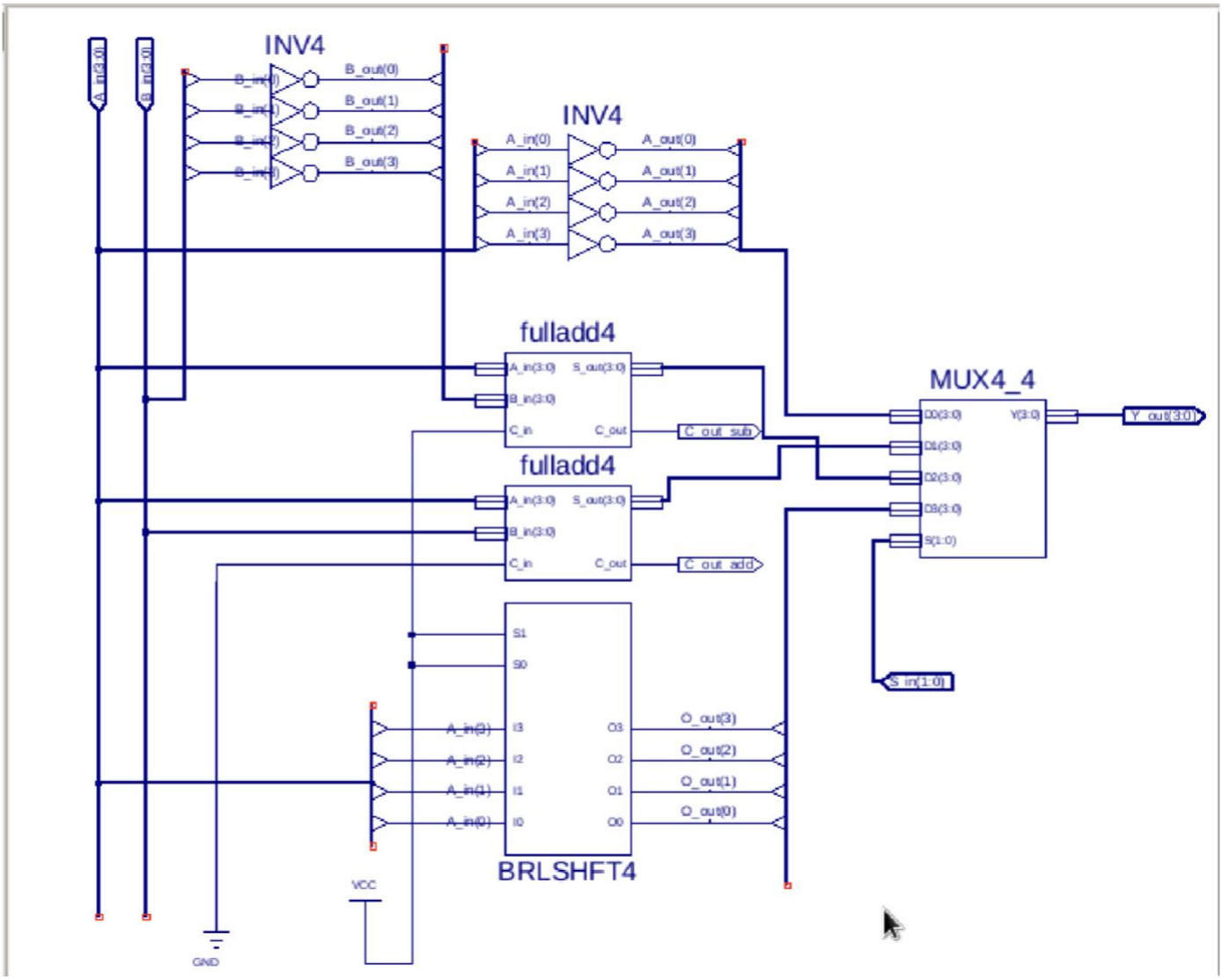
CDA 4203L
Computer System Design Lab

Lab 1 Report ALU Design

Today's Date:	Feb 7, 2025
Your Name:	Claude Watson
Your U Number:	U72087839
No. of Hours Spent:	10
Exercise Difficulty: (Easy, Average, Hard)	Average
Any Other Feedback:	N/A

Question 1: ALU Schematic



Question 1: Briefly describe how your design works.

