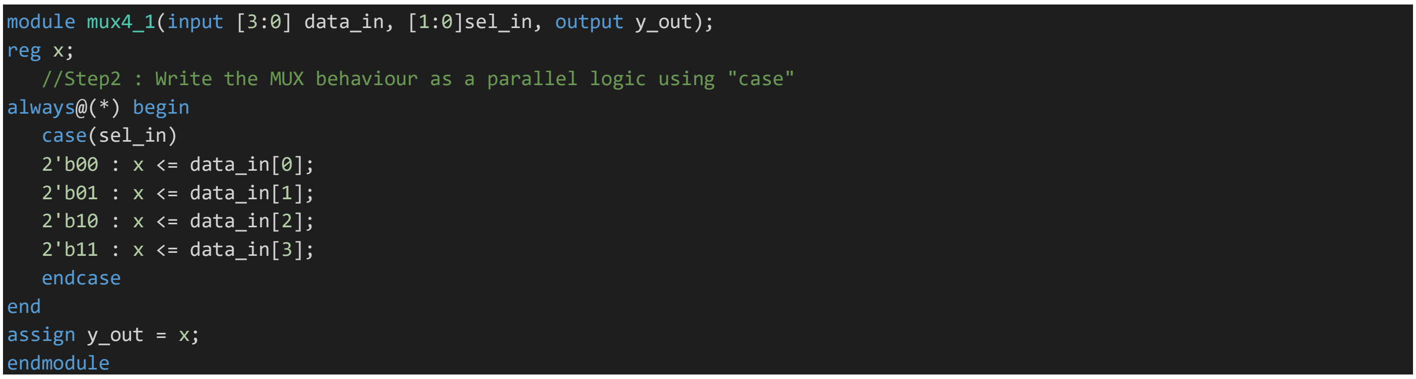
LAB 3 – Raja Aadhithan

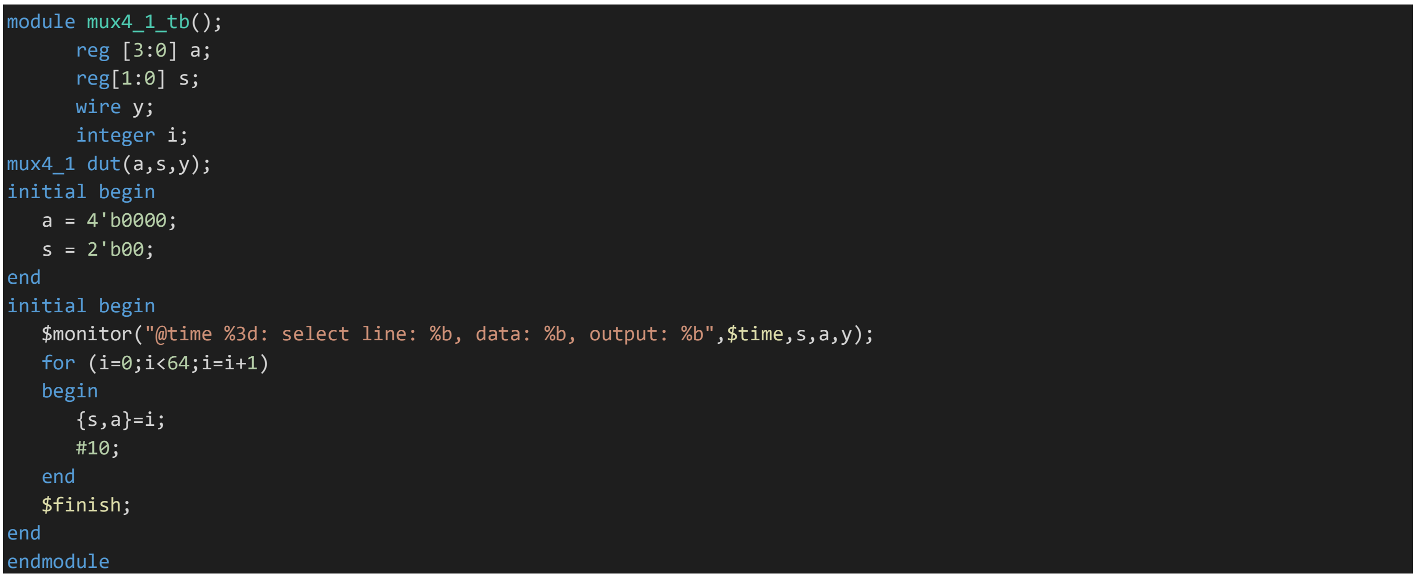
Design – 4x1 mux using behavioral:

Code:



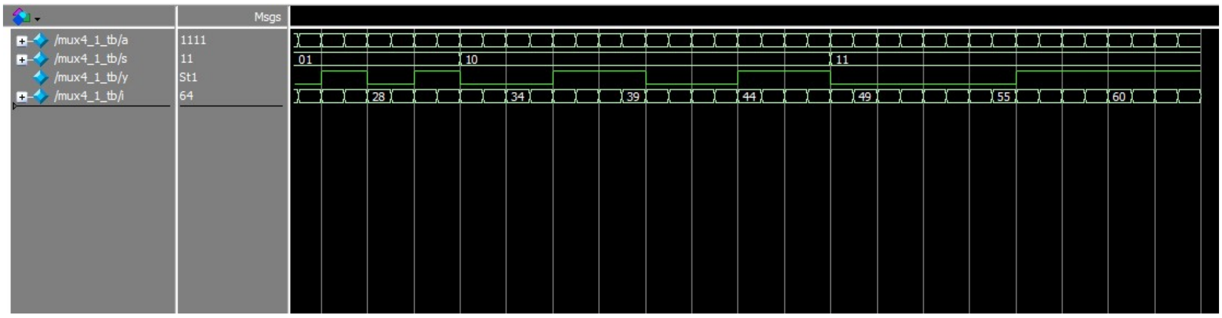
```
module mux4_1(input [3:0] data_in, [1:0]sel_in, output y_out);
reg x;
   //Step2 : Write the MUX behaviour as a parallel logic using "case"
always@(*) begin
   case(sel_in)
   2'b00 : x <= data_in[0];
   2'b01 : x <= data_in[1];
   2'b10 : x <= data_in[2];
   2'b11 : x <= data_in[3];
   endcase
end
assign y_out = x;
endmodule
```

Testbench:

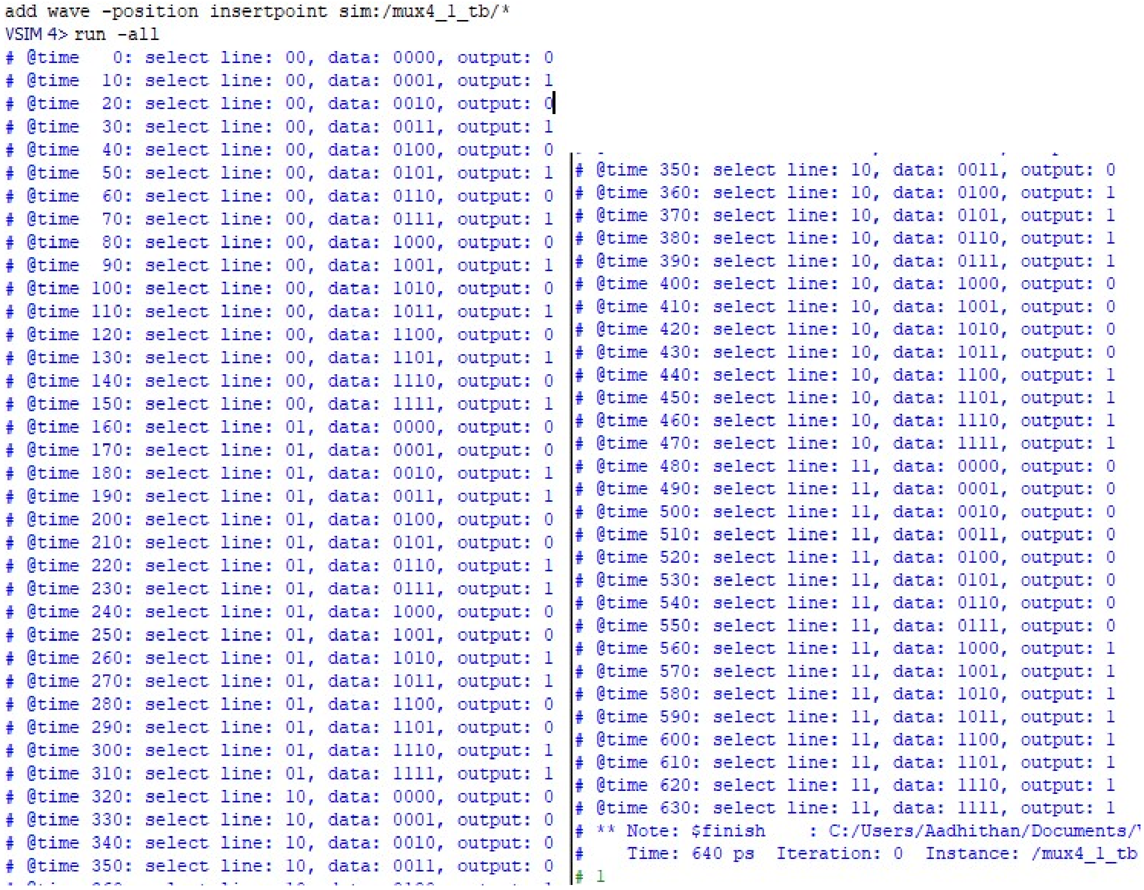


```
module mux4_1_tb();
      reg [3:0] a;
      reg[1:0] s;
      wire y;
      integer i;
mux4_1 dut(a,s,y);
initial begin
   a = 4'b0000;
   s = 2'b00;
end
initial begin
   $monitor("@time %3d: select line: %b, data: %b, output: %b",$time,s,a,y);
   for (i=0;i<64;i=i+1)
   begin
      {s,a}=i;
      #10;
   end
   $finish;
end
endmodule
```

