Field Programmable Gate Array Implementation Technology

Syed.Awais Hyder, D.Sri Kanth, C.Chandrasekhar, E.Sammaiah

Abstract— A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer after manufacturing hence "field-programmable". The FPGA configuration is generally specified using hardware (HDL), similar to that used for an application-specific integrated circuit (ASIC) (circuit diagrams were previously used to specify the configuration, as they were for ASICs, but this is increasingly rare). FPGAs can be used to implement any logical function that an ASIC could perform. The ability to update the functionality after shipping, partial re-configuration of a portion of the design and the low non-recurring engineering costs relative to an ASIC design (notwithstanding the generally higher unit cost), offer advantages for many applications. FPGAs contain programmable logic components called "logic blocks", and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together"—somewhat like many (changeable) logic gates that can be inter-wired in (many) different configurations. Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory. In addition to digital functions, some FPGAs have analog features.

Index Terms— Field Programmable Gate Array Implementation Technology.

I. INTRODUCTION

Field-programmable gate array (FPGA) technology continues to gain momentum, and the worldwide FPGA market is expected to grow to $3.5 billion USD by 20131. Since their invention by Xilinx in 1984, FPGAs have gone from being simple glue logic chips to actually replacing custom application-specific integrated circuits (ASICs) and processors for signal processing and control applications. Why has this technology been so successful? This article provides an introduction to FPGAs and highlights some of the benefits that make FPGAs unique. At the highest level, FPGAs are reprogrammable silicon chips. Using prebuilt logic blocks and programmable routing resources, you can configure these chips to implement custom hardware functionality without ever having to pick up a breadboard or soldering iron. You develop digital computing tasks in software and compile them down to a configuration file or bit stream that contains information on how the components should be wired together. In addition, FPGAs are completely reconfigurable and instantly take on a brand new "personality" when you recompile a different configuration of circuitry. In the past, FPGA technology could be used only by engineers with a deep understanding of digital hardware design. The rise of high-level design tools, however, is changing the rules of FPGA programming, with new technologies that convert graphical block diagrams or even C code into digital hardware circuitry. FPGA chip adoption across all industries is driven by the fact that FPGAs combine the best parts of ASICs and processor-based systems. FPGAs provide hardware-timed speed and reliability, but they do not require high volumes to justify the large upfront expense of custom ASIC design. Reprogrammable silicon also has the same flexibility of software running on a processor-based system, but it is not limited by number of processing cores available. Unlike processors, FPGAs are truly parallel in nature, so different processing operations do not have to compete for the same resources. Each independent processing task is assigned to a dedicated section of the chip, and can function autonomously without any influence from other logic blocks. As a result, the performance of one part of the application is not affected when you add more processing. The most common analog feature is programmable slew rate and drive strength on each output pin, allowing the engineer to set slow rates on lightly loaded pins that would otherwise ring unacceptably, and to set stronger, faster rates on heavily loaded pins on high-speed channels that would otherwise run too slow. Another relatively common analog feature is differential comparators on input pins designed to be connected to differential signaling channels. A few "mixed signal FPGAs" have integrated peripheral analog-to-digital converters (ADCs) and digital-to-analog converters (DACs) with analog signal conditioning blocks allowing them to operate as a system-on-a-chip. Such devices blur the line between an FPGA, which carries digital ones and zeros on its internal programmable interconnect fabric, and field-programmable analog array (FPAA), which carries analog values on its internal programmable interconnect fabric.

II. BENEFITS OF FPGA TECHNOLOGY

Below are the benefits of FPGA Technology:

A. Performance
B. Time to Market
C. Cost
D. Reliability
E. Long-Term Maintenance

A. *Performance*

Taking advantage of hardware parallelism, FPGAs exceed the computing power of digital signal processors (DSPs) by breaking the paradigm of sequential execution and accomplishing more per clock cycle. BDTI, a noted analyst and benchmarking

Manuscript Received on October, 2012.

