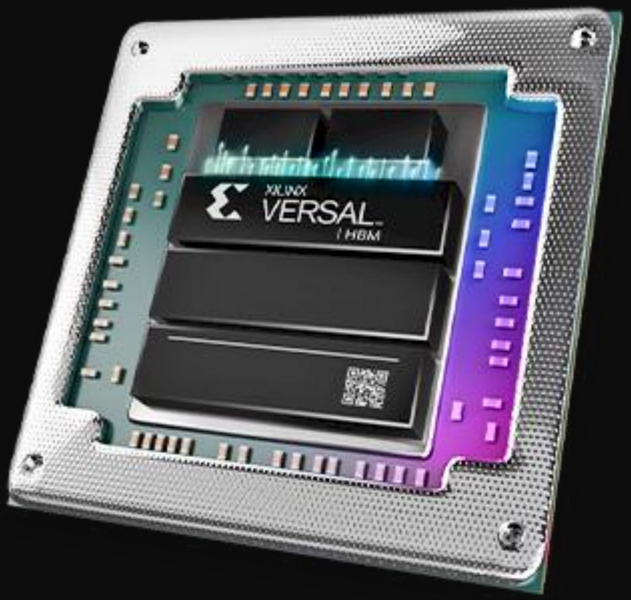
EC204

Digital System Design Lab

Lab – 2



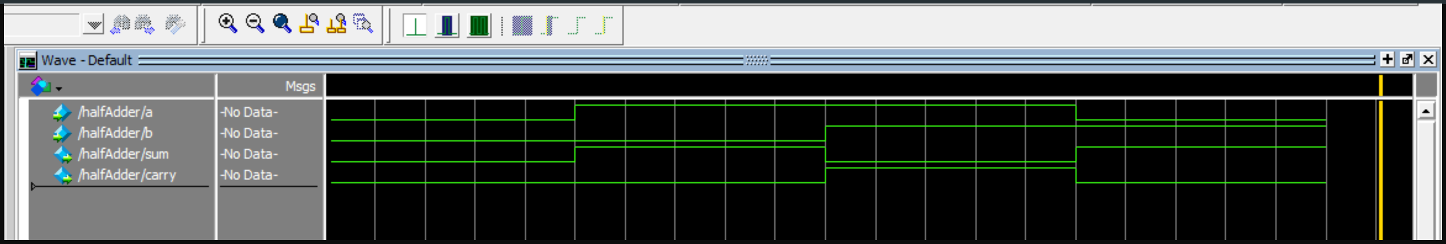
Utkarsh R Mahajan

201EC164, S11

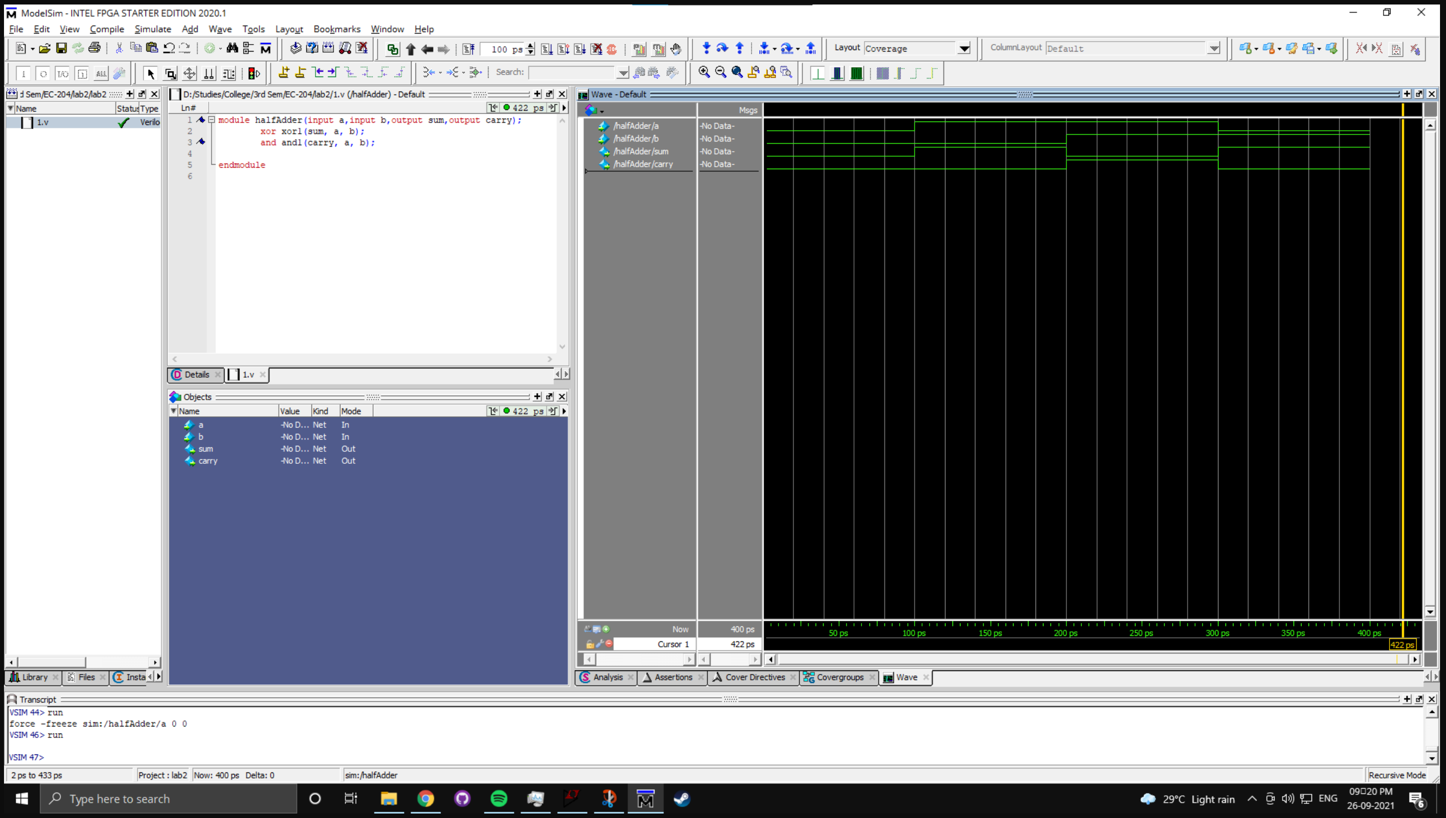
1] Half adder (a) using gates (b) using dataflow model (using assign statement)

(a) using gates

```verilog
module halfAdder(input a,input b,output sum,output carry);
    xor xor1(sum, a, b);
    and and1(carry, a, b);

endmodule
```



Wave - Default	Msgs
/halfAdder/a	-No Data-
/halfAdder/b	-No Data-
/halfAdder/sum	-No Data-
/halfAdder/carry	-No Data-



ModelSim - INTEL FPGA STARTER EDITION 2020.1

D:/Studies/College/3rd Sem/EC-204/lab2/1.v (/halfAdder) - Default

```verilog
module halfAdder(input a,input b,output sum,output carry);
        xor xor1(sum, a, b);
        and and1(carry, a, b);

endmodule
```