The ALU is designed to perform four operations: bitwise inversion, addition, subtraction, and doubling. It has two 4-bit input buses (A and B) and a 4-bit output bus (Y), with a 2-bit selector (S1, S0) determining the operation. A custom 4-bit 4-to-1 multiplexer is used to select the final output.

First, let's look at the inverter function. Four inverters were placed vertically, each accepting one bit of the input from the A bus. The inverted result is then grouped into a 4-bit output bus and sent to the first multiplexer input (D0).

Next is the addition function. A 4-bit full adder was implemented and integrated into the design. This full adder takes 4-bit inputs from buses A and B, computes their sum, and stores the result in a 4-bit bus. The sum is then connected to the second multiplexer input (D1). The carry-out (C_out_add) is discarded since only 4 bits are used.

For the subtraction function, another 4-bit full adder is used, but with a key modification: the B input is inverted (~B)and the carry-in (C_in) is fixed to 1. This effectively computes A + (-B + 1) using two's complement arithmetic. Just like the addition function, the result is placed on a 4bit bus and connected to the third multiplexer input (D2). The carry-out (C_out_sub) is discarded.

Finally, for the doubling function, a 4-bit barrel shifter (BRLSHFT4) from the Xilinx library is used to perform left shifting. Setting S1 = 0, S0 = 0 tells the shifter to shift left by one bit, which is equivalent to multiplying A by 2. The inputs on the shifter are connected to the four bits of the A bus, and the outputs are combined into a single 4-bit bus. The result is then connected to the last multiplexer input (D3).

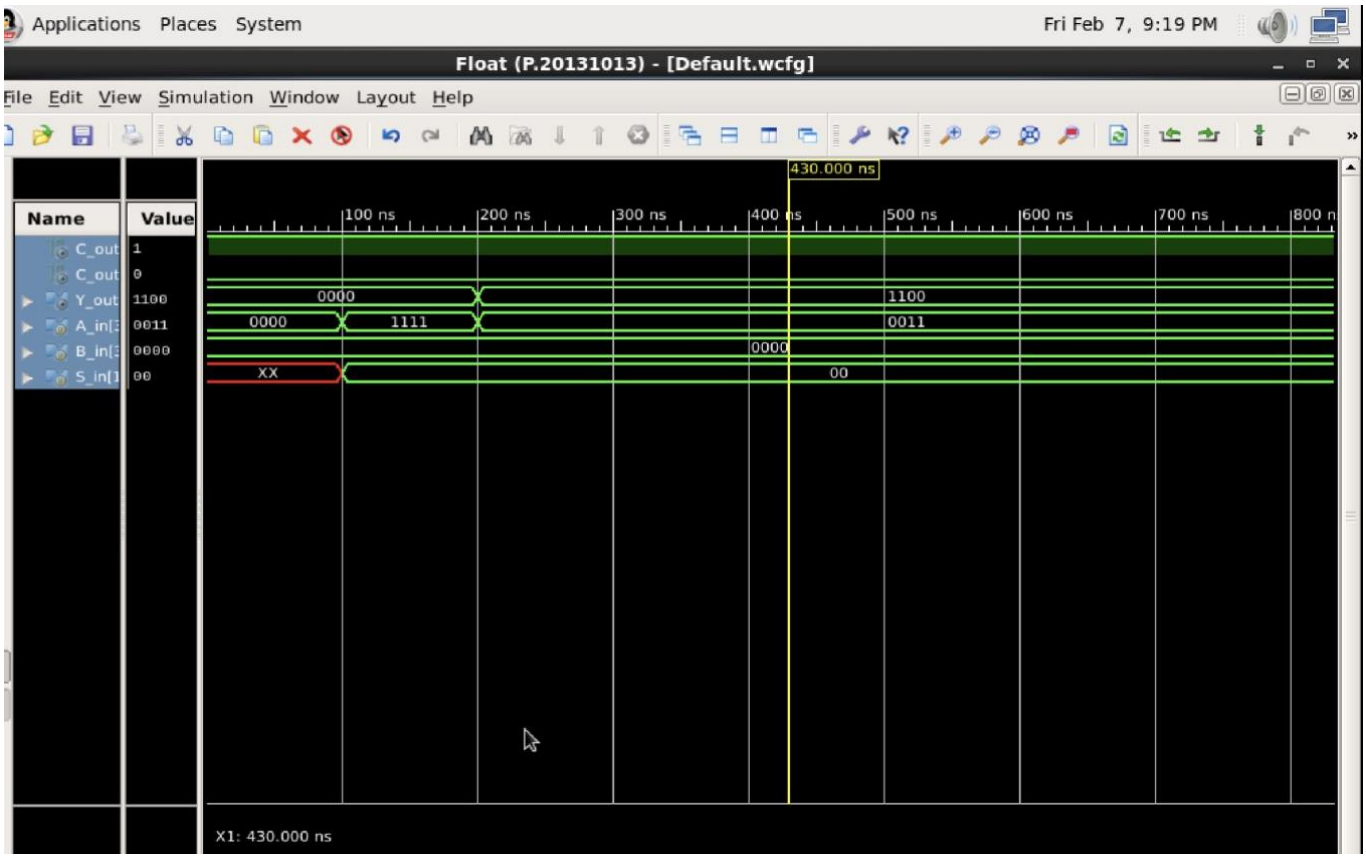
The multiplexer (MUX4_4) was custom-made to accept four 4-bit inputs, two 1-bit selectors

