Scott Nidell

Lab 4 Notes

1000921465

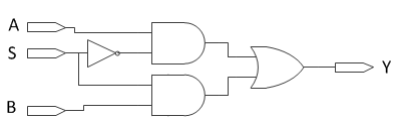
CSE 2441-001

**Introduction** Lab 5 uses multiple sub assemblies to create an Arithmetic Logic Unit (ALU). This modules consist of 2 selection logic circuits made of 4 two-to-1 selectors, 1-Quad Input XOR unit, 1-Quad input AND unit, and a modified Add/Subtractor to include over flow. These were all designed in prelab.

**Theory:**  Given the logical gate equivalent circuits of each module, this are created in quartus and saved as symbol files for easier use for later projects. A quad XOR would provide a XOR logic to 4A bits and 4B bits. A Quad AND would provide AND logic for 4A bits and 4B bits, and also the Adder/Subtractor would add and subtract 4A bits and 4B bits while detecting overflow. These modules were designed in series and managed through selection logic that would pick one of the 4 operations.

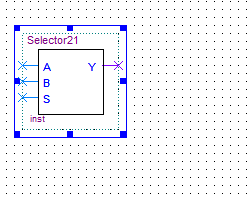
**Procedure:** First step in designing the ALU is creating a 2-to-1 selector from the given logic gates (Figure1):

**Figure 1: Gate logic for a 2-to-1 selector**

****

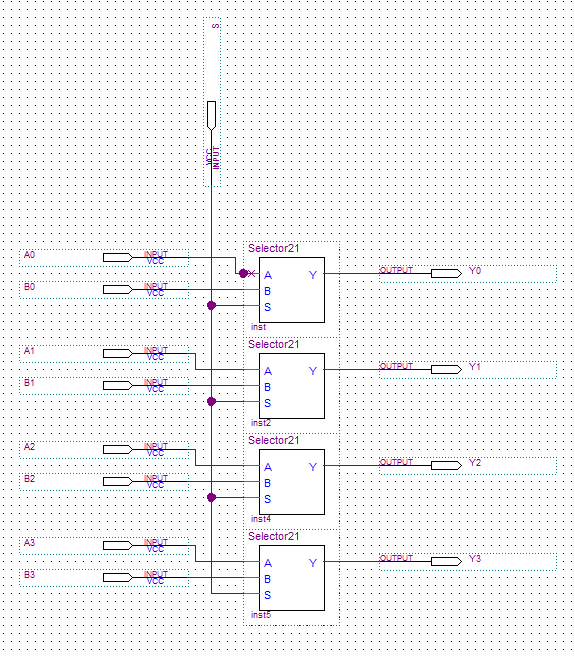
This was implemented into Quartus and a block symbol was created (Figure 2):

**Figure 2: Block Symbol for 2-to-1 selector**

****

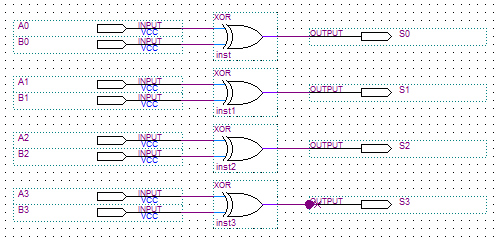
Next a quad 2-to-1 selector was created using the block above (Figure 3)

**Figure 3: Quad 2-to-1**

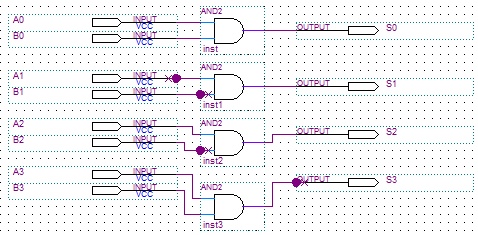
****

The Quad XOR and AND were created next using Quartus (Figure 4 and 5)

