ASSIGNMENT 4

Digital Design and Computer Organization

UE21CS251A

3rd Semester, Academic Year 2021-22

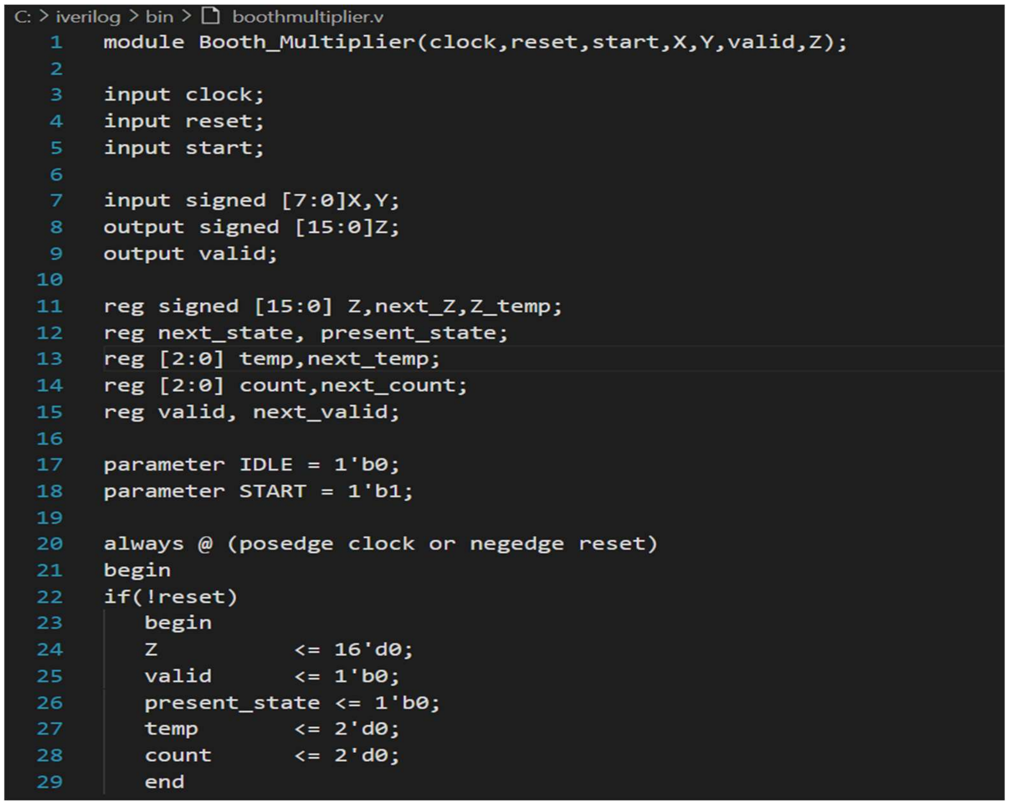
Date:

Name: Sujal.S	SRN:PES2UG21CS548	Section I

Program Number: ___1____

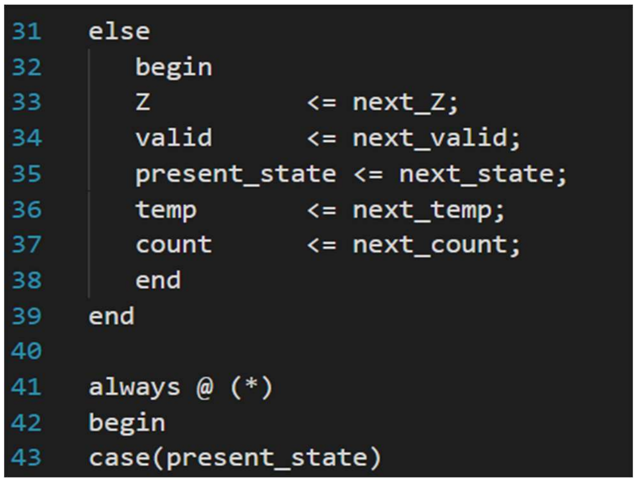
Title of the Program

AIM: Write a Verilog code and test bench for an 8-bit Booth Multiplier

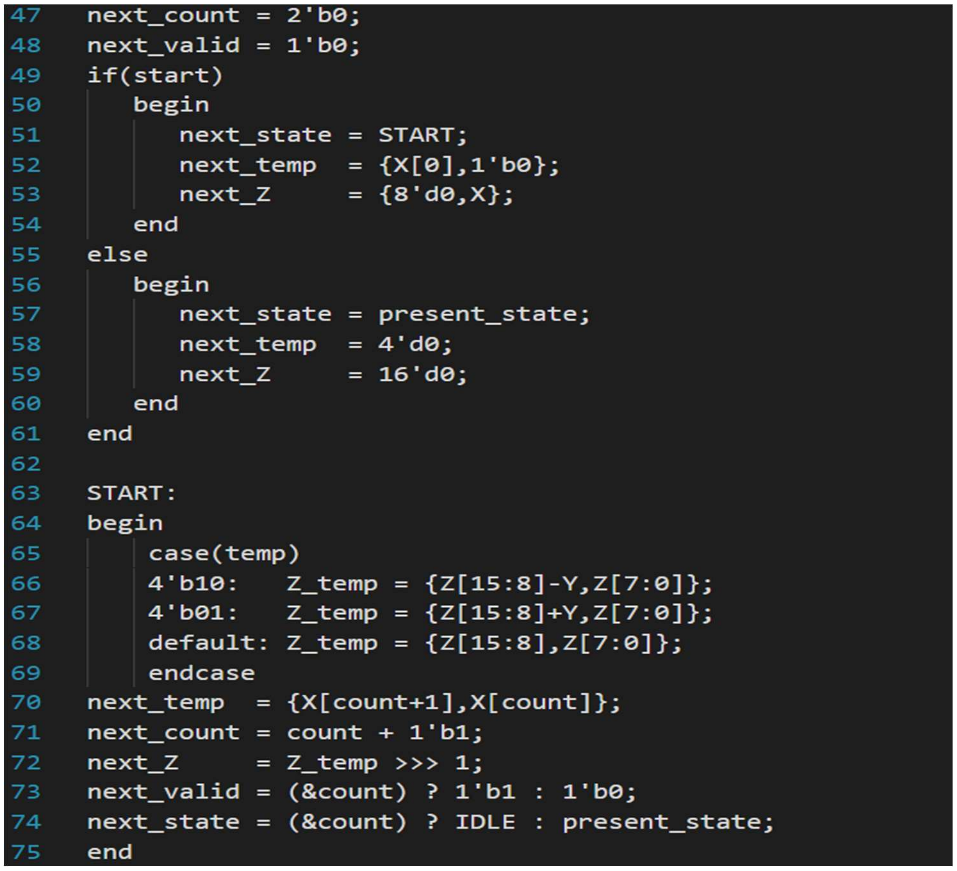


C: > iverilog > bin > boothmultiplier.v

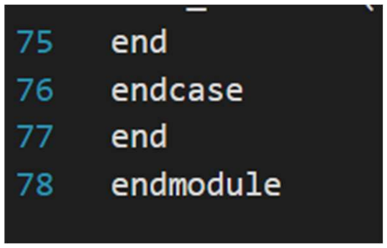
```
module Booth_Multiplier(clock,reset,start,X,Y,valid,Z);

input clock;
input reset;
input start;

input signed [7:0]X,Y;
output signed [15:0]Z;
output valid;

reg signed [15:0] Z,next_Z,Z_temp;
reg next_state, present_state;
reg [2:0] temp,next_temp;
reg [2:0] count,next_count;
reg valid, next_valid;

parameter IDLE = 1'b0;
parameter START = 1'b1;

always @ (posedge clock or negedge reset)
begin
if(!reset)
   begin
   Z           <= 16'd0;
   valid       <= 1'b0;
   present_state <= 1'b0;
   temp        <= 2'd0;
   count       <= 2'd0;
   end
```



```
else
    begin
    Z            <= next_Z;
    valid        <= next_valid;
    present_state <= next_state;
    temp         <= next_temp;
    count        <= next_count;
    end
end

always @ (*)
begin
case(present_state)
```



