Lab: Add-Sub unit with Register file (3-ported RAM)

Chue Zhang

Csc343 Fall 2021

Professor Gertner

Table of contents

Objective………………………………………………………………………………………………………………………………………………………………………………...…………...…………….………….3

Specification……………………………………………………………………………………………………………………………………………………………………………….………….……………….………4

32-Bit Adder/Subtractor…………………………………………………………………………………………………………………………………………………………………………………….5

2-Port RAM [32x32]…………………………………………………………………………………………………………………………………………………………………….…………………….6

3-Port RAM [32x32]……………………………………………………………………………………………………………………………………………………………………….….………………8

Memory Initialization File [MIF]……………………………………………………………………………………………………………………………………………………….……….……….9

Instruction File…………………………………………………………………………………………….….……………………………………………………………………………………………….10

3-Port RAM Adder-Sub Unit …………………………………………………………………………………………….….………………………………………………………………………….11

Simulations…………………………………………………………………………………………………………………………………………………………………………………………….....………………….12

3-Port RAM Adder-Sub Unit ………………………………………………………………………………………………………………………………………………………..………………….12

Most Positive 32-bit Integer + 1…………………………………………………………………………………………………………………………………………………………13

Most Positive 32-bit Integer – 1…………………………………………………………………………………………………………………………………………………………14

Most Negative 32-bit Integer + 1……………………………………………………………………………………………………………………………………………………….15

Most Negative 32-bit Integer – 1………………………………………………………………………………………………………………………….……………………………16

Most Positive 32-bit Integer – Most Negative 32-bit integer………………………………………………………………………………………..…………………...17

Most Positive 32-bit Integer + Most Negative 32-bit integer………………………………………………………………………………………………………………18

Most Positive 32-bit Integer – Most positive 32-bit integer……………………………………………………………………………………………………….……...19

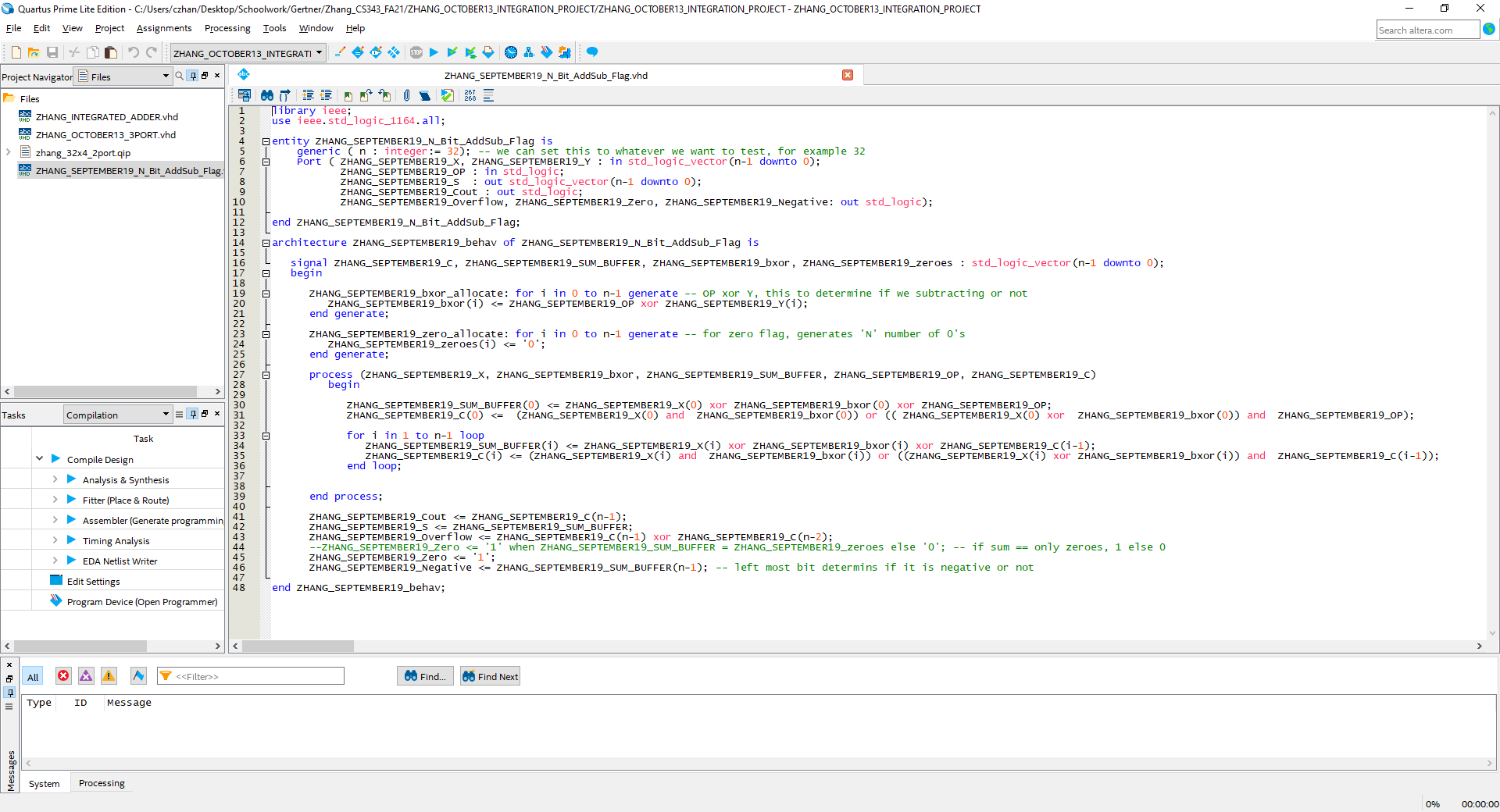
Conclusion……………………………………………………………………………………………………………………………………………………………………………....…………...…………..………….20

