Expanding Applications

For Low Cost FPGAs

A Lattice Semiconductor White Paper

April 2007

Revised August 2007

Lattice Semiconductor
5555 Northeast Moore Ct.
Hillsboro, Oregon 97124 USA
Telephone: (503) 268-8000
www.latticesemi.com

The Low Cost FPGA Marketplace

Over the last several years, the popularity of low cost Field Programmable Gate Arrays (FPGAs) has increased dramatically, to the point where low cost FPGAs now represent approximately 25% of the overall market. However, the features these products provide (or do not provide) have limited the applications that can utilize low cost FPGAs. The challenge for FPGA providers is to increase the feature set of these devices while continuing to deliver the attractive price points that have made these devices so popular.

This white paper examines the need for, and approaches to, providing enhanced low cost FPGA capability in the areas of SERDES, DSP, high-speed source synchronous I/O, memory capacity and device configuration. The white paper concludes with a summary of the second generation EConomy Plus FPGAs from Lattice Semiconductor, and the approaches taken to address these five capabilities.

SERDES Functionality: A New Need In Low Cost FPGAs?

Once the domain of high-end communications equipment, SERDES technologies are becoming increasingly important for more cost-sensitive devices. The adoption of SERDES-based PCI Express in the PC is well underway, and the technology now is migrating to a variety of other equipment as a replacement for the original PCI interface, which is now over a decade old. In cost sensitive edge and access equipment, including that targeted for rapidly emerging triple play (phone, video and data) applications, Gigabit Ethernet (GbE) and Serial Gigabit Media Independent Interface (SGMII) are becoming increasing common as interfaces between PHY devices and the remainder of the system. In the wireless base station market, cost reduction is becoming increasingly important and likely will be enabled in part by the standardization that the serial Common Public Radio Interface (CPRI) and Open Base Station Standards Initiative (OBSAI) provide.

Despite the tremendous pent up demand for SERDES in low cost systems, to date only higher performance, higher cost FPGAs have provided SERDES capability. Generally,

the SERDES in high performance FPGAs have focused on driving performance in a number of directions, such as link speed, number of channels, ultra low jitter, support for long Consecutive Identical Digits (CID, as found in SONET) and the capability to drive long backplanes. While appropriate for high end FPGAs, this focus on performance has resulted in cost and power characteristics that are inappropriate for low power applications.

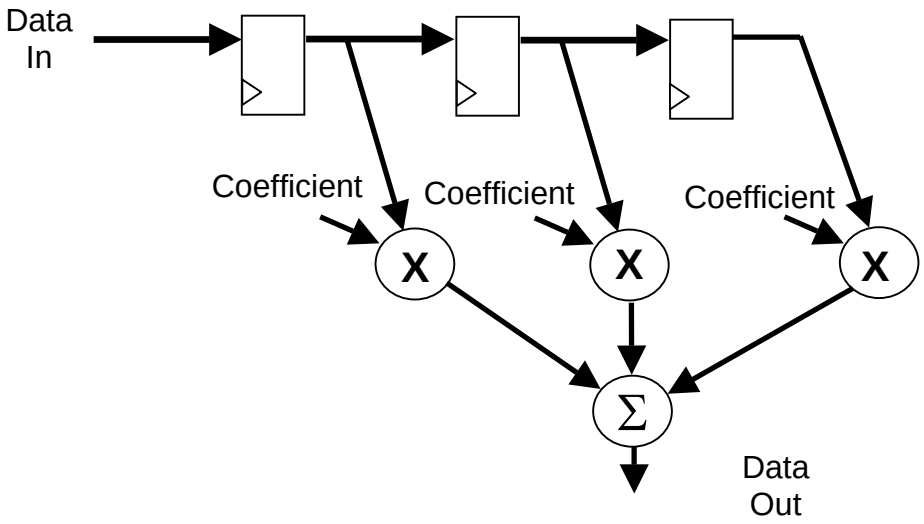
DSP Functionality In Low Cost FPGAs

The applications of Digital Signal Processing (DSP) continue to expand, driven by trends such as the increased use of video and still images and the demand for increasingly reconfigurable systems such as Software Defined Radio (SDR). Many of these applications combine the need for significant DSP processing with cost sensitivity, creating demand for high performance, low cost DSP solutions.