**Figure 4: Quad XOR**

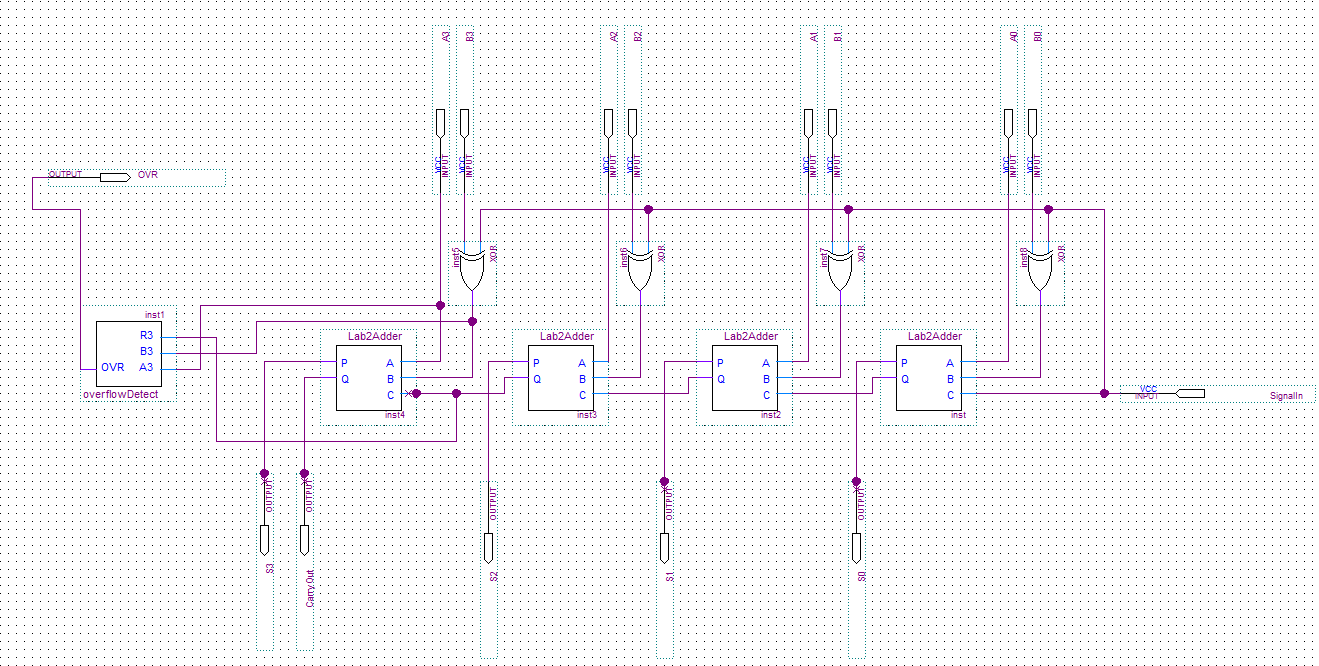
****

**Figure 5: Quad AND**

****

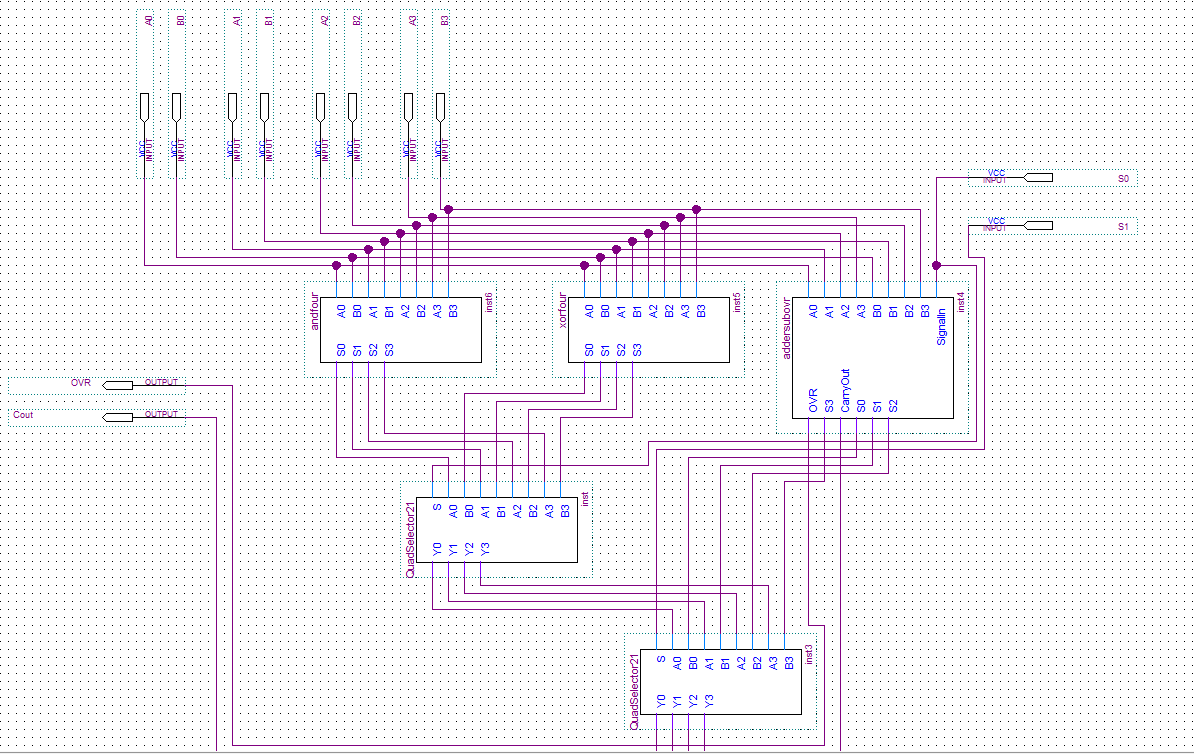
Next overflow detection was implemented into the Add/Subtractor (Figure 6)

