CS 223 Digital Design

Section 1

Lab 3

Ahmet Faruk Ulutaş

21803717

24.10.2020

B) Behavioral SystemVerilog module for 2-to-4 decoder and a testbench for it

Behavioral SystemVerilog

```
module behavioral_two_to_four_decoder( input logic w0, w1, output logic d0, d1, d2, d3);
   logic inv0, inv1;

   assign inv0 = ~w0;
   assign inv1 = ~w1;

   assign d0 = inv0 & inv1;
   assign d1 = w0 & inv1;
   assign d2 = w1 & inv0;
   assign d3 = w1 & w0;
endmodule
```

Testbench

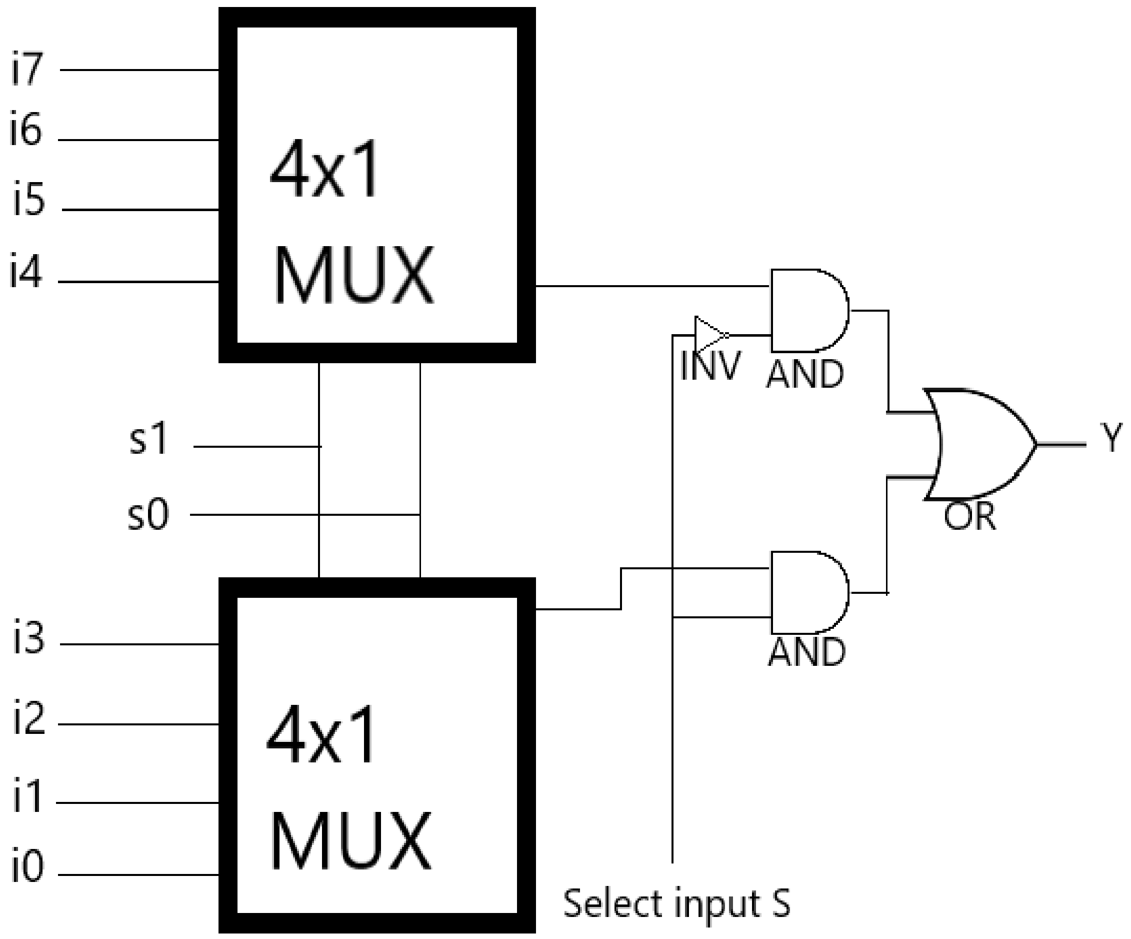
```
module testbench_of_two_to_four_decoder();
   logic w0, w1, d0, d1, d2, d3;
   behavioral_two_to_four_decoder dut( w0, w1, d0, d1, d2, d3);

   initial begin
      w0 = 0; w1 = 0; #10;
      w1 = 1; #10;
      w0 = 1; w1 = 0; #10;
      w1 = 1; #10;
   end
endmodule
```

C) Behavioral SystemVerilog module for 4-to-1 multiplexer.

```
module behavioral_four_to_one_mux( input logic A, B, C, D, a, b, output logic Q);

   logic in2_nand0, in2_nand1, in3_and0, in3_and1, in3_and2, in3_and3, in4_or0;



   assign in2_nand0 = ~(b & b);

   assign in2_nand1 = ~(a & a);



   assign in3_and0 = (A & in2_nand0 & in2_nand1);

   assign in3_and1 = (B & in2_nand0 & a);

   assign in3_and2 = (C & in2_nand1 & b);

   assign in3_and3 = (D & b & a);

   assign in4_or0 = in3_and0 | in3_and1 | in3_and2 | in3_and3;



   assign Q = in4_or0;

endmodule
```

D) 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.