Wave:



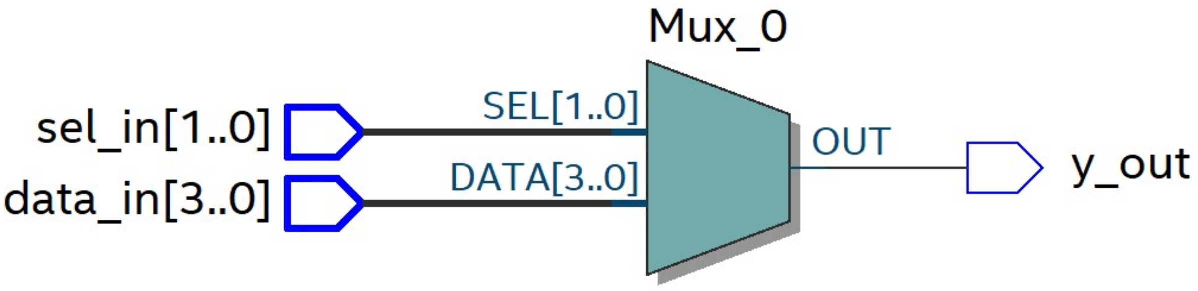
Output:



```
add wave -position insertpoint sim:/mux4_1_tb/*
VSIM 4> run -all
# @time   0: select line: 00, data: 0000, output: 0
# @time  10: select line: 00, data: 0001, output: 1
# @time  20: select line: 00, data: 0010, output: 0
# @time  30: select line: 00, data: 0011, output: 1
# @time  40: select line: 00, data: 0100, output: 0
# @time  50: select line: 00, data: 0101, output: 1
# @time  60: select line: 00, data: 0110, output: 0
# @time  70: select line: 00, data: 0111, output: 1
# @time  80: select line: 00, data: 1000, output: 0
# @time  90: select line: 00, data: 1001, output: 1
# @time 100: select line: 00, data: 1010, output: 0
# @time 110: select line: 00, data: 1011, output: 1
# @time 120: select line: 00, data: 1100, output: 0
# @time 130: select line: 00, data: 1101, output: 1
# @time 140: select line: 00, data: 1110, output: 0
# @time 150: select line: 00, data: 1111, output: 1
# @time 160: select line: 01, data: 0000, output: 0
# @time 170: select line: 01, data: 0001, output: 0
# @time 180: select line: 01, data: 0010, output: 1
# @time 190: select line: 01, data: 0011, output: 1
# @time 200: select line: 01, data: 0100, output: 0
# @time 210: select line: 01, data: 0101, output: 0
# @time 220: select line: 01, data: 0110, output: 1
# @time 230: select line: 01, data: 0111, output: 1
# @time 240: select line: 01, data: 1000, output: 0
# @time 250: select line: 01, data: 1001, output: 0
# @time 260: select line: 01, data: 1010, output: 1
# @time 270: select line: 01, data: 1011, output: 1
# @time 280: select line: 01, data: 1100, output: 0
# @time 290: select line: 01, data: 1101, output: 0
# @time 300: select line: 01, data: 1110, output: 1
# @time 310: select line: 01, data: 1111, output: 1
# @time 320: select line: 10, data: 0000, output: 0
# @time 330: select line: 10, data: 0001, output: 0
# @time 340: select line: 10, data: 0010, output: 0
# @time 350: select line: 10, data: 0011, output: 0
```

```
