Maven Lab 3

1)Write behavioral description and test bench for 4:1 MUX

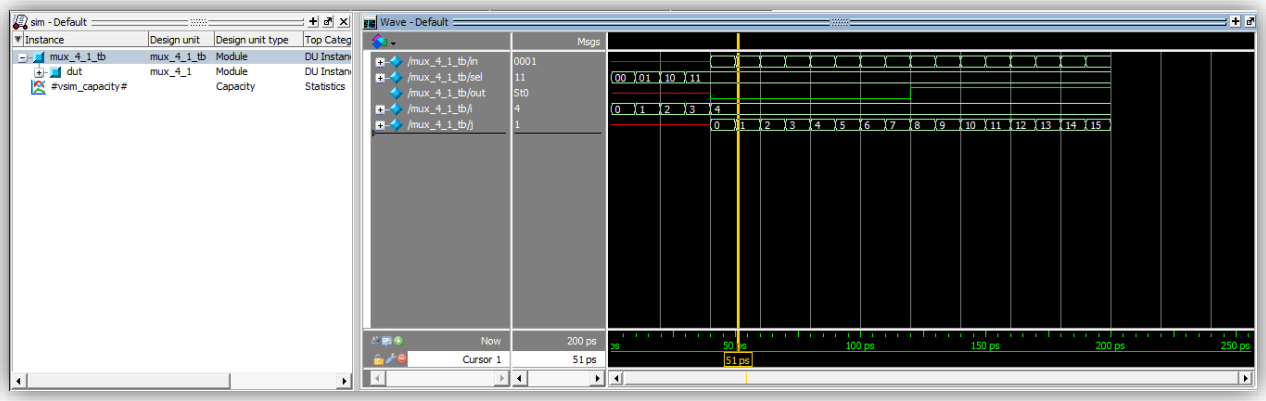
CODE:

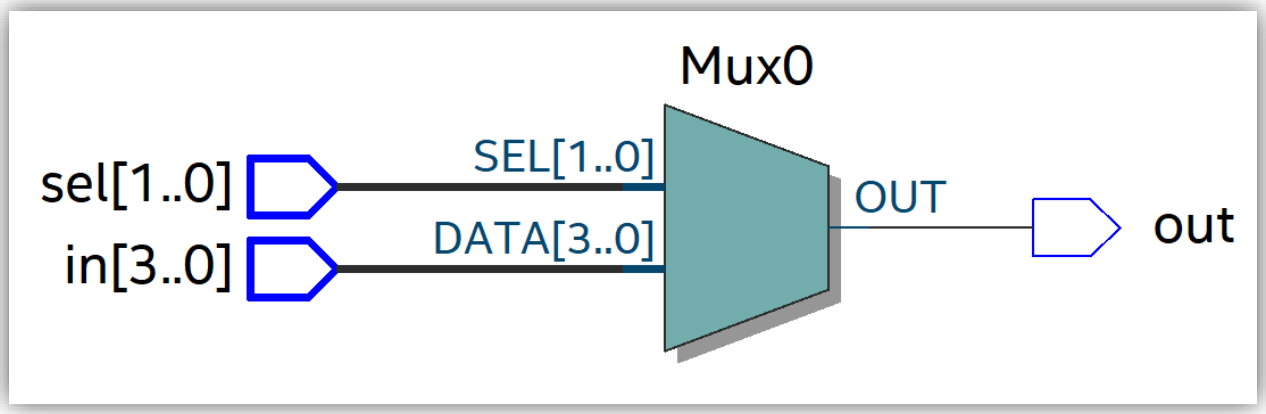
```
module mux_4_1(input [3:0]in,input [1:0]sel,output reg out);
always@(*)
        begin
        case(sel)
        2'b00:out<=in[0];
        2'b01:out<=in[1];
        2'b10:out<=in[2];
        2'b11:out<=in[3];
        default : out<=1'b0;
        endcase
        end
endmodule
```

