MSPA-01 Evaluation

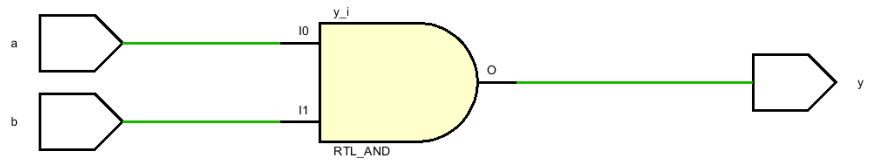
Name : **SHIVAM NITIN SALVE**	Roll No. **56**
Branch : **VLSI**	Sem **5**
Batch: **A4**	Reg no: **23070029**

Classroom Examples (5 Marks)

Code

```
22  //SHIVAM SALVE
23  module and_gate(y,a,b);
24  output wire y;
25  input wire a,b;
26  assign y=a&b;
27  endmodule
28
```

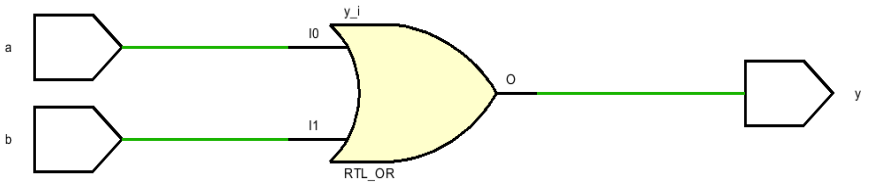
RTL View



Code

```
//SHIVAM SALVE
module or_gate(y,a,b);
output y;
input a,b;
assign y=a|b;
endmodule
```

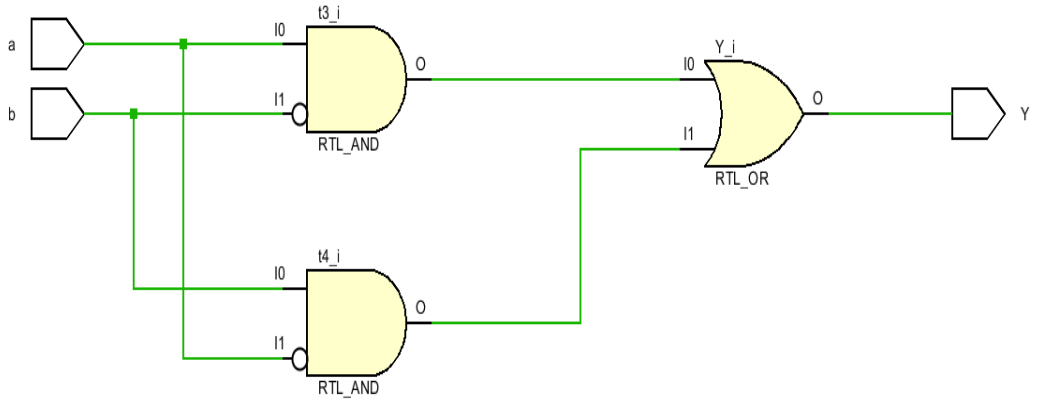
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module ex_or_gate(Y,a,b);
22  input a,b;
23  output Y;
24  wire t1,t2,t3,t4;
25  assign t1=~a;
26  assign t2=~b;
27  assign t3=a&t2;
28  assign t4=b&t1;
29  assign Y=t3|t4;
30  endmodule
31
```

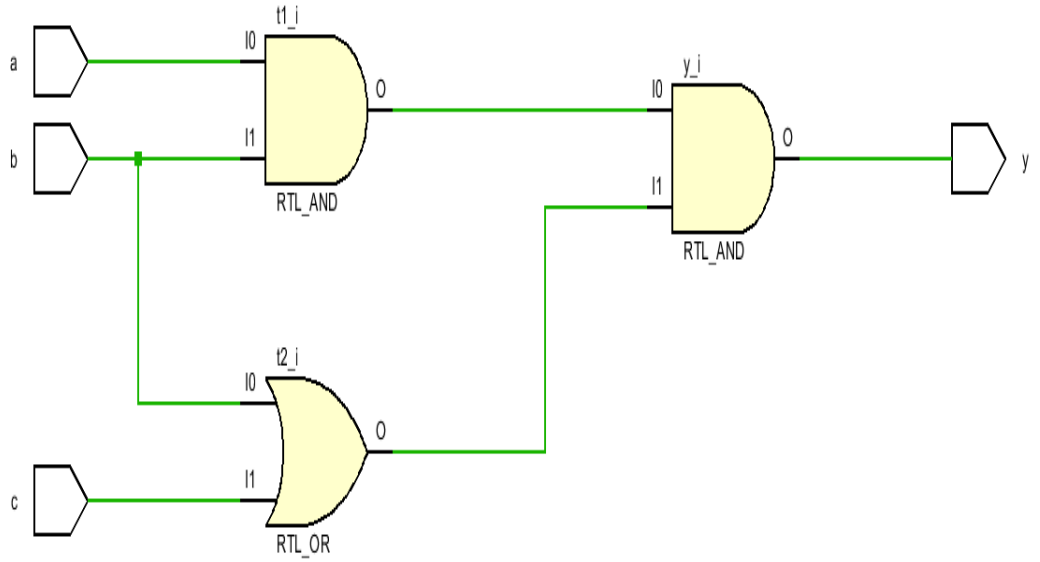
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module ckt_1(y,a,b,c);
22  output y;
23  input a,b,c;
24  wire t1,t2;
25  assign t1=a&b;
26  assign t2=b|c;
27  assign y=t1&t2;
28  endmodule
```

