Experiment 3. Digital Circuit Prototyping Using FPGAs

Masud ul Hasan · Muhammad Elrabaa · Ahmad Khayyat – Version 151, 11 September 2015

Table of Contents

1. Objectives

- Get introduced to FPGA prototyping boards.
- Get familiar with the software tools used with FPGAs (design and simulation).
- Learn how to design digital circuits using schematic design entry, and how to bind board switches and LEDS to signals in your design.
- Prototype, simulate, and verify simple logic gates using an FPGA board.

2. Materials Required

- An FPGA prototyping board.
- Design and simulation software tools.

3. Background

In this experiment, you will get familiar with the FPGA boards available in the lab. FPGA stands for *Field Programmable Gate Array*, which is a configurable hardware device that allows you to implement and test complex digital circuit designs.

3.1. FPGA Devices, Families, and Boards

An FPGA device is an integrated circuit chip that is designed to be configured by a customer after manufacturing, hence *field-programmable*. An FGPA contains a lot of hardware resources that can be configured to implement any digital circuit design. Resources available on an FPGA chip include programmable logic blocks, reconfigurable interconnect, memory blocks, and often some special-purpose components such as DSP blocks or embedded processors.

FPGA vendors typically classify their FPGA products into families, where devices belonging to each family share many of their features, and are optimized for a certain type of applications.

There are few vendors for FPGAs, the most notable of which are Xilinx and Altera[1]. Some of Xilinx FPGA offerings include:

- Virtex families: Xilinx high-end and most expensive FPGA families. Examples: Virtex-4, Virtex-5, and Virtex-6 families.
- Spartan: Xilinx low-end and low-cost FPGA families. Examples: Spartan-6, Spartan-3, and Spartan-3E families.
- Artix-7: Xilinx mid-range, power-optimized family.

Some of Altera FPGA offerings include:

- Stratix series: Altera's high-end and most expensive FPGA families. Examples: Stratix IV, Stratix V, Stratix 10.
- Arria series: Altera's mid-range and power-efficient families. Examples: Arria II, Arria V, and Arria 10.
- Cyclone series: Altera's low-power, low-cost families. Examples: Cyclone III, Cyclone IV, and Cyclone V.

Each of these families includes a number of FPGA devices, or chips.

An FPGA board is a printed circuit board (PCB) that features an FPGA chip, in addition to a number of other devices that makes it useful for a given application. Examples of other devices include various types of memory, e.g. DDR3 RAM and Flash memory. An FPGA board also typically includes some input and output devices, such as push buttons, switches, and LED lights. For the FPGA chip to be able to interact with these input/output devices, the board connects some of the FPGA chip pins to these devices.

> *The Board, Family, and Chip Used in This Course*
> In this course, we are going to use the *Nexys3* board, by Digilent Inc., which features a Xilinx Spartan-6 FPGA chip. The specific chip is XC6LX16-CS324.
>
> For more information about the Nexys3 board, refer to its website at: http://www.digilentinc.com/nexys3/.

3.2. Design Software

FPGA vendors provide special software tools that allow users to capture their designs, simulate them, and implement them on their FPGA devices by producing a configuration bitstream that is used to configure the FPGA to perform the user-specified design. FPGA design software typically includes an IDE and a number of special-purpose, advanced programs for various stages of the design process.

The design process, often referred to as the *design flow*, involves a number of steps, which can all be performed using the vendor-provided design software tools.

Xilinx and Altera Software Tools

Xilinx has two sets of software tools, each supporting a different range of FPGA devices. Xilinx *ISE Design Suite* supports the device families and their previous generations: Spartan-6, Virtex-6, and Coolrunner. The newer Xilinx *Vivado* software suite supports the device families: Ultrascale, Virtex-7, Kintex-7, Artix-7, and Zynq-7000.

Altera provides *Quartus II* design software for all Altera devices, including the Stratix, Cyclone, and Aria families.

3.3. Design Flow

The design flow can slightly differ depending on the used design software. However, it generally involves the following steps:

- *Creating a project*: which includes all of your design files and settings.
- *Design entry*: where you specify or enter your design using one of a number of design entry methods.
- *Design simulatation* (optional): where you apply inputs and observe outputs, to verify the functionality of your design using simulation software.
- *Pin assignment*: where you connect the inputs and outputs of your design to specific pins on your FPGA device, so that you can interact with your design when it runs on the FPGA device.
- *Design synthesis and implementation*: where your design is compiled into a binary configuration file, or bitstream, that can be used to configure the FPGA device to implement your design.
- *Timing analysis* (optional): where you can check the delay of specific signals due to propagation through FPGA logic and interconnect resources.
- *FPGA device configuration*: where you download the configuration bitstream, of the design to the FPGA device. At the end of this step, your design would be running on the FPGA device.

