CS-223 Digital Design

Lab3 Preliminary Report

Ege İpekci 21902333

08.06.2023

B)

```
module one_to_two_decoder(
input logic  d,
input logic en,
output logic [1:0]y
   );
   always @ ( d or en)
   if( en == 1'b1 )
   begin
     casez(d)
   1'b0 : y = 2'b01;
   1'b1 : y = 2'b10;
   endcase
   end
      else
   begin
    y = 2'b00;
   end
endmodule

module testbench();

logic d;
logic en;
logic [1:0] y;

one_to_two_decoder uut( d, en, y);

initial begin
   en = 0; #10;
   d=0;  #10;
```

```
   d=1; #10;
   en = 1; #10;
   d=0; #10;
   d=1; #10;
   end
endmodule
```

C)

```
module two_to_four_decoder(
input logic [1:0]d,
input logic en,
output logic [3:0]y
   );
   logic [1:0]t;
   one_to_two_decoder d1(d[1], en,t );
   one_to_two_decoder d2(d[0], t[0], y[1:0] );
   one_to_two_decoder d3( d[0], t[1], y[3:2]);
   endmodule

module testbench(
   );
 logic [1:0]d;
 logic en;
 logic [3:0]y;

   two_to_four_decoder uut(d,en,y);

initial begin
   en=0; #10;
   d[1]=0; d[0]=0; ; #10;
   d[0]=1; #10;
```

```
   d[1]=1; d[0]=0; #10;
   d[0]=1; #10;
   en=1; #10;
   d[1]=0; d[0]=0; ; #10;
   d[0]=1; #10;
   d[1]=1; d[0]=0; #10;
   d[0]=1; #10;
   end
endmodule
```

D)

```
module twoToOneMux(
input logic m[1:0],
input logic s,
output logic y
   );

   always @ (m or s or y)
   if ( s == 0 )
      y = m[0];
   else
      y = m[1];
endmodule
```

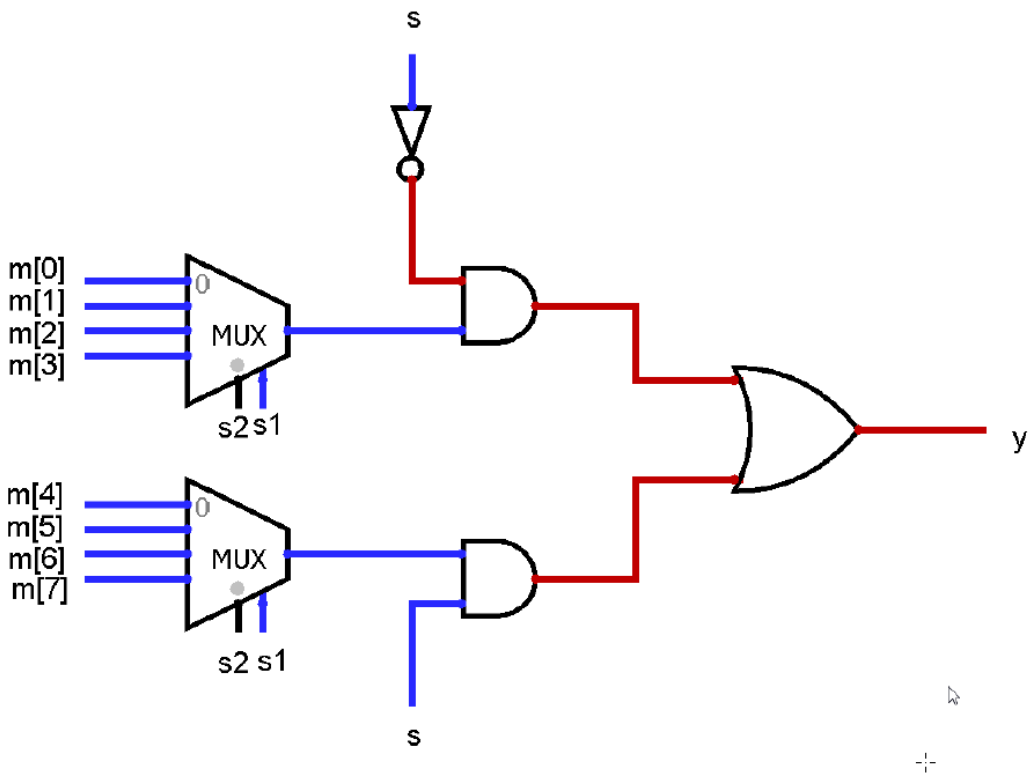
E)

```
module four_to_one_mux(
input logic m[3:0],
input logic s[1:0],
output logic y
   );
   logic y1, y2;
```

```
   twoToOneMux m1( m[1:0], s[0], y1 );
   twoToOneMux m2( m[3:2], s[0], y2 );

   logic temp[1:0];
   assign temp[0] = y1;
   assign temp[1] = y2;

   twoToOneMux m3( temp, s[1], y) ;
endmodule

module testbench();
logic m[3:0];
logic s[1:0];
logic y;
   four_to_one_mux uut( m[3:0],s, y );


   initial begin
   for(int i = 0; i <64; i++)  begin
      if(i % 2 == 0)
         m[0] = 0;
      else
         m[0] = 1;

      if(i % 4 == 0 || i % 4 == 1 )
         m[1] = 0;
      else
         m[1] = 1;
      if(i%8==0||i%8==1||i%8==2||i%8==3)
         m[2] = 0;
      else
```

```
        m[2] = 1;
      if(i%16==0||i%16==1||i%16==2||i%16==3||i%16==4||i%16==5||i%16==6||i%16==7)
        m[3] = 0;
      else
        m[3] = 1;
      if( ( i % 32 ) - 15 <0 )
        s[0] = 0;
      else
        s[0] = 1;
      if(i - 32 < 0)
        s[1] = 0;
      else
        s[1] = 1;

      #10;
    end
  end
endmodule
```

