Introduction to Digital Design
Using Digilent FPGA Boards
— Block Diagram / Verilog Examples

Richard E. Haskell
Darrin M. Hanna

Oakland University, Rochester, Michigan

NOTE: For the FPGA Labs you will NOT be using the Altec Active HDL software. Instead, you will be using the Xilinx tool chain as described in the Screen-shot document.

LBE Books
Rochester Hills, MI

Introduction

Digital Design Using FPGAs

The first integrated circuits that were developed in the early 1960s contained less that 100 transistors on a chip and are called small-scale integrated (SSI) circuits. Medium-scale integrated (MSI) circuits, developed in the late 1960s, contain up to several hundreds of transistors on a chip. By the mid 1970s large-scale integrated (LSI) circuits containing several thousands of transistors had been developed. Very-large-scale integrated (VLSI) circuits containing over 100,000 transistors had been developed by the early 1980s. This trend has continued to the present day with 1,000,000 transistors on a chip by the late 1980s, 10,000,000 transistors on a chip by the mid-1990s, over 100,000,000 transistors by 2004, and up to 1,000,000,000 transistors on a chip today. This exponential growth in the amount of digital logic that can be packed into a single chip has produced serious problems for the digital designer. How can an engineer, or even a team of engineers, design a digital logic circuit that will end up containing millions of transistors?

In Appendix C we show that any digital logic circuit can be made from only three types of basic gates: AND, OR, and NOT. In fact, we will see that any digital logic circuit can be made using only NAND gates (or only NOR gates), where each NAND or NOR gate contains four transistors. These basic gates were provided in SSI chips using various technologies, the most popular being transistor-transistor logic (TTL). These TTL chips were the mainstay of digital design throughout the 1960s and 1970s. Many MSI TTL chips became available for performing all types of digital logic functions such as decoders, adders, multiplexers, comparators, and many others.

By the 1980s thousands of gates could fit on a single chip. Thus, several different varieties of *programmable logic devices* (PLDs) were developed in which arrays containing large numbers of AND, OR, and NOT gates were arranged in a single chip without any predetermined function. Rather, the designer could design any type of digital circuit and implement it by connecting the internal gates in a particular way. This is usually done by opening up fuse links within the chip using computer-aided tools. Eventually the equivalent of many PLDs on a single chip led to *complex programmable logic devices* (CPLDs).

Field Programmable Gate Arrays (FPGAs)

A completely different architecture was introduced in the mid-1980's that uses RAM-based lookup tables instead of AND-OR gates to implement combinational logic. These devices are called *field programmable gate arrays* (FPGAs). The device consists of an array of *configurable logic blocks* (CLBs) surrounded by an array of I/O blocks. The Spartan-3E from Xilinx also contains some blocks of RAM, 18 x 18 multipliers, as well as Digital Clock Manager (DCM) blocks. These DCMs are used to eliminate clock distribution delay and can also increase or decrease the frequency of the clock.

Each CLB in the Spartan-3E FPGA contains four slices, each of which contains two 16 x 1 RAM look-up tables (LUTs), which can implement any combinational logic function of four variables. In addition to two look-up tables, each slice contains two D flip-flops which act as storage devices for bits. The basic architecture of a Spartan-3E FPGA is shown in Fig. 1.

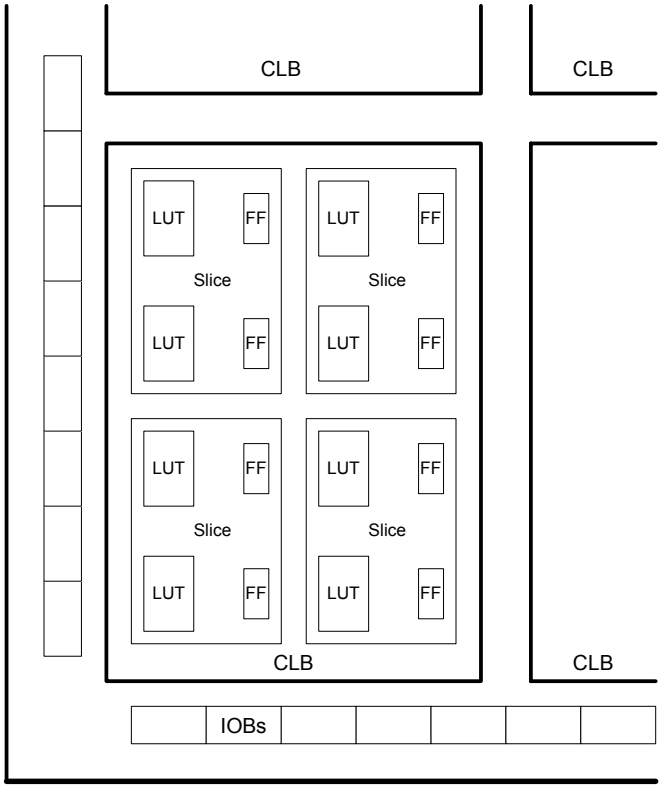


Figure 1 Architecture of a Spartan-3E FPGA

The BASYS board from Digilent contains a Xilinx Spartan3E-100 TQ144 FPGA. This chip contains 240 CLBs arranged as 22 rows and 16 columns. There are therefore 960 slices with a total of 1,920 LUTs and flip-flops. This part also contains 73,728 bits of block RAM. Half of the LUTs on the chip can be used for a maximum of 15,360 bits of distributed RAM.