Syed Awais Hyder, Master Of Technology (VLSI System Design), VIF College of Engineering and Technology, Hyderabad, India.
D. Srikanth, Master Of Technology (VLSI System Design), VIF College of Engineering and Technology, Hyderabad, India.
C. Chandrasekhar, Electronics and Communication Engineering, Anantapur Jawaharlal Nehru Technological University, Anantapur, India..
E. Sammaiah, VLSI and Embedded System Design, Jawaharlal Nehru Technological University, Hyderabad, India.

firm, released benchmarks showing how FPGAs can deliver many times the processing power per dollar of a DSP solution in some applications.2 Controlling inputs and outputs (I/O) at the hardware level provides faster response times and specialized functionality to closely match application requirements.

B. Time to market

FPGA technology offers flexibility and rapid prototyping capabilities in the face of increased time-to-market concerns. You can test an idea or concept and verify it in hardware without going through the long fabrication process of custom ASIC design. You can then implement incremental changes and iterate on an FPGA design within hours instead of weeks. Commercial off-the-shelf (COTS) hardware is also available with different types of I/O already connected to a user-programmable FPGA chip. The growing availability of high-level software tools decreases the learning curve with layers of abstraction and often offers valuable IP cores (prebuilt functions) for advanced control and signal processing.

C. Cost

The nonrecurring engineering (NRE) expense of custom ASIC design far exceeds that of FPGA-based hardware solutions. The large initial investment in ASICs is easy to justify for OEMs shipping thousands of chips per year, but many end users need custom hardware functionality for the tens to hundreds of systems in development. The very nature of programmable silicon means you have no fabrication costs or long lead times for assembly. Because system requirements often change over time, the cost of making incremental changes to FPGA designs is negligible when compared to the large expense of re-spinning an ASIC.

D. Reliability

While software tools provide the programming environment, FPGA circuitry is truly a “hard” implementation of program execution. Processor-based systems often involve several layers of abstraction to help schedule tasks and share resources among multiple processes. The driver layer controls hardware resources and the OS manages memory and processor bandwidth. For any given processor core, only one instruction can execute at a time, and processor-based systems are continually at risk of time-critical tasks preempting one another. FPGAs, which do not use OSs, minimize reliability concerns with true parallel execution and deterministic hardware dedicated to every task.

E. Long-term maintenance

As mentioned earlier, FPGA chips are field-upgradable and do not require the time and expense involved with ASIC redesign. Digital communication protocols, for example, have specifications that can change over time, and ASIC-based interfaces may cause maintenance and forward-compatibility challenges. Being reconfigurable, FPGA chips can keep up with future modifications that might be necessary. As a product or system matures, you can make functional enhancements without spending time redesigning hardware or modifying the board layout.

III. FPGA COMPARISONS

Historically, FPGAs have been slower, less energy efficient and generally achieved less functionality than their fixed ASIC counterparts. A study has shown that designs implemented on FPGAs need on average 40 times as much area, draw 12 times as much dynamic power, and are three times slower than the corresponding ASIC implementations. An Altera Cyclone II FPGA, on an Altera teraSIC DE1 Prototyping board .Advantages include the ability to re-program in the field to fix bugs, and may include a shorter time to market and lower non-recurring engineering costs. Vendors can also take a middle road by developing their hardware on ordinary FPGAs, but manufacture their final version so it can no longer be modified after the design has been committed. Xilinx claims that several market and technology dynamics are changing the ASIC/FPGA paradigm. Integrated circuit costs are rising aggressively ASIC complexity has lengthened development time R&D resources and headcount are decreasing Revenue losses for slow time-to-market are increasing Financial constraints in a poor economy are driving low-cost technologies These trends make FPGAs a better alternative than ASICs for a larger number of higher-volume applications than they have been historically used for, to which the company attributes the growing number of FPGA design starts (see History)Some FPGAs have the capability of partial re-configuration that lets one portion of the device be re-programmed while other portions continue running.