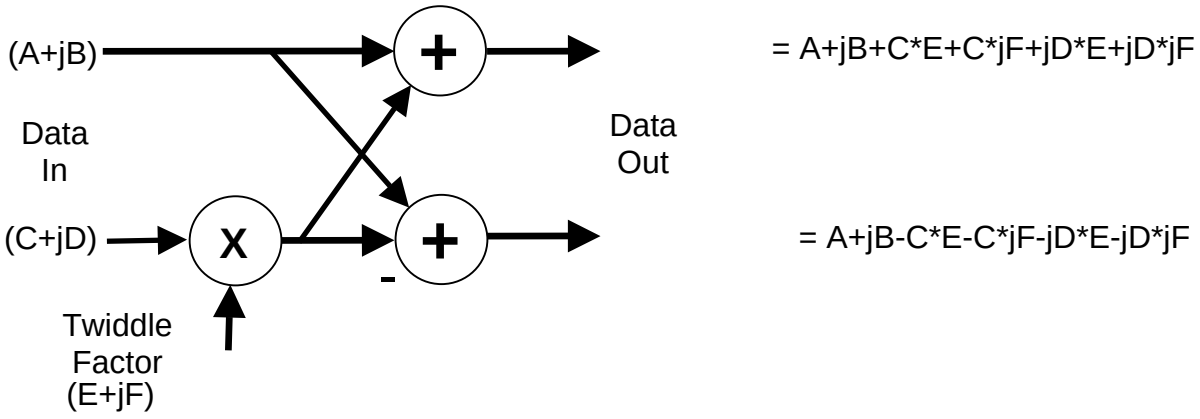
Designers often implement structured DSP functions that require significant computation resources within FPGAs. This approach allows the selection of a lower performance general purpose DSP processor, and results in lower overall system costs. Examples of structured functions that are commonly implemented within FPGAs include Finite Impulse Response (FIR) filters, Fast Fourier Transforms (FFTs) and mixers. Figure 1 shows typical implementations of FIR and FFT functions. These implementations share the characteristic of requiring multiplication followed by addition, subtraction or accumulation.

To date, three approaches have been taken to implementing DSP functions within low cost FPGAs. The first approach adopted was to implement the DSP functions using the general purpose Look-Up Tables (LUTs) available within the device. While flexible, this method provides relatively low performance and consumes a significant quantity of the general purpose FPGA logic. As a result, all current generation low cost FPGAs take one of two enhanced approaches. The first approach is to implement multipliers as hard logic within the FPGA fabric. This reduces the general-purpose FPGA resources that are required to implement DSP functions. However, as previously noted, most typical DSP functions implemented within FPGAs require additional subtraction, addition

or accumulation functions after the multiplier. These functions in themselves can consume significant FPGA resources and with typical final data widths running upwards of 36-bits, they are often the performance bottleneck of the design. To address this challenge, the latest FPGAs utilize efficient hard logic to provide programmable addition, subtraction and accumulation after the multiplier. By providing DSP functionality in this manner, usage of general-purpose resources is minimized and high performance operation is easily achieved. Table 1 summarizes the three different approaches.



Typical Finite Impulse Response FIR Filter



Radix-2 Butterfly Element Commonly Used In Fast Fourier Transformation

Figure 1 – Typical Implementations of FFT and FIR Functions

Approach	Flexibility	Performance	General Purpose Resource Usage
General Purpose LUTs	High	Moderate	High
Dedicated Multipliers	High	Moderate	Moderate
Dedicated Multipliers Plus Adders, Subtractors and Accumulators	High	High	Low

Table 1 –Approaches to Providing DSP Functions in FPGAs

High-Speed Parallel Interfaces For Low Cost FPGAs

Designers of low cost systems find it increasingly important to implement high performance parallel interfaces (often referred to as source synchronous interfaces) within low cost FPGAs. Several system requirements typically drive this need. In some cases it is necessary to interface with DDR SDRAM memory, which is rapidly becoming the lowest cost high capacity memory available to designers. In other instances it is necessary to support the high performance SPI4.2 communications standard that is common to many forms of communication equipment. In yet other instances it is driven by the need to interface to high performance Analog to Digital Converters (ADCs) of Digital to Analog Converters (DACs).