The rest of this section explains the steps involved in the design flow in more detail.

3.3.1. Create a Project in the Design Software

A *project* is a collection of all the files needed to create and download a design to the FPGA device. When creating a new project, usually you need to specify the project name, the location of its files on the hard drive, the type of input files (also called source files), and the target FPGA device that this design is created for.

3.3.2. Create Your Design

To specify a digital design in your project, you add a source file to it. The most commonly used source formats are:

- Schematic drawing of the circuit
- Verilog description of the circuit
- VHDL description of the circuit

Verilog and VHDL are standard hardware description languages that are used to specify hardware design. We will explore using Verilog to specify designs in later experiments.

In this experiment, we will use schematic source files.

When using schematic design entry, the schematic editor provides a *library* of *symbols*. A symbol represents a circuit component, such as a logic gate. The library is a collection of ready-made design components. For example, you will find a symbol for the AND gate, and a symbol for the OR gate in the symbol library.

When creating a design, you must specify the inputs and the outputs of your design, so that you can connect them to the outside world. By specifying them in your schematic, you instruct the design software to make them available outside the FPGA, by connecting them later to pins of the FPGA chip.

To specify the input and output pins, use input and output markers in your schematic. Name your inputs and outputs appropriately to be able to identify them later.

Finally, you need to connect, or wire, all the components in your schematic design.

3.3.3. Simulate Your Design

Simulation allows for testing the design and verifying its functionality without having to use the FPGA hardware. Instead, you use simulation software, or a *simulator*.

To simulate the design, you need to specify values for all inputs, run the simulator, and observe the output values and compare them to the expected output to determine if the circuit is working properly.

See the Simulation of a Half Adder Circuit figure for an example simulation. The Implementing a Half Adder section explains what the simulated circuit does.

3.3.4. Pin Assignment: Connect Input and Output Pins

In order to control the inputs and observe the outputs when your design is downloaded to the FGPA board, you need to configure the input signals of your design to be connected to some input peripherals on the FPGA board, and to configure your output signals to be connected to some output peripherals on the FPGA board.

Proper pin assignment requires that you carefully choose the input and output pins of your FPGA chip. You can lookup which FPGA chip pins are connected to which board peripherals by consulting the boards' user manual.

Nexys 3 Spartan-6 FPGA Board

The website of the Nexys 3 FPGA board is: www.digilentinc.com/nexys3/. You can learn about which FPGA pins are connected to which input/output peripherals in this board by consulting the board's reference manual [nexys3].

3.3.5. Synthesize and Implement Your Design

The synthesis and implementation phases collectively refer to a number of more specific steps. The first step accepts your design specification as input (schematic, Verilog, or VHDL), and the last step produces the FPGA configuration bitstream that implements your input design on the target FPGA.

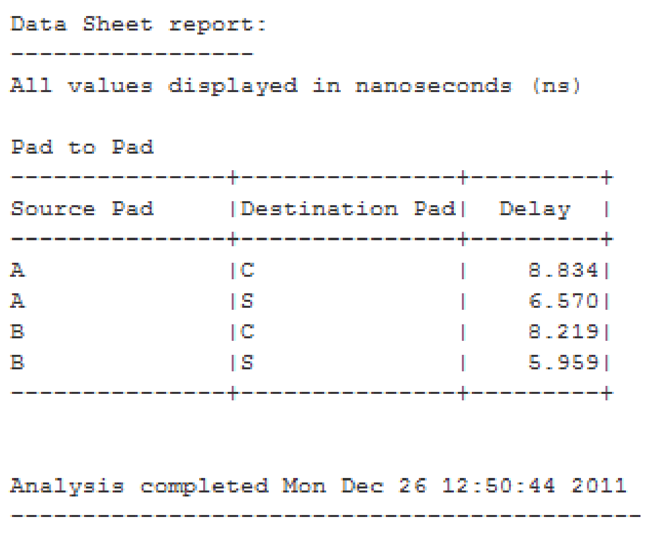
These specific steps include transforming your input design to *netlists*, *mapping* the design to FPGA resources, *placing* the required resources on specific FGPA resources, and *routing* signals through FPGA interconnect resources.

More generally, this step combines all the available information, including your design, information about the target device, and any other additional constraints, to generate the configuration bitstream of the design.

Depending on the size of your design, and the size of the target FPGA device, this step can take anywhere from a fraction of a minute to several hours.

