ASSIGNMENT-1

DIGITAL SYSTEM DESIGN

(EEU33C07)

SUBMITTED TO:

Dr. Shreejith Shanker

SUBMITTED BY:

Prachi Singhroha (Student ID: 21355131)

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Declaration concerning plagiarism

I have read and I understand the plagiarism provisions in the General Regulations of the *University Calendar* for the current year, found at http://www.tcd.ie/calendar

I have completed the Online Tutorial in avoiding plagiarism 'Ready, Steady, Write', located at http://tcd-ie.libguides.com/ plagiarism/ready-steady-write

STUDENT NUMBER: 21355131

SIGNED:

DATE: 30.10.2021

WAVEFORMS

Following are the waveforms obtained when inputs are

A = 6’b000011

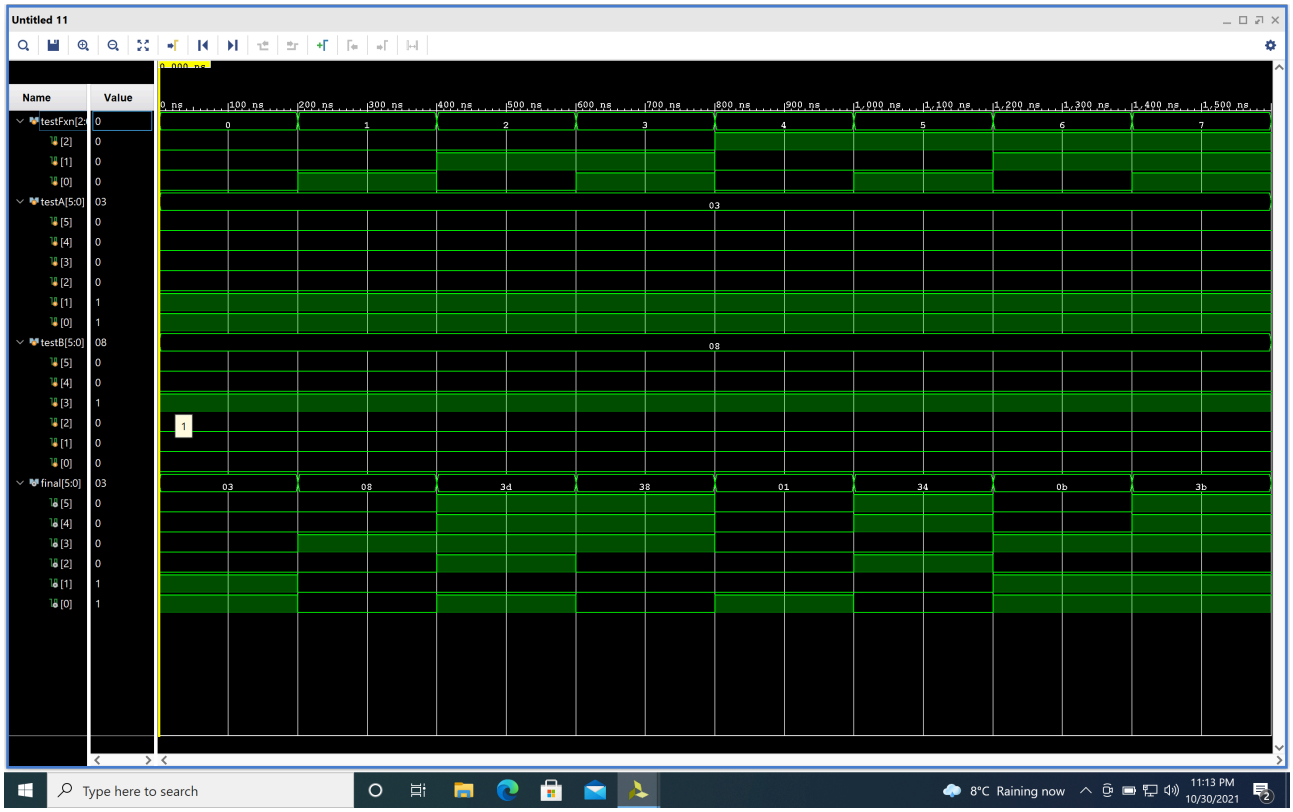
B = 6’b000100

Fxn as follows:

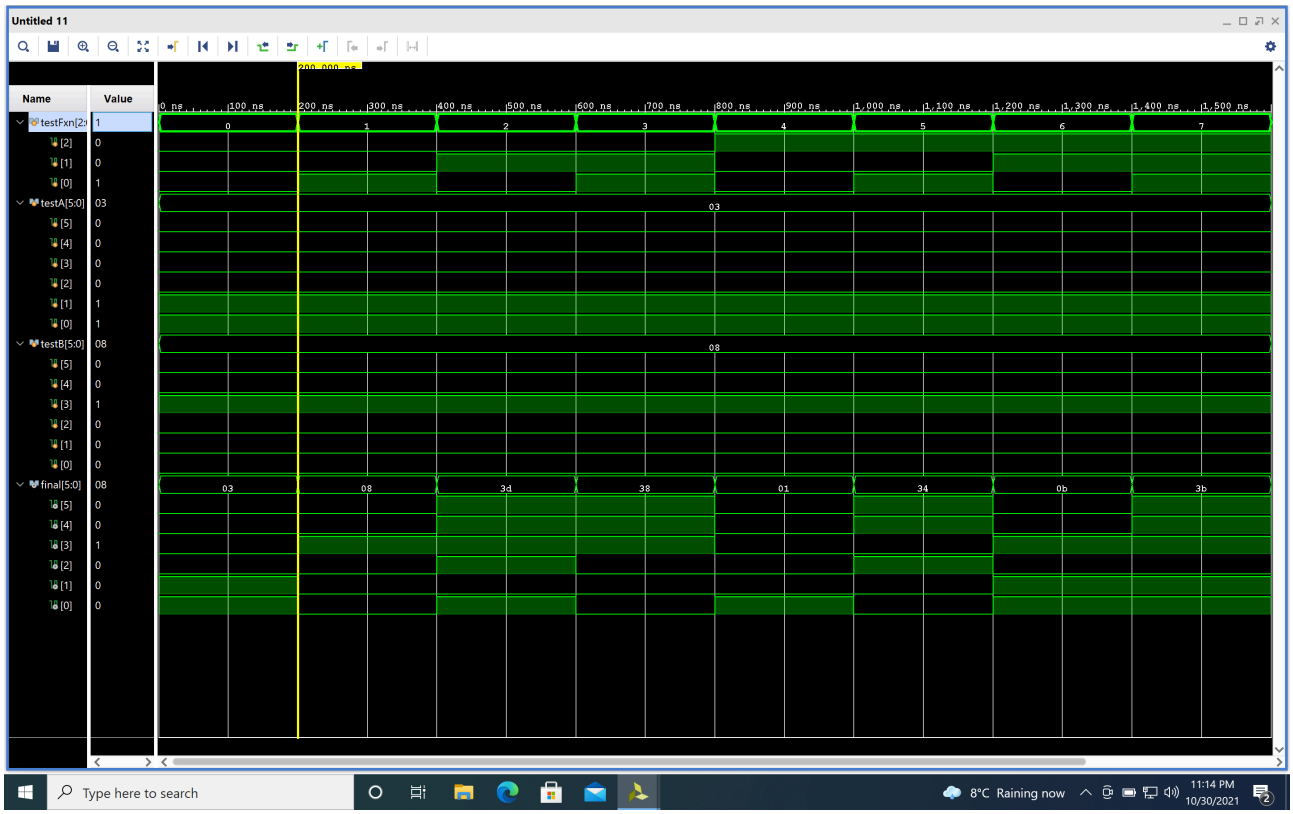
fxn	X[5:0]
000	A
001	B
010	-A
011	-B
100	A<B (is A less than B)
101	(A *nxor* B) (Bitwise XNOR)
110	A+B
111	A-B

(Output is *final* in the following screenshots)