IV. COMPLEX PROGRAMMABLE LOGIC DEVICES

The primary differences between CPLDs (Complex Programmable Logic Devices) and FPGAs are architectural. A CPLD has a somewhat restrictive structure consisting of one or more programmable sum-of-products logic arrays feeding a relatively small number of clocked registers. The result of this is less flexibility, with the advantage of more predictable timing delays and a higher logic-to-interconnect ratio. The FPGA architectures, on the other hand, are dominated by interconnect. This makes them far more flexible (in terms of the range of designs that are practical for implementation within them) but also far more complex to design for. In practice, the distinction between FPGAs and CPLDs is often one of size as FPGAs are usually much larger in terms of resources than CPLDs. Typically only FPGA's contain more advanced embedded functions such as adders, multipliers, memory, serdes and other hardened functions. Another common distinction is that CPLDs contain embedded flash to store their configuration while FPGAs usually, but not always, require an external flash.

V. FPGA IMPLEMENTATION

After synthesis is done coming to FPGA implementation. For the FPGA implementation Virtex II pro broad and the logic analyzers are used.

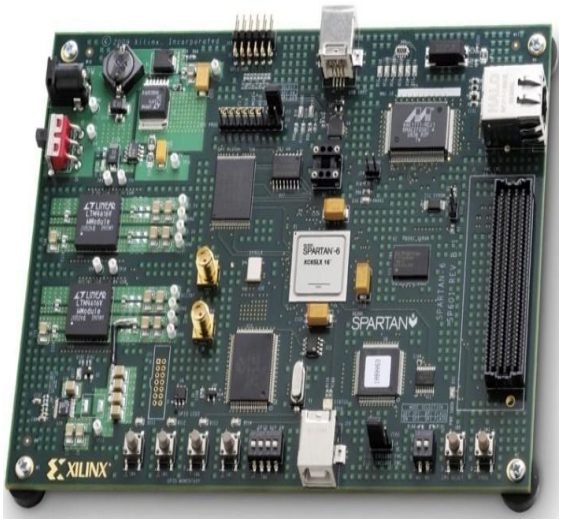


Fig 1: Programming/evaluation board

Security considerations: With respect to security, FPGAs have both advantages and disadvantages as compared to ASICs or secure microprocessors. FPGAs' flexibility makes malicious modifications during fabrication a lower risk... For many FPGAs, the loaded design is exposed while it is loaded (typically on every power-on). To address this issue, some FPGAs support bit stream encryption. although in July 2011, researchers published papers highlighting vulnerabilities in the bit stream encryption of some devices related to the analysis of the device's power usage fluctuations. These vulnerabilities apply to the current devices of most FPGA manufacturers, including Altera and Xilinx.

Modern developments: A recent trend has been to take the coarse-grained architectural approach a step further by combining the logic blocks and interconnects of traditional FPGAs with embedded microprocessors and related peripherals to form a complete "system on a programmable chip". This work mirrors the architecture by Ron Perlof and Hana Potash of Burroughs Advanced Systems Group which combined a reconfigurable CPU architecture on a single chip called the SB24. That work was done in 1982. Examples of such hybrid technologies can be found in the Xilinx Virtex-II PRO and Virtex-4 devices, which include one or more PowerPC processors embedded within the FPGA's logic fabric. The Atmel FPSLIC is another such device, which uses an AVR processor in combination with Atmel's programmable logic architecture. The Actel SmartFusion devices incorporate an ARM architecture Cortex-M3 hard processor core (with up to 512 kB of flash and 64 kB of RAM) and analog peripherals such as a multi-channel ADC and DACs to their flash-based FPGA fabric. In 2010, an extensible processing platform was introduced for FPGAs that fused features of an ARM high-end microcontroller (hard-core implementations of a 32-bit processor, memory, and I/O) with an FPGA fabric to make FPGAs easier for embedded designers to use. By incorporating the ARM processor-based platform into a 28 nm FPGA family, the extensible processing platform enables system architects and embedded software developers to apply a combination of serial and parallel processing to address the challenges they face in designing today's embedded systems, which must meet ever-growing demands to perform highly complex functions. By allowing them to design in a familiar ARM environment, embedded designers can benefit from the time-to-market advantages of an FPGA platform compared to more traditional design cycles associated with ASIC. An alternate approach to using hard-macro processors is to make use of soft processor cores that are implemented within the FPGA logic.

Market size: 1985: First commercial FPGA technology invented by Xilinx 1987: $14 million
~1993: >$385 million
2005: $1.9 billion
2010 estimates: $2.75 billion.

