Ve 270 Introduction to Logic Design

Lab 4
Design of a Counter

UM-SJTU Joint Institute
Shanghai Jiao Tong University
October 2017

1. Objective

- To understand and design a special FSM – Counter.
- To design the counter by HDL modeling
- To experience the development process of an FPGA based digital device using HDL.

2. Requirement

In this lab, you are to design a device commonly used in modern computers, a 4-bit up/down synchronous binary counter. The counter is defined as follows:

- A 4-bit binary counter uses 4 flip-flop outputs to represent a binary number. It is capable of counting from 0 to 15.
- A synchronous counter implies that all flip-flops are sharing the same clock signal.
- An up/down counter is capable of counting up (incrementing) or down (decrementing).

The counter has two 1-bit control inputs called "Up/Down" and "Reset". The counter is to increment if "Up/Down" is 1 and decrement if "Up/Down" is 0. When "Reset" is 1 the counter should go to 0 no matter what the other inputs might be. "Up/Down" input can be provided with a toggle switch. The "Reset" input can be provided with a push button. Another input to the counter is "Clock". Rising edges of "Clock" should trigger increment or decrement of the counter.

Model the 4-bit up/down synchronous binary counter with Verilog HDL.

The outputs of the counter should be captured with 4 LEDs on the FPGA board, **and also supplied to an SSD Driver modeled with Verilog HDL so that the binary number is displayed in its hexadecimal equivalent**.

NOTE: the "Clock" signal can be connected to a push button or any other regular input source directly. Thus, each button press produces a rising edge.

3. Setting up Xilinx Vivado

In this section, you will set up the Xilinx Vivado digital system development environment and get it ready for circuit simulation and implementation.

1) Click on the icon to open the Vivado software.
2) Click "Create Project" in the left side panel shown below or from the drop down menu: File → New Project;
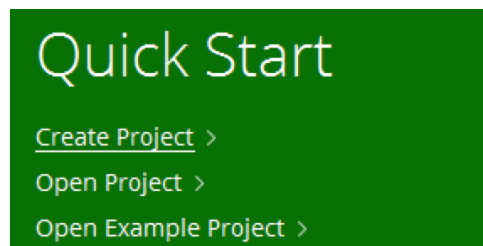
3) Follow the navigator to choose project file location and project name;

4) Select "RTL Project" option in the Project Type window, and Next;

You could check "Do not specify sources at this time" box if you want to add source file later.

5) Add Sources (skip for now)
6) Add Constrains (skip for now)
7) Choose Default Part. There are two ways to choose parts, select by Parts or by Boards:
   a. Select by Parts:
      Search and Choose parts: **XC7A35TCPG236-1**
   b. Select by Boards:
      i. Download Digilent Vivado boards file from github.com.
      ii. Copy folders in "new/board_files" into "vivado install location/Vivado/version/data/boards/board_files". For detail, please refer to the instruction on Digilent website.
      iii. You will find board file for Basys3

8) Finish

Now a new project is created in the Vivado software.

9) Find "Sources" window and click "+" to add or create source;
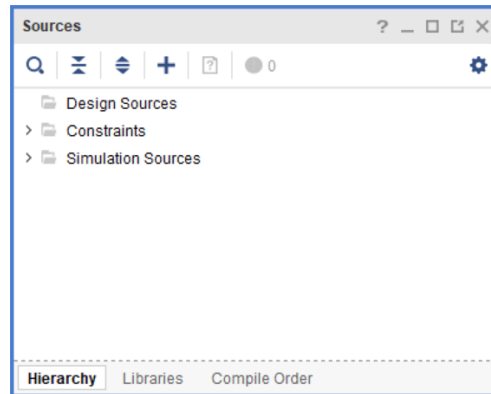
10) Choose "Add or create design sources"; Next;
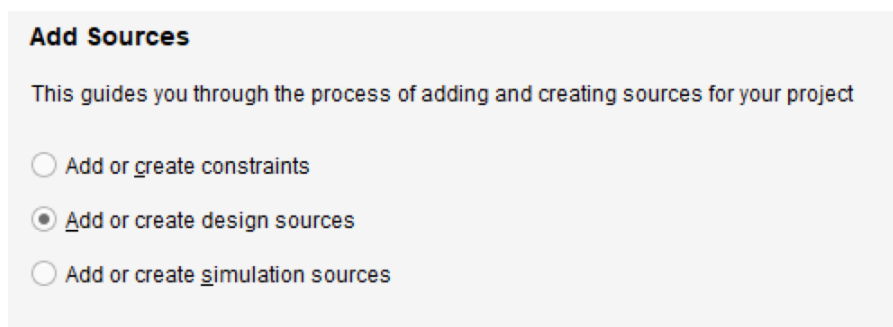
11) Click "Create File"
12) Choose "File type" to be Verilog, enter File name.

