BIRZEIT UNIVERSITY

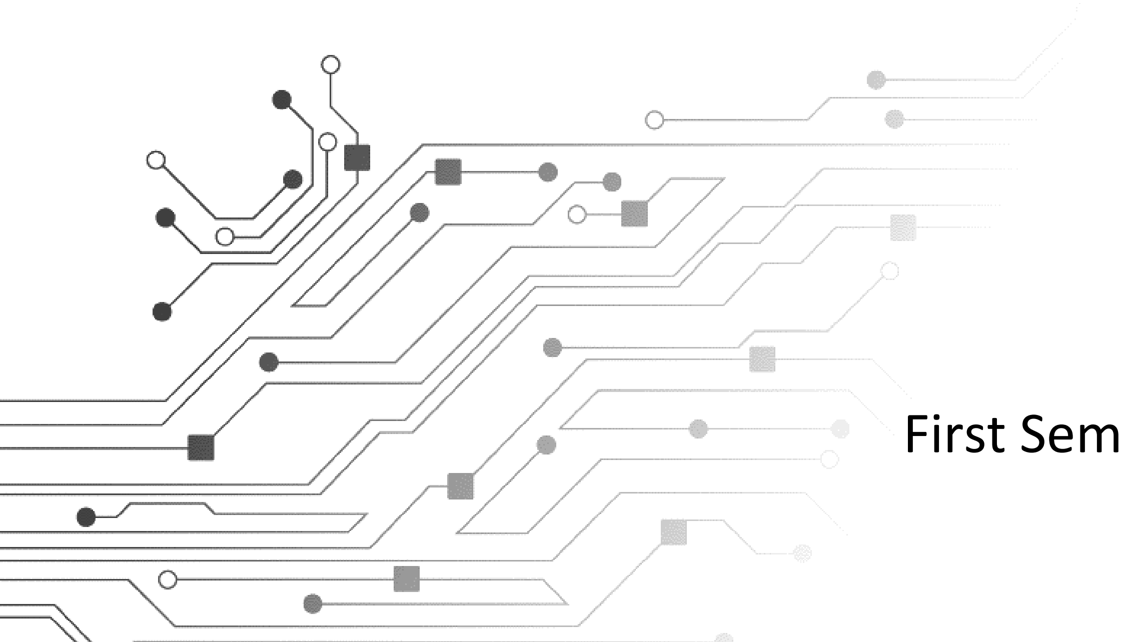
First Semester, 2022/2023

Digital Systems ENCS234 – **Verilog Project**

Name: Mohammed Abed Alkareem

ID: 1210708

Instructor: Dr. Bilal Karaki



First Semester (2022-2023)

a) Specify the size of the output (O) in bits so the overflow can never occur.

Based on the operations that are represented:

The operation that may requires most extra bits so that overflow does not occur is at **C = 001**
Because when adding or subtracting two n-bit numbers with same sign may produce a final carry that it has to be stored in an extra bit, and from multiplying the number by two this requires also another extra bit
➔**So the output has to be n+2 bits**

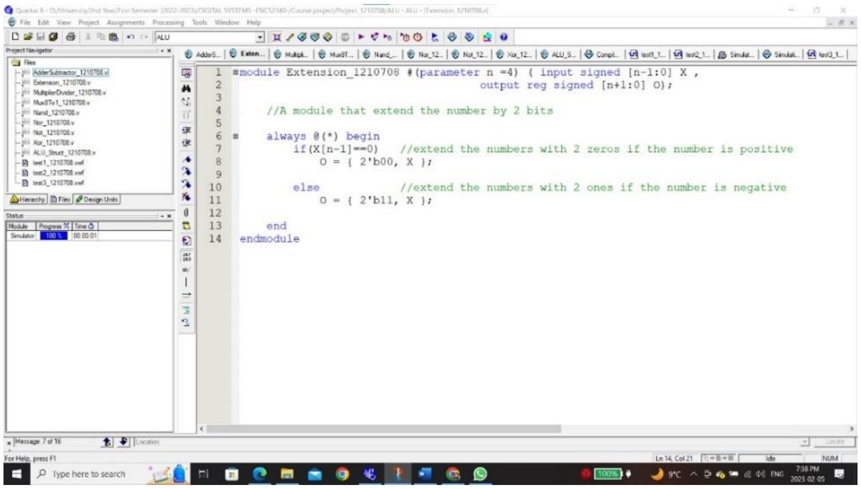
b) Show the ALU implementation using medium-scale integration (MSI) components and minimum number of gates (i.e., in blocks with their sizes). Note that, you might use some kind of extension (sign- or zero-extension).

For implementing this ALU we need:

1. Sign- Or Zero-Extension Module.
2. Adder / Subtractor Module.
3. Multiplier / Divider Module.
4. 8 to 1 Multiplexer Module.
5. Nand Module.
6. Not Module.
7. Nor Module
8. Xor Module.

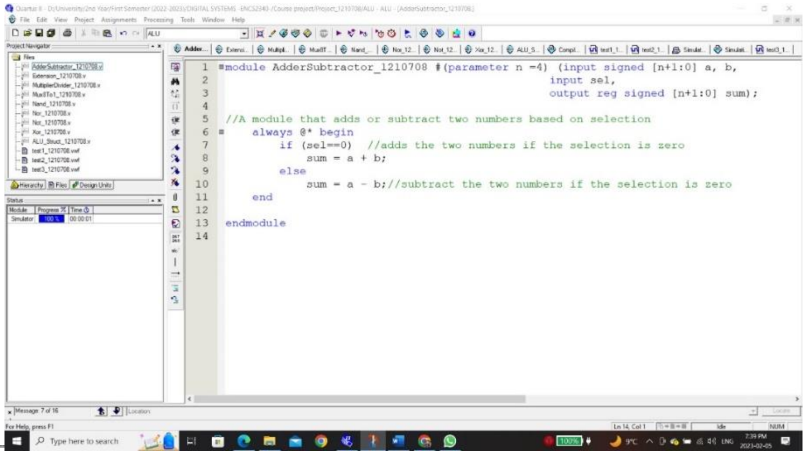
c) Write behavioral Verilog modules for your elements you defined in Part (b). Be noted that the size of every element you define should be parameterized, so that you can vary the design during the testing phase.

