CS 223 Digital Design

Section 5

Lab 3

Name: Esad Ismail Tok

ID: 21801679

Date: 17.10.2020

Behavioral SystemVerilog Module for 2:4 Decoder and Test Bench for It

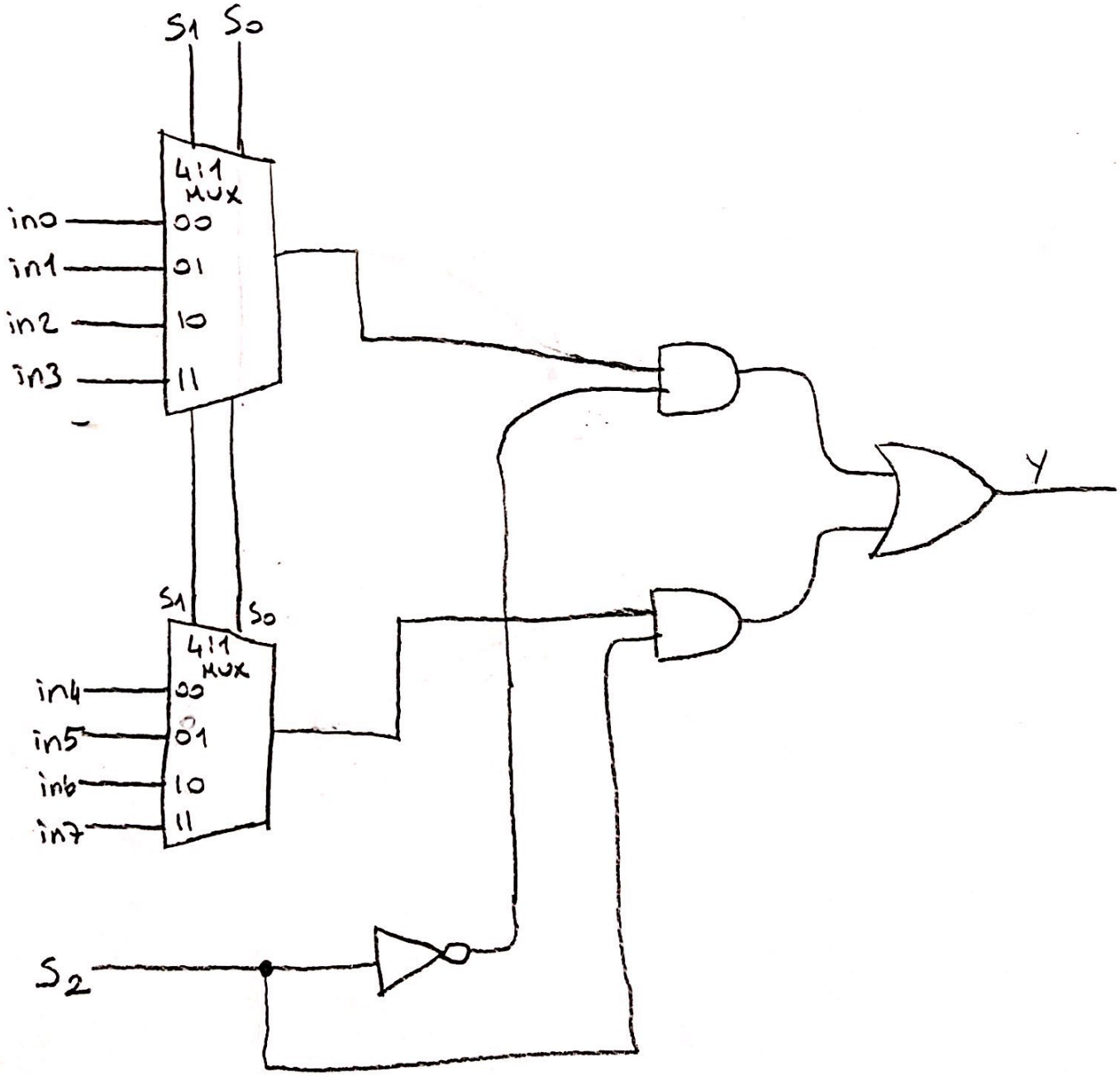
```
`timescale 1ns / 1ps

// 2 :4 decoder implementation with behavioral style
module decoder2to4(input logic in1, in0, output logic y3, y2, y1, y0);

    assign y3 = in1 & in0;
    assign y2 = in1 & ~in0;
    assign y1 = ~in1 & in0;
    assign y0 = ~in1 & ~in0;

endmodule

// TestBench for the 2:4 decoder
module tbDecoder2to4();

logic in1, in0;
logic y3, y2, y1, y0;

decoder2to4 sampledecoder(in1, in0, y3, y2, y1, y0);

initial begin
    $monitor(in1, in0);
    for (int i = 0; i < 4; i = i+1) begin
        {in1, in0} = i; #10;
        $display(y3, y2, y1, y0);
    end
end
endmodule
```

