ÖMER FARUK SERT

040170244

I3	I2	I1	I0	O15	O14	O13	O12	O11	O10	O9	O8	O7	O6	O5	O4	O3	O2	O1	O0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DECODER

MSI_library.v

```
`timescale 1ns / 1ps


module Decoder(IN , O, EN);
        input [3:0]IN;
        input EN;
        output reg [15:0]O;
        always @(*)
            begin
                if(EN==1'b1)
                begin
                    case (IN)
                        4'b0000:O=16'b0000000000000001;
                        4'b0001:O=16'b0000000000000010;
                        4'b0010:O=16'b0000000000000100;
                        4'b0011:O=16'b0000000000001000;
                        4'b0100:O=16'b0000000000010000;
                        4'b0101:O=16'b0000000000100000;
```

```
                        4'b0110:O=16'b0000000001000000;
                        4'b0111:O=16'b0000000010000000;
                        4'b1000:O=16'b0000000100000000;
                        4'b1001:O=16'b0000001000000000;
                        4'b1010:O=16'b0000010000000000;
                        4'b1011:O=16'b0000100000000000;
                        4'b1100:O=16'b0001000000000000;
                        4'b1101:O=16'b0010000000000000;
                        4'b1110:O=16'b0100000000000000;
                        4'b1111:O=16'b1000000000000000;
                    endcase
                end
                else
                O=16'h00;
            end

endmodule
```

Top_module.v

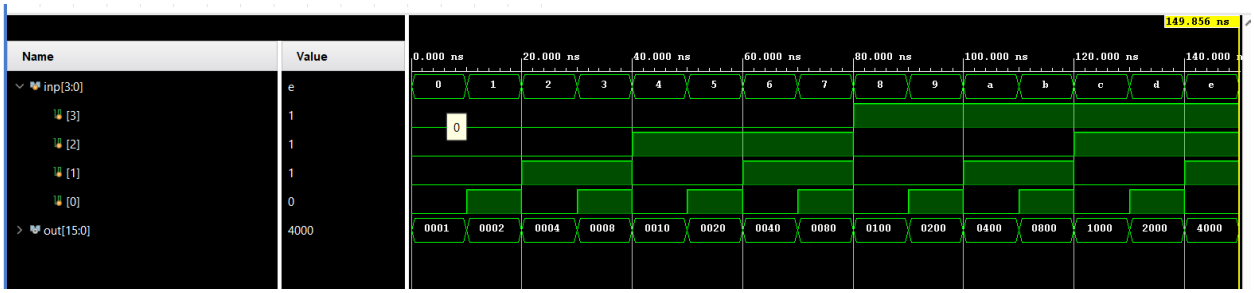
```
`timescale 1ns / 1ps



module Top_module(SW,BTN,LED,CAT,AN,DP);
        input [7:0]SW;
        input [3:0]BTN;
        output [7:0]LED;
        output [6:0]CAT;
        output [3:0]AN;
        output DP;
        wire [15:0]OX;
        wire HIGH;
        assign AN = 4'b1110;
        assign DP=OX[15];
        assign CAT=OX[14:8];
        assign LED=OX[7:0];
        assign HIGH=1'b1;
        Decoder decoder1(.IN(SW[3:0]),.O(OX),.EN(HIGH));
endmodule
```

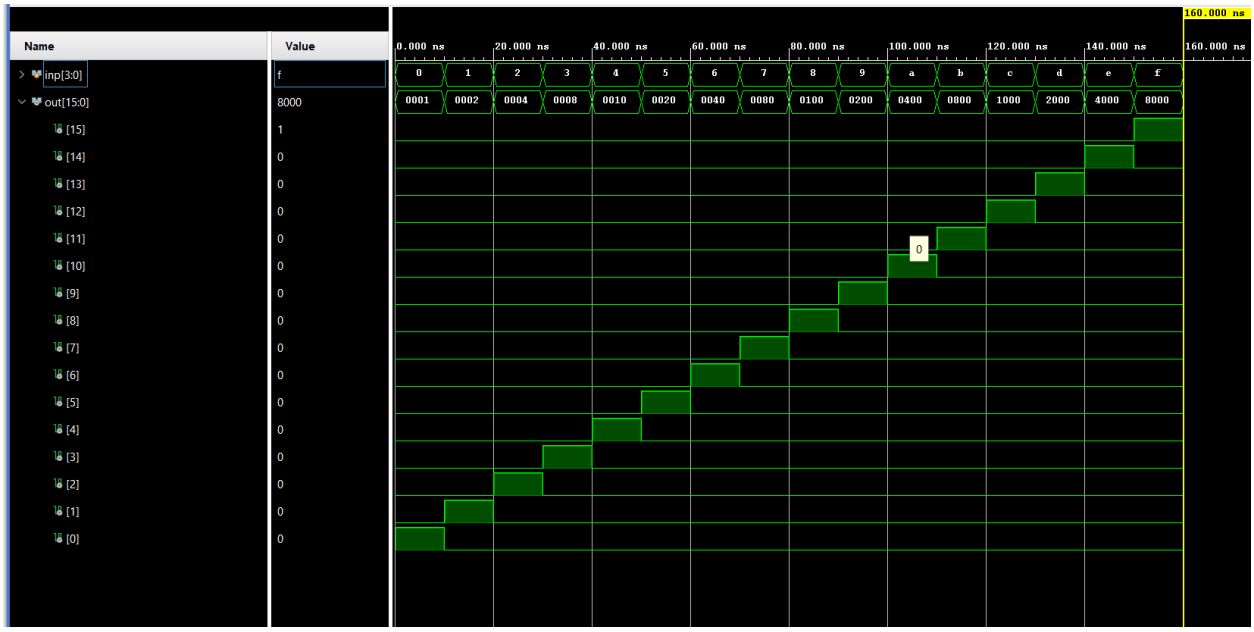
Top_module_tb.v

```
`timescale 1ns / 1ps


module Top_module_tb;
    reg [3:0] inp;
    wire [15:0] out;

    Top_module uut(.SW(inp), .DP(out[15]),.CAT(out[14:8]),.LED(out[7:0]));

    initial
    begin
    // inp = 0000
    inp=4'b0000;
    #10
    // inp = 0001
    inp=4'b0001;
    #10
    // inp = 0010
    inp=4'b0010;
    #10
    // inp = 0011
    inp=4'b0011;
    #10
    // inp = 0100
    inp=4'b0100;
    #10
    // inp = 0101
    inp=4'b0101;
    #10
    // inp = 0110
    inp=4'b0110;
    #10
    // inp = 0111
    inp=4'b0111;
    #10
    // inp = 1000
    inp = 4'b1000;
    #10
    // inp = 1001
    inp = 4'b1001;
```