1. Sign- Or Zero-Extension Module.



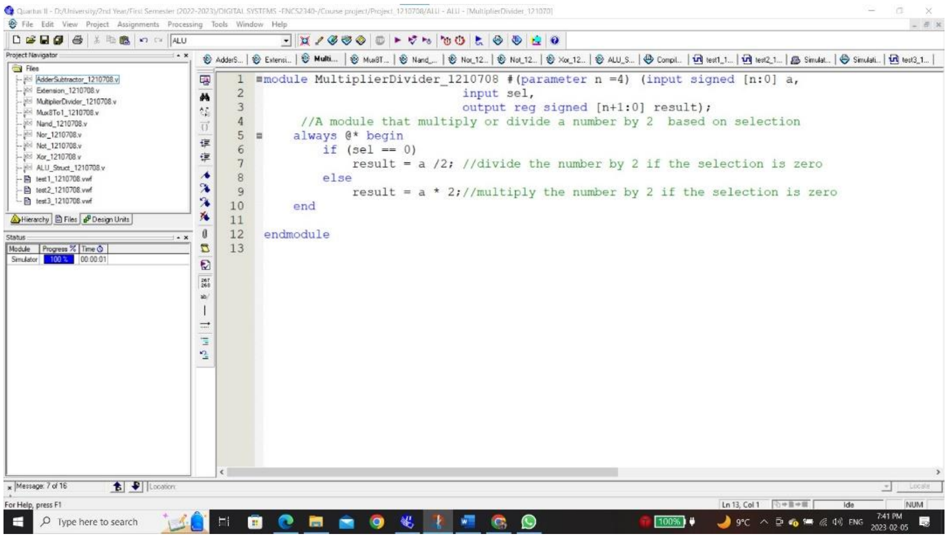
```
module Extension_1210708 #(parameter n =4) ( input signed [n-1:0] X ,
                                          output reg signed [n+1:0] O);

    //A module that extend the number by 2 bits

    always @(*) begin
        if(X[n-1]==0)   //extend the numbers with 2 zeros if the number is positive
            O = { 2'b00, X };

        else            //extend the numbers with 2 ones if the number is negative
            O = { 2'b11, X };

    end
endmodule
```

2. Adder / Subtractor Module.



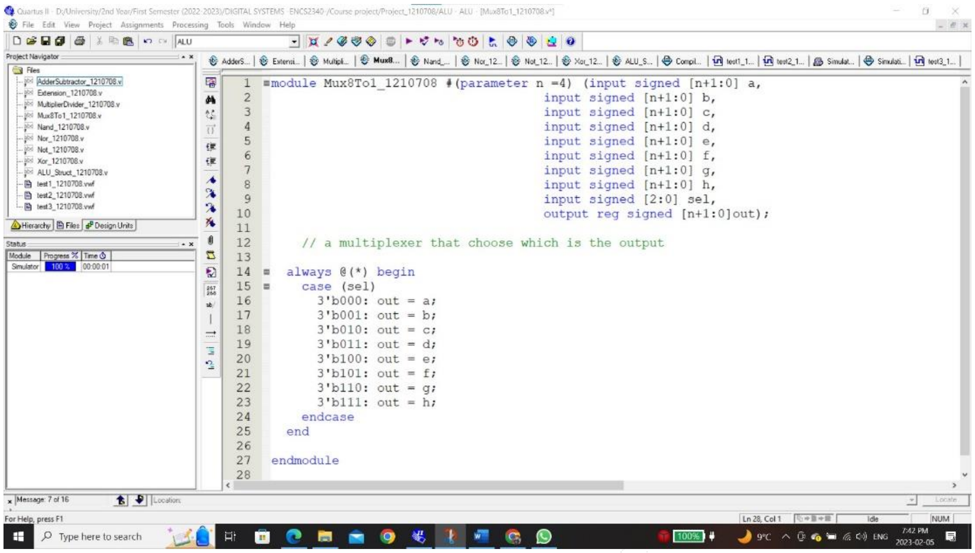
```
module AdderSubtractor_1210708 #(parameter n =4) (input signed [n+1:0] a, b,
                                                  input sel,
                                                  output reg signed [n+1:0] sum);

 //A module that adds or subtract two numbers based on selection
     always @* begin
         if (sel==0)  //adds the two numbers if the selection is zero
             sum = a + b;
         else
             sum = a - b;//subtract the two numbers if the selection is zero
     end

 endmodule
```

3. Multiplier / Divider Module.



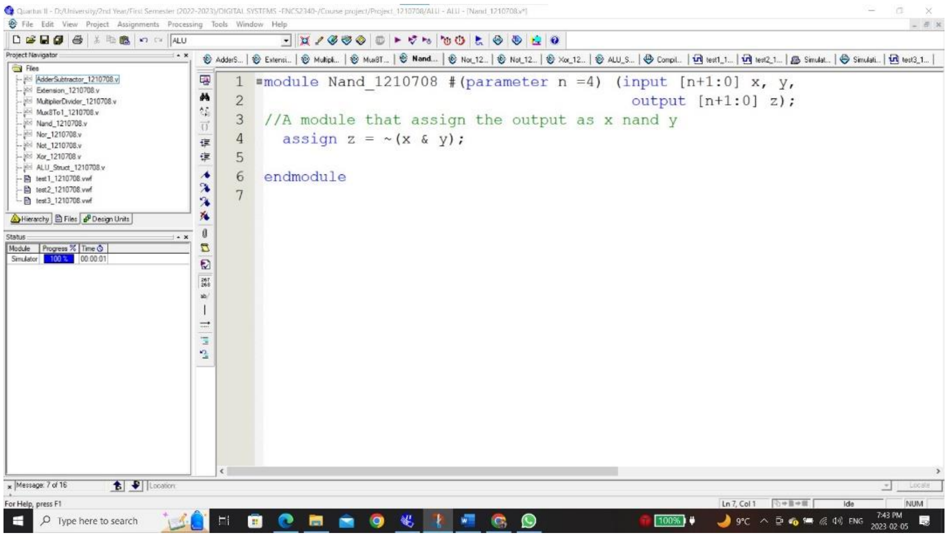
```
module MultiplierDivider_1210708 #(parameter n =4) (input signed [n:0] a,
                                    input sel,
                                    output reg signed [n+1:0] result);
    //A module that multiply or divide a number by 2  based on selection
   always @* begin
       if (sel == 0)
           result = a /2; //divide the number by 2 if the selection is zero
       else
           result = a * 2;//multiply the number by 2 if the selection is zero
   end

endmodule
```

4. 8 to 1 Multiplexer Module.