Behavioral SystemVerilog Module for 4:1 Multiplexer

```
`timescale 1ns / 1ps

// 4:1 multiplexer implementation in behavioral style
module mux4to1(input logic [3:0] a, b, c, d, input logic [1:0] slct, output logic [3:0] out);

    // Nested conditional assignment will aloow the multiplexer to select the outmut among inputs
    assign out = slct[1] ? (slct[0] ? d : c) : (slct[0] ? b : a);

endmodule

// TestBench for the 4: multiplexer
module tbMux4to1();

    // Declaring the internal signals;
    logic [3:0] a, b, c, d;
    logic [1:0] slct;
    logic [3:0] out;

    // Instantiating the unit under test
    mux4to1 sampleMux4to1(a, b, c, d, slct, out);

    initial begin

        // monitoring the signals whenever one of them changes
        $monitor("a = %b, b = %b, c = %b, d = %b, select input = %b, output = %b", a, b, c, d, slct, out);

        // Initially the inputs are random and select input will change gradually
        a = $random;
        b = $random;
```

```
        c = $random;
        d = $random;

        // Change dthe select input in order to observe the change at output
        for (int i = 0; i < 4; i=i+1) begin
            slct = i; #10;
        end

    end

endmodule
```

Block Diagram of 8:1 MUX by using
~ Two 4:1 MUXs, Two AND gates, One Inverter, and one OR gate ~