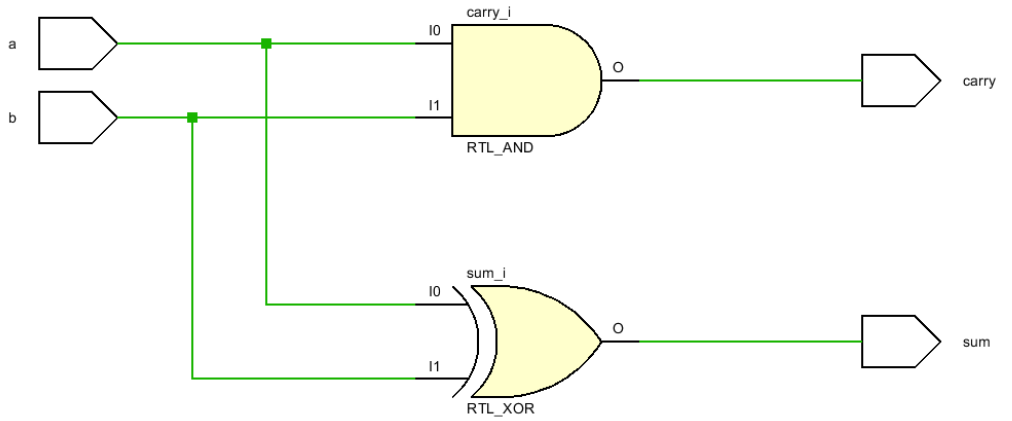
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module half_adder(sum,carry,a,b);
22  output sum,carry;
23  input a,b;
24  assign sum=a^b;
25  assign carry=a&b;
26  endmodule
27
```

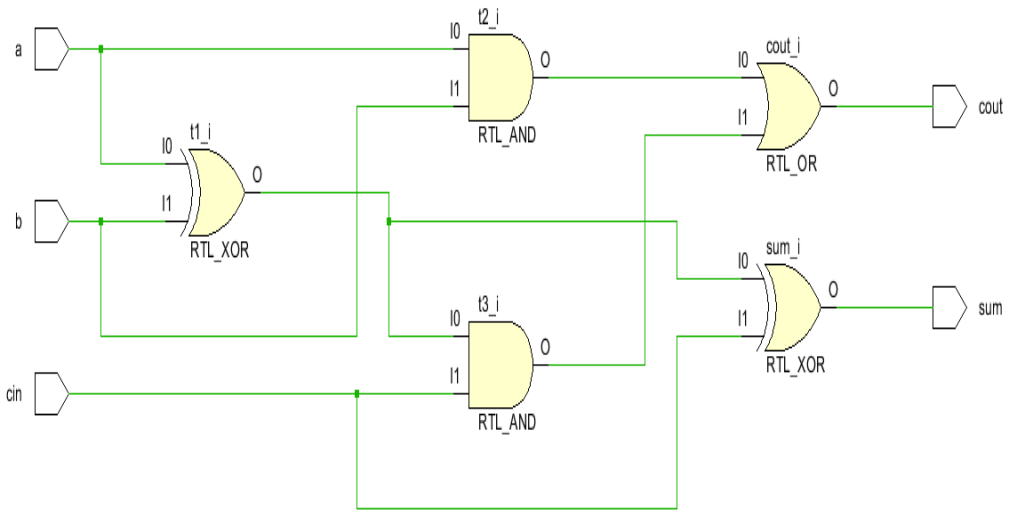
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module fullAdder(sum,cout,a,b,cin);
22  input a,b,cin;
23  output sum,cout;
24  wire t1,t2,t3;
25  assign t1=a^b;
26  assign t2=a&b;
27  assign sum=t1^cin;
28  assign t3=t1&cin;
29  assign cout=t2|t3;
30  endmodule
```