Schematic (block diagram)



Structural System Verilog module

```
module structural_eight_to_one_mux( input logic i0, i1, i2, i3, i4, i5, i6, i7, s0, s1, S, output
Y);

    logic mux4to1_0, mux4to1_1, and0, and1, inv0;

    mux4to1 mux4to1__0( i7, i6, i5, i4, s1, s0, mux4to1_0);

    mux4to1 mux4to1__1( i3, i2, i1, i0, s1, s0, mux4to1_1);

    inv0 inv_0( S, inv0);

    and0 and_0( mux4to1_0, inv0, and0);

    and0 and_1( mux4to1_1, S, and1);

    or0 or_0( and0, and1, Y);

endmodule

module mux4to1( input logic a, b, c, d, s1, s0, output logic out);

    assign out = (~s1 & ~s0 & d) | (~s1 & s0 & c) | (s1 & ~s0 & b) | (s1 & s0 & a);

endmodule

module inv0( input logic a, output logic out);
    assign out = ~a;

endmodule
```

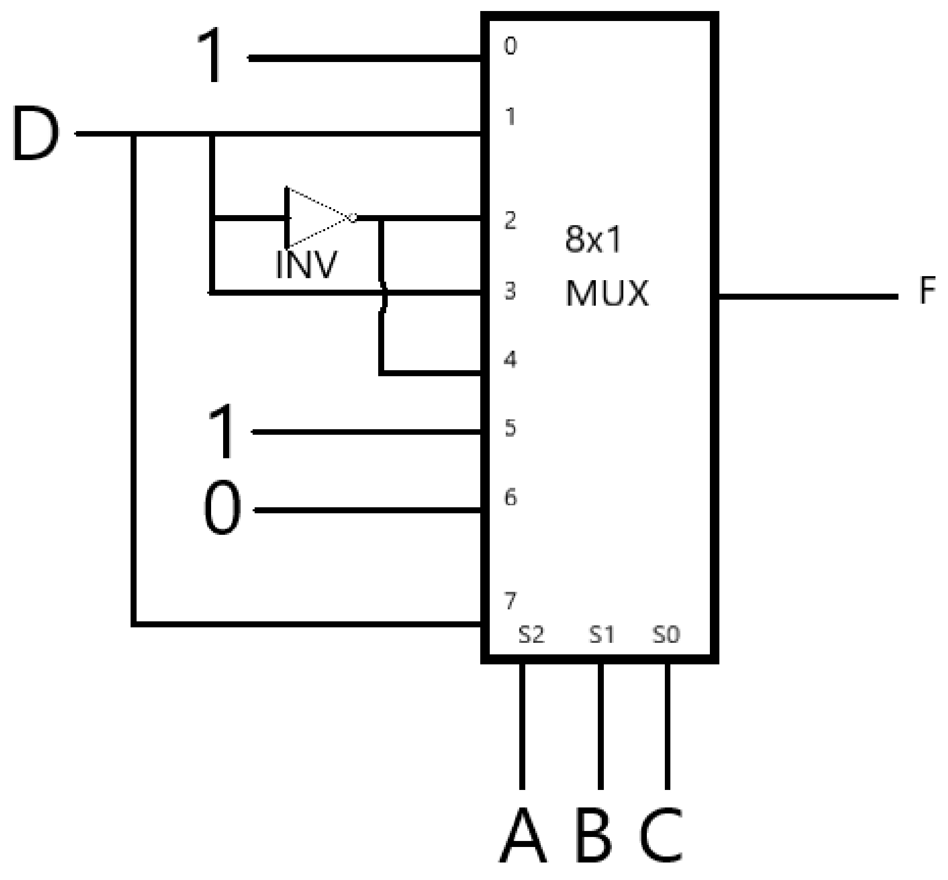
```
module and0( input logic a, b, output logic out);




   assign out = a & b;

endmodule

module or0( input logic a, b, output logic out);
   assign out = a | b;

endmodule
```

Testbench

```
module testbench_structural_eight_to_one_mux();
   logic i0, i1, i2, i3, i4, i5, i6, i7, s0, s1, S, Y;
   structural_eight_to_one_mux dut( i0, i1, i2, i3, i4, i5, i6, i7, s0, s1, S, Y);
   initial begin
      i0 = 0; i1 = 1; i2 = 0; i3 = 1; i4 = 0; i5 = 1; i6 = 0; i7 = 1;
      S = 0; s1 = 0; s0 = 0; #10;
      s0 = 1; #10;
      s1 = 1; s0 = 0; #10;
      s0 = 1; #10;
      S = 1; s1 = 0; s0 = 0; #10;
      s0 = 1; #10;
      s1 = 1; s0 = 0; #10;
      s0 = 1; #10;
   end
endmodule
```

E) Schematic (block diagram) and SystemVerilog module for F(A,B,C,D)=∑(0,1,3,4,7,8,10,11,15) function, using one (not two) 8-to-1 multiplexer and an inverter.

Schematic (block diagram)



SystemVerilog module

```
module in4f( input logic S2, S1, S0, D, output logic F);

logic inv0;
inv1 inv1( D, inv0);

eight_to_one_mux eight_to_one_mux0( 1, D, inv0, D, inv0, 1, 0, D, S2, S1, S0, F);

endmodule

module inv1( input logic a, output logic out);
    assign out = ~a;

endmodule

module eight_to_one_mux( input logic in0, in1, in2, in3, in4, in5, in6, in7, s2, s1, s0, output
logic out);
    assign out = (in0 & ~s2 & ~s1 & ~s0) | (in1 & ~s2 & ~s1 & s0) | (in2 & ~s2 & s1 & ~s0) |
(in3 & ~s2 & s1 & s0) | (in4 & s2 & ~s1 & ~s0) | (in5 & s2 & ~s1 & s0) | (in6 & s2 & s1 &
~s0) | (in7 & s2 & s1 & s0);

endmodule
```