**Objective**

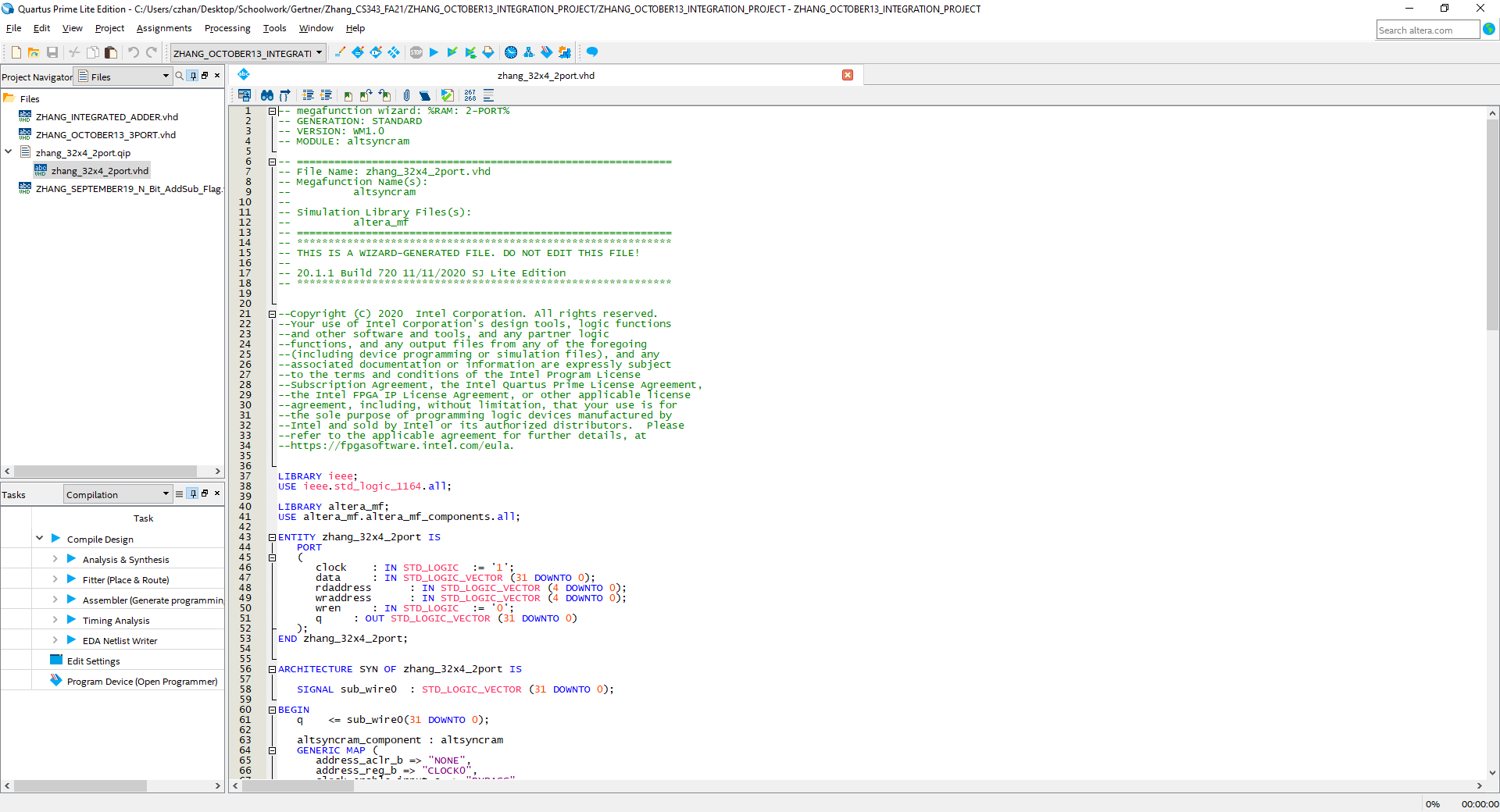
In this Laboratory Assignment, we will be Integrating an 32-bit Adder/Subtractor Unit into a 32 x 32 3-Port RAM with a Memory Instruction File(MIF). Furthermore, what will be included is a 32-bit MIPS like instruction that will be read and inputted into the 32-bit Adder/Subtractor unit.

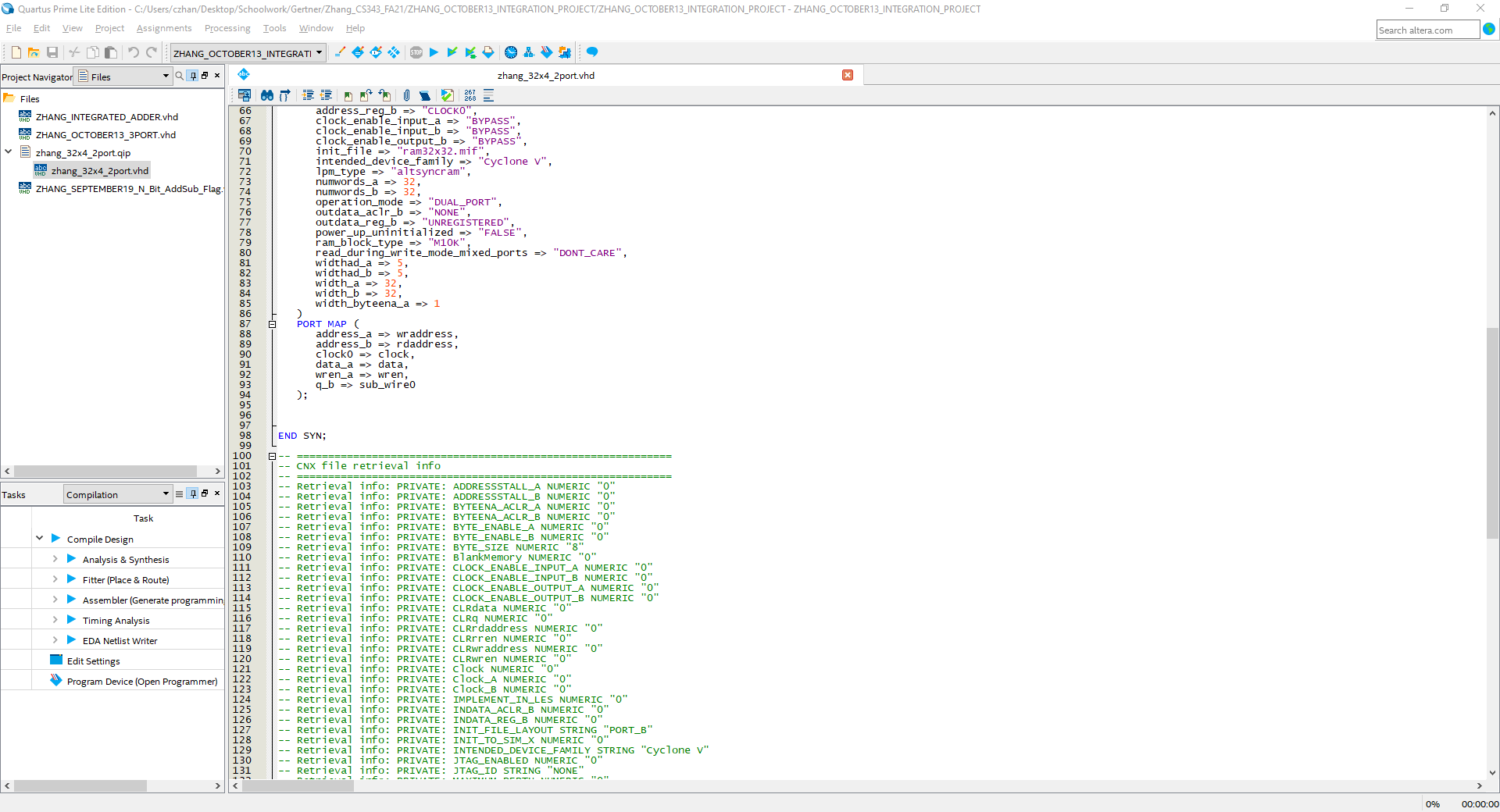
**Specification**

A 32-Bit MIPS like instruction is inputted in which is broken down in the **operation code [6 bits], RS[5 bits], RT[5 bits], RD[5 bits]** with the remaining bits being 0’s**. Operation code** is 6-bit longs where if “000000” , it will add and “000001” would be to subtract. Furthermore, **RS** and **RT** are the input values that we insert into the Adder/Subtractor unit and **RD** is where we are writing to. Next, we integrate an Adder/Subtractor Unit into the 3-Port Ram where the first 3 addresses of our MIF file are instantiated with the **Most Positive 32-Bit Integer**, **Most Negative 32-bit Integer** **and 1.** The User will specify the address in the 32-bit instruction to obtain the 32-Bit Integer in which Addition or Subtraction can be performed on these values. After computation is completed, the value will be written into and of the 29 other addresses that were not instantiated in the MIF file. For example, we compute **Most Positive 32-Bit Integer + 1,** where once computations are finished, we can write the value into the 4th address in the MIF file which has not been instantiated.