# @time 350: select line: 10, data: 0011, output: 0
# @time 360: select line: 10, data: 0100, output: 1
# @time 370: select line: 10, data: 0101, output: 1
# @time 380: select line: 10, data: 0110, output: 1
# @time 390: select line: 10, data: 0111, output: 1
# @time 400: select line: 10, data: 1000, output: 0
# @time 410: select line: 10, data: 1001, output: 0
# @time 420: select line: 10, data: 1010, output: 0
# @time 430: select line: 10, data: 1011, output: 0
# @time 440: select line: 10, data: 1100, output: 1
# @time 450: select line: 10, data: 1101, output: 1
# @time 460: select line: 10, data: 1110, output: 1
# @time 470: select line: 10, data: 1111, output: 1
# @time 480: select line: 11, data: 0000, output: 0
# @time 490: select line: 11, data: 0001, output: 0
# @time 500: select line: 11, data: 0010, output: 0
# @time 510: select line: 11, data: 0011, output: 0
# @time 520: select line: 11, data: 0100, output: 0
# @time 530: select line: 11, data: 0101, output: 0
# @time 540: select line: 11, data: 0110, output: 0
# @time 550: select line: 11, data: 0111, output: 0
# @time 560: select line: 11, data: 1000, output: 1
# @time 570: select line: 11, data: 1001, output: 1
# @time 580: select line: 11, data: 1010, output: 1
# @time 590: select line: 11, data: 1011, output: 1
# @time 600: select line: 11, data: 1100, output: 1
# @time 610: select line: 11, data: 1101, output: 1
# @time 620: select line: 11, data: 1110, output: 1
# @time 630: select line: 11, data: 1111, output: 1
# ** Note: $finish    : C:/Users/Aadhithan/Documents/
#    Time: 640 ps  Iteration: 0  Instance: /mux4_1_tb
# 1
```

RTL:



Design: 3:8 Decoder:

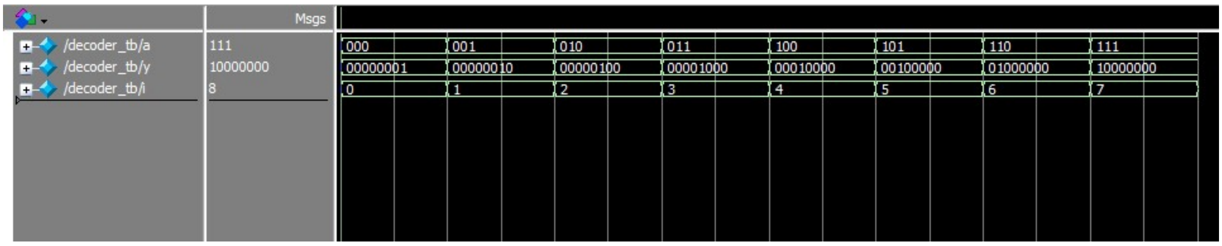
Code:

```
module decoder(input [2:0] in, output [7:0] out);
reg [7:0]temp;
always@(*)begin
    case(in)
    3'd0: temp <= 8'd1;
    3'd1: temp <= 8'd2;
    3'd2: temp <= 8'd4;
    3'd3: temp <= 8'd8;
    3'd4: temp <= 8'd16;
    3'd5: temp <= 8'd32;
    3'd6: temp <= 8'd64;
    3'd7: temp <= 8'd128;
    default : temp <= 8'd0;
    endcase
end
assign out = temp;
endmodule;
```

