ECE327 Digital System Design
Fall 2017
Lab 1: BCD conversion and Adder

Lab Overview: The purpose of this exercise is to design combinatorial circuits that can perform binary-to-decimal number conversion (Part I). In Part II of this lab, you will create a ripple carry adder. Finally in Part III of this lab you will use the converters from Part I and the adder from part II in a hierarchical design to form a binary-coded-decimal (BCD) adder. The requirements for this lab consist of completing the QUARTUS II designs, submitting VHDL source files to the assign server, and completing the laboratory report using the LaTeX template provided. Please note that sharelatex.com is an easy to use, online, LaTeX editor if you are new to LaTeX.

Submission Instructions: Please follow all instructions given in the "Lab Report Instructions" and "Assign Server Instructions" on Canvas. See Announcements for due dates and times.

Specifications Part I: You are to design a circuit that converts a four-bit binary number $V = v_3v_2v_1v_0$ into its two-digit decimal equivalent $D = d_1d_0$. Table 1 shows the required output values that should be displayed on two 7-segment displays. Your VHDL entity should have the four-bit input V (the binary number), the output M (the BCD representation). M should consist of 4-bits representing the least significant BCD digit and another bit representing the upper BCD digit, since it can only take the values 0 and 1.

Binary Value (V)	Decimal Digits (D)		Hardware representation of BCD (M)	
0000	0	0	0	0000
0001	0	1	0	0001
0010	0	2	0	0010
…	…	…	…	…
1001	0	9	0	1001
1010	1	0	1	0000
1011	1	1	1	0001
…	…	…	…	…
1111	1	5	1	0101

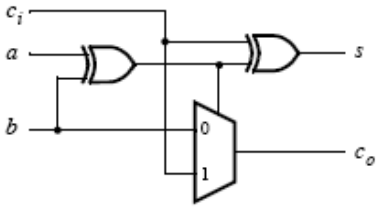
Use switches SW3-0 on the DE1-SoC board to represent the binary number V, and the displays HEX1 and HEX0 to show the values of decimal digits d1 and d0 from Table 1. Simulate your circuit by trying all possible values of V and observing the output displays. Finally test your circuit on the DE1-SoC board. The DE1-SoC user manual has information regarding how to use the 7-segment displays.

Specifications Part II: The figure below shows a circuit for a *full adder*, which has the inputs a, b, and c_i, and produces the outputs s and c_o. Parts b and c of the figure show a circuit symbol and truth table for the full adder, which produces the two-bit binary sum $c_os = a + b + c_i$. Figure 2d shows how four instances of this full adder entity can be used to design a circuit that adds two four-bit numbers. This type of circuit is usually called a *ripple-carry*

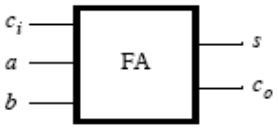
adder (we will discuss this in more detail later in the course), because of the way that the carry signals are passed from one full adder to the next.

Write VHDL code that implements a full adder and write a top-level VHDL entity that instantiates four instances of this full adder. Use switches SW_{7-4} and SW_{3-0} to represent the inputs A and B, respectively. Use SW_8 for the carry-in c_{in} of the adder. Connect the SW switches to their corresponding LED lights, and connect the outputs of the adder, c_{out} and S, to the corresponding LED lights.

Use functional simulations to verify the correct operation of your adder. Test your circuit by trying different values for numbers A, B, and c_{in} both in simulation and on the DE1-SoC board. As we have discussed in class, make sure you use both appropriate and enough test cases to prove that your design works. **In your report, discuss the test cases that you have chosen and justify that they are sufficient.**



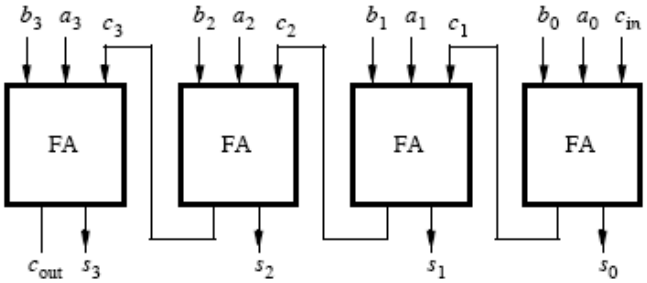
a) Full adder circuit



b) Full adder symbol

b	a	c_i	c_o	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

c) Full adder truth table



d) Four-bit ripple-carry adder circuit

Specifications Part III: In Part I you created a BCD converter. It is often useful to build circuits that use this method of representing decimal numbers, in which each decimal digit is represented using four bits. As an example, the decimal value 59 is encoded in BCD form as 0101 1001.

You are to design a circuit that adds two BCD digits. The inputs to the circuit are BCD numbers A and B, plus a carry-in, c_{in}. The output should be a two-digit BCD sum S_1S_0. Note that the largest sum that must be handled by this circuit is $S_1S_0 = 9 + 9 + 1 = 19$. If the sum exceeds this value, you should indicate this by lighting one of the LEDs.

In your design, you should use the four-bit adder circuit from Part II to produce a four-bit sum and carry-out for the operation $A + B$. A circuit that converts this five-bit result, which has the maximum value 19, into two BCD digits S_1S_0 can be designed in a very similar way as the binary-to-decimal converter from Lab I.

Use switches SW_{7-4} and SW_{3-0} for the inputs A and B, respectively, and use SW_8 for the carry-in. Connect the SW switches to their corresponding LED lights, and connect the four-bit sum and carryout produced by the operation $A + B$ to the LED lights. Display the BCD values of A, B, and result S_1S_0 on the 7-segment displays. Since your circuit handles only BCD digits, make sure to check for the cases when the input A or B is greater than nine. If this occurs, indicate an error by turning on a LED light.

Use functional simulation to verify the correct operation. Test your circuit by trying different values for numbers A, B, and c_{in}.