DIGITAL DESIGN 3rd LAB

PRELIMINARY REPORT

Course Code: CS 223

Course Name: Digital Design

Section: 1

Name: Arda

Surname: İynem

Student ID: 22002717

Date: 13.03.2022

1-TO-2 DECODER

BEHAVIORAL MODULE TESTBENCH

(b) Behavioral SystemVerilog Module for 1-to-2 Decoder

```
module one_to_two_dec(input e, a0, output y[1:0]);
    assign y[0] = e & ~a0;
    assign y[1] = e & a0;
endmodule
```

(b) Testbench for 1-to-2 Decoder

```
module tb_lab();
    logic e, a0, y[1:0];

    one_to_two_dec dut(e, a0, y[1:0]);
    initial begin
        e = 0; a0 = 0; #10;
        a0 = 1; #10;
        e = 1; a0 = 0; #10;
        a0 = 1; #10;
    end
endmodule
```

2-TO-4 DECODER

STRUCTURAL MODULE TESTBENCH

(c) Structural SystemVerilog Module for 2-to-4 Decoder (Using three 1-to-2 Decoders)

```
module two_to_four_dec(input e, a[1:0], output y[3:0]);
    logic m[3:0];

    one_to_two_dec dec1(e, a[1], m[1:0]);
    one_to_two_dec dec2(m[0], a[0], y[1:0]);
    one_to_two_dec dec3(m[1], a[0], y[3:2]);
endmodule
```

(c) Testbench for 2-to-4 Decoder

```
module tb_lab();
    logic e, a[1:0], y[3:0];

    two_to_four_dec dut(e, a[1:0], y[3:0]);
    initial begin
        e = 0; a[1] = 0; a[0] = 0; #10;
        a[0] = 1; #10;
        a[1] = 1; a[0] = 0; #10;
        a[0] = 1; #10;
        e = 1; a[1] = 0; a[0] = 0; #10;
        a[0] = 1; #10;
        a[1] = 1; a[0] = 0; #10;
        a[0] = 1; #10;
    end
endmodule
```

2-TO-1 MULTIPLEXER

BEHAVIORAL MODULE TESTBENCH

(d) Behavioral SystemVerilog Module for 2-to-1 Multiplexer

```
module two_to_one_mux(input d[1:0], s, output y);
    assign y = s ? d[1] : d[0];
endmodule
```

(d) Testbench for 2-to-1 Multiplexer

```
module tb_lab();
    logic d[1:0], s, y;

    two_to_one_mux dut(d[1:0], s, y);
    initial begin
        s = 0; d[1] = 0; d[0] = 0; #10;
        d[0] = 1; #10;
        d[1] = 1; d[0] = 0; #10;
        d[0] = 1; #10;
        s = 1; d[1] = 0; d[0] = 0; #10;
        d[0] = 1; #10;
        d[1] = 1; d[0] = 0; #10;
        d[0] = 1; #10;
    end
endmodule
```

4-TO-1 MULTIPLEXER

STRUCTURAL MODULE TESTBENCH

(e) Structural SystemVerilog Module for 4-to-1 Multiplexer (Using three 2-to-1 Multiplexers)

```
module four_to_one_mux(input d[3:0], s[1:0],
output y);
    logic m[1:0];

    two_to_one_mux mux1(d[1:0], s[0], m[0]);
    two_to_one_mux mux2(d[3:2], s[0], m[1]);
    two_to_one_mux mux3(m[1:0], s[1], y);
endmodule
```

(e) Testbench for 4-to-1 Multiplexer

```
module tb_lab();
    logic d[3:0], s[1:0], y;

    four_to_one_mux dut(d[3:0], s[1:0], y);
    initial begin
        s[1] = 0; s[0] = 0;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        s[1] = 0; s[0] = 1;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 1; #10;
```

(e) Testbench for 4-to-1 Multiplexer (Continues)

```
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        s[1] = 1; s[0] = 0;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        s[1] = 1; s[0] = 1;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 0; d[2] = 1; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 0; d[1] = 1; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 0; d[0] = 1; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 0; #10;
        d[3] = 1; d[2] = 1; d[1] = 1; d[0] = 1; #10;
    end
endmodule
```

8-TO-1 MULTIPLEXER

STRUCTURAL MODULE BLOCK DIAGRAM