Implementation of high performance source synchronous interfaces typically poses four challenges to FPGA designers:

- The conversion of signals from Signal Data Rate (SDR) to Double Data Rate (DDR), a process that results in significantly less margin than is available in the processing of the SDR data typically used within FPGAs.
- The adaptation of speed between high performance I/Os and the FPGA fabric. Again, this processing is done with significantly less margin than is typically available in FPGA designs.
- The alignment of the clock (sometimes referred to as DQS or Strobe) and data in order to correctly transmit or receive information. This requires the ability to insert precision delays in the clock path relative to the datapath.
- The transfer of data from the external clock domain to the system clock domain used within the FPGA. If this task is not approached correctly, a design risks

problems in the field or during volume manufacture, even if the prototypes operated correctly in the lab.

Generally, hard logic to implement these four functions can be added to the FPGA design with minimum impact on die area and therefore device cost. Today all low cost FPGAs provide some of these capabilities as hard logic in the I/O circuitry, with the most advanced providing all four functions as hard logic. FPGAs that provide all four functions deliver higher performance, lower general-purpose logic usage and faster design cycle time.

High Memory Capacity

Many functions require relatively large amounts of on-chip memory for buffering data. For example, serial applications require added capacity to buffer large frames and packets, as well as for system-level flow control capability. For the DSP and SERDES functions described previously in this white paper, this requirement is particularly prevalent. To date low cost FPGAs have provided only relatively small amounts of on-chip memory. The amount of on-chip memory available in low cost FPGAs has typically been approximately 20% of that available in higher cost, high performance devices.

Low Cost FPGA Configuration

All low cost SRAM-based FPGAs require configuration at system power-up. Typically, designers configure FPGAs from either dedicated boot memory or more general system memory via the microprocessor. For those designers who choose to configure their FPGAs from dedicated boot memory most, but not all, current generation low cost FPGAs support interfacing directly with low cost Serial Peripheral Interface (SPI) Flash memory. Use of this industry standard memory significantly reduces the overall cost of FPGA logic implementation compared to the use of the proprietary boot memory still required by some FPGAs.

In addition to the availability of flexible low cost configuration options, designers of low cost FPGAs seek to solve three challenges associated with configuration: increasing design security, reliably updating the stored FPGA configuration while equipment is

deployed in the field and updating the FPGA configuration while equipment continues to operate.

Improving FPGA Design Security

As FPGAs continue to replace ASICs and ASSPs, playing a more significant role in systems, designers are becoming increasingly concerned about design security. SRAM FPGAs, which by their nature must receive a configuration bitstream at every system power-up, are at particular risk. Concerns include reverse engineering, overbuilding by sub-contractors and cloning of systems. System designers are eager to secure their designs against these threats.

Reliably Updating FPGA Configurations In The Field

It is increasingly important to provide the capability to update the stored FPGA configuration while equipment is deployed in the field. This capability provides equipment suppliers with a competitive advantage by allowing them to respond rapidly to changes in standards, add new services and, if necessary, fix bugs. However, while the stored FPGA configuration is being updated there is a risk that a power or communication failure could result in a corrupted configuration and a non-operational system. If this occurs then an expensive visit from a field technician typically is required to solve the problem.

To guard against the possibility of non-operational systems, designers who configure from a system memory via a microprocessor typically store two versions of the configuration and update only one at a time. The microprocessor arbitrates between the copies and ensures that the FPGA always can be configured. However, designers who configure directly from Flash memory have not been able to adopt this approach, as historically there has been no device to arbitrate between the different configuration copies.