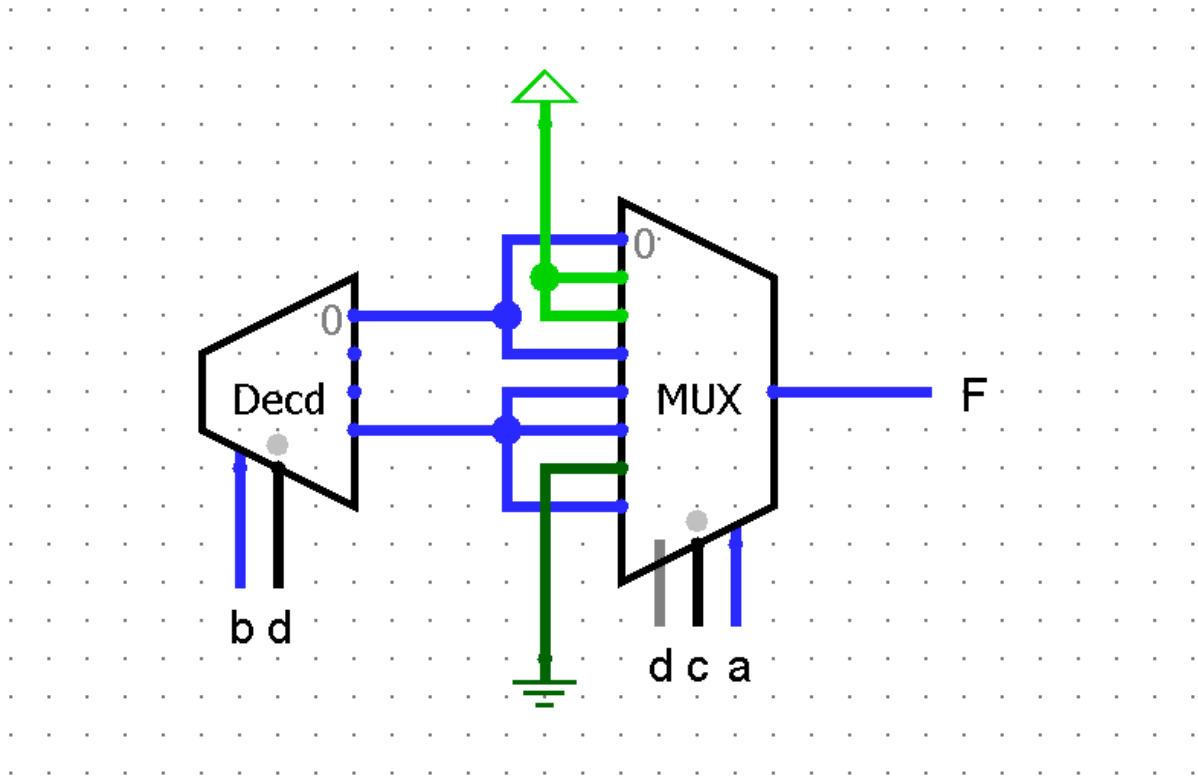
F)




```
module AND(
   input logic a,b,
   output logic v
   );
   assign v = a & b;
endmodule

module INV(
   input logic a,
   output logic y
   );
   assign y = ~a;
endmodule
```

```
module OR(
   input logic a,b,
   output logic z
   );
   assign z = a || b;
endmodule

module eight_to_one_mux(
   input logic [7:0]m,
   input logic [2:0]s3,
   output logic y
   );
   logic y1, y2, y3, y4, y5;

   fourToOneMux mux1( m[3:0],s3[1:0], y1 );
   INV inv1( s3[2], y5 );
   AND and1(y1, y5, y3 );

   fourToOneMux mux2( m[7:4],s3[1:0], y2 );
   AND and2(y2, s3[2], y4 );
   OR or1( y3, y4, y);
Endmodule
```

G)




```
module mux2(
   input logic a,b,c,d,
  output logic y
   );
      logic [3:0]o;
      logic [2:0]s;
      logic [1:0]s1;
      logic [7:0]m;

      assign s[2] = d;
      assign s[1] = c;
      assign s[0] = a;

      assign s1[1] = b;
      assign s1[0] = d;
       two_to_four_decoder decoder(s1, 1, o);
```

```
    assign m[0] = o[0];
    assign m[1] = 1;
    assign m[2] = 1;
    assign m[3] = o[0];
    assign m[4] = o[3];
    assign m[5] = o[3];
    assign m[6] = 0;
    assign m[7] = o[3];


    eight_to_one_mux mux(m,s,y);

endmodule
```