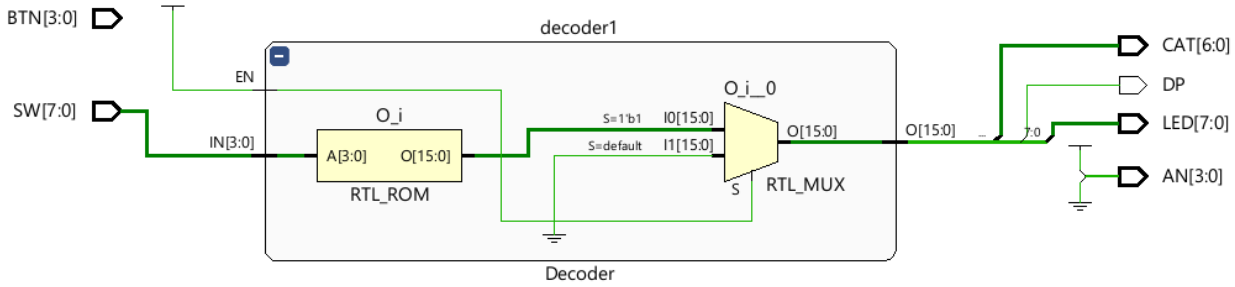
```
    #10
    // inp = 1010
    inp = 4'b1010;
    #10
    // inp = 1011
    inp = 4'b1011;
    #10
    // inp = 1100
    inp = 4'b1100;
    #10
    // inp = 1101
    inp =4'b1101;
    #10
    // inp = 1110
    inp= 4'b1110;
    #10
    // inp = 1111
    inp = 4'b1111;
    $finish;

    end
endmodule
```

