Scott Nidell

Lab 5 Notes

1000921465

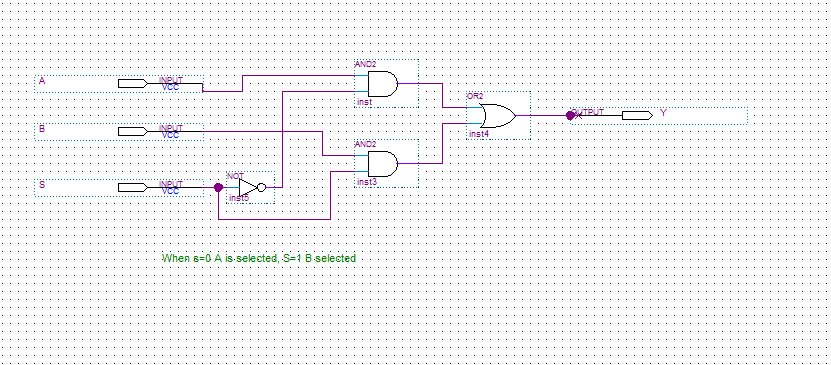
CSE 2441-001

**Introduction**: Lab5 introduced the ALU of the final project. This unit should be able to conduct several functions including Add, subtract, AND and XOR two four bit operands together. This lab utilizes the adder subtractor from lab three and also implements a 2 to 1 selector and a quad XOR and quad AND circuit to create the top level implementation.

**Theory:**  Using a good modular design, implement a correct efficient design to realize an Arithmetic logic unit. Once implemented test using the provided table of values to extensively test the unit for accuracy. This unit is the heart of the TRISC project at the end of the semester and will perform all basic functions of a computer give its opcode.

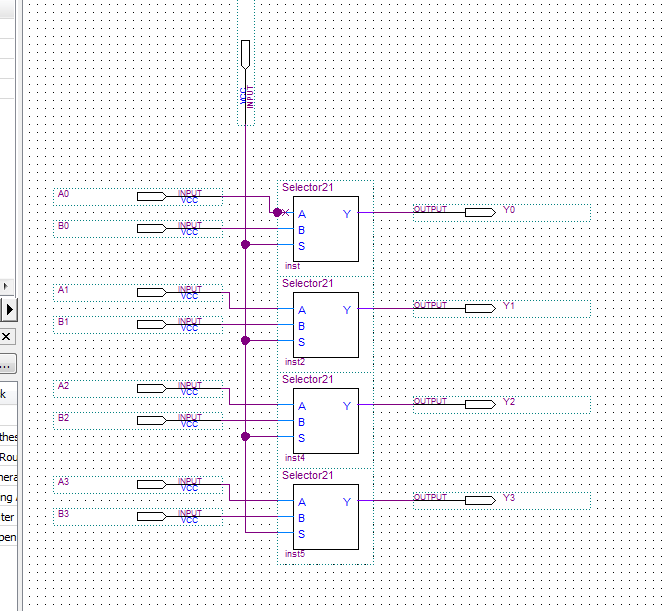
The first module to be implemented in Quartus is the 2 to 1 selector. This will then be used in the four input solution. This unit will allow the switching affect needed to select different modules in the ALU.

**Figure 1:** *2 to 1 selector implemented in Quartus.*

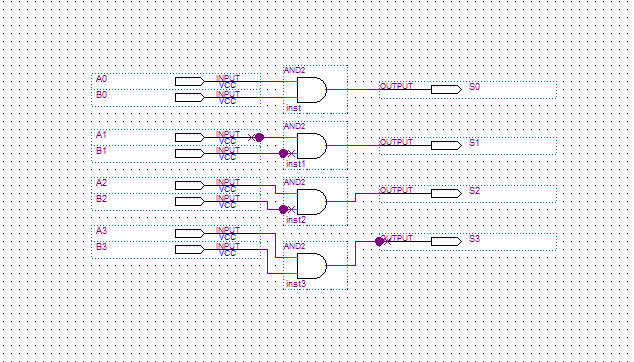
**

The four input version is then realized and tested in a Quartus design. This unit will be used to switch between the different features of the ALU.

