1.THREE BIT RING COUNTER

CODE:

module ringcounter(out, clk, reset);

input clk, reset;

output reg [2:0] out;

always @(posedge clk, posedge reset)

begin

if (reset)

out=4'b100;

else

begin

out = {out[0],out[2:1]};

end

end

endmodule

TEST\_BENCH CODE:

module ringcounter\_tb;

reg t\_clk,t\_reset;

wire [2:0] t\_out;

ringcounter mycounter(.clk(t\_clk),.reset(t\_reset),.out(t\_out));

initial t\_clk = 0;

always #10 t\_clk = ~t\_clk;

initial begin

// Apply Inputs

$monitor("clk=%d reset=%d Output %b",t\_clk,t\_reset,t\_out);

t\_reset = 0;

#100;

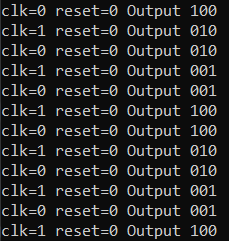
t\_reset = 1;

#100;

t\_reset = 0;

end

endmodule

OUTPUT:  


2.THREE BIT DOWN COUNTER

CODE:

module downcounter(input clk,input reset,output[2:0] counter);

reg [2:0] counter;

always@(posedge clk)

begin

if(reset==1'b1)

begin

counter=3'b000;

end

else

begin

counter=counter-1;

end

end

endmodule

TEST-BENCH CODE:

module c\_tb();

reg clk,reset;

wire [0:2] counter;

downcounter mycounter(clk,reset,counter);

initial

begin

reset=1;clk=1;#10;

reset=0;clk=0;#10;

repeat(32) begin

clk=~clk;#10;

if(clk==1'b1)begin

$display("clk=%b reset=%b counter=%b",clk,reset,counter);

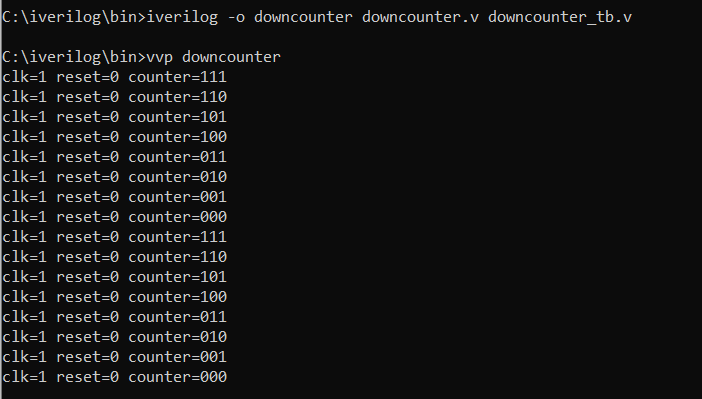
end

end

end

endmodule

OUTPUT:



3.THREE BIT COMPARATOR

Code:

module comparator3bit(a,b,equal,gt,lt);

input [2:0] a,b;

output reg equal,gt,lt;

always@(a,b)

begin

if(a==b)

begin

equal=1'b1;

gt=1'b0;

lt=1'b0;

end

else if(a<b)

begin

equal=1'b0;

gt=1'b0;

lt=1'b1;

end

else if(a>b)

begin

equal=1'b0;

gt=1'b1;

lt=1'b0;

end

end

endmodule

Test Bench Code:

module comparator3bit\_tb;

reg [2:0] t\_a,t\_b;

comparator3bit cm(t\_a,t\_b,t\_equal,t\_gt,t\_lt);

initial begin

$display(" 3-bit Comparator ");

$display("");

$display("A \tB \t\tA=B\tA>B\tA<B");

$monitor("%b\t%b\t\t%b \t%b \t%b",t\_a,t\_b,t\_equal,t\_gt,t\_lt);

t\_a=3'b111;

t\_b=3'b001;

#5

t\_a=3'b101;

t\_b=3'b110;

#5

t\_a=3'b000;

t\_b=3'b101;

#5

t\_a=3'b110;

t\_b=3'b101;

#5

t\_a=3'b100;

t\_b=3'b100;