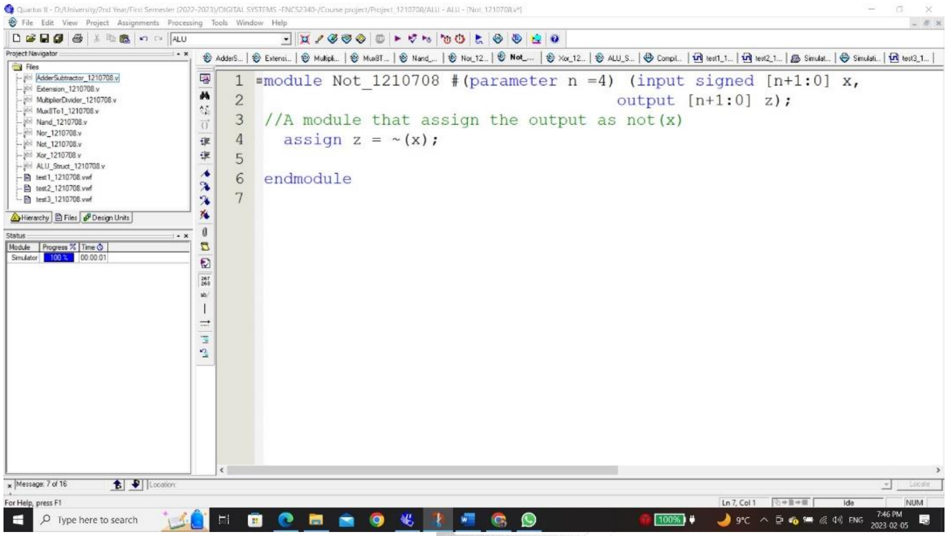
```
module Mux8To1_1210708 #(parameter n =4) (input signed [n+1:0] a,
                                          input signed [n+1:0] b,
                                          input signed [n+1:0] c,
                                          input signed [n+1:0] d,
                                          input signed [n+1:0] e,
                                          input signed [n+1:0] f,
                                          input signed [n+1:0] g,
                                          input signed [n+1:0] h,
                                          input signed [2:0] sel,
                                          output reg signed [n+1:0]out);

    // a multiplexer that choose which is the output

  always @(*) begin
    case (sel)
      3'b000: out = a;
      3'b001: out = b;
      3'b010: out = c;
      3'b011: out = d;
      3'b100: out = e;
      3'b101: out = f;
      3'b110: out = g;
      3'b111: out = h;
    endcase
  end

endmodule
```

5. Nand Module.



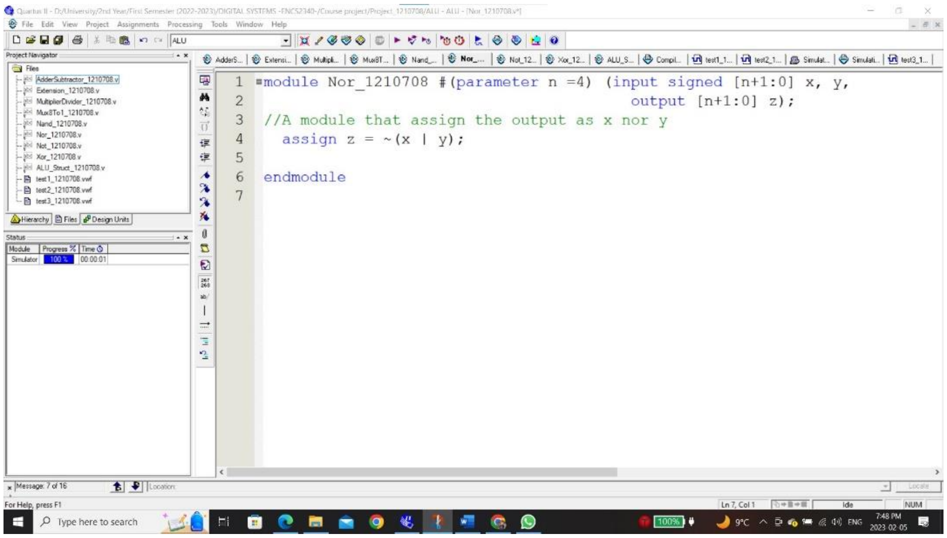
```
module Nand_1210708 #(parameter n =4) (input [n+1:0] x, y,
                                        output [n+1:0] z);
//A module that assign the output as x nand y
  assign z = ~(x & y);

endmodule
```

6. Not Module.



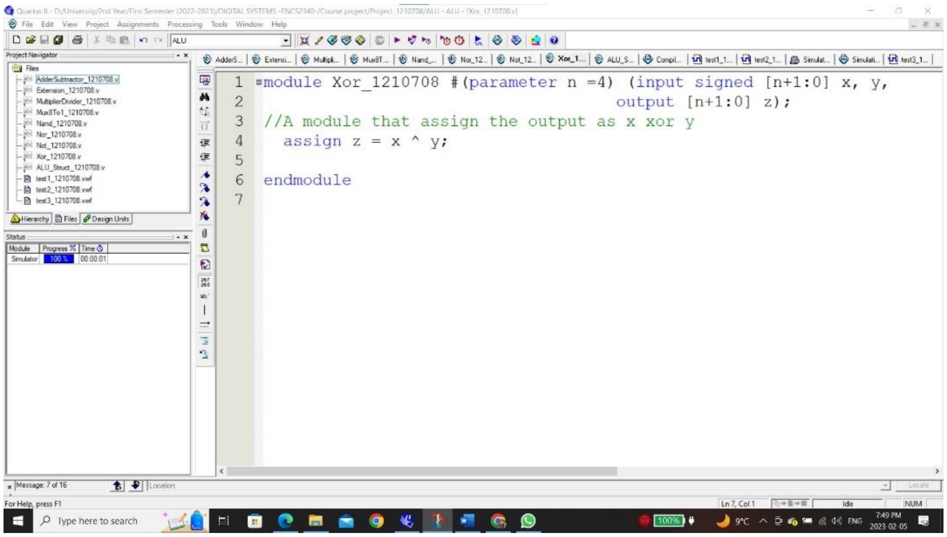
```
module Not_1210708 #(parameter n =4) (input signed [n+1:0] x,
                                      output [n+1:0] z);
//A module that assign the output as not(x)
  assign z = ~(x);

endmodule
```