Behavioral Simulation Wave Screenshots

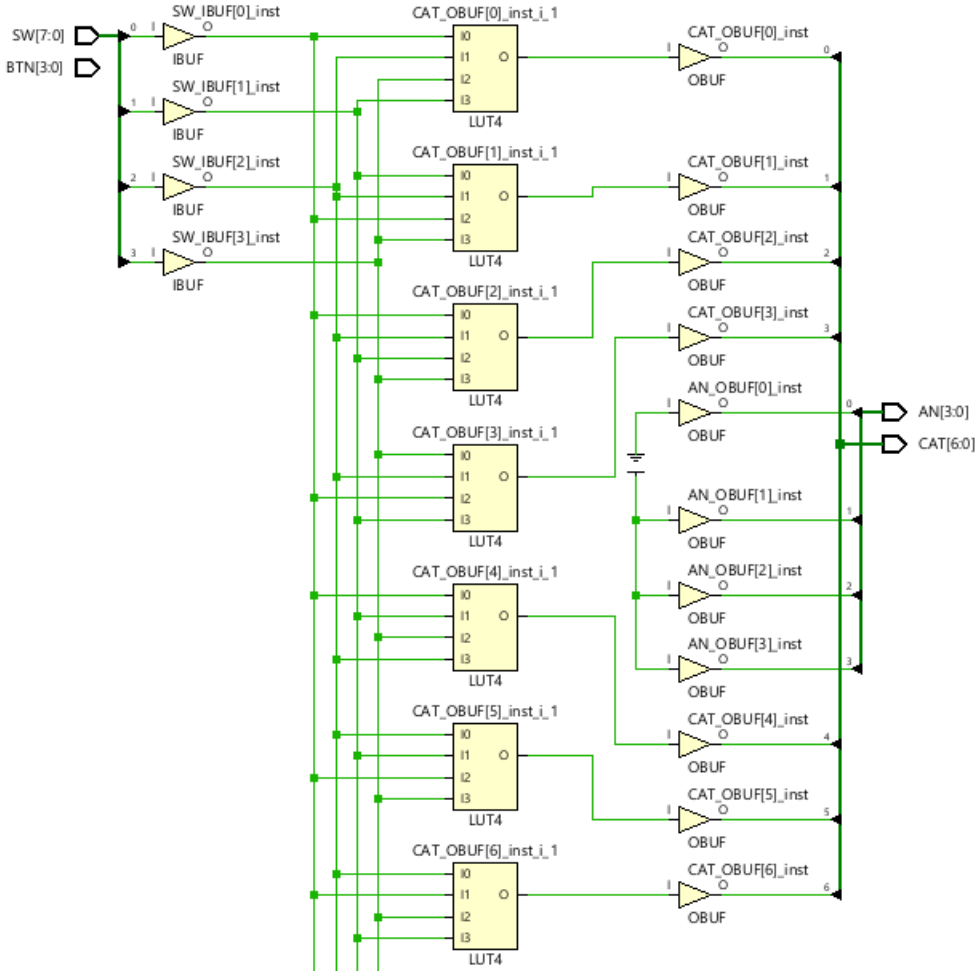


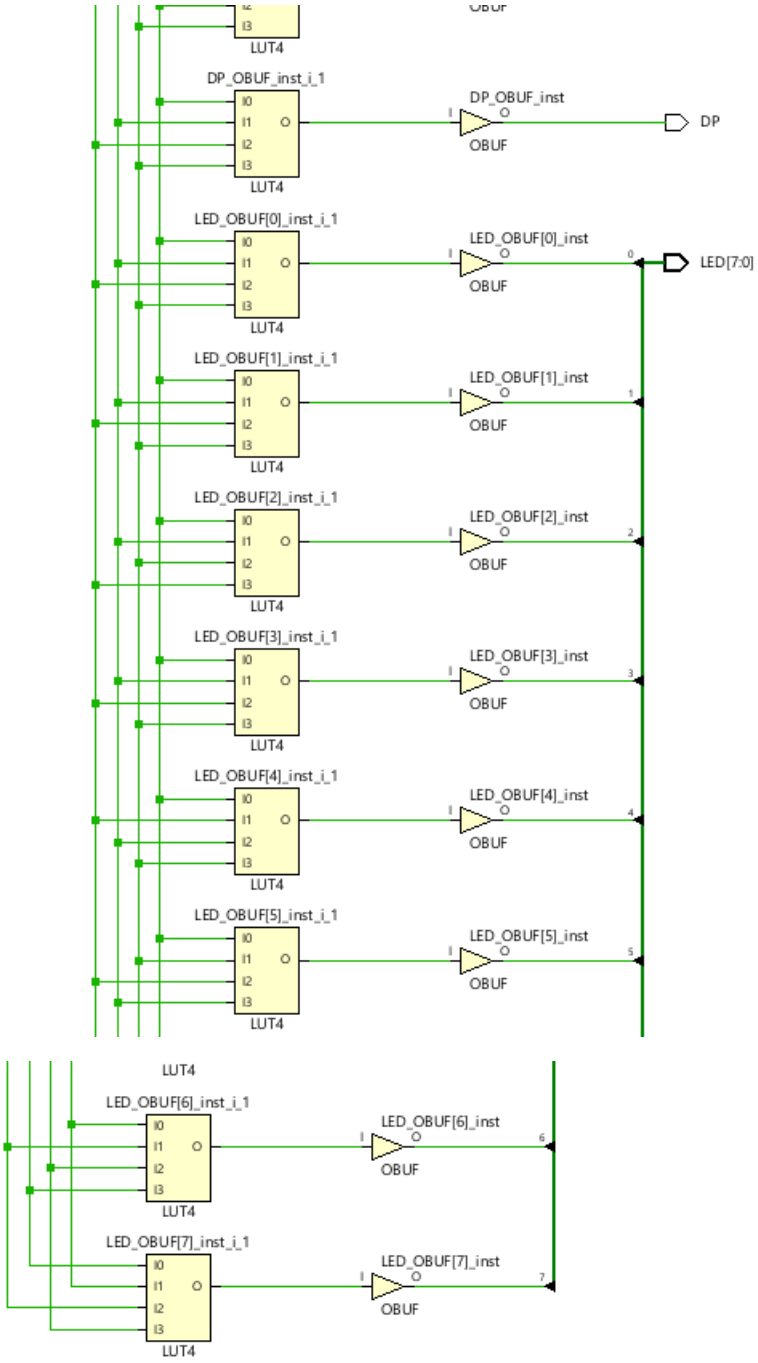


RTL Schematic

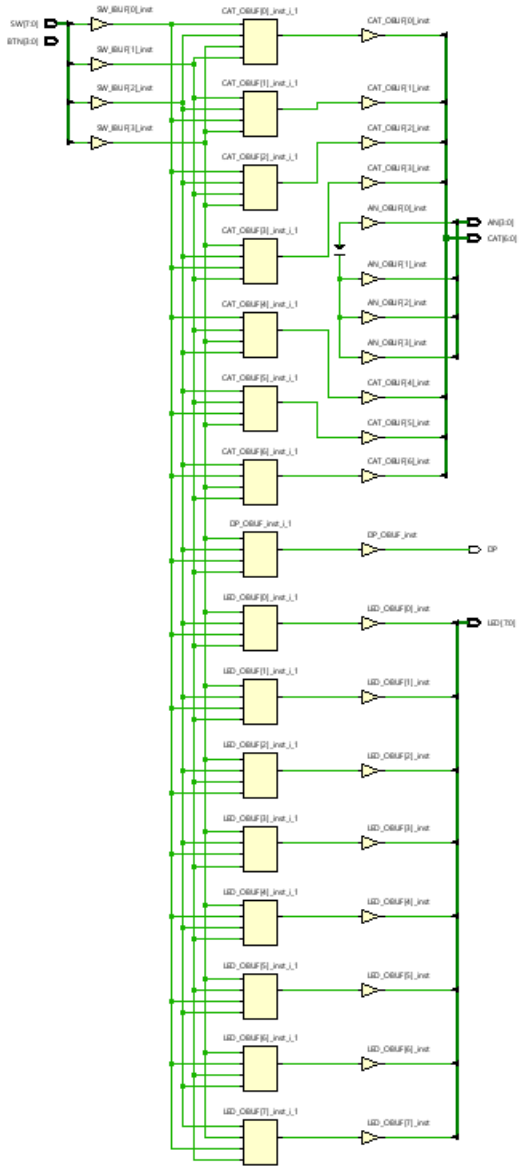


Technology Schematic




LUT4
DP_OBUF_inst_i_1
I0
I1
I2
I3
LUT4
DP_OBUF_inst
I
O
OBUF
DP
LED_OBUF[0]_inst_i_1
LED_OBUF[0]_inst
0
LED[7:0]
LED_OBUF[1]_inst_i_1
LED_OBUF[1]_inst
1
LED_OBUF[2]_inst_i_1
LED_OBUF[2]_inst
2
LED_OBUF[3]_inst_i_1
LED_OBUF[3]_inst
3
LED_OBUF[4]_inst_i_1
LED_OBUF[4]_inst
4
LED_OBUF[5]_inst_i_1
LED_OBUF[5]_inst
5

LUT4
LED_OBUF[6]_inst_i_1
I0
I1
I2
I3
LUT4
LED_OBUF[6]_inst
I
O
6
OBUF
LED_OBUF[7]_inst_i_1
LED_OBUF[7]_inst
7
OBUF


SW[7:0]
BTN[3:0]
SW_IBUF[0]_inst
SW_IBUF[1]_inst
SW_IBUF[2]_inst
SW_IBUF[3]_inst
CAT_OBUF[0]_inst_i_1
CAT_OBUF[1]_inst_i_1
CAT_OBUF[2]_inst_i_1
CAT_OBUF[3]_inst_i_1
CAT_OBUF[4]_inst_i_1
CAT_OBUF[5]_inst_i_1
CAT_OBUF[6]_inst_i_1
DP_OBUF_inst_i_1
LED_OBUF[0]_inst_i_1
LED_OBUF[1]_inst_i_1
LED_OBUF[2]_inst_i_1
LED_OBUF[3]_inst_i_1
LED_OBUF[4]_inst_i_1
LED_OBUF[5]_inst_i_1
LED_OBUF[6]_inst_i_1
LED_OBUF[7]_inst_i_1
CAT_OBUF[0]_inst
CAT_OBUF[1]_inst
CAT_OBUF[2]_inst
CAT_OBUF[3]_inst
AN_OBUF[0]_inst
AN_OBUF[1]_inst
AN_OBUF[2]_inst
AN_OBUF[3]_inst
CAT_OBUF[4]_inst
CAT_OBUF[5]_inst
CAT_OBUF[6]_inst
DP_OBUF_inst
LED_OBUF[0]_inst
LED_OBUF[1]_inst
LED_OBUF[2]_inst
LED_OBUF[3]_inst
LED_OBUF[4]_inst
LED_OBUF[5]_inst
LED_OBUF[6]_inst
LED_OBUF[7]_inst
AN[3:0]
CAT[6:0]
DP
LED[7:0]

How many LUTs are there in the Technology schematic? What are the logic statements of the LUTs and how does these logic statements implement the decoding operation? ? Just explain.