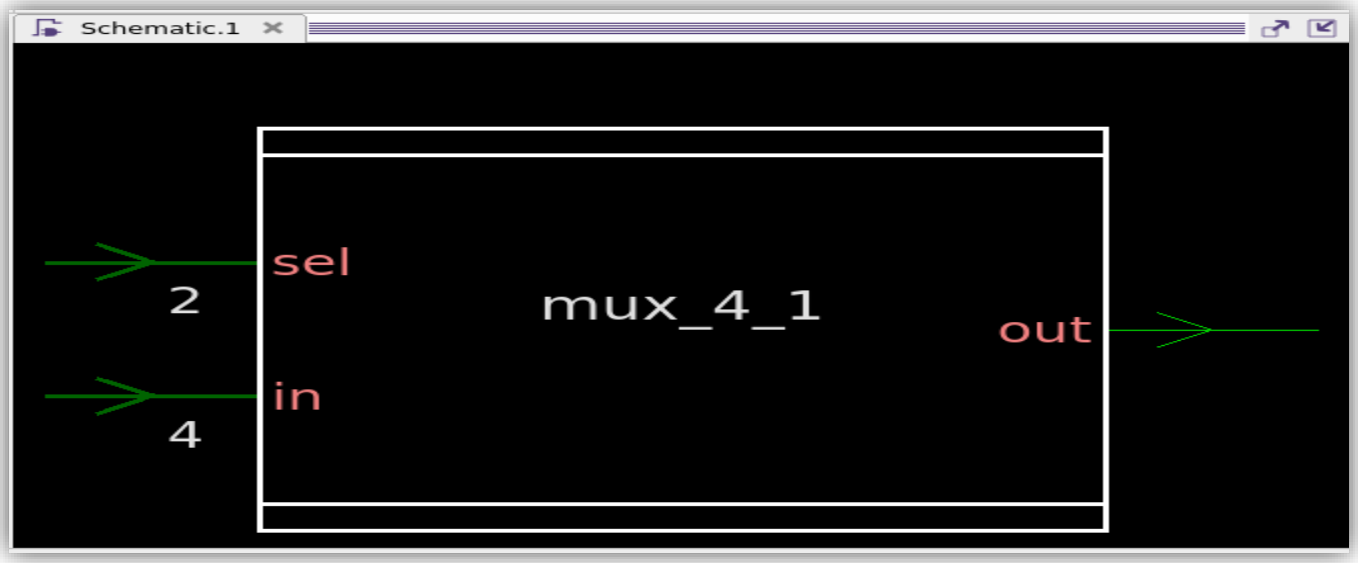
TESTBENCH:

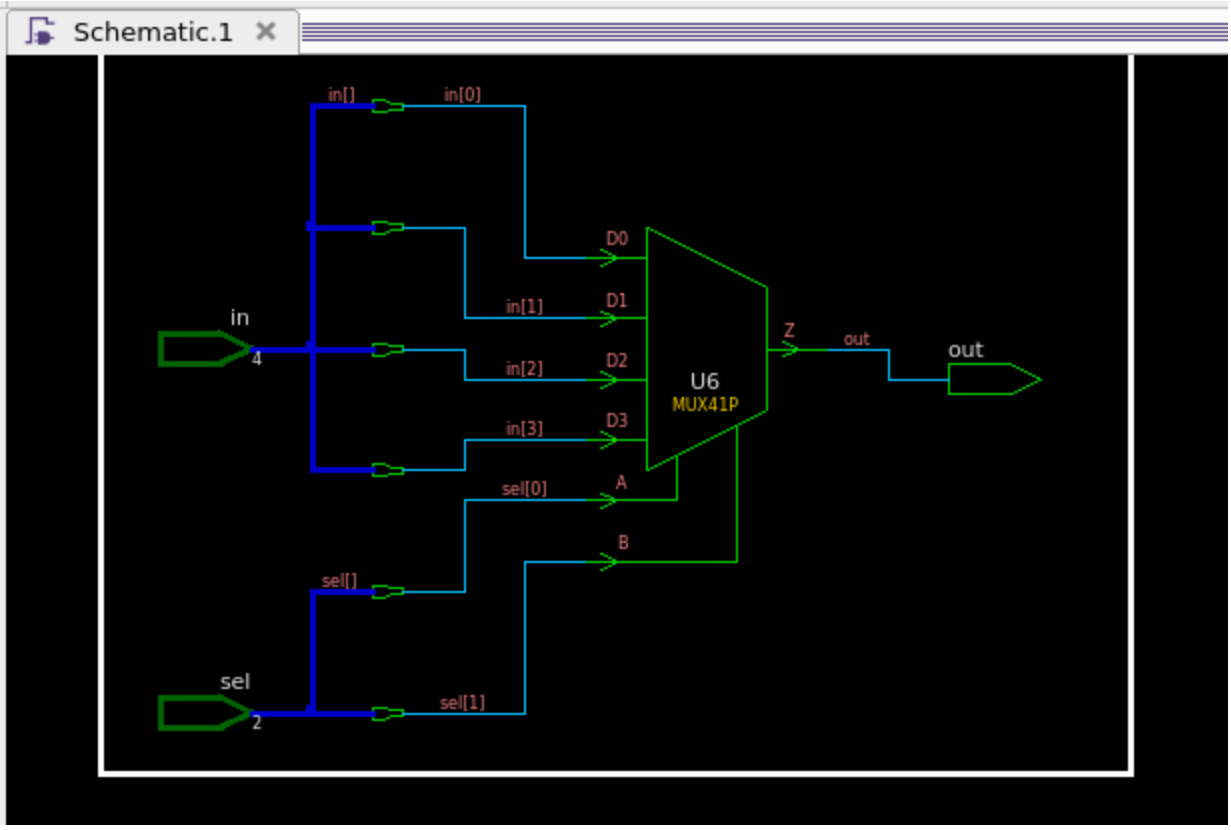
```
module mux_4_1_tb;
reg [3:0]in;
reg [1:0]sel;
wire out;
integer i;
integer j;
mux_4_1 dut(in,sel,out);
initial
        begin
        for(i=0;i<4;i=i+1)
                begin
                sel=i;
                #10;
                end
        for(j=0;j<16;j=j+1)
                begin
```

```
            in=j;
            #10;
            end
        end
endmodule
```