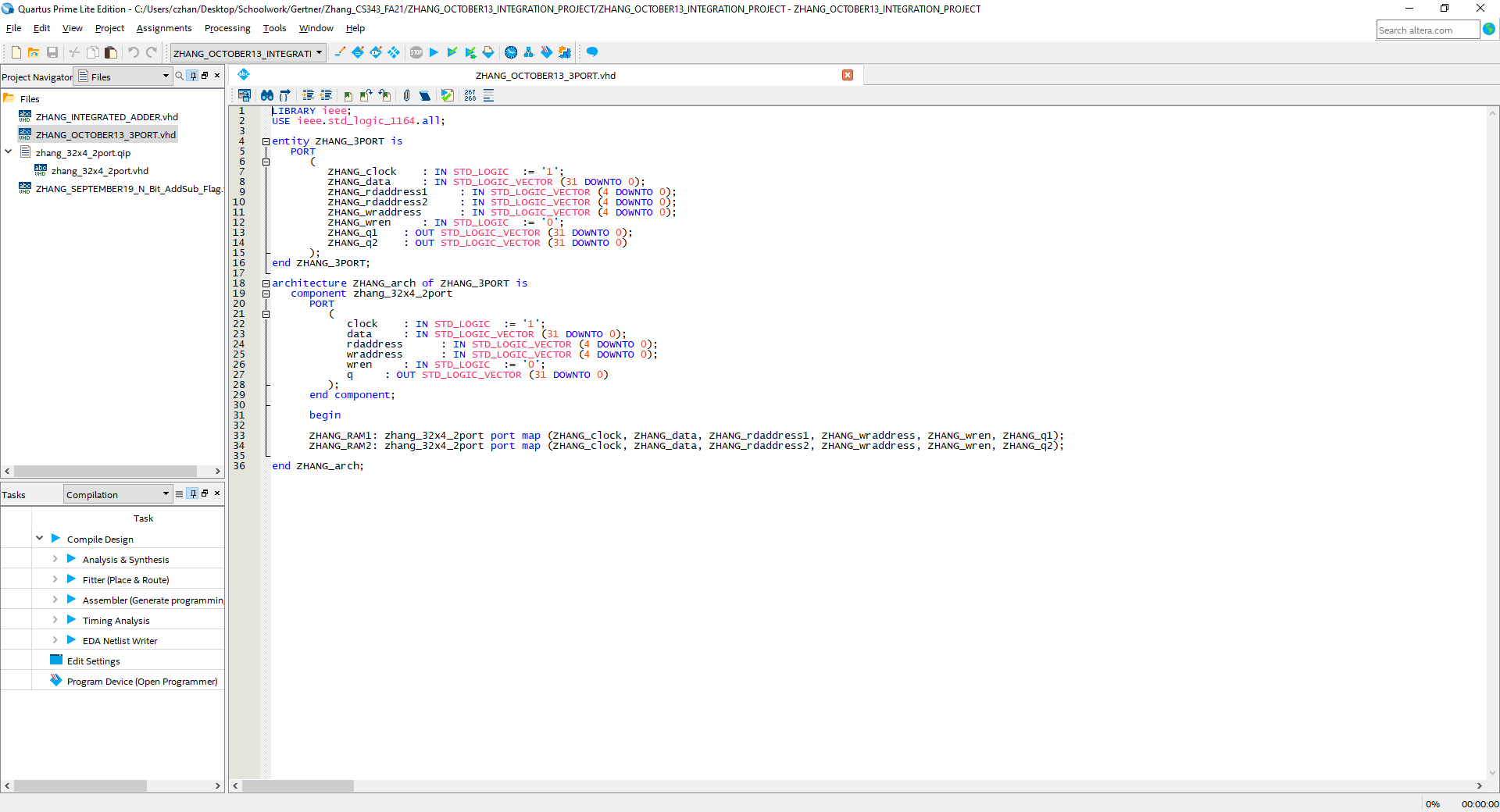


32-Bit Adder/subtractor unit that was created previously in the Adder Laboratory assignment. What is included is logic that detects for Overflow, Negative and Zero as well as an Operator Bit that determines whether or not a number should perform subtraction or addition.