13) Click "OK" and "Finish"; then a "Define Module" window will pop up. You can skip this step, or you can also name your module ports here;
14) Write Verilog code in the file to implement your design;
15) Repeat steps 9) to 14) to create multiple source files;

4. Simulate your Verilog design

In this section, you will create a test bench to simulate your Verilog design in Vivado.

1) Find "Sources" window and click "+" to add source (test bench);
2) Choose "Add or create simulation sources";

Add Sources

This guides you through the process of adding and creating sources for your project
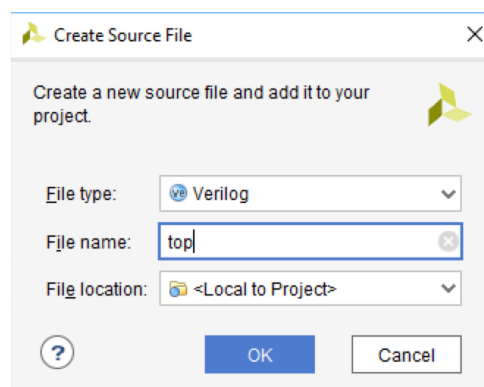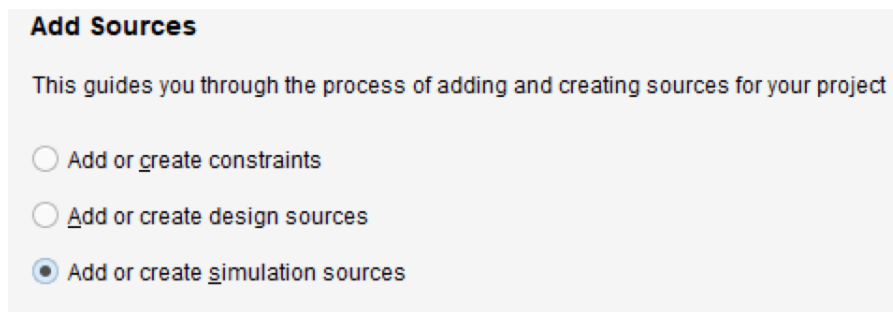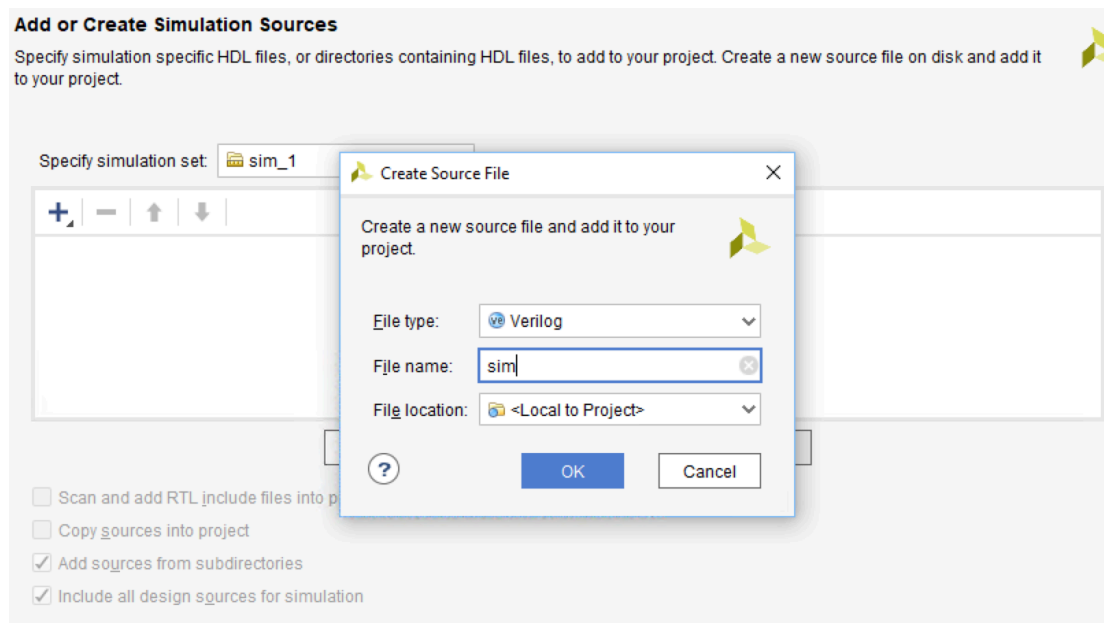
○ Add or _c_reate constraints

○ Add or create design sources

● Add or create _s_imulation sources

3) Click "Create File", Name file and choose location; **Note**: the simulation source (test bench) will also be a Verilog file.

