CS223 Laboratory Assignment 2

A 2-bit Adder

Lab dates and times:
Section 1: 24.10.2016 Monday 08:40-12:25
Section 2: 25.10.2016 Tuesday 08:40-12:25
Section 3: 24.10.2016 Monday 13:40-17:25
Section 4: 25.10.2016 Tuesday 13:40-17:25
Section 5: 20.10.2016 Thursday 08:40-12:25
Section 6: 21.10.2016 Friday 08:40-12:25

Location: EA Z04 (in the EA building, straight ahead past the elevators)
Groups: Each student will do the lab individually. Group size = 1

Preliminary Report (25points)

Today's lab needs considerable advance preparation. You need to learn how to work with Xilinx's design tool set before attending the lab. Besides, schematics and SystemVerilog models should be prepared in advance, and assembled neatly into a Preliminary Report with a printed cover page and printed pages for the schematics and SystemVerilog codes. Each page should have a proper heading. The content of the report is as follows:

a) A cover page which includes the following: course name and code number, the number of the lab, your name and student ID, date, number of your trainer pack.
b) Schematic (block diagram) of a 2-bit adder, made from 2 connected full-adders for part A-a. You need to draw the circuit using only 2-input XOR, AND and OR gates, those you saw in Lab1. All IC pin numbers have to be specified clearly on gate pins (refer again to Circuit_schematic_versus_Logic_Diagram.doc). Try to use minimum number of ICs to make your circuit.
c) Dataflow SystemVerilog module for a Full-adder for Part B-a. Prepare a testbench for it.
d) Structural SystemVerilog module for a 2-bit adder for part B-c. Prepare a testbench for it.

Additional pre-lab work:

You should study the following documents (available on Unilica) to be familiar with steps of design flow (Simulation, Synthesis, Implementation, Generation of Programming File, Downloading to FPGA board), using Xilinx Vivado tool. You can download, install and practice working with Xilinx Vivado on your own computer with free webpack license.

- Suggestions for Lab Success with SystemVerilog, Vivado, and BASYS3.
- Basys3 Vivado Decoder Tutorial.
- Vivado Tutorial.
- BASYS-3 FPGA Board Reference Manual (just take a look, and later use it as reference when needed).

You will need a copy of your designs and SystemVerilog programs with you in the lab to refer to or possibly correct and change it. The Preliminary Report will be turned in at the start of lab. Therefore, you must make a photocopy of it before you come to the lab, for your own use.

Part A: 2-bit adder on the breadboard (40 points)

In part A, you will build a 2-bit adder one the breadboard using gates. First, you will test your adder manually by switches. Later, you will make a Vivado project to generate test vectors automatically using FPGA and feed them to your circuit on the breadboard. For doing c) and d), you will need to connect your BASYS-3 board to the Beti board. If this is the first time you are doing this, you may ask your TA for help.

a) Build circuit on breadboard: Implement a 2-bit adder by connecting 2 full-adders. Connect Carry-in (Cin) of first adder to GND. Then connect carry-out (Cout) of first full-adder to Cin of second full-adder. Use only XOR, AND and OR gates to build your circuit on the breadboard. Obey the recomendations you had been given in Lab1, otherwise you may get confused with too many number of wires and complicated routings.

b) Test it: Using available switches and LEDs on Beti board, test your circuit by applying all possible combinations of inputs. You need to apply 16 test values: [A_1A_0, B_1B_0]=[00,00]…[A_1A_0, B_1B_0]=[11,11]. Output has 3 bits [Cout, C_1C_0]. Draw the truth table completely. When you are convinced that your circuit works correctly, show it to your TA. It is important to have input and output bits on switches and LEDs in a natural order (i.e. MSB bit on left, LSB on right). Then you and your TA can check it quickly. Be prepared to answer questions that you may be asked.

- *Create a new Xilinx Vivado Project to do c) and d). Use appropriate names for files and folders, keeping the project in a directory where you can find it later and erase it (at the end of lab).*

c) Make FPGA project: create a New Project named Tester, and make the following SystemVerilog program its HDL source:

```
module hdrive(
input clk,
output [3:0] out
);
logic [28:0] count;
always @(posedge clk)
   count=count+1;
assign out[3]=count[28];
assign out[2]=count[27];
assign out[1]=count[26];
assign out[0]=count[25];
endmodule
```

Now create the following constraint file.(hdrive.xdc):

```
# Clock signal
set_property PACKAGE_PIN W5 [get_ports clk]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

##Sch name = JC7
set_property PACKAGE_PIN L17 [get_ports {out[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[0]}]
##Sch name = JC8
set_property PACKAGE_PIN M19 [get_ports {out[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[1]}]
##Sch name = JC9
set_property PACKAGE_PIN P17 [get_ports {out[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[2]}]
##Sch name = JC10
set_property PACKAGE_PIN R18 [get_ports {out[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {out[3]}]
```