By contrast the Nexys-2 board from Digilent contains a Xilinx Spartan3E-500 FG320 FPGA. This chip contains 1,164 CLBs arranged as 46 rows and 34 columns. There are therefore 4,656 slices with a total of 9,312 LUTs and flip-flops. This part also contains 368,640 bits of block RAM. Half of the LUTs on the chip can be used for a maximum of 74,752 bits of distributed RAM.

In general, FPGAs can implement much larger digital systems than CPLDs as illustrated in Table 1. The column labeled *No. of Gates* is really equivalent gates as we have seen that FPGAs really don't have AND and OR gates, but rather just RAM look-up tables. (Each slice does include two AND gates and two XOR gates as part of carry and arithmetic logic used when implementing arithmetic functions including addition and

multiplication.) Note from Table 1 that FPGAs can have the equivalent of millions of gates and tens of thousands of flip-flops.

Table 1 Comparing Xilinx CPLDs and FPGAs

Xilinx Part	No. of Gates	No. of I/Os	No. of CLBs	No. of Flip-flops	Block RAM (bits)
CPLDs					
9500 family	800 – 6,400	34 – 192		36 - 288	
FPGAs					
Spartan	5,000 – 40,000	77 – 224	100 – 784	360 – 2,016	
Spartan II	15,000 – 200,000	86 – 284	96 – 1,176	642 – 5,556	16,384 – 57,344
Spartan IIE	23,000 – 600,000	182 – 514	384 – 3,456	2,082 – 15,366	32,768 – 294,912
Spartan 3	50,000 – 5,000,000	124 – 784	192 – 8,320	2,280 – 71,264	73,728 – 1,916,928
Spartan-3E	100,000 – 1,600,000	108 – 376	240 – 3,688	1,920 – 29,505	73,728 – 663,552
Virtex	57,906 – 1,124,022	180 – 512	384 – 6,144	2,076 – 26,112	32,768 – 131,072
Virtex E	71,693 – 4,074,387	176 – 804	384 – 16,224	1,888 – 66,504	65,536 – 851,968
Virtex-II	40,960 – 8,388,608	88 – 1,108	64 – 11,648	1,040 – 99,832	73,728 – 3,096,576

Modern Design of Digital Systems

The traditional way of designing digital circuits is to draw logic diagrams containing SSI gates and MSI logic functions. However, by the late 1980s and early 1990s such a process was becoming problematic. How can you draw schematic diagrams containing hundreds of thousands or millions of gates? As programmable logic devices replaced TTL chips in new designs a new approach to digital design became necessary. Computer-aided tools are essential to designing digital circuits today. What has become clear over the last decade is that today's digital engineer designs digital systems by writing software! This is a major paradigm shift from the traditional method of designing digital systems. Many of the traditional design methods that were important when using TTL chips are less important when designing for programmable logic devices.

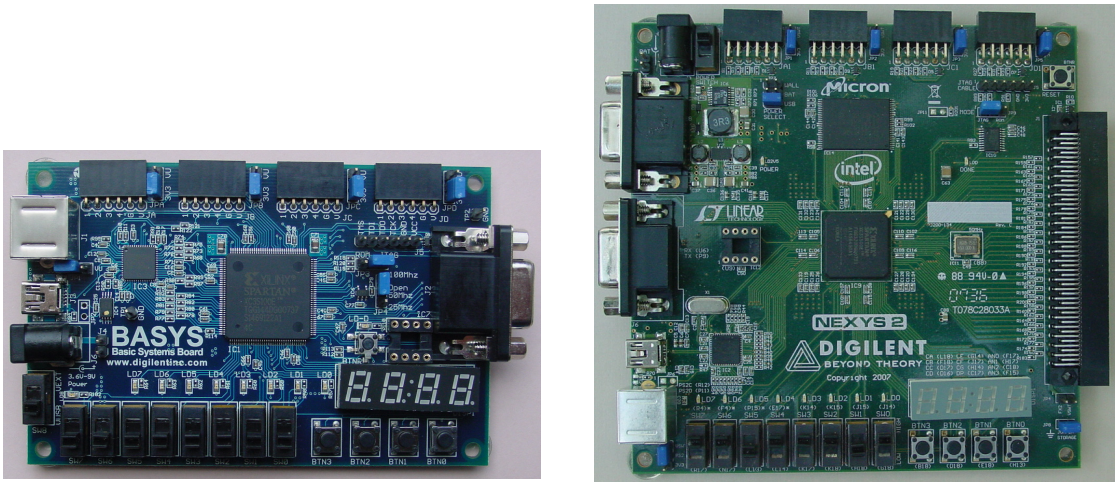
Today digital designers use *hardware description languages* (HDLs) to design digital systems. The most widely used HDLs are VHDL and Verilog. Both of these hardware description languages allow the user to design digital systems by writing a program that describes the behavior of the digital circuit. The program can then be used to both *simulate* the operation of the circuit and *synthesize* an actual implementation of the circuit in a CPLD, an FPGA, or an application specific integrated circuit (ASIC).

