DIGITAL DESIGN

CS223

Section: 001

Lab3 Report

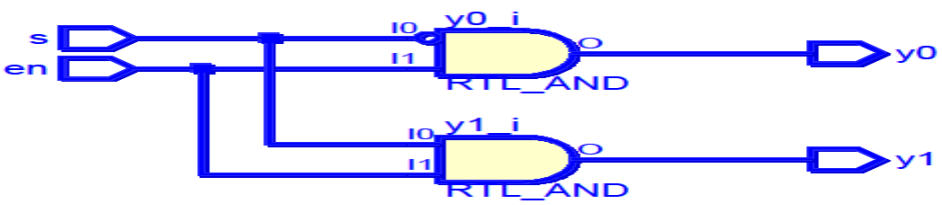
Name: Tolga Han Arslan

ID: 22003061

Date: 31.10.2022

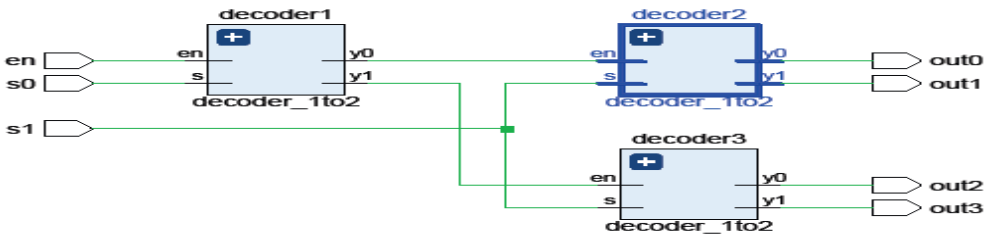
b) Behavioral SystemVerilog module for a 1-to-2 decoder (including enable signal) and a testbench for it.

```
module decoder_1to2(
    input en,
    input s,
    output y0,
    output y1
    );
    assign y0 = ~s & en;
    assign y1 = s & en;
endmodule

module decoder_1to2_tb();
    logic s,en,y1,y2;
    decoder_1to2 dut(en,s,y1,y2);
    initial begin
    s=0; en=0; #10;
    s=1; #10;
    en=1; s=0; #10;
    s=1; #10;
    end
endmodule
```



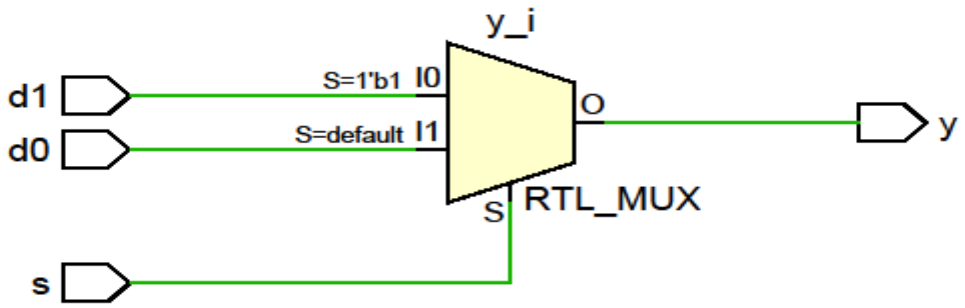
c) Structural SystemVerilog module for a 2-to-4 decoder (including enable signal) using three 1-to-2 decoders. Prepare a testbench for it.

```
module decoder_2to4(
    input en,
    input s0,
    input s1,
    output out0,
    output out1,
    output out2,
    output out3
    );
    logic y0,y1;
    decoder_1to2 decoder1(en,s0,y0,y1);
    decoder_1to2 decoder2(y0,s1,out0,out1);
    decoder_1to2 decoder3(y1,s1,out2,out3);
endmodule

module decoder_2to4_tb();
logic en,s0,s1,y0,y1,y2,y3;
decoder_2to4 dut(en,s0,s1,y0,y1,y2,y3);
    initial begin
    en=0; s0=0; s1=0; #10;
    s1=1; #10;
    s1=0; s0=1; #10;
    s1=1; #10;
    en = 1; s0=0; s1=0; #10;
    s1=1; #10;
    s0=1; s1=0; #10;
    s1=1; #10;
    end
endmodule
```



d) Behavioral SystemVerilog module for a 2-to-1 multiplexer.