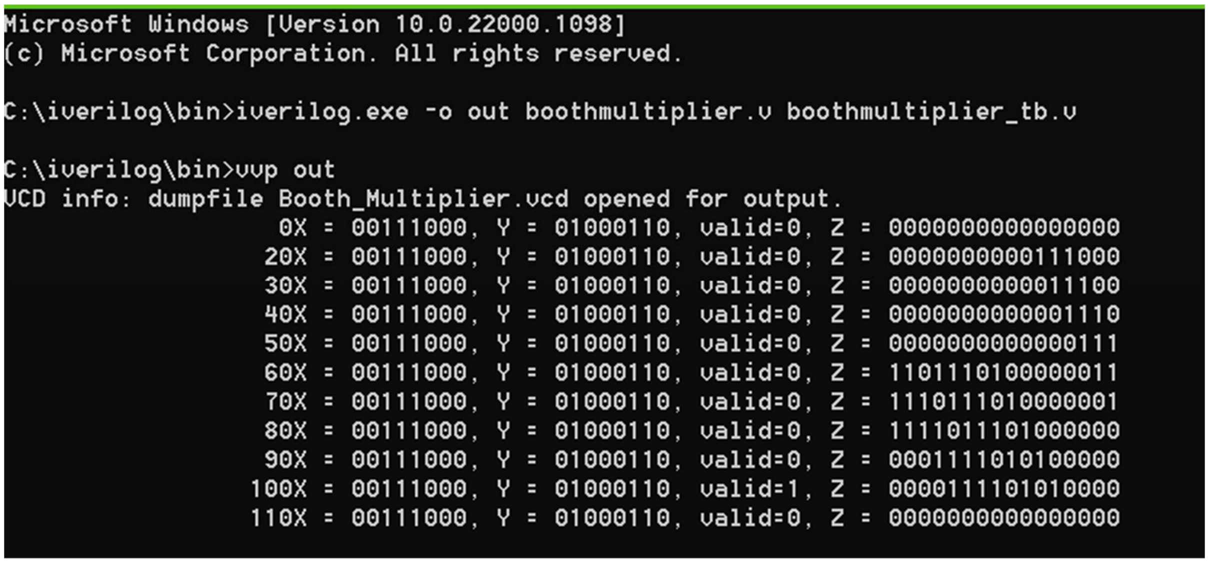
```
next_count = 2'b0;
next_valid = 1'b0;
if(start)
    begin
        next_state = START;
        next_temp  = {X[0],1'b0};
        next_Z     = {8'd0,X};
    end
else
    begin
        next_state = present_state;
        next_temp  = 4'd0;
        next_Z     = 16'd0;
    end
end

START:
begin
     case(temp)
     4'b10:   Z_temp = {Z[15:8]-Y,Z[7:0]};
     4'b01:   Z_temp = {Z[15:8]+Y,Z[7:0]};
     default: Z_temp = {Z[15:8],Z[7:0]};
     endcase
next_temp  = {X[count+1],X[count]};
next_count = count + 1'b1;
next_Z     = Z_temp >>> 1;
next_valid = (&count) ? 1'b1 : 1'b0;
next_state = (&count) ? IDLE : present_state;
end
```