Another recent trend is to design digital circuits using block diagrams or graphic symbols that represent higher-level design constructs. These block diagrams can then be *compiled* to produce Verilog or VHDL code. We will illustrate this method in this book.

We will use Active-HDL from Aldec for designing our digital circuits. This integrated tool allows you to enter your design using either a block diagram editor (BDE) or by writing Verilog or VHDL code using the hardware description editor (HDE). Once your hardware has been described you can use the functional simulator to produce waveforms that will verify your design. This hardware description can then be synthesized to logic equations and implemented or mapped to the FPGA architecture.

We include a tutorial for using Active-HDL in Appendix A. A free student version of Active-HDL is available on their website.[1] We will use Xilinx ISE for synthesizing our VHDL designs. You can download a free version of ISE™ WebPACK™ from the Xilinx website.[2] This WebPACK™ synthesis tool can be run from within the Aldec Active-HDL development environment as shown in the tutorial in Appendix A. The implementation process creates a *.bit* file that is downloaded to a Xilinx FPGA on the BASYS board or Nexys-2 shown in Fig. 2. The BASYS board is available to students for $59 from Digilent, Inc.[3] This board includes a 100k-gate equivalent Xilinx Spartan3E FPGA (250k-gate capacity is also available), 8 slide switches, 4 pushbutton switches, 8 LEDs, and four 7-segment displays. The frequency of an on-board clock can be set to 25 MHz, 50 MHz, or 100 MHz using a jumper. There are connectors that allow the board to be interfaced to external circuits. The board also includes a VGA port and a PS2 port. The use of these ports are described in a different book.[4] Another more advanced board, the Nexys-2 board, is also available to students for $99 from Digilent. The Nexys-2 board is similar to the BASYS board except that it contains a 500k- or 1200k-gate equivalent Spartan 3E FPGA, a Hirose FX2 interface for additional add-on component boards, 16 MB of cellular RAM, 16 MB of flash memory, a 50 MHz clock and a socket for a second oscillator. The Nexys-2 is ideally suited for embedded processors.

All of the examples in this book can be used on both the BASYS board and the Nexys-2 board. The only difference is that you would use the file *basys2.ucf* to define the pinouts on the BASYS board and you would use the file *nexys2.ucf* to define the pinouts on the Nexys-2 board. Both of these files are available to download from www.lbebooks.com. Table 2 shows the jumper settings you would use on the two boards.



(a) (b)

Figure 2 (a) BASYS board, (b) Nexys-2 Board

[1] http://www.aldec.com/education/

[2] http://www.xilinx.com

[3] http://www.digilentinc.com

[4] *Digital Design Using Digilent FPGA Boards – Verilog / Active-HDL Edition*; available from www.lbebooks.com.

Table 1.2 Board Jumper Settings

BASYS Boad	Nexys-2 Board
Set the JP3 jumper to JTAG	Set the POWER SELECT jumper to USB
Remove the JP4 jumper to select a 50 MHz clock	Set the MODE jumper to JTAG

Verilog

Verilog is based on the C programming language but it is *not* C. Verilog is a *hardware description language* that is designed to model digital logic circuits. It simply has the same syntax as the C programming language but the way it behaves is different. In this book we begin by using the Active-HDL block diagram editor to draw logic circuits using basic gates. When you *compile* these block diagrams Active-HDL will generate the corresponding Verilog code. The block diagram representing your logic circuit can then be used as a module in a higher-level digital design. This higher-level design can then be compiled to produce its corresponding Verilog code. This hierachical block diagram editor will make it easy to design top-level designs.

Sometimes it will be easier to design a digital module by writing a Verilog program directly rather than drawing it using gates. When you do this you can still use the block diagram for this module in higher-level designs. We will illustrate this process in many of our examples.

Just like any programming language, you can only learn Verilog by actually writing Verilog programs and simulating the designs using a Verilog simulator that will display the waveforms of the signals in your design. This is a good way to learn not only Verilog but digital logic as well.

A companion book[5] that uses VHDL instead of Verilog is available from Digilent or www.lbebooks.com. More comprehensive Verilog and VHDL books are also available.[6,7]

---

[5] *Introduction to Digital Design Using Digilent FPGA Boards – Block Diagram / VHDL Examples*, LBE Books, 2009.

[6] *Digital Design Using Digilent FPGA Boards – Verilog / Active-HDL Edition*, LBE Books, 2009.

[7] *Digital Design Using Digilent FPGA Boards – VHDL / Active-HDL Edition*, LBE Books, 2009.