Updating FPGA Configurations While Equipment Operates

Another challenge facing many designers is the need to update FPGA configurations while equipment continues to operate. The requirements that many types of equipment

have for high availability such as ‘Five Nines’ (99.999%) often make it imperative to update logic in the field without ceasing equipment operation. Unfortunately, FPGAs have traditionally tri-stated their I/Os while applying a new configuration, which makes it difficult if not impossible to achieve the dual goals of field updates and high uptime. FPGAs that can precisely control the state of I/Os during the configuration cycle make it possible to achieve these two goals.

LatticeECP2 and LatticeECP2M Families

The LatticeECP2™ and LatticeECP2M™ devices (also referred to as LatticeECP2/M) are unique low cost FPGAs that address the need for enhanced DSP, Source Synchronous I/O and improved configuration support within a low cost architecture. The LatticeECP2M adds SERDES and additional memory capacity.

The LatticeECP2 architecture consists of an array of up to 68K LUTs surrounded by flexible I/O buffers that incorporate pre-engineered I/O support to enable the easy implementation of high performance source synchronous interfaces such as the DDR2 at 400Mbps and SPI4.2. Strips of sysDSP blocks pass through the logic array to provide up to 88 18x18 multipliers followed by addition, subtraction and accumulation functions. These blocks run up to 375MHz, providing an aggregate DSP capability of 33 Giga Multiply Accumulates per Second (GMACs). A second strip through the device provides sysCLOCK™ Delay Locked Loop (DLL) and Phased Locked Loop (PLL) capability, up to 1Mbit of sysMEM™ embedded memory and enhanced configuration capabilities including support for encrypted bitstreams, dual boot and TransFR™ I/O.

The LatticeECP2M architecture is similar to the LatticeECP2 but adds up to 16 channels of SERDES optimized for PCI Express, Gigabit Ethernet, SGMII, CPRI and OBSAI applications. The number of LUTs is increased 30% to 95K while DSP support increases to 168 18x18 multipliers, yielding 63GMACs capability, and memory support increases five times to 5.3Mbits. Figure 2 shows the block diagrams for the LatticeECP2 and LatticeECP2M FPGAs.