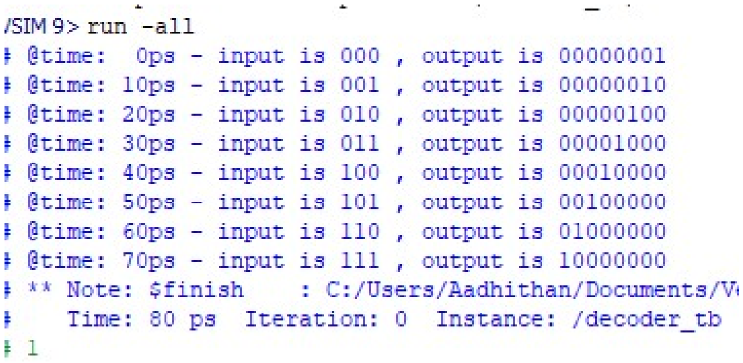
Test bench:

```
module decoder_tb();
reg [2:0]a;
wire [7:0]y;
integer i;
decoder dut(a,y);
initial begin
    $monitor("@time: %2dps - input is %b , output is %b",$time,a,y);
    for(i=0;i<8;i=i+1)begin
        a = i;
        #10;
    end
    $finish;
end
endmodule
```

Wave

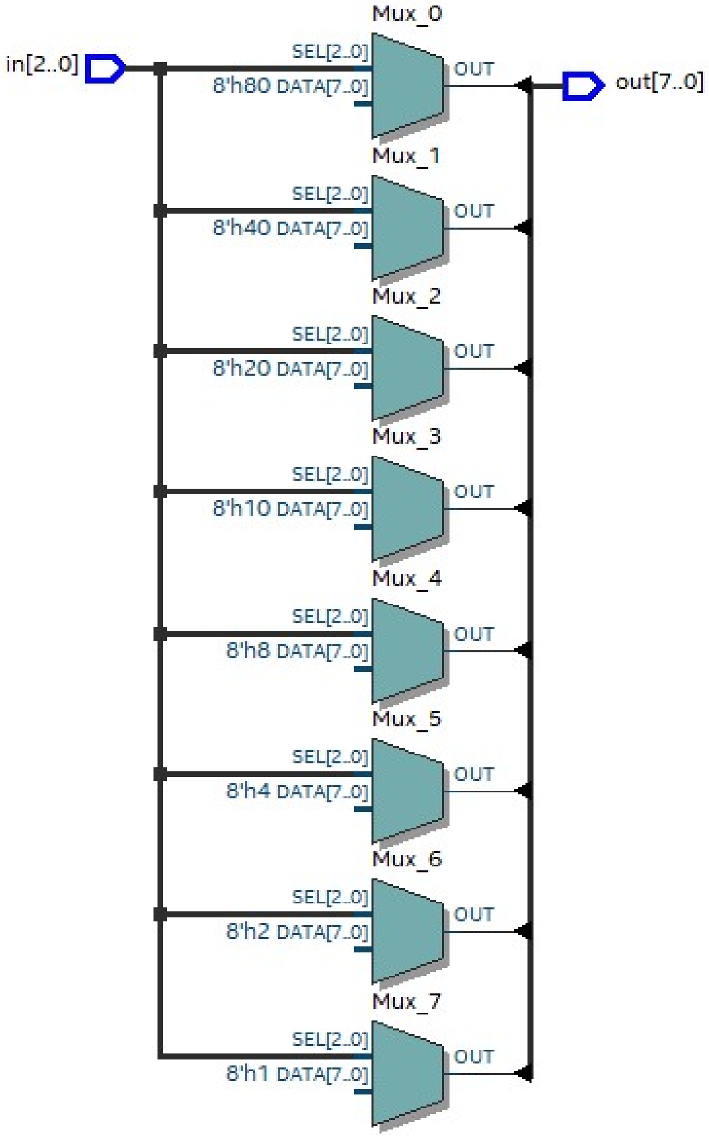


Output :



```
/SIM 9> run -all
# @time:  0ps - input is 000 , output is 00000001
# @time: 10ps - input is 001 , output is 00000010
# @time: 20ps - input is 010 , output is 00000100
# @time: 30ps - input is 011 , output is 00001000
# @time: 40ps - input is 100 , output is 00010000
# @time: 50ps - input is 101 , output is 00100000
# @time: 60ps - input is 110 , output is 01000000
# @time: 70ps - input is 111 , output is 10000000
# ** Note: $finish    : C:/Users/Aadhithan/Documents/V
#    Time: 80 ps  Iteration: 0  Instance: /decoder_tb
# 1
```