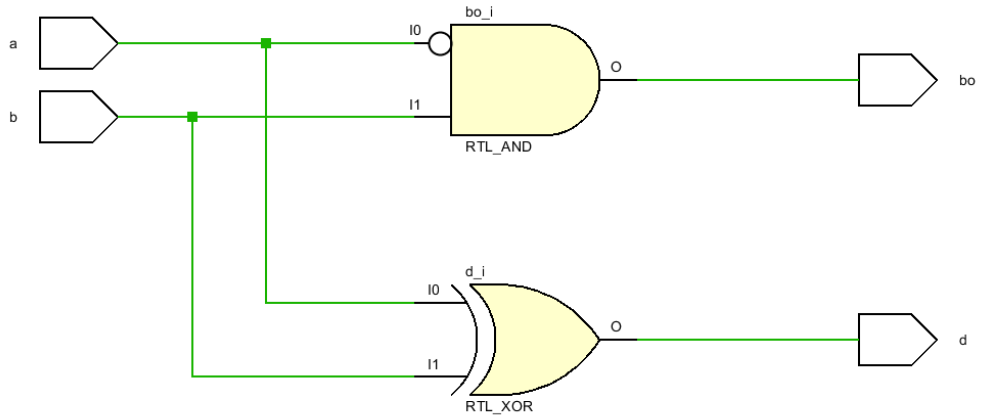
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module halfSub(d,bo,a,b);
22  input a,b;
23  output d,bo;
24  wire t1;
25  assign t1=~a;
26  assign d=a^b;
27  assign bo=t1&b;
28  endmodule
```

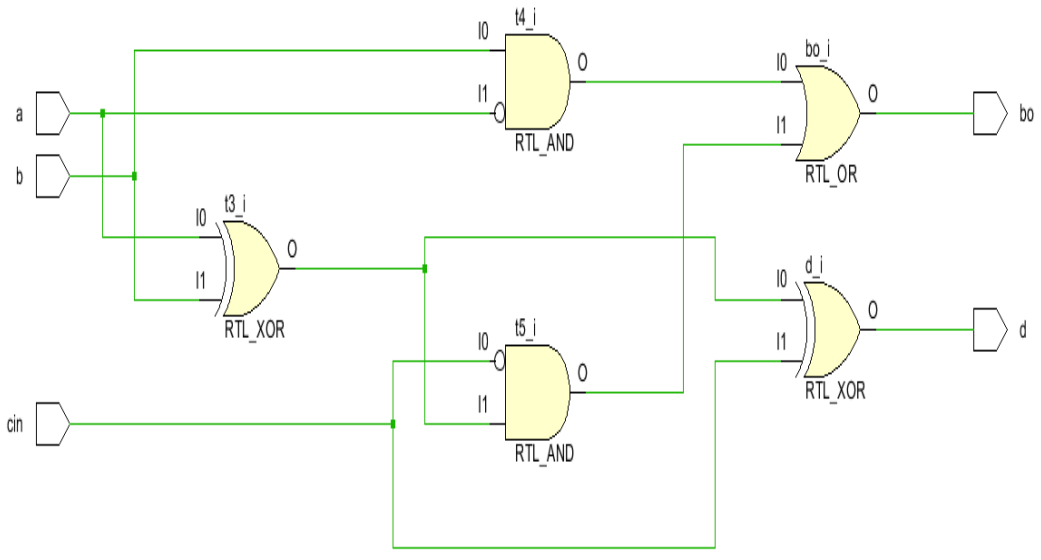
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module fullSub(d,bo,a,b,cin);
22  input a,b,cin;
23  output d,bo;
24      wire t1,t2,t3,t4,t5;
25      assign t1=~a;
26      assign t2=~cin;
27      assign t3=a^b;
28      assign d=t3^cin;
29      assign t4=b&t1;
30      assign t5=t2&t3;
31      assign bo=t4|t5;
32  endmodule
```