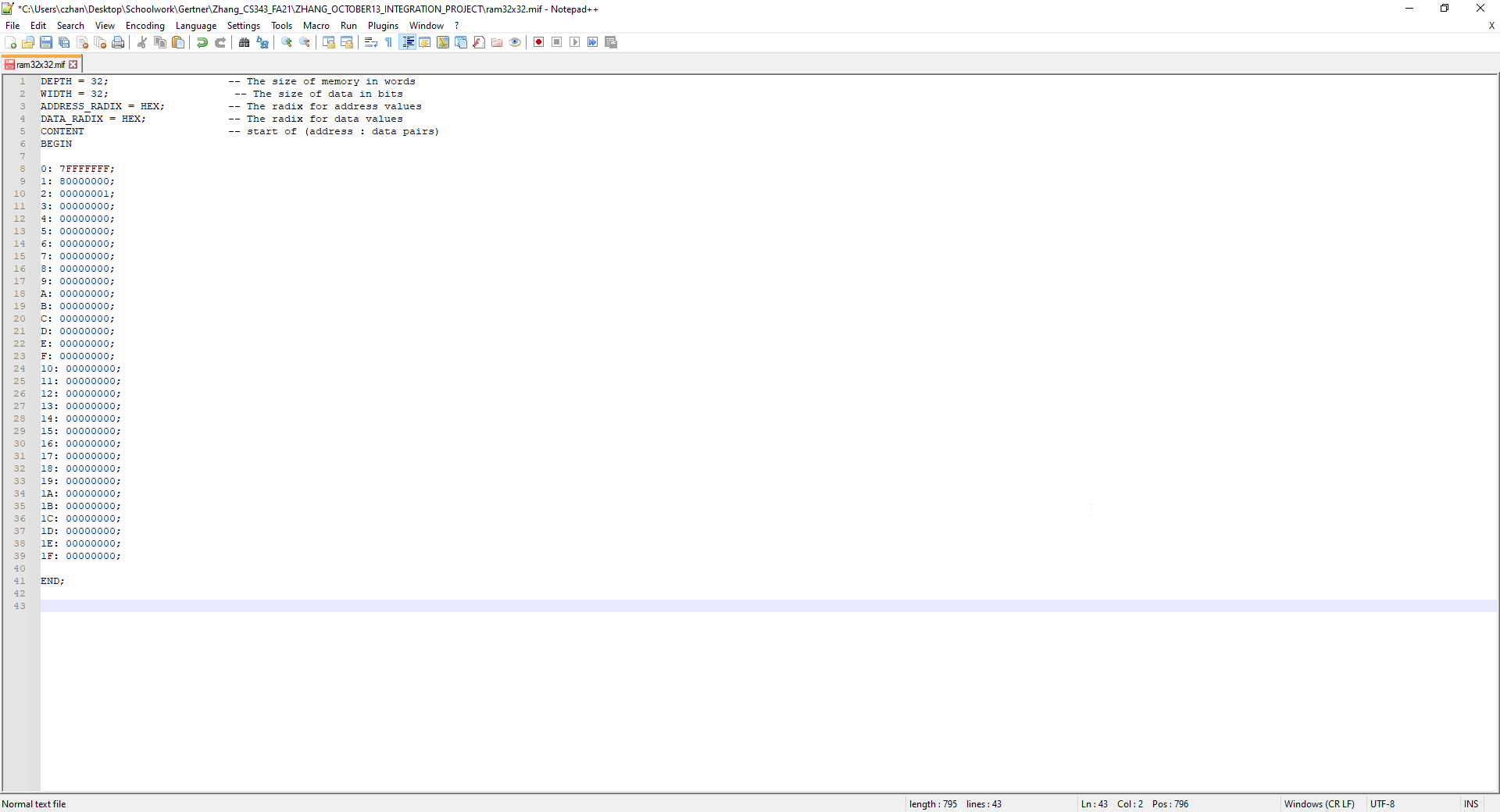




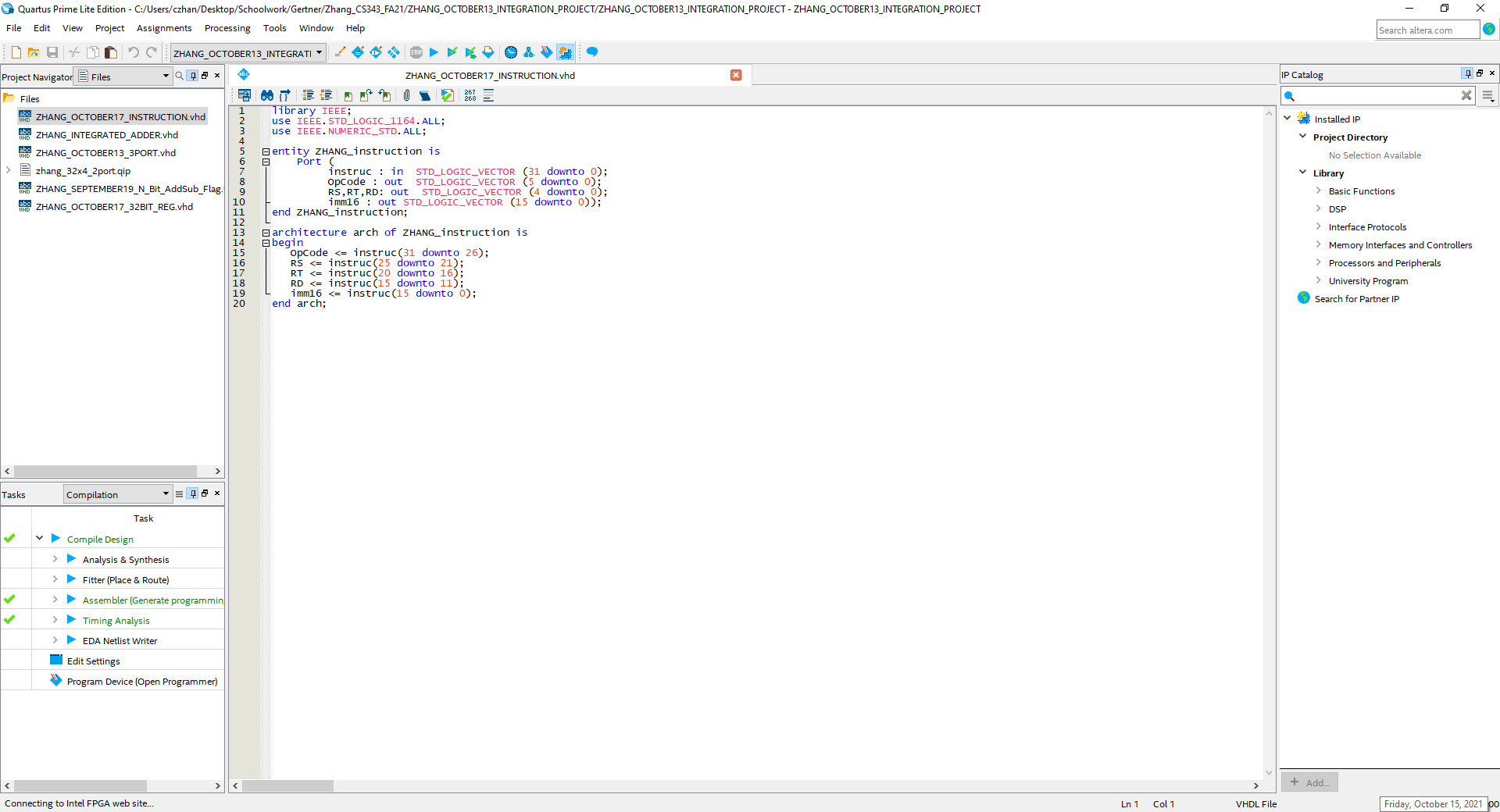
Here we have a 2-Port with 32x32 Memory that was generated using LPM in Quartus. This file was generated to require a MIF File to perform any functionality.