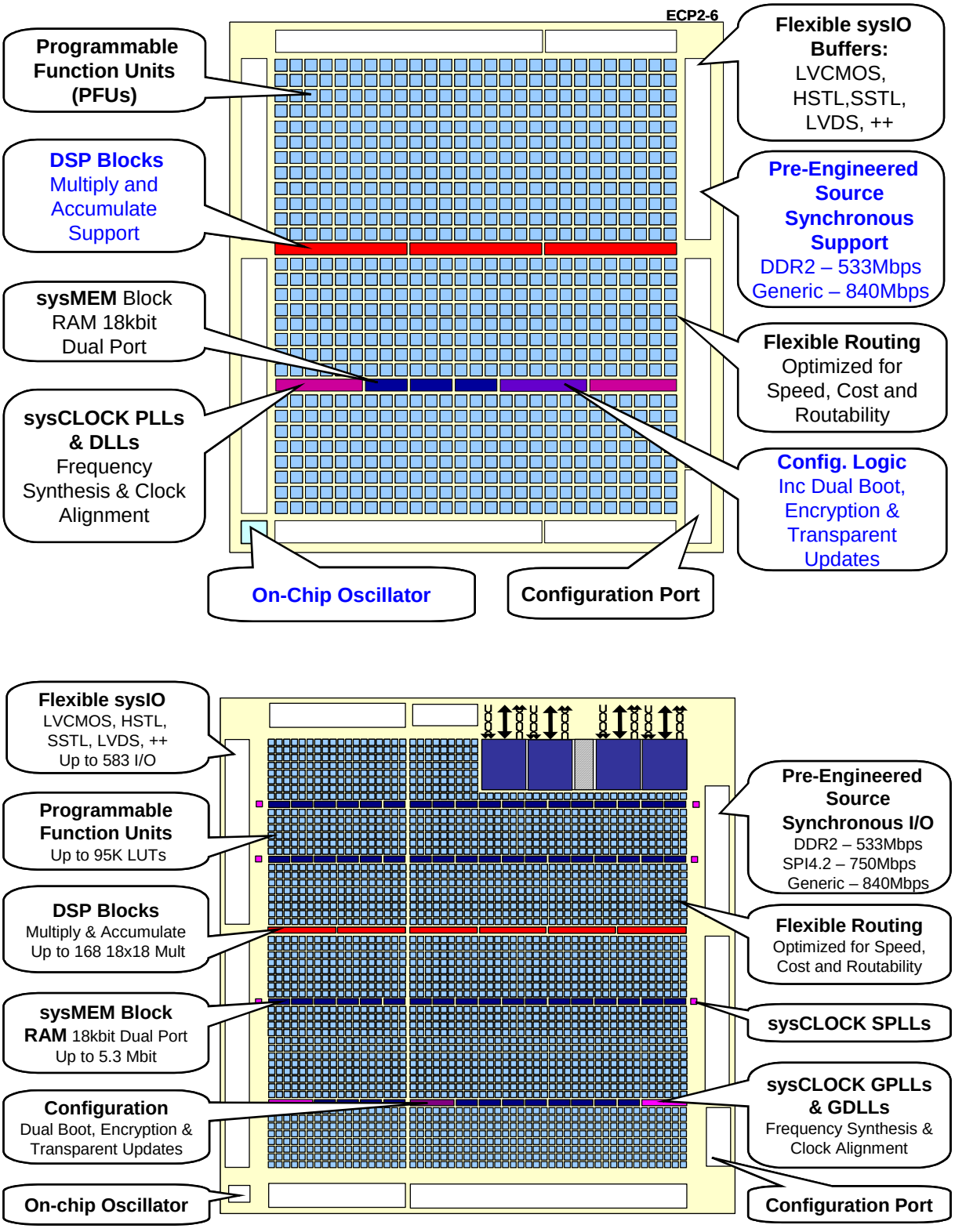


Figure 2 – LatticeECP2 and ECP2M Architectures

The LatticeECP2 family consists of six devices with densities from 6K to 68K LUTs. The LatticeECP2M family consists of five devices with densities from 19K to 95K LUTs. Table 2 provides more device details, including the memory supported, available PLLs/DLLs and DSP capability. The devices support a variety of low cost TQFP, PQFP and fpBGA packages. Density migration is supported within each family, allowing designers to begin a design in one device and later move to a lower cost device if not all the capacity is required, or to a higher capacity device if more functionality is needed.

Device	ECP2						ECP2M				
	6	12	20	35	50	70	20	35	50	70	100
LUTs (K)	6	12	21	32	48	68	19	34	48	67	95
18x18 Multipliers	12	24	28	32	72	88	24	32	88	96	168
Distributed RAM (Kbits)	12	24	42	65	96	136	41	71	101	145	202
EBR SRAM Blocks	3	12	15	18	21	60	66	114	225	246	288
EBR Block SRAM (Kbits)	55	221	276	332	387	1106	1217	2101	4147	4534	5308
PLLs/DLLs	2/2	2/2	2/2	2/2	4/2	6/2	8/2	8/2	8/2	8/2	8/2
DDR1/DDR2 Memory (Mbps)	400/533	400/533	400/533	400/533	400/533	400/533	400/533	400/533	400/533	400/533	400/533
Package	I/O						I/O/SERDES				
100-pin TQFP (14x14mm)											
144-pin TQFP (20x20mm)	90	93									
208-pin PQFP (28x28mm)		131	131								
256-ball fpBGA (17x17mm)	190	193	193				4/140	4/140			
484-ball fpBGA (23x23mm)		297	331	331	339		4/304	4/303	4/270		
672-ball fpBGA (27x27mm)			402	450	500	500		4/410	8/372		
900-ball fpBGA (31x31mm)						583			8/410	16/416	16/416
1152-ball fpBGA (35x35mm)										16/436	16/520