(S1, S0), and output a single 4-bit result (Y). This design allows each function's output to be directly connected to the multiplexer, ensuring that the correct operation is selected based on the values of S1 and S0.

Question 2: Simulation Waveforms (add as many pages as you need).

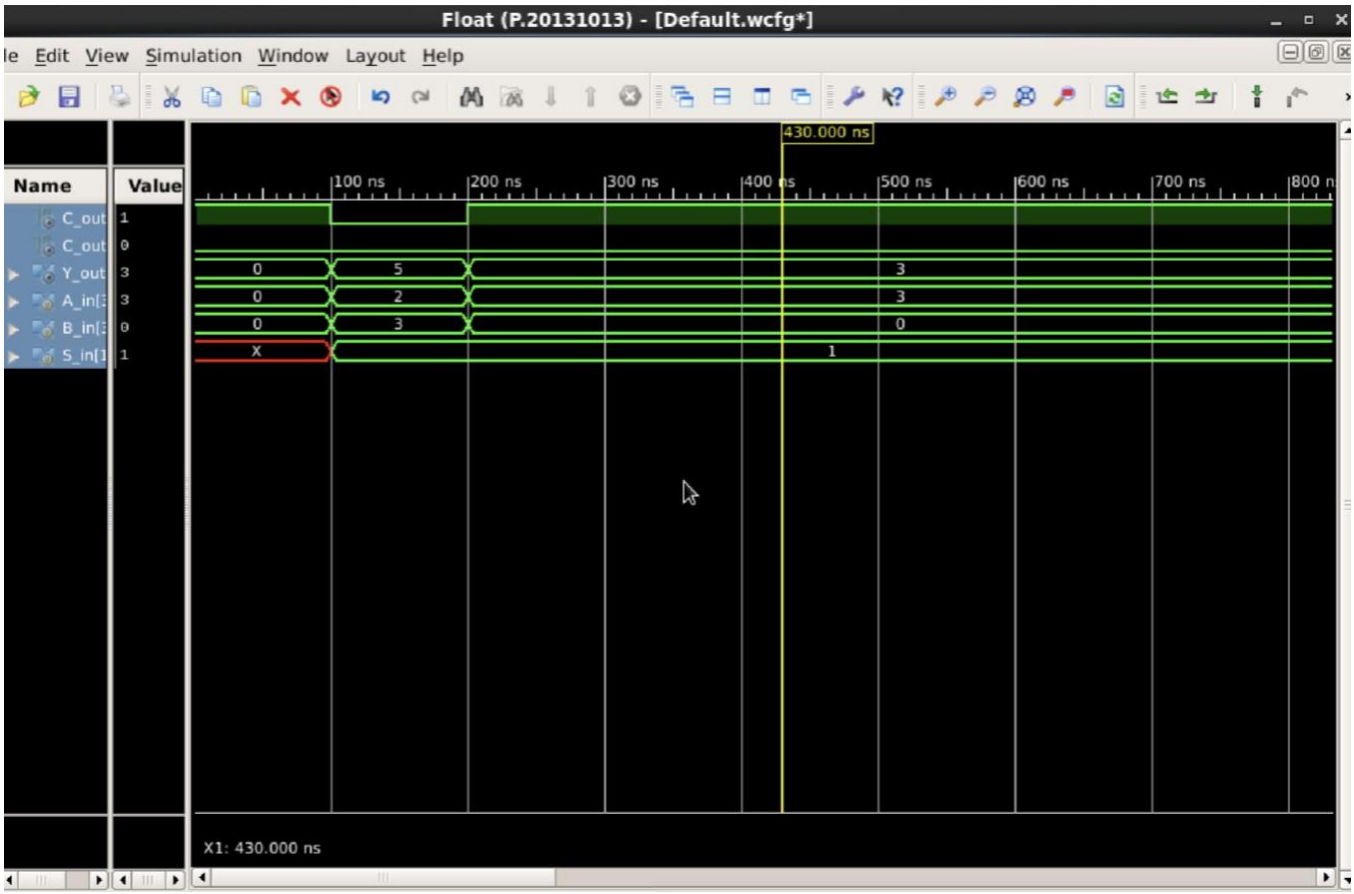
Inverter Function:

```
26        .Y_out(Y_out)
27    );
28  // Initialize Inputs
29        initial begin
30        // initalize inputs to 0
31        A_in = 4'b0000;
32        B_in = 4'b0000;
33        #100;
34
35        // invert A
36        A_in = 4'b1111; B_in = 4'b0000; S_in = 00; #100;
37        A_in = 4'b0011; B_in = 4'b0000; S_in = 00;#100;
38
39        //add A+B
40        //A_in = 4'b0000; B_in = 4'b0111; S_in = 01; #100;
41        //A_in = 4'b0011; B_in = 4'b0000; S_in = 01;#100;
42
43        // sub A-B
44        //A_in = 4'b0100; B_in = 4'b0001; S_in = 10; #100;
45        //A_in = 4'b0110; B_in = 4'b001; S_in = 10;#100;
46
47        // double A
48        //A_in = 4'b0001; B_in = 4'b0000; S_in = 11; #100;
49        //A_in = 4'b0011; B_in = 4'b0000; S_in = 11;#100;
50    end
51  endmodule
52
```