#5

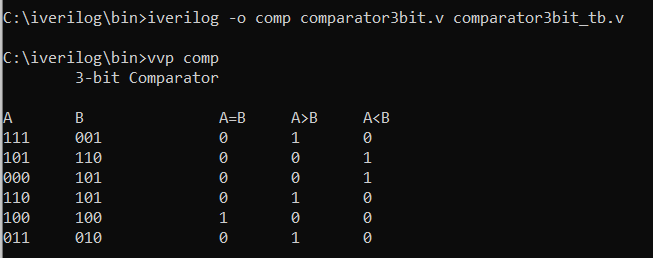
t\_a=3'b011;

t\_b=3'b010;

end

endmodule

OUTPUT:



4.8X3 ENCODER:

CODE:

module encoder83(Y0,Y1,Y2,Y3,Y4,Y5,Y6,Y7,A,B,C);

input Y0,Y1,Y2,Y3,Y4,Y5,Y6,Y7;

output A,B,C;

wire a,b,c,d,e,f;

or(a,Y4,Y5);

or(b,a,Y6);

or(A,b,Y7);

or(c,Y2,Y3);

or(d,c,Y6);

or(B,d,Y7);

or(e,Y1,Y3);

or(f,e,Y5);

or(C,f,Y7);

endmodule

TEST BENCH CODE:

module encoder83\_tb;

reg t\_Y0,t\_Y1,t\_Y2,t\_Y3,t\_Y4,t\_Y5,t\_Y6,t\_Y7;

encoder83 ec(t\_Y0,t\_Y1,t\_Y2,t\_Y3,t\_Y4,t\_Y5,t\_Y6,t\_Y7,t\_A,t\_B,t\_C);

initial begin

$display("");

$display("Y7\tY6\tY5\tY4\tY3\tY2\tY1\tY0\t\tA \tB \tC");

$monitor("%b\t%b\t%b\t%b\t%b\t%b\t%b\t%b\t\t%b\t%b\t%b\t",t\_Y7,t\_Y6,t\_Y5,t\_Y4,t\_Y3,t\_Y2,t\_Y1,t\_Y0,t\_A,t\_B,t\_C);

t\_Y7=1'b0;

t\_Y6=1'b0;

t\_Y5=1'b0;

t\_Y4=1'b0;

t\_Y3=1'b0;

t\_Y2=1'b0;

t\_Y1=1'b0;

t\_Y0=1'b1;

#5

t\_Y7=1'b0;

t\_Y6=1'b0;

t\_Y5=1'b0;

t\_Y4=1'b0;

t\_Y3=1'b0;

t\_Y2=1'b0;

t\_Y1=1'b1;

t\_Y0=1'b0;

#5

t\_Y7=1'b0;

t\_Y6=1'b0;

t\_Y5=1'b0;

t\_Y4=1'b0;

t\_Y3=1'b0;

t\_Y2=1'b1;

t\_Y1=1'b0;

t\_Y0=1'b0;

#5

t\_Y7=1'b0;

t\_Y6=1'b0;

t\_Y5=1'b0;

t\_Y4=1'b0;

t\_Y3=1'b1;

t\_Y2=1'b0;

t\_Y1=1'b0;

t\_Y0=1'b0;

#5

t\_Y7=1'b0;

t\_Y6=1'b0;

t\_Y5=1'b0;

t\_Y4=1'b1;

t\_Y3=1'b0;

t\_Y2=1'b0;

t\_Y1=1'b0;

t\_Y0=1'b0;

#5

t\_Y7=1'b0;

t\_Y6=1'b0;

t\_Y5=1'b1;

t\_Y4=1'b0;

t\_Y3=1'b0;

t\_Y2=1'b0;

t\_Y1=1'b0;

t\_Y0=1'b0;

#5

t\_Y7=1'b0;

t\_Y6=1'b1;

t\_Y5=1'b0;

t\_Y4=1'b0;

t\_Y3=1'b0;

t\_Y2=1'b0;

t\_Y1=1'b0;

t\_Y0=1'b0;

#5

t\_Y7=1'b1;

t\_Y6=1'b0;

t\_Y5=1'b0;

t\_Y4=1'b0;

t\_Y3=1'b0;

t\_Y2=1'b0;

t\_Y1=1'b0;

t\_Y0=1'b0;

end

endmodule

OUTPUT:

