Doruk Taneli **LAB3:** 23.02.2018

**Design and Implementation of a 4-bit**

**Arithmetic and Logic Unit**

**Introduction:**

The aim of this lab is to design and implement a 4-bit Arithmetic and Logic Unit using modular design.

**Methodology:**

There are 3 modules in this ALU. Arithmetic Unit, Logic Unit and the Multiplexer.

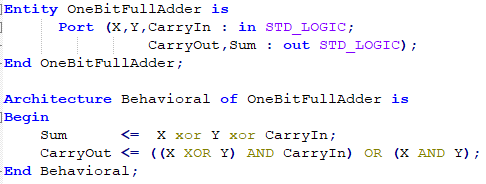
Arithmetic Units makes use of Full adder. The arguments of the full adder are determined according to the select bits in the BitSliceALU. The arguments are then passed to the OneBitFullAdder which does the operations and returns CarryOut and ArithOut.

Logic Unit determines which operations to be done to A and B from select bits. Then it returns the answer to LogicOut. It is all done in BitSliceALU.

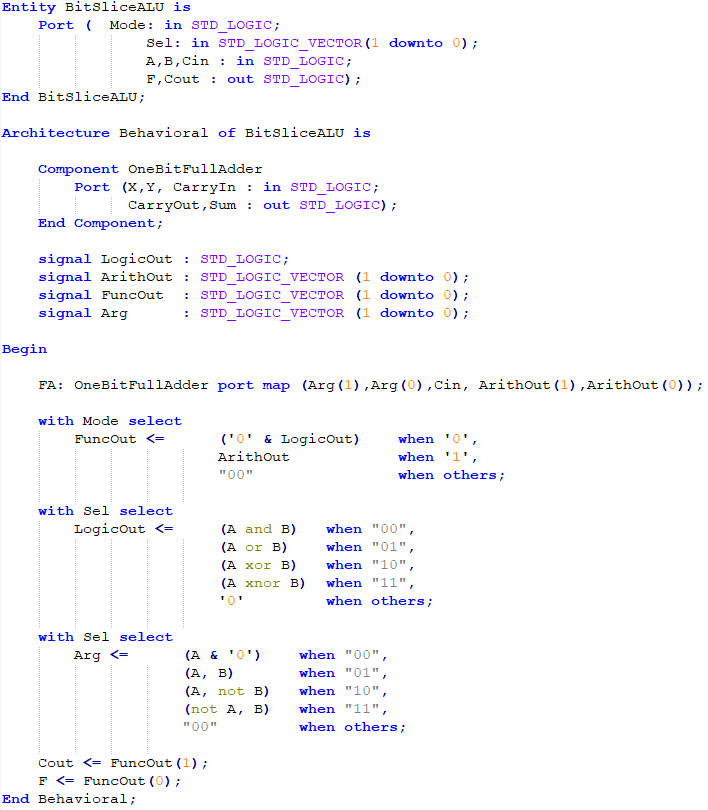
The Multiplexer was implemented separately. In BitSliceALU, according to which mode is selected, FuncOut either returns ArithOut or LogicOut. In FourBitALU it is implemented so that, In Arithmetic Unit mode, The LEDs are off and Seven Segment Display is on; in Logic Unit mode, The LEDs are working and Seven Segment Display is off.

I first implemented the BitSliceALU and then added 4 of them to create FourBitALU using vectors and portmap.

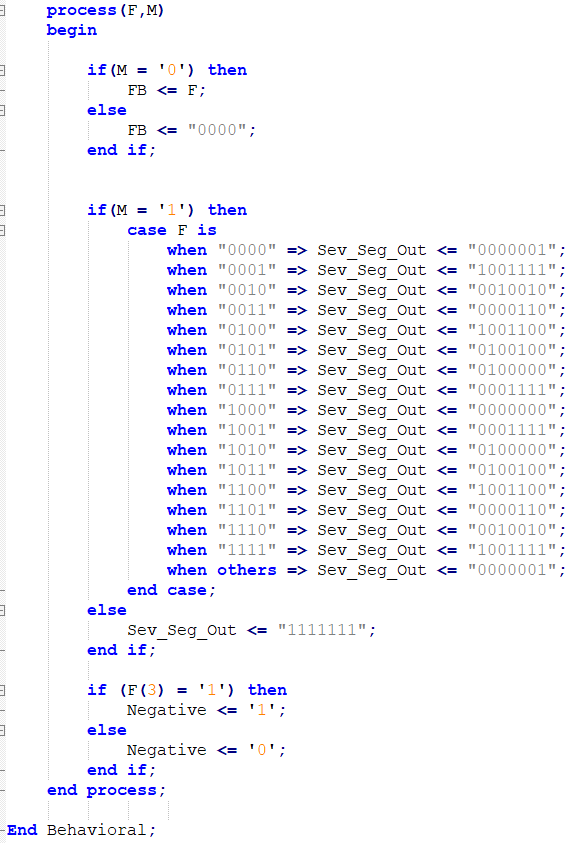
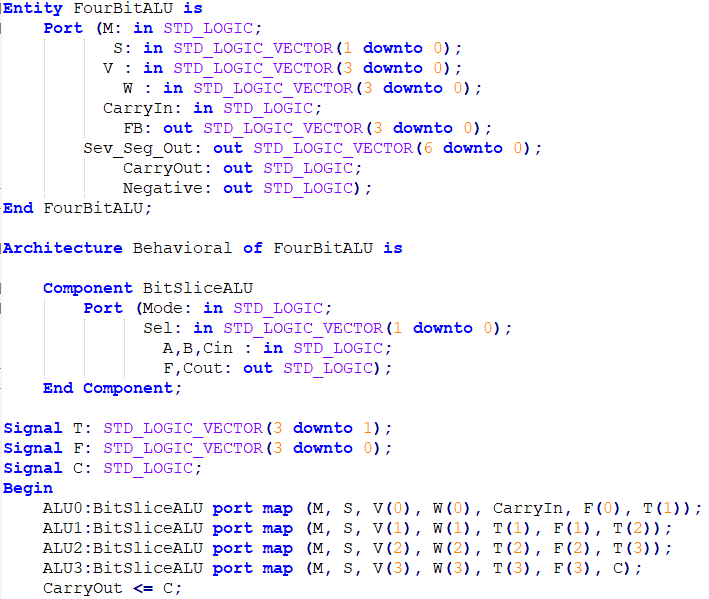
The codes are present below. The colors are different because I used notepad++ to view instead of Xilinx.



*OneBitFullAdder code*



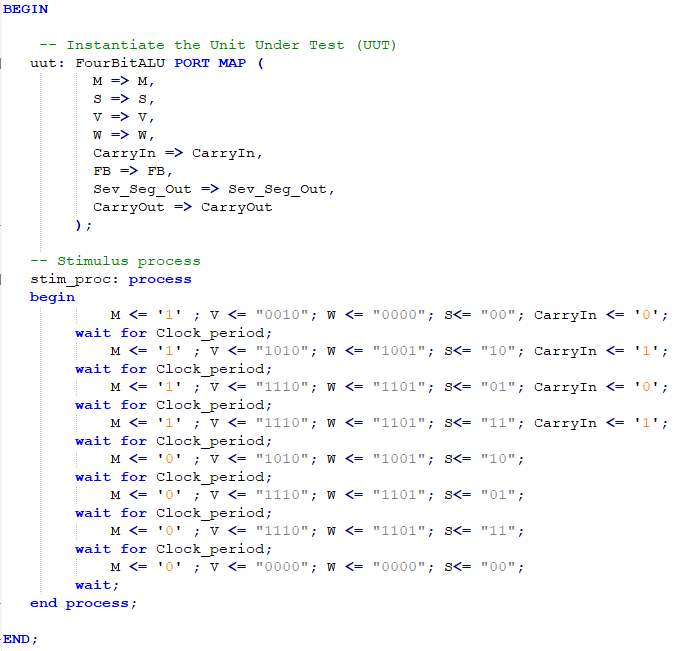
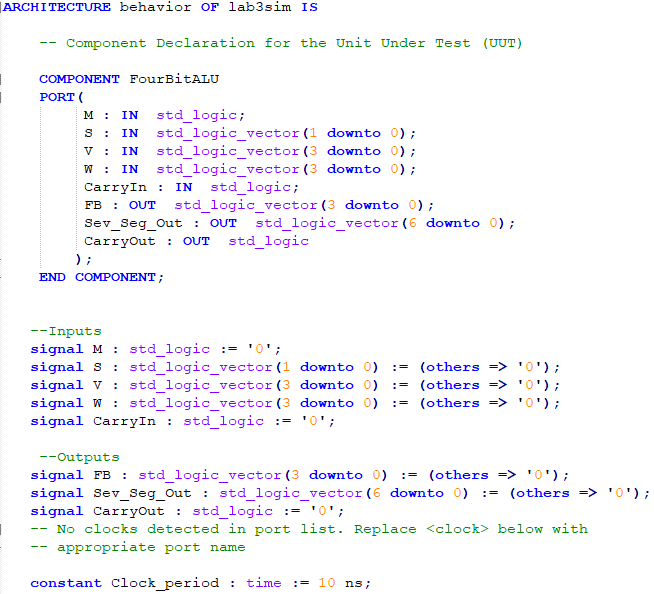
*BitSliceALU code*