16 LUTS are exist in technology schematic. Logic statements are one rom and two to one 16-bit multiplexer. By 4-bit input choosing the 16-bit output of ROM.

Greatest Delay Without Timing Constraint In Implementation

10.502

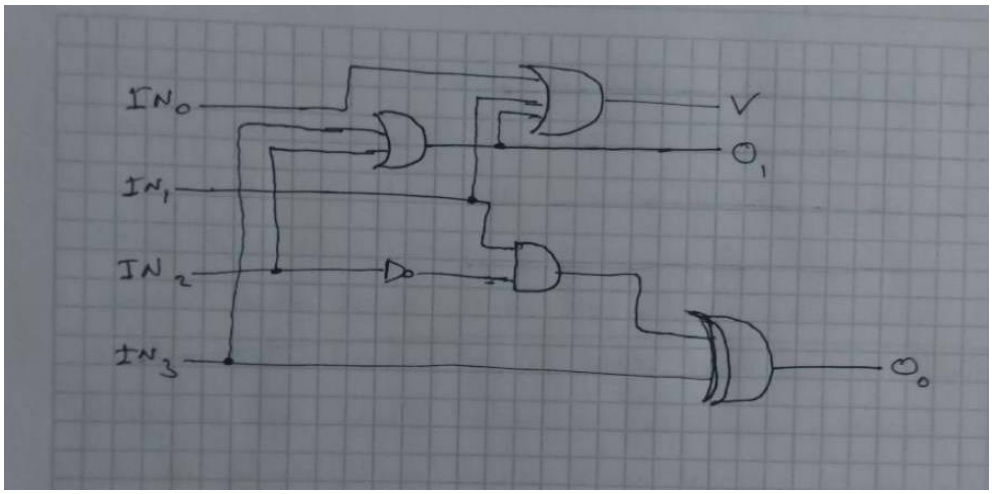
Greatest Delay with Timing Constraint In Implementation

9.299

Priority Encoder

IN0	IN1	IN2	IN3	O1	O0	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

O2 = IN3 + IN2 , O0 = IN1xIN2’ + IN3



MSI_library.v

```
`timescale 1ns / 1ps

module ENCODER (IN,O,V);
        input [3:0]IN;
        output [1:0]O;
        output V;

        assign O[0]=((~IN[2])&IN[1])|IN[3];
        assign O[1]=IN[2]|IN[3];
        assign V=IN[0]|IN[1]|IN[2]|IN[3];
endmodule
```

```
module ENCODER1 (IN,O,V);
        input [3:0]IN;
        output reg [1:0]O;
        output reg V;

        always @(*)
        begin
            case(IN)
                    4'b0000: begin O=2'b00;V=1'b0; end
                    4'b0001: begin O=2'b00;V=1'b1; end
                    4'b0010: begin O=2'b01;V=1'b1; end
                    4'b0100: begin O=2'b10;V=1'b1; end
                    4'b1000: begin O=2'b11;V=1'b1; end
```

```
                    default:
                    begin O=2'bxx;V=1'b0; end
       endcase
        end
endmodule
```

Top_module.v

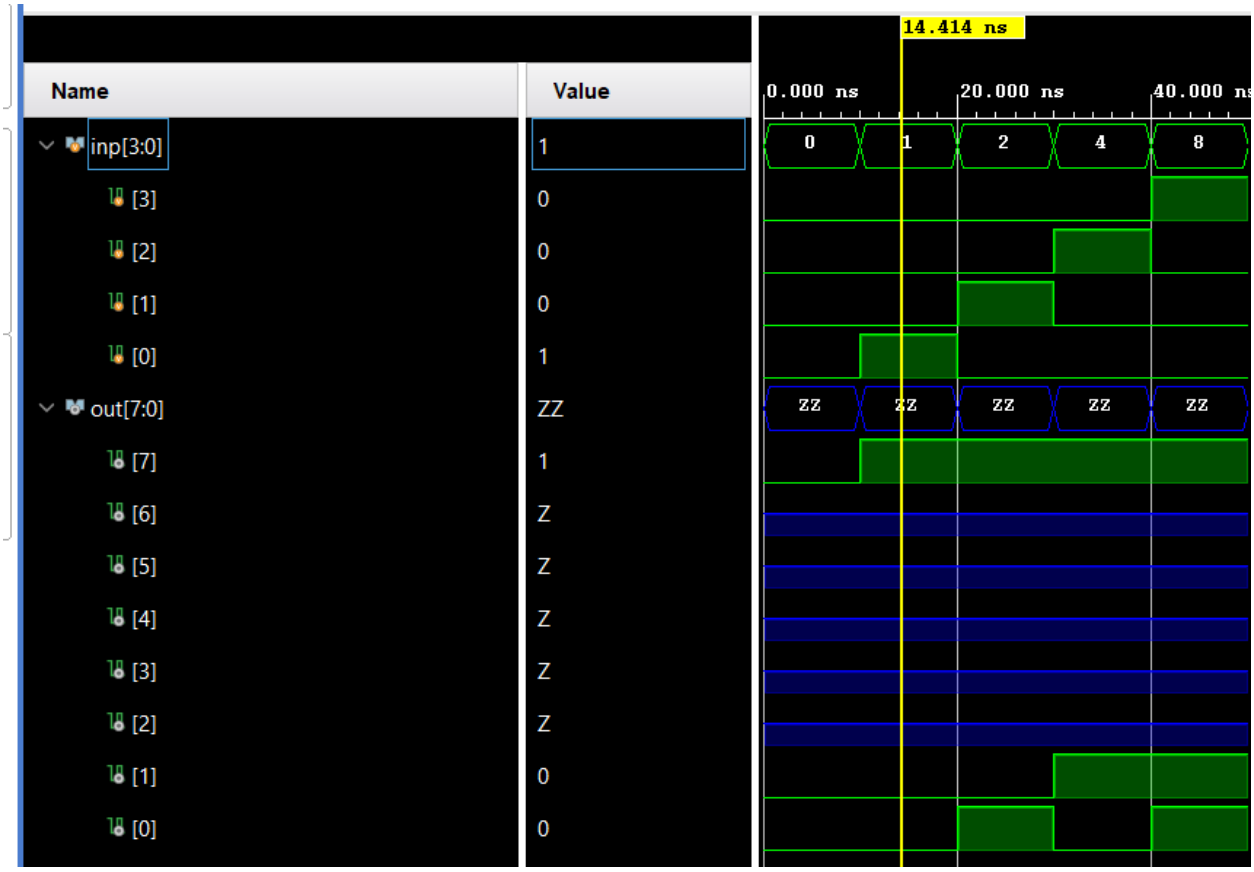
```
`timescale 10ns / 1ps



module Top_module(SW,BTN,LED,CAT,AN,DP);
        input [7:0]SW;
        input [3:0]BTN;
        output [7:0]LED;
        output [6:0]CAT;
        output [3:0]AN;
        output DP;
        wire [15:0]OX;
        //wire HIGH;
        //assign AN = 4'b1110;//for decoder
        //assign DP=OX[15];//for decoder
        //assign CAT=OX[14:8];//for decoder
        //assign LED=OX[7:0];//for decoder
        //assign HIGH=1'b1;//for decoder
        //Decoder decoder1(.IN(SW[3:0]),.O(OX),.EN(HIGH));
        ENCODER encoder1(.IN(SW[3:0]),.O(LED[1:0]),.V(LED[7]));
endmodule
```