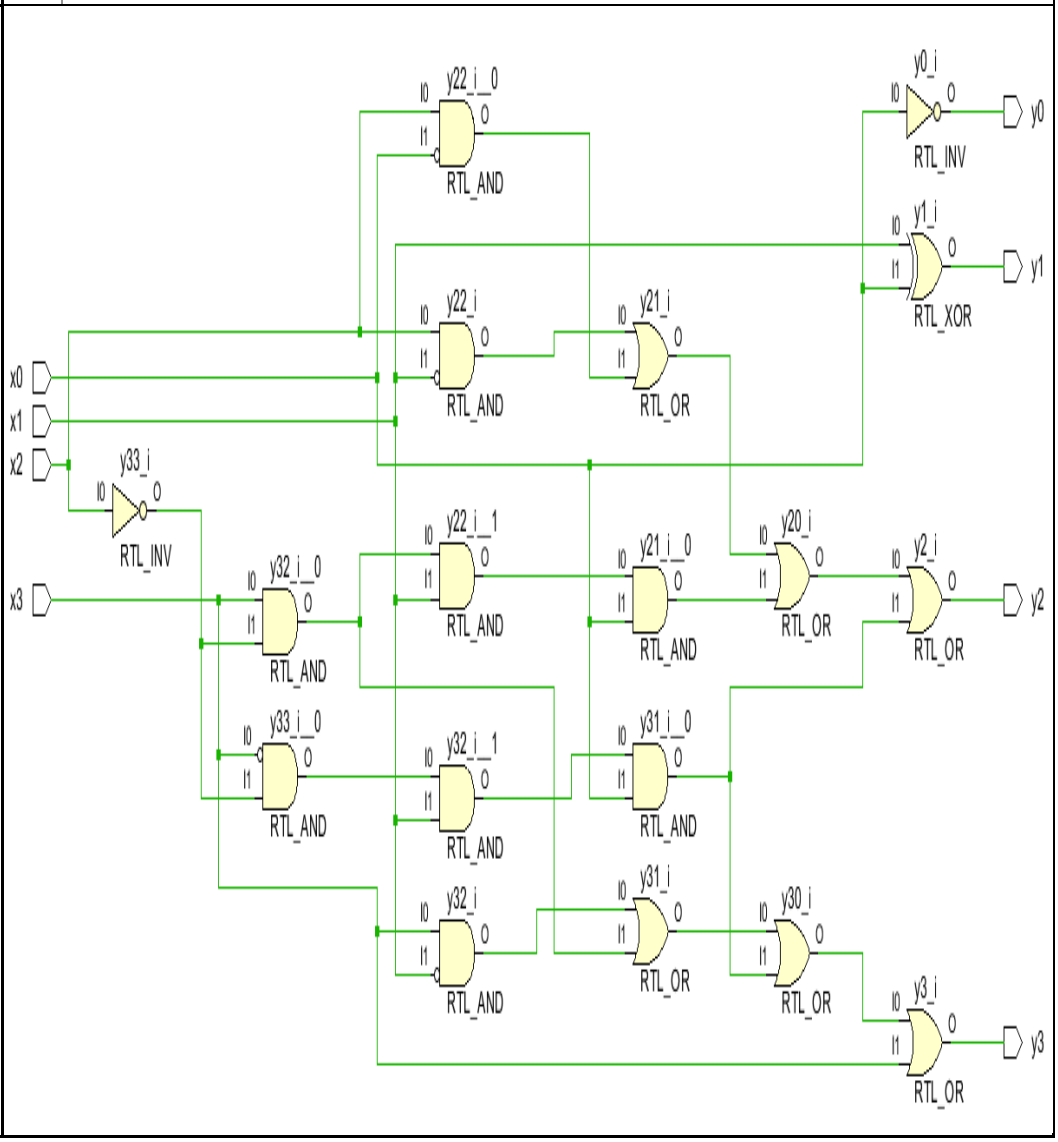
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module incremental_code(y3,y2,y1,y0,x3,x2,x1,x0);
22  output y3,y2,y1,y0;
23  input x3,x2,x1,x0;
24  assign y3=(x3&~ x1)|(x3&~x2)|((~x3)&(~x2)&x1&x0)|(x3);
25  assign y2=(x2&(~x1))|(x2&~x0)|((x3)&(~x2)&x1&x0)|((~x3)&(~x2)&x1&x0);
26  assign y1=x1^x0;
27  assign y0=~x0;
28  endmodule
```

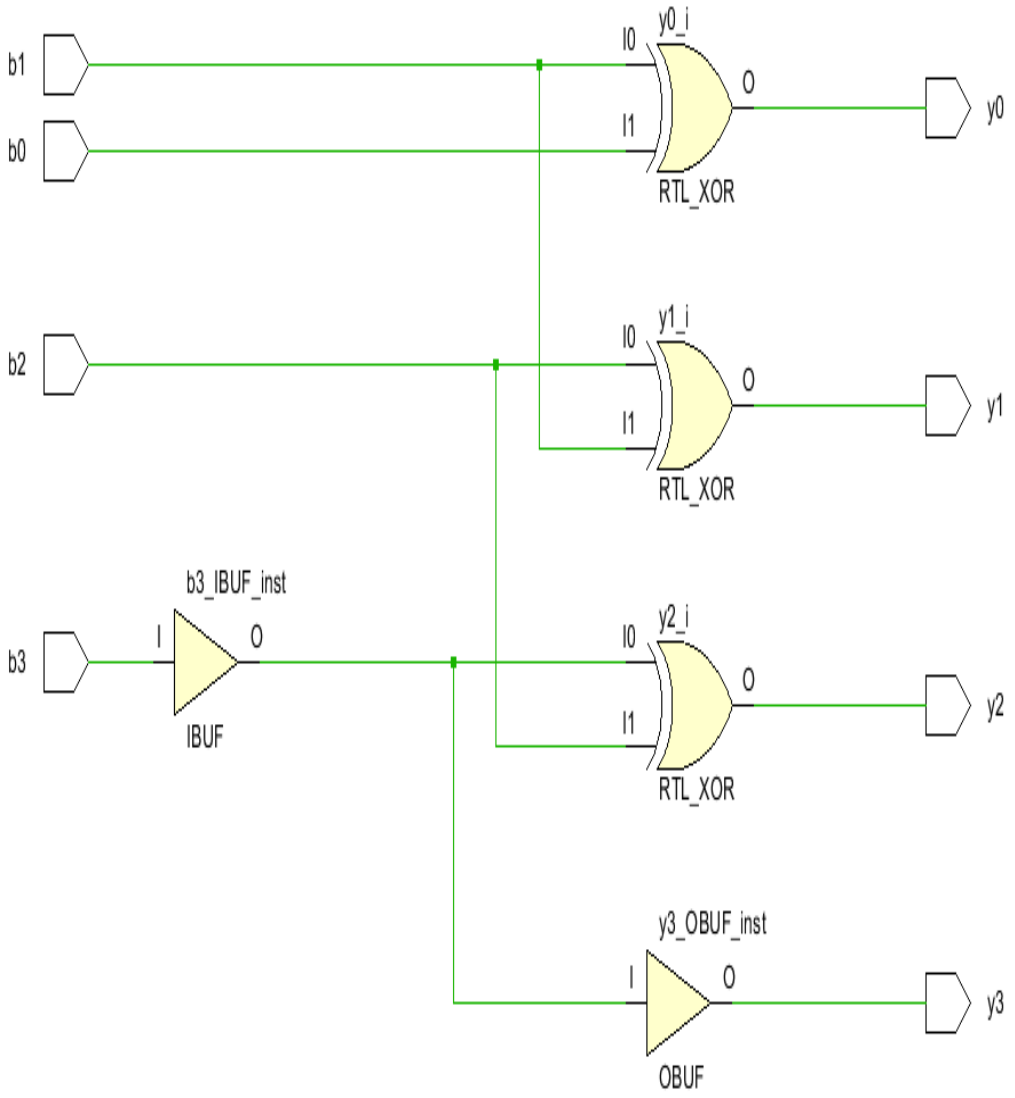
RTL View



Code