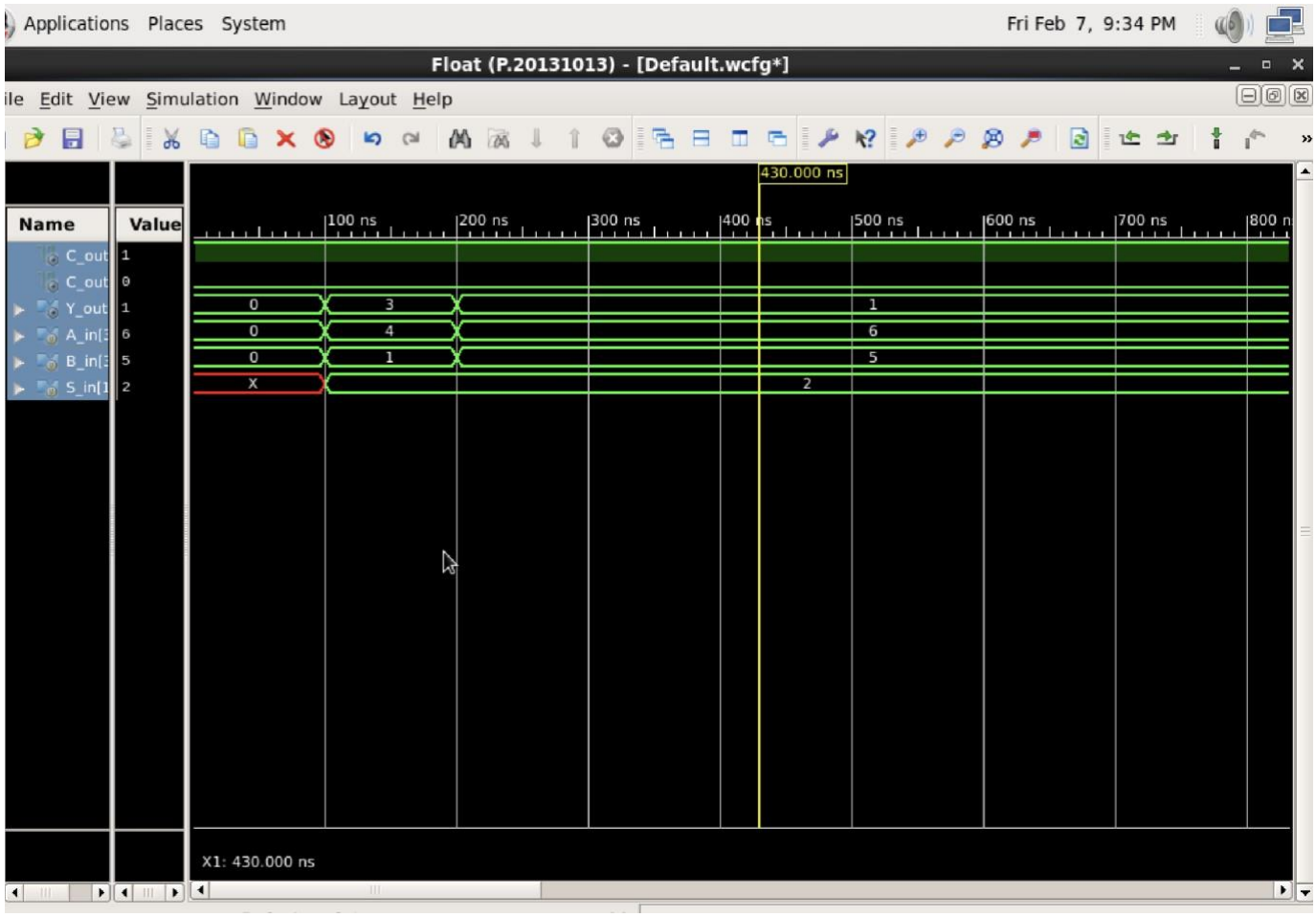
Add Function:

```
26          .Y_out(Y_out)
27      );
28  // Initialize Inputs
29          initial begin
30          // initalize inputs to 0
31          A_in = 4'b0000;
32          B_in = 4'b0000;
33          #100;
34
35          // invert A
36          //A_in = 4'b1111; B_in = 4'b0000; S_in = 00; #100;
37          //A_in = 4'b0011; B_in = 4'b0000; S_in = 00;#100;
38
39          // add A+B
40          A_in = 4'b0010; B_in = 4'b0011; S_in = 01; #100;
41          A_in = 4'b0011; B_in = 4'b0000; S_in = 01;#100;
42
43          // sub A-B
44          //A_in = 4'b0100; B_in = 4'b0001; S_in = 10; #100;
45          //A_in = 4'b0110; B_in = 4'b001; S_in = 10;#100;
46
47          // double A
48          //A_in = 4'b0001; B_in = 4'b0000; S_in = 11; #100;
49          //A_in = 4'b0011; B_in = 4'b0000; S_in = 11;#100;
50      end
51  endmodule
52
```