7. Nor Module



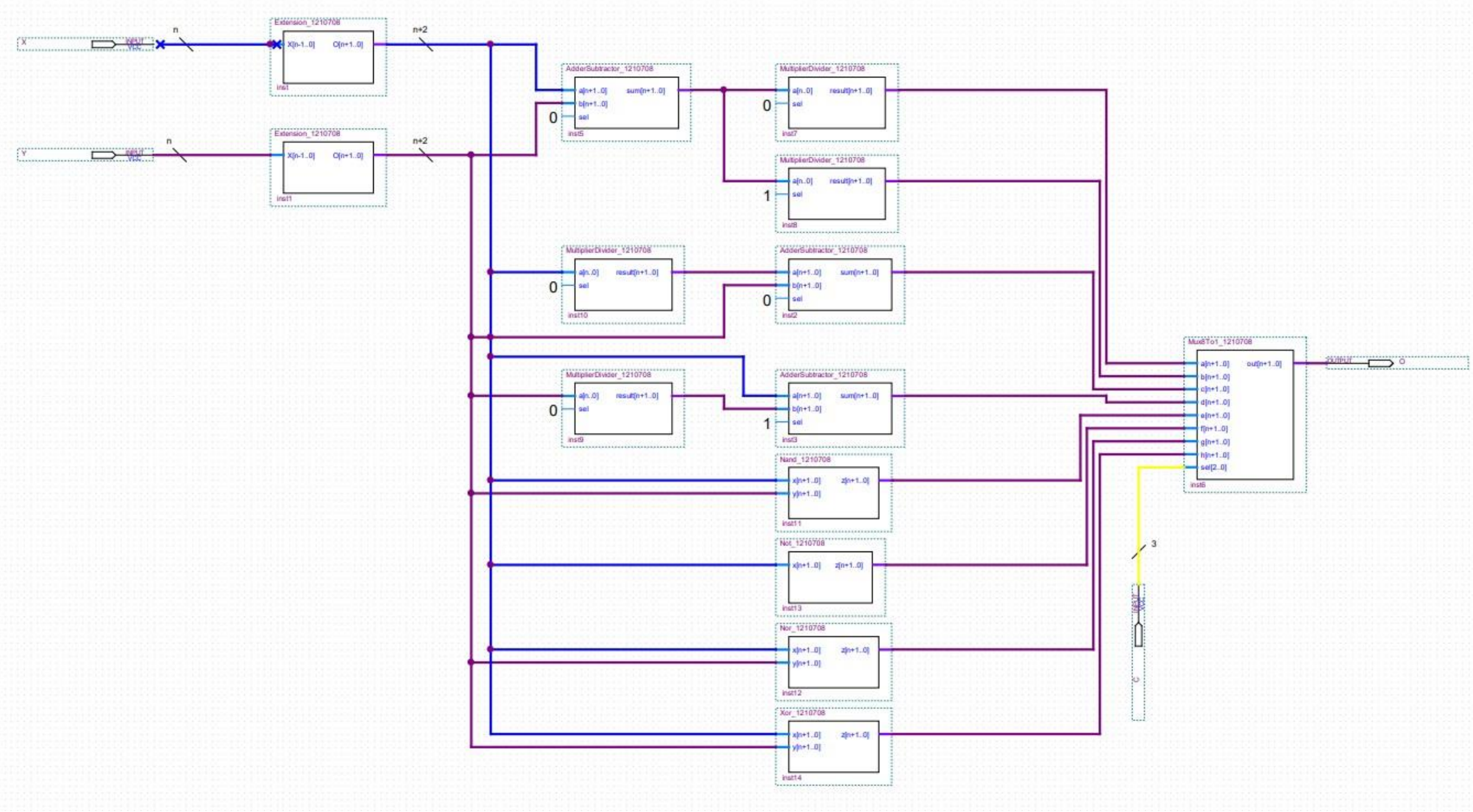
```
module Nor_1210708 #(parameter n =4) (input signed [n+1:0] x, y,
                                         output [n+1:0] z);
//A module that assign the output as x nor y
  assign z = ~(x | y);

endmodule
```

8. Xor Module.

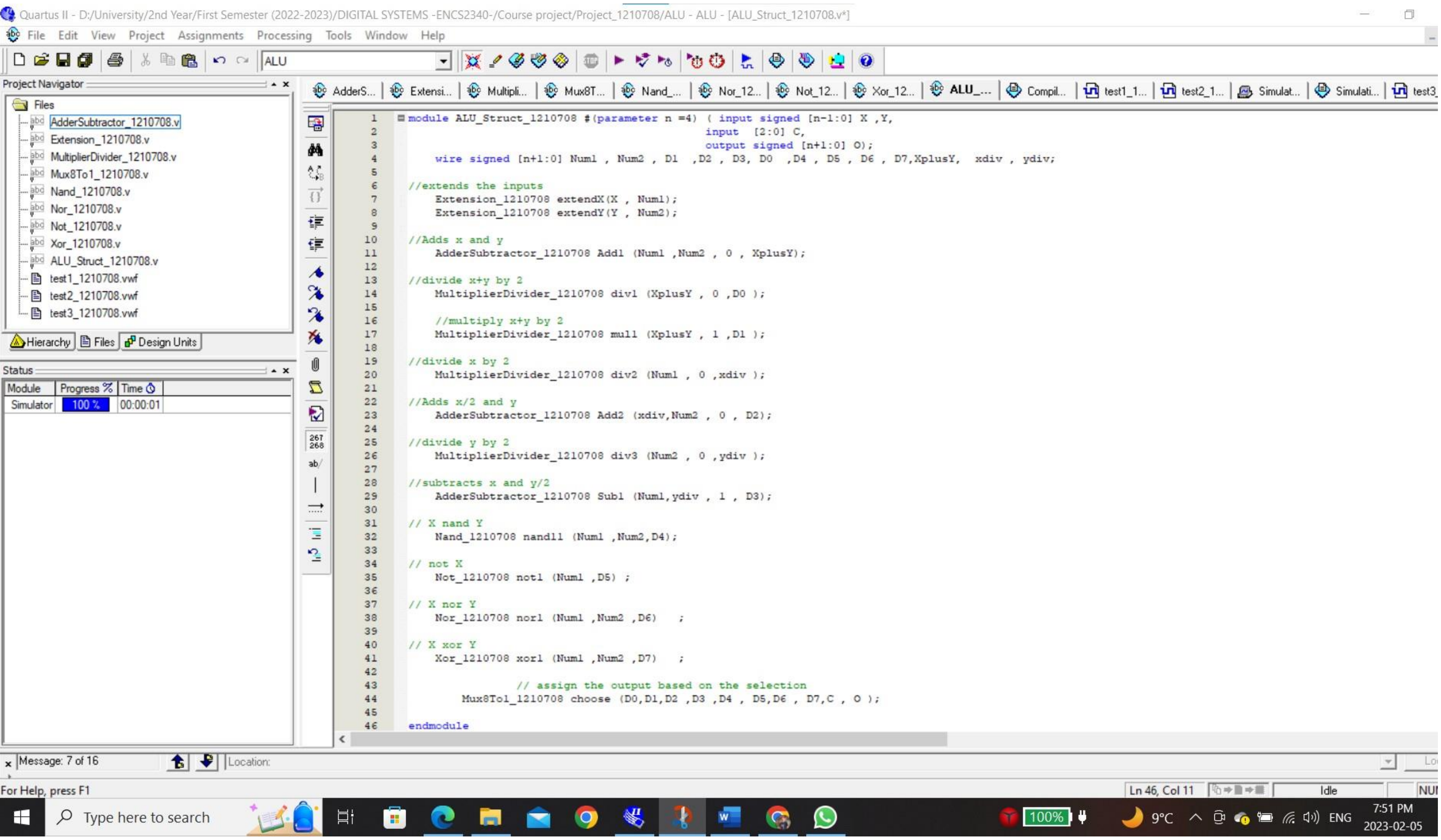


```
module Xor_1210708 #(parameter n =4) (input signed [n+1:0] x, y,
                                    output [n+1:0] z);
//A module that assign the output as x xor y
  assign z = x ^ y;

endmodule
```

