduction

Altera® APEX™ 20K devices feature the MultiCore™ architecture, which combines product-terms, look-up tables (LUTs), and embedded memory for System-on-a-Programmable-Chip™ integration. The MultiCore architecture improves system performance: the product-term architecture provides higher performance for combinatorial functions (e.g., address decoding and complex state machines), while the LUT architecture contributes superior performance for registered data path functions. This architecture eliminates off-chip delays that result from using separate product-term and LUT devices.

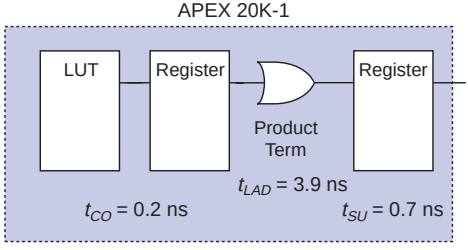
Figure 1 shows system performance using separate product-term and LUT devices, including a typical board delay. In contrast, Figure 2 shows system performance with an integrated product-term and LUT architecture.

Figure 1. System Performance with Two PLDs



Critical Path = 3.7 ns + 1.0 ns + 3.0 ns = 7.7 ns

Figure 2. System Performance with MultiCore Architecture



Critical Path = 0.2 ns + 3.9 ns + 0.7 ns = 4.8 ns

For more information on APEX 20K devices, see the *APEX 20K Programmable Logic Device Family Data Sheet*.

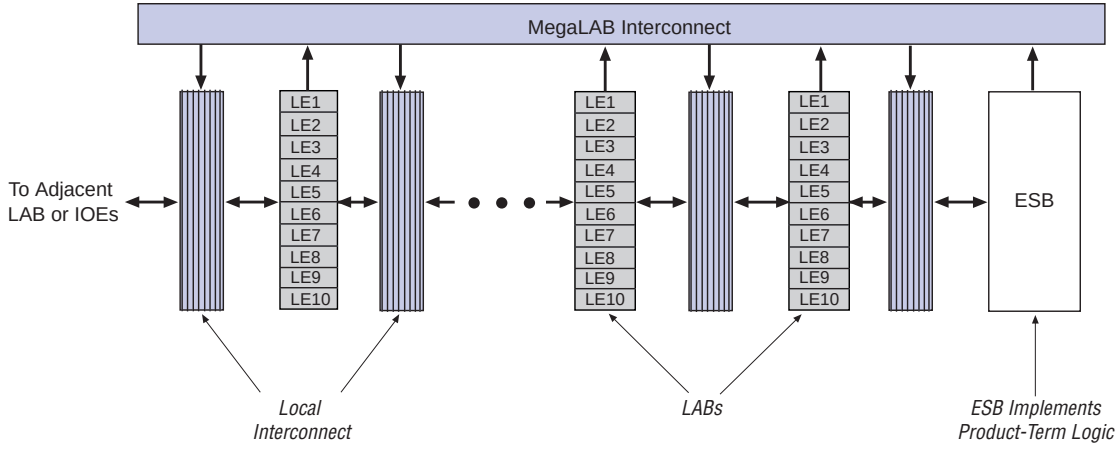
This application note describes the APEX 20K architecture and explains how to implement product-term logic.

MegaLAB Structure

The basic building block of the APEX 20K family is the MegaLAB™ structure, which contains 16 logic array blocks (LABs), each comprised of 10 logic elements (LEs); these LEs are architecturally equivalent to FLEX® 6000 LEs. Each APEX MegaLAB structure also has an embedded system block (ESB) that is configurable as 2,048-bit dual-port RAM, ROM, or content-addressable memory (CAM), or as 16 product-term macrocells.

Each macrocell contains two product terms that can be combined through an OR gate or an XOR gate, and a programmable inverter for wide-input OR functions. The output of each macrocell can be registered with each register containing a clock enable and an asynchronous clear. The register can also emulate an asynchronous preset by using the *NOT-Gate Push-Back* option in the Quartus™ software. Additionally, the ESB macrocell includes parallel expanders that can feed or be fed by an adjacent macrocell. Parallel expanders improve system performance and routability, making the ESB product-term architecture ideal for applications that require wide multiplexing and high fan-in. Figure 3 shows the MegaLAB structure.