```
20  // SHIVAM SALVE VLSI 56
21  module gray_code(y3,y2,y1,y0,b3,b2,b1,b0);
22    output y3,y2,y1,y0;
23    input b3,b2,b1,b0;
24    assign y3=b3;
25    assign y2=b3^b2;
26    assign y1=b2^b1;
27    assign y0=b1^b0;
28  endmodule
```

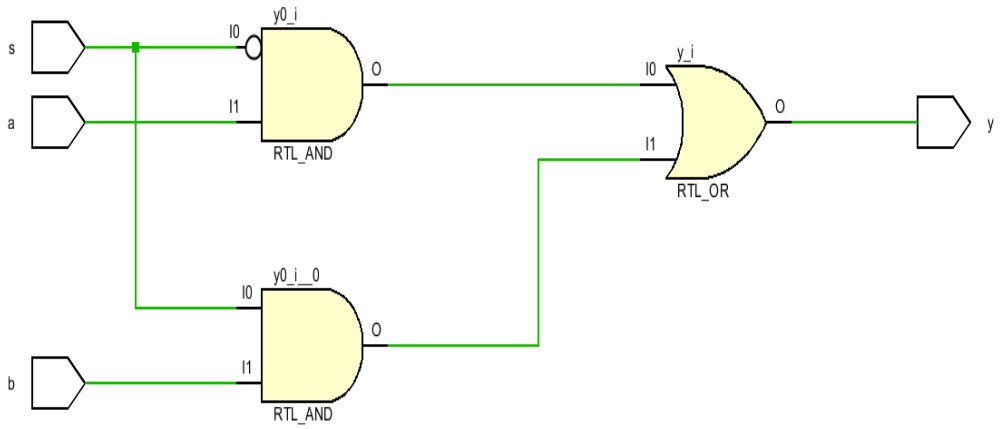
RTL View



Code

```
20  // SHIVAM SALVE VLSI 56
21  module mux_2X1(y,a,b,s);
22    input a,b,s;
23    output y;
24    assign y=((~s)&a)|(s&b);
25  endmodule
```

RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module nand_gate (a,b,y);
22    input a, b;
23    output y;
24    assign y = ~(a & b);
25  endmodule
```

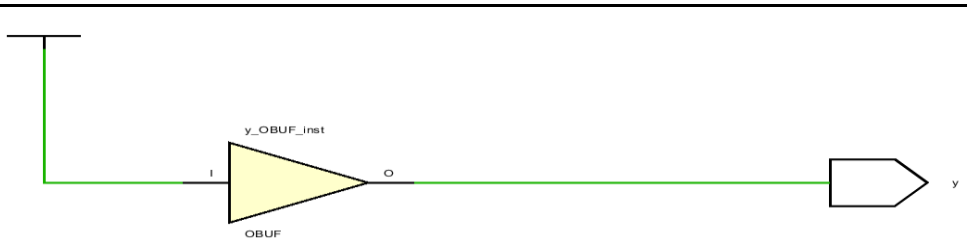
RTL View

EXTRA EXAMPLES TAKEN IN CLASS

Code

```
20  //SHIVAM SALVE VLSI 56
21  module test_1(y);
22  output wire y;
23  assign y=1'b1;
24  endmodule
25
```

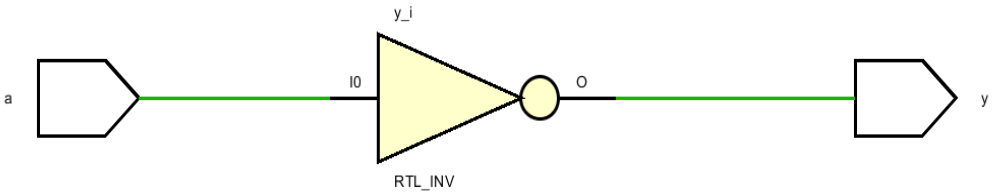
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module test_3(y,a);
22  output y;
23  input a;
24  assign y=~a;
25  endmodule
```

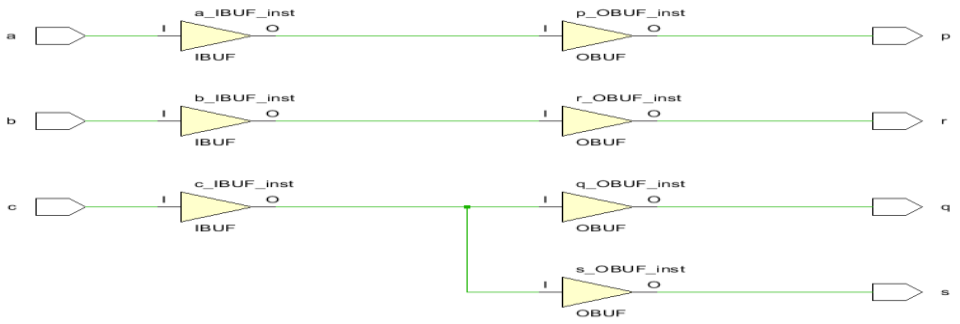
RTL View



Code