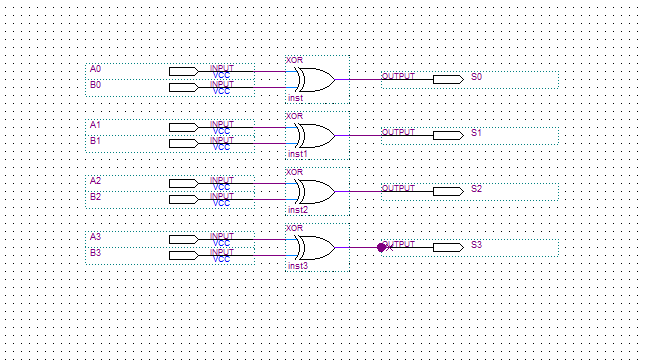
**Figure2:** *A Quad 2 to 1 selector implemented in Quartus.*

**

Next the quad AND and quad XOR units are designed.

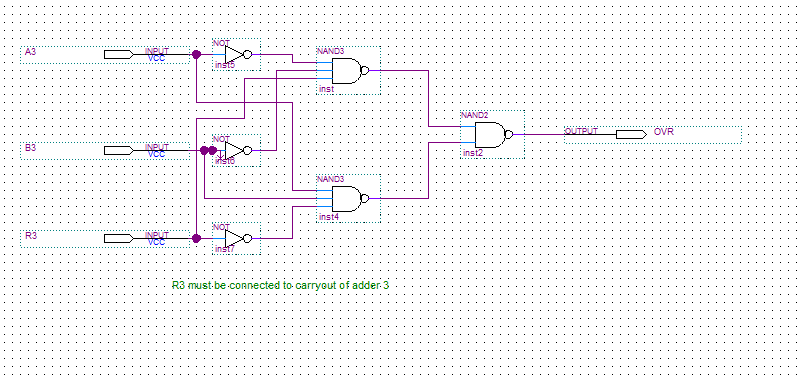
**Figure 3:** *quad AND implemented in Quartus*

***Figure 4:*** *quad XOR implemented in Quartus*

**

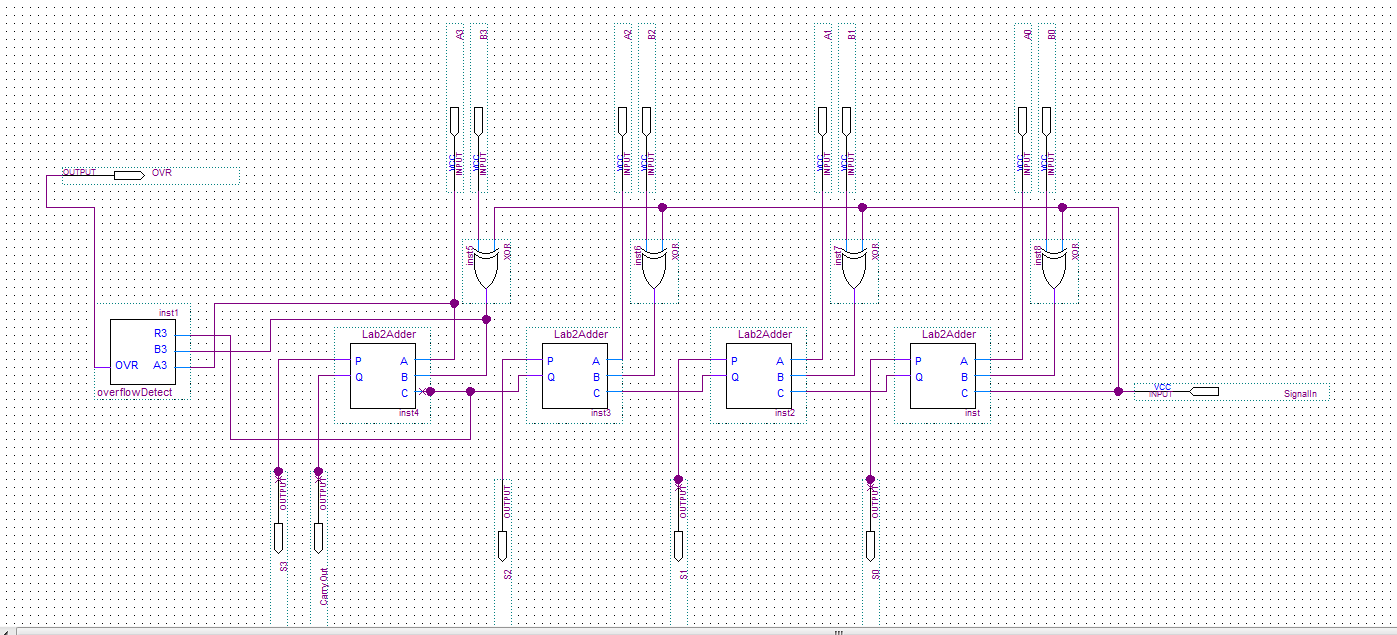
Lab 4 created a basic 4 bit adder subtractor. However this unit did not have overflow detection and is now implemented through a separate module in Quartus. This unit allows detection of overflow when two bits create a value outside its bit range.

