**Lab 5: Simple Arithmetic Logic Unit**

Faizan Bangash

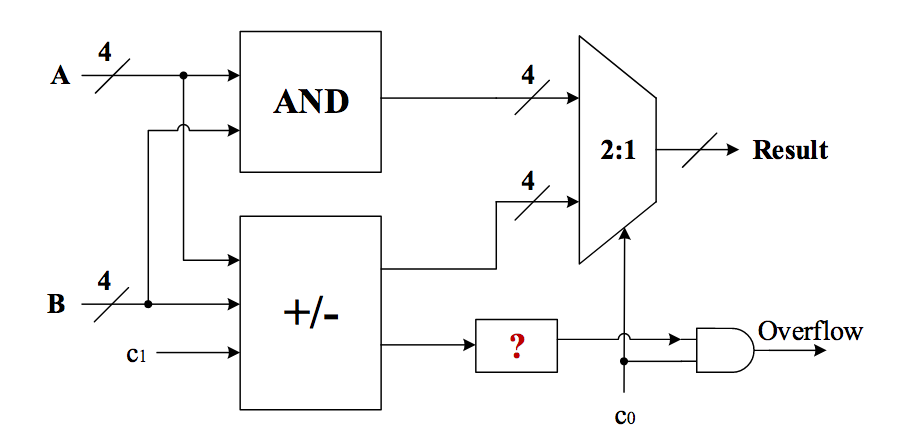
TA : Saurav Kumar Sahu

ECEN 248 – 302

July 10th , 2018.

**Objectives**: In this lab, I learned how to implement a simple ALU that can do addition, subtraction, as well as bitwise AND. Additionally, I know have more experience with Mux and more complex circuits.

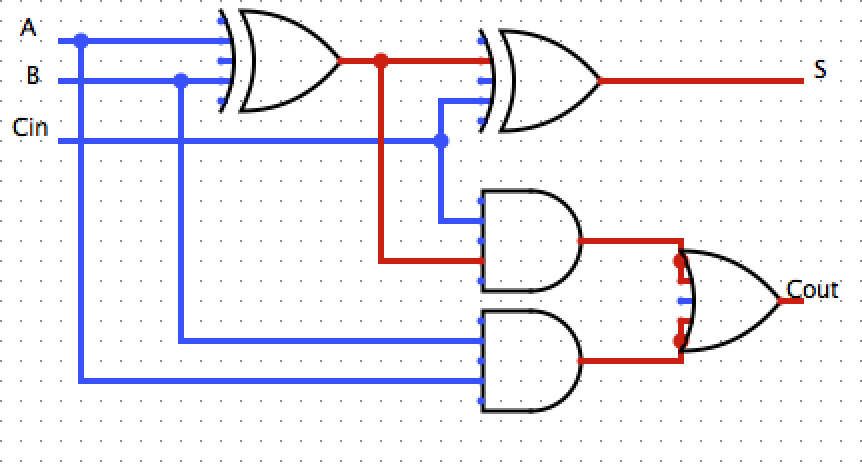
**Design**: In this lab, we did addition and subtraction using the SN74HC283E and a C1 control to give the SN74HC283E the sign to either add or subtract. After this, we added the 2:1 MUX SN74CT257N chip to the circuit. From here, we added another control, C0, to determine if we would do the AND operation or the addition/subtraction. Seen in diagram below is an overview of the ALU design. Additionally, the ALU outputs to multiple LEDs for the results as seen below in the diagram.



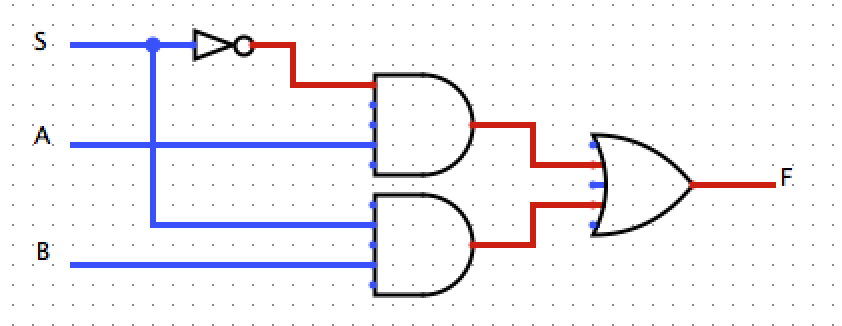
ALU Design from the lab manual

|  |  |  |  |
| --- | --- | --- | --- |
| S | A | B | Y |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