Add or Create Simulation Sources

Specify simulation specific HDL files, or directories containing HDL files, to add to your project. Create a new source file on disk and add it to your project.

Specify simulation set: 📁 sim_1

Create Source File   ✕

Create a new source file and add it to your project.

File type: 🆅 Verilog   ⌄

File name: sim   ⊗

File location: 📁 <Local to Project> ⌄

(?)   OK   Cancel

☐ Scan and add RTL include files into p
☐ Copy _s_ources into project
☑ Add sources from subdirectories
☑ Include all design sources for simulation

4) Click OK and Finish.
5) Create a test bench Verilog file in the simulation source;
6) From the drop down menu: Flow → Run Simulation → Run Behavioral Simulation; the input/output waveforms of your circuit will appear; this can also be done by clicking Run Simulation → Run Behavioral Simulation in the "Flow Navigator" window on the left hand side;
7) Verify the functions of your design using the timing diagram;

5. Synthesize and implement the design

After a successful simulation, you will synthesize and implement your design with Vivado tools and Basys 3 FPGA board.

1) Highlight the top Verilog module of your design;
2) Run Synthesis in "Flow Navigator" with default setting to convert the Verilog models into real circuits;
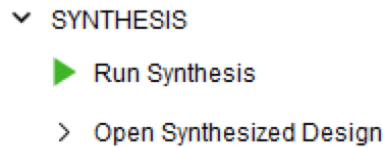
3) Click "Run Implementation" in Flow Navigator to generate a .bit file to implement the circuits in the FPGA board;
4) Click "Open Implemented Design" below "Run Implementation";
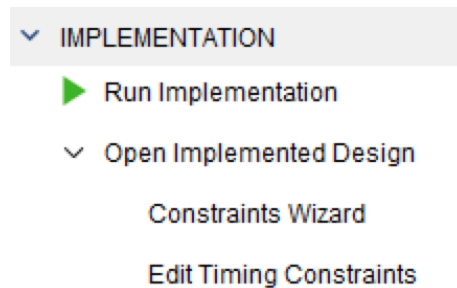
5) Choose "I/O Planning" from the drop down menu in the upper right corner of the software window;
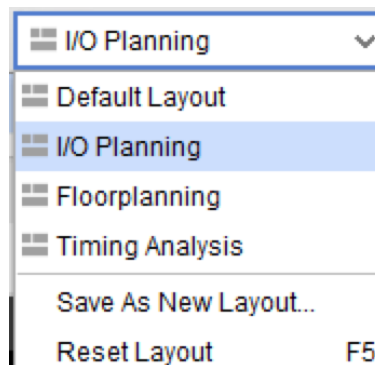
6) Select "I/O Ports" tab in the bottom window; the window can also be opened from the drop down menu: Window → I/O Ports;

