Name: Rasha Mansour

ID:1210773

8.6 Task in lab

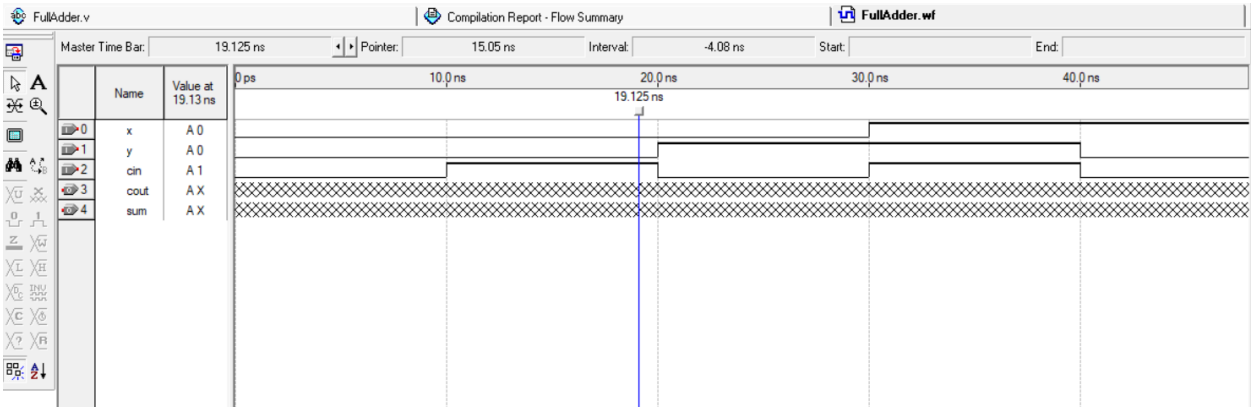
1. Create schematic symbols as shown in the figure, Use: Y3Y2Y1Y0 = 0111. What does this circuit do?

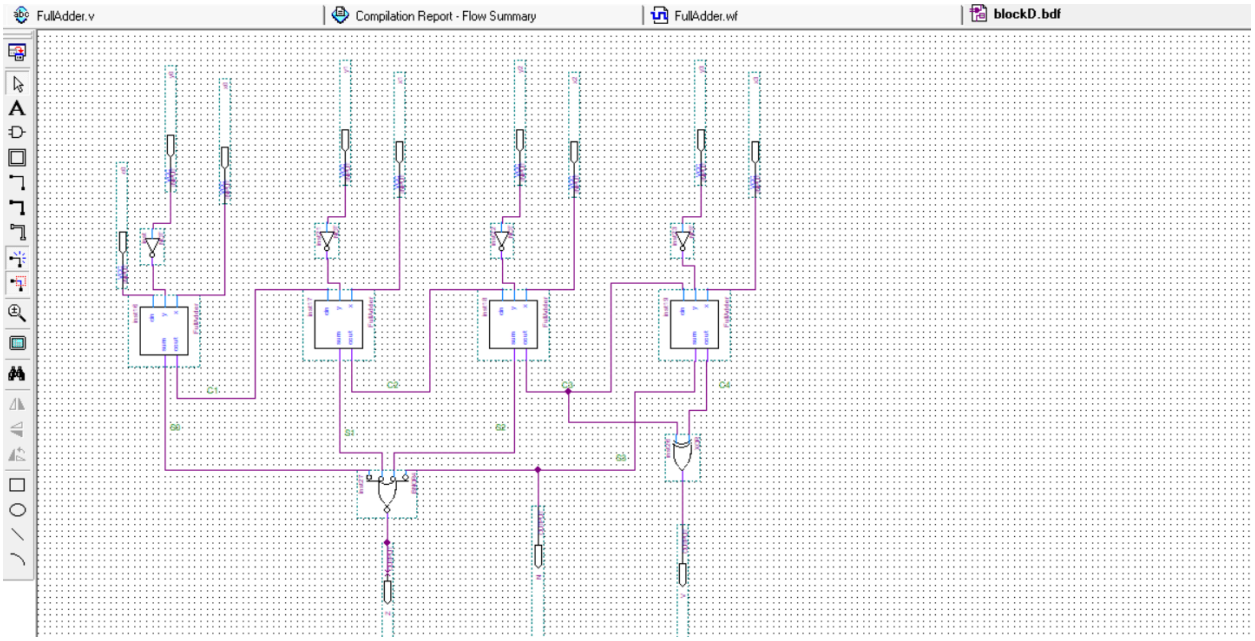
Use Verilog to implement the circuit and run a meaningful simulation for this circuit.