**Figure 5:** *Overflow detection implemented in Quartus.*



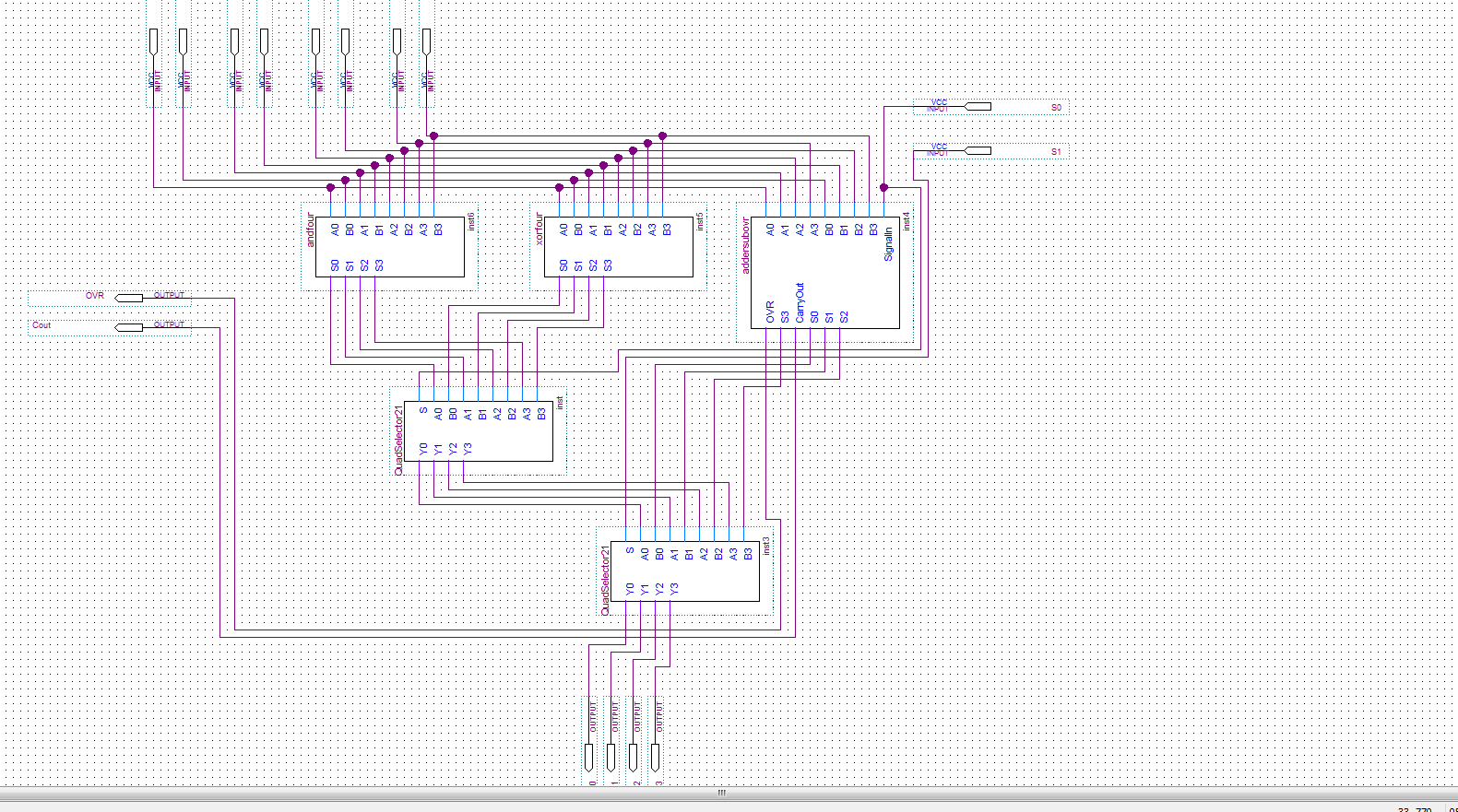
The overflow is then implemented into the adder subtractor module to complete the sub assembly.