Table 2 – LatticeECP2 and LatticeECP2M Family Members

SERDES Block

The SERDES within the LatticeECP2M devices has been optimized for chip-to-chip, chip-to-daughter card and small form factor (<20 inches of FR-4 material) applications using 8b/10b encoding in the 270Mbps to 3.125Gbps range. The block supports common SERDES applications including PCI Express, Gigabit Ethernet, SGMII, CPRI and OBSAI. As a result of these focused applications, the ECP2M devices provide excellent support for the most common applications while avoiding the high cost and power penalties of more performance intensive SERDES implementations.

sysDSP Block

The sysDSP block in the LatticeECP2/M devices supports four functional elements in three data path widths: 9, 18 and 36. The resources in each sysDSP block can be configured to support the following four elements: MULT, MAC, MULTADD and MULTADDSUM.

The number of elements available in each block depends upon the width selected from the three available options: x9, x18, and x36. A number of these elements can be concatenated for highly parallel implementations of DSP. Figure 3 shows an overview of the sysDSP block.

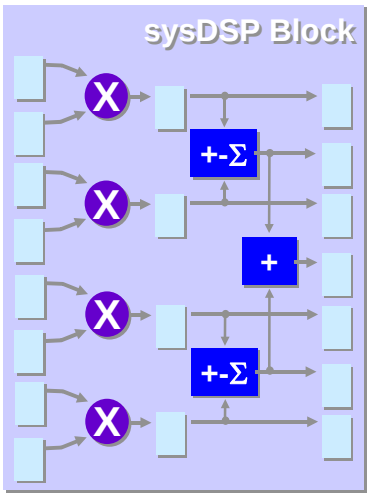


Figure 3 – sysDSP Block

Pre-Engineered Source Synchronous Interfaces

The I/O cells in the LatticeECP2/M devices contain a number of pre-engineered elements to allow the easy implementation of source synchronous interfaces such as those found on DDR1/2 memories, SPI4.2 systems and high speed ADC/DACs.

- Precision DQS/Strobe Delay Control
- Dedicated DDR Registers (For Mux and Demuxing)
- Automatic DQS to System Clock Transfer
- 2:1 Gearbox Logic to Match I/O Speed with FPGA Fabric
- Low Skew Edge Clocks

The elements highlighted above can be combined easily in the ispLEVER design tool to implement a variety of interfaces including 400/533Mbps DDR1/2 memory interfaces and 840Mbps generic source synchronous interfaces. Figure 4 shows a diagram of the LatticeECP2/M I/O cell.

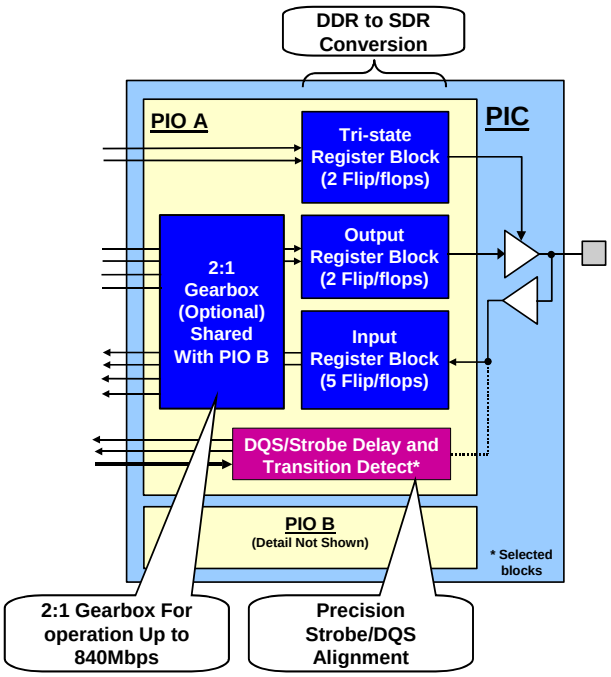


Figure 4 – LatticeECP2/M Pre-Engineered Source Synchronous Support

Enhanced Configuration Options

Each LatticeECP2/M device can be configured quickly using one of three different methods: from a low cost industry standard SPI Flash memory, from the device's serial or parallel microprocessor port or from the device's JTAG port. In addition to providing flexibility in terms of programming sources, the LatticeECP2/M devices also provide a number of enhanced configuration options as detailed below.