Figure 3. MegaLAB Structure

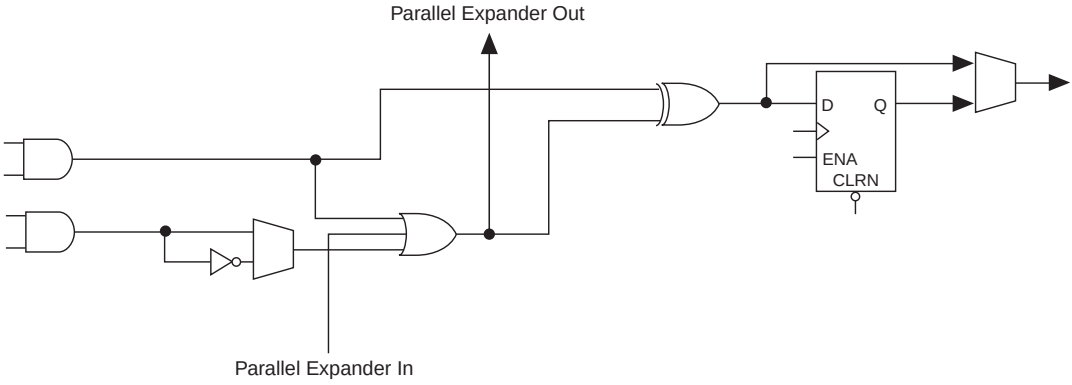


Product-Term Logic in the ESB

The product-term portion of the MultiCore architecture is implemented with the ESB. An ESB can be configured to act as a block of macrocells on an ESB-by-ESB basis. Each ESB is fed by 32 inputs from the adjacent local interconnect; therefore, it can be driven by the MegaLAB interconnect or the adjacent LAB. Also, 9 ESB macrocells feed back into the ESB through the local interconnect for higher performance. Dedicated clock pins, global signals, and additional inputs from the local interconnect drive the ESB control signals.

In product-term mode, each ESB contains 16 macrocells. Each macrocell consists of two product terms and a programmable register. The programmable register can implement D, T, JK, or SR flipflops. Parallel expanders make the ESB product-term configuration ideal for applications requiring wide multiplexing and high fan-in. Figure 4 shows the APEX ESB macrocell.

Figure 4. APEX ESB Macrocell



Parallel expanders make the ESB product-term configuration ideal for applications requiring wide multiplexing and high fan-in because they can be used to drive up to 32 product terms to a single macrocell. Figures 5 and 6 show the APEX ESB parallel expanders and feedback, respectively.