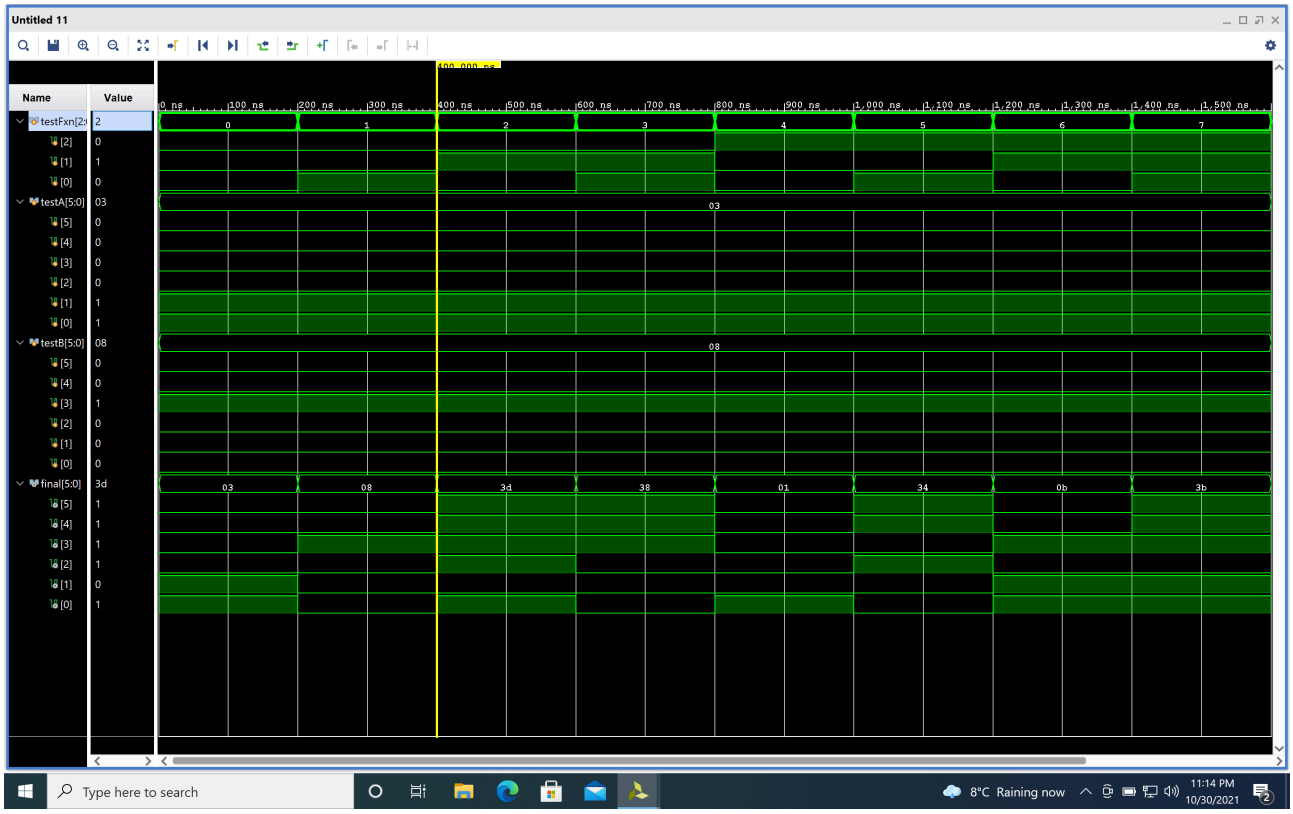
Fxn = 000



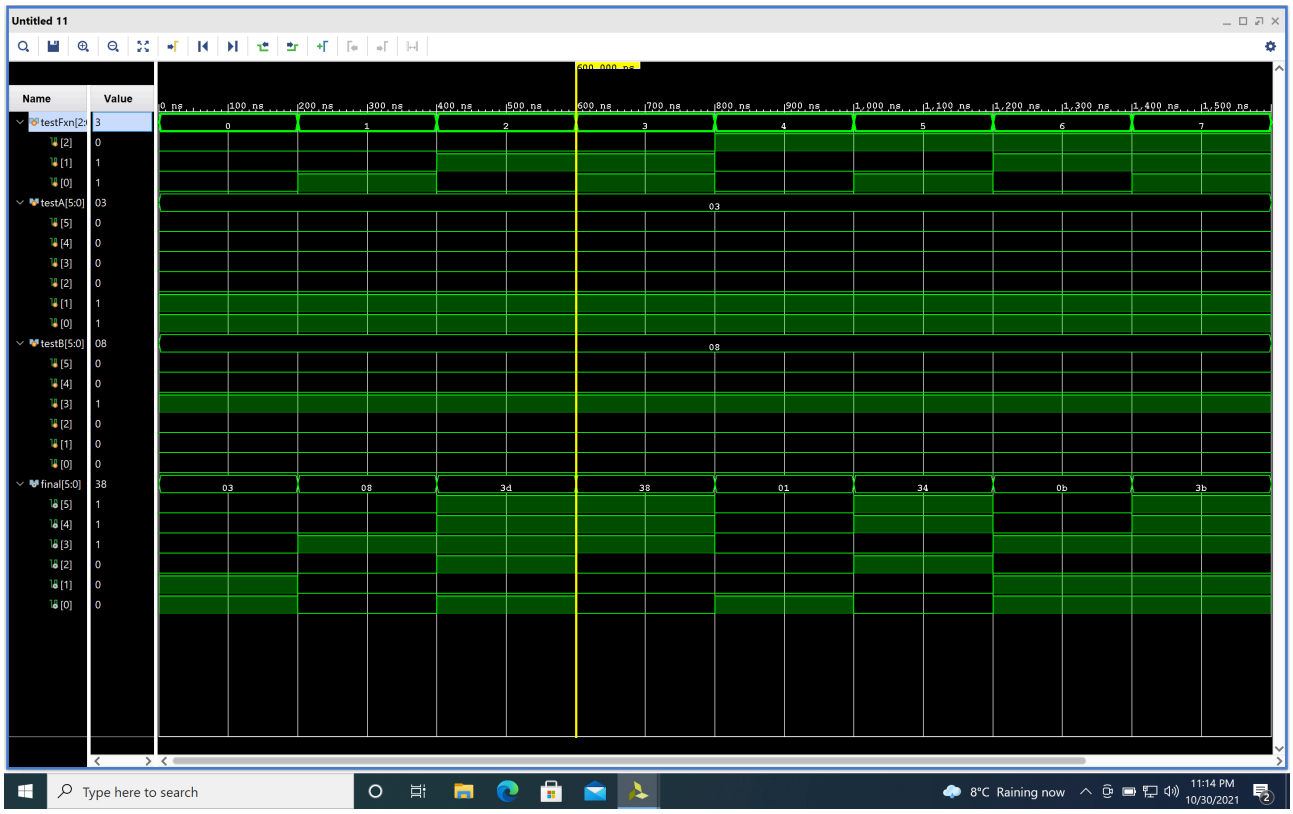
Fxn = 001



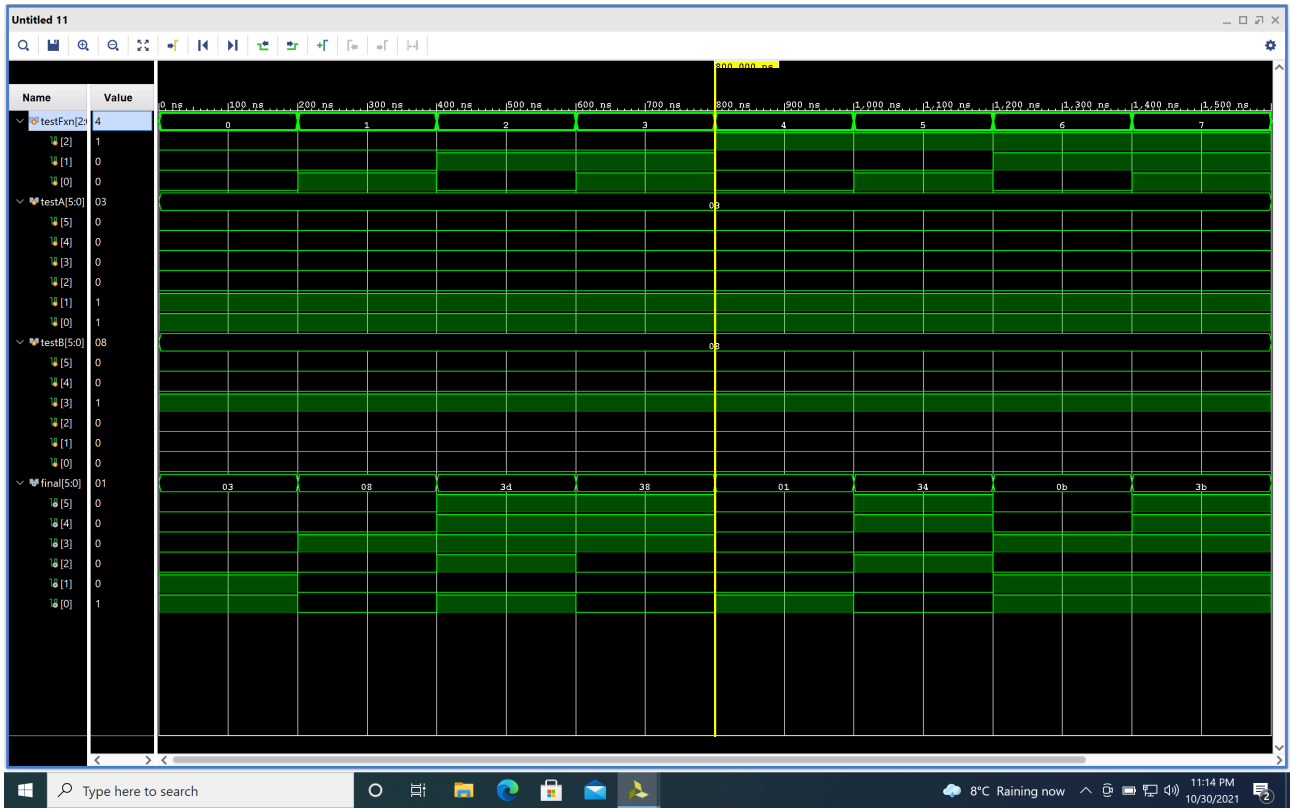
Fxn = 010



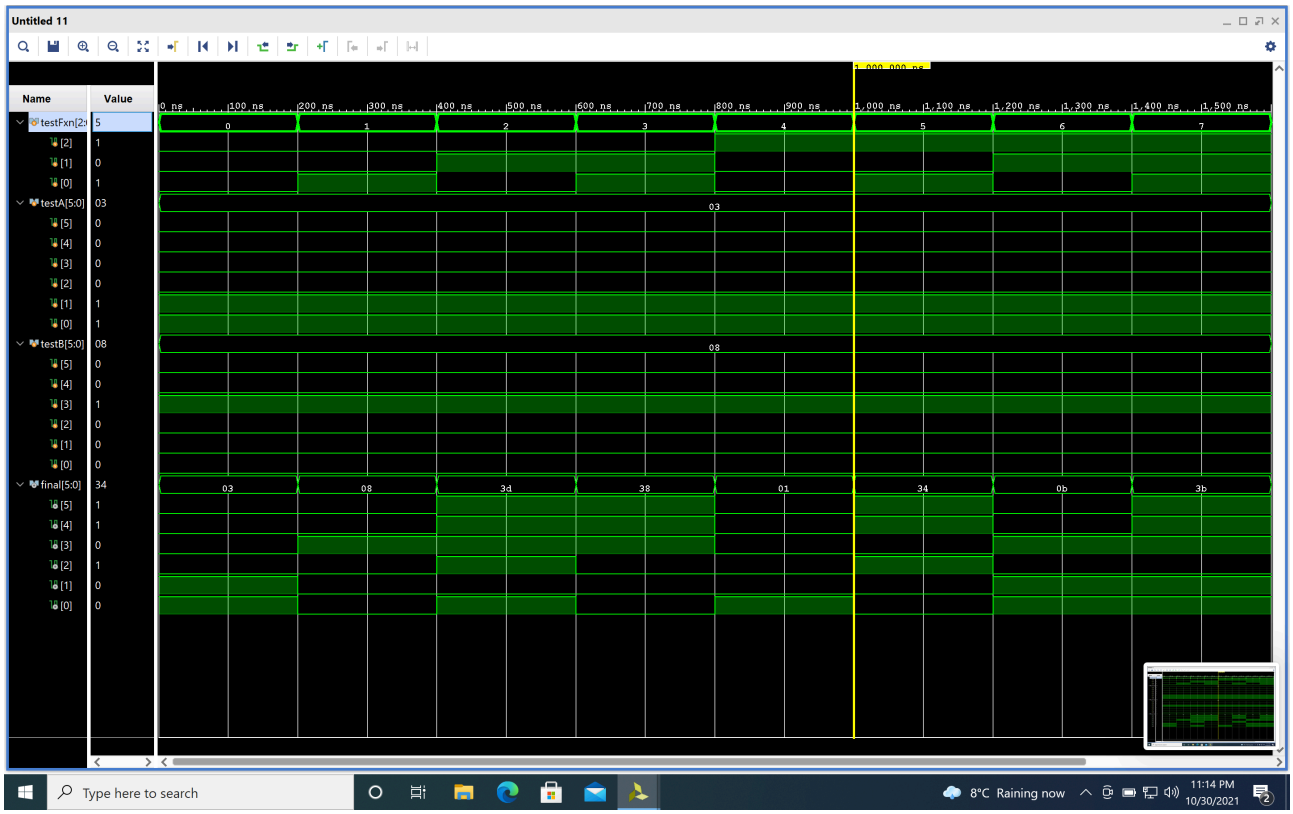
Fxn = 011



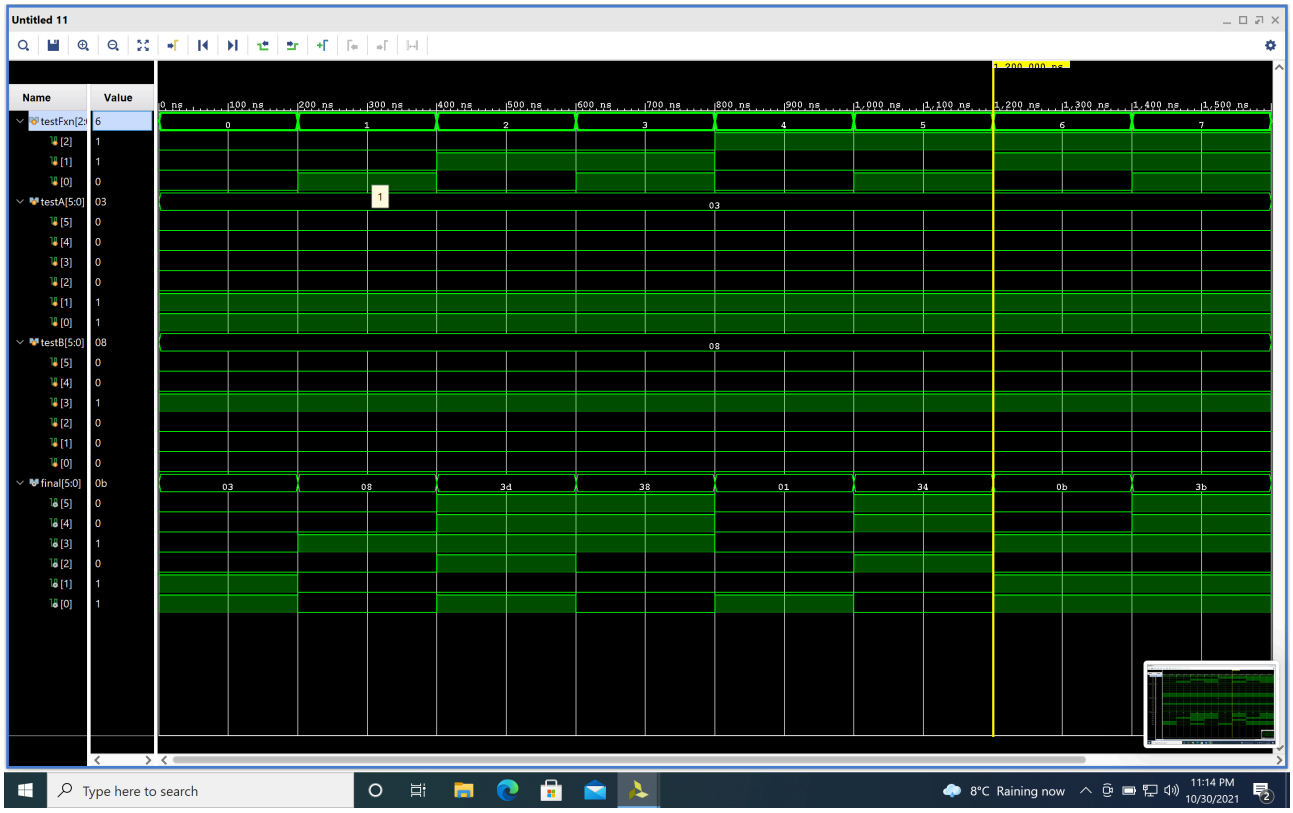
Fxn = 100



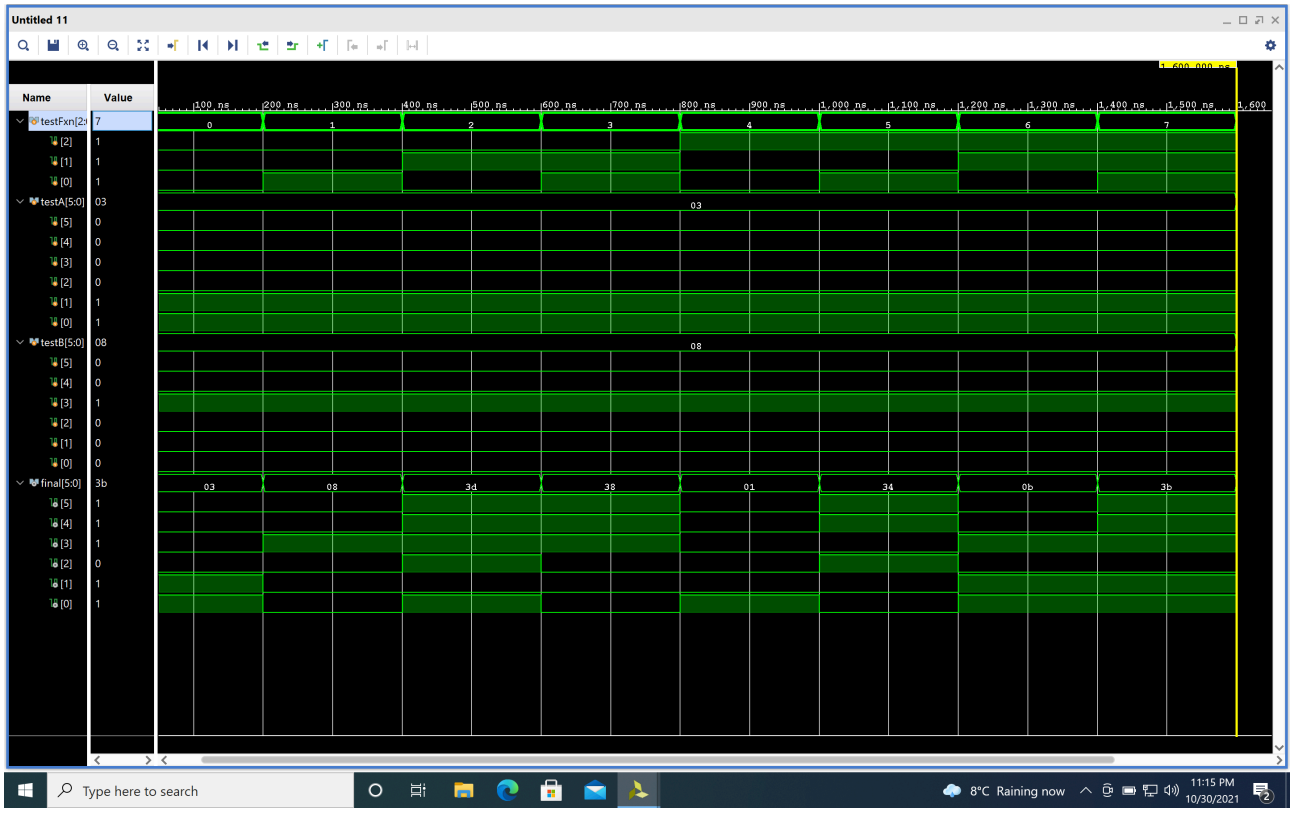
Fxn = 101



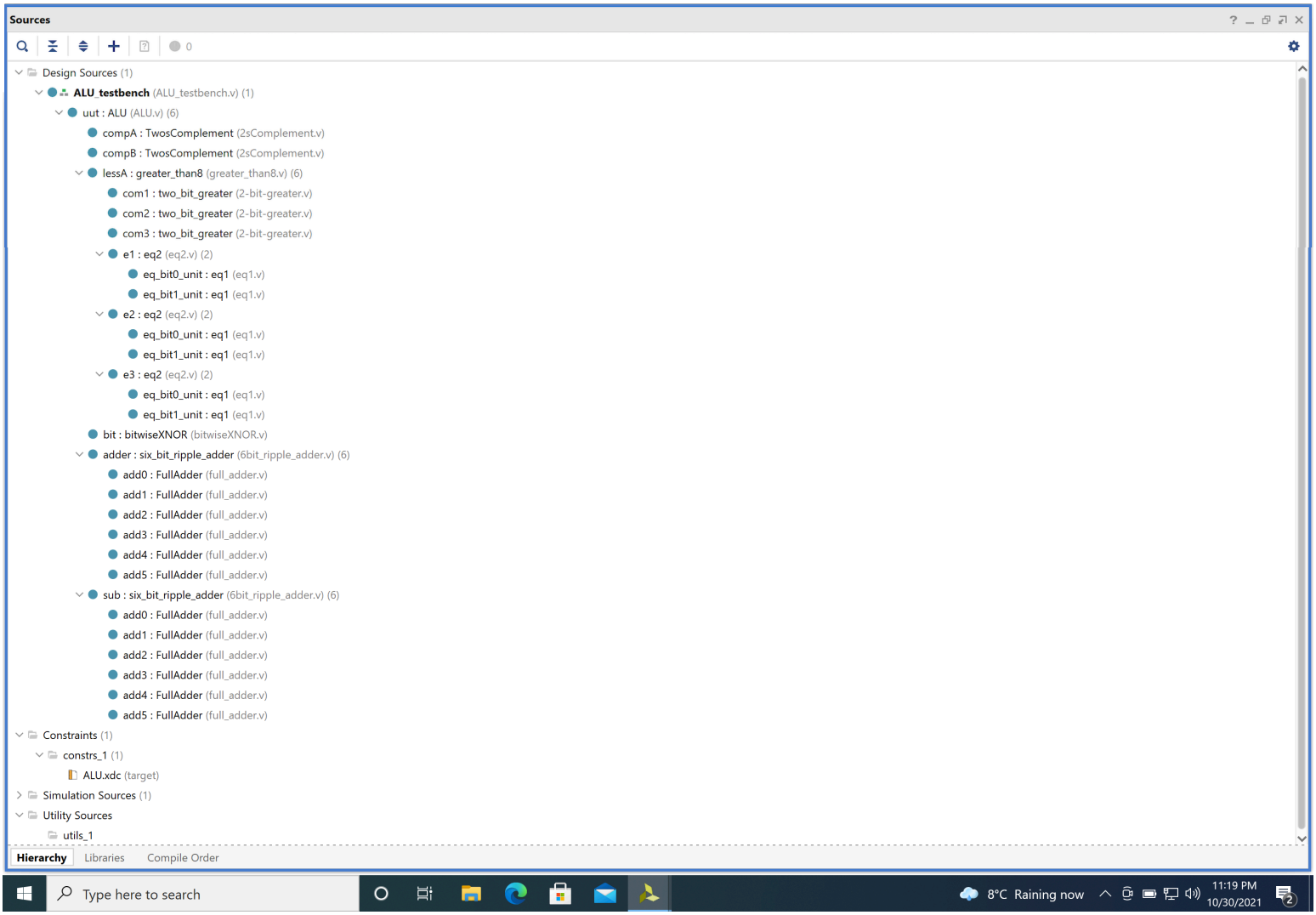
Fxn = 110



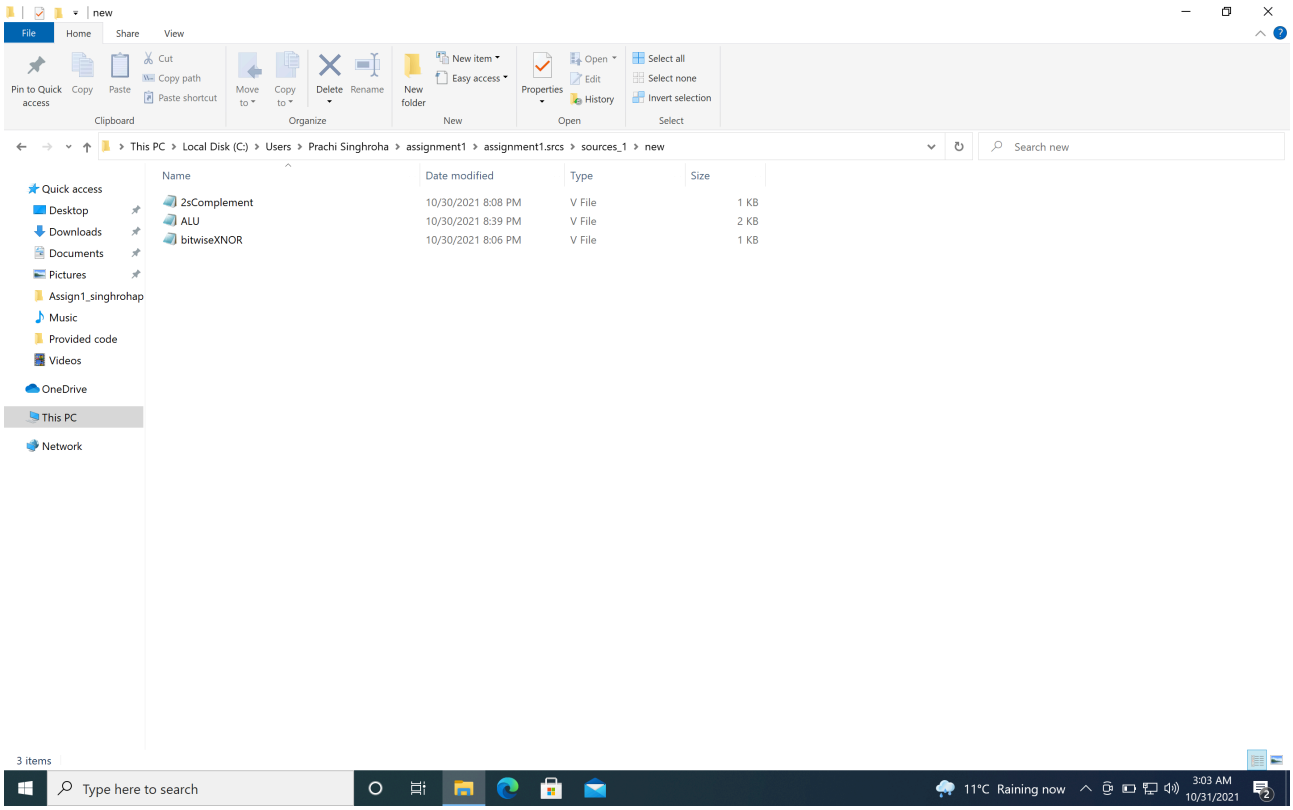
Fxn = 111



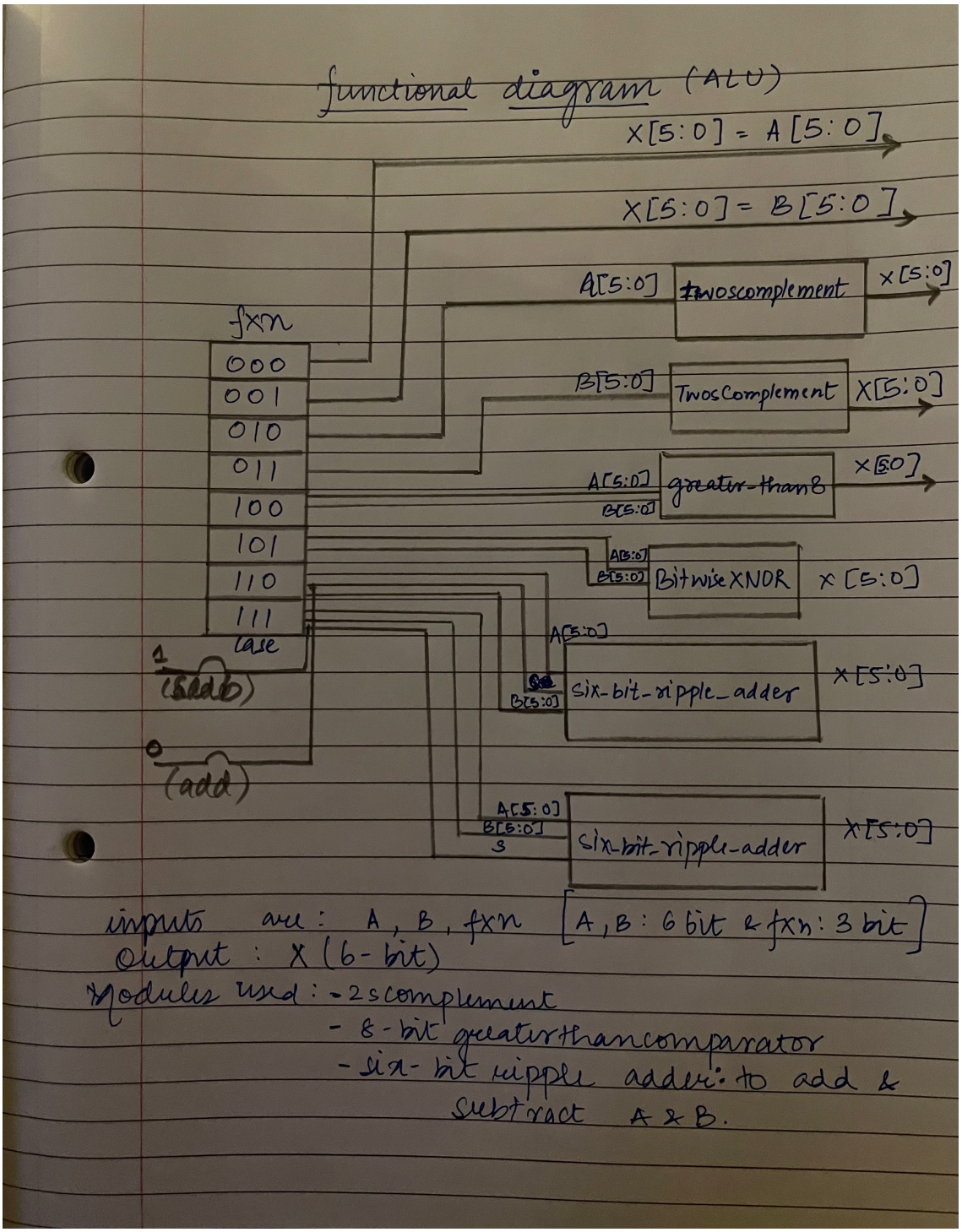
HIERARCHY


Sources
Design Sources (1)
ALU_testbench (ALU_testbench.v) (1)
uut : ALU (ALU.v) (6)
compA : TwosComplement (2sComplement.v)
compB : TwosComplement (2sComplement.v)