Structural SystemVerilog Module for 8:1 Multiplexer and Test Bench for It

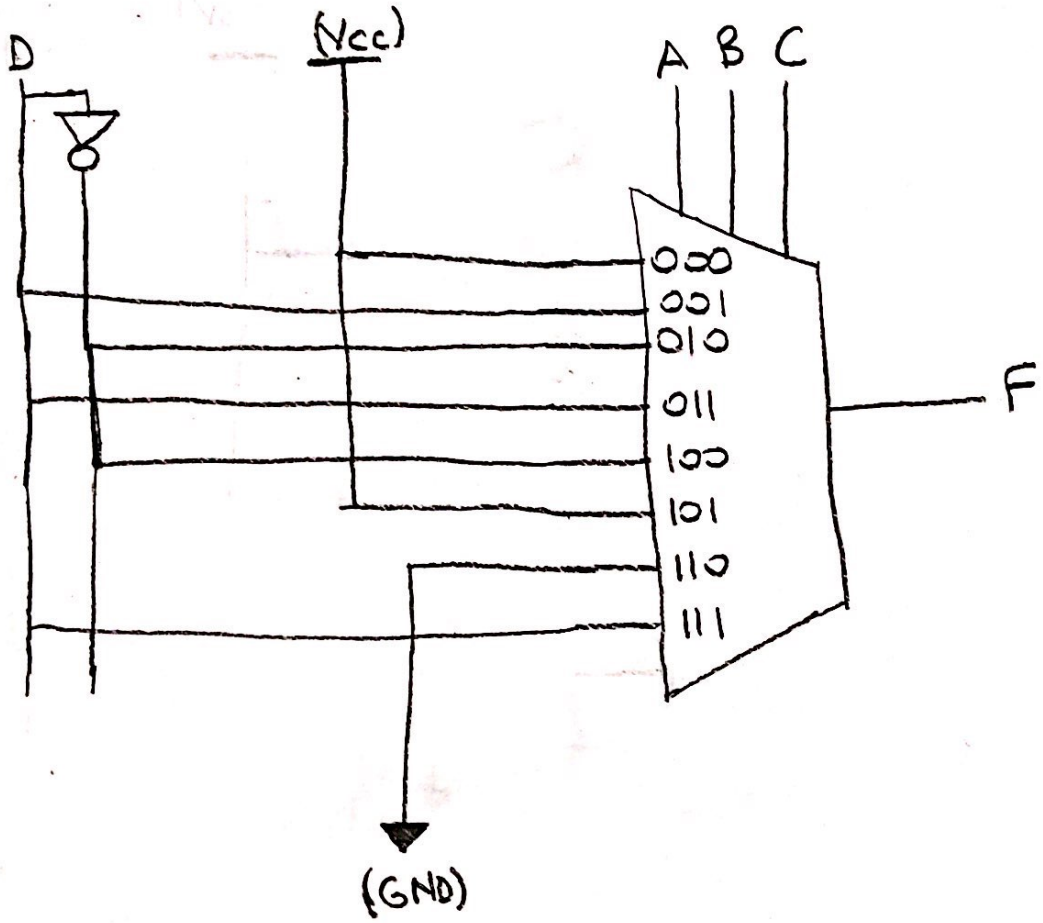
```
`timescale 1ns / 1ps
`include "mux4to1.sv" // To use 4:1 Mux

// Top Level Structural 8:1 mux using tw0 4:1 mux, two AND gates, one inverter, and one OR gate
module mux8to1(input logic in0, in1, in2, in3, in4, in5, in6, in7, input logic [2:0] s, output logic y);

    // Internal signals
    logic n1, n2, n3;
    logic n4, n5;

    // First level internal signals
    mux4to1 firstMux4(in0, in1, in2, in3, s[1:0], n1);
    mux4to1 secondMux4(in4, in5, in6, in7, s[1:0], n2);
    inv inverter(s[2], n3);

    // Second level internal signals
    and2 firstAnd(n1, n3, n4);
    and2 secondAnd(n2, s[2], n5);

    // Third level output
    or2 firstOr(n4, n5, y);

endmodule

// Low level AND module implementation
module and2(input logic a, b, output logic y);
    assign y = a & b;
endmodule
```

```
// Low level OR module implementation
module or2(input logic a, b, output logic y);
    assign y = a | b;
endmodule

// Low level Inverter module implementation
module inv(input logic a, output logic y);
    assign y = ~a;
endmodule



// Test Bench for the Top Level module
module tbMux8to1();

    // Internal signals
    logic in0, in1, in2, in3, in4, in5, in6, in7;
    logic [2:0] s;
    logic y;

    // Instantating unit under test
    mux8to1 uut(in0, in1, in2, in3, in4, in5, in6, in7, s, y);

    initial begin

        $monitor("in0 = %b, in1 = %b, in2 = %b, in3 = %b, in4 = %b, in5 = %b, in6 = %b, in7 = %b, select input = %b, output = %b", in0, in1, in2, in3, in4, in5, in6, in7, s, y);
        in0 = $random;
        in1 = $random;
        in2 = $random;
        in3 = $random;
```

```
        in4 = $random;
        in5 = $random;
        in6 = $random;
        in7 = $random;

        for(int i = 0; i < 8; i=i+1) begin
            s = i; #10;
        end

    end

endmodule
```

Block Diagram for $F(A,B,C,D) = \sum(0,1,3,4,7,8,10,11,15)$

Using one 8:1 Multiplexer and an Inverter



SystemVerilog Module for F(A,B,C,D)=∑(0,1,3,4,7,8,10,11,15) Function Using One 8:1 Multiplexer and an Inverter

```
`timescale 1ns / 1ps
`include "mux8to1.sv"

// 8:1 Mux implementation for a function
module functionMux8to1(input logic [3:0] s, output logic y);

    logic n1; // n1 is ~s[0]
    inv inverter(s[0], n1);

    // Since it is a function the inputs are known
    logic in0 = 1;
    logic in1 = s[0];
    logic in2 = n1;
    logic in3 = s[0];
    logic in4 = n1;
    logic in5 = 1;
    logic in6 = 0;
    logic in7 = s[0];

    mux8to1 multiplexer(in0, in1, in2, in3, in4, in5, in6, in7, s[3:1], y);

endmodule
```