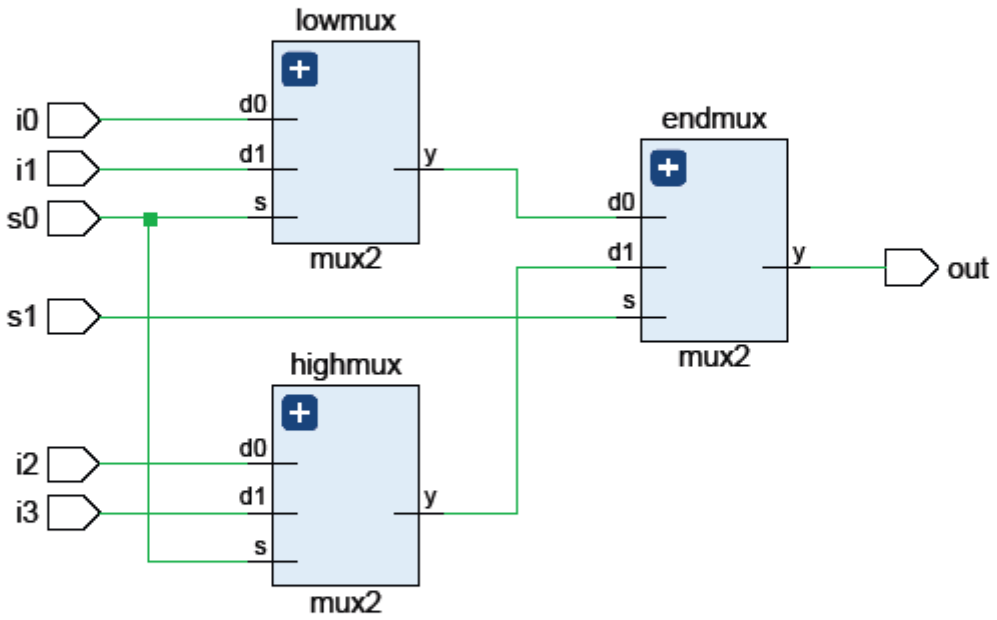
```
module mux2(
    input logic d0, d1,
    input logic s,
    output logic y
    );
    assign y = s ? d1: d0;
endmodule
```



e) Structural SystemVerilog module for a 4-to-1 multiplexer using three 2-to-1 multiplexers. Prepare a testbench for it.