**Figure 6: Adder/Subtractor with overflow**

****

Lastly, all sub-modules were combined to create the ALU (Figure 7)

**Figure 7: ALU (Final)**

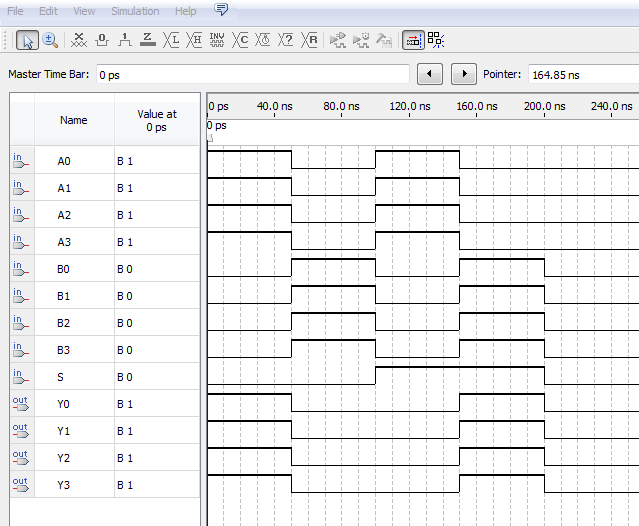
****

The ALU was then tested and verified by the lab instructor using the following values:

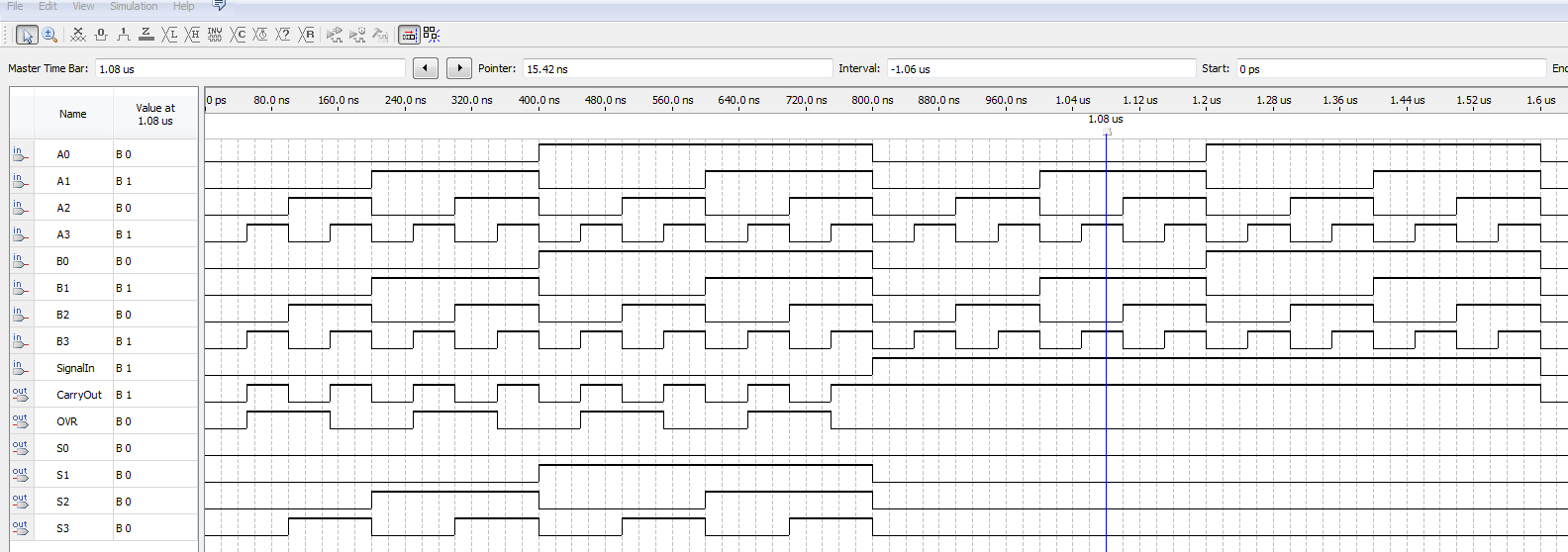
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *ALU Test Values* | | | | | | | | | |
| A | B | R=A+B | OVE | Cout | R=A-B | OVR | Cout | A\*B | AXORB |
| 0110 | 0001 | 0111 | 0 | 0 | 0101 | 0 | 1 | 0000 | 0111 |
| 0110 | 0010 | 1000 | 1 | 0 | 0100 | 0 | 1 | 0010 | 0100 |
| 0010 | 1001 | 1011 | 0 | 0 | 1001 | 1 | 0 | 0000 | 1011 |
| 1101 | 1111 | 1100 | 0 | 1 | 1110 | 0 | 0 | 1101 | 0010 |
| 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 | 000 | 0 | 1 | 0010 | 1 | 1 | 0001 | 1110 |