Figure 5. APEX ESB Parallel Expanders

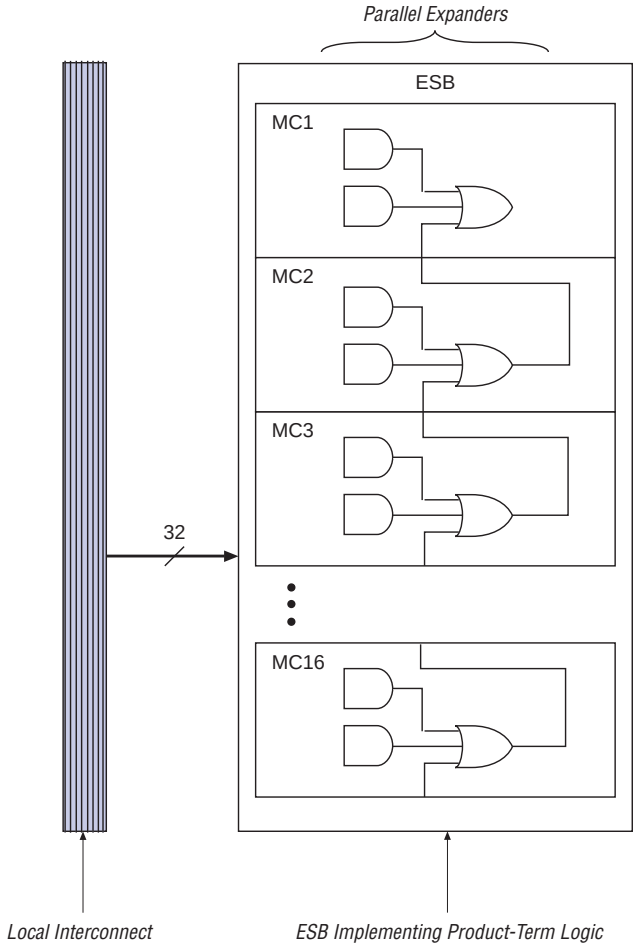
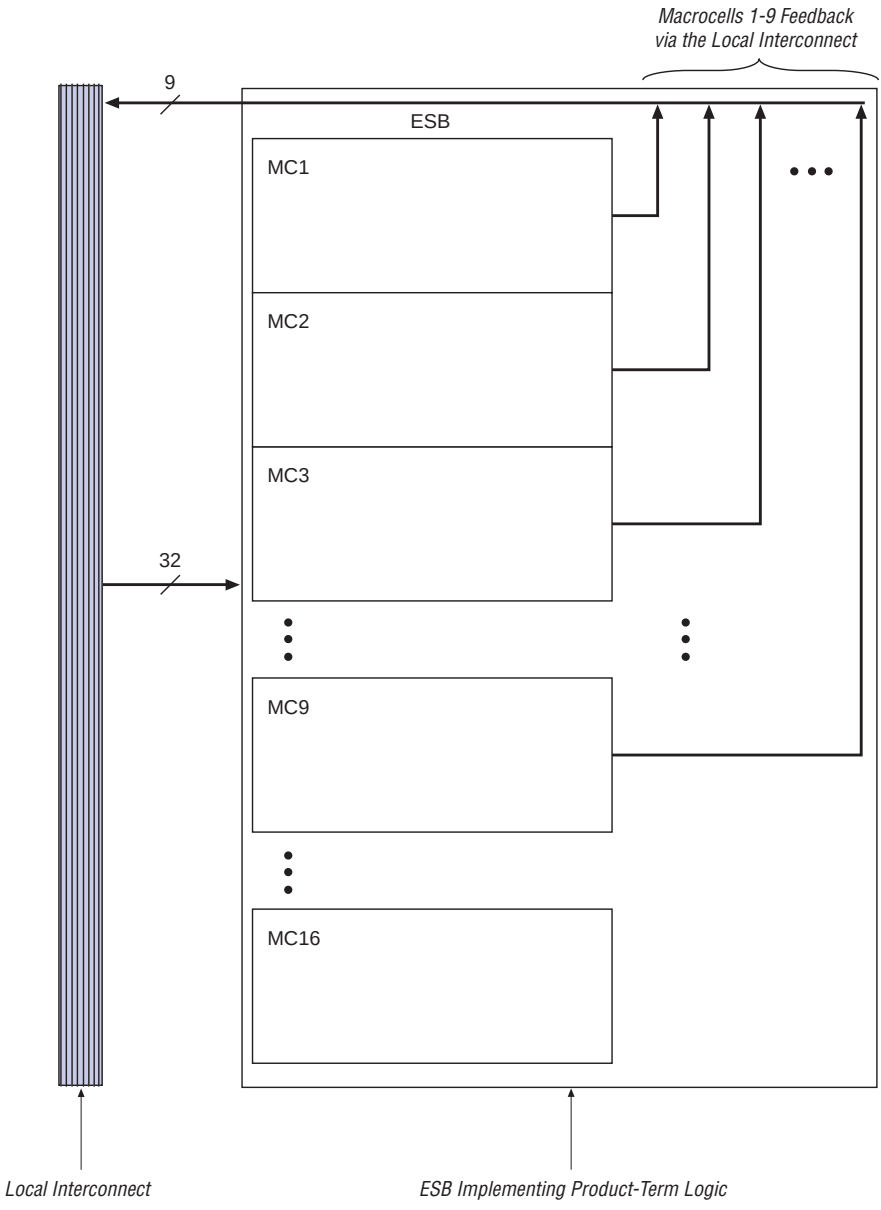


Figure 6. APEX ESB Feedback



Improving Performance by Using Product Terms

The MultiCore architecture improves system performance: the product-term architecture provides higher performance for combinatorial functions such as address decoding and state machines, while the LUT architecture contributes superior performance for registered data path functions. Subdesigns such as wide-input functions and state machines are implemented more efficiently in product terms; therefore, combining two architectures in one device results in better performance and device utilization.

