SANTA CLARA UNIVERSITY	ELEN 21 Spring 2017	Dr. Sally Wood, Dr. Samiha Mourad, Dr. Radhika Grover

Laboratory #5: 4-bit Ripple-Carry Adder

For lab sections May 9-12, 2017

I. OBJECTIVES

- Learn to do hierarchical design and create symbols for circuits to be used as building blocks.
- Learn how to use busses for multi-bit inputs such as numbers and ASCII characters
- Design a 4-bit ripple carry adder using half adders and full adders as building blocks.
- Use 7-segment displays to show inputs and outputs of the design, and add a Verilog module to generate the 7-segment display.

PROBLEM STATEMENT

A 2-level circuit that would add two four-bit numbers would have 9 inputs and five outputs, and the design would be complex in terms of the number of logic gates and the number of inputs for each gate. This would increase the complexity of testing. Instead, hierarchical design methods can be used. Simple building blocks are designed and tested, and then these blocks are used much in the same way as simple logic gates in the design of a larger circuit.

A half adder adds two one-bit numbers to create a sum bit and a carry-out bit. It has two inputs, INA and INB, and two outputs, SUM and CARRY.

A full adder adds two one-bit numbers but also has a carry-in bit. The three-bit addition of the two inputs and the carry produces a sum output bit and a carry output bit. The three inputs are A and B and CIN, and the outputs SOUT and COUT

Full adders can be connected in sequence with the carry out of one adder connected to the carry in of the next to add multi-bit input values. An *n*-bit adder would use *n* full adders to create an *n*-bit sum and a carry out.

II. PRE-LAB

1. Draw the logic gate schematic of a half adder in Quartus.
   (i) Clearly show the inputs of the half adder INA and INB and the outputs SUM and CARRY.
   (ii) Show all connections and internal connections. Label all inputs and outputs.
   (iii) Create a symbol for this half adder as you did in previous labs.
2. Draw the schematic of the full adder that uses the half adder from (a). You are doing a hierarchical design and so you will use instances of the symbol you created for the half adder as well as other logic symbols in your full adder schematic. Clearly show the inputs of the full adder A and B and CIN and the outputs SOUT and COUT. Show all connections and internal connections. Label all inputs and outputs.
   (i) Create a symbol for the full adder.

III. LABORATORY PROCEDURE

1. Creating a Half-Adder:

   (i) Review the schematic of your half-adder from your prelab.
   (ii) Simulate the half adder and confirm that it functions correctly.

2. Creating a Full-Adder:

   (i) Review the schematic of your full adder from your prelab.
   (ii) Make it the top level entity by following the instructions below
   From the Project navigator, select: Files
   Right click the required file
   Select " Set as Top-Level Entity " from the drop down menu.
   This must be done in order to compile, simulate and program the correct schematic.
   (iii) Simulate the full adder and confirm that it functions correctly.

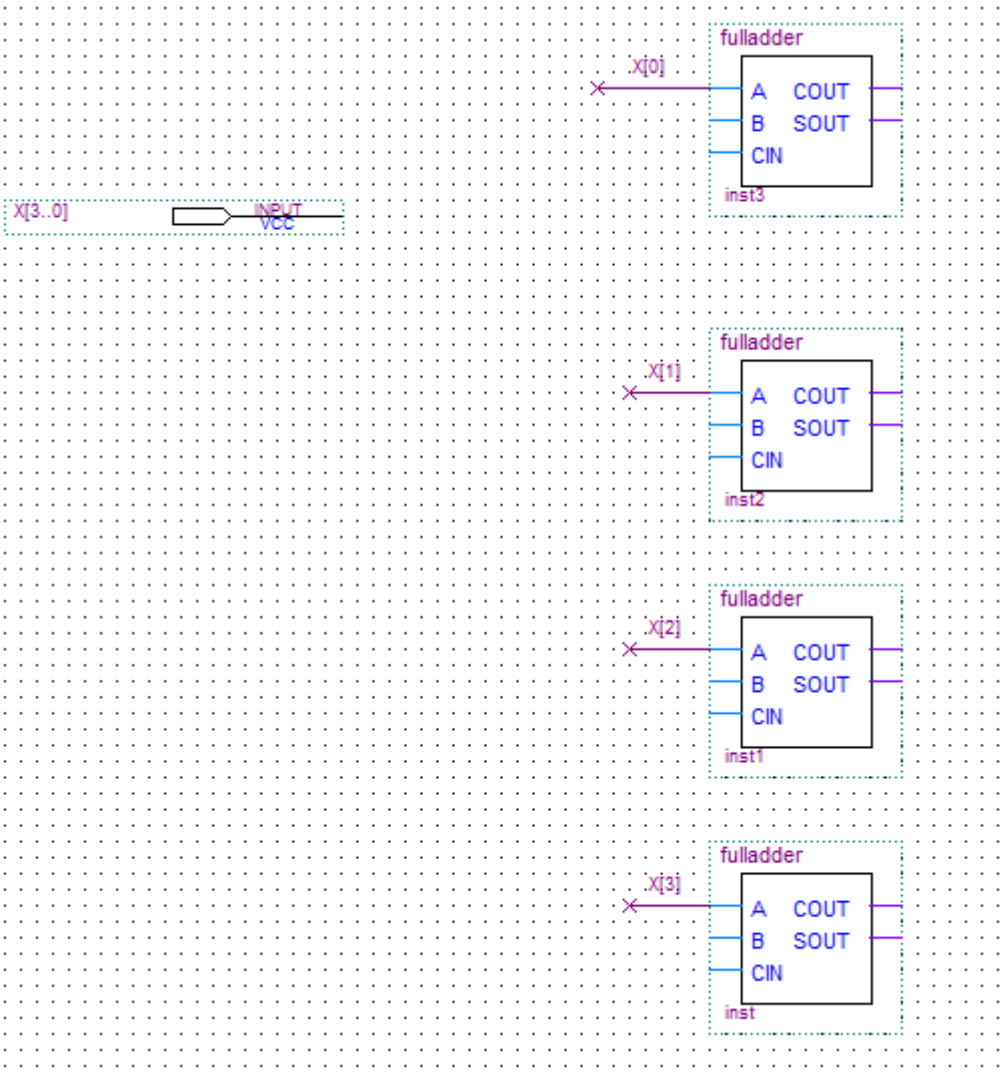
3. Creating a 4-bit Ripple Carry Adder:

   (i) Draw the schematic of your ripple carry adder. You will be using the symbol that you have created of your full-adder in the schematic. Do this new design in the same project.
   (ii) The inputs of the module are 4-bit numbers X and Y and the output is a 4 bit number S. A very useful way to specify connections in the schematic without using a lot of wires is naming the wires as vectors. For example, the output S of the adder can be called S[3..0] (4-bit bus of data).

   You will create buses for the input X, input Y and the S output. Make sure you name the ports **X[3..0] , Y[3..0] and S[3..0].**

   The buses will split to X[3], X[2], X[1], X[0] and Y[3],Y[2],Y[1],Y[0] and S[3],S[2], S[1], S[0] when named correctly. The figure below demonstrates the proper method to name the wires in order to split the bus X [3..0] for example.
   Note that this is not a complete schematic.

   The CarryIN and CarryOUT will be single wires.
   Show all connections and internal connections. Label all inputs and outputs.



(iii) Simulate the ripple carry adder and confirm that it functions correctly.
Remember to make it your top level entity before simulating.

(iv) Once you have designed and simulated your ripple carry adder, you will make a symbol for it.

4. <u>Assigning Inputs and Outputs:</u>

(i) Create a test circuit for your Ripple carry adder using its symbol.

(ii) When assigning pins for the ripple-carry adder, use switches for X(0) to X(3) and Y(0) to Y(3). Also use a switch for CarryIN, which is the very first carry in. Connect the sum output S(0) to S(3) and CarryOUT to LEDs. Make sure you assign pins in a way that you can see the MSB on the left and the LSB on the right.

(iii) You will also connect the inputs and outputs to the 7-segment display.
Follow the seven-segment tutorial in the "Seven_Segment" file and generate the numbers on the FPGA's 7-segment displays.
Use the txt file "bin_7seg_temp.txt " for the 7 segment display.

(iv) The input of the module is a 4-bit binary number. Make sure you name the port as a bus **"…[3..0]"**. The input of the module will be connected to a 4-bit number like X, Y and S. You will use the **Orthogonal Bus Tool** for wiring.. It appears at the top of your page as shown in the following figure.

(iv) The output of the module is a 7-bit value with one bit to control each segment of the seven-segment display. Make sure you name the port as a bus **"…[6..0]"**. The output of the module will be connected to a 7-bit value (remember to use the **Orthogonal Bus Tool** for wiring) which goes directly to the pins of the board that correspond to each line of the 7-Segment Display.

The Altera DE2-115 board has eight 7-Segment Displays as described in the tutorial. Choose one for X, one for Y and two for S for the display of the number defined by Cout and S. The rightmost would display the four-bit value S, and the one on the left would display "0" or "1" depending on the value of Cout. The four bit input to this display would be (0 0 0 Cout). Cout should also be connected to an LED.

5. Programming the FPGA:

Download and demo your design.
Take a picture of the working FPGA with 7-segment display.

IV. REPORT

For your lab report, include the schematics and simulation waveforms of all the components you designed. In addition, include answers to the following questions.

- What would you expect to happen if you were to add two numbers and the result was greater than 15? What will be displayed in each bit of the Sum, and what will happen to the Cout LED? How would you interpret these results?
- How many logic gates are on the path from X0, Y0, and Cin to the output Cout for the 4-bit ripple carry adder?
- If you had to make an 8-bit adder, how would you use the 4 bit module you have built in this lab to make a 8 bit adder module?
- How many logic gates are on the path from X0, Y0, and Cin to the output Cout for the 8-bit ripple carry adder?