The Block Diagram For the ALU:



d) Write a structural Verilog model for your ALU designed in Part (b) using the elements you defined in Part (c).



```
module ALU_Struct_1210708 #(parameter n =4) ( input signed [n-1:0] X ,Y,
                                            input  [2:0] C,
                                            output signed [n+1:0] O);
    wire signed [n+1:0] Num1 , Num2 , D1  ,D2 , D3, D0  ,D4 , D5 , D6 , D7,XplusY,  xdiv , ydiv;

//extends the inputs
    Extension_1210708 extendX(X , Num1);
    Extension_1210708 extendY(Y , Num2);

//Adds x and y
    AdderSubtractor_1210708 Add1 (Num1 ,Num2 , 0 , XplusY);

//divide x+y by 2
    MultiplierDivider_1210708 div1 (XplusY , 0 ,D0 );

    //multiply x+y by 2
    MultiplierDivider_1210708 mul1 (XplusY , 1 ,D1 );

//divide x by 2
    MultiplierDivider_1210708 div2 (Num1 , 0 ,xdiv );

//Adds x/2 and y
    AdderSubtractor_1210708 Add2 (xdiv,Num2 , 0 , D2);

//divide y by 2
    MultiplierDivider_1210708 div3 (Num2 , 0 ,ydiv );

//subtracts x and y/2
    AdderSubtractor_1210708 Sub1 (Num1,ydiv , 1 , D3);

// X nand Y
    Nand_1210708 nand11 (Num1 ,Num2,D4);

// not X
    Not_1210708 not1 (Num1 ,D5) ;

// X nor Y
    Nor_1210708 nor1 (Num1 ,Num2 ,D6)   ;

// X xor Y
    Xor_1210708 xor1 (Num1 ,Num2 ,D7)   ;

                // assign the output based on the selection
        Mux8To1_1210708 choose (D0,D1,D2 ,D3 ,D4 , D5,D6 , D7,C , O );

endmodule
```

e) Generate the waveforms of the ALU defined in Part (d), assumes that X and Y are 4-bits and their values based on your student ID should be set as follows:

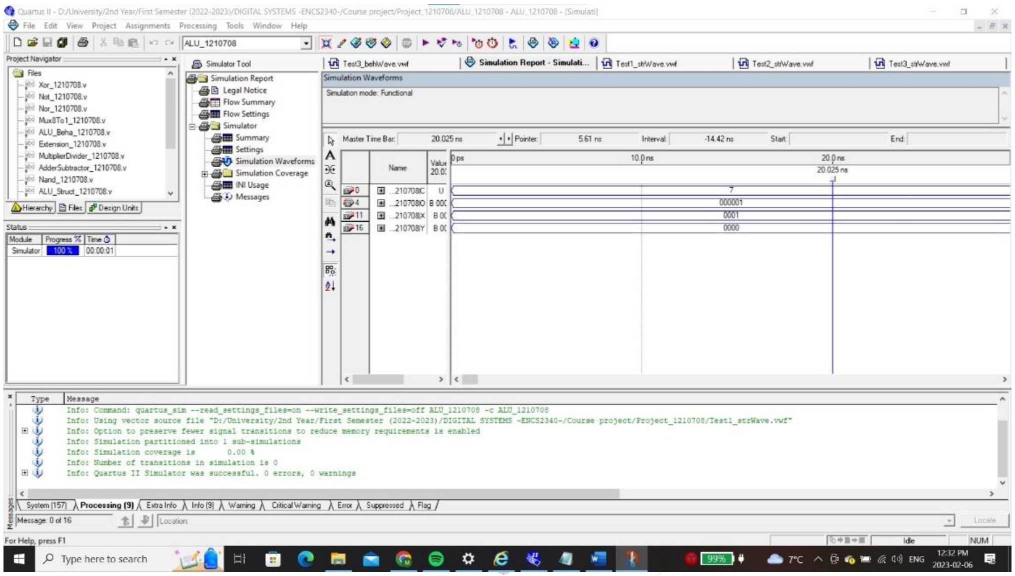
The general representation of the student ID is 1C2Y2X2C1Y1X1:

My ID: 1 2 1 0 7 0 ~~8~~ 1

1 C2 Y2 X2 C1 Y1 X1

Test	X	Y	C	O
1	X1 = 1	Y1 = 0	C1 = 7	(1) XOR (0) = 1
2	X2 = 0	Y2 = 1	C2 = 2	(0/2)+1 = 1
3	X3 = -X1= -1	Y3 = -Y1= 0	C3 = C2 = 2	(-1/0)+0 = 0

Test 1 Wave Form.