Top_module_tb.v

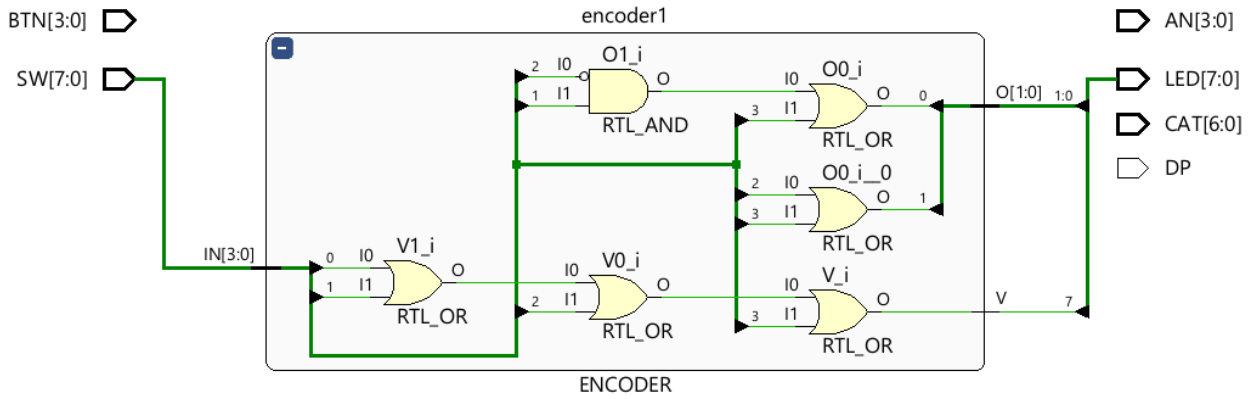
```
`timescale 1ns / 1ps


module Top_module_tb;
    reg [3:0] inp;
    wire [7:0] out;
```

```
    //Top_module uut(.SW(inp), .DP(out[15]),.CAT(out[14:8]),.LED(out[7:0]));
//for decoder

    Top_module uut(.SW(inp), .LED(out[7:0])); //for encoder

    initial
    begin
    // inp = 0000
    inp=4'b0000;
    #10
    // inp = 0001
    inp=4'b0001;
    #10
    // inp = 0010
    inp=4'b0010;
    #10
    // inp = 0100
    inp=4'b0100;
    #10
    inp = 4'b1000;
    #10
    $finish;

    end
endmodule
```

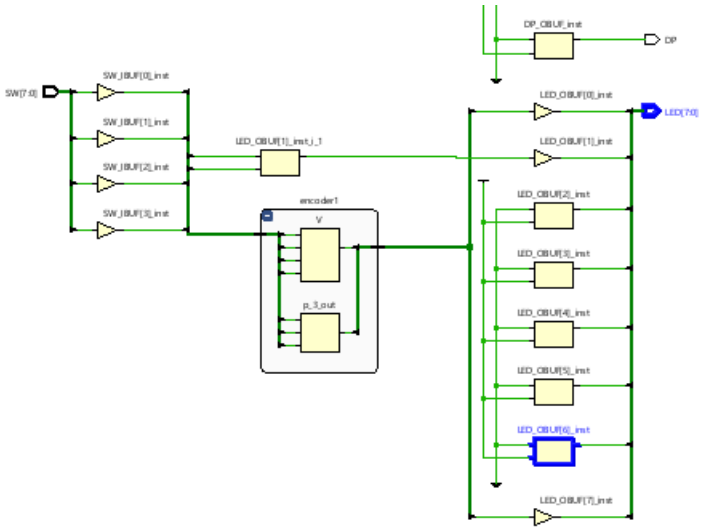
Behavioral Simulation Wave Screenshots



RTL Schematic



Technology Schematic



In technology schematic there is 3 LUT but in physical there is 2 LUT.

In RTL schematic there is 5 OR and 1 AND gate and 1 not input in the AND gate.

Timing Report for Structural Design

Synthesis

6.76 ns

Implementation

9.68 ns

Timing Report for Structural Design

Synthesis

6.77 ns

Implementation

9.549 ns

MUX

Top_module.v

```
`timescale 10ns / 1ps



module Top_module(SW,BTN,LED,CAT,AN,DP);
        input [7:0]SW;
        input [3:0]BTN;
        output [7:0]LED;
        output [6:0]CAT;
        output [3:0]AN;
        output DP;
        wire [15:0]OX;
        //wire HIGH;
        //assign AN = 4'b1110;//for decoder
        //assign DP=OX[15];//for decoder
        //assign CAT=OX[14:8];//for decoder
        //assign LED=OX[7:0];//for decoder
        //assign HIGH=1'b1;//for decoder
        //Decoder decoder1(.IN(SW[3:0]),.O(OX),.EN(HIGH));
        //ENCODER1 encoder1(.IN(SW[3:0]),.O(LED[1:0]),.V(LED[7]));
        MUX mux1(.IN(SW[3:0]),.S(BTN[1:0]),.O(LED[0]));
endmodule
```

MSI_library.v

```
`timescale 1ns / 1ps

module MUX (IN,S,O);
        input [3:0]IN;
        input [1:0]S;
        output O;
        assign O= (IN[0] & ~S[1] & ~S[0]) |
                (IN[1] & ~S[1] & S[0]) |
                (IN[2] & S[1] & ~S[0]) |
                (IN[3] & S[1] & S[0]);
endmodule
```

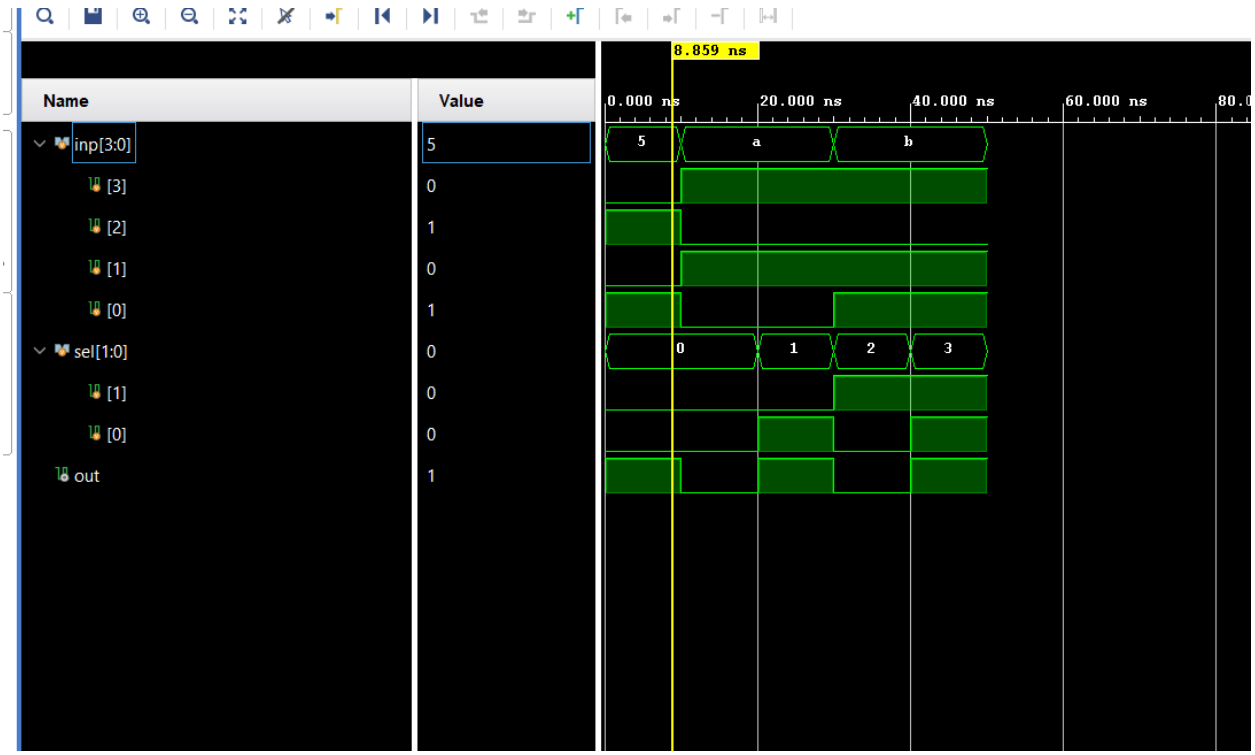
```
module MUX1 (IN,S,O);
        input [3:0]IN;
        input [1:0]S;
        output reg O;
        always @(*)
        begin
            case(S)
            2'b00:O=IN[0];
            2'b01:O=IN[1];
            2'b10:O=IN[2];
            2'b11:O=IN[3];
        endcase
        end
endmodule
```

Top_module_tb.v

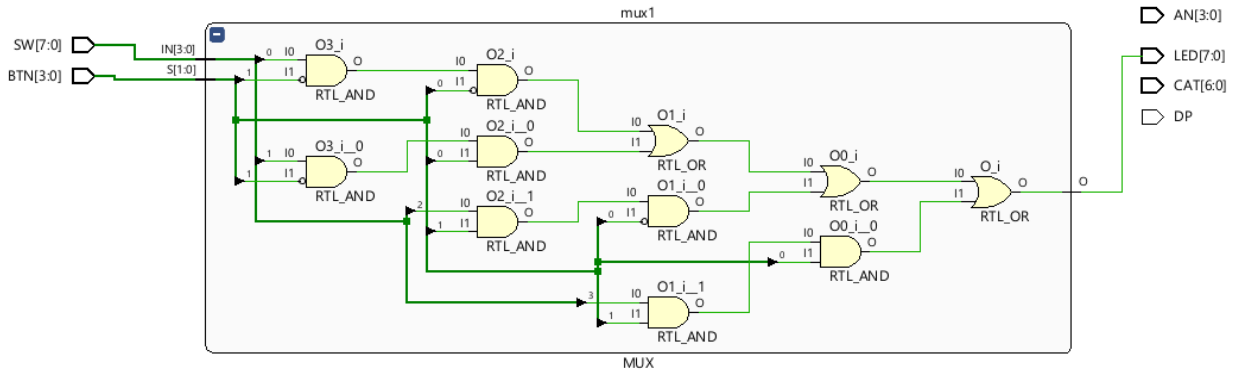
```
`timescale 1ns / 1ps


module Top_module_tb;
    reg [3:0] inp;
    reg [1:0] sel;
    //wire [7:0] out; // for encoder and decoder
    wire out;

    //Top_module uut(.SW(inp), .DP(out[15]),.CAT(out[14:8]),.LED(out[7:0]));
//for decoder

    //Top_module uut(.SW(inp), .LED(out[7:0])); //for encoder
    Top_module uut(.SW(inp),.BTN(sel) , .LED(out));//for mux
    initial //for mux
    begin
        // inp = 0101 sel = 00
        inp = 4'b0101;
        sel = 2'b00;
        #10
        // inp = 1010 sel =00
        inp = 4'b1010;
        sel = 2'b00;
        #10
        // inp = 1010 sel =01
        inp = 4'b1010;
```

```
        sel = 2'b01;
        #10
        // inp =1011 sel = 10
        inp = 4'b1011;
        sel = 2'b10;
        #10
        // inp = 1011 sel =11
        inp = 4'b1011;
        sel = 2'b11;
        #10
        $finish;
    end
endmodule
```