**Figure 6:** *Overflow added to the adder/subtractor circuit in Quartus.*



Now that all modules have been created and tested individually they now must be assembled at a top level in the ALU.

**Figure 7:** *ALU fully implemented with all sub modules in Quartus*



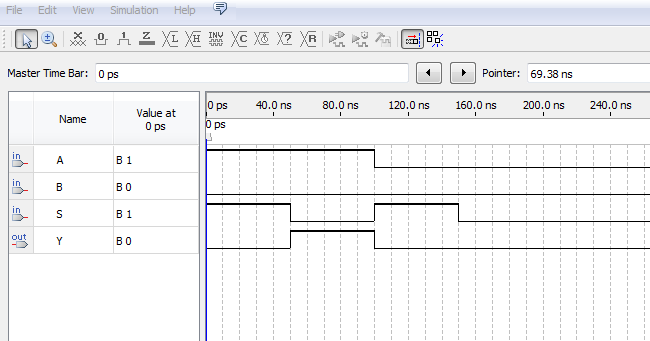
The ALU was then tested with a waveform and also demonstrated in lab using the DE1 and verified by the lab TA. A table with these values is recorded below:

**Figure 8:** *Table for final results*

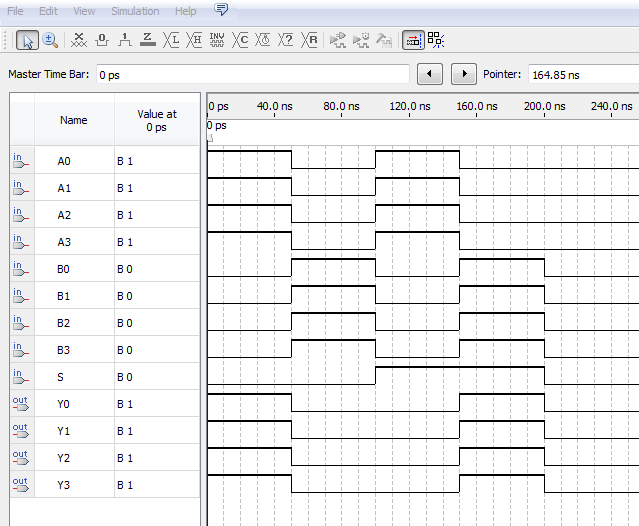
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***A*** | ***B*** | ***R=A+B*** | ***OVR*** | ***Cout*** | ***R=A-B*** | ***OVR*** | ***Cout*** | ***A\*B*** | ***AXORB*** |
| *0110* | *0001* | *0111* | *0* | *0* | *0101* | *0* | *1* | *0000* | *0111* |
| *0110* | *0010* | *1000* | *1* | *0* | *0100* | *0* | *1* | *0010* | *0100* |
| *0010* | *1001* | *1011* | *0* | *0* | *1001* | *1* | *0* | *0000* | *1011* |
| *1101* | *1111* | *1100* | *0* | *1* | *1110* | *0* | *0* | *1000* | *0101* |
| *1100* | *1001* | *1010* | *1* | *1* | *0011* | *0* | *1* | *1000* | *0101* |
| *1010* | *1110* | *1000* | *0* | *1* | *1100* | *0* | *0* | *1010* | *0100* |
| *0110* | *1111* | *0101* | *0* | *1* | *0111* | *0* | *0* | *0110* | *1001* |
| *1001* | *0111* | *0000* | *0* | *1* | *0010* | *1* | *1* | *0001* | *1110* |

All modules were tested using waveforms derived in Quartus below are the results.