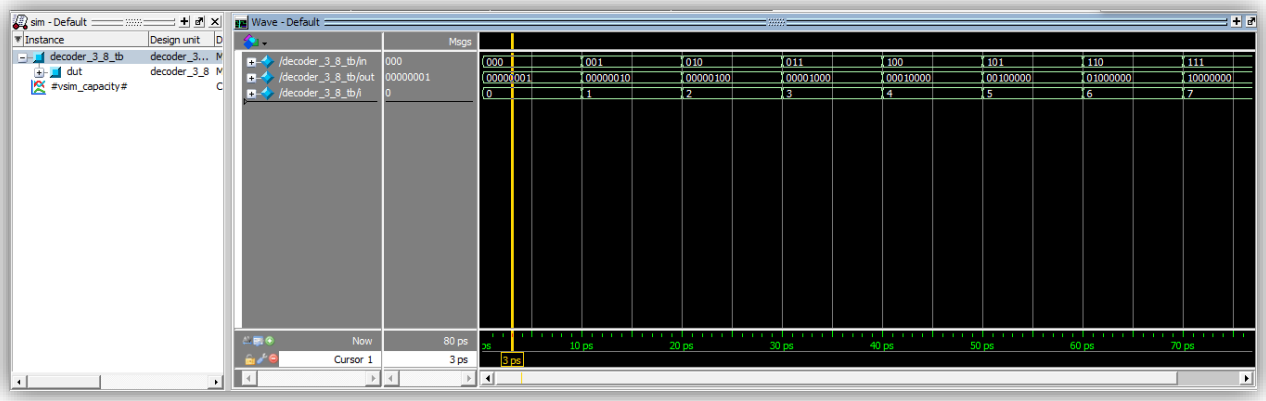
2. Write a behavioral description for a 3:8 decoder and verify using a test bench.

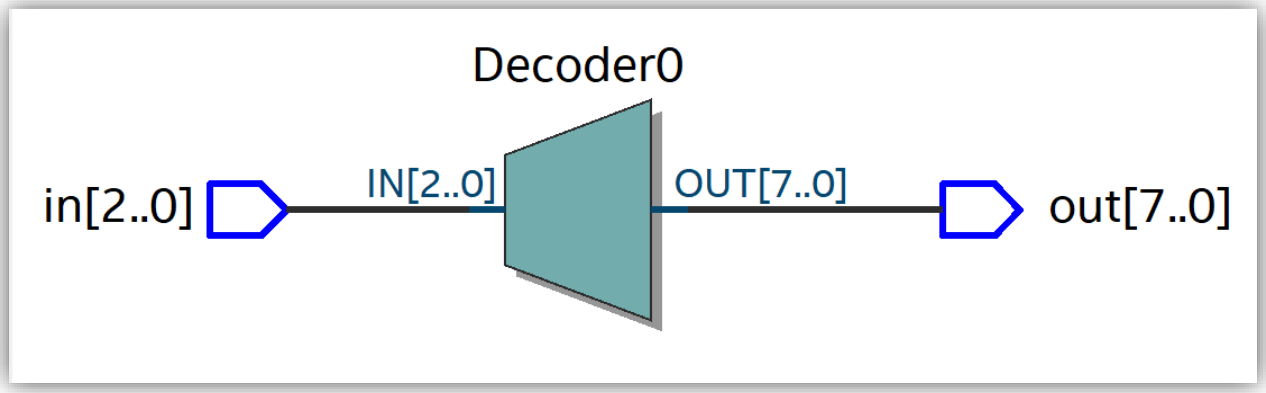
CODE:

```
module decoder_3_8(input [2:0]in, output reg [7:0]out);
always@(*)
begin
        case(in)
        3'b000:out<=8'b0000_0001;
        3'b001:out<=8'b0000_0010;
        3'b010:out<=8'b0000_0100;
        3'b011:out<=8'b0000_1000;
        3'b100:out<=8'b0001_0000;
        3'b101:out<=8'b0010_0000;
        3'b110:out<=8'b0100_0000;
        3'b111:out<=8'b1000_0000;
        default:out<=8'b0000_0000;
        endcase
end
endmodule
```

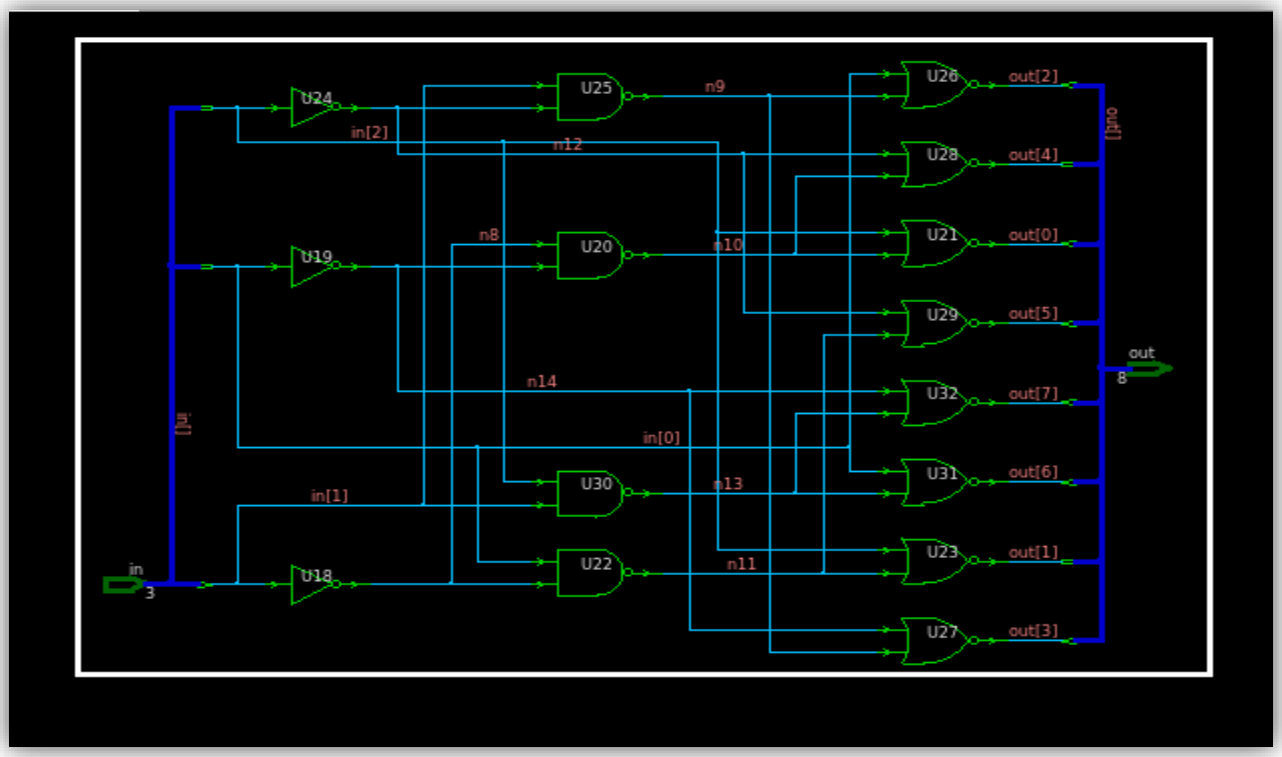
TESTBENCH:

```
module decoder_3_8_tb;
reg [2:0]in;
wire [7:0]out;
integer i;
decoder_3_8 dut(in,out);
initial
        begin
        in=3'b000;
        end
initial
        begin
        for(i=0;i<8;i=i+1)
                begin
                in=i;
                #10;
                end
        end
endmodule
```