lessA : greater_than8 (greater_than8.v) (6)
com1 : two_bit_greater (2-bit-greater.v)
com2 : two_bit_greater (2-bit-greater.v)
com3 : two_bit_greater (2-bit-greater.v)
e1 : eq2 (eq2.v) (2)
eq_bit0_unit : eq1 (eq1.v)
eq_bit1_unit : eq1 (eq1.v)
e2 : eq2 (eq2.v) (2)
eq_bit0_unit : eq1 (eq1.v)
eq_bit1_unit : eq1 (eq1.v)
e3 : eq2 (eq2.v) (2)
eq_bit0_unit : eq1 (eq1.v)
eq_bit1_unit : eq1 (eq1.v)
bit : bitwiseXNOR (bitwiseXNOR.v)
adder : six_bit_ripple_adder (6bit_ripple_adder.v) (6)
add0 : FullAdder (full_adder.v)
add1 : FullAdder (full_adder.v)
add2 : FullAdder (full_adder.v)
add3 : FullAdder (full_adder.v)
add4 : FullAdder (full_adder.v)
add5 : FullAdder (full_adder.v)
sub : six_bit_ripple_adder (6bit_ripple_adder.v) (6)
add0 : FullAdder (full_adder.v)
add1 : FullAdder (full_adder.v)
add2 : FullAdder (full_adder.v)
add3 : FullAdder (full_adder.v)
add4 : FullAdder (full_adder.v)
add5 : FullAdder (full_adder.v)
Constraints (1)
constrs_1 (1)
ALU.xdc (target)
Simulation Sources (1)
Utility Sources
utils_1
Hierarchy
Libraries
Compile Order
Type here to search
8°C Raining now
11:19 PM
10/30/2021

FILE DIRECTORY



FUNCTIONAL DIAGRAM



DEMO

Connect board to your PC. Program it using the files in the zip files of assignment1. Input switches are given below using which one can provide input to the board and get the desired output!