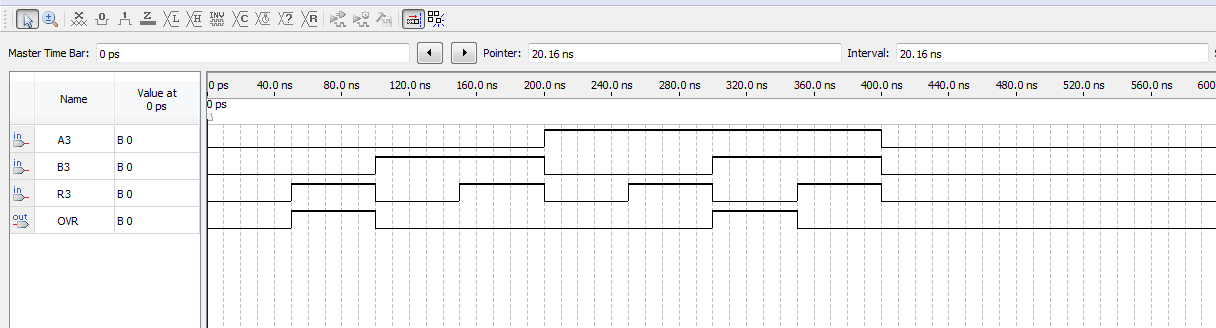
**Figure 9:** *Two to 1 selector wave form*

**

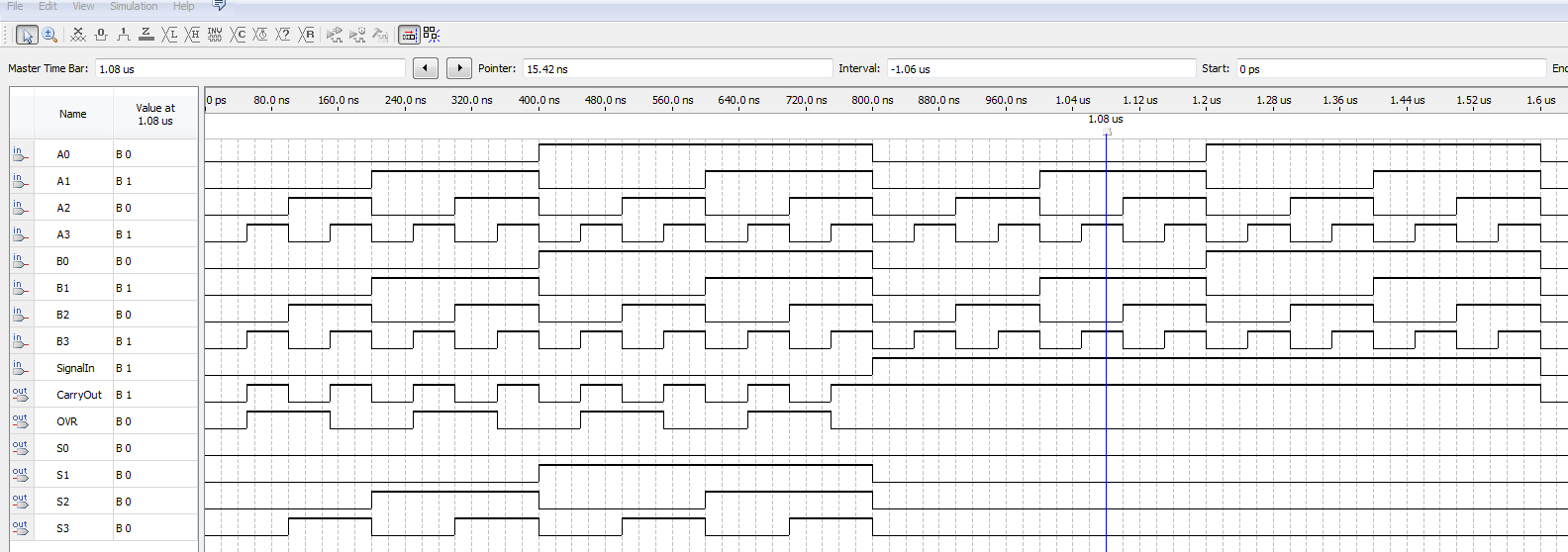
**Figure 10:** *Quad two to one wave form*