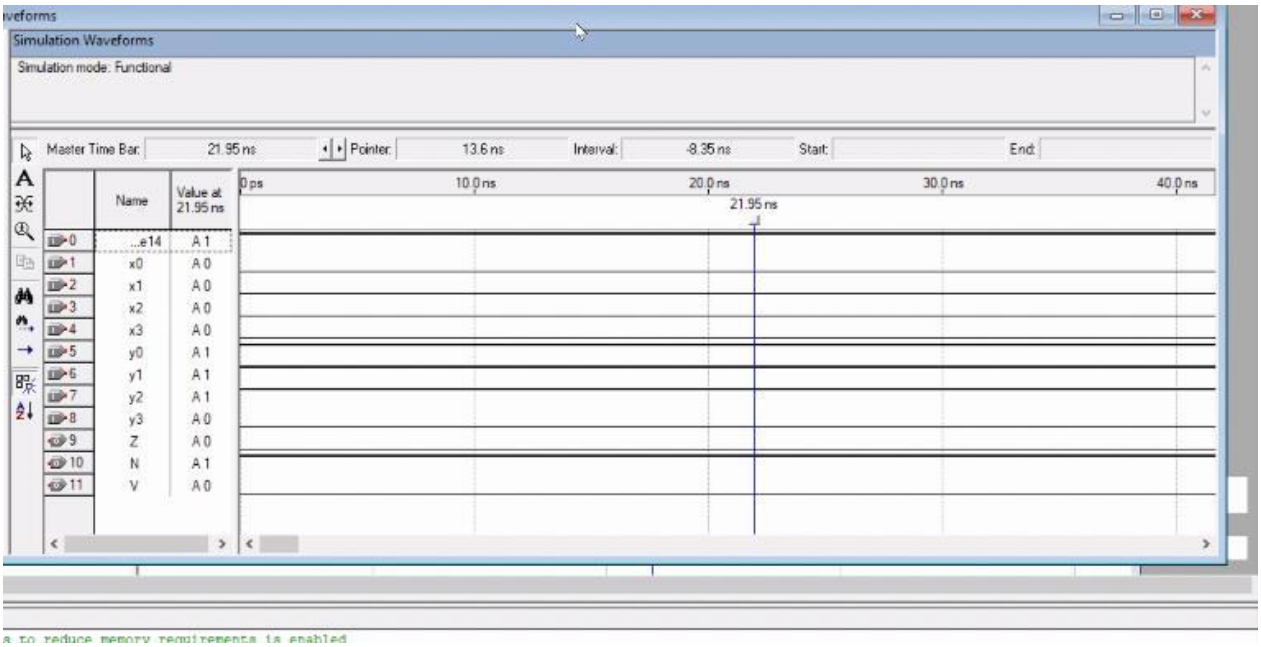
x+2s Complement y

x-y

```
1  module FullAdder(x,y,cin,cout,sum);
2    input x,y,cin;
3    output cout,sum;
4
5    assign sum=(x^y)^cin;
6    assign cout=(x&&y)||(cin&&(x&&y));
7
8
9    endmodule
```





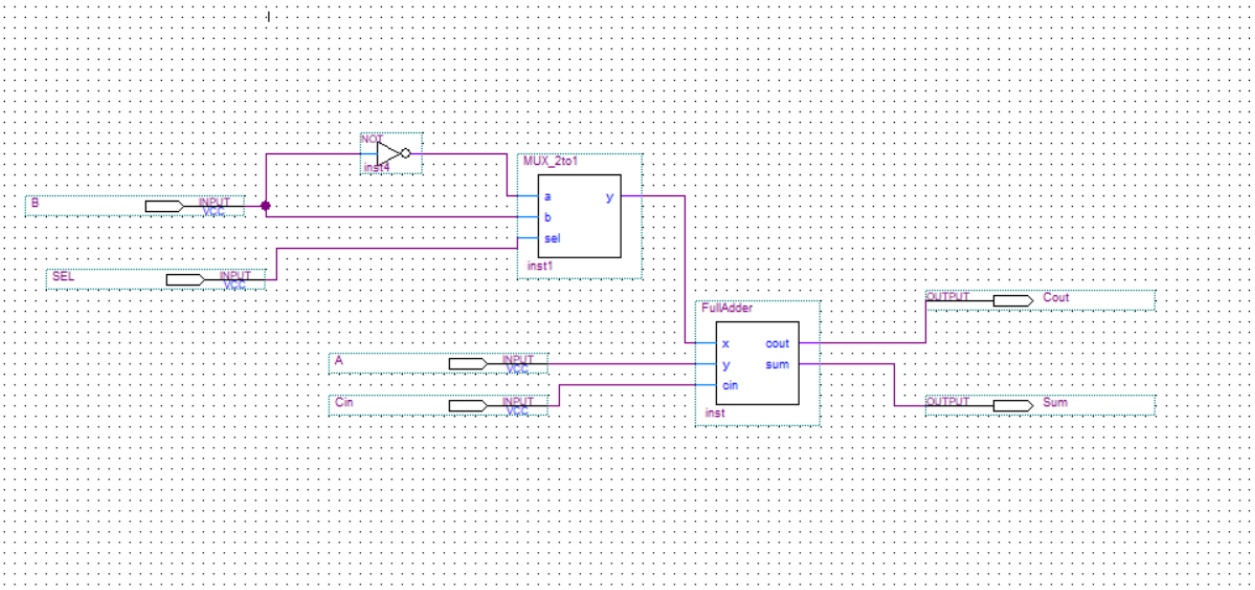


2. Use Verilog HDL to implement a 2-to-1 MUX. Use Verilog HDL to implement a Full Adder. Create schematic symbols for both the MUX and the Full adder, then connect them as shown in the figure. Run a meaningful simulation for this circuit.