```
20  //SHIVAM SALVE VLSI 56
21  module test_4(p,q,r,s,a,b,c);
22   output p,q,r,s;
23  input a,b,c;
24  assign p=a;
25  assign q=c;
26  assign r=b;
27  assign s=c;
28  endmodule
```

RTL View

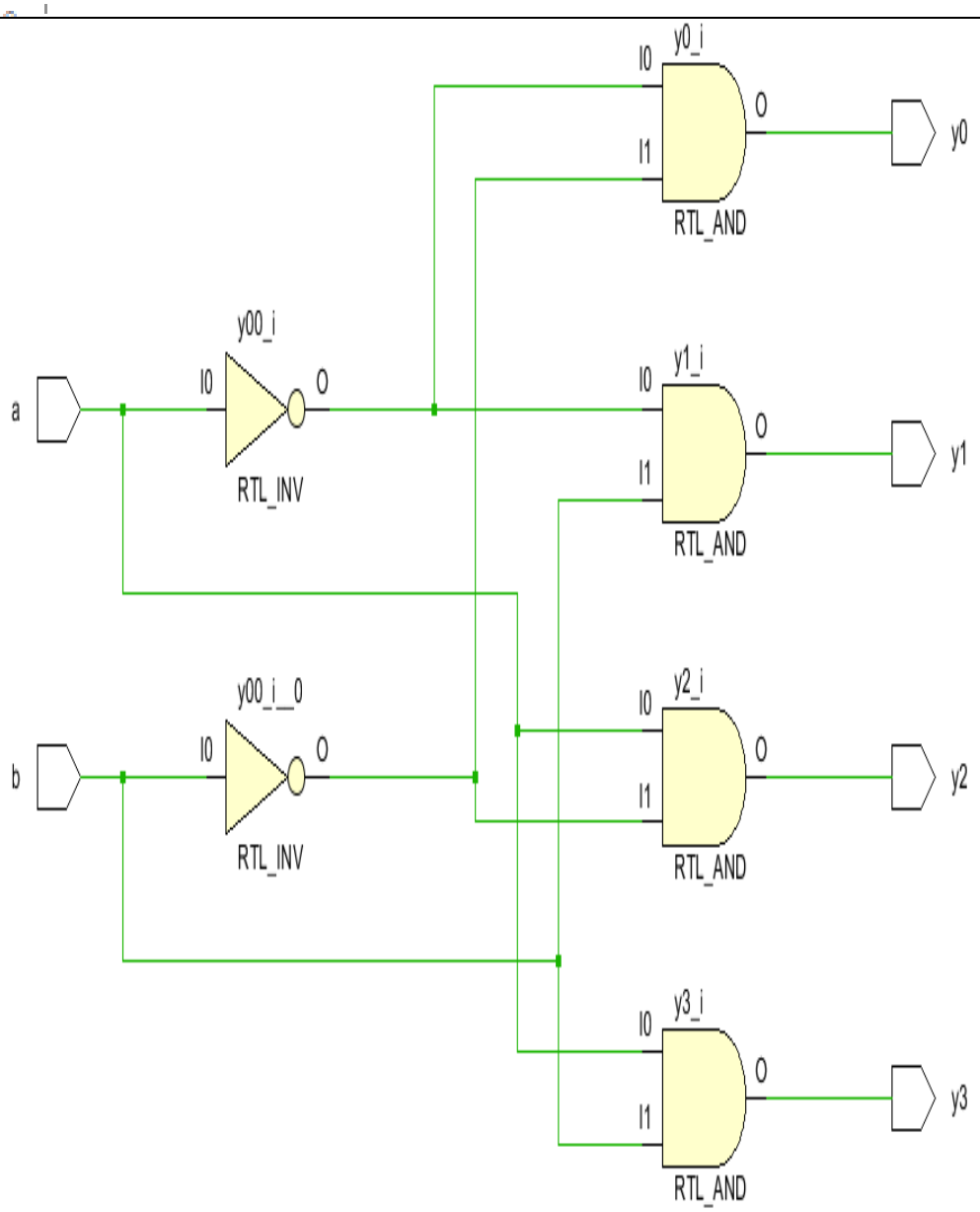


Design Any Combinational Circuit (5 Marks)

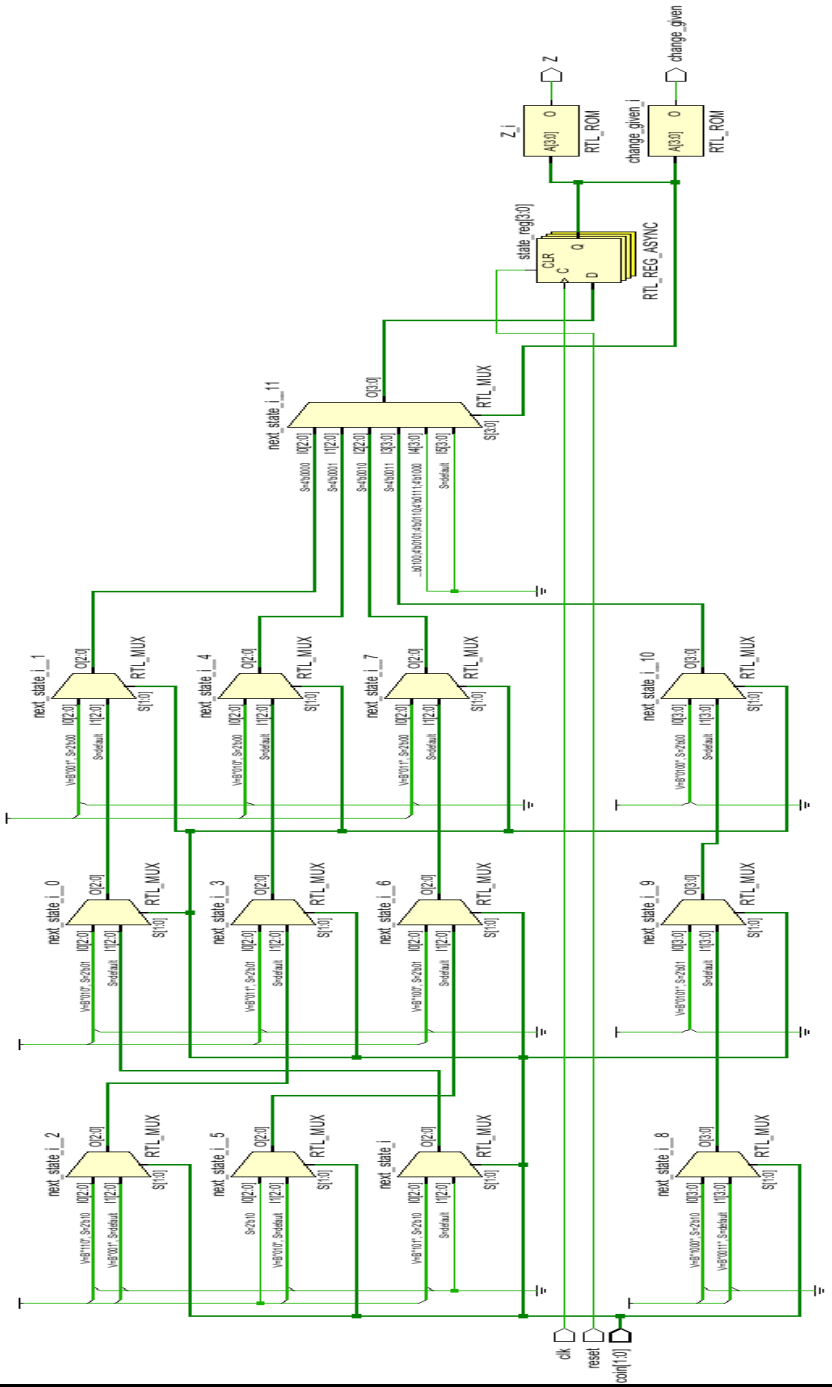
Code