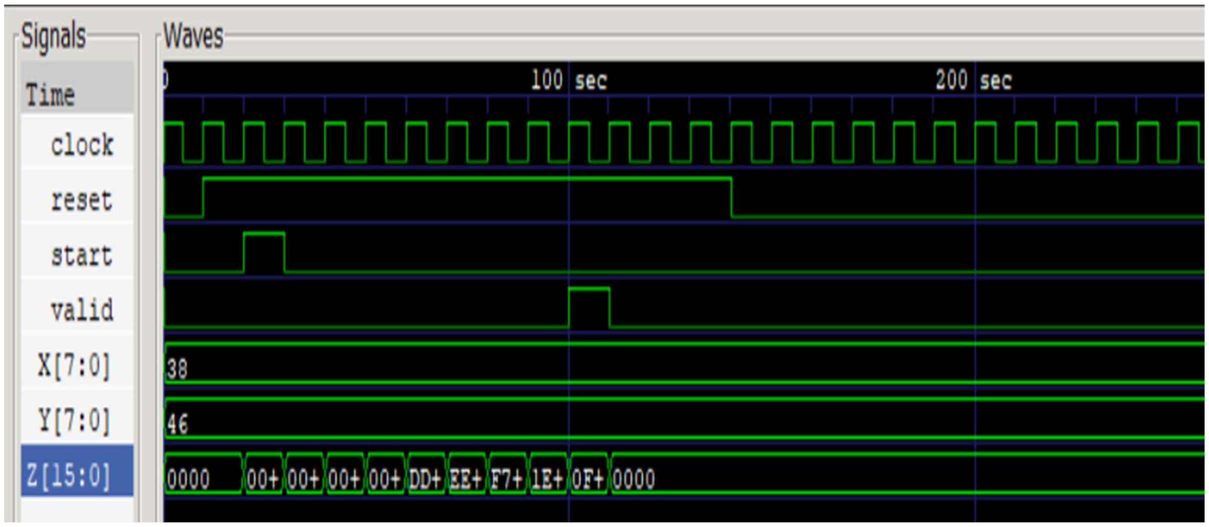
```
end
endcase
end
endmodule
```

Case 1: +ve number * +ve number

56 * 70 = 3920

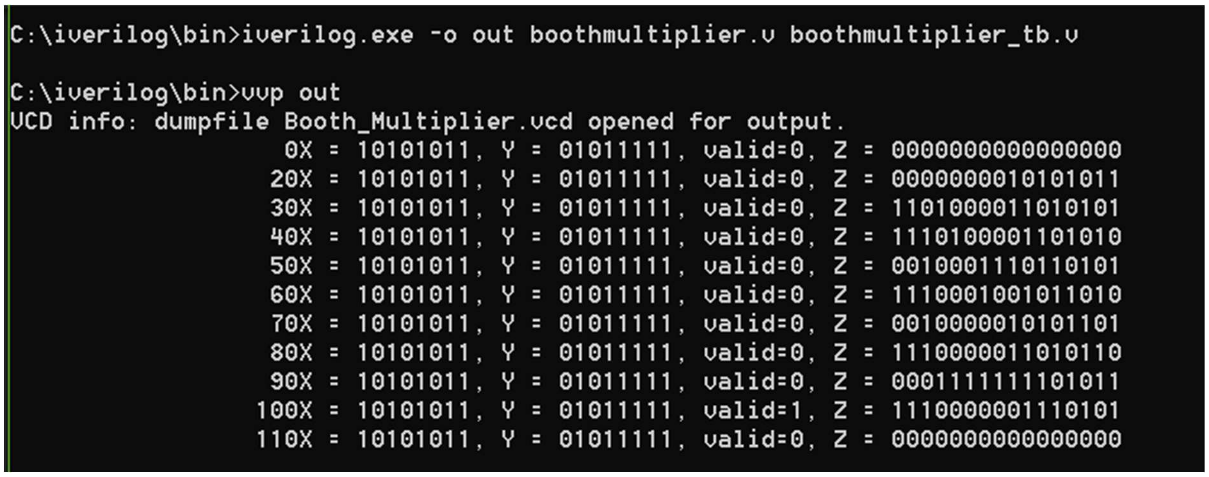


```
Microsoft Windows [Version 10.0.22000.1098]
(c) Microsoft Corporation. All rights reserved.

C:\iverilog\bin>iverilog.exe -o out boothmultiplier.v boothmultiplier_tb.v

C:\iverilog\bin>vvp out
VCD info: dumpfile Booth_Multiplier.vcd opened for output.
                   0X = 00111000, Y = 01000110, valid=0, Z = 0000000000000000
                  20X = 00111000, Y = 01000110, valid=0, Z = 0000000000111000
                  30X = 00111000, Y = 01000110, valid=0, Z = 0000000000011100
                  40X = 00111000, Y = 01000110, valid=0, Z = 0000000000001110
                  50X = 00111000, Y = 01000110, valid=0, Z = 0000000000000111
                  60X = 00111000, Y = 01000110, valid=0, Z = 1101110100000011
                  70X = 00111000, Y = 01000110, valid=0, Z = 1110111010000001
                  80X = 00111000, Y = 01000110, valid=0, Z = 1111011101000000
                  90X = 00111000, Y = 01000110, valid=0, Z = 0001111010100000
                 100X = 00111000, Y = 01000110, valid=1, Z = 0000111101010000
                 110X = 00111000, Y = 01000110, valid=0, Z = 0000000000000000
```