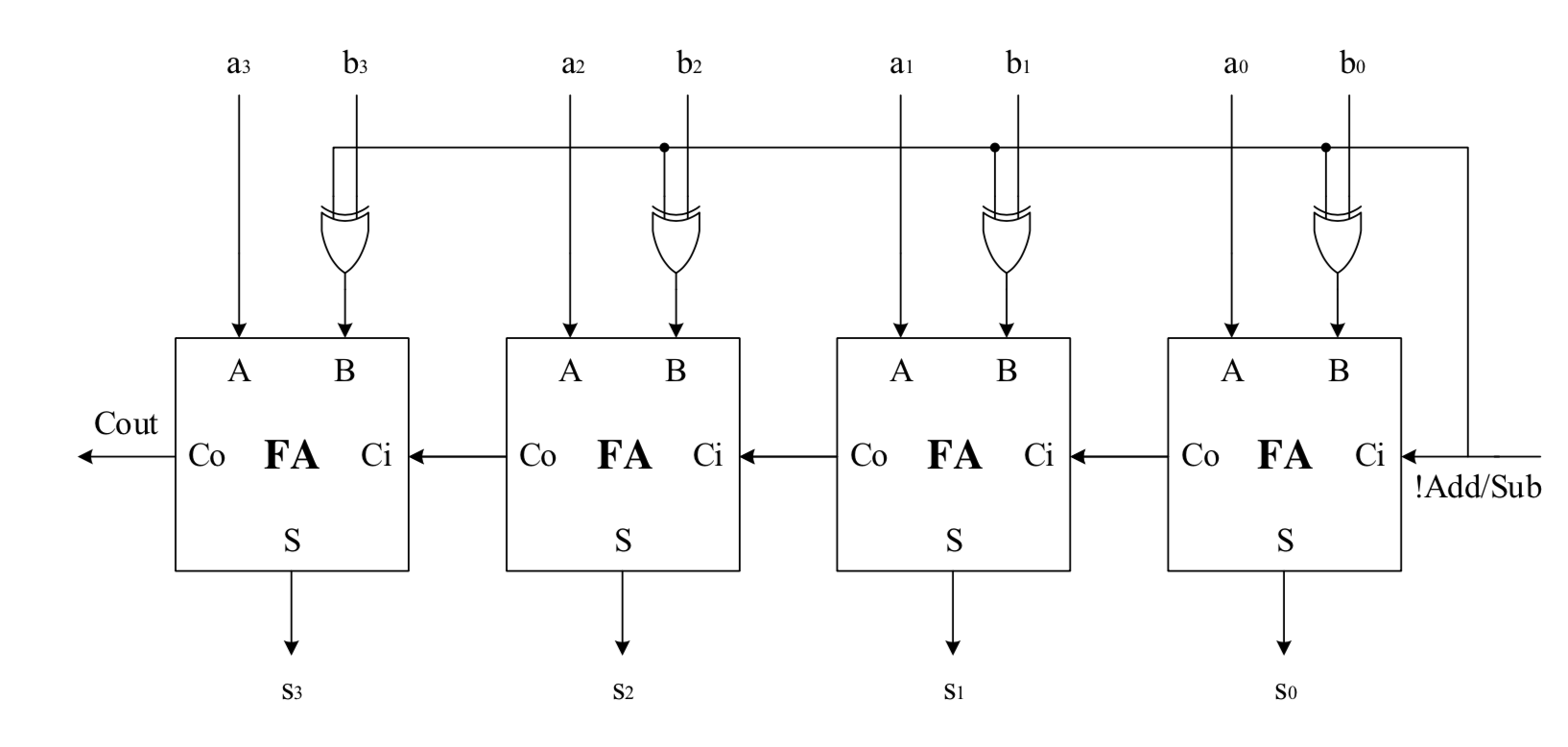
Truth table for 2:1 MUX and Boolean Equation: Y=BS+AS’