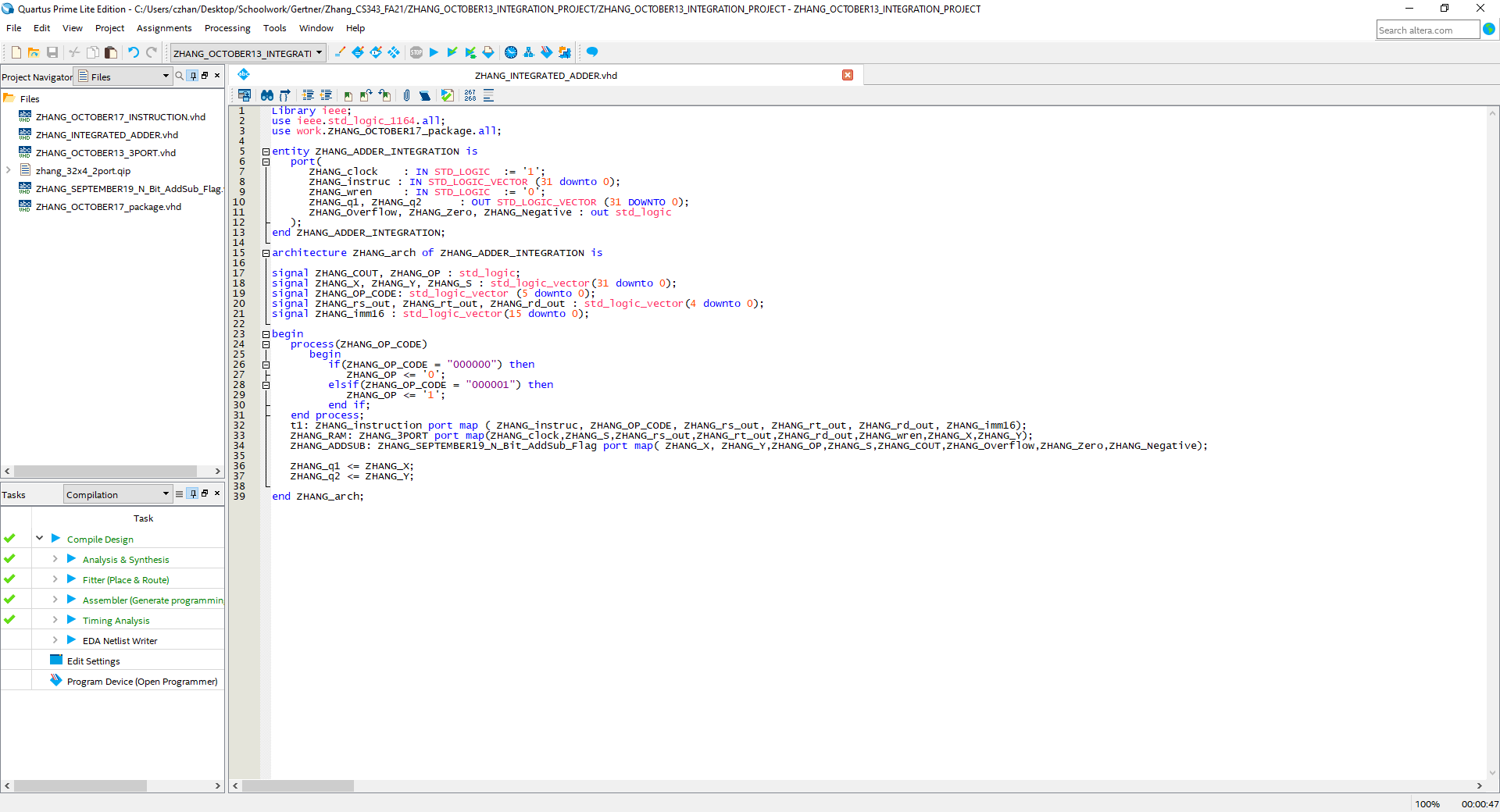
The code above showcases the usage of the previously generated 2-Port RAM in developing a 3-Port RAM. In a 3-port RAM, what is included is 2 read addresses and two q’s which are the outputs after reading an address. Notice ZHANG\_raddress1, ZHANG\_raddress2, ZHANG\_q1 and ZHANG\_q2.



Above is the MIF file that will be used for reading and writing to. Notice that the 1st address points towards the Largest Positive value, the 2nd address points to the Largest Negative value and the 3rd address points to 1. The rest of the address point to 0 as a placeholder where we will later be writing new values into.



In the screenshot above showcases how I process the 32-bit MIPS like instruction



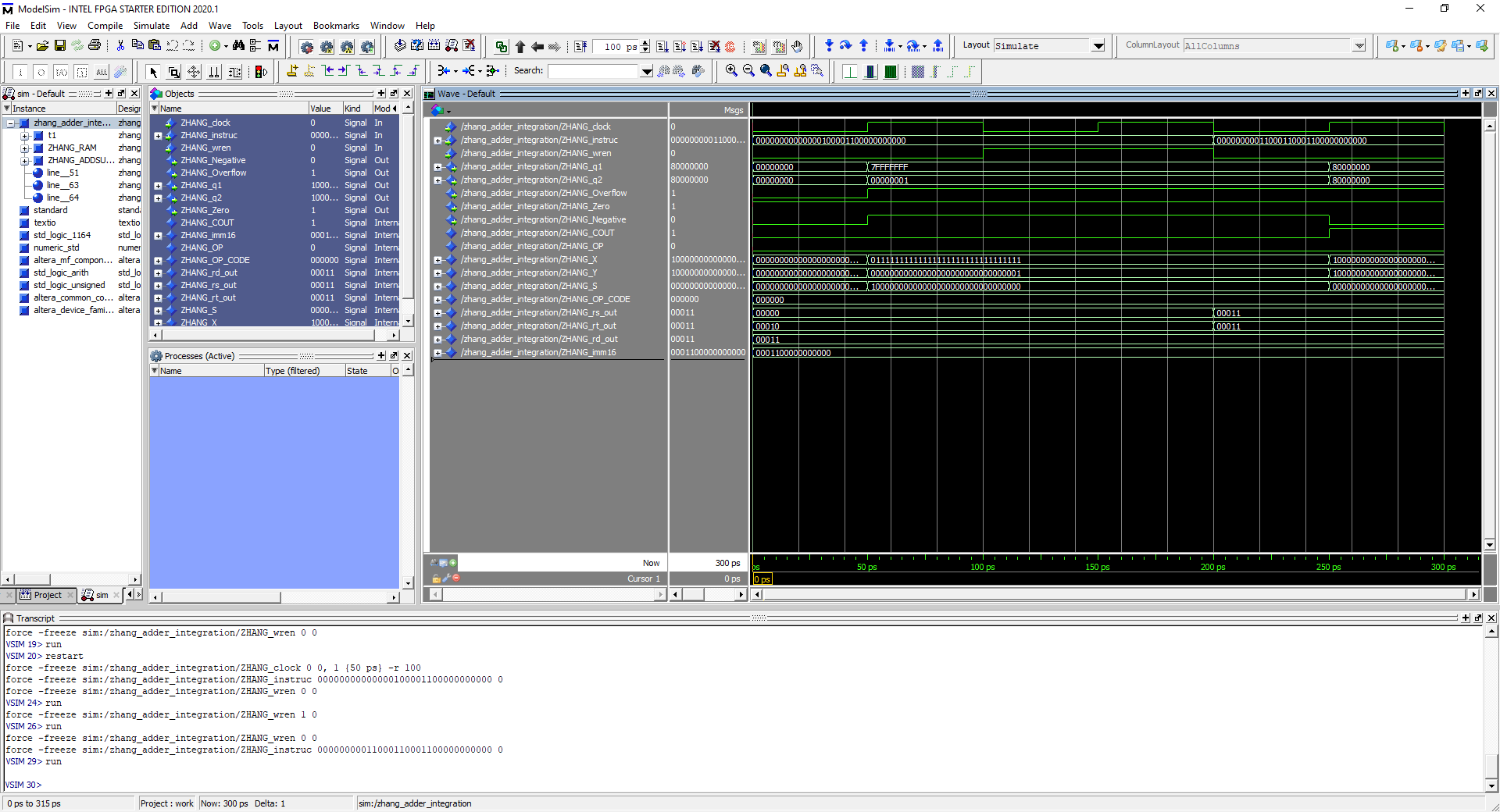
Above showcases the usage of the Instruction File, the 3-port ram and the 32-Bit Adder/Subtractor Unit