What you are going to do in this tab is to map the desired I/O resources (switches, LEDs, etc.) to the inputs and outputs of your circuit. Some I/O resources available on the Basys 3 FPGA board as well as their labels are shown in the following figure. The labels may also be found on the board next to each I/O device.
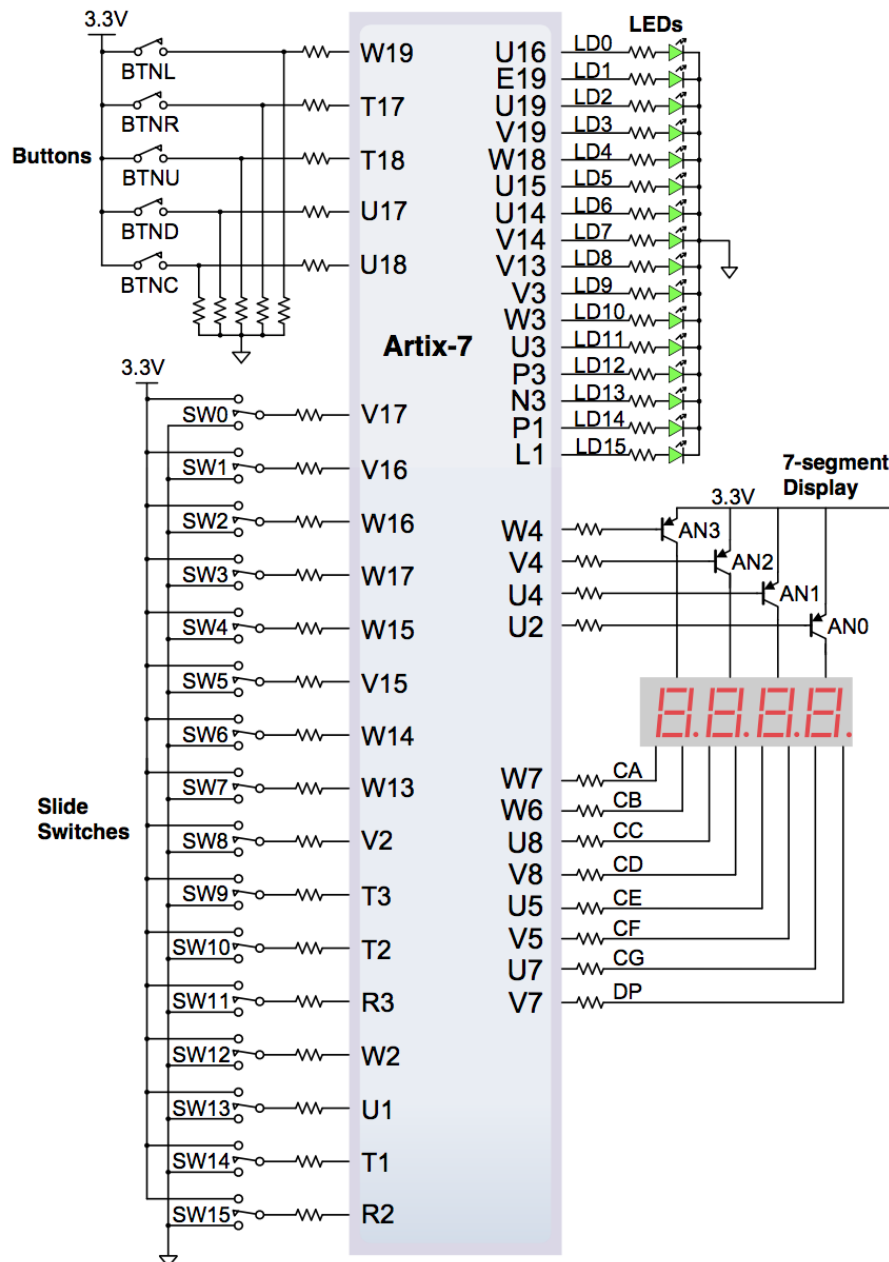
7) In the "I/O Ports" window, pay attention to these columns:
   - Name: inputs and outputs of your designed circuit;
   - Package Pin: labels of desired I/O resources, such as V17, W19, and U16. Change the labels to select the I/O resources that you want to use;
   - Fixed: all should be checked after I/O resources are mapped to the circuit;
   - I/O Std: change all to "LVCMOS33*"
8) Save the changes in a constraints file with extension .xdc;

Note: Basys 3 provides a 100 MHz clock at label W5. But we will use a push button to provide the clock signal to your circuits in this lab. However, the Vivado system will report an error if it finds the clock input of a circuit is not connected to W5. In order to override this default error, the

following should be done:

9) In the "Sources" window where you added the Verilog source files, find the .xdc constraints file you just created in step 8) under Constraints → constrs_1;
10) Double click the file to open it which should look similar to this:

```
 9 | set_property IOSTANDARD LVCMOS33 [get_ports clock]
10 | set_property IOSTANDARD LVCMOS33 [get_ports load]
11 | set_property IOSTANDARD LVCMOS33 [get_ports reset]
12 | set_property PACKAGE_PIN U19 [get_ports {Q[2]}]
13 | set_property PACKAGE_PIN E19 [get_ports {Q[1]}]
14 | set_property PACKAGE_PIN U16 [get_ports {Q[0]}]
```

11) Add this line to the .xdc file:
    set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clock_IBUF]

12) Rerun Synthesis and Implementation;
13) Connect a Basys 3 board to your computer, and turn it on;
14) Click "Generate Bitstream";
15) Click "Open Hardware Manager;

```
∨ PROGRAM AND DEBUG
    ▮▮ Generate Bitstream
    >  Open Hardware Manager
```
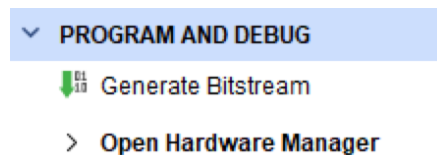
16) Click "Open Target";
17) Click "Program Device"

Now your designed circuit has been implemented in the FPGA chip on the Basys 3 board. Verify the functions of your circuit against the simulation results obtained in Section 4.

6. Deliverables

This is a 1-week lab. The full score for this lab is 200 points.

1) Demonstrate your circuits to the TAs before your lab session ends.
2) Upload source files on Canvas by **10pm, October 27, 2017**.
3) Upload peer evaluation report on Canvas by **10pm, October 27, 2017**.

Peer evaluation form:

Name	Percentage of contribution (sum to 100%)	Responsibilities
(yourself)		
(your lab partner)		