Table 1 compares performance and device utilization for common applications implemented in product-term and LUT-based architectures. Thirty-two product terms require the same silicon area as 50 LEs. The wide-input AND gate and state machine are faster and implemented more efficiently in the product-term architecture, while the multiplier and multiplexer are better implemented in the LUT.

The product-term delay of the APEX ESB is approximately 3.9 ns; the parallel expander delays of the ESB are approximately 0.7 ns.

Table 1. APEX Performance & Utilization for Common Applications

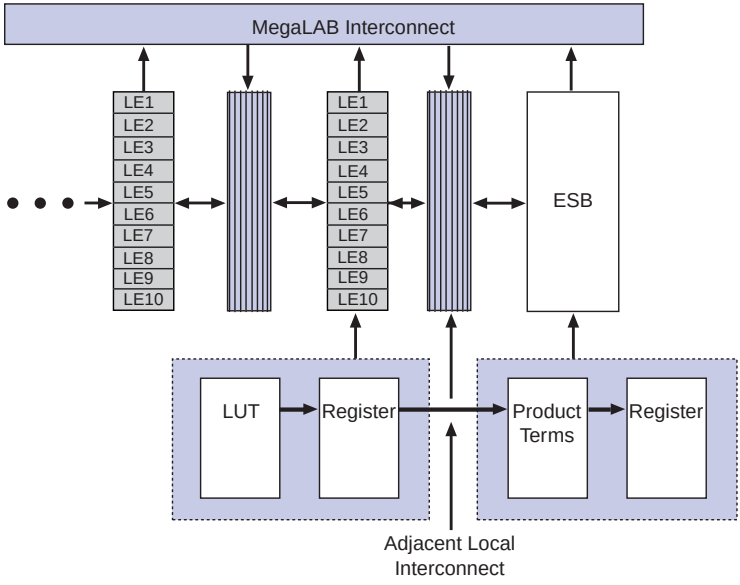
Function	Product Terms		LUTs		Ideal Solution		APEX Performance (MHz)
	Performance (MHz)	Utilization	Performance (MHz)	Utilization	Product Term	LUT	
32-bit AND gate with registered inputs and outputs	192	1 product term (die size equivalent to 1.6 LEs) *(1)*	172	8 LEs *(1)*	✓		192
8-state, 6-input/ 11-output 148-transition state machine	76	204 product terms (die size equivalent to 319 LEs)	66	366 LEs	✓		76
16-to-1 registered I/O multiplexer	149	20 product terms (die size equivalent to 30 LEs) *(1)*	185	10 LEs *(1)*		✓	185
8 × 8 registered I/O multiplier	52	702 product terms (die size equivalent to 1,097 LEs) *(1)*	188	135 LEs *(1)*		✓	188

Note:
(1) Input registers are not included in utilization numbers.

Register Placement for Optimal Performance

For optimal performance in product-term mode, registers that are used to drive product terms can be placed in the LAB that is adjacent to the ESB. This LAB directly drives the local interconnect that drives the ESB, eliminating any routing delays through the MegaLAB interconnect. Timing-driven compilation in the Quartus software will also use this placement to meet user-specified timing requirements. See Figure 7.

Figure 7. Optimal Placement of Logic



Using Turbo Mode

The APEX ESB has a "turbo" mode that improves performance of logic implemented in the ESB. The Quartus software provides designers with the option to turn on this feature for improved performance or turn off this feature for reduced power consumption. Using the Quartus software, you can implement this APEX ESB feature on an ESB-by-ESB basis.

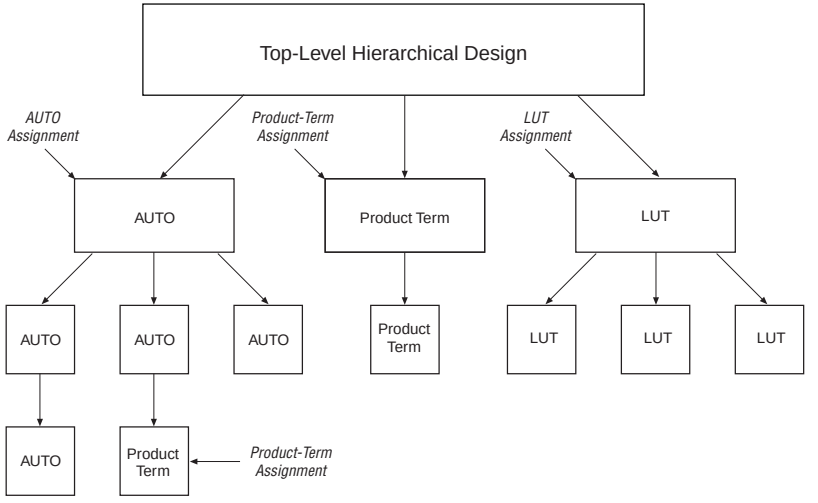
Using Quartus Software to Implement Product-Term Logic