RTL:



Design: 8:3 priority encoder:

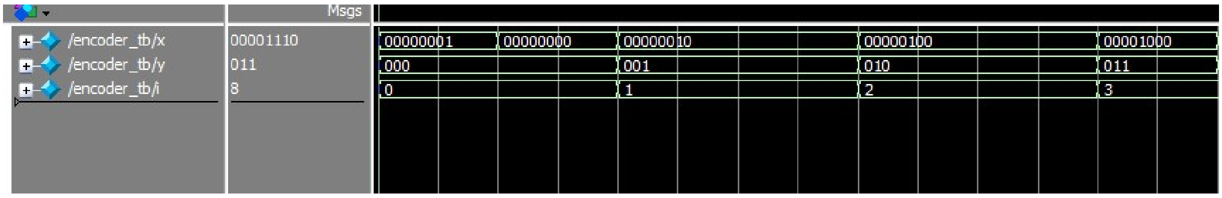
Code:

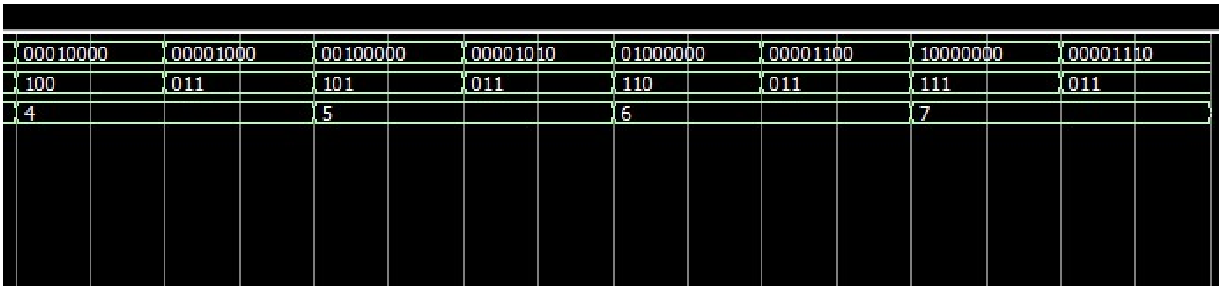
```
module encoder(input [7:0]in , output [2:0] out);
assign out[2] = in[7]|in[6]|in[5]|in[4];
assign out[1] = in[7]|in[6]| (~in[5]&~in[4]&(in[3]|in[2]));
assign out[0] = in[7] | (~in[6]&in[5]) | (~in[6]&~in[5]&~in[4]&~in[3]&~in[2]&in[1]) | (~in[6]&~in[5]&~in[4]&in[3]);
endmodule
```

Test bench:

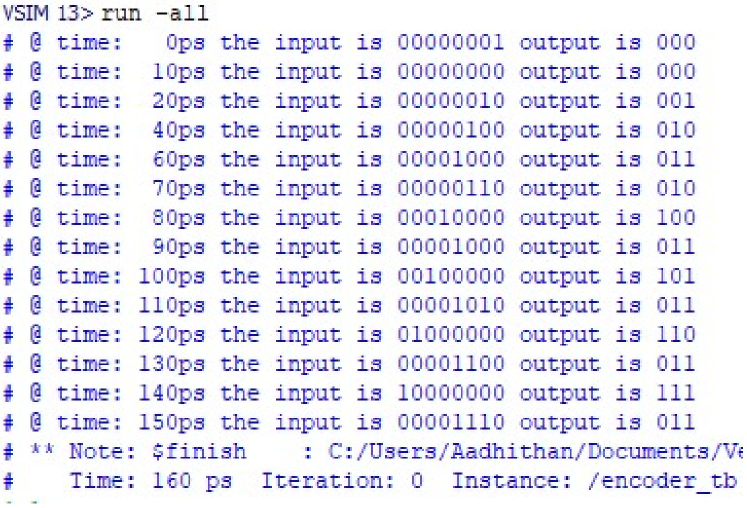
```
module encoder_tb();
reg [7:0] x;
wire [2:0] y;
integer i;
encoder dat(x,y);
initial begin
    for(i=0; i<8; i = i+1)
    begin
        x = 2**i;
        #10;
        x=2*i;
        #10;
    end
$finish;
end

initial $monitor("@ time: %3dps the input is %8b output is %3b",$time,x,y);

endmodule
```