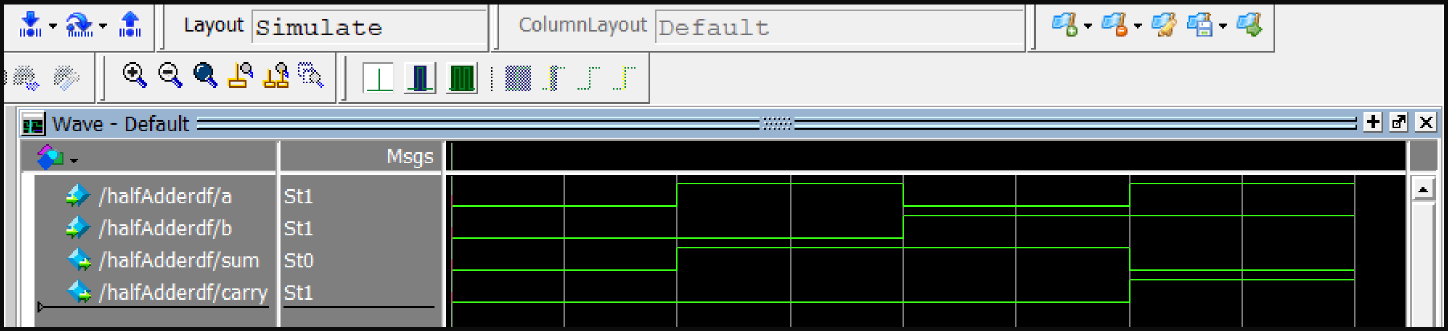
Name	Value	Kind	Mode
a	-No D...	Net	In
b	-No D...	Net	In
sum	-No D...	Net	Out
carry	-No D...	Net	Out

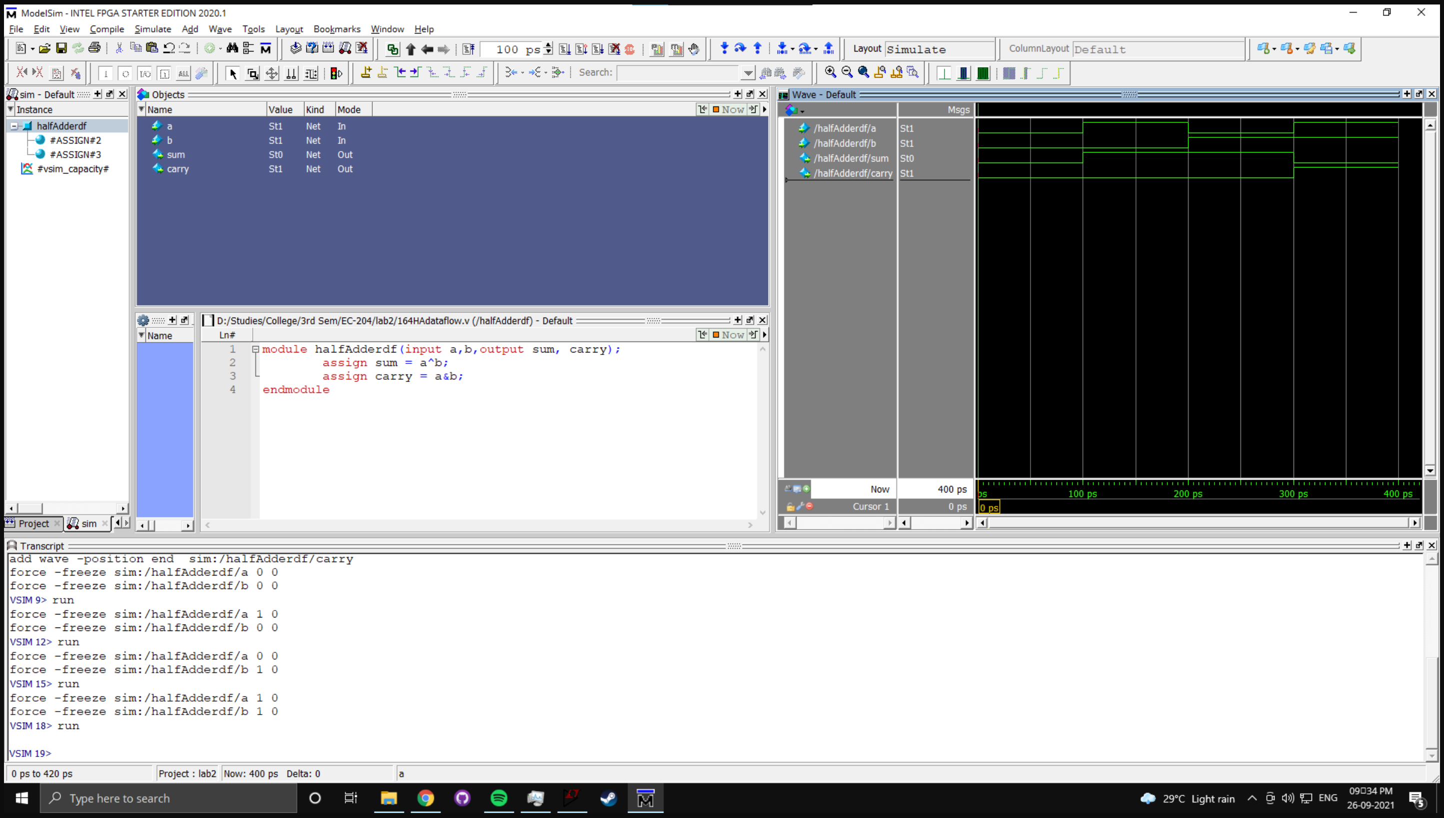
```
VSIM 44> run
force -freeze sim:/halfAdder/a 0 0
VSIM 46> run

VSIM 47>
```

2 ps to 433 ps | Project : lab2 | Now: 400 ps Delta: 0 | sim:/halfAdder

(b) using dataflow model

```verilog
module halfAdderdf(input a,b,output sum, carry);
    assign sum = a^b;
    assign carry = a&b;
endmodule
```


```
/halfAdderdf/a       St1
/halfAdderdf/b       St1
/halfAdderdf/sum     St0
/halfAdderdf/carry   St1
```


Name	Value	Kind	Mode
a	St1	Net	In
b	St1	Net	In
sum	St0	Net	Out
carry	St1	Net	Out

D:/Studies/College/3rd Sem/EC-204/lab2/164HAdataflow.v (/halfAdderdf) - Default

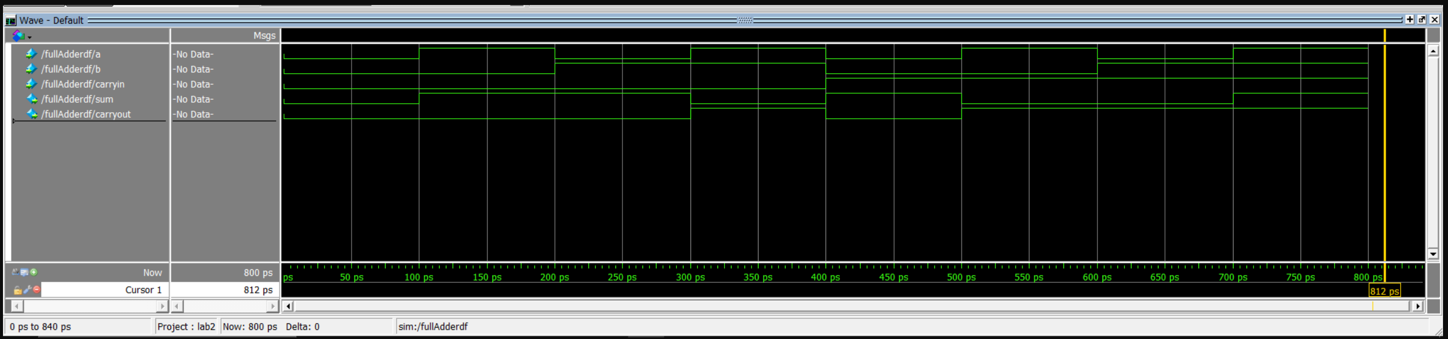
```
module halfAdderdf(input a,b,output sum, carry);
        assign sum = a^b;
        assign carry = a&b;
endmodule
```

```
add wave -position end  sim:/halfAdderdf/carry
force -freeze sim:/halfAdderdf/a 0 0
force -freeze sim:/halfAdderdf/b 0 0
VSIM 9> run
force -freeze sim:/halfAdderdf/a 1 0
force -freeze sim:/halfAdderdf/b 0 0
VSIM 12> run
force -freeze sim:/halfAdderdf/a 0 0
force -freeze sim:/halfAdderdf/b 1 0
VSIM 15> run
force -freeze sim:/halfAdderdf/a 1 0
force -freeze sim:/halfAdderdf/b 1 0
VSIM 18> run

VSIM 19>
```

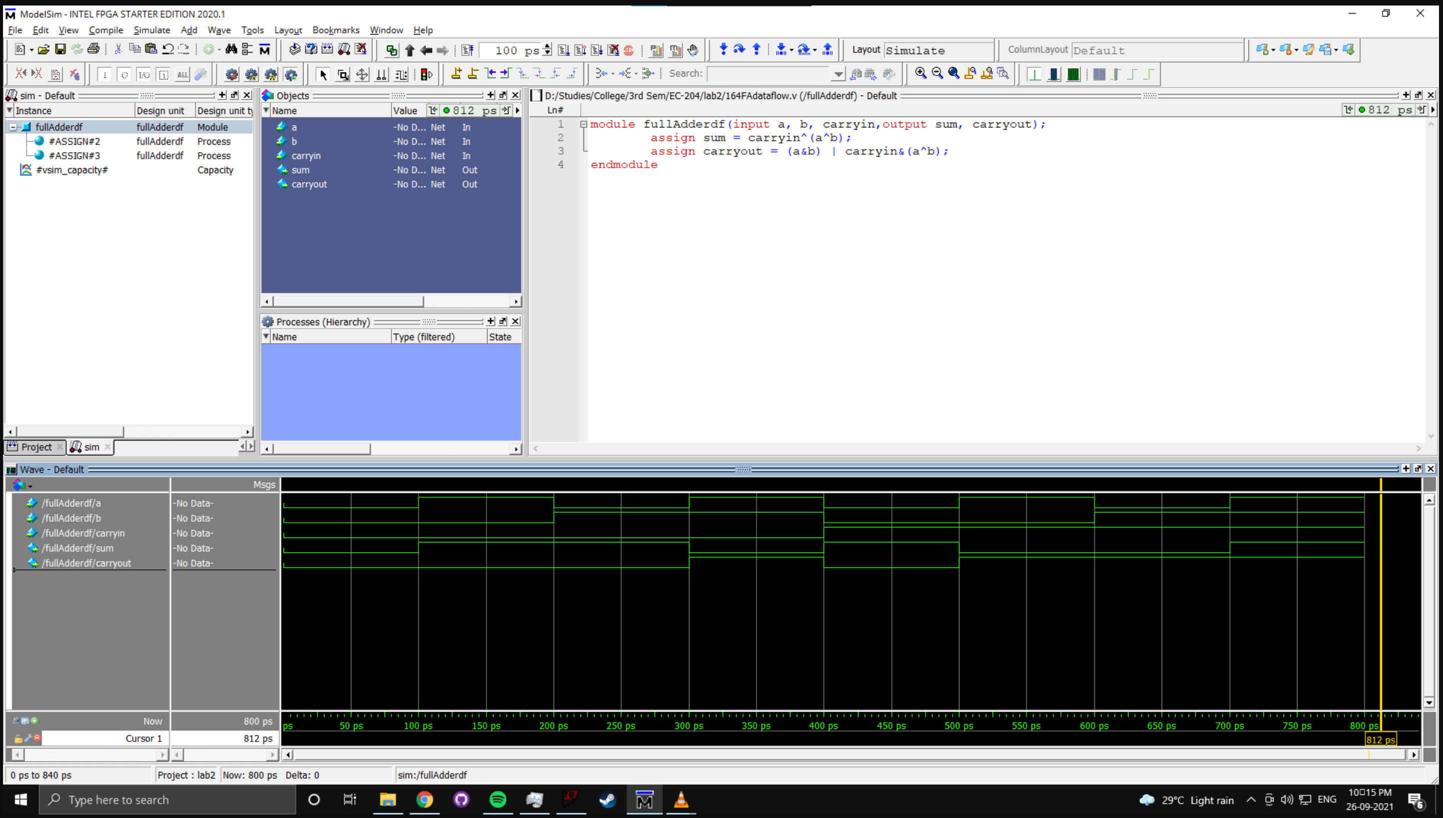
2] Full adder (a) using dataflow model (using assign statement) (b) using two half adders and an OR gate

(a) using dataflow model

```
module fullAdderdf(input a, b, carryin,output sum, carryout);
    assign sum = carryin^(a^b);
    assign carryout = (a&b) | carryin&(a^b);
endmodule
```



```
/fullAdderdf/a          -No Data-
/fullAdderdf/b          -No Data-
/fullAdderdf/carryin    -No Data-
/fullAdderdf/sum        -No Data-
/fullAdderdf/carryout   -No Data-
Now        800 ps
Cursor 1   812 ps
0 ps to 840 ps    Project : lab2   Now: 800 ps   Delta: 0    sim:/fullAdderdf
```



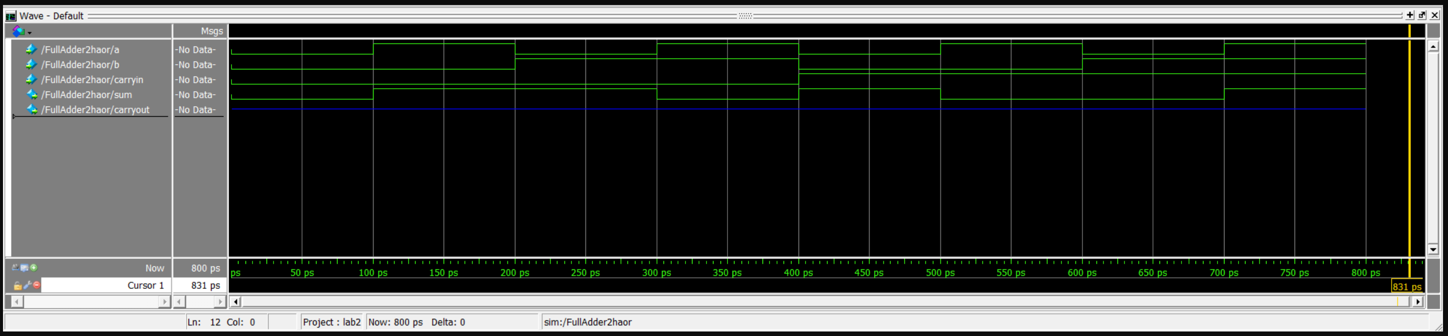
```
D:/Studies/College/3rd Sem/EC-204/lab2/164FAdataflow.v (/fullAdderdf) - Default
module fullAdderdf(input a, b, carryin,output sum, carryout);
        assign sum = carryin^(a^b);
        assign carryout = (a&b) | carryin&(a^b);
endmodule
```

```
Instance          Design unit   Design unit type
fullAdderdf       fullAdderdf   Module
  #ASSIGN#2       fullAdderdf   Process
  #ASSIGN#3       fullAdderdf   Process
#vsim_capacity#                 Capacity

Name       Value      Kind  Mode
a          -No D...   Net   In
b          -No D...   Net   In
carryin    -No D...   Net   In
sum        -No D...   Net   Out
carryout   -No D...   Net   Out
```

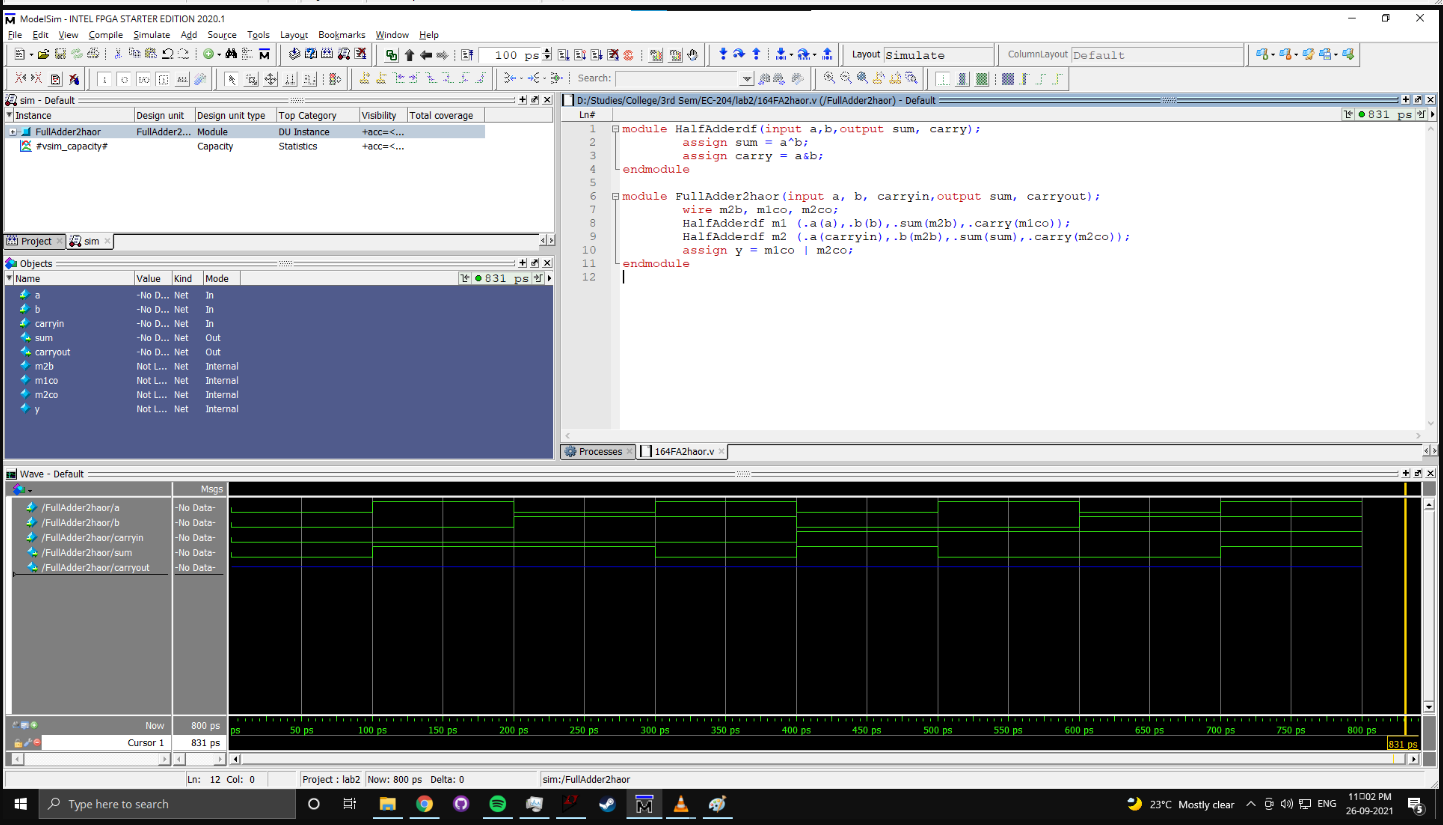
```
/fullAdderdf/a          -No Data-
/fullAdderdf/b          -No Data-
/fullAdderdf/carryin    -No Data-
/fullAdderdf/sum        -No Data-
/fullAdderdf/carryout   -No Data-
Now        800 ps
Cursor 1   812 ps
0 ps to 840 ps    Project : lab2   Now: 800 ps   Delta: 0    sim:/fullAdderdf
```

(b) using two half adders and an OR gate

```
module HalfAdderdf(input a,b,output sum, carry);
    assign sum = a^b;
    assign carry = a&b;
endmodule

module FullAdder2haor(input a, b, carryin,output sum, carryout);
    wire m2b, m1co, m2co;
    HalfAdderdf m1 (.a(a),.b(b),.sum(m2b),.carry(m1co));
    HalfAdderdf m2 (.a(carryin),.b(m2b),.sum(sum),.carry(m2co));
    assign y = m1co | m2co;
endmodule
```


Wave - Default

```
/FullAdder2haor/a         -No Data-
/FullAdder2haor/b         -No Data-
/FullAdder2haor/carryin   -No Data-
/FullAdder2haor/sum       -No Data-
/FullAdder2haor/carryout  -No Data-

Now       800 ps
Cursor 1  831 ps
```


ModelSim - INTEL FPGA STARTER EDITION 2020.1

Instance	Design unit	Design unit type	Top Category	Visibility	Total coverage
FullAdder2haor	FullAdder2...	Module	DU Instance	+acc=<...	
#vsim_capacity#		Capacity	Statistics	+acc=<...	

Name	Value	Kind	Mode
a	-No D...	Net	In
b	-No D...	Net	In
carryin	-No D...	Net	In
sum	-No D...	Net	Out
carryout	-No D...	Net	Out
m2b	Not L...	Net	Internal
m1co	Not L...	Net	Internal
m2co	Not L...	Net	Internal
y	Not L...	Net	Internal

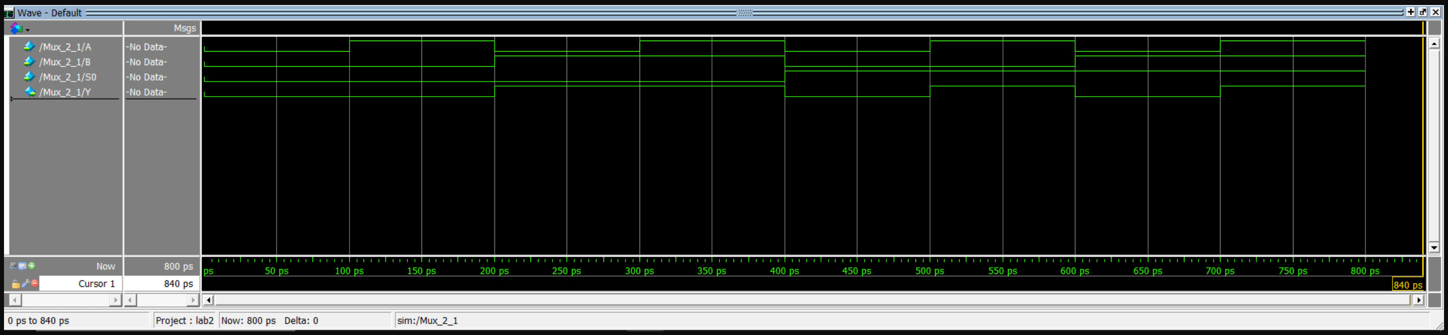
D:/Studies/College/3rd Sem/EC-204/lab2/164FA2haor.v (/FullAdder2haor) - Default

```
module HalfAdderdf(input a,b,output sum, carry);
        assign sum = a^b;
        assign carry = a&b;
endmodule

module FullAdder2haor(input a, b, carryin,output sum, carryout);
        wire m2b, m1co, m2co;
        HalfAdderdf m1  (.a(a),.b(b),.sum(m2b),.carry(m1co));
        HalfAdderdf m2  (.a(carryin),.b(m2b),.sum(sum),.carry(m2co));
        assign y = m1co | m2co;
endmodule
```

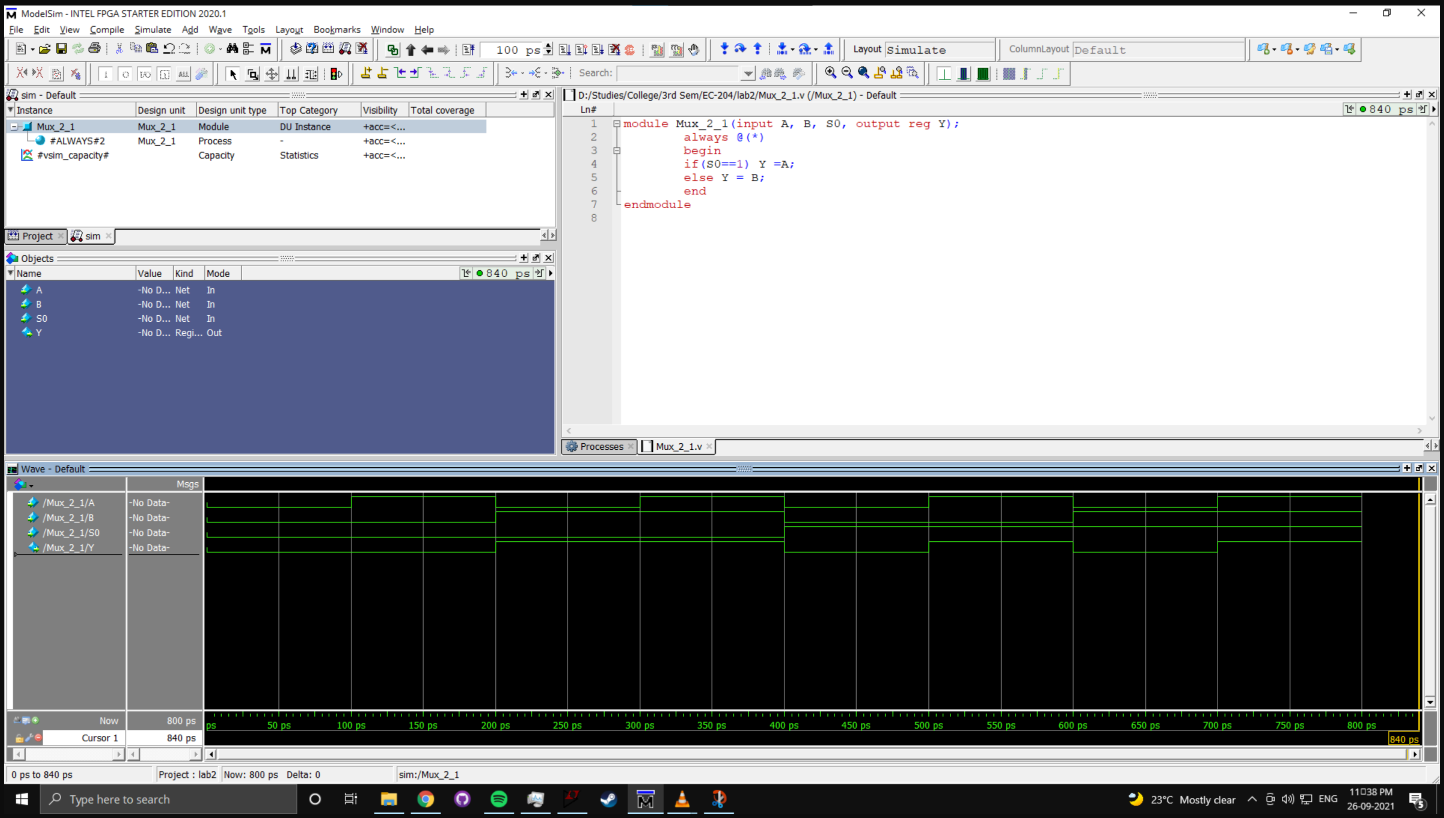
Project : lab2 Now: 800 ps Delta: 0 sim:/FullAdder2haor

3] Two to one multiplexer using behavioral model (using if statement)

```
module Mux_2_1(input A, B, S0, output reg Y);
    always @(*)
    begin
    if(S0==1) Y =A;
    else Y = B;
    end
endmodule
```



```
Wave - Default
/Mux_2_1/A    -No Data-
/Mux_2_1/B    -No Data-
/Mux_2_1/S0   -No Data-
/Mux_2_1/Y    -No Data-
Now       800 ps
Cursor 1  840 ps
0 ps to 840 ps    Project : lab2  Now: 800 ps  Delta: 0    sim:/Mux_2_1
```



```
ModelSim - INTEL FPGA STARTER EDITION 2020.1

sim - Default
Instance          Design unit   Design unit type   Top Category   Visibility
Mux_2_1           Mux_2_1       Module             DU Instance    +acc=<...
  #ALWAYS#2       Mux_2_1       Process            -              +acc=<...
#vsim_capacity#                 Capacity           Statistics     +acc=<...

Objects
Name   Value       Kind     Mode
A      -No D...    Net      In
B      -No D...    Net      In
S0     -No D...    Net      In
Y      -No D...    Regi...  Out
```

D:/Studies/College/3rd Sem/EC-204/lab2/Mux_2_1.v (/Mux_2_1) - Default

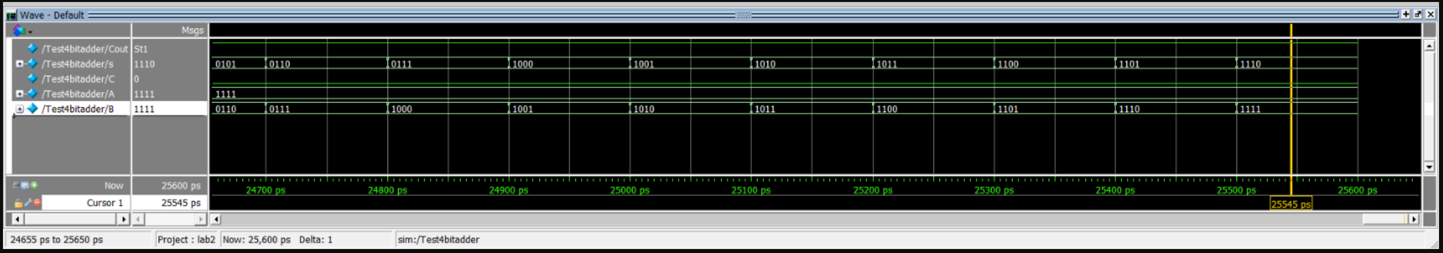
```verilog
module Mux_2_1(input A, B, S0, output reg Y);
        always @(*)
        begin
        if(S0==1) Y =A;
        else Y = B;
        end
endmodule
```

```
Wave - Default
/Mux_2_1/A    -No Data-
/Mux_2_1/B    -No Data-
/Mux_2_1/S0   -No Data-
/Mux_2_1/Y    -No Data-
Now       800 ps
Cursor 1  840 ps
0 ps to 840 ps    Project : lab2  Now: 800 ps  Delta: 0    sim:/Mux_2_1
```

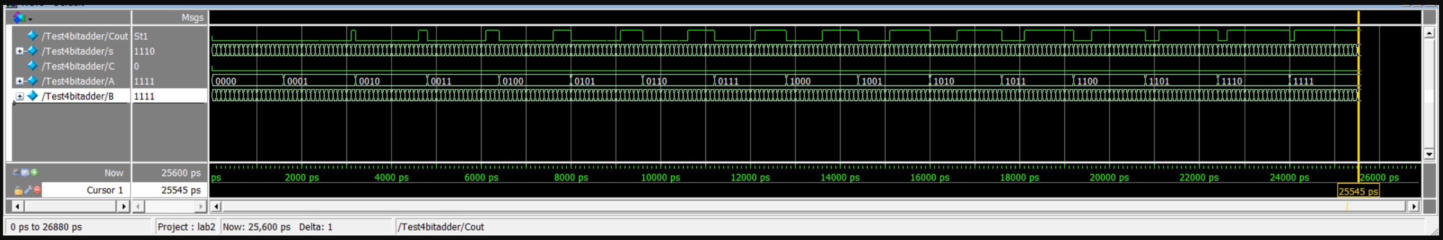
4] 4 bit ripple carry adder using full adder modules.

```verilog
module FullAdder(input A, B, Ci,output S, Co);
    assign S = Ci^(A^B);
    assign Co = (A&B) | Ci&(A^B);
endmodule

module FourBitAdder(input [3:0]A, B,
            input Ci,
            output [3:0]S,
            output Co);
    wire  C1, C2,C3;
    FullAdder FA1(.A(A[0]),.B(B[0]),.Ci(Ci),.S(S[0]),.Co(C1));
    FullAdder FA2(.A(A[1]),.B(B[1]),.Ci(C1),.S(S[1]),.Co(c2));
    FullAdder FA3(.A(A[2]),.B(B[2]),.Ci(c2),.S(S[2]),.Co(c3));
    FullAdder FA4(.A(A[3]),.B(B[3]),.Ci(c3),.S(S[3]),.Co(Co));

endmodule


module Test4bitadder();
/*201ec164*/
    reg  [1:0] ci ;
    integer ca, cb;/* ca and cb are counts*/
    wire Cout;
    wire [3:0] s;
    reg C;
    reg [3:0] A, B;
    reg [3:0] array[0:15];
    assign array[0] = 4'b0000; assign array[1] = 4'b0001;
    assign array[2] = 4'b0010; assign array[3] = 4'b0011;
    assign array[4] = 4'b0100; assign array[5] = 4'b0101;
    assign array[6] = 4'b0110; assign array[7] = 4'b0111;
    assign array[8] = 4'b1000; assign array[9] = 4'b1001;
    assign array[10] = 4'b1010; assign array[11] = 4'b1011;
    assign array[12] = 4'b1100; assign array[13] = 4'b1101;
    assign array[14] = 4'b1110; assign array[15] = 4'b1111;
    assign C=0;
    FourBitAdder test(.A(A), .B(B), .Ci(C), .S(s), .Co(Cout));
    initial
    begin   for( ca =0; ca<16; ca=ca+1)
            begin   for(cb=0; cb<16; cb=cb+1)
                begin
                A=array[ca];
```

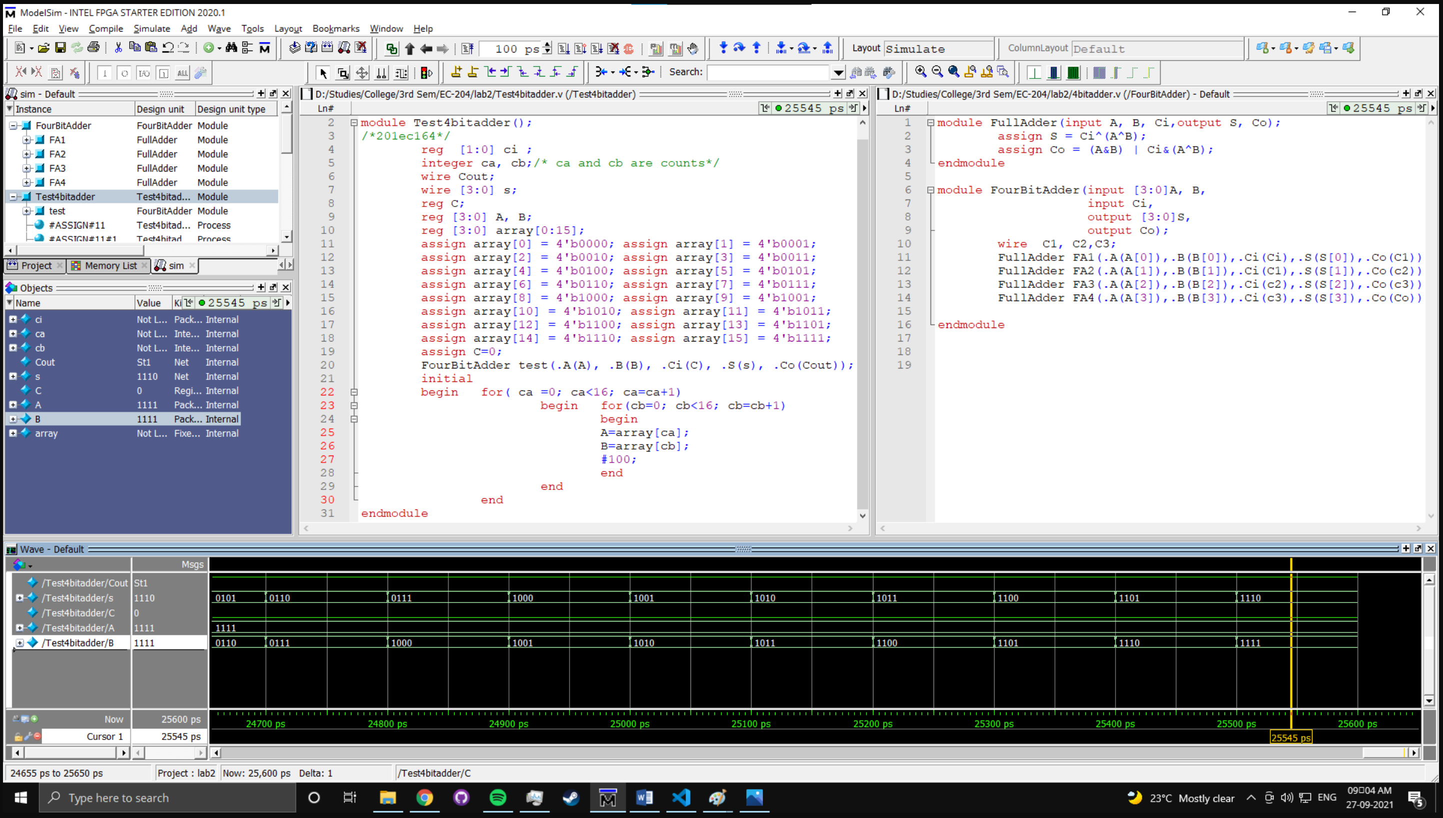
```
                    B=array[cb];
                    #100;
                    end
                end
            end
endmodule
```



```
Wave - Default
                          Msgs
/Test4bitadder/Cout   St1
/Test4bitadder/s      1110   0101 0110 0111 1000 1001 1010 1011 1100 1101 1110
/Test4bitadder/C      0
/Test4bitadder/A      1111   1111
/Test4bitadder/B      1111   0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
Now        25600 ps   24700 ps  24800 ps  24900 ps  25000 ps  25100 ps  25200 ps  25300 ps  25400 ps  25500 ps  25600 ps
Cursor 1   25545 ps   25545 ps
24655 ps to 25650 ps   Project : lab2   Now: 25,600 ps   Delta: 1   sim:/Test4bitadder
```



```
                          Msgs
/Test4bitadder/Cout   St1
/Test4bitadder/s      1110
/Test4bitadder/C      0
/Test4bitadder/A      1111   0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
/Test4bitadder/B      1111
Now        25600 ps   ps  2000 ps  4000 ps  6000 ps  8000 ps  10000 ps  12000 ps  14000 ps  16000 ps  18000 ps  20000 ps  22000 ps  24000 ps  26000 ps
Cursor 1   25545 ps   25545 ps
0 ps to 26880 ps   Project : lab2   Now: 25,600 ps   Delta: 1   /Test4bitadder/Cout
```