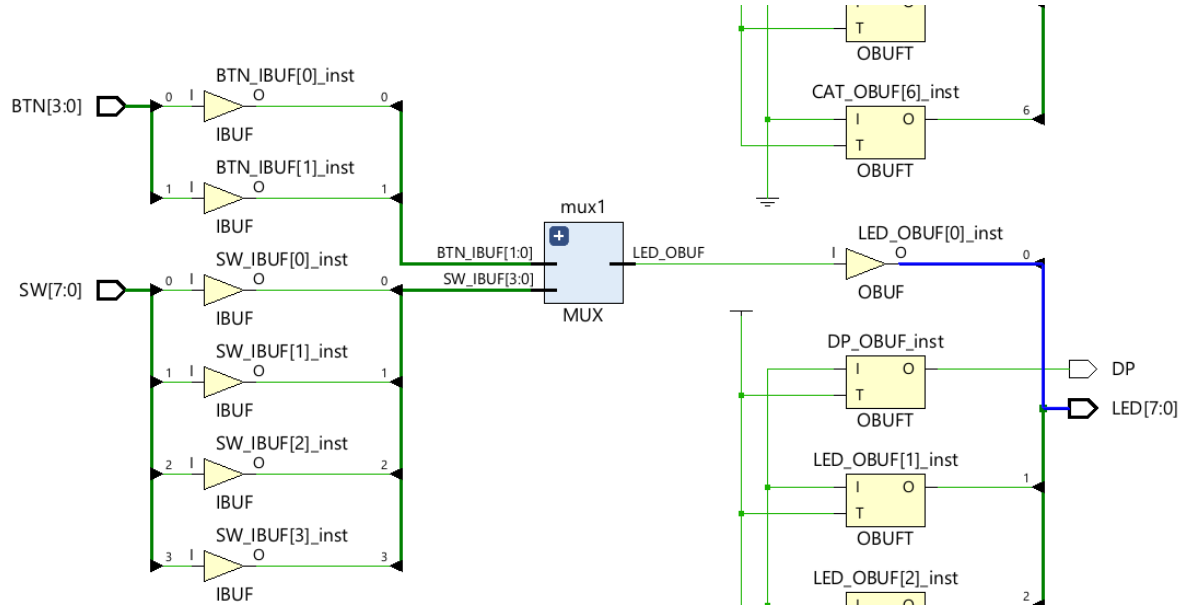
Wave Screenshots of Behavioral Simulation



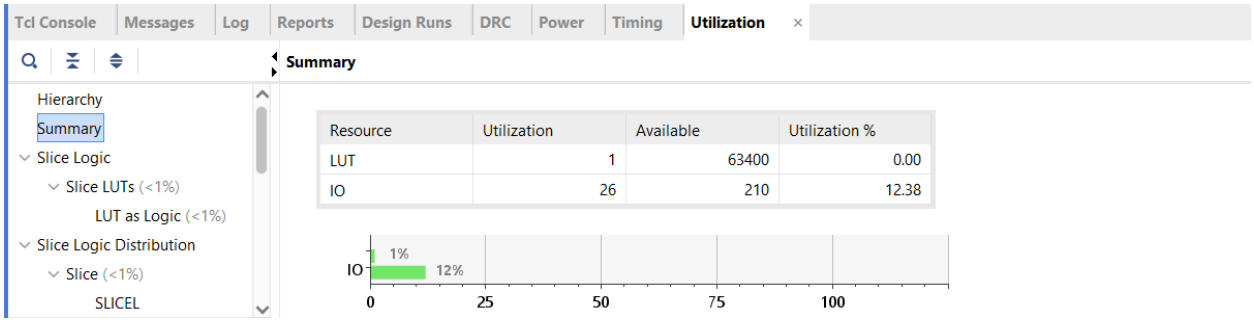
RTL Schematic



Technology Schematic



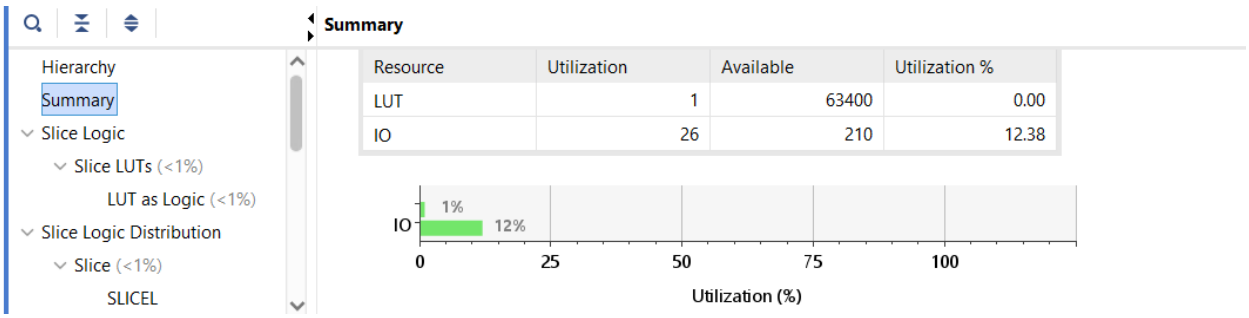
Timing and Utilization Report



Maximum Combinational Delay

9.24ns

Timing and Utilization Report of always case written design



Maximum Combinational Delay

9.241ns

DEMUX

Top_module.v

```
`timescale 10ns / 1ps



module Top_module(SW,BTN,LED,CAT,AN,DP);
        input [7:0]SW;
        input [3:0]BTN;
        output [7:0]LED;
        output [6:0]CAT;
        output [3:0]AN;
        output DP;
        wire [15:0]OX;
        wire HIGH;
```

```
        //assign AN = 4'b1110;//for decoder
        //assign DP=OX[15];//for decoder
        //assign CAT=OX[14:8];//for decoder
        //assign LED=OX[7:0];//for decoder
        //assign HIGH=1'b1;//for decoder
        //Decoder decoder1(.IN(SW[3:0]),.O(OX),.EN(HIGH));
        //ENCODER1 encoder1(.IN(SW[3:0]),.O(LED[1:0]),.V(LED[7]));
        //MUX mux1(.IN(SW[3:0]),.S(BTN[1:0]),.O(LED[0]));
        //MUX1 mux2(.IN(SW[3:0]),.S(BTN[1:0]),.O(LED[0]));
        DEMUX demux1(.IN(SW[0]),.S(BTN[1:0]),.O(LED[3:0]));
endmodule
```