The above SystemVerilog module describes a 4-bit counter that slowly counts up from 0000, 0001, 0010, ... to 1111 and then repeats. The above constraint file routes these 4 signals to L17, M19, P17 and R18 pins of the FPGA (all are on JC connector on right of BASYS-3). Study the board manual, the PMOD connectors section, to find where these are. After you connect BASYS-3 on Beti board (by an adapter board), these four pins goes to PORT-D of Beti board. You will use these 4 signals from the BASYS-3 board as the inputs A_1A_0 and B_1B_0 to your 2-bit counter that you have built on the breadboard. Disconnet 2-bit adder inputs from Beti switches and connect them to BASYS-3 now. Don't disconnect LEDs.

d) Program the FPGA: Now do the Xilinx design flow, in order to implement the 25-bit fast counter and the 4-bit slow-count outputs on the BASYS-3 board, including generating the bit file and programming the FPGA using USB cable (When FPGA programming is finished successfully, 'programming done LED', at the corner of the BASYS-3, turns on). When the FPGA outputs are counting the 4 inputs of the 2-bit adder, observe the outputs of the adder (on Beti's LEDs). What do you see?

Part B: 2-bit adder on the FPGA (35 points)

In Part B, you will implement 2-bit adder again, but this time on FPGA. For this part, you don't need to connect your BASYS-3 board to the Beti board. Working with standalone BASYS-3 and having it connected to your computer is enough. There are some switches and LEDs available on BASYS-3.

- *Create a new Xilinx Vivado Project to do a), b), c) and d). Use appropriate names for files and folders, keeping the project in a directory where you can find it later and erase it (at the end of lab).*

a) Write HDL code: Give the SystemVerilog code which models a single full-adder, in the dataflow style. (This means modeling with Boolean equations, using continuous assignment statements.) Name your SystemVerilog module *FullAdder*.
b) Simulate the module: Using SystemVerilog testbench code, verify in simulation that your full-adder is working correctly. Note that the testing is complete, using all possible input combinations.
c) Write HDL code: Give the structural SystemVerilog code which makes a 2-bit adder by using two full-adders.
d) Simulate the module: Now simulate 2-bit adder by checking all possible input combinations. When you are convinced that it works correctly, show the simulation results of single full-adder and 2-bit adder to your TA. Be prepared to answer questions that you may be asked.
e) Program the FPGA: Now, follow the Xilinx design flow to synthesize, create programming file, and program your 2-bit adder to BASYS-3 FPGA board.
f) Test your design: Using the switches and LEDs (on BASYS-3, not Beti board) that you have assigned in constraint file, test your 2-bit adder. When you are convinced that it works correctly, show the physical implementation results to the TA. Be prepared to answer questions that you may be asked. Assign input and output numbers to switches and LEDs in natural order: MSB on left,...,LSB on right.

Submit your code for MOSS similarity testing

Finally, when you are done and before leaving the lab, you need to upload the file *StudentID_SVerilog.txt* created in the Implementation with FPGA part. Be sure that the file contains exactly and only the codes which are specifically detailed above. Don't include the codes which are given to you. If you have multiple files, just copy and paste them in order, one after another inside text file. Check the specifications! Even if you didn't finish, or didn't get the SystemVerilog part working, you must submit your code to the Unilica Assignment for similarity checking. Your codes will be compared against all the other codes in all sections of the class, by the MOSS program, to determine how similar it is (as an indication of plagiarism). So be sure that the code you submit is code that you actually wrote yourself! All students must upload their code to the 'Unilica>Assignment' specific for your section. Check submission time and don't miss it before leaving the lab. After taking a backup of your work, don't forget to delete it from computer. Because students of other sections will work with your system too.

Clean Up!

1) Clean up your lab station, and return all the parts, wires, the Beti trainer board, etc. Leave your lab workstation.

2) CONGRATULATIONS! You are finished with Lab #2 and are one step closer to becoming a computer engineer.

NOTES

--Advance work on this lab, and all labs, is strongly suggested.

--Be sure to read and follow the Policies for CS223 labs, posted in Unilica.

LAB POLICIES

1. There are three computers in each row in the lab. Don't use middle computers, unless you are allowed by lab supervisor.
2. You borrow a Lab-board containing the development board, connectors, etc. in the beginning. The lab supervisor takes your signature. When you are done, return it to her, otherwise you will be responsible and lose points.
3. Each Lab-board has a number. You must always use the same trainer board pack throughout the semester.
4. You must be in the lab, working on the lab, from the time lab starts until you finish and leave. (Bathroom and snack breaks are the exception to this rule). Absence from the lab, at any time, is counted as absence from the whole lab that day.
5. No cell phone usage during lab. Tell friends not to call during the lab hours--you are busy learning how digital circuits work !.
6. Internet usage is permitted only to lab-related technical sites. No Facebook, Twitter, email, news, video games, etc--you are busy learning how digital circuits work !.
7. If you come to lab later than 20 minutes, you will loose that session completely.
8. When you are done, DO NOT return IC parts into the IC boxes, where you've taken them first. Just put them inside your lab pack box. Lab coordinator will check and return them later.