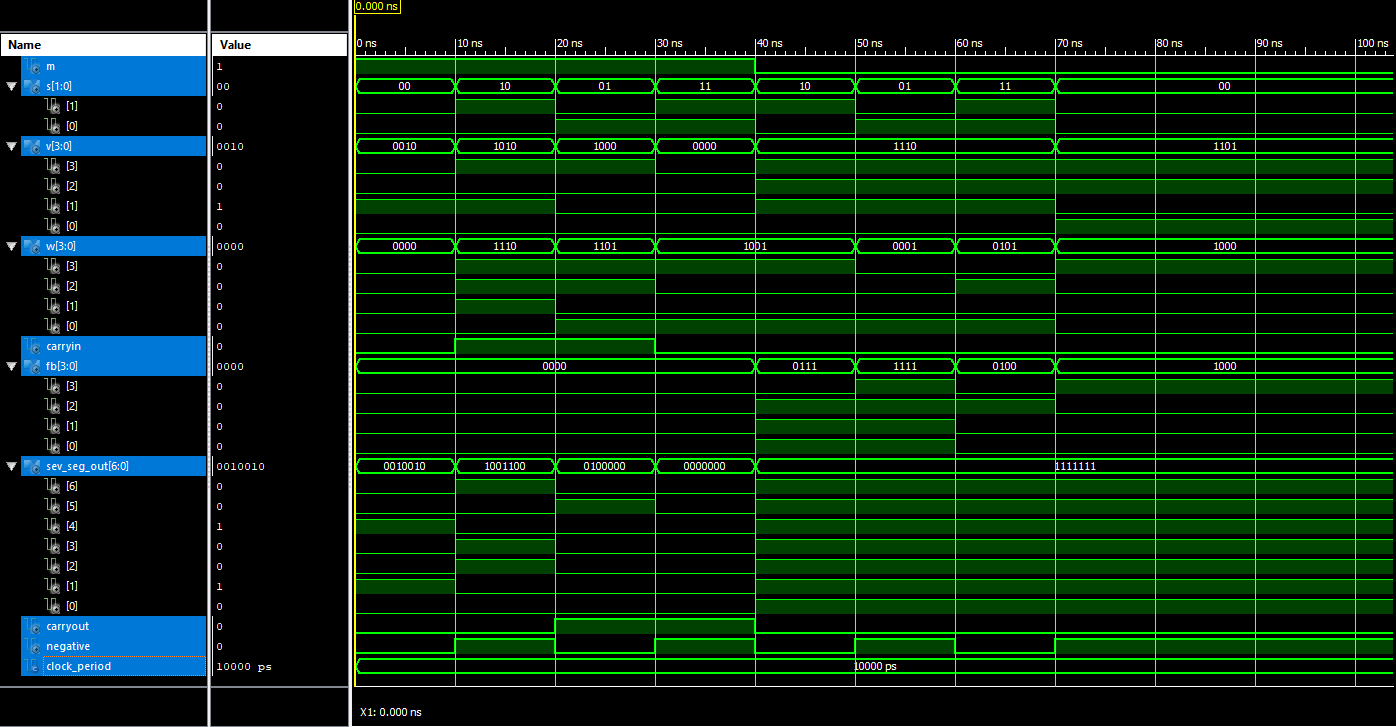
*FourBitALU code*

**Experimental Results:**

I mapped A, B and Select to switches, carry in and mode to buttons, carry out and F to given LEDs, Negative to leftmost LED and set up the Seven Segment Display. The design worked correctly as expected on the FPGA board. There are test cases to check both arithmetic and the logic unit on the simulation. The simulation code and simulation waveforms are shown below.



*Simulation Code*

**

*Simulation Code*

**Discussion and Conclusion:**

In this lab, I have implemented a 4-bit Arithmetic and Logic Unit using four 1-bit ALUs. I learned about multiplexers and I also learned much more about the syntax of VHDL to be able to write the code.