FPGA design starts:
2005: 80,000,
2008: 90,000.

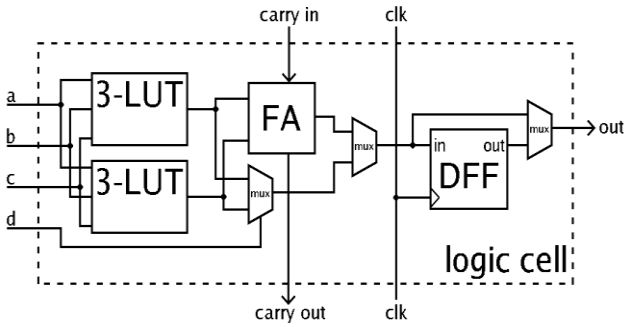
VI. APPLICATIONS

Applications of FPGAs include digital signal processing, software-defined radio, aerospace and defense systems, ASIC prototyping, medical imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation, radio astronomy, metal detection and a growing range of other areas. FPGAs originally began as competitors to CPLDs and competed in a similar space, that of glue logic for PCBs. As their size, capabilities, and speed increased, they began to take over larger and larger functions to the state where some are now marketed as full systems on chips (SoC). Particularly with the introduction of dedicated multipliers into FPGA architectures in the late 1990s, applications which had traditionally been the sole reserve of DSPs began to incorporate FPGAs instead. Traditionally, FPGAs have been reserved for specific vertical applications where the volume of production is small for these low-volume applications.

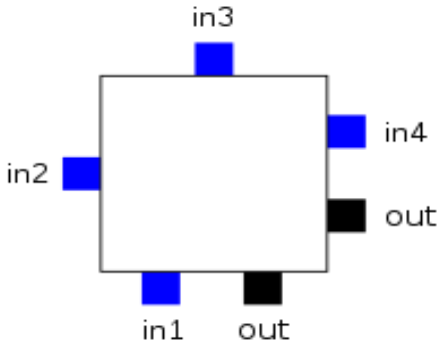
VII. ARCHITECTURE

The most common FPGA architecture consists of an array of logic blocks (called Configurable Logic Block, CLB, or Logic Array Block, LAB, depending on vendor), I/O pads, and routing channels. Generally, all the routing channels have the same width (number of wires). Multiple I/O pads may fit into the height of one row or the width of one column in the array. An application circuit must be mapped into an FPGA with adequate resources. While the number of CLBs/LABs and I/Os required is easily determined from the design, the number of routing tracks needed may vary considerably even among designs with the same amount of logic. For

example, a crossbar switch requires much more routing than a systolic array with the same gate count. Since unused routing tracks increase the cost (and decrease the performance) of the part without providing any benefit, FPGA manufacturers try to provide just enough tracks so that most designs that will fit in terms of Lookup tables (LUTs) and IOs can be routed.

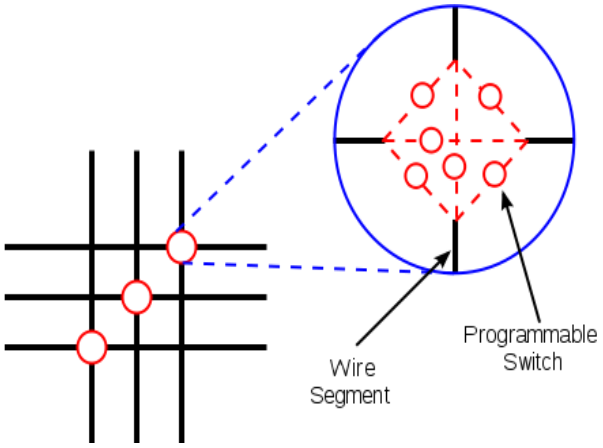


In general, a logic block (CLB or LAB) consists of a few logical cells (called ALM, LE, Slice etc.). A typical cell consists of a 4-input LUT, a Full adder (FA) and a D-type flip-flop, as shown below. The LUTs are in this figure split into two 3-input LUTs. In normal mode those are combined into a 4-input LUT through the left mux. In arithmetic mode, their outputs are fed to the FA. The selection of mode is programmed into the middle multiplexer. The output can be either synchronous or asynchronous, depending on the programming of the mux to the right, in the figure example. In practice, entire or parts of the FA are put as functions into the LUTs in order to save space.