ModelSim - INTEL FPGA STARTER EDITION 2020.1

```
sim - Default
Instance         Design unit      Design unit type
FourBitAdder     FourBitAdder     Module
  FA1            FullAdder        Module
  FA2            FullAdder        Module
  FA3            FullAdder        Module
  FA4            FullAdder        Module
Test4bitadder    Test4bitad...    Module
  test           FourBitAdder     Module
  #ASSIGN#11     Test4bitad...    Process

Objects
Name    Value    Ki    
ci      Not L... Pack... Internal
ca      Not L... Inte... Internal
cb      Not L... Inte... Internal
Cout    St1      Net     Internal
s       1110     Net     Internal
C       0        Regi... Internal
A       1111     Pack... Internal
B       1111     Pack... Internal
array   Not L... Fixe... Internal
```

D:/Studies/College/3rd Sem/EC-204/lab2/Test4bitadder.v (/Test4bitadder)

```
module Test4bitadder();
/*201ec164*/
        reg  [1:0] ci ;
        integer ca, cb;/* ca and cb are counts*/
        wire Cout;
        wire [3:0] s;
        reg C;
        reg [3:0] A, B;
        reg [3:0] array[0:15];
        assign array[0] = 4'b0000; assign array[1] = 4'b0001;
        assign array[2] = 4'b0010; assign array[3] = 4'b0011;
        assign array[4] = 4'b0100; assign array[5] = 4'b0101;
        assign array[6] = 4'b0110; assign array[7] = 4'b0111;
        assign array[8] = 4'b1000; assign array[9] = 4'b1001;
        assign array[10] = 4'b1010; assign array[11] = 4'b1011;
        assign array[12] = 4'b1100; assign array[13] = 4'b1101;
        assign array[14] = 4'b1110; assign array[15] = 4'b1111;
        assign C=0;
        FourBitAdder test(.A(A), .B(B), .Ci(C), .S(s), .Co(Cout));
        initial
        begin   for( ca =0; ca<16; ca=ca+1)
                    begin   for(cb=0; cb<16; cb=cb+1)
                            begin
                            A=array[ca];
                            B=array[cb];
                            #100;
                            end
                    end
                end
endmodule
```

D:/Studies/College/3rd Sem/EC-204/lab2/4bitadder.v (/FourBitAdder) - Default

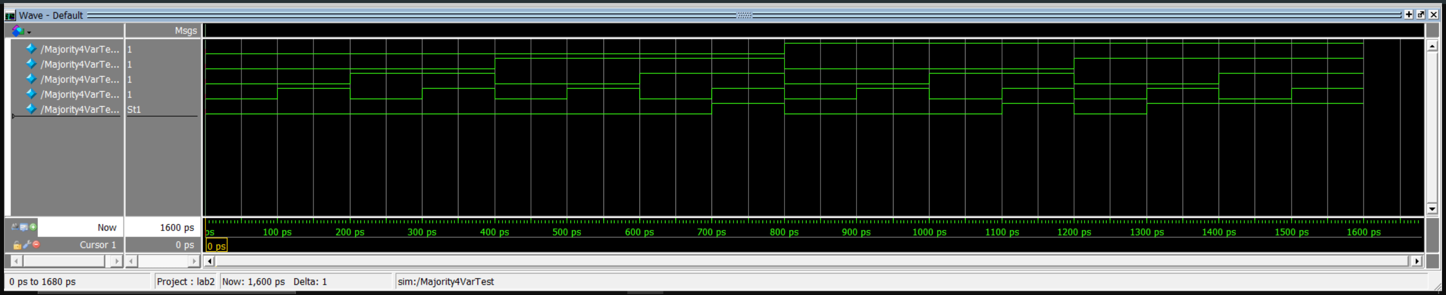
```
module FullAdder(input A, B, Ci,output S, Co);
        assign S = Ci^(A^B);
        assign Co = (A&B) | Ci&(A^B);
endmodule

module FourBitAdder(input [3:0]A, B,
                    input Ci,
                    output [3:0]S,
                    output Co);
        wire  C1, C2,C3;
        FullAdder FA1(.A(A[0]),.B(B[0]),.Ci(Ci),.S(S[0]),.Co(C1))
        FullAdder FA2(.A(A[1]),.B(B[1]),.Ci(C1),.S(S[1]),.Co(c2))
        FullAdder FA3(.A(A[2]),.B(B[2]),.Ci(c2),.S(S[2]),.Co(c3))
        FullAdder FA4(.A(A[3]),.B(B[3]),.Ci(c3),.S(S[3]),.Co(Co))

endmodule
```

```
Wave - Default
                          Msgs
/Test4bitadder/Cout   St1
/Test4bitadder/s      1110   0101 0110 0111 1000 1001 1010 1011 1100 1101 1110
/Test4bitadder/C      0
/Test4bitadder/A      1111   1111
/Test4bitadder/B      1111   0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
Now        25600 ps   24700 ps  24800 ps  24900 ps  25000 ps  25100 ps  25200 ps  25300 ps  25400 ps  25500 ps  25600 ps
Cursor 1   25545 ps   25545 ps
24655 ps to 25650 ps   Project : lab2   Now: 25,600 ps   Delta: 1   /Test4bitadder/C
```

