DIGITAL DESIGN AND COMPUTER ARCHITECTURE

CS 223

LAB 3

YASEMİN AKIN

22101782

30 OCTOBER 2022

```
//b) Behavioral SystemVerilog module for a 1-to-2
//decoder (including enable signal).
`timescale 1ns/1ps
module decoder1to2 (input logic d, E, output logic y0, y1);

    always @(d, E)
        begin
           y0<=d&E;
           y1<=~d&E;
        end

endmodule

//And a testbench for it.
`timescale 1ns/1ps
module testDecoder1to2();
    logic d, E;
    logic y0,y1;

    decoder1to2 uut(d, E, y0, y1);
    initial begin
        E=0; d=0; #10;
        d=1; #10;
        E=1; d=0; #10;
        d=1; #10;
    end

endmodule
```

```
//c) Structural SystemVerilog module for a 2-to-4 decoder (including
//enable signal) using three 1-to-2 decoders.
`timescale 1ns/1ps
module decoder2to4 (input logic [1:0] d, input logic E, output logic [3:0]
y);

    logic [1:0] second;

    decoder1to2 decoder0(d[0], E, second[0], second[1]);
    decoder1to2 decoder1(d[1], second[0], y[0], y[1]);
    decoder1to2 decoder2(d[1], second[1], y[2], y[3]);

endmodule

//And a testbench for it.
`timescale 1ns/1ps
module testDecoder2to4();

    logic [1:0] d;
    logic E;
    logic [3:0] y;

    decoder2to4 uut(d, E, y);
    initial begin
        E=0; d[0]=0; d[1]=0; #10;
        d[0]=1; d[1]=0; #10;
        d[0]=0; d[1]=1; #10;
        d[0]=1; d[1]=1; #10;
        E=1; d[0]=0; d[1]=0; #10;
        d[0]=1; d[1]=0; #10;
        d[0]=0; d[1]=1; #10;
        d[0]=1; d[1]=1; #10;
    end

endmodule
```

```
//d) Behavioral SystemVerilog module for a 2-to-1 multiplexer.
`timescale 1ns/1ps
module mux2 (input logic d0, d1, s, output logic y);

    always@(d0, d1, s)
        begin
            if(s==0) y<=d0;
            else y<=d1;
        end

endmodule

//e) Structural SystemVerilog module for a 4-to-1 multiplexer using three
//2-to-1 multiplexers.
`timescale 1ns/1ps
module mux4 (input d0,d1,d2,d3,s0,s1,output logic y );

    logic second0, second1;
    mux2 mux2_1(d0, d1, s0, second0);
    mux2 mux2_2(d2, d3, s0, second1);
    mux2 mux2_3(second0, second1, s1, y);

endmodule

//And a testbench for it.
`timescale 1ns/1ps
module testMux4();

    logic d0, d1, d2,d3,s0,s1;
    logic y;

    mux4 uut(d0, d1, d2,d3,s0,s1,y);
    initial begin
        d0=0; d1=0;d2=0;d3=0;
        s0=0; s1=0; d0=1; #10; d0=0;
        s0=1; d1=1; #10; d1=0;
        s1=1; s0=0; d2=1; #10; d2=0;
        s0=1; d3=1; #10;
    end

endmodule
```

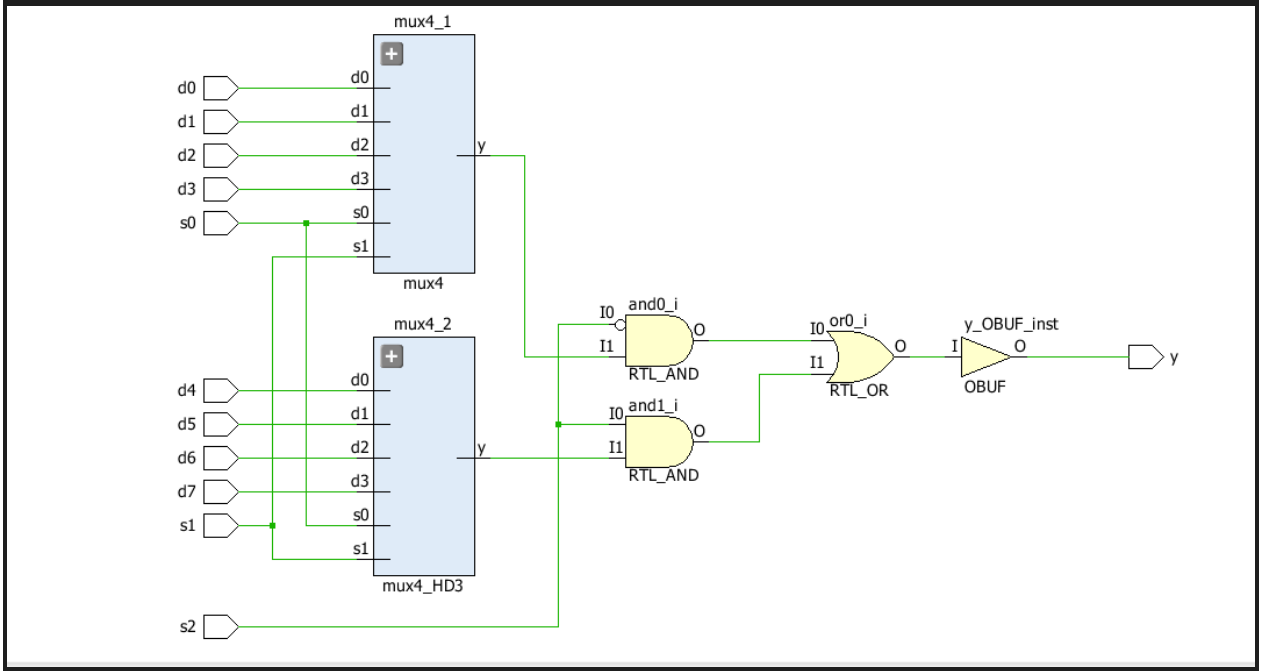
```
//f) Structural System Verilog module of 8-to-1 MUX by using two
//4-to-1 MUX modules, two AND gates, an INVERTER, and an OR gate.
`timescale 1ns/1ps
module mux8 (input logic d0,d1,d2,d3,d4,d5,d6,d7,s0,s1,s2,output logic y);

    logic first, second, w1, w2, s2inv;

    mux4 mux4_1( d0,d1,d2,d3,s0,s1, first);
    mux4 mux4_2(d4,d5,d6,d7,s0,s1 , second);
    not inv0(s2inv,s2);
    and and0(w1, s2inv, first);
    and and1(w2, s2, second);
    or or0(y, w1, w2);

endmodule

//Block Diagram of Part f.
```



```systemverilog
//g) SystemVerilog module for the function F, using only one 8-to1
//multiplexer and an INVERTER (if you wish).
module labPartg(input logic A, B, C, D, output logic y);

    logic k;

    not not0(k,C);
    mux8 mux8Test(C,C,C,C,k,k,k,k,B,D,A,y);

endmodule

//Block Diagram of Part g.
```