ALU.xdc

C:/Users/Prachi Singhroha/assignment1/assignment1.srcs/constrs_1/new/ALU.xdc

```
set_property IOSTANDARD LVCMOS25 [get_ports {A[5]}]
set_property IOSTANDARD LVCMOS25 [get_ports {A[4]}]
set_property IOSTANDARD LVCMOS25 [get_ports {A[3]}]
set_property IOSTANDARD LVCMOS25 [get_ports {A[2]}]
set_property IOSTANDARD LVCMOS25 [get_ports {A[1]}]
set_property IOSTANDARD LVCMOS25 [get_ports {A[0]}]
set_property IOSTANDARD LVCMOS25 [get_ports {B[5]}]
set_property IOSTANDARD LVCMOS25 [get_ports {B[4]}]
set_property IOSTANDARD LVCMOS25 [get_ports {B[3]}]
set_property IOSTANDARD LVCMOS25 [get_ports {B[2]}]
set_property IOSTANDARD LVCMOS25 [get_ports {B[1]}]
set_property IOSTANDARD LVCMOS25 [get_ports {B[0]}]
set_property IOSTANDARD LVCMOS25 [get_ports {fxn[2]}]
set_property IOSTANDARD LVCMOS25 [get_ports {fxn[1]}]
set_property IOSTANDARD LVCMOS25 [get_ports {fxn[0]}]
set_property IOSTANDARD LVCMOS25 [get_ports {X[5]}]
set_property IOSTANDARD LVCMOS25 [get_ports {X[4]}]
set_property IOSTANDARD LVCMOS25 [get_ports {X[3]}]
set_property IOSTANDARD LVCMOS25 [get_ports {X[2]}]
set_property IOSTANDARD LVCMOS25 [get_ports {X[1]}]
set_property IOSTANDARD LVCMOS25 [get_ports {X[0]}]
set_property PACKAGE_PIN V17 [get_ports {A[0]}]
set_property PACKAGE_PIN V16 [get_ports {A[1]}]
set_property PACKAGE_PIN W16 [get_ports {A[2]}]
set_property PACKAGE_PIN W17 [get_ports {A[3]}]
set_property PACKAGE_PIN W15 [get_ports {A[4]}]
set_property PACKAGE_PIN V15 [get_ports {A[5]}]
set_property PACKAGE_PIN W14 [get_ports {B[0]}]
set_property PACKAGE_PIN W13 [get_ports {B[1]}]
set_property PACKAGE_PIN V2 [get_ports {B[2]}]
set_property PACKAGE_PIN T3 [get_ports {B[3]}]
set_property PACKAGE_PIN T2 [get_ports {B[4]}]
set_property PACKAGE_PIN R3 [get_ports {B[5]}]
set_property PACKAGE_PIN W2 [get_ports {fxn[0]}]
set_property PACKAGE_PIN U1 [get_ports {fxn[1]}]
set_property PACKAGE_PIN T1 [get_ports {fxn[2]}]
set_property PACKAGE_PIN U16 [get_ports {X[0]}]
set_property PACKAGE_PIN E19 [get_ports {X[1]}]
set_property PACKAGE_PIN U19 [get_ports {X[2]}]
set_property PACKAGE_PIN V19 [get_ports {X[3]}]
set_property PACKAGE_PIN W18 [get_ports {X[4]}]
set_property PACKAGE_PIN U15 [get_ports {X[5]}]
```

Type here to search 8°C Raining now 11:34 PM 10/30/2021

APPENDIX

Previous submissions for adder and greater than circuit

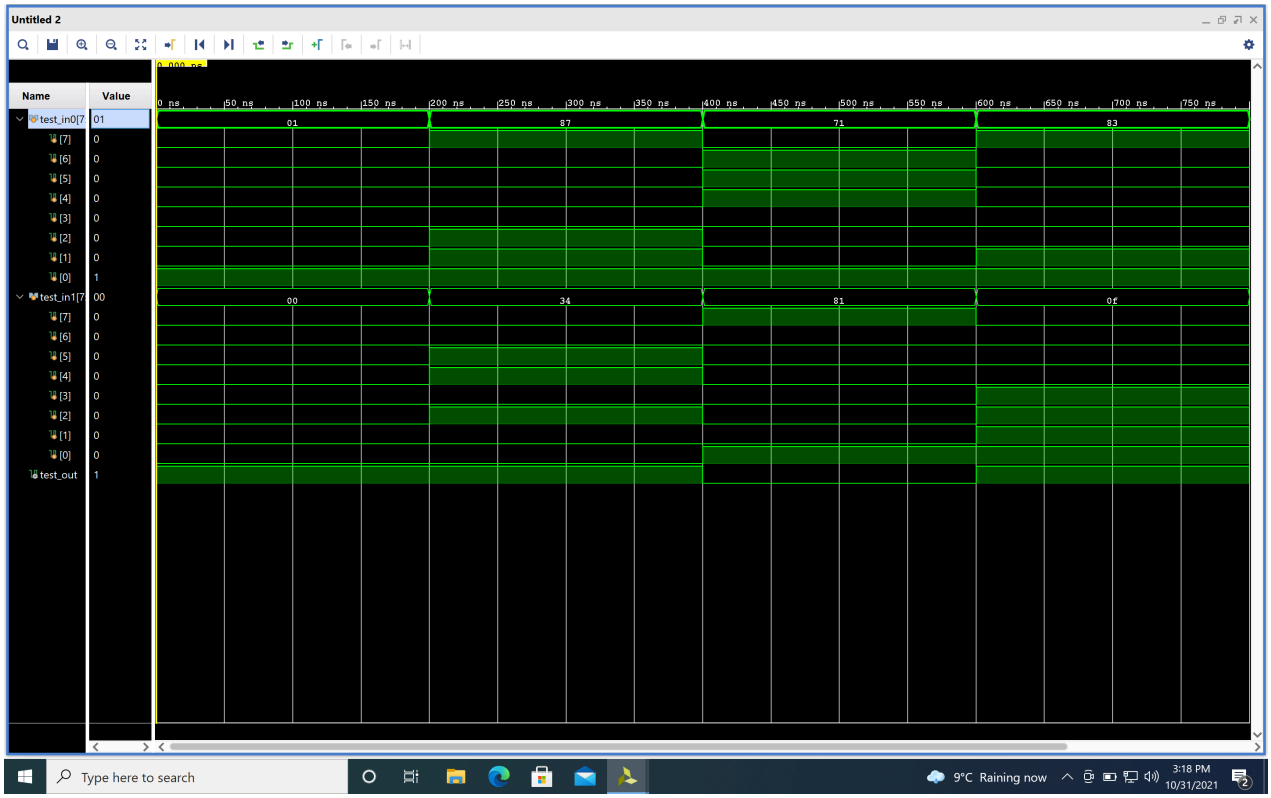
LAB B

Waveforms

1. For test vector 1 where input are as follows

   test_in0 = 8'b00000001;

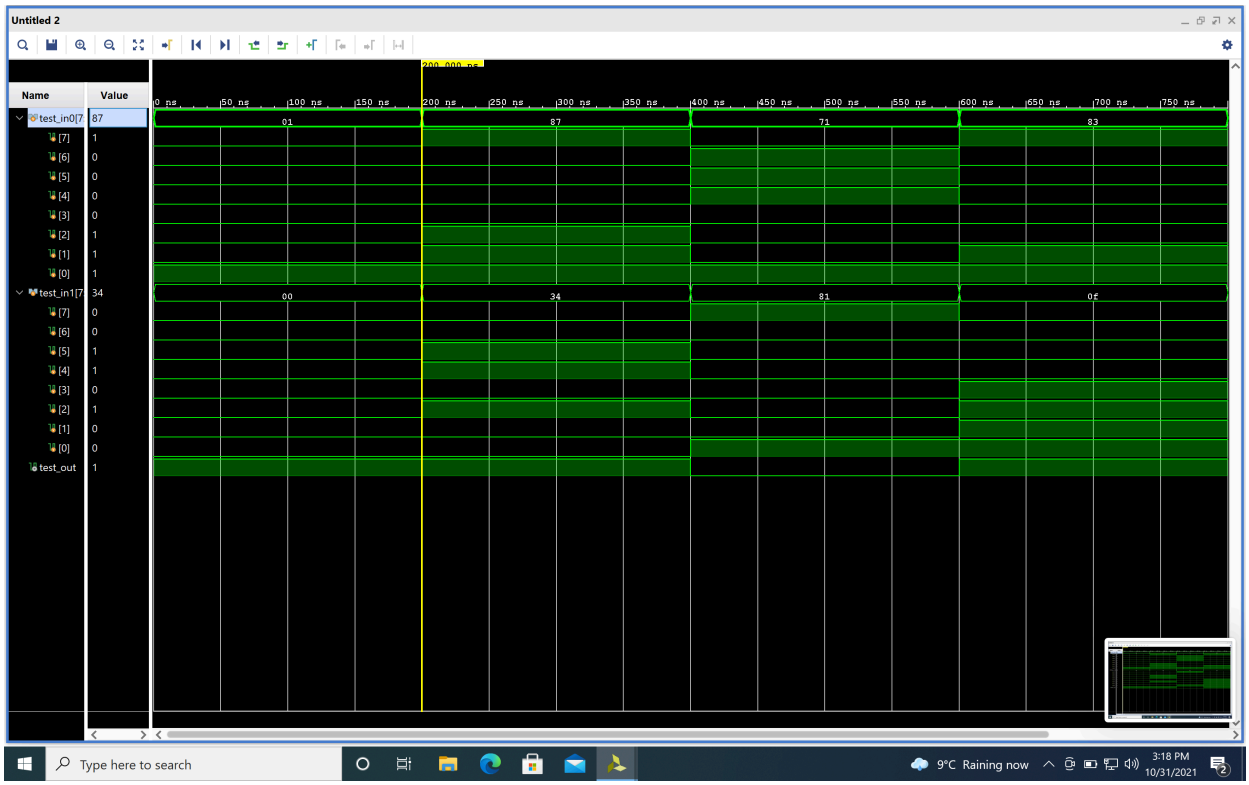
   test_in1 = 8’b00000000;