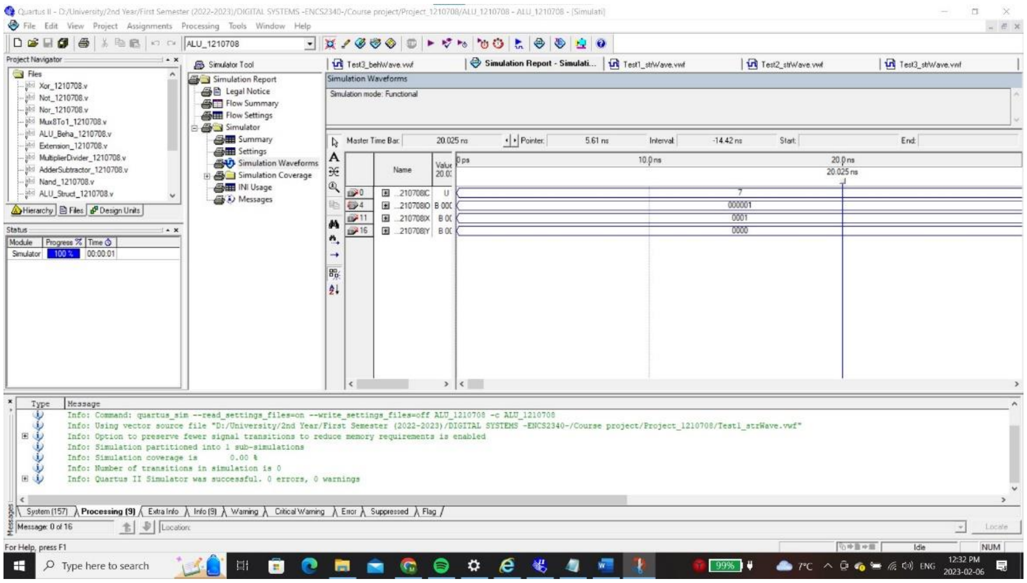


Simulation Waveforms

Simulation mode: Functional

```
Info: Command: quartus_sim --read_settings_files=on --write_settings_files=off ALU_1210708 -c ALU_1210708
Info: Using vector source file "D:/University/2nd Year/First Semester (2022-2023)/DIGITAL SYSTEMS -ENCS2340-/Course project/Project_1210708/Test1_strWave.vwf"
Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
Info: Simulation partitioned into 1 sub-simulations
Info: Simulation coverage is      0.00 %
Info: Number of transitions in simulation is 0
Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

Test 2 Wave Form.



Simulation Waveforms

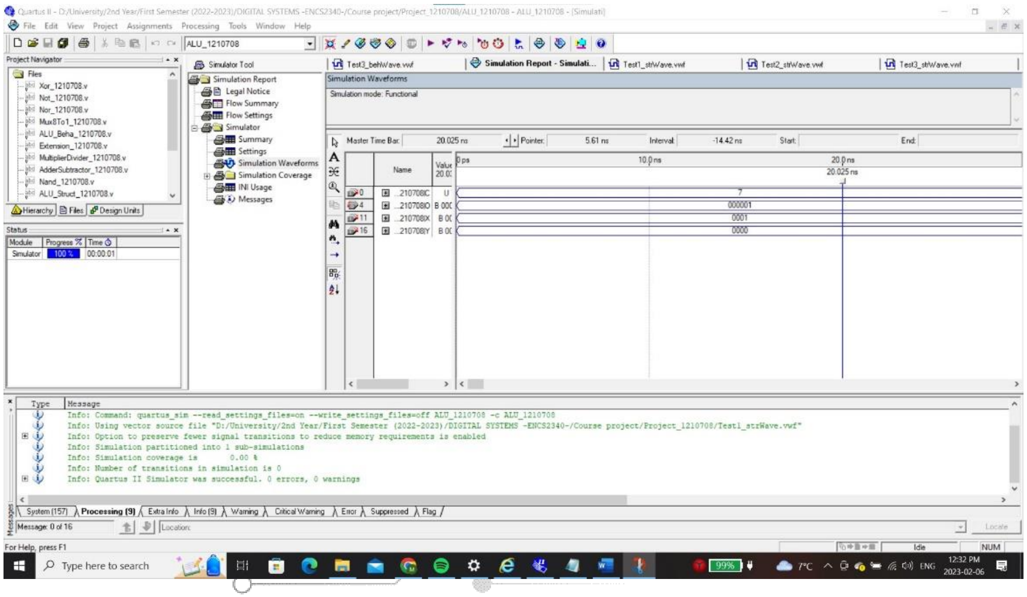
Simulation mode: Functional

Master Time Bar: 20.025 ns Pointer: 5.61 ns Interval: -14.42 ns Start: End:

```
0   ...210708C   U      7
4   ...210708O   B 000  000001
11  ...210708X   B 00   0001
16  ...210708Y   B 00   0000
```

```
Type  Message
      Info: Command: quartus_sim --read_settings_files=on --write_settings_files=off ALU_1210708 -c ALU_1210708
      Info: Using vector source file "D:/University/2nd Year/First Semester (2022-2023)/DIGITAL SYSTEMS -ENCS2340-/Course project/Project_1210708/Test1_strWave.vwf"
      Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
      Info: Simulation partitioned into 1 sub-simulations
      Info: Simulation coverage is      0.00 %
      Info: Number of transitions in simulation is 0
      Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

Test 3 Wave Form.