**Simulations**

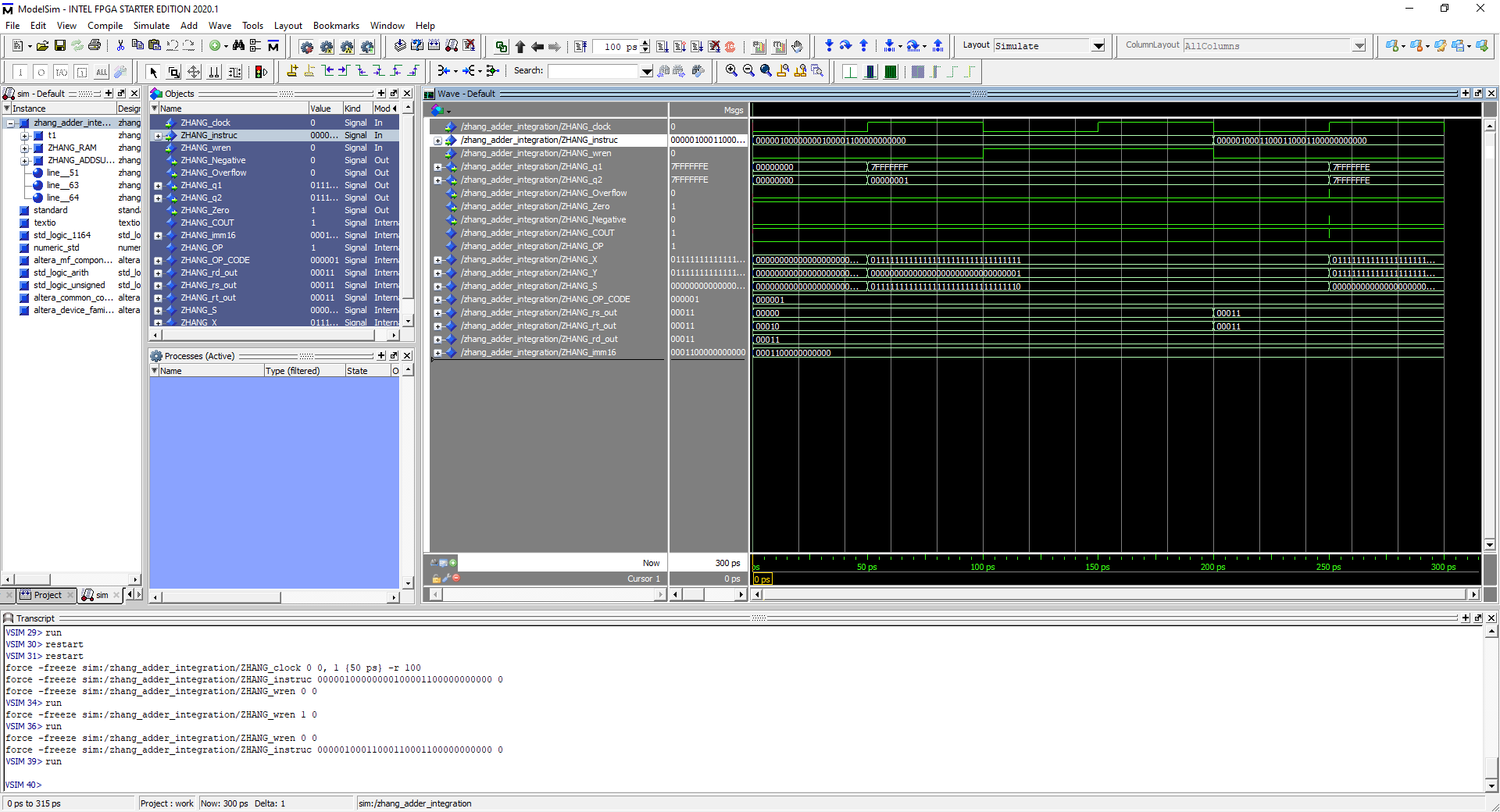
Simulations will be done to verify correctness of the 3-Port RAM Adder/Sub Unit and performed on Modelsim with waveform screenshots and explanations to further verify correctness.

What will be Tested is the following

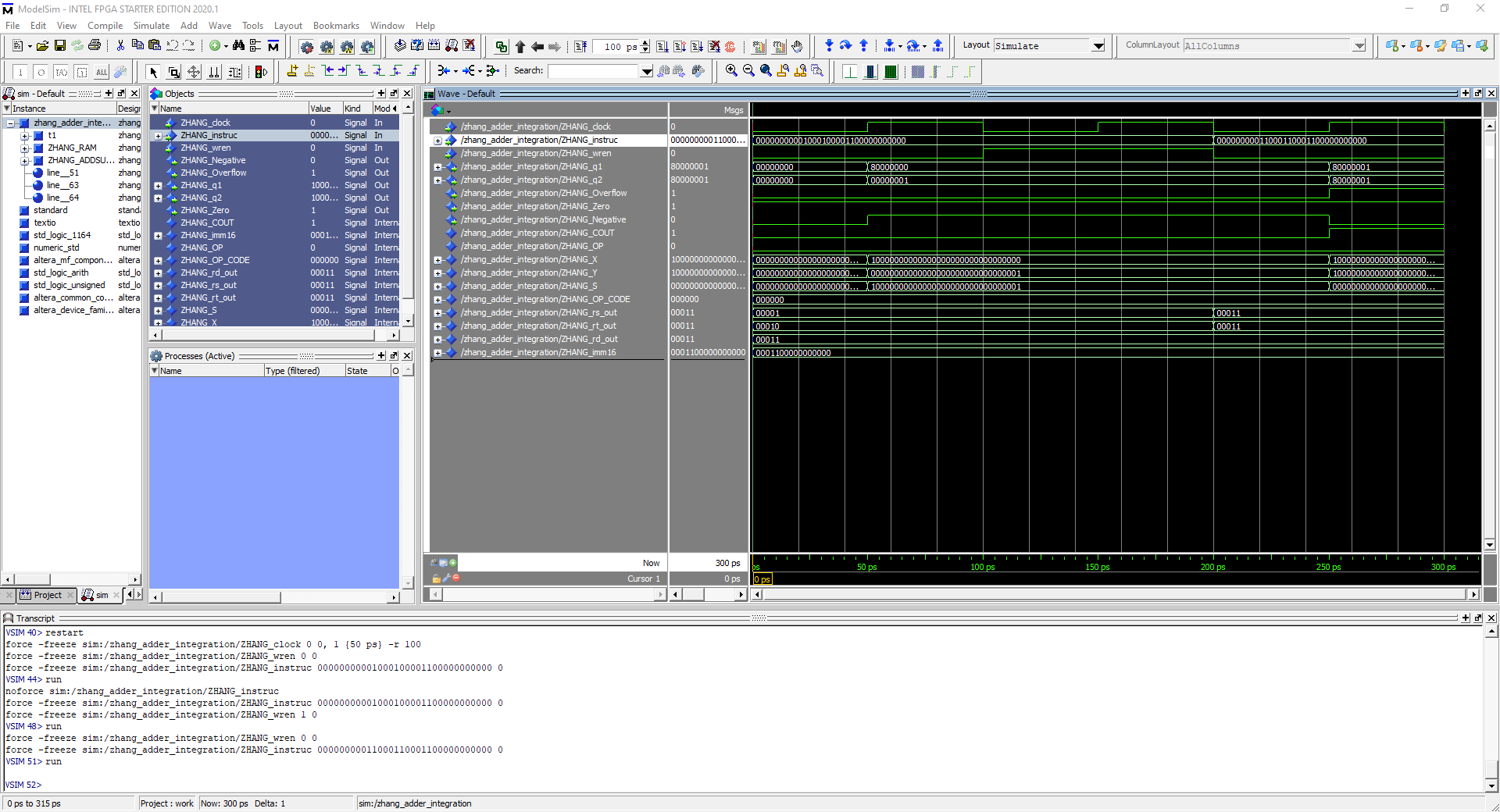
1. **Most Positive 32-Bit Integer + 1**
2. **Most Positive 32-Bit Integer – 1**
3. **Most Negative 32-Bit Integer + 1**
4. **Most Negative 32-Bit Integer – 1**
5. **Most Positive 32-Bit Integer – Most Negative 32-Bit Integer**
6. **Most Positive 32-Bit Integer + Most Positive 32-Bit Integer**
7. **Most positive 32-Bit Integer – Most Positive 32-Bit Integer**



What is performed is the **Most Positive 32-Bit Integer + 1** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000000 00000 00010 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice that the 1st Address and the 3rd Address is being called from the MIF, computed, then written into the 4th Address. Furthermore, Overflow flag and Negative flag has been triggered.



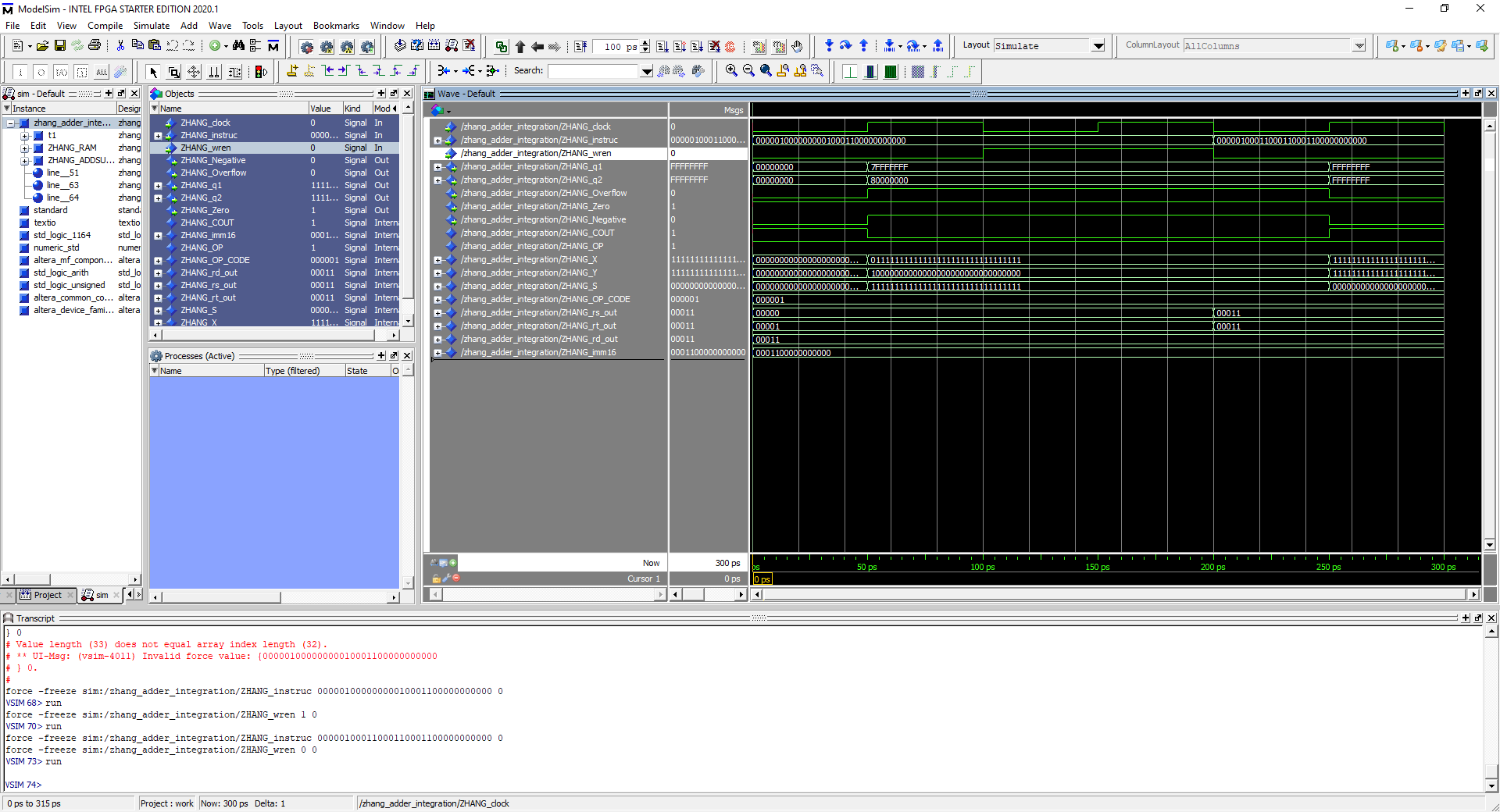
What is performed is the **Most Positive 32-Bit Integer - 1** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000001 00000 00010 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice that the 1st Address and the 3rd Address is being called from the MIF, computed, then written into the 4th Address. Furthermore, zero flag has been triggered



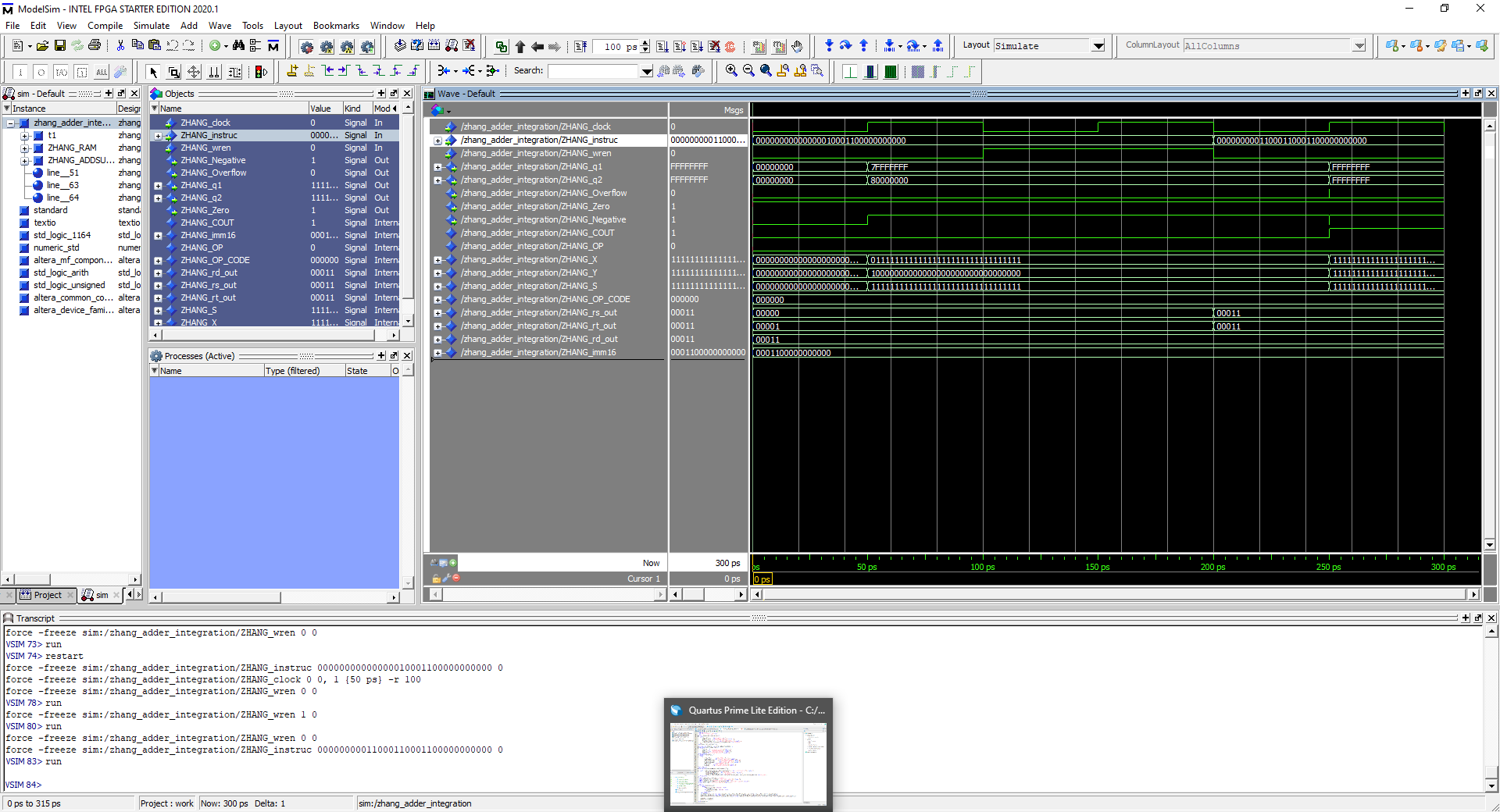
What is performed is the **Most Negative 32-Bit Integer + 1** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000000 00001 00010 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice that the 2nd Address and the 3rd Address is being called from the MIF, computed, then written into the 4th Address. Furthermore, notice the Overflow Flag.