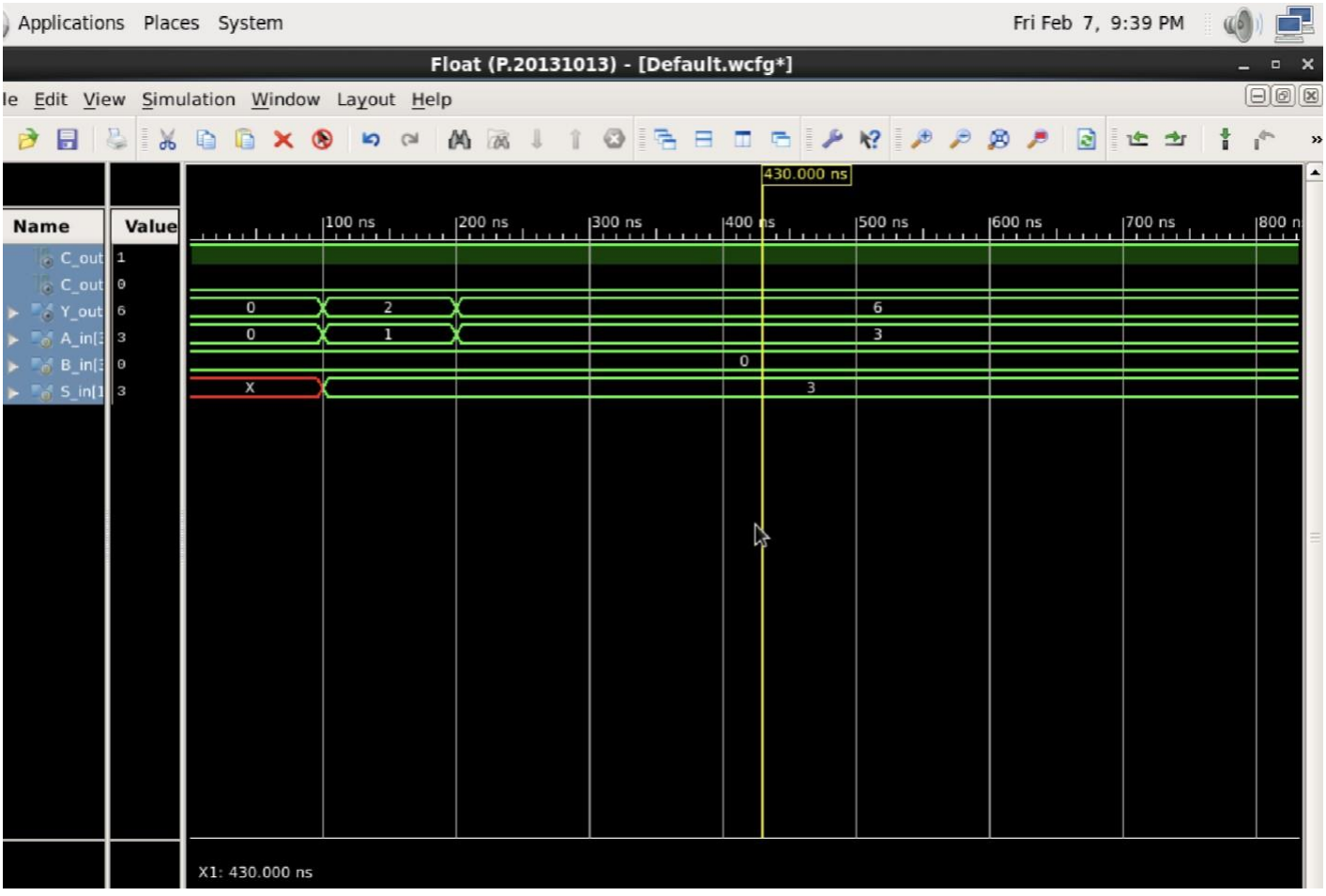
Subtract Function:

```
      .Y_out(Y_out)
   );
// Initialize Inputs
      initial begin
      // initalize inputs to 0
      A_in = 4'b0000;
      B_in = 4'b0000;
      #100;

      // invert A
      //A_in = 4'b1111; B_in = 4'b0000; S_in = 00; #100;
      //A_in = 4'b0011; B_in = 4'b0000; S_in = 00;#100;

      // add A+B
      //A_in = 4'b0010; B_in = 4'b0011; S_in = 01; #100;
      //A_in = 4'b0011; B_in = 4'b0000; S_in = 01;#100;

      // sub A-B
      A_in = 4'b0100; B_in = 4'b0001; S_in = 10; #100;
      A_in = 4'b0110; B_in = 4'b0101; S_in = 10;#100;

      // double A
      //A_in = 4'b0001; B_in = 4'b0000; S_in = 11; #100;
      //A_in = 4'b0011; B_in = 4'b0000; S_in = 11;#100;
   end
endmodule
```