Wave:



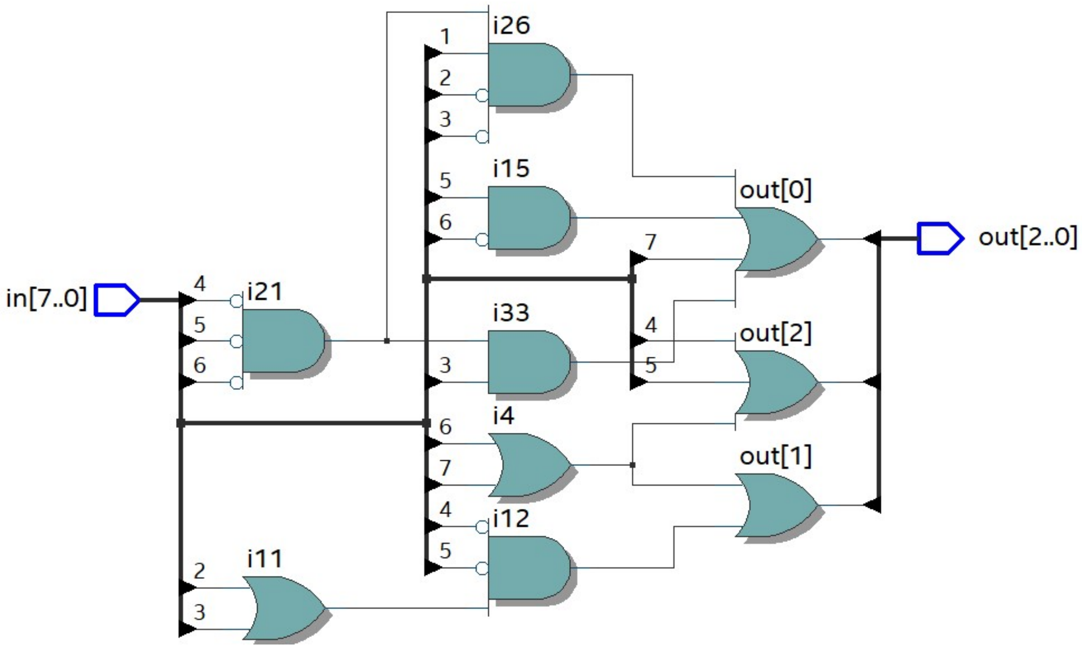


Output:



```
VSIM 13> run -all
# @ time:   0ps the input is 00000001 output is 000
# @ time:  10ps the input is 00000000 output is 000
# @ time:  20ps the input is 00000010 output is 001
# @ time:  40ps the input is 00000100 output is 010
# @ time:  60ps the input is 00001000 output is 011
# @ time:  70ps the input is 00000110 output is 010
# @ time:  80ps the input is 00010000 output is 100
# @ time:  90ps the input is 00001000 output is 011
# @ time: 100ps the input is 00100000 output is 101
# @ time: 110ps the input is 00001010 output is 011
# @ time: 120ps the input is 01000000 output is 110
# @ time: 130ps the input is 00001100 output is 011
# @ time: 140ps the input is 10000000 output is 111
# @ time: 150ps the input is 00001110 output is 011
# ** Note: $finish    : C:/Users/Aadhithan/Documents/Ve
#    Time: 160 ps  Iteration: 0  Instance: /encoder_tb
```

RTL :



in[7..0]
4
5
6
i21
2
3
i11
1
2
3
i26
5
6
i15
3
i33
6
7
i4
4
5
i12
7
4
5
out[0]
out[2]
out[1]
out[2..0]