Digital Design I

CS 223 - 01

Lab 03

Uğur Erdem Seyfi

21801744

04.11.2019

Trainer Pack No:

**b) Behavioral SystemVerilog module for 2-to-4 decoder and a testbench for it.**

// 2 to 4 decoder

module decoder2\_4(

input logic a0,

input logic a1,

output logic y0, y1, y2, y3

);

assign y0 = ~a1 & ~a0; // a1 = 0, a0 = 0

assign y1 = ~a1 & a0; // a1 = 0, a0 = 1

assign y2 = a1 & ~a0; // a1 = 1, a0 = 0

assign y3 = a1 & a0; // a1 = 1, a0 = 1

endmodule

// test bench for 2 to 4 decoder

module test\_bench\_decoder\_2\_4();

logic a0, a1;

logic y0, y1, y2, y3;

decoder2\_4 D1(a0, a1, y0, y1, y2, y3);

// apply inputs one at a time

initial begin

a1 = 0; #10;

a0 = 0; #10;

a0 = 1; #10;

a1 = 1; #10;

a0 = 0; #10;

a0 = 1; #10;

end

endmodule

**c) Behavioral SystemVerilog module for 2-to-1 multiplexer.**

module mux2\_1(

input logic d0, d1, s0,

output logic y

);

assign y = s0 ? d1 : d0;

endmodule;

**d) Structural SystemVerilog module for 4-to-1 multiplexer by using three 2-to-1 multiplexer.**

module mux4\_1(

input logic d0, d1, d2, d3, s0, s1,

output logic y

);

logic m0, m1;

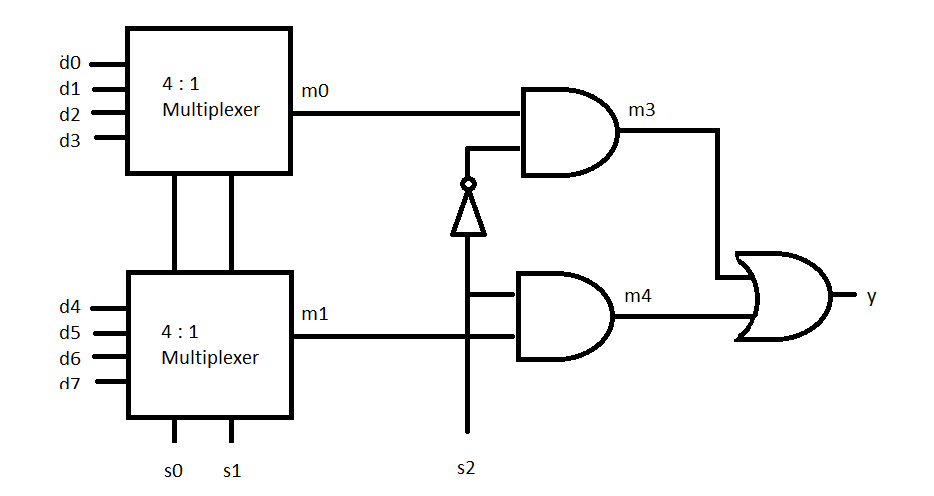
mux2\_1 M0(d0, d1, s0, m0);

mux2\_1 M1(d2, d3, s0, m1);

mux2\_1 M2(m0, m1, s1, y);

endmodule;

**e) Schematic (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. Prepare a test bench for it.**



module mux8\_1(

input logic d0, d1, d2, d3, d4, d5, d6, d7,

s0, s1, s2,

output logic y

);

logic m0, m1, i, m2, m3;

mux4\_1 M0(d0, d1, d2, d3, s0, s1, m0);

mux4\_1 M1(d4, d5, d6, d7, s0, s1, m1);

inv I1(s2, i);

and2 A1(m0, i, m3);

and2 A2(m1, s2, m4);

or2 O1(m3, m4, y);

endmodule

**// testbench**

module testbench\_mux\_8\_1();

logic d0,d1,d2,d3,d4,d5,d6,d7,s0, s1, s2;

logic y;

mux8\_1 test(d0, d1, d2, d3 ,d4, d5, d6, d7,

s0, s1, s2, y);

initial begin

s2 = 0;

s1 = 0;

// case for 000

s0 = 0;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

// case for 001

s0 = 1;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

s1 = 1;

// case for 010

s0 = 0;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

// case for 011

s0 = 1;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

s2 = 1;

s1 = 0;

// case for 100

s0 = 0;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

// case for 101

s0 = 1;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

s1 = 1;

// case for 110

s0 = 0;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

// case for 111

s0 = 1;

d0 = 1; d1 = 0; d2 = 0; d3 = 0; d4 = 0; d5 = 0; d6 = 0; d7 = 0; #10;

d0 = 0; d1 = 1; #10;

d1 = 0; d2 = 1; #10;

d2 = 0; d3 = 1; #10;

d3 = 0; d4 = 1; #10;

d4 = 0; d5 = 1; #10;

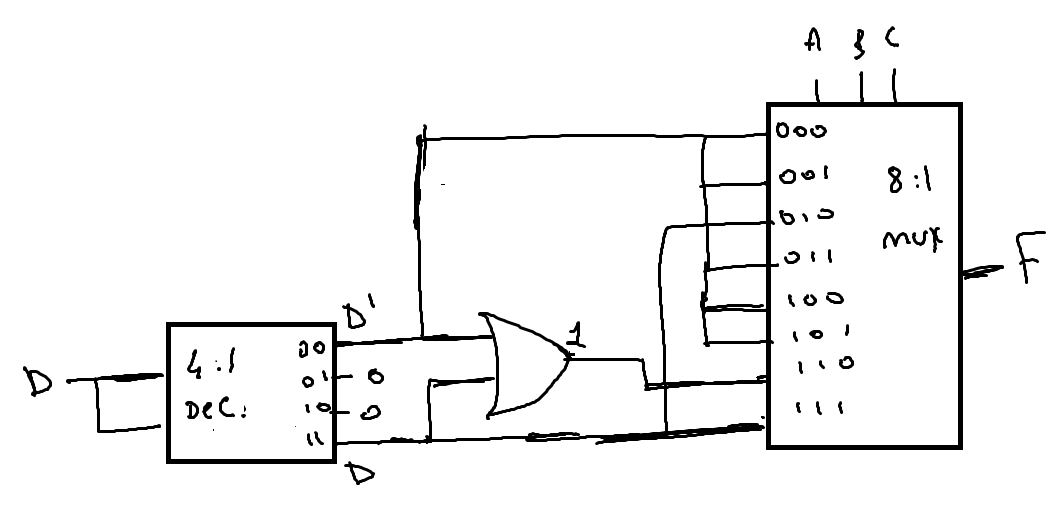
d5 = 0; d6 = 1; #10;

d6 = 0; d7 = 1; # 10;

end

endmodule;

**f) Schematic (block diagram) and SystemVerilog module for F(A,B,C,D)=∑(0,2,5,6,8,10,12,13,15) function, using one (not two) 8-to-1 multiplexer and a 2-to-4 decoder.**



// Structural SystemVerilog module for the function F

module F(

input logic a, b, c, d,

output logic y

);

logic D, D\_not, dummyWire1, dummyWire2;

decoder2\_4(d, d, D\_not, dummyWire1, dummyWire2, D);

logic I;

or2(D, D\_not, I);

mux8\_1(a, b, c, D\_not, D\_not, D, D\_not, D\_not, D\_not, I, D, y);

endmodule;