```
20  //SHIVAM SALVE VLSI 56
21  //DECODER 2TO4
22  module Dec2to4(a,b,y0,y1,y2,y3);
23  output y0,y1,y2,y3;
24  input a,b;
25  assign y0 = (~a)&(~b);
26  assign y1 = (~a)&(b);
27  assign y2 = (a)&(~b);
28  assign y3 = (a)&(b);
29  endmodule
```

RTL View



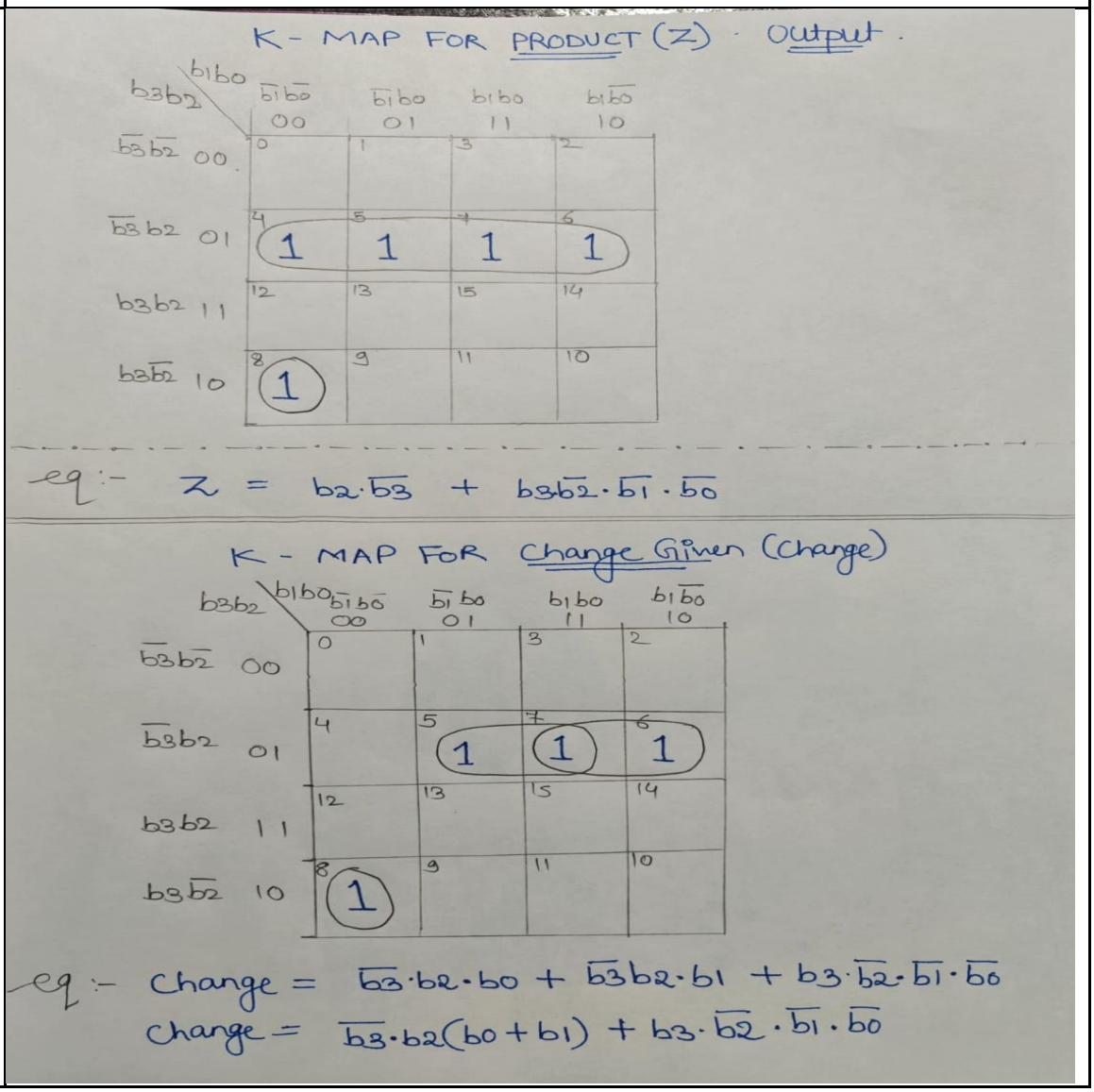
Design combinational design project (5 Marks)

Logic	Vending Machine with Change Accepts **3 types of coins/currency notes**: 10, 20, 50. Product costs **40**. FSM keeps track of balance using states: • S10 = balance 10 • S20 = balance 20 • S30 = balance 30 • S40 = balance 40 → Product given (Z=1, no change) • S50, S60, S70, S80 → Product given (Z=1, with change) After product delivery, FSM goes back to **Sin**.
Gate Level diagram	

Truth Table

State	Balance	Z (Product)	Change
Sin	0	0	0
S10	10	0	0
S20	20	0	0
S30	30	0	0
S40	40	1	0
S50	50	1	1
S60	60	1	1
S70	70	1	1
S80	80	1	1

Kmap And Equations



Code