Logic_block_pins.svg (SVG file, nominally 205 × 185 pixels, file size: 6 KB)This image rendered as PNG in other sizes: 200px, 500px, 1000px, 2000px.Each input is accessible from one side of the logic block, while the output pin can connect to routing wires in both the channel to the right and the channel below the logic block. Each logic block output pin can connect to any of the wiring segments in the channels adjacent to it. Similarly, an I/O pad can connect to any one of the wiring segments in the channel adjacent to it. For example, an I/O pad at the top of the chip can connect to any of the W wires (where W is the channel width) in the horizontal channel immediately below it. Generally, the FPGA routing is segmented. That is, each wiring segment spans one logic block before it terminates in a switch box. By turning on some of the programmable switches within a switch box, longer paths can be constructed. For higher speed interconnect, some FPGA architectures use longer routing lines that span multiple logic blocks. Whenever a vertical and a horizontal channel intersect, there is a switch box. In this architecture, when a wire enters a switch box, there are three programmable switches that allow it to connect to three other wires in adjacent channel segments. The pattern, or topology, of switches used in this architecture is the planar or domain-based switch box topology.

Modern FPGA families expand upon the above capabilities to include higher level functionality fixed into the silicon. Having these common functions embedded into the silicon reduces the area required and gives those functions increased speed compared to building them from primitives. Examples of these include multipliers, generic DSP blocks, embedded processors, high speed IO logic and embedded memories. FPGAs are also widely used for systems validation including pre-silicon validation, post-silicon validation, and firmware development. This allows chip companies to validate their design before the chip is produced in the factory, reducing the time-to-market. To shrink the size and power consumption of FPGAs, vendors such as Tabula and Xilinx have introduced new 3D or stacked Architectures. Following the introduction of its 28 nm 7-series FPGAs, Xilinx revealed that several of the highest-density parts in those FPGA product lines will be constructed using multiple dies in one package, employing technology developed for 3D construction and stacked-die assemblies.



Basic Process Technology Types: SRAM - based on static memory technology. In-system programmable and re-programmable requires external boot devices.

CMOS currently in use.

Antifuse - One-time programmable CMOS.

PROM - Programmable Read-Only Memory technology. One-time programmable because of plastic packaging. Obsolete.

EPROM - Erasable Programmable Read-Only Memory technology. One-time programmable but with window, can be erased with ultraviolet (UV) light. CMOS Obsolete.

EEPROM - Electrically Erasable Programmable Read-Only Memory technology. Can be erased, even in plastic packages. Some but not all EEPROM devices can be in-system programmed CMOS.

Flash - Flash-erase EPROM technology. can be erased, even in plastic packages. Some but not all flash devices can be in-system programmed. Usually, a flash cell is smaller than an equivalent EEPROM cell and is therefore less expensive to manufacture CMOS.

Major Manufactures: Xilinx and Altera are the current FPGA market leaders and long-time industry rivals. Together, they control over 80 percent of the

market,[40] with Xilinx alone representing over 50 percent .Both Xilinx and Altera provide free Windows and Linux design software which provides limited set of devices. Other competitors include Lattice Semiconductor (SRAM based with integrated configuration Flash, instant-on, low power, live reconfiguration), Actel (now Micro semi, anti fuse, flash-based, mixed-signal), Silicon Blue Technologies (extremely low power SRAM-based FPGAs with optional integrated nonvolatile configuration memory), Achro nix (RAM based, 1.5 GHz fabric speed) who will be building their chips on Intels' 22 nm process, and Quick Logic (handheld focused CSSP, no general purpose FPGAs).In March 2010, Tabula announced their FPGA technology.

