EE 231 Lab 5

Adder/Subtractor

In this lab you will learn how to write several modules and instantiate them. To demonstrate this process you will design a 4-bit full adder/subtractor.

1. Prelab

1. Write the truth table for a full adder.
2. Write the truth table for a full subtrator.
3. Show how you can use half adders to build a full adder.
4. Figure 1 shows how to implement a ripple adder using a sequence of 1-bit full adders. Using an example, verify that this circuit functions as a 4-bit adder.

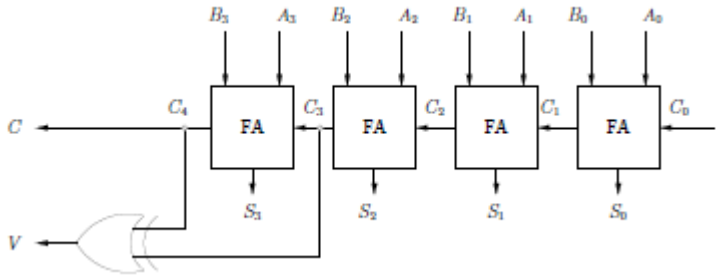

Figure 1. 4-bit Adder

5. What does the V signal, which may be computed as V = A3B3S'3 +A'3B'3S3, represent? In order to answer this question try to use examples when you are adding two positive numbers and another when you are adding two negative numbers.
6. By slightly modifying the circuit shown in Figure 1 we can design an adder/subtractor as shown in Figure 2. Why does this circuit makes and adder when the sel is 0, and why does it behave as a subtractor when the sel is 1.
7. Fill in Table 1

sel	Input B	Output D in terms of B
0	$B_3B_2B_1B_0$	
1	$B_3B_2B_1B_0$	

Table 1. Outputs of an adder/subtractor

8. Using Table 1, write a Verilog program to implement a decoder that selects the proper input to the full adder depending on the sel signal.

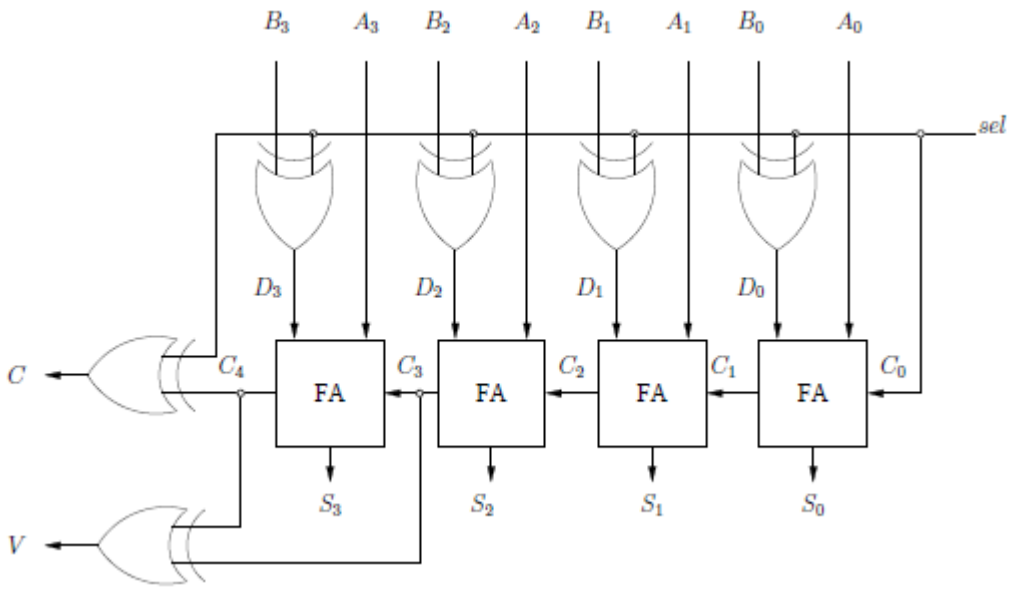

Figure 2. 4-bit Adder/Subtractor.

2. Lab

1. Design a 4-bit adder using the half adders.
2. Using a decode add a sel and the needed components to be able to select whether your design acts as a 4-bit adder or a 4-bit subtractor.
3. Simulate your circuit and verify its operation. You will need to try enough combinations to verify that the results of the addition and subtraction along with C and V are correct.
4. Add the BCD decoder you designed in last week's lab to output the add/subtractor so you can display the output on your 7-segment display.
5. Connect your sel and your inputs to hardware switches, and connect the output to the 7-segment display.

3. Supplementary Material

3.1 Verilog

In the previous lab you have defined a half adder by instantiating primitive gates. As the design becomes more and more complex, it is convenient to create separate modules and then combine them in one file. For example, in the previous lab you have created a half adder module and in this lab you will create a full adder/subtractor which can be designed

by using the half adder module. So, in this lab you will instantiate two half adders to form the full adder, then instantiate four full adders to create the 4-bit adder/subtractor. Program 1 illustrates this concept.

Program 1. An Example of a Module Instantiating Other Modules.

```
module sample_moduleA(
        output C,
        input A,B
        );

        wire w1;

        xor (w1,A,B);
        and (C,A,w1);
endmodule

module sample_moduleB(
        output D,
        input E,F
        );

        wire G,H;

        sample_moduleA SA1(G,E,F);
        sample_moduleA SA2(D,G,E);
endmodule
```

In this sample program the module sample_moduleA is instantiated twice as SA1, and SA2 in the higher level module sample_moduleB.