Double Function:

```
26          .Y_out(Y_out)
27      );
28  // Initialize Inputs
29          initial begin
30          // initalize inputs to 0
31          A_in = 4'b0000;
32          B_in = 4'b0000;
33          #100;
34
35          // invert A
36          //A_in = 4'b1111; B_in = 4'b0000; S_in = 00; #100;
37          //A_in = 4'b0011; B_in = 4'b0000; S_in = 00;#100;
38
39          // add A+B
40          //A_in = 4'b0010; B_in = 4'b0011; S_in = 01; #100;
41          //A_in = 4'b0011; B_in = 4'b0000; S_in = 01;#100;
42
43          // sub A-B
44          //A_in = 4'b0100; B_in = 4'b0001; S_in = 10; #100;
45          //A_in = 4'b0110; B_in = 4'b0101; S_in = 10;#100;
46
47          // double A
48          A_in = 4'b0001; B_in = 4'b0000; S_in = 11; #100;
49          A_in = 4'b0011; B_in = 4'b0000; S_in = 11;#100;
50      end
51  endmodule
52
```



Feedback

This lab took over 10 hours to complete. The majority of the time was spent designing the schematic, while the rest was dedicated to testing and debugging. The process required significant attention to detail, as even small wiring errors or bus mismatches could cause unexpected issues. I would consider this lab to be of average difficulty, though it requires patience, problem-solving skills, and a solid understanding of digital circuits to complete successfully. Familiarity with Xilinx ISE, multiplexers, adders, and shift registers was essential in ensuring the schematic was correctly implemented.

Conclusion

This lab provided valuable hands-on experience in designing and implementing a basic Arithmetic Logic Unit (ALU). By integrating inverters, full adders, a barrel shifter, and a custom multiplexer, I was able to develop a functional ALU capable of performing bitwise inversion, addition, subtraction, and doubling. Throughout the process, I gained a deeper understanding of binary arithmetic operations, two's complement subtraction, and the importance of signal routing in hardware design.

The most challenging aspect of the lab was debugging schematic errors, particularly bus width mismatches and incorrect multiplexer wiring. However, overcoming these challenges reinforced my understanding of circuit design principles and troubleshooting techniques. Moving forward, I would aim to improve efficiency in schematic design by planning connections more systematically and verifying signal paths before running simulations. Overall, this lab was a valuable learning experience that enhanced my digital design skills and prepared me for more complex circuit implementations in future projects and labs.