3.3.6. Run Timing Analysis

The Timing Report Table figure shows an example timing report, which lists estimates of how long it will take signals to propagate through your design when it is *placed* on the FPGA and the signals are *routed* through.



```
Data Sheet report:
-----------------
All values displayed in nanoseconds (ns)

Pad to Pad
---------------+---------------+---------+
Source Pad     |Destination Pad|  Delay  |
---------------+---------------+---------+
A              |C              |    8.834|
A              |S              |    6.570|
B              |C              |    8.219|
B              |S              |    5.959|
---------------+---------------+---------+


Analysis completed Mon Dec 26 12:50:44 2011
--------------------------------------------
```

Figure 1. Timing Report Table

Timing reports allow you to verify whether your design can meet specific timing requirements.

3.3.7. Device Configuration (Programming)

Lastly, you *configure* the FPGA device by downloading the design configuration file (bitstream), which was generated at the end of the *synthesize and implement* step - you download it to the FPGA device.

To download the configuration file, you use the *programmer* tool of the design software suite. The programmer software can detect the FPGA device when it is connected to the development PC, typically using a USB cable.

Xilinx and Altera Programming Tools

The Xilinx programmer software included in the Xilinx design software suite, ISE, is called *iMPACT*.

Similarly, Altera provides *Quartus II Programmer* as a part of its Quartus II design software suite.

Once the design is downloaded to the FPGA device successfully, you will be able to verify your design of the circuit by controlling the configured inputs and observing the outputs.

3.4. Implementing a Half Adder

A half adder is a digital circuit that adds two bits, `A` and `B`, and outputs two bits, `S` for *sum*, and `C` for *carry*. The Half Adder Circuit figure shows how to implement a half adder using an XOR gate and an AND gate.

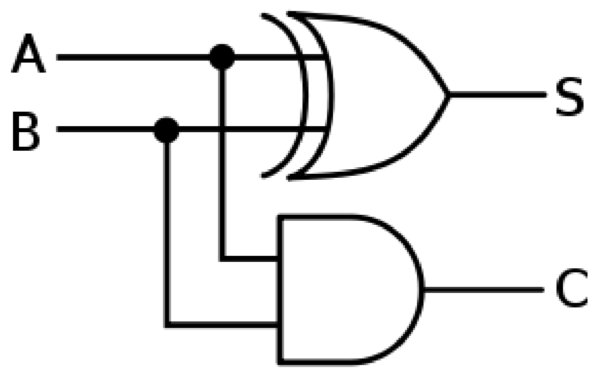


Figure 2. Half Adder Circuit

3.4.1. Simulating a Half Adder

The Simulation of a Half Adder Circuit figure shows the behavior of the half-adder circuit, as obtained from a simulator. The figure shows the two inputs (`A` and `B`) and the two outputs (`S` and `C`). It also shows how the input values are changing over time, and how the outputs are responding to these changes.

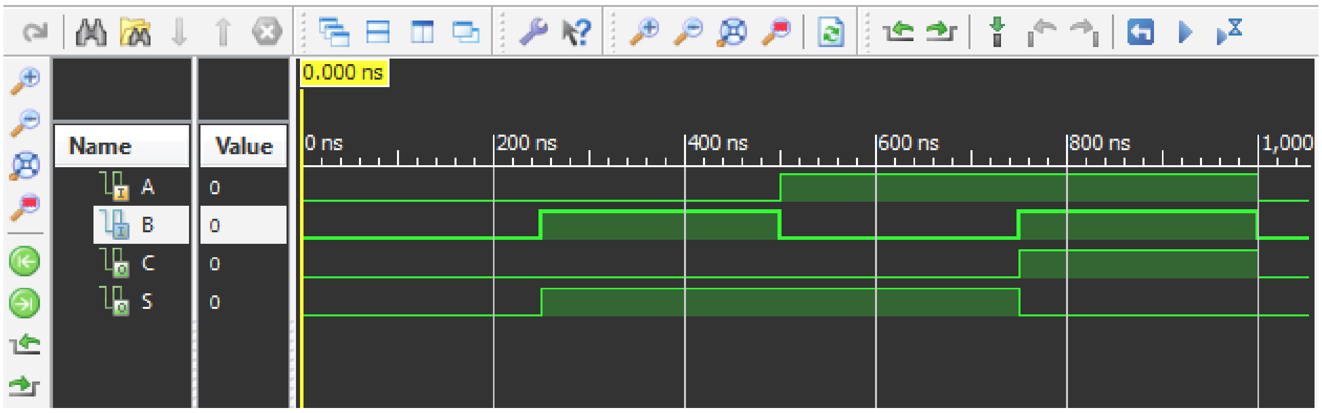


Figure 3. Simulation of a Half Adder Circuit

Exercise

Consider the Simulation of a Half Adder Circuit figure at time 400 ns.