```
//Shivam Salve VlSI 56 MSPA 1 PROJECT
module Vending_machine(
    input wire clk,
    input wire reset,
    input wire [1:0] coin,    // 2-bit coin input
    output reg Z,             // Product given
    output reg change_given   // Change return
);

parameter Sin  = 4'b0000,   // Initial state
          S10  = 4'b0001,
          S20  = 4'b0010,
          S30  = 4'b0011,
          S40  = 4'b0100,
          S50  = 4'b0101,
          S60  = 4'b0110,
          S70  = 4'b0111,
          S80  = 4'b1000;

parameter ten    = 2'b00,   // 10 Rs coin
          twenty = 2'b01,   // 20 Rs coin
          fifty  = 2'b10;   // 50 Rs coin

reg [3:0] state, next state;

// ------------------ State Update ------------------
always @(posedge clk or posedge reset) begin
    if (reset)
        state <= Sin;
    else
        state <= next_state;
end

// ------------------ Next State Logic ------------------
always @(*) begin
    case (state)
        Sin: begin
            if (coin == ten)       next_state = S10;
            else if (coin == twenty) next_state = S20;
            else if (coin == fifty)  next_state = S50;
            else next_state = Sin;
        end

        S10: begin
            if (coin == ten)       next_state = S20;
            else if (coin == twenty) next_state = S30;
            else if (coin == fifty)  next_state = S60;
            else next_state = S10;
        end

        S20: begin
            if (coin == ten)       next_state = S30;
            else if (coin == twenty) next_state = S40;
            else if (coin == fifty)  next_state = S70;
            else next_state = S20;
        end

        S30: begin
            if (coin == ten)       next_state = S40;
            else if (coin == twenty) next_state = S50;
            else if (coin == fifty)  next_state = S80;
            else next_state = S30;
        end

        S40, S50, S60, S70, S80: next state = Sin;
```

```
86
87          default: next_state = Sin;
88      endcase
89  end
90
91  // ----------------- Output Logic -----------------
92  always @(*) begin
93      case (state)
94          Sin, S10, S20, S30: begin
95              Z = 0; change_given = 0;
96          end
97
98          S40: begin
99              Z = 1; change_given = 0;  // Product delivered, no change
00          end
01          S50, S60, S70, S80: begin
02              Z = 1; change_given = 1;  // Product delivered, change returned
03          end
04
05          default: begin
06              Z = 0; change_given = 0;
07          end
08      endcase
09  end
10
11  endmodule
```

Evaluation by faculty **(Dr. P.P. Zode)**

5 Marks	5 Marks	5 Marks	15 Marks