Decoder0
in[2..0]
IN[2..0]
OUT[7..0]
out[7..0]


decoder_3_8
in
3
out
8


U24
U25
U26
U28
U19
U20
U21
U29
U32
U30
U31
U22
U23
U18
U27
in[2]
in[1]
in[0]
n8
n9
n10
n11
n12
n13
n14
out[2]
out[4]
out[0]
out[5]
out[7]
out[6]
out[1]
out[3]
out
8
in
3

3.Write a behavioral description and testbench 8:3 priority encoder.

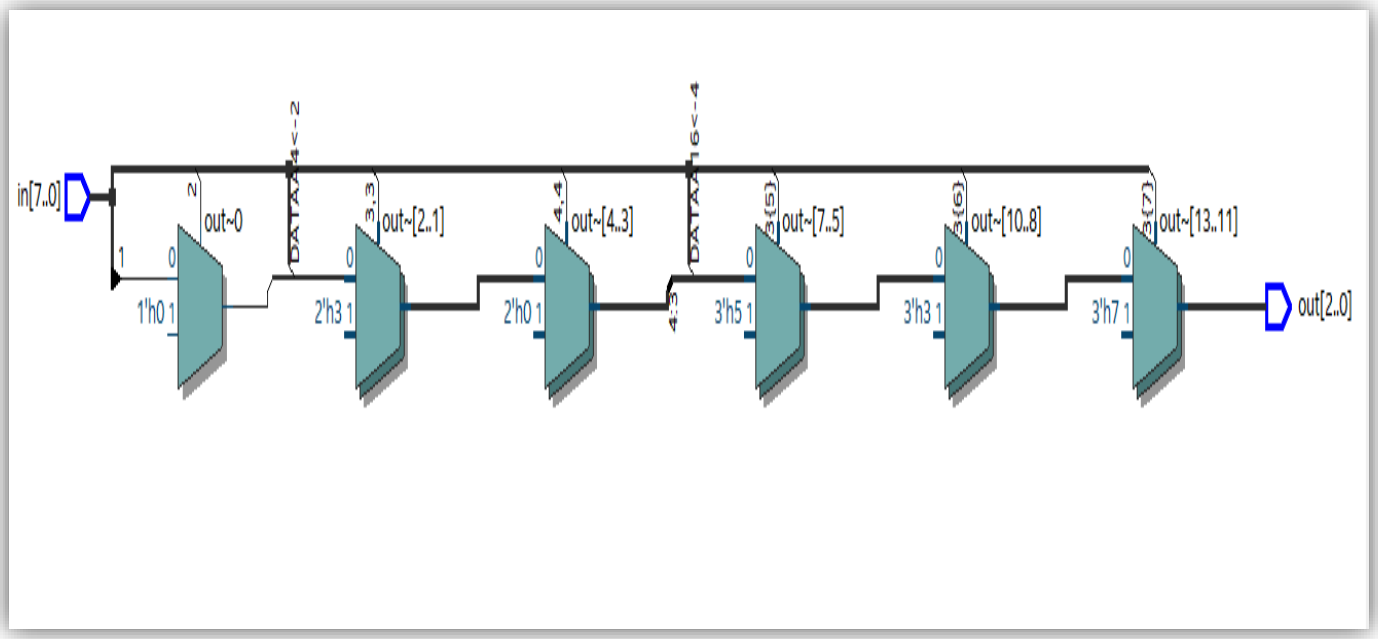
CODE:

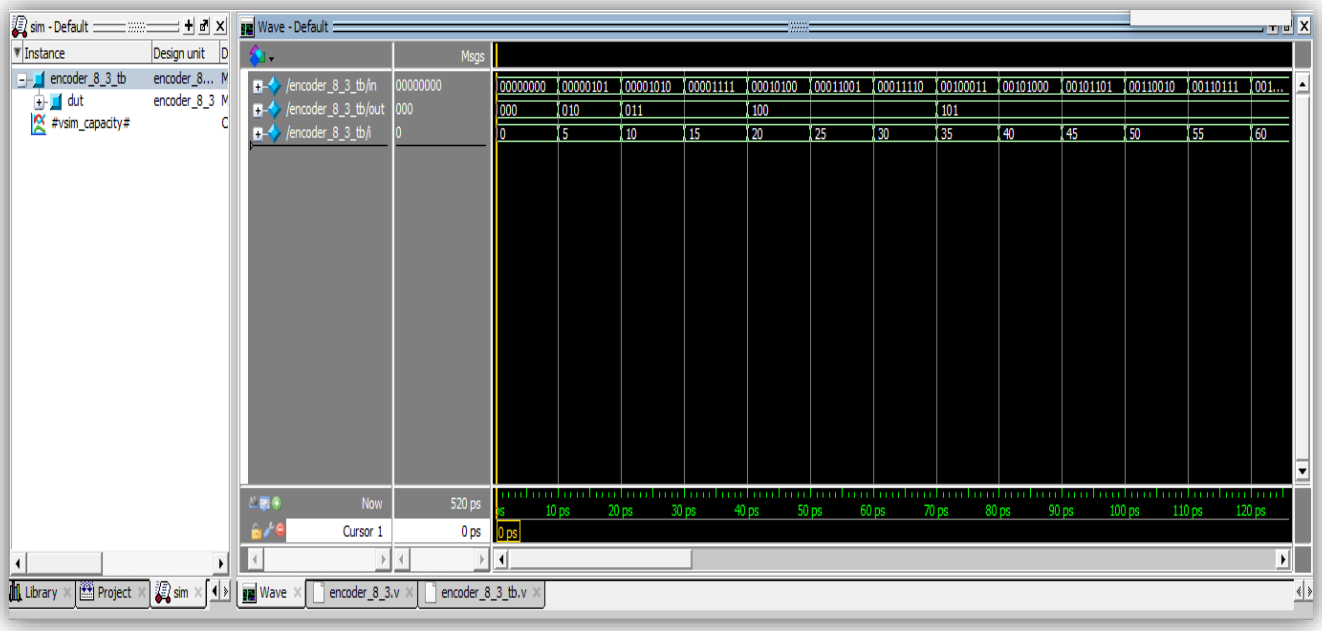
```
module encoder_8_3 (
  input [7:0] in,
  output reg [2:0] out
);

always @*
begin
    if (in[7] == 1)
        out = 3'b111;
    else if (in[6] == 1)
        out = 3'b110;
    else if (in[5] == 1)
        out = 3'b101;
    else if (in[4] == 1)
        out = 3'b100;
    else if (in[3] == 1)
        out = 3'b011;
    else if (in[2] == 1)
        out = 3'b010;
    else if (in[1] == 1)
        out = 3'b001;
    else
        out = 3'b000;
end
endmodule
```

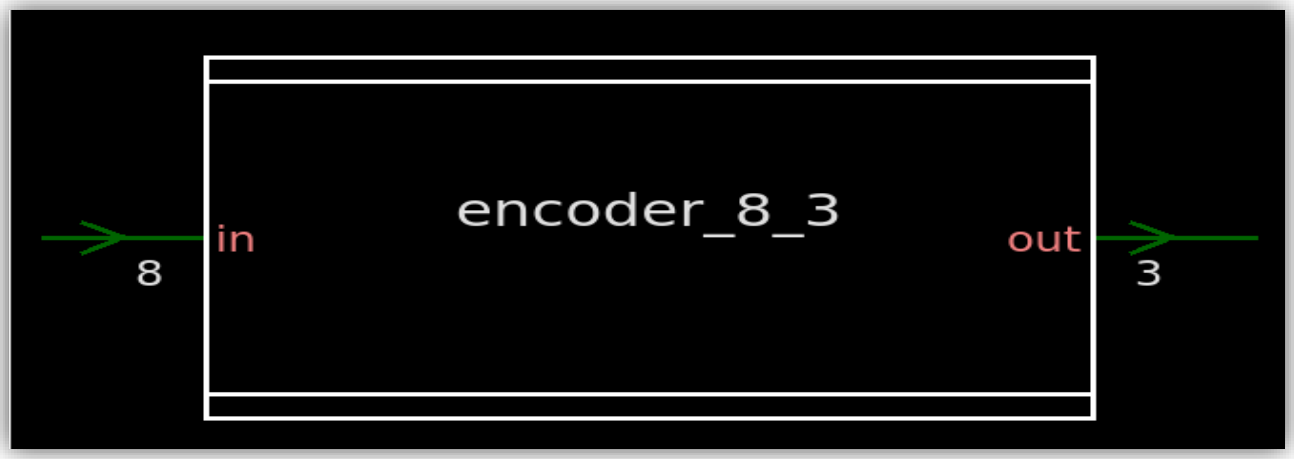
TESTBENCH:

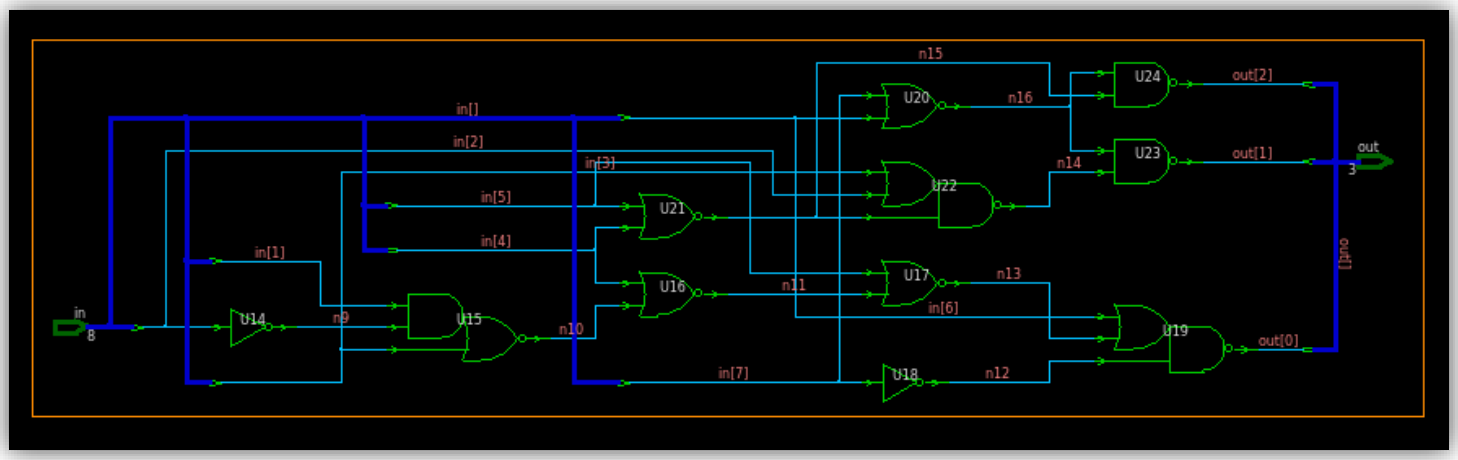
```
module encoder_8_3_tb;
reg [7:0]in;
wire [2:0]out;
integer i;
encoder_8_3 dut(in,out);
```

```
initial
        begin
    for(i=0;i<256;i=i+5)
                begin
                in=i;
                #10;
                end
        end
endmodule
```




sim - Default
Instance
Design unit
encoder_8_3_tb
encoder_8...
dut
encoder_8_3
#vsim_capacity#
Wave - Default
Msgs
/encoder_8_3_tb/in
/encoder_8_3_tb/out
/encoder_8_3_tb/i
00000000
000
0
00000101
00001010
00001111
00010100
00011001
00011110
00100011
00101000
00101101
00110010
00110111
010
011
100
101
5
10
15
20
25
30
35
40
45
50
55
60
Now
520 ps
Cursor 1
0 ps
10 ps
20 ps
30 ps
40 ps
50 ps
60 ps
70 ps
80 ps
90 ps
100 ps
110 ps
120 ps
Library
Project
sim
Wave
encoder_8_3.v
encoder_8_3_tb.v


encoder_8_3
in
8
out
3


in
8
in[]
in[2]
in[3]
in[5]
in[4]
in[1]
in[7]
in[6]
U14
U15
U16
U17
U18
U19
U20
U21
U22
U23
U24
n9
n10
n11
n12
n13
n14
n15
n16
out[2]
out[1]
out[0]
out
3