Simulation Waveforms

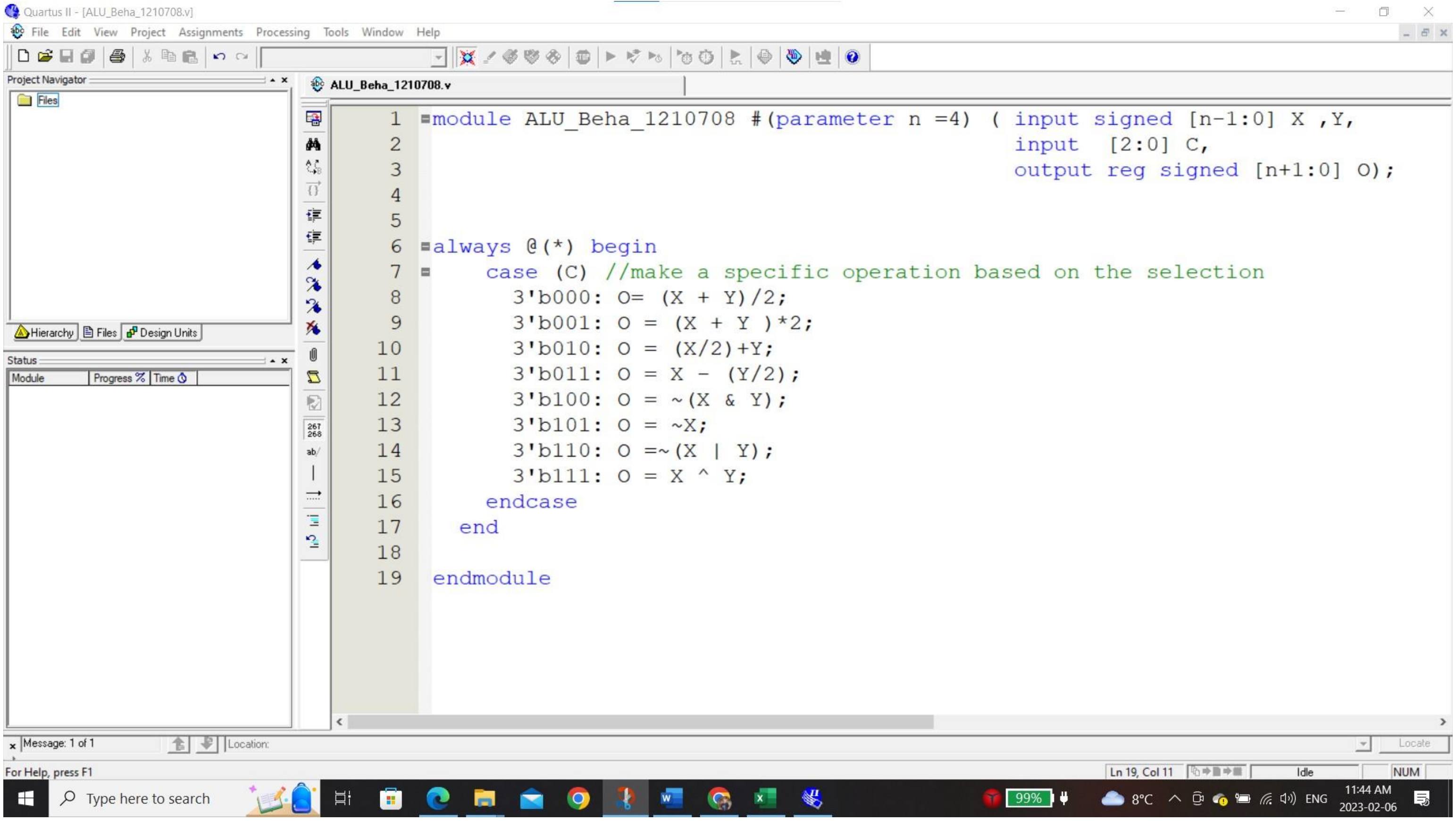
Simulation mode: Functional

Master Time Bar: 20.025 ns Pointer: 5.61 ns Interval: -14.42 ns Start: End:

```
0   ...210708C   U      7
4   ...210708O   B 000  000001
11  ...210708X   B 00   0001
16  ...210708Y   B 00   0000
```

```
Type  Message
      Info: Command: quartus_sim --read_settings_files=on --write_settings_files=off ALU_1210708 -c ALU_1210708
      Info: Using vector source file "D:/University/2nd Year/First Semester (2022-2023)/DIGITAL SYSTEMS -ENCS2340-/Course project/Project_1210708/Test1_strWave.vwf"
      Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
      Info: Simulation partitioned into 1 sub-simulations
      Info: Simulation coverage is      0.00 %
      Info: Number of transitions in simulation is 0
      Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

f) Write a single behavioral Verilog module that models the designed ALU.



```verilog
module ALU_Beha_1210708 #(parameter n =4) ( input signed [n-1:0] X ,Y,
                                            input  [2:0] C,
                                            output reg signed [n+1:0] O);


always @(*) begin
    case (C) //make a specific operation based on the selection
      3'b000: O= (X + Y)/2;
      3'b001: O = (X + Y )*2;
      3'b010: O = (X/2)+Y;
      3'b011: O = X - (Y/2);
      3'b100: O = ~(X & Y);
      3'b101: O = ~X;
      3'b110: O =~(X | Y);
      3'b111: O = X ^ Y;
    endcase
  end

