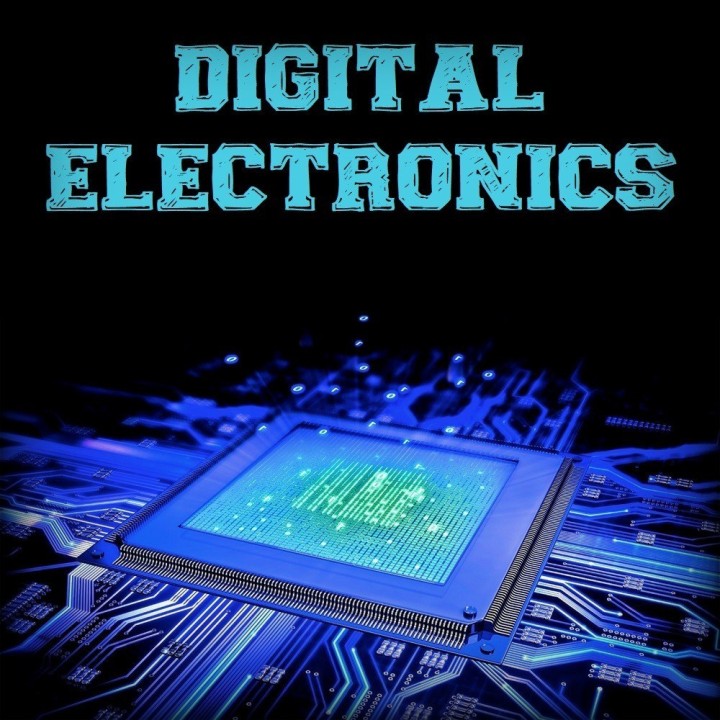
***Abanob Evram***

***Assignmen1[Extra]***



**[Q1]**

A screenshot of a computer program

Description automatically generated

**The design code:**

module encoder(A,B);

parameter USE\_GRAY=0;

input [2:0] A;

output reg [6:0] B;

generate

if (USE\_GRAY==0)

always @(A)

case(A)

0:B=0;

1:B=1;

2:B=2;

3:B=4;

4:B=8;

5:B=16;

6:B=32;

7:B=64;

endcase

else

always @(A)

case(A)

0:B=0;

1:B=1;

2:B=3;

3:B=2;

4:B=6;

5:B=7;

6:B=5;

7:B=4;

endcase

endgenerate

endmodule

**The testbench code for gray:**

module encoder\_gray\_tb();

parameter USE\_GRAY\_tb=1;

reg [2:0] A\_tb;

reg [6:0] B\_expected;

wire [6:0] B\_dut;

encoder #(USE\_GRAY\_tb) dut(A\_tb,B\_dut);

integer i ;

initial begin

for(i=0;i<8;i=i+1) begin

A\_tb=i;

case(A\_tb)

0:B\_expected=0;

1:B\_expected=1;

2:B\_expected=3;

3:B\_expected=2;

4:B\_expected=6;

5:B\_expected=7;

6:B\_expected=5;

7:B\_expected=4;

endcase

#10

if (B\_expected!=B\_dut) begin

$display("Errror....");

$stop;

end

end

$stop;

end

initial begin

$monitor("A\_tb=%d,B\_expected=%b",A\_tb,B\_expected);

end

endmodule

**The testbench code for one hot:**

module encoder\_one\_hot\_tb();

parameter USE\_GRAY\_tb=0;

reg [2:0] A\_tb ;

reg [6:0] B\_expected;

wire [6:0] B\_dut;

encoder #(USE\_GRAY\_tb) dut(A\_tb,B\_dut);

integer i ;

initial begin

for(i=0;i<8;i=i+1) begin

A\_tb = i;

case(A\_tb)

0:B\_expected=0;

1:B\_expected=1;

2:B\_expected=2;

3:B\_expected=4;

4:B\_expected=8;

5:B\_expected=16;

6:B\_expected=32;

7:B\_expected=64;

endcase

#10

if (B\_expected!=B\_dut) begin

$display("Errror....");

$stop;

end

end

$stop;

end

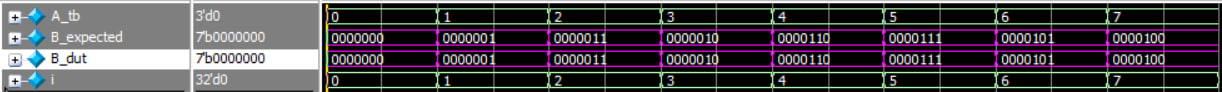
initial begin

$monitor("A\_tb=%d,B\_expected=%b",A\_tb,B\_expected);

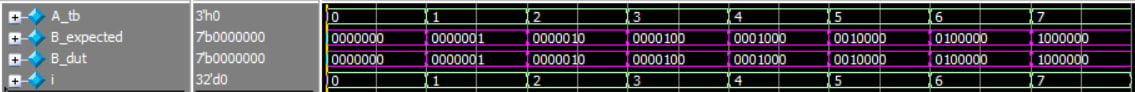
end

endmodule

**Wave of Gray\_code :-**



**Wave of One\_hot :-**



A diagram of a computer

Description automatically generated**[Q2]**

**The design code:**

module Demux(D,S,Y);

input D;

input [1:0] S;

output reg [3:0] Y;

always @(\*) begin

case(S)

0:Y={3'b000,D};

1:Y={2'b00,D,1'b0};

2:Y={1'b0,D,2'b00};

3:Y={D,3'b000};

endcase

end

endmodule

**The testbench code:**

module Demux\_tb();

reg D\_tb;

reg [1:0] S\_tb;

reg [3:0] Y\_excpected;

wire [3:0] Y\_dut;

Demux dut(D\_tb,S\_tb,Y\_dut);

integer i,j;

initial begin

for(i=0;i<2;i=i+1) begin

D\_tb=i;

for(j=0;j<4;j=j+1)begin

S\_tb=j;

case(S\_tb)

0:Y\_excpected={3'b000,D\_tb};

1:Y\_excpected={2'b00,D\_tb,1'b0};

2:Y\_excpected={1'b0,D\_tb,2'b00};

3:Y\_excpected={D\_tb,3'b000};

endcase

#10

if(Y\_excpected!=Y\_dut)begin

$display("Errror");

$stop;

end

end

end

$stop;

end

initial begin

$monitor("D\_tb=%d , Y\_excpected=%b",D\_tb,Y\_excpected);

end

A screenshot of a computer

Description automatically generatedendmodule