Full Adder Gate schematic



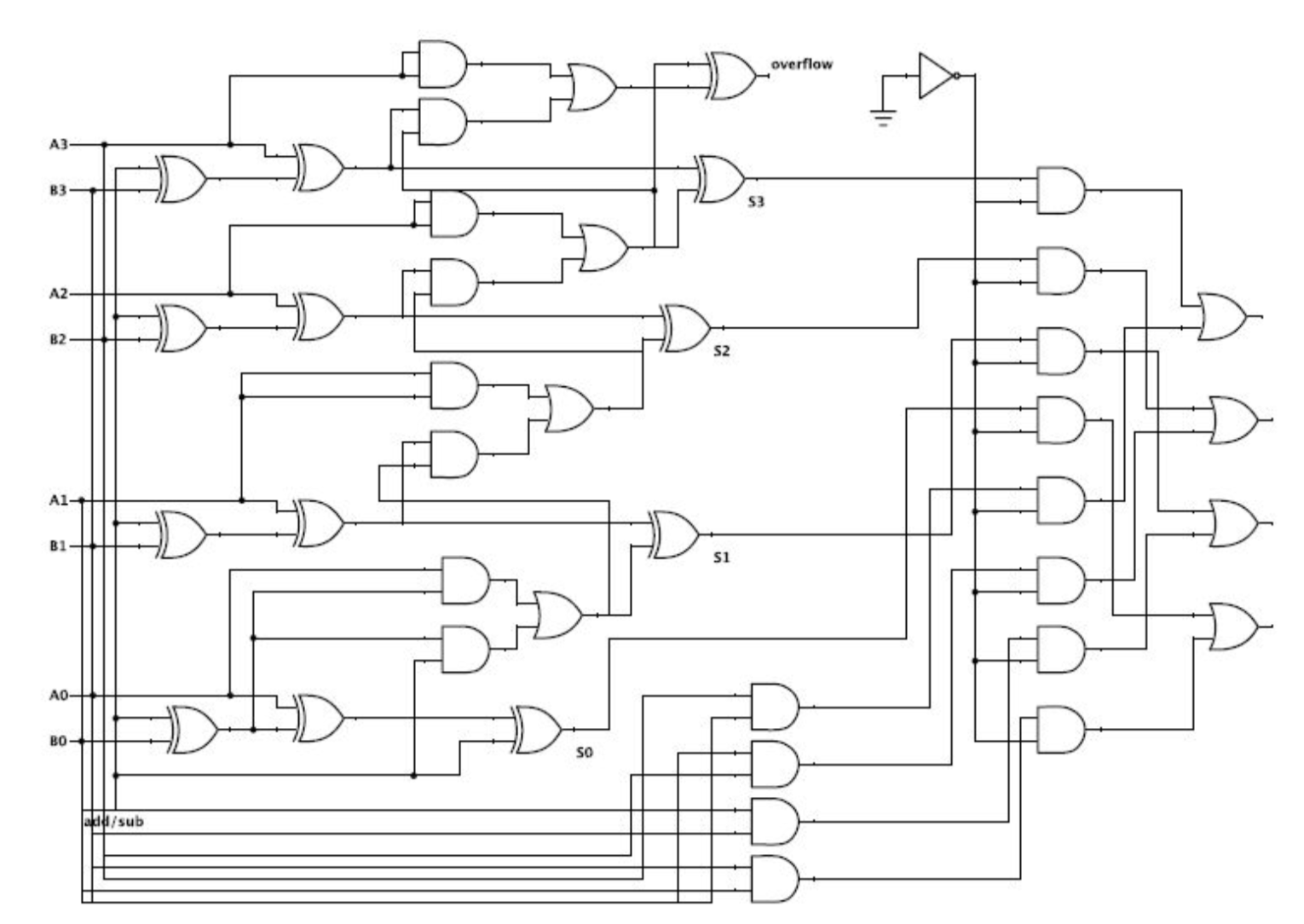
Mux gate schematic



ALU Design simplified for easier viewing from lab manual

|  |  |  |
| --- | --- | --- |
| C0 | C1 |  |
| 0 | 0 | ADD |
| 0 | 1 | SUB |
| 1 | 0 | AND |
| 1 | 1 | AND |

ALU control signals and the operation that is performed



Full ALU Design- Green line represents the critical path

(I got Full ALU design from online source at digikey.com)

**Results –**

In this lab, I was able to get the addition working correctly in one try but my and bit wasn’t working properly. After help from the TA I figured out that I had flipped flop the wires into the MUX. I ended up hard wiring the C values and I was able to get the proper results.

**Conclusion –**

Overall, I was able to learn a lot about debugging and ALUs in this lab. Because of the issues I ran into, I was forced to take a deeper look at the circuit and try to find my errors. Now, I also have more experience with a 2:1 MUX where I can now apply this for more realistic circuits. One thing I would have changed would have been to organize my wires more to allow for me to debug my design easier.

**Questions:**

1.) Table for overflow

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C0 | C1 | Operation | A | B | Result | Overflow |
| 0 | 0 | AND | 0100 | 0110 | 0100 | 0 |
| 0 | 1 | AND | 0110 | 1101 | 0100 | 0 |
| 1 | 0 | ADD | 0100 | 0110 | 1010 | 0 |
| 1 | 0 | ADD | 0100 | 1101 | 0001 | 1 |
| 1 | 0 | ADD | 1101 | 1001 | 0110 | 1 |
| 1 | 1 | SUB | 0100 | 0111 | 1101 | 0 |
| 1 | 1 | SUB | 0110 | 1001 | 1101 | 0 |

2.) The critical path is shown above in green line. The path has a delay of 5 units.

3.) To detect overflow, we first have to look at question 1 and see which values trigger overflow. We see this in the adder/subtractor figure. We see that C2 and C3, the carry out values from the 3rd and 4th adder. Overflow occurs when we run out of digits to represent the result in the 4 bit format. we can see that overflow can be detected using the follow gate schematic