2. For test vector 2 where input are as follows

   test_in0 = 8'b10000111;

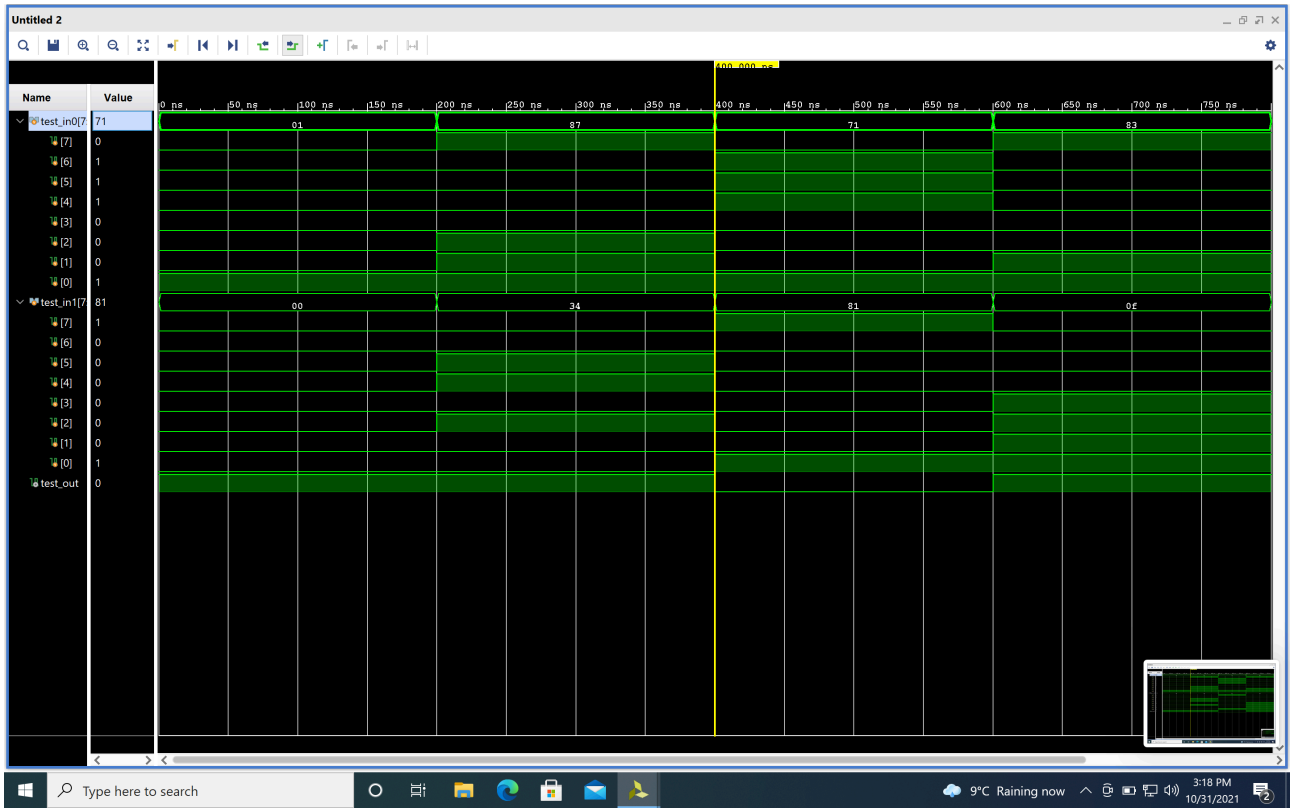
   test_in1 = 8'b00110100;



3. For test vector 3 where input are as follows

test_in0 = 8'b01110001;

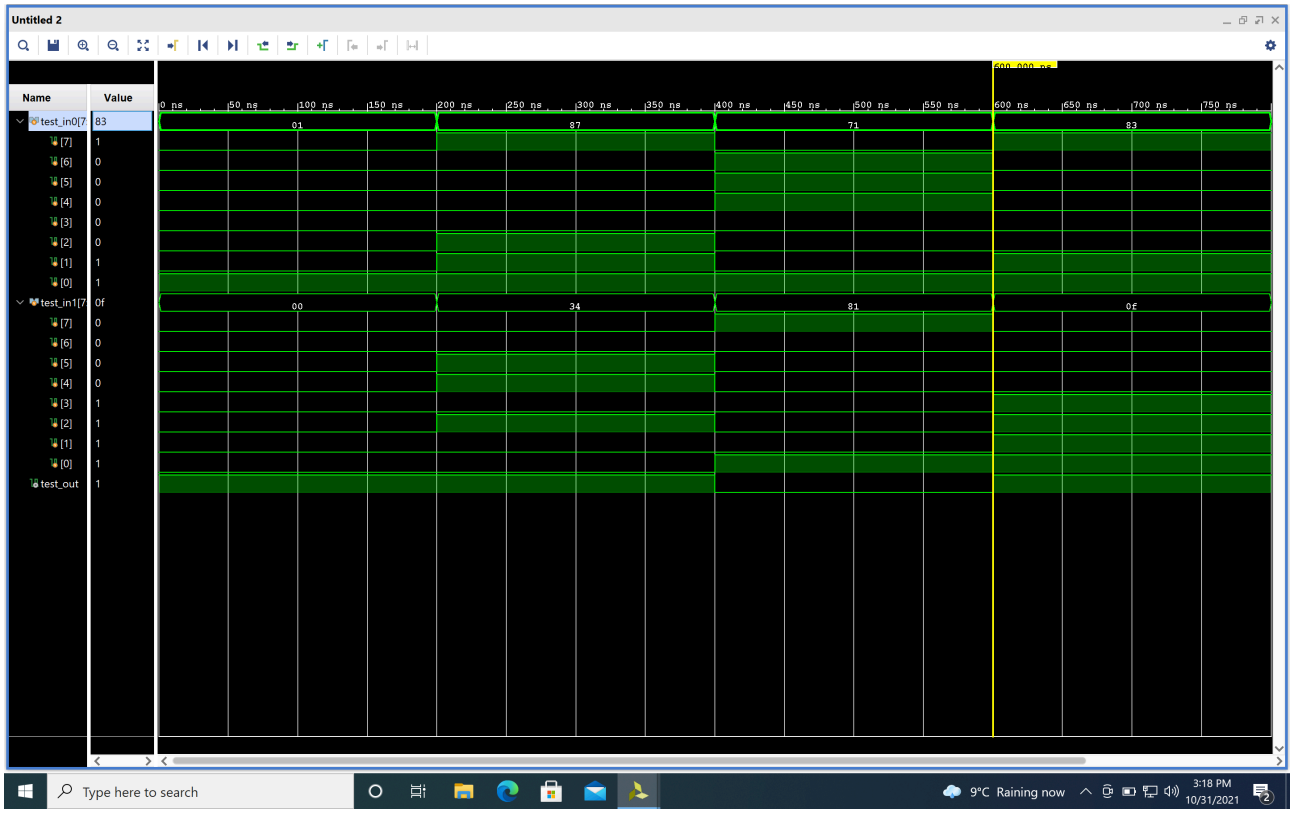
test_in1 = 8'b10000001;



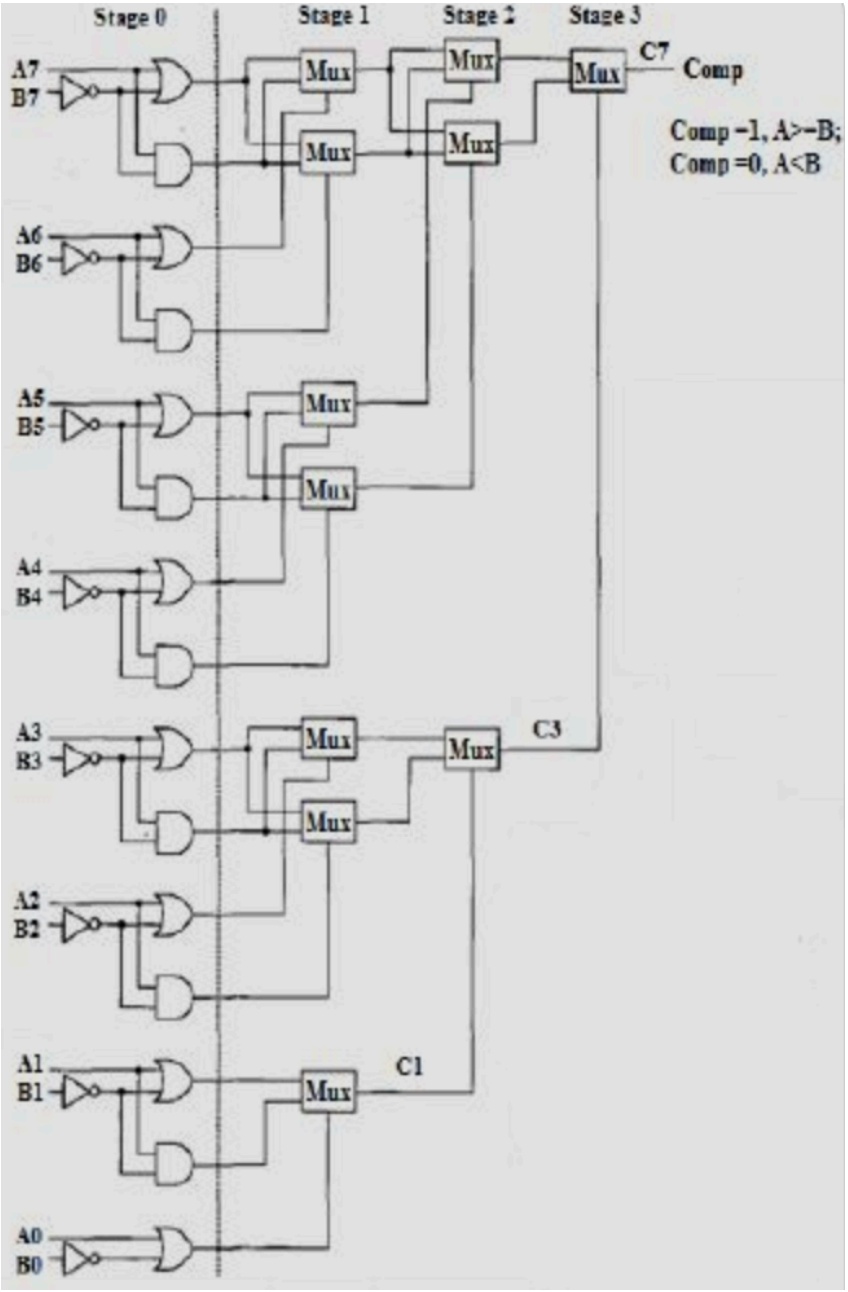
4. For test vector 4 where input are as follows