****

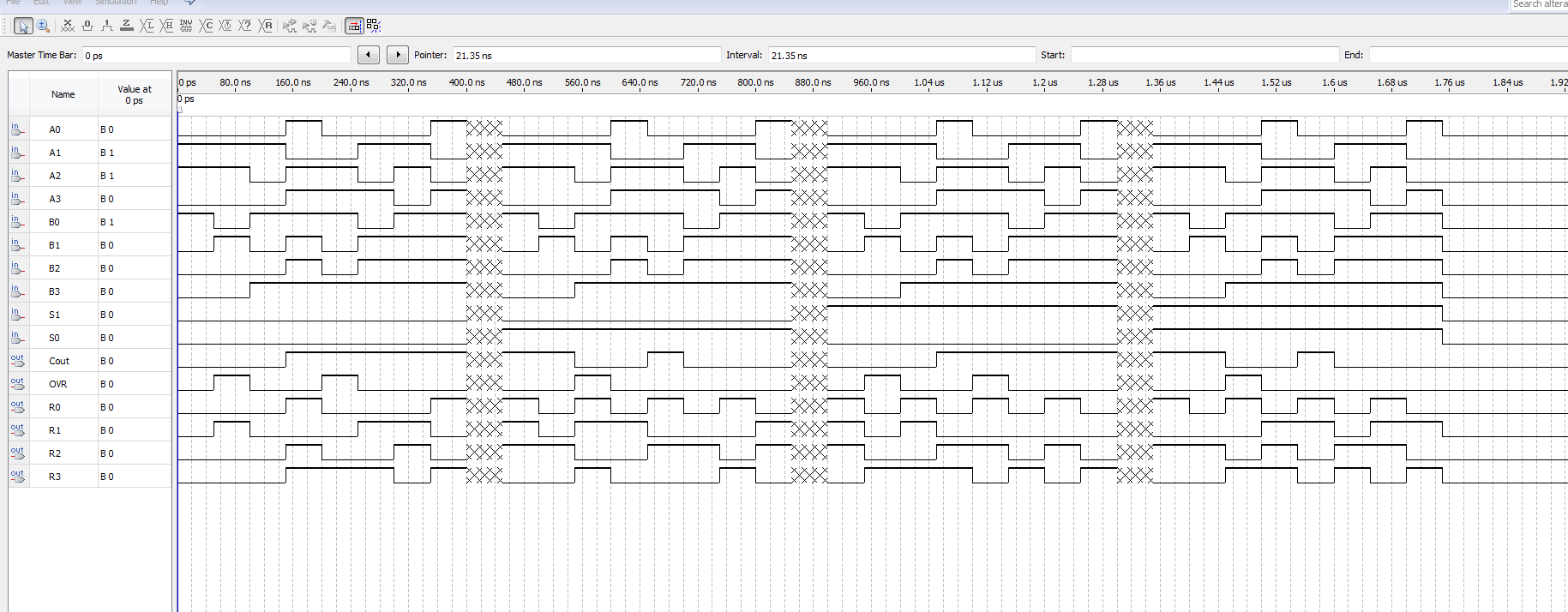
**Figure 11:** *Over flow detection wave form*

**

**Figure 12:** *Adder subtractor with over flow detection wave form*

**

**Figure 13:** *Full ALU implementation Waveform.*

**

**Conclusion:** The lab went smoothly. Using the lectures notes and PowerPoints along with guidance from the lab instructor made the transition a lot easier. Every module was first unit tested and then integrated and tested again to verify functions. This is necessary as to reduce the troubleshooting required when the final project is assembled.