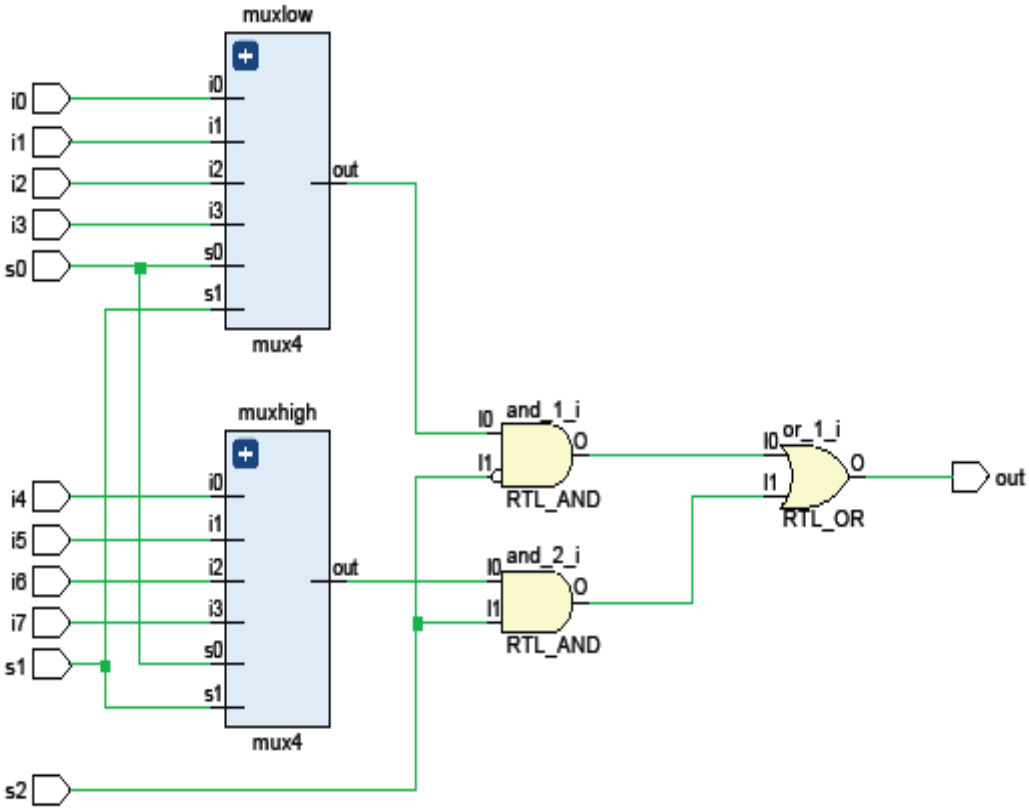
```
module mux4(
   input i0,
   input i1,
   input i2,
   input i3,
   input s0,s1,
   output out
   );
   logic y1,y2;
   mux2 lowmux(i0,i1,s0,y1);
   mux2 highmux(i2,i3,s0,y2);
   mux2 endmux(y1,y2,s1,out);
endmodule

module mux4_tb();
   logic i0,i1,i2,i3;
   logic s0, s1;
   logic out;
   mux4 dut(i0,i1,i2,i3,s0,s1,out);
   initial begin
   s0=0; s1=0; i0=0; i1=0; i2=0; i3=0; #10;
      i3=1; #10;
      i3=0; i2=1; #10;
      i3=1; #10;
      i3=0; i2=0; i1=1; #10;
      i3=1; #10;
      i3=0; i2=1; #10;
      i3=1; #10;
      i0=1; i1=0; i2=0; i3=0; #10;
      i3=1; #1;
      i3=0; i2=1; #10;
      i3=1; #10;
      i1=1; i2=0; #10; i3=0; #10;
      i3=1; #10;
      i3=0; i2=1; #10;
      i3=1; #10;
   s1=1; i0=0; i1=0; i2=0; i3=0; #10;
      i3=1; #10;
      i3=0; i2=1; #10;
      i3=1; #10;
      i3=0; i2=0; i1=1; #10;
      i3=1; #10;
```

```
        i3=0; i2=1; #10;
        i3=1; #10;
        i0=1; i1=0; i2=0; i3=0; #10;
        i3=1; #1;
        i3=0; i2=1; #10;
        i3=1; #10;
        i1=1; i2=0; #10; i3=0; #10;
        i3=1; #10;
        i3=0; i2=1; #10;
        i3=1; #10;
    s1=0; s0=1; i0=0; i1=0; i2=0; i3=0; #10;
            i3=1; #10;
            i3=0; i2=1; #10;
            i3=1; #10;
            i3=0; i2=0; i1=1; #10;
            i3=1; #10;
            i3=0; i2=1; #10;
            i3=1; #10;
            i0=1; i1=0; i2=0; i3=0; #10;
            i3=1; #1;
            i3=0; i2=1; #10;
            i3=1; #10;
            i1=1; i2=0; #10; i3=0; #10;
            i3=1; #10;
            i3=0; i2=1; #10;
            i3=1; #10
    s1=1; i0=0; i1=0; i2=0; i3=0; #10;
                i3=1; #10;
                i3=0; i2=1; #10;
                i3=1; #10;
                i3=0; i2=0; i1=1; #10;
                i3=1; #10;
                i3=0; i2=1; #10;
                i3=1; #10;
                i0=1; i1=0; i2=0; i3=0; #10;
                i3=1; #1;
                i3=0; i2=1; #10;
                i3=1; #10;
                i1=1; i2=0; #10; i3=0; #10;
                i3=1; #10;
                i3=0; i2=1; #10;
                i3=1; #10;
    end
endmodule
```



f) Block diagram and structural System Verilog module of 8-to-1 MUX by using two 4-to-1 MUX modules, two AND gates, an INVERTER, and an OR gate.

```
module mux8(
    input i0,
    input i1,
    input i2,
    input i3,
    input i4,
    input i5,
    input i6,
    input i7,
    input s0,
    input s1,
    input s2,
    output out
    );
    logic r0,r1;
    mux4 muxlow(i0,i1,i2,i3,s0,s1,r0);
    mux4 muxhigh(i4,i5,i6,i7,s0,s1,r1);
    logic a,b,c;
    not not_1(a,s2);
    and and_1(b,r0,a);
    and and_2(c,r1,s2);
    or or_1(out,b,c);
endmodule
```



g) Block diagram and SystemVerilog module for the function F, using only one 8-to-1 multiplexer and an INVERTER (if you wish). The function F is as follows:
F(A,B,C,D)= {1 *if* 8<*ABCD*+*CADB*<220 *otherwise*
where ABCD and CADB are 4-bit unsigned binary numbers. For ABCD, A is the most significant bit and D is the least significant bit. For CADB, C is the most significant bit and B is the least significant bit. E.g. F(0,1,0,1)=1 since ABCD=1001, CADB=0110, and ABCD+CADB =15 that results in 8 < ABCD+CADB < 22.

```
module bool_func(
    input a,
    input b,
    input c,
    input d,
    output y
    );
    logic inv_c;
    not(inv_c,c);
```

mux8 mux_1(c,c,c,c,inv_c,inv_c,inv_c,inv_c,d,b,a,y); //a and d should be reverse for s0 s1 s2

endmodule