Altera's Quartus software, which supports APEX 20K devices, allows you to control APEX ESB implementation in product-term mode on either specific blocks of logic or on an ESB-by-ESB basis. You can implement product-term, LUT, and ROM configurations of the ESB using the **Assignment Organizer** dialog box (see Figure 11 on page 11). RAM, ROM, and CAM logic designs are implemented through megafunctions.

The Quartus software supports multiple methods for implementing logic in product-term mode. For example, logic can be targeted for product-term mode on a hierarchical level. If you target a hierarchy level for product-term mode, that hierarchy level and all of its lower levels will be implemented using the product-term mode configuration. By using the Quartus software, you can also target a hierarchy level as *AUTO*. Based on an area utilization algorithm, the Quartus software automatically decides if the logic is better implemented in product terms or LUTs. A lower hierarchy level may be targeted for a different logic implementation than its parent hierarchy level (only if the targeted higher level is set for *LUT* or *AUTO*).

Figure 8 demonstrates hierarchical implementation capability of logic in product-term mode using the Quartus software.

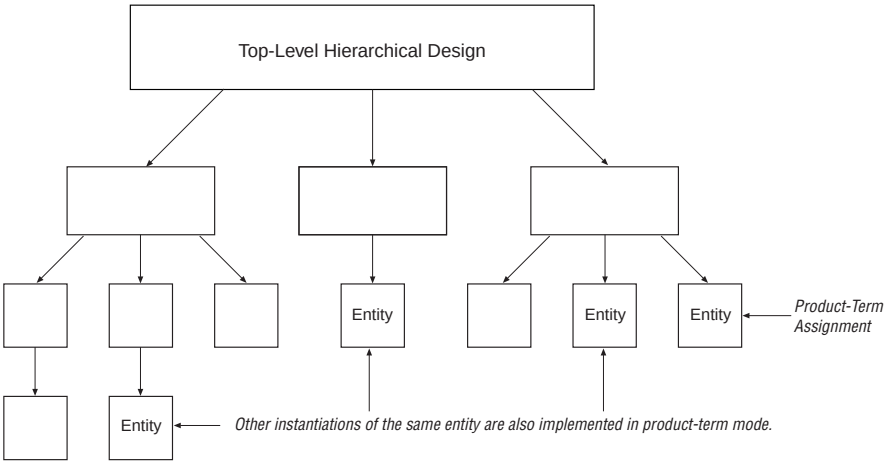
Figure 8. Hierarchical Implementation



Logic can be targeted globally or on an entity-by-entity basis. You can target an entity for product-term implementation. If other instances of the entity occur within the design, the Quartus software also provides the option to have the other instances implemented in product-term mode.

Figure 9 demonstrates the entity-based implementation capability of logic in product-term mode using the Quartus software.

Figure 9. Entity-Based Implementation

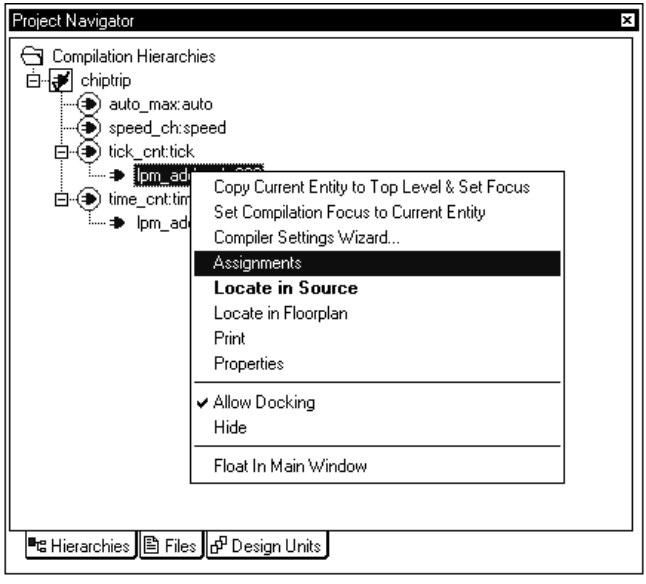


Using the Quartus software, perform the following steps to designate a hierarchy level for product-term mode.