endmodule
```

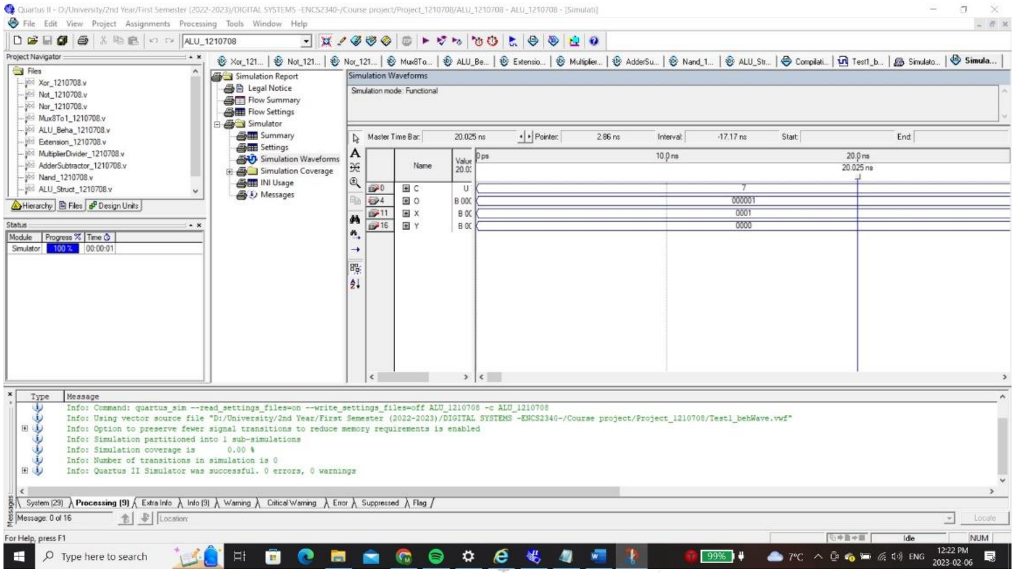
g) Generate the waveforms of the behavioral ALU defined in Part (f), assumes that X and Y are 4-bits and their values based on your student ID should be set as follows:

The general representation of the student ID is 1C2Y2X2C1Y1X1:

My ID:	1	2	1	0	7	0	~~8~~ 1
	1	C2	Y2	X2	C1	Y1	X1

Test	X	Y	C	O
1	X1 = 1	Y1 = 0	C1 = 7	(1) XOR (0) = 1
2	X2 = 0	Y2 = 1	C2 = 2	(0/2)+1 = 1
3	X3 = -X1= -1	Y3 = -Y1= 0	C3 = C2 = 2	(-1/0)+0 = 0

Test 1 Wave Form.



Simulation Waveforms

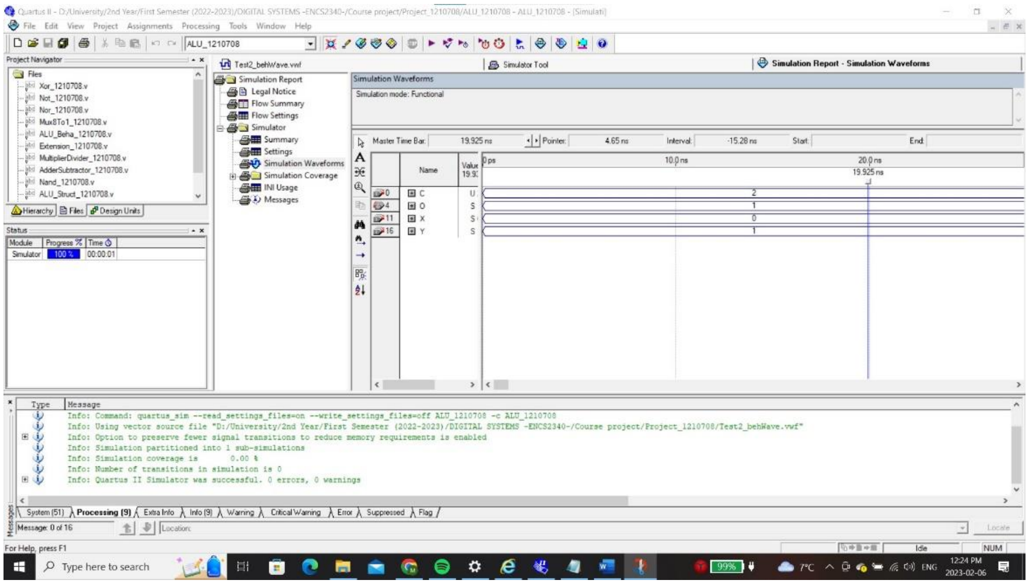
Simulation mode: Functional

Master Time Bar: 20.025 ns Pointer: 2.86 ns Interval: -17.17 ns Start: End:

	Name	Value 20.0	
0	C	U	7
4	O	B 000	000001
11	X	B 00	0001
16	Y	B 00	0000

```
Info: Command: quartus_sim --read_settings_files=on --write_settings_files=off ALU_1210708 -c ALU_1210708
Info: Using vector source file "D:/University/2nd Year/First Semester (2022-2023)/DIGITAL SYSTEMS -ENCS2340-/Course project/Project_1210708/Test1_behWave.vwf"
Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
Info: Simulation partitioned into 1 sub-simulations
Info: Simulation coverage is      0.00 %
Info: Number of transitions in simulation is 0
Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

Test 2 Wave Form.

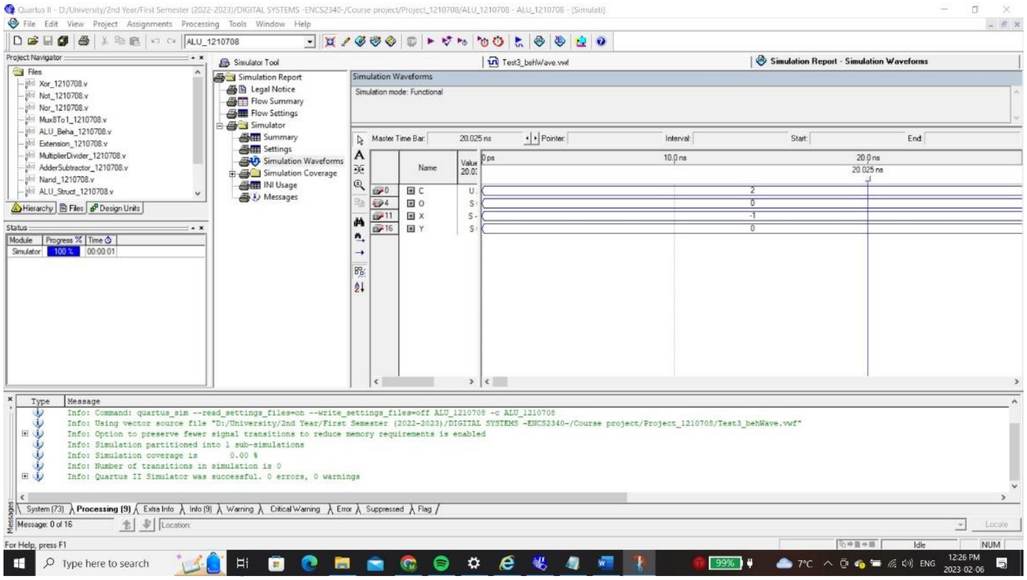


Simulation mode: Functional

Master Time Bar: 19.925 ns   Pointer: 4.65 ns   Interval: -15.28 ns   Start: End:

```
Info: Command: quartus_sim --read_settings_files=on --write_settings_files=off ALU_1210708 -c ALU_1210708
Info: Using vector source file "D:/University/2nd Year/First Semester (2022-2023)/DIGITAL SYSTEMS -ENCS2340-/Course project/Project_1210708/Test2_behWave.vwf"
Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
Info: Simulation partitioned into 1 sub-simulations
Info: Simulation coverage is      0.00 %
Info: Number of transitions in simulation is 0
Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

Test 3 Wave Form.



Simulation mode: Functional

Master Time Bar: 20.025 ns   Pointer: Interval: Start: End:

```
Info: Command: quartus_sim --read_settings_files=on --write_settings_files=off ALU_1210708 -c ALU_1210708
Info: Using vector source file "D:/University/2nd Year/First Semester (2022-2023)/DIGITAL SYSTEMS -ENCS2340-/Course project/Project_1210708/Test3_behWave.vwf"
Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
Info: Simulation partitioned into 1 sub-simulations
Info: Simulation coverage is      0.00 %
Info: Number of transitions in simulation is 0
Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```