ECE 5727

Lab 1: RTL Basics

Due Wednesday, March 3
Total Points: 56

Objectives

In this lab you will:

1. Get Xilinx Vivado installed and learn how to create an FPGA project.
2. Synthesize Verilog HDL and explore RTL schematics

Instructions

Install Vivado

You will find instructions for installing Vivado on Canvas: Xilinx -> Vivado Installation Instructions.pdf. It will walk you through installation of the Vivado Design Suite, as well as setting up a Virtual Machine with Linux if you are a Mac user, and installation of the Xilinx Cable Drivers so that the computer will be able to communicate with the board.

Lab Walk-through

Once the installation of Vivado is complete, go ahead and launch it. You will also need to download the lab project files either from Canvas (under Files -> Labs -> Lab1) or from the course GitHub (https://github.com/daberman/CT-ECE5727-SP21)

1. I have already created the Vivado project, so all you need to do is select *Open Project* under Quick Start. Navigate to where you have the Lab 1 files are and select Lab1.xpr
2. Once the project finishes loading, you'll need to open the Lab1.sv source file. You should see it under **Design Sources**. Double-click to open it.
3. You'll see there are three different always blocks. Two are commented out, while the one labeled reset_1 should not be. Take a look at the code in that block, what kind of RTL do you expect it to generate? See if your thoughts were correct by opening the RTL schematic. You do this by going to the Flow Navigator pane on the left hand side of the GUI, then selecting **RTL ANALYSIS -> Open Elaborated Design**. It will take a few seconds to load, and then you should see the schematic appear. Take a screenshot for future reference. *You can now answer 1a in the Lab Questions*.
4. Comment out reset_1 and uncomment reset_2. Saving the change will result in Vivado informing you the elaborated design is out of date. Click the option to refresh. Take a look at the new schematic, and take a screenshot for reference. Repeat for reset_3. *You can now answer 1b-c in the Lab Questions*.

5. Complete Lab Question 2a-c

6. There are two dout assignment statements within the reset_3 block at lines 36 and 37. Uncomment line 37 so both statements are now uncommented. Take a look at the RTL schematic. *You can now answer 2d in the Lab Questions*.

7. We will know run Vivado synthesis to convert the elaborated RTL into FPGA components. From the Flow Navigator pane select **SYNTHESIS -> Run Synthesis**. This will take a minute or two. When it is finished, a popup will appear asking about next steps; select to open the synthesized design. After it loads, select **SYNTHESIS -> Open Synthesized Design -> Schematic**.

8. Select one of the LUTs being used as a 4-to-1 mux (has inputs from din and counter). Expand the Cell Properties pane and then select the Truth Table tab. *You can now answer 2e-f in the Lab Questions*.

9. You have finished the lab. Complete Lab Question 3 and submit your (typed) responses to Canvas.

Lab Questions

Complete these questions as you work through the lab.

1. (a) (*5 points*) Why is the rst_n signal connected to the CE input for dout?
   (b) (*5 points*) Other than the change in name for reset, what changed in the code from reset_1 to reset_2? How was this reflected in the RTL schematic?
   (c) (*5 points*) Why does reset_3 not have a CE for dout but still has a reset for counter?

2. (a) (*3 points*) Recall from Lecture 2 how a 4-input LUT can be used to create a 2-to-1 multiplexer by using one input for the select signal, two inputs for the inputs to the mux, and one input unused. On the FPGA we will be using we have 6-input LUTs. Draw and/or describe how you could use a 6-input LUT to make a 4-to-1 multiplexer.
   (b) (*3 points*) What will the truth table be for the LUT? Remember, you can save rows by putting X's in for inputs we don't care about
   (c) (*3 points*) Draw and/or describe how you could use 6-input LUTs to make a 16-to-1 mux. (FYI this was a question I was asked at my last interview). Remember, you can make a 4-to-1 mux from a single LUT, and lecture 2 has an example of making a 4-to-1 mux from 2-to-1 muxes.
   (d) (*3 points*) Does the RTL schematic depict what you'd expect from the code?
   (e) (*5 points*) For the synthesized design: were the 6-input LUTs used how you expected based on 2c? What are they doing differently? What's the same?

(f) (*5 points*) Does the truth table look roughly the same as yours? (FYI, for me I2 and I4 were the select signals so things might be ordered differently) What is the logic equation used? Does it make sense?

3. Consider a system with a 3-bit binary input (A,B,C) and two outputs (X,Y). X is true when the input is a prime number. Y is true when the input is *not* divisible by 3 (treat 0 as indivisible).

   (a) (*3 points*) I have done the first few rows of the truth table for you, where n is the decimal value for the binary value represented by ABC. Complete it.

n	A	B	C	X	Y
0	0	0	0	0	1
1	0	0	1	0	1
2	0	1	0	1	1

   (b) (*3 points*) Going row by row, you can write out the logic equation as a sum of products for both outputs. I have written out the one for X below. Derive the one for Y.

$$X = A'BC' + A'BC + AB'C + ABC$$

   (c) (*5 points*) Use boolean algebra to reduce both equations to a minimal number of terms. Show your steps.

   (d) You can also derive the logic equations using the 0s from the table.

      i. (*3 points*) I have done X'. Derive Y'.

$$X' = A'B'C' + A'B'C + AB'C' + ABC'$$

      ii. (*5 points*) Reduce, showing your steps. Remember, you will need to use DeMorgan at the end to get from X' and Y' to X and Y.