VIII. ACKNOWLEDGMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowned our efforts with success. We extend our deep sense of gratitude to Principal Mr. P. Srinivas Rao, VIF College of Engineering and Technology, HOD, Ms. Imtiyaaz, VIF College of Engineering and Technology, Dept- Electronics Communication Engineering for her support and encouragement. I am indebted to Ms. Zubaida, Associate professor, VIF College of Engineering and Technology ,786, Himayath Nagar, Gandipet 'X' Road,Moinabad Mandal,Ranga Reddy Dist - 500 075,Andhra Pradesh, India for giving her helpful comments and sharing ideas in carrying out this research work.

REFERENCES

1. a b c FPGA Architecture for the Challenge
2. FPGA Signal Integrity tutorial
3. NASA: FPGA drive strength
4. Mike Thompson. "Mixed-signal FPGAs provide GREEN POWER". EE Times, 2007-07-02.
5. a b c History of FPGAs
6. Google Patent Search, "Re-programmable PLA". Retrieved February 5, 2009.
7. Google Patent Search, "Dynamic data re-programmable PLA". Retrieved February 5, 2009.
8. Peter Clarke, EE Times, "Xilinx, ASIC Vendors Talk Licensing." June 22, 2001. Retrieved February 10, 2009.
9. Funding Universe. “Xilinx, Inc.” Retrieved January 15, 2009.
10. Clive Maxfield, Programmable Logic DesignLine, "Xilinx unveil revolutionary 65nm FPGA architecture: the Virtex-5 family. May 15, 2006. Retrieved February 5, 2009.
11. Press Release, "Xilinx Co-Founder Ross Freeman Honored as 2009 National Inventors Hall of Fame Inductee for Invention of FPGA"
12. a b Clive Maxfield, book, "The Design Warrior's Guide to FPGAs".Published by Elsevier, 2004. ISBN 0-7506-7604-3,ISBN 978-0-7506-7604-5. Retrieved February 5, 2009
13. McConnel, Toni. EETimes. "ESC - Xilinx Extensible Processing Platform combines best of serial and parallel processing." April 28, 2010. Retrieved February 14, 2011.
14. Cheung, Ken, FPGA Blog. "Xilinx Extensible Processing Platform for Embedded Systems." April 27, 2010. Retrieved February 14, 2011.
15. Nass, Rich, EETimes. "Xilinx puts ARM core into its FPGAs." April 27, 2010. Retrieved February 14, 2011.
16. Leibson, Steve, Design-Reuse. "Xilinx redefines the high-end microcontroller with its ARM-based Extensible Processing Platform - Part 1." May. 03, 2010. Retrieved February 15, 2011.
17. Wilson, Richard, Electronics Weekly. "Xilinx acquires ESL firm to make FPGAs easier to use." January 31, 2011. Retrieved February 15, 2011.
18. a b Dylan McGrath, EE Times, "FPGA Market to Pass $2.7 Billion by 10, In-Stat Says". May 24, 2006. Retrieved February 5, 2009.
19. Dylan McGrath, EE Times, "Gartner Dataquest Analyst Gives ASIC, FPGA Markets Clean Bill of Health". June 13, 2005.
20. Retrieved February 5, 2009.
21. Virtex-4 Family Overview
22. Kuon, I.; Rose, J. (2006). "Measuring the gap between FPGAs and ASICs". Proceedings of the internation symposium on Field programmable gate arrays - FPGA'06. pp. 21.doi:10.1145/ 1117201. 1117205. ISBN 1595932925. edit
23. a b Tim Erjavec, White Paper, "Introducing the Xilinx Targeted Design Platform: Fulfilling the Programmable Imperative." February 2, 2009. Retrieved February 2, 2009
24. Huffmire Paper "Managing Security in FPGA-Based Embedded Systems." Nov-Dec 2008. Retrieved Sept 22, 2009
25. "Virtex-5 FPGA Configuration User Guide". Xilinx Inc.. August 2010. pp. 33–35. Retrieved 2010-10-31.
26. "Protecting Intellectual Property Through FPGA Design Security". Altera Corporation. Retrieved 2010-11-01.
27. Wiśniewski, Remigiusz (2009). Synthesis of compositional microprogram control units for programmable devices. Zielona Góra: University of Zielona Góra. pp. 153. ISBN 978-83-7481-293.