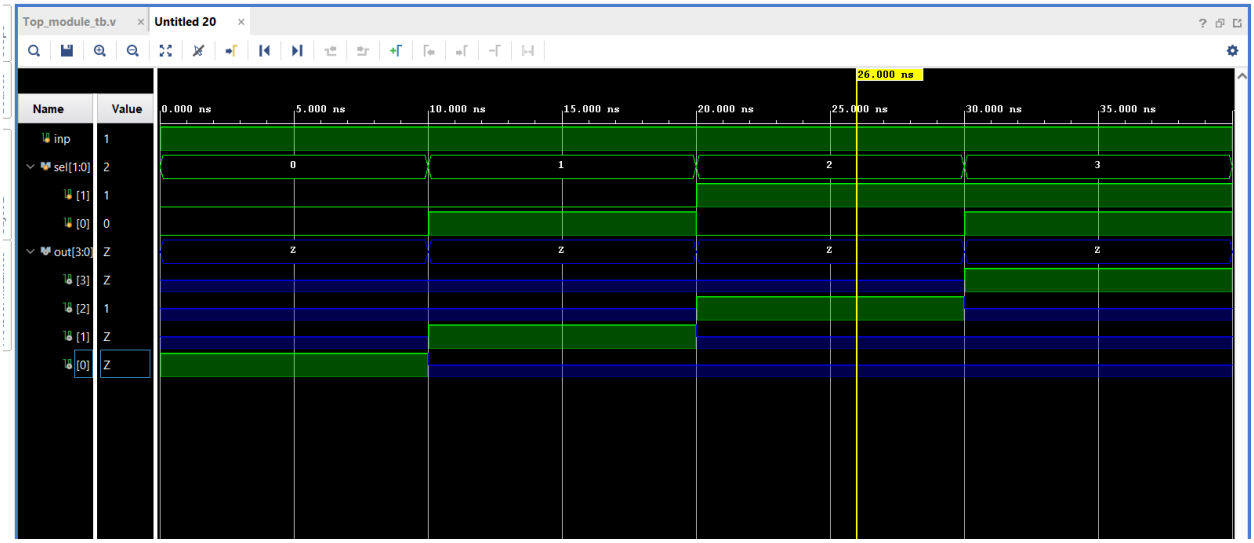
MSI_library.v

```
`timescale 1ns / 1ps


module DEMUX (IN,S,O);
        input IN;
        input [1:0]S;
        output [3:0]O;
        wire [3:0]OX;
        wire [3:0]N;
        //least significant 00
        NOT not2(.O(N[0]),.I(S[0]));
        NOT not3(.O(N[1]),.I(S[1]));
        AND and1(.O(OX[0]),.I1(N[0]),.I2(N[1]));
        TRI tri5(.O(O[0]),.E(OX[0]),.I(IN));
        //01
        NOT not4(.O(N[2]),.I(S[1]));
        AND and2(.O(OX[1]),.I1(S[0]),.I2(N[2]));
        TRI tri2(.O(O[1]),.E(OX[1]),.I(IN));
        //10
        NOT not5(.O(N[3]),.I(S[0]));
        AND and3(.O(OX[2]),.I1(N[3]),.I2(S[1]));
        TRI tri3(.O(O[2]),.E(OX[2]),.I(IN));
        //11
        AND and4(.O(OX[3]),.I1(S[0]),.I2(S[1]));
        TRI tri4(.O(O[3]),.E(OX[3]),.I(IN));

endmodule
```

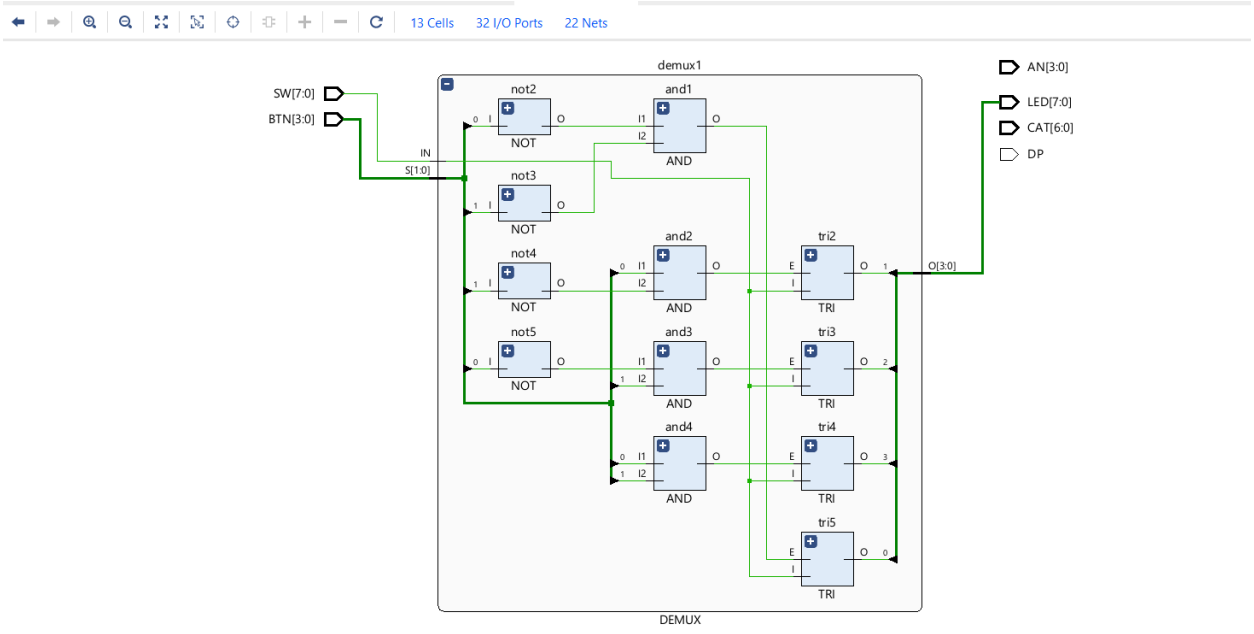
Top_module_tb.v

```
`timescale 1ns / 1ps


module Top_module_tb;
    reg inp;
    reg [1:0]sel;
    //reg [1:0] sel;
    //wire [7:0] out; // for encoder and decoder
    wire [3:0]out;


    Top_module uut(.SW(inp),.BTN(sel),.LED(out));
    initial
    begin
        //inp = 1 sel = 00
        inp=1'b1;
        sel=2'b00;
        #10
        sel=2'b01;
        #10
        sel=2'b10;
        #10
        sel=2'b11;
        #10
        $finish;
    end
endmodule
```

Behavioral Simulation Wave Screenshots



RTL Schematic



Technology Schematic