Dual Boot Operation

The LatticeECP2/M devices allow two images to be stored in sectors 0 and 1 of an SPI boot memory. The FPGA will attempt first to configure itself from sector 1. If this configuration fails, configuration then will be attempted from sector 0. (If the bitstream image is larger than one sector, then a jump command can be used to span it across multiple sectors.) This dual image operation is ideal for situations in which the configuration is to be changed in the field as it enables a valid "golden configuration" to be held in memory at all times. Figure 5 illustrates the dual boot concept.

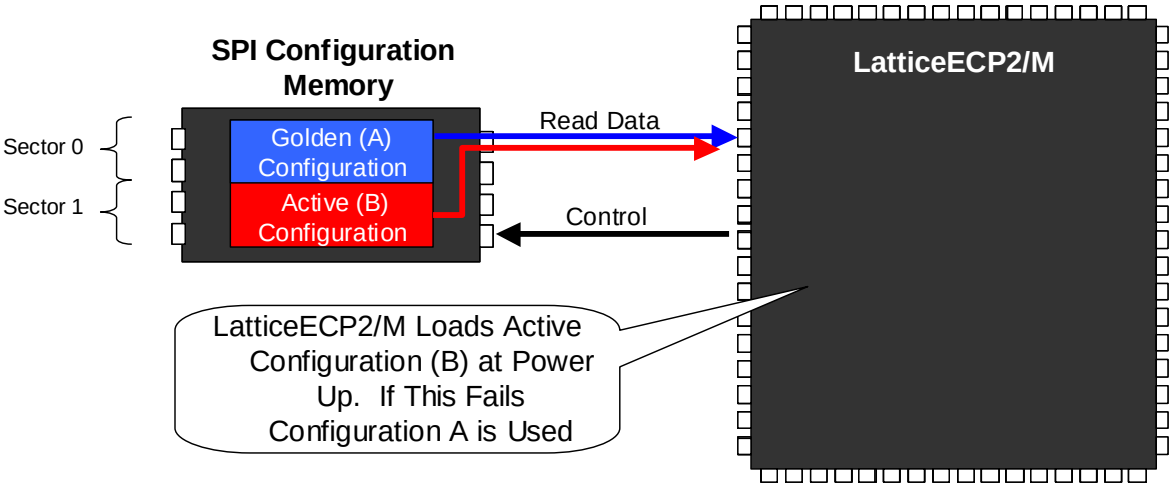


Figure 5 -- Using Dual Boot Configuration With Dual Boot

Bitstream Encryption

The LatticeECP2/M devices contain non-volatile memory elements that can be used for the storage of a 128-bit customer-specific decryption key. Bitstream files can be encrypted with this key prior to programming them into the configuration memory. As the encrypted bitstream enters the FPGA, it is decrypted using the key stored on the device. This capability provides a highly effective method to combat design piracy and overbuilding, as illustrated in figure 6.