1. Open the **Project Navigator** in the Quartus software.

2. Right-click on the desired hierarchy level and choose **Assignments**. See Figure 10.

Figure 10. Targeting Logic for Product-Term Mode

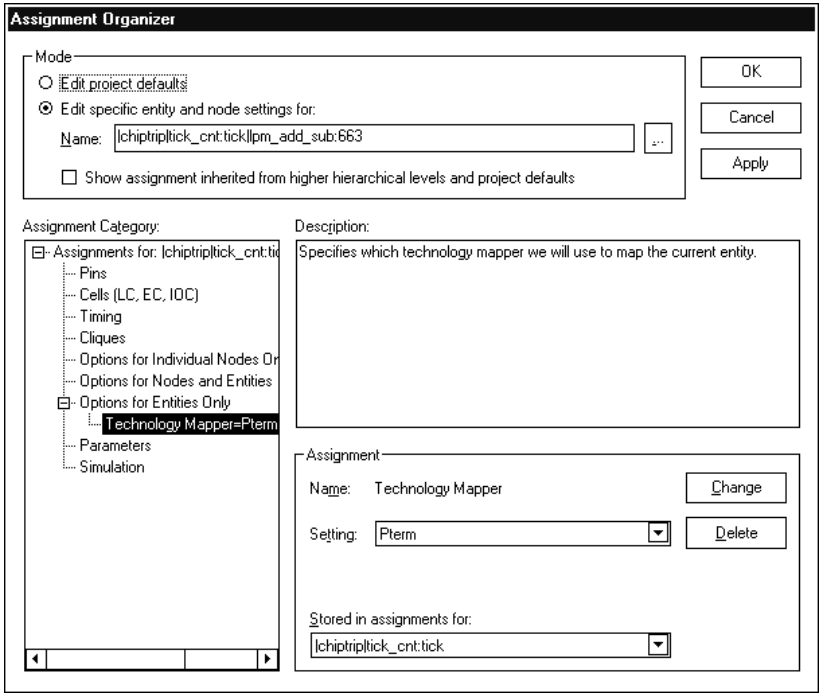


3. Open the list of files under **Options for Entities Only** and select the **Technology Mapper** assignment category in the **Assignment Organizer** dialog box. This selection maps logic to a specified mode (see Figure 11).

4. Choose *Pterm* in the *Setting* drop-down list box and click **Add/Change**.

5. Click **Apply** to continue to adjust assignments or click **OK** if you are finished.

Figure 11. Assignment Organizer Dialog Box



Third-Party Software Support

The Quartus software supports product-term logic designs created in Exemplar Logic, Synopsys, Synplicity, and Viewlogic synthesis tools. The Quartus software features a true WYSIWYG (What-You-See-Is-What-You-Get) option that allows third-party tools to synthesize product-term logic in the APEX ESB architecture. Logic blocks pass through the Quartus Compiler without further synthesis, ensuring optimal design implementation. The WYSIWYG product-term structure can be passed from third-party tools to the Quartus software through Verilog HDL, EDIF, and VHDL files. In addition to ESB product-term mode, you can access LUT logic, ESB RAM, or ROM storage through third-party tools.

Conclusion

Altera's APEX 20K devices feature the MultiCore architecture, which combines product-term, LUT, and embedded memory architectures. By using the MultiCore architecture, you can integrate your design into one device, improving system performance.

For more information on APEX 20K devices and the Quartus software, see the *APEX 20K Programmable Logic Device Data Sheet* and the *Quartus Programmable Logic Development System & Software Data Sheet*.

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com
Applications Hotline:
(800) 800-EPLD
Customer Marketing:
(408) 544-7104
Literature Services:
(888) 3-ALTERA
lit_req@altera.com