Also all modules were verified in Quartus before test verification with lab instructor was conducted. The wave forms are below:

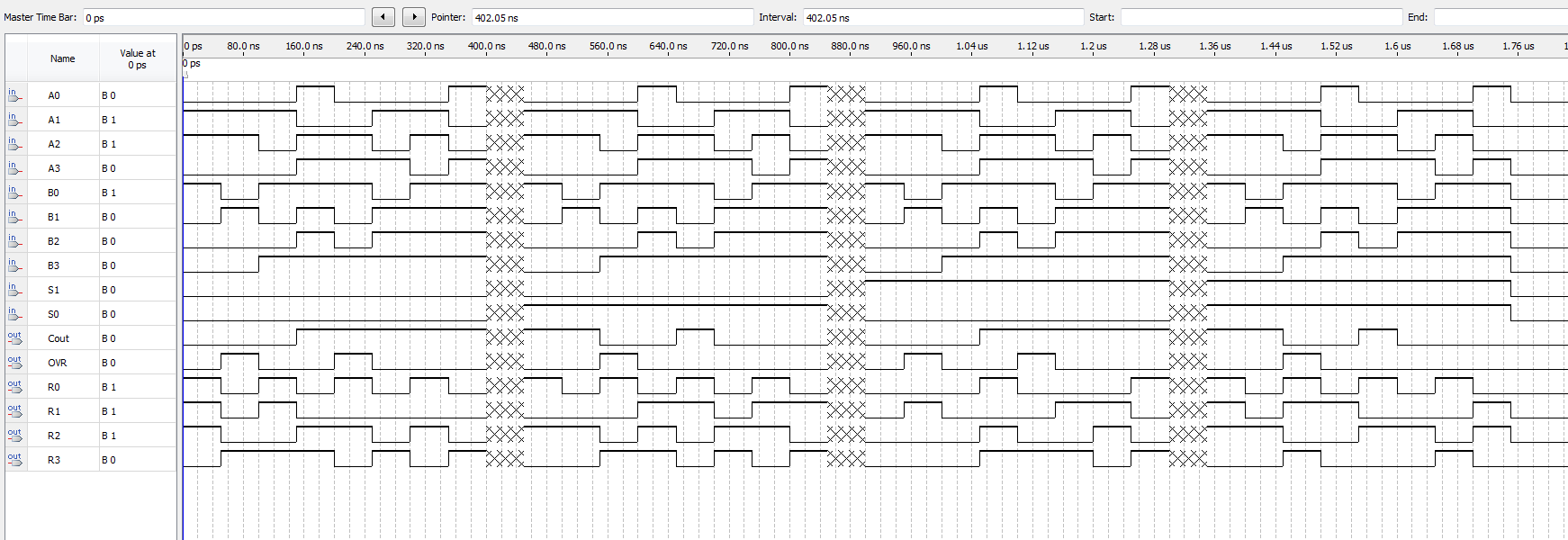
**Quad 2-to-1 Selector**



**Adder Subtractor**

****

**ALU**

****

**Conclusion:** Prelab was intensive as far as time. The utilization of class slides helped procure the final product in terms of gate equivalent logic. Extensive testing was done on the DE1 to ensure accuracy as this is on part of the final project and must be accurate for ease of troubleshooting.