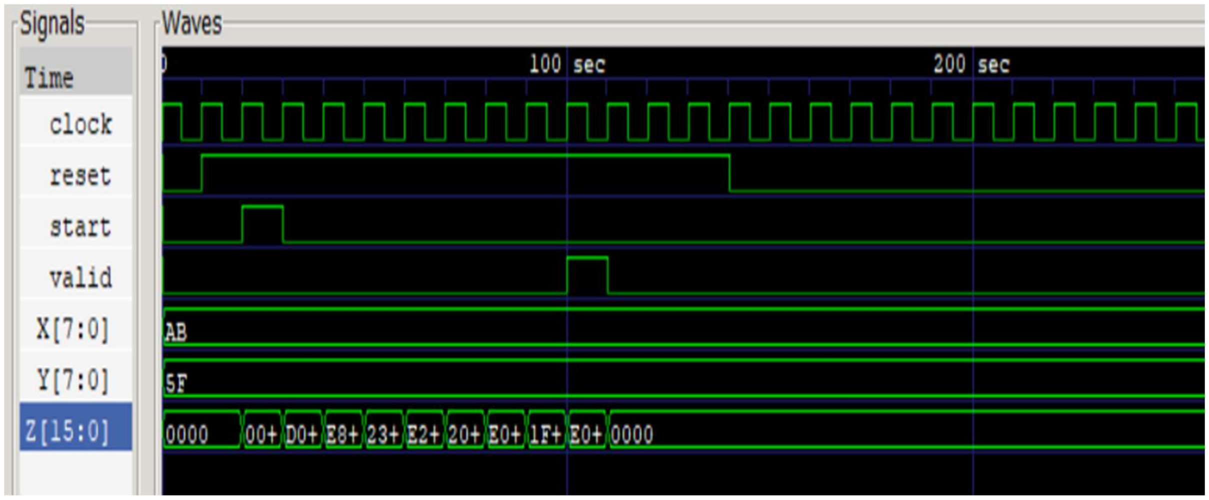


Case 2 : -ve number * +ve number

-85 * 95 = -8075

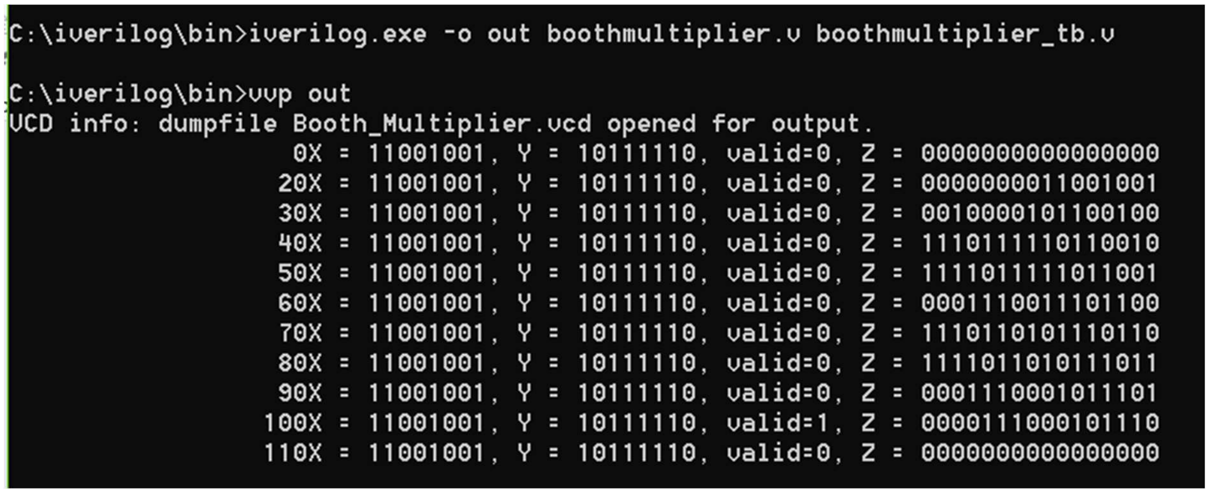


```
C:\iverilog\bin>iverilog.exe -o out boothmultiplier.v boothmultiplier_tb.v

C:\iverilog\bin>vvp out
VCD info: dumpfile Booth_Multiplier.vcd opened for output.
                  0X = 10101011, Y = 01011111, valid=0, Z = 0000000000000000
                 20X = 10101011, Y = 01011111, valid=0, Z = 0000000010101011
                 30X = 10101011, Y = 01011111, valid=0, Z = 1101000011010101
                 40X = 10101011, Y = 01011111, valid=0, Z = 1110100001101010
                 50X = 10101011, Y = 01011111, valid=0, Z = 0010001110110101
                 60X = 10101011, Y = 01011111, valid=0, Z = 1110001001011010
                 70X = 10101011, Y = 01011111, valid=0, Z = 0010000010101101
                 80X = 10101011, Y = 01011111, valid=0, Z = 1110000011010110
                 90X = 10101011, Y = 01011111, valid=0, Z = 0001111111101011
                100X = 10101011, Y = 01011111, valid=1, Z = 1110000001110101
                110X = 10101011, Y = 01011111, valid=0, Z = 0000000000000000
```