- What are values of each input (`A` and `B`)?
- What are the corresponding values of each output (`S` and `C`)?
- According to these values, is the circuit behaving correctly?

3.4.2. Half Adder Pin Assignment

To implement a half adder on an FPGA, the design must be created first. One way to create the design is using a schematic drawing, as explained in the Create Your Design section.

Then, the inputs of your half adder circuit, `A` and `B`, should be connected to two of the input pins of the FPGA chip that are already connected to switches on the board. Likewise, the outputs of your circuit, `S` and `C` should be connected to two output pins of the FPGA chip that are already connected to LEDs on the board.

This will allow you to manually test the circuit on the board by manipulating the connected switches and observing the connected LEDs.

4. Tasks

In this experiment, you will create a digital circuit for a *half adder* and for a *full adder*. A full adder is a digital circuit that adds three bits.

4.1. Implement a Half Adder

1. Create a new project. Name it `half_adder`.
2. Create a schematic circuit design for a half adder.
3. Verify the functionality of your design using behavioral simulation

> Behavioral simulation is also known as *functional simulation*, or *RTL simulation*.

4. Assign the input and output (I/O) ports of the design to physical switches and LEDs on the board.
5. Implement your design and download it to the FPGA device.
6. Verify that your design is working correctly on the FPGA board.

4.2. Implement a Full Adder

Go through the same steps for implementing a half adder to implement a full adder. Name your project `full_adder`. A full adder is a digital circuit that adds three bits, `A`, `B` and `Cin` (*carry in*), and outputs two bits, `S` for *sum*, and `Cout` for *carry out*. The Full Adder Circuit figure shows how to implement a full adder using different gates.

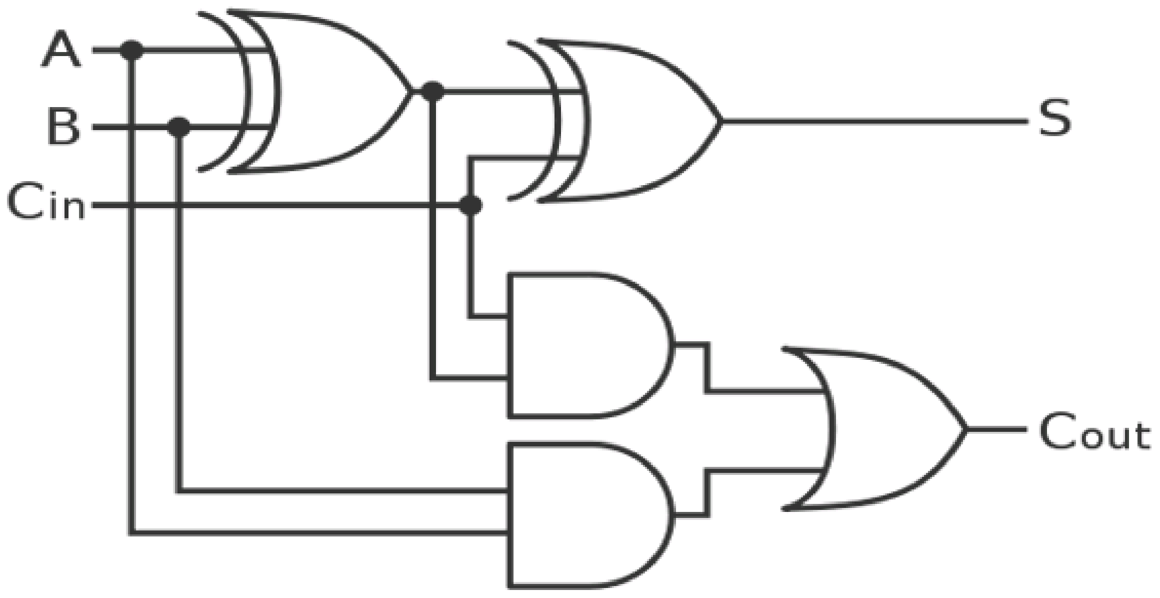


Figure 4. Full Adder Circuit

5. Grading Sheet

Task	Points
Half adder simulation	15

Task	Points
Half adder implementation on FPGA board	15
Full adder simulation	20
Full adder implementation on FPGA board	25
Lab notebook and discussion	25

6. Resources

[nexys3]

Digilent Inc. *Nexys3 Board Reference Manual.* April 3, 2013. Doc: 502-182. https://www.digilentinc.com/Data/Products/NEXYS3/Nexys3_rm.pdf

1. Altera was acquired by Intel in June 2015: http://fortune.com/2015/08/27/why-intel-altera/.

Version 151
Last updated 2016-02-06 01:18:47 AST