test_in0 = 8'b10000011;

test_in1 = 8'b00001111;



Block Diagram of 8 bit comparator



Codes

1. eq2_tb(testbench)

eq2_tb.v

C:/Users/Prachi Singhroha/Downloads/labB_code/Provided code/eq2_tb.v

```
// Listing 1.7
// The `timescale directive specifies that
// the simulation time unit is 1 ns  and
// the simulation timestep is 10 ps
`timescale 1 ns/10 ps

module eq2_testbench;
   // signal declaration
   reg  [7:0] test_in0, test_in1;
   wire  test_out;

   // instantiate the circuit under test
   greater_than8 com
      (.a(test_in0), .b(test_in1), .o(test_out));

   //  test vector generator
   initial
   begin
      // test vector 1
      test_in0 = 8'b00000001;
      test_in1 = 8'b00000000;
      # 200;
      // test vector 2
      test_in0 = 8'b10000111;
      test_in1 = 8'b00110100;
      # 200;
      // test vector 3
      test_in0 = 8'b01110001;
      test_in1 = 8'b10000001;
      # 200;
      // test vector 4
      test_in0 = 8'b10000011;
      test_in1 = 8'b00001111;
      # 200;

      // stop simulation
      $stop;
   end


endmodule
```

2. eq1

eq1.v

C:/Users/Prachi Singhroha/Downloads/labB_code/Provided code/eq1.v

```
// Listing 1.1
module eq1
    // I/O ports
    (
     input wire i0, i1,
     output wire eq
    );

    // signal declaration
    wire p0, p1;

    // body
    // sum of two product terms
    assign eq = p0 | p1;
    // product terms
    assign p0 = ~i0 & ~i1;
    assign p1 = i0 & i1;

endmodule
```

3. eq2

eq2.v

C:/Users/Prachi Singhroha/Downloads/labB_code/Provided code/eq2.v

```
// Listing 1.4
module eq2
    (
     input  wire[1:0] a, b,          // a adn b are the two 2-bit numbers to compare
     output wire aeqb                // single bit output. Should be high if a adn b the same
    );

    // internal signal declaration, used to wire outpus of the 1 bit comparators
    wire e0, e1;

    // body
    // instantiate two 1-bit comparators that we already know are tested and work
    // named instantiation allows us to change order of ports.
    eq1 eq_bit0_unit (.i0(a[0]), .i1(b[0]), .eq(e0));

     // following changes are made:
     // i. semicolon added
     // ii. module name changed
    eq1 eq_bit1_unit (.eq(e1), .i0(a[1]), .i1(b[1]));

    // a and b are equal if individual bits are equal, which comes from the 1-bit comparators
    assign aeqb = e0 & e1; // operator changed from xor to and

endmodule
```

4. two_bit_greater (2 bit equality comparator)

2-bit-greater.v

C:/Users/Prachi Singhroha/project_1/project_1.srcs/sources_1/new/2-bit-greater.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 10/29/2021 10:38:32 PM
// Design Name:
// Module Name: 2-bit-greater
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module two_bit_greater(
        input wire[1:0] a, b,
        output wire c

    );

    wire p,q,r;

    assign p = a[0]&~b[1]&b[0];
    assign q = a[1]&~b[1];
    assign r = a[1]&a[0]&~b[0];

    assign c = p|q|r;


endmodule
```

5. greater_than8 (8 bit greater than equal to comparator)

greater_than8.v

C:/Users/Prachi Singhroha/project_1/project_1.srcs/sources_1/new/greater_than8.v

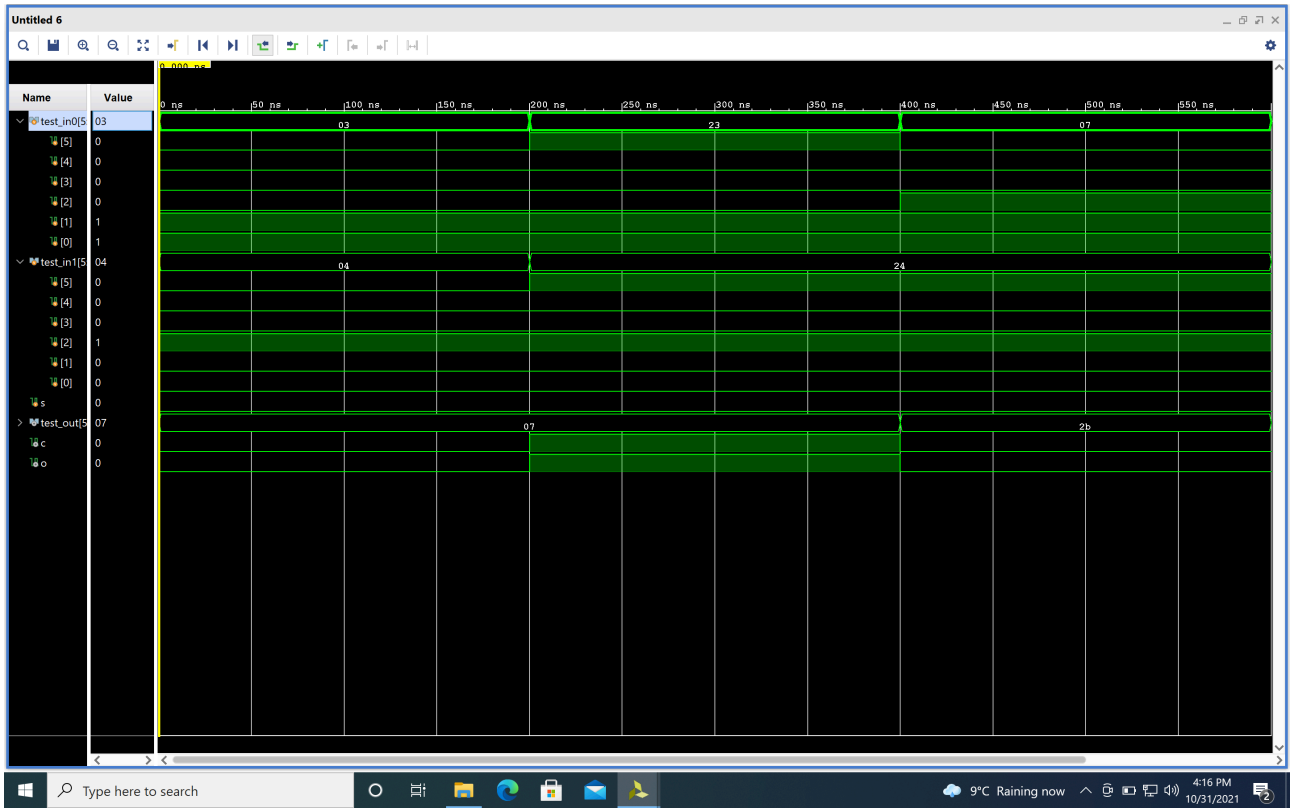
```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 10/29/2021 10:53:57 PM
// Design Name:
// Module Name: greater_than8
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module greater_than8(
        input wire [7:0] a, b,
        output wire o
    );

    wire q, w, e, r;
    two_bit_greater com1 (.a(a[1:0]), .b(b[1:0]), .c(q));
    two_bit_greater com2 (.a(a[3:2]), .b(b[3:2]), .c(w));
    two_bit_greater com3 (.a(a[5:4]), .b(b[5:4]), .c(e));
    two_bit_greater com4 (.a(a[7:6]), .b(b[7:6]), .c(r));

    wire s, d, f, g;
    eq2 e1 (.a(a[1:0]), .b(b[1:0]), .aeqb(s));
    eq2 e2 (.a(a[3:2]), .b(b[3:2]), .aeqb(d));
    eq2 e3 (.a(a[5:4]), .b(b[5:4]), .aeqb(f));
    eq2 e4 (.a(a[7:6]), .b(b[7:6]), .aeqb(g));

    assign o = r|g&(e|f&(w|d&(q|s)));