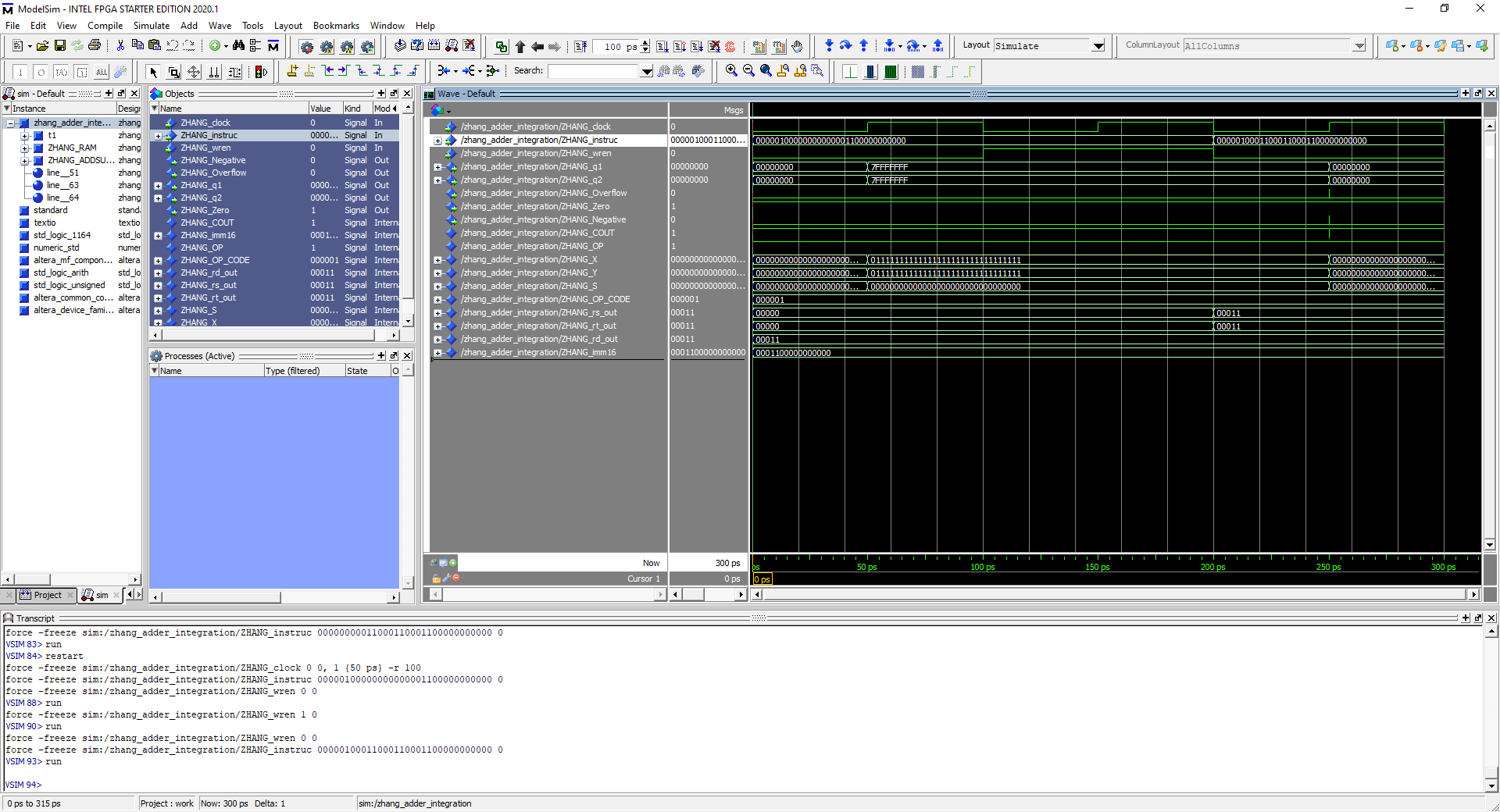
Graphical user interface

Description automatically generated What is performed is the **Most Negative 32-Bit Integer - 1** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000001 00001 00010 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice that the 2nd Address and the 3rd Address is being called from the MIF, computed, then written into the 4th Address.



What is performed is the **Most Positive 32-Bit Integer -** **Most Negative 32-Bit Integer** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000001 00000 00001 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice that the 1st Address and the 2nd Address is being called from the MIF, computed, then written into the 4th Address.

 What is performed is the **Most Positive 32-Bit Integer +** **Most Negative 32-Bit Integer** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000000 00000 00001 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice the 1st Address and the 2nd Address is being called from the MIF, computed, then written into the 4th address. Notice the negative flag being triggered.

 What is performed is the **Most Positive 32-Bit Integer -** **Most Positive 32-Bit Integer** where a 32-bit Instruction is read where once the instruction is broken down they will be inputted into the adder/subtractor where the 3-port ram calls the address from in the MIF File so that the Mathematical process can be performed then stored. Instruction bit is 000001 00000 00000 00011 00000000000, OP code, RS, RT, RD, Function respectively. Notice the Zero flag being triggered.

**Conclusion**

By doing this laboratory assignment, I was able to get familiar with the usage of the MIF file and how it works. I also learned more about how Register block works as well as how things are read and written. By the end of this lab, I have gained much knowledge on the importance of the usage of RAM as well as how to integrate units into it.