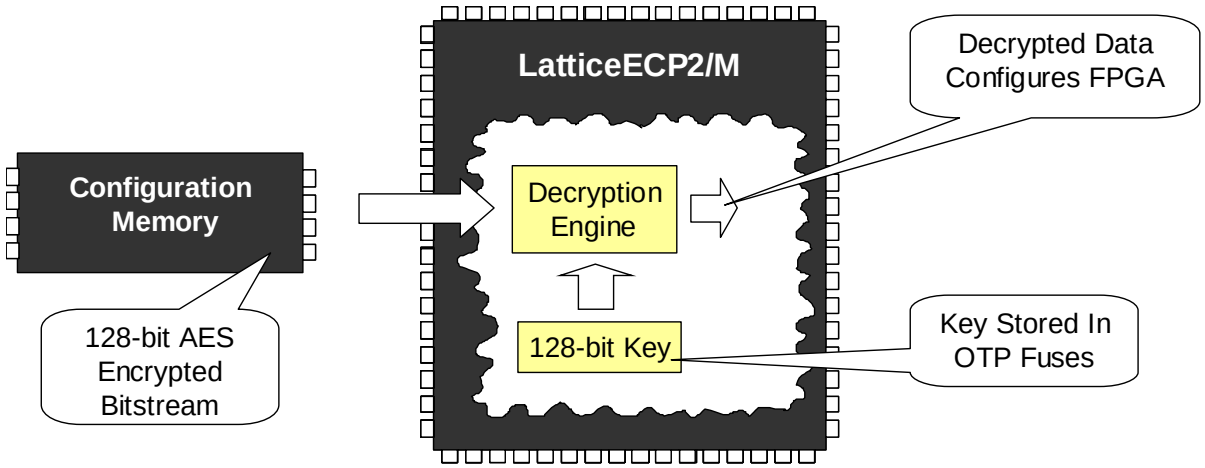


Figure 6 -- Bitstream Encryption Deters Design Piracy

TransFR I/O

LatticeECP2/M devices feature TransFR I/O that allows I/O states to be frozen during device configuration. This allows the devices to be field updated with a minimum of system disruption and downtime, allowing designers to meet the dual requirements of high system uptime, such as '5 nines' (99.999%) availability, and field updating of logic. Figure 7 illustrates the four steps to implement these updates.

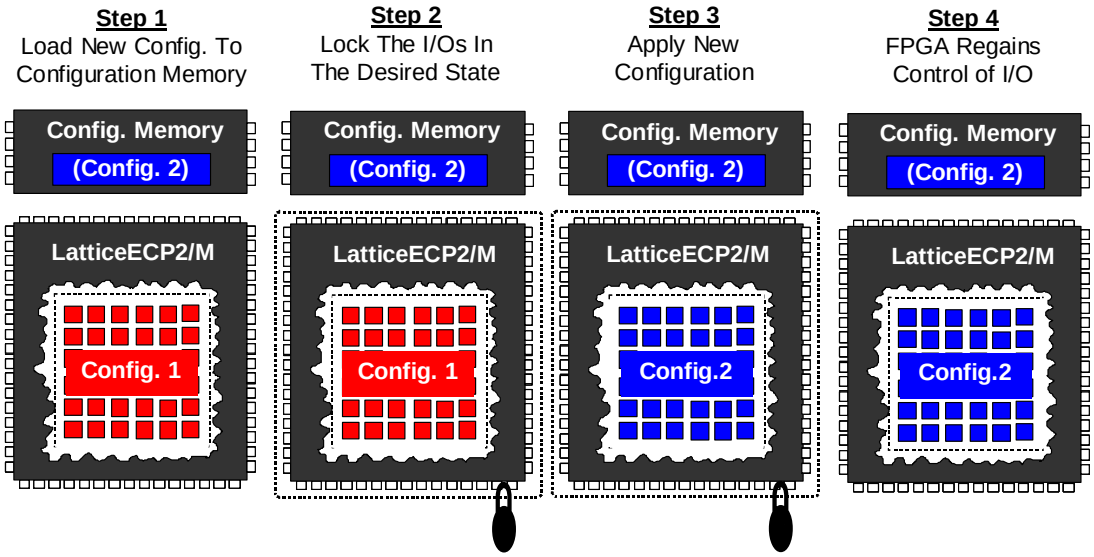


Figure 7 -- TransFR I/O Allows FPGA Update While Equipment Operates

Summary

Low cost FPGAs must have a more robust feature set while maintaining attractive pricing if they are to become a viable option for an expanded number of users. Promising areas in which this can be achieved include SERDES, DSP functionality, high-speed source synchronous I/O, block memory capacity and device configuration. In all five areas, the implementation of a small amount of hard logic can dramatically increase device functionality with minimum impact on device cost. The second-generation EConomy Plus FPGA families, the LatticeECP2 and LatticeECP2M from Lattice Semiconductor, are redefining the low cost FPGA industry by providing high-end features and capabilities. As a result, the number of designers who can use this already popular class of FPGAs is bound to increase.

###