endmodule
```

LAB C

Waveforms

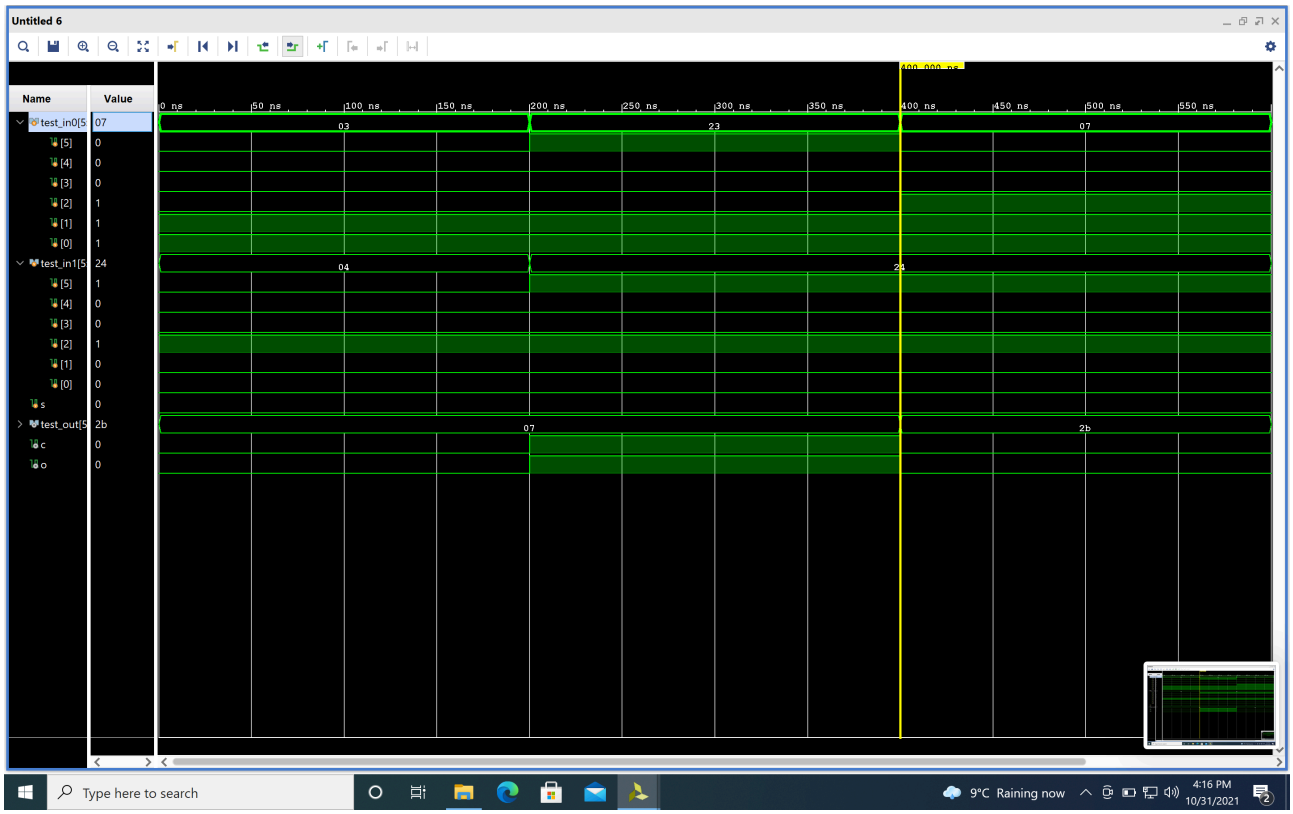
A. For test vector 1 where input are as follows

test_in0 = 6'b000011;
test_in1 = 6'b000100;
s = 1’b0;



B. For test vector 2 where input are as follows

test_in0 = 6'b100011;
test_in1 = 6'b100100;
s = 1’b0;



C. For test vector 3 where input are as follows

test_in0 = 6'b000111;
test_in1 = 6'b100100;
s = 1’b0;

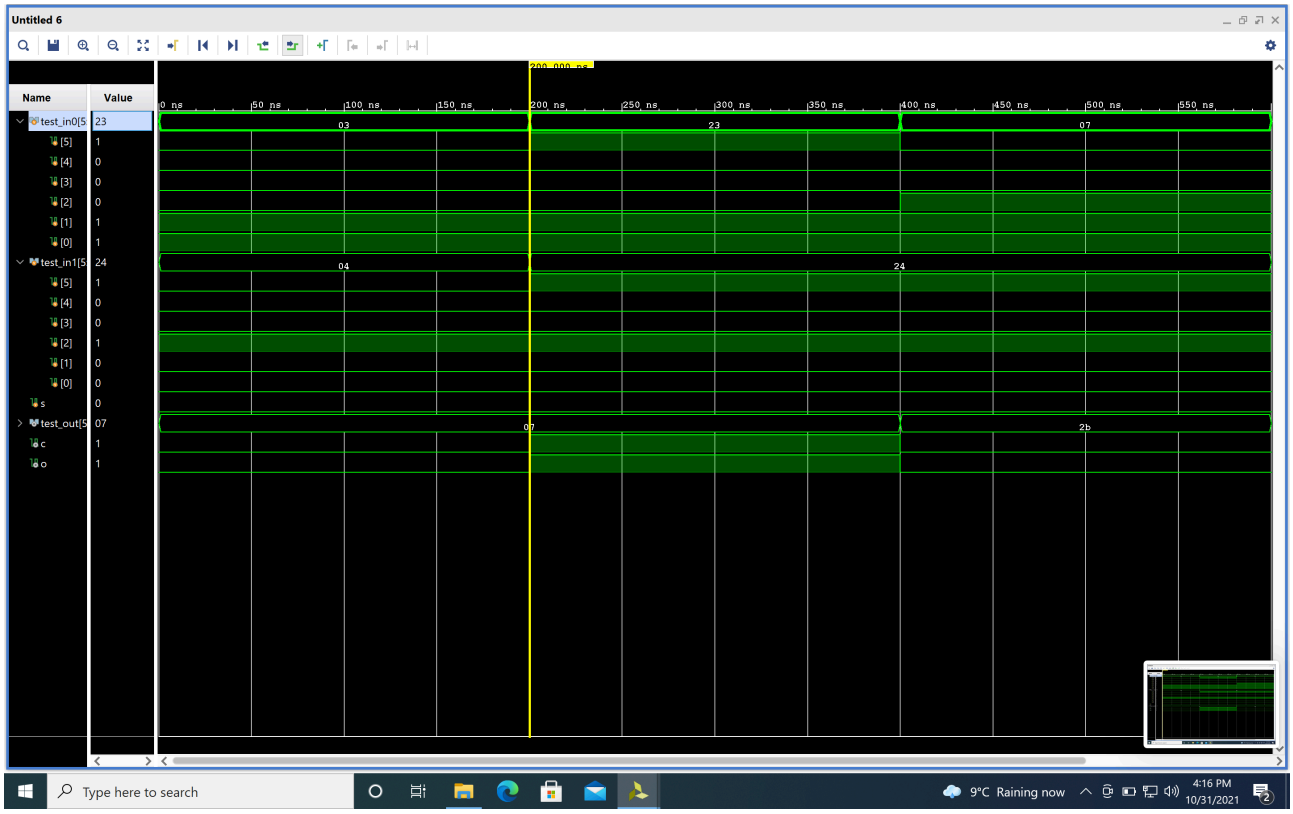


Table of test vectors

Test ID	X	Y	sel	c_out expected	overflow expected	sum expected	c_out observed	overflow observed	sum observed
1	b000011	b000100	0	0	0	b000111	0	0	b000111
2	b100011	b100100	0	1	1	b000111	1	1	b000111
3	b000111	b100100	0	1	1	b101011	1	1	b101011

Code

For testbench

ripple_tb.v

C:/Users/Prachi Singhroha/Desktop/Assign1_singhrohap_codes/labC/ripple_tb.v

```
// Listing 1.7
// The `timescale directive specifies that
// the simulation time unit is 1 ns  and
// the simulation timestep is 10 ps
`timescale 1 ns/10 ps

module six_bit_ripple_adder_testbench;
   // signal declaration
   reg  [5:0] test_in0, test_in1;
   reg  s;
   wire  [5:0] test_out;
   wire c, o;

  // instantiate the circuit under test
   six_bit_ripple_adder out (.a(test_in0), .b(test_in1), .sel(s), .overflow(o), .c_out(c), .sum(test_out));

   //  test vector generator
   initial
   begin
      // test vector 1
      test_in0 = 6'b000011;
      test_in1 = 6'b000100;
      s = 1'b0;
      #200;
      // test vector 2
      test_in0 = 6'b100011;
      test_in1 = 6'b100100;
      s = 1'b0;
      #200;
      // test vector 3
      test_in0 = 6'b000111;
      test_in1 = 6'b100100;
      s = 1'b0;
      # 200;
      // stop simulation
      $stop;
   end


endmodule
```

For full adder

full_adder.v

C:/Users/Prachi Singhroha/Downloads/full_adder.v

```
module FullAdder(a, b, cin, s, cout);
  // 3C7 LabD 2010
  // a and b are the bits to add
  // cin is carry in
  input wire a, b, cin;

  // s is the sum of a and b. cout is any carry out bit
  // wires since just using assign here
  output wire s, cout;

  // logic for sum and carry
  assign s = cin ^ a ^ b;
  assign cout = (b & cin) | (a & cin) | (a & b);

endmodule
```

For 6-bit-ripple-adder

6bit_ripple_adder.v

C:/Users/Prachi Singhroha/project_2/project_2.srcs/sources_1/new/6bit_ripple_adder.v

```
// Module Name: 6bit_ripple_adder
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module six_bit_ripple_adder(
    input wire [5:0] a, b,
    input wire sel,
    output wire overflow, c_out,
    output wire [5:0] sum

    );

    wire s0, s1, s2, s3, s4, s5;
    wire x0, x1, x2, x3, x4;

    assign s0 = b[0] ^ sel;
    assign s1 = b[1] ^ sel;
    assign s2 = b[2] ^ sel;
    assign s3 = b[3] ^ sel;
    assign s4 = b[4] ^ sel;
    assign s5 = b[5] ^ sel;

    FullAdder add0(.a(a[0]), .b(s0), .cin(sel), .s(sum[0]), .cout(x0));
    FullAdder add1(.a(a[1]), .b(s1), .cin(x0), .s(sum[1]), .cout(x1));
    FullAdder add2(.a(a[2]), .b(s2), .cin(x1), .s(sum[2]), .cout(x2));
    FullAdder add3(.a(a[3]), .b(s3), .cin(x2), .s(sum[3]), .cout(x3));
    FullAdder add4(.a(a[4]), .b(s4), .cin(x3), .s(sum[4]), .cout(x4));
    FullAdder add5(.a(a[5]), .b(s5), .cin(x4), .s(sum[5]), .cout(c_out));

    assign overflow = c_out ^ x4;

endmodule
```