FullAdder.v | Compilation Report - Flow Summ...

```
1  module MUX_2to1 (a,b,sel,y);
2  input a,b,sel;
3  output y;
4  wire a,b,sel,y;
5
6  assign y = (sel) ? b : a;
7
8  endmodule
```

```
1  module FullAdder(x,y,cin,cout,sum);
2  input x,y,cin;
3  output cout,sum;
4
5  assign sum=(x^y)^cin;
6  assign cout=(x&&y)||(cin&&(x&&y));
7
8
9  endmodule
```



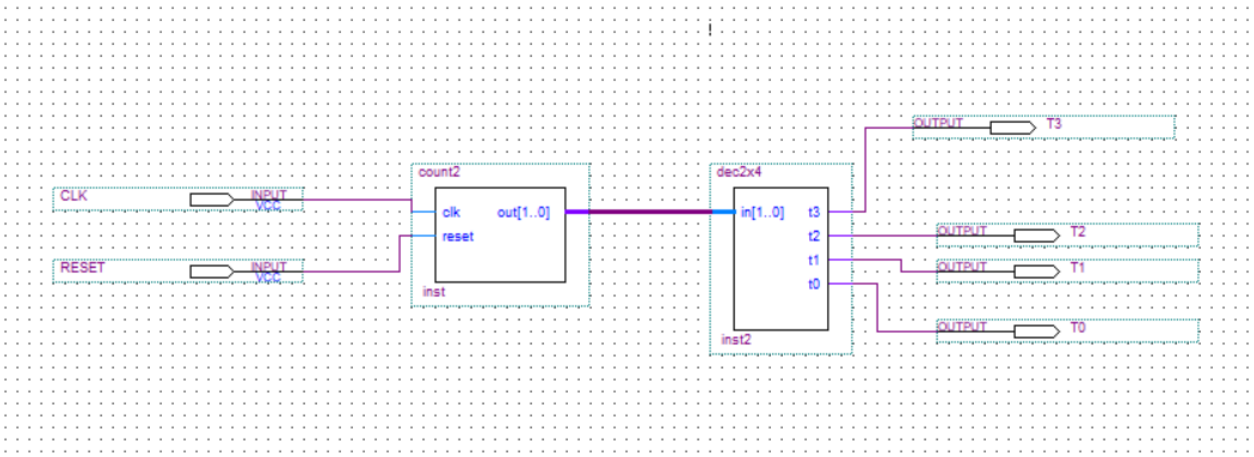
3. Use Verilog HDL to implement a 2-bit counter with direct reset input (RESET). Use Verilog HDL to implement a 2-to-4 Decoder. Create schematic symbols for both the counter and the decoder, then connect them as shown in figure below. Run a meaningful simulation for this circuit

FullAdder.v | Compilation Repo... | FullAdder.wf | block

```
1   module count2( clk,reset,out);
2   input clk, reset;
3   output reg [1:0] out;
4
5   always @ (posedge clk or posedge reset)
6   begin
7       if (reset)
8           out <= 2'd0;
9       else
10          out <= out + 2'd1;
11  end
12  endmodule
```

FullAdder.v | Compilation Repor... | FullAdder.wf | blockD.bdf | wff.v

```
1   module dec2x4(in, t3, t2, t1, t0);
2     input [1:0] in;
3     output reg t3, t2, t1, t0;
4
5     always @(*)
6     begin
7       if (in[1] == 1'b0 && in[0] == 1'b0) begin
8         t0 = 1'b1;
9         t1 = 1'b0;
10        t2 = 1'b0;
11        t3 = 1'b0;
12      end
13      else if (in[1] == 1'b0 && in[0] == 1'b1) begin
14        t0 = 1'b0;
15        t1 = 1'b1;
16        t2 = 1'b0;
17        t3 = 1'b0;
18      end
19      else if (in[1] == 1'b1 && in[0] == 1'b0) begin
20        t0 = 1'b0;
21        t1 = 1'b0;
22        t2 = 1'b1;
23        t3 = 1'b0;
24      end
25      else begin
26        t0 = 1'b0;
27        t1 = 1'b0;
28        t2 = 1'b0;
29        t3 = 1'b1;
30      end
31    end
32  endmodule
```


CLK
INPUT
VCC
RESET
INPUT
VCC
count2
clk
reset
out[1..0]
inst
dec2x4
in[1..0]
t3
t2
t1
t0
inst2
OUTPUT
T3
OUTPUT
T2
OUTPUT
T1
OUTPUT
T0