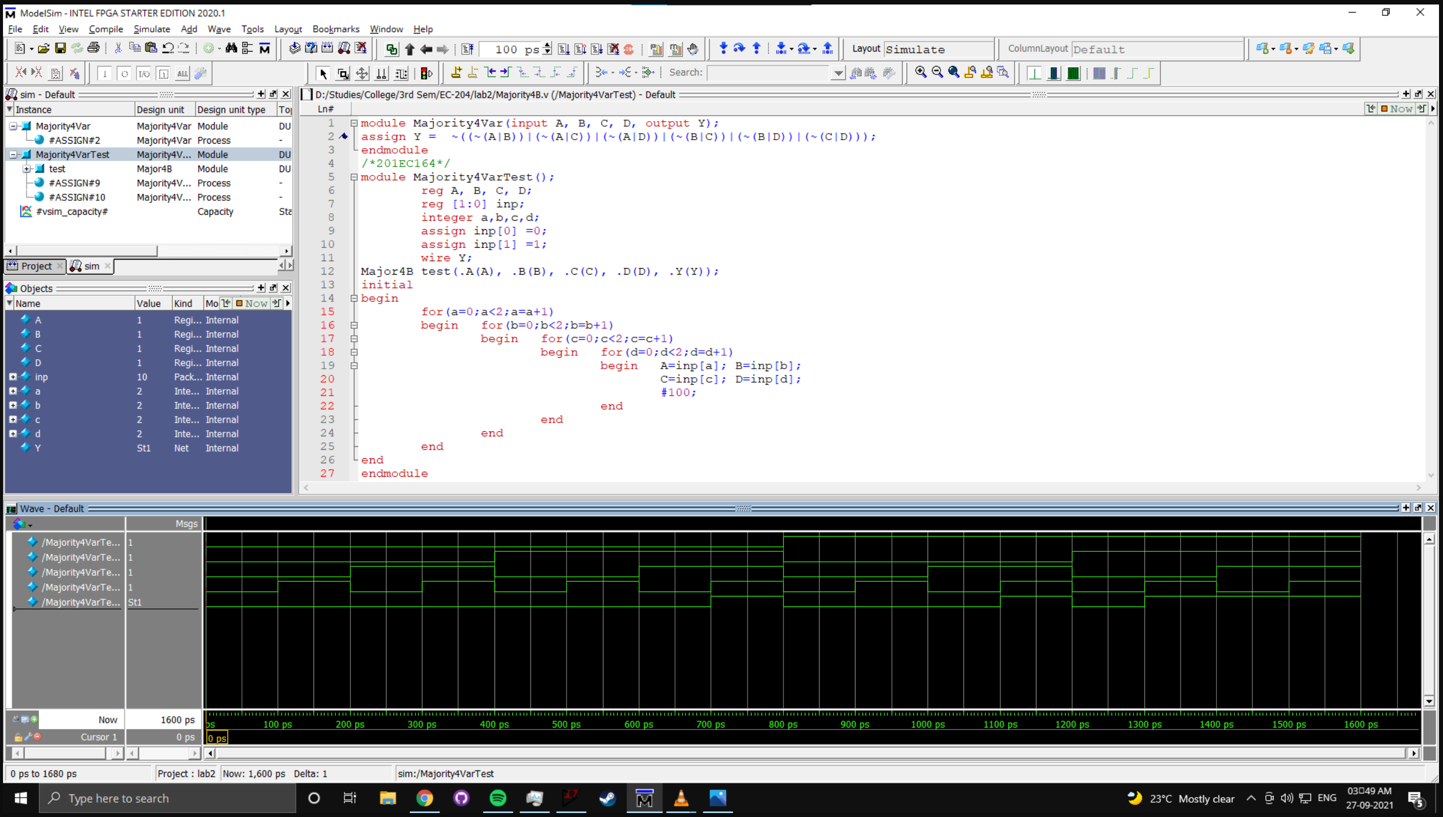
23°C Mostly clear   ENG   09:04 AM   27-09-2021

5] A four variable logic function that is equal to 1 if any three or all four of its variables are equal to 1 is called a majority function. Write a Verilog code that implements this majority function. Use the Boolean equation derived in Lab1 in assign statement.

```
module Majority4Var(input A, B, C, D, output Y);
assign Y =  ~((~(A|B))|(~(A|C))|(~(A|D))|(~(B|C))|(~(B|D))|(~(C|D)));
endmodule
/*201EC164*/
module Majority4VarTest();
    reg A, B, C, D;
    reg [1:0] inp;
    integer a,b,c,d;
    assign inp[0] =0;
    assign inp[1] =1;
    wire Y;
Major4B test(.A(A), .B(B), .C(C), .D(D), .Y(Y));
initial
begin
    for(a=0;a<2;a=a+1)
    begin   for(b=0;b<2;b=b+1)
        begin   for(c=0;c<2;c=c+1)
            begin   for(d=0;d<2;d=d+1)
                begin   A=inp[a]; B=inp[b];
                    C=inp[c]; D=inp[d];
                    #100;
                end
            end
        end
    end
end
endmodule
```



0 ps to 1680 ps   Project : lab2   Now: 1,600 ps Delta: 1   sim:/Majority4VarTest



ModelSim - INTEL FPGA STARTER EDITION 2020.1

sim - Default

Instance	Design unit	Design unit type	To
Majority4Var	Majority4Var	Module	DU
#ASSIGN#2	Majority4Var	Process	-
Majority4VarTest	Majority4V...	Module	DU
test	Major4B	Module	DU
#ASSIGN#9	Majority4V...	Process	-
#ASSIGN#10	Majority4V...	Process	-
#vsim_capacity#		Capacity	Sta

Objects

Name	Value	Kind	Mo
A	1	Regi...	Internal
B	1	Regi...	Internal
C	1	Regi...	Internal
D	1	Regi...	Internal
inp	10	Pack...	Internal
a	2	Inte...	Internal
b	2	Inte...	Internal
c	2	Inte...	Internal
d	2	Inte...	Internal
Y	St1	Net	Internal

D:/Studies/College/3rd Sem/EC-204/lab2/Majority4B.v (/Majority4VarTest) - Default

```
module Majority4Var(input A, B, C, D, output Y);
assign Y =  ~((~(A|B))|(~(A|C))|(~(A|D))|(~(B|C))|(~(B|D))|(~(C|D)));
endmodule
/*201EC164*/
module Majority4VarTest();
        reg A, B, C, D;
        reg [1:0] inp;
        integer a,b,c,d;
        assign inp[0] =0;
        assign inp[1] =1;
        wire Y;
Major4B test(.A(A), .B(B), .C(C), .D(D), .Y(Y));
initial
begin
        for(a=0;a<2;a=a+1)
        begin   for(b=0;b<2;b=b+1)
                begin   for(c=0;c<2;c=c+1)
                        begin   for(d=0;d<2;d=d+1)
                                begin   A=inp[a]; B=inp[b];
                                        C=inp[c]; D=inp[d];
                                        #100;
                                end
                        end
                end
        end
end
endmodule
```

Wave - Default

	Msgs
/Majority4VarTe...	1
/Majority4VarTe...	1
/Majority4VarTe...	1
/Majority4VarTe...	1
/Majority4VarTe...	St1

Now 1600 ps
Cursor 1 0 ps

0 ps to 1680 ps | Project : lab2 | Now: 1,600 ps | Delta: 1 | sim:/Majority4VarTest