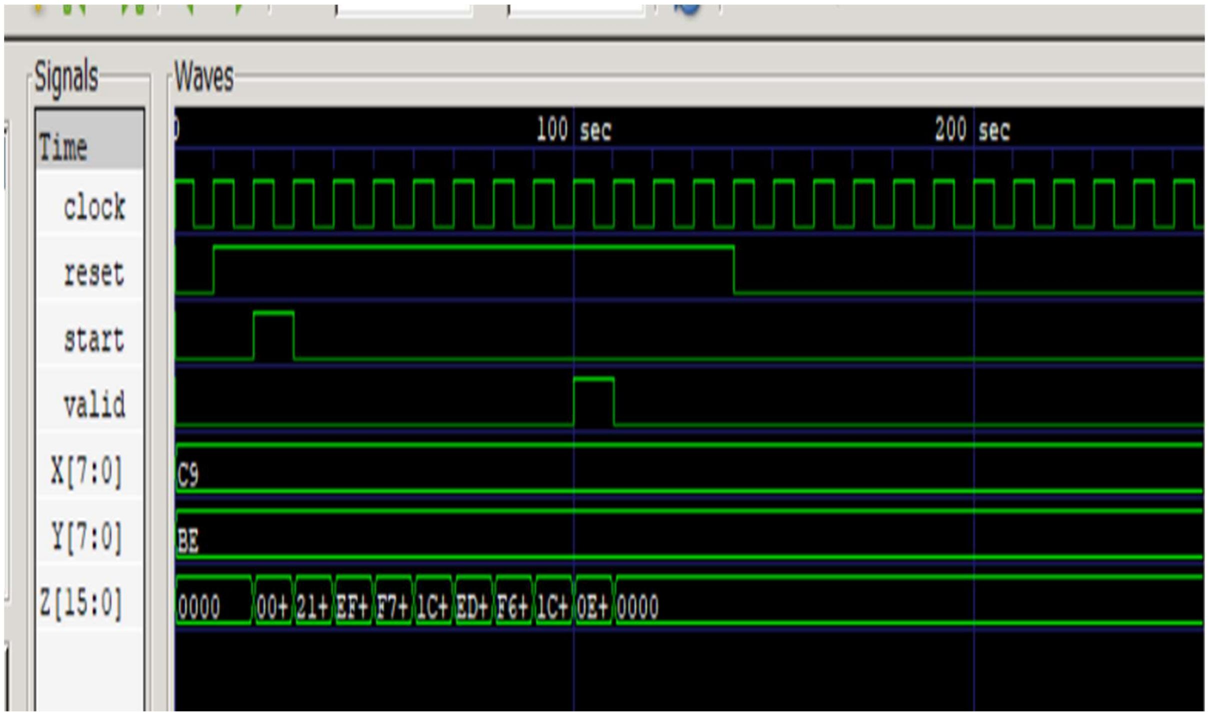


Case 3 : -ve number * -ve number

-55 * -66 =3630

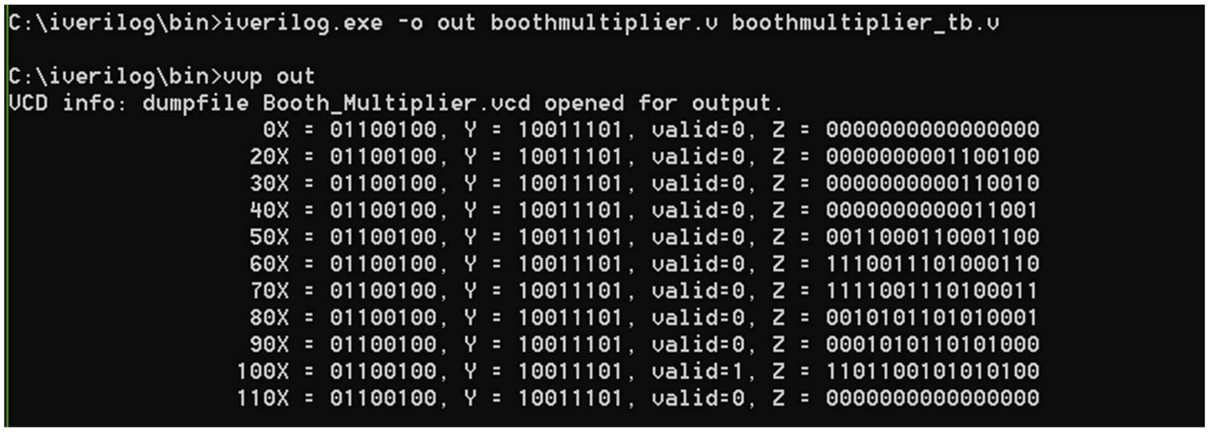


```
C:\iverilog\bin>iverilog.exe -o out boothmultiplier.v boothmultiplier_tb.v

C:\iverilog\bin>vvp out
VCD info: dumpfile Booth_Multiplier.vcd opened for output.
                   0X = 11001001, Y = 10111110, valid=0, Z = 0000000000000000
                  20X = 11001001, Y = 10111110, valid=0, Z = 0000000011001001
                  30X = 11001001, Y = 10111110, valid=0, Z = 0010000101100100
                  40X = 11001001, Y = 10111110, valid=0, Z = 1110111110110010
                  50X = 11001001, Y = 10111110, valid=0, Z = 1111011111011001
                  60X = 11001001, Y = 10111110, valid=0, Z = 0001110011101100
                  70X = 11001001, Y = 10111110, valid=0, Z = 1110110101110110
                  80X = 11001001, Y = 10111110, valid=0, Z = 1111011010111011
                  90X = 11001001, Y = 10111110, valid=0, Z = 0001110001011101
                 100X = 11001001, Y = 10111110, valid=1, Z = 0000111000101110
                 110X = 11001001, Y = 10111110, valid=0, Z = 0000000000000000
```

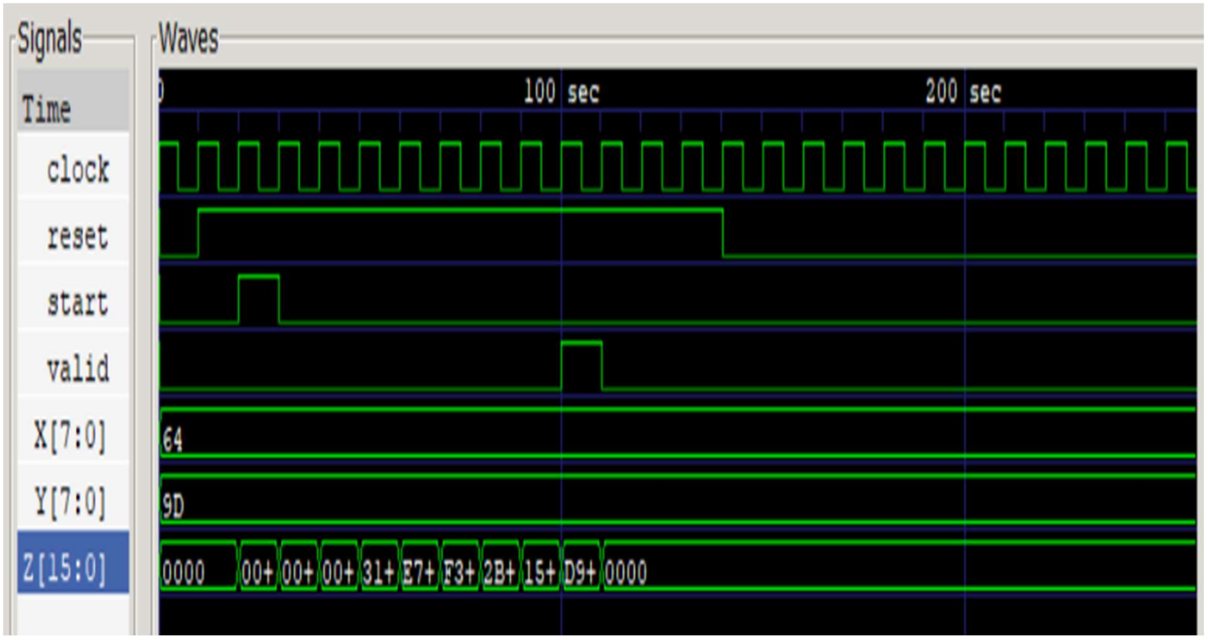


Case 4 : +ve number * - ve number

100*-99 = -9900



```
C:\iverilog\bin>iverilog.exe -o out boothmultiplier.v boothmultiplier_tb.v

C:\iverilog\bin>vvp out
VCD info: dumpfile Booth_Multiplier.vcd opened for output.
                   0X = 01100100, Y = 10011101, valid=0, Z = 0000000000000000
                  20X = 01100100, Y = 10011101, valid=0, Z = 0000000001100100
                  30X = 01100100, Y = 10011101, valid=0, Z = 0000000000110010
                  40X = 01100100, Y = 10011101, valid=0, Z = 0000000000011001
                  50X = 01100100, Y = 10011101, valid=0, Z = 0011000110001100
                  60X = 01100100, Y = 10011101, valid=0, Z = 1110011101000110
                  70X = 01100100, Y = 10011101, valid=0, Z = 1111001110100011
                  80X = 01100100, Y = 10011101, valid=0, Z = 0010101101010001
                  90X = 01100100, Y = 10011101, valid=0, Z = 0001010110101000
                 100X = 01100100, Y = 10011101, valid=1, Z = 1101100101010100
                 110X = 01100100, Y = 10011101, valid=0, Z = 0000000000000000
```



Disclaimer:

- The programs and output submitted is duly written, verified and executed my me.
- I have not copied from any of my peers nor from the external resource such as internet.
- If found plagiarized, I will abide with the disciplinary action of the University.

Signature: Sujal.S
Name:Sujal S
SRN: PES2UG21CS548
Section: I
Date:27/10/2022