Lab 3: Adder/Subtractor

Introduction

As the designs becomes more complex, it is convenient to create separate modules and then combine them. In this lab one will learn how to write modules and instantiate them. To demonstrate this process a 4-bit full adder/subtractor will be designed.

1 Prelab

1.1. Write the truth table for a full adder.

1.2. Write the truth table for a full subtractor.

1.3. Show how you can use half adders to build a full adder.

1.4. 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.

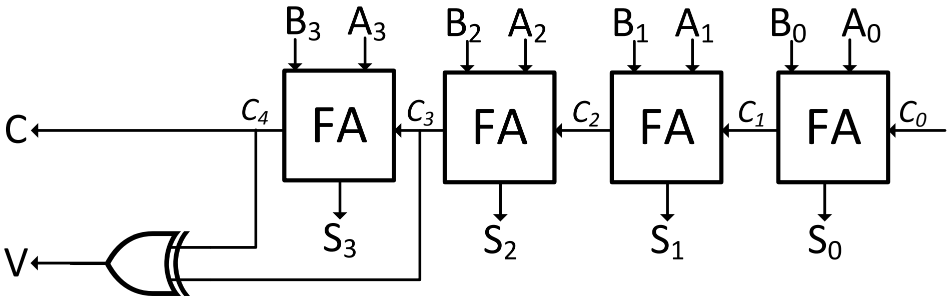

Figure 1: Ripple Carry Adder Circuit

1.5. **What does V, which is computed as $V = C_3 \oplus C_4$, represent?** Consider when adding two positive numbers and when are adding two negative numbers.

1.6. By slightly modifying the circuit shown in Figure 1 we can design an adder/subtractor as shown in Figure 2. **Why does this circuit make an adder when the $Sel == 0$, and why does it behave as a subtractor when the $Sel == 1$?**

1.7. Fill in Table 1.

Table 1: Outputs of an adder/subtractor

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

1.8. 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.

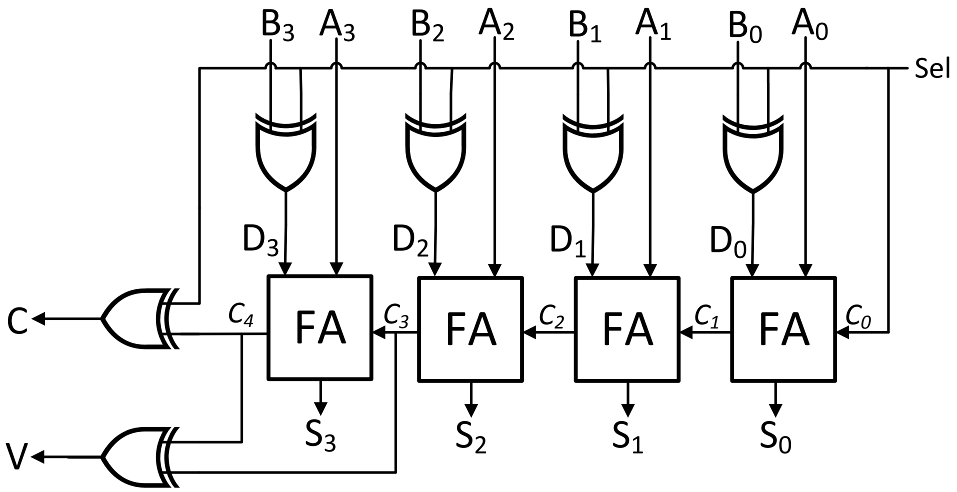

Figure 2: Ripple Carry Adder/Subtractor Circuit

2 Lab

In the following sample program (Listing 1), the module *sample_moduleA* is instantiated twice as *SA1*, and *SA2* in the higher level module *sample_moduleB*.

Listing 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
```

2.1. Design a 4-bit adder using the half adders.

2.2. Add the needed components to be able to select whether your design acts as a 4-bit adder or a 4-bit subtractor based on an input `sel` signal.

2.3. 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.

2.4. Add the BCD decoder you designed in last week's lab to output the adder/subtractor so you can display the output on your 7-segment display.

2.5. Connect your `sel` and your inputs to hardware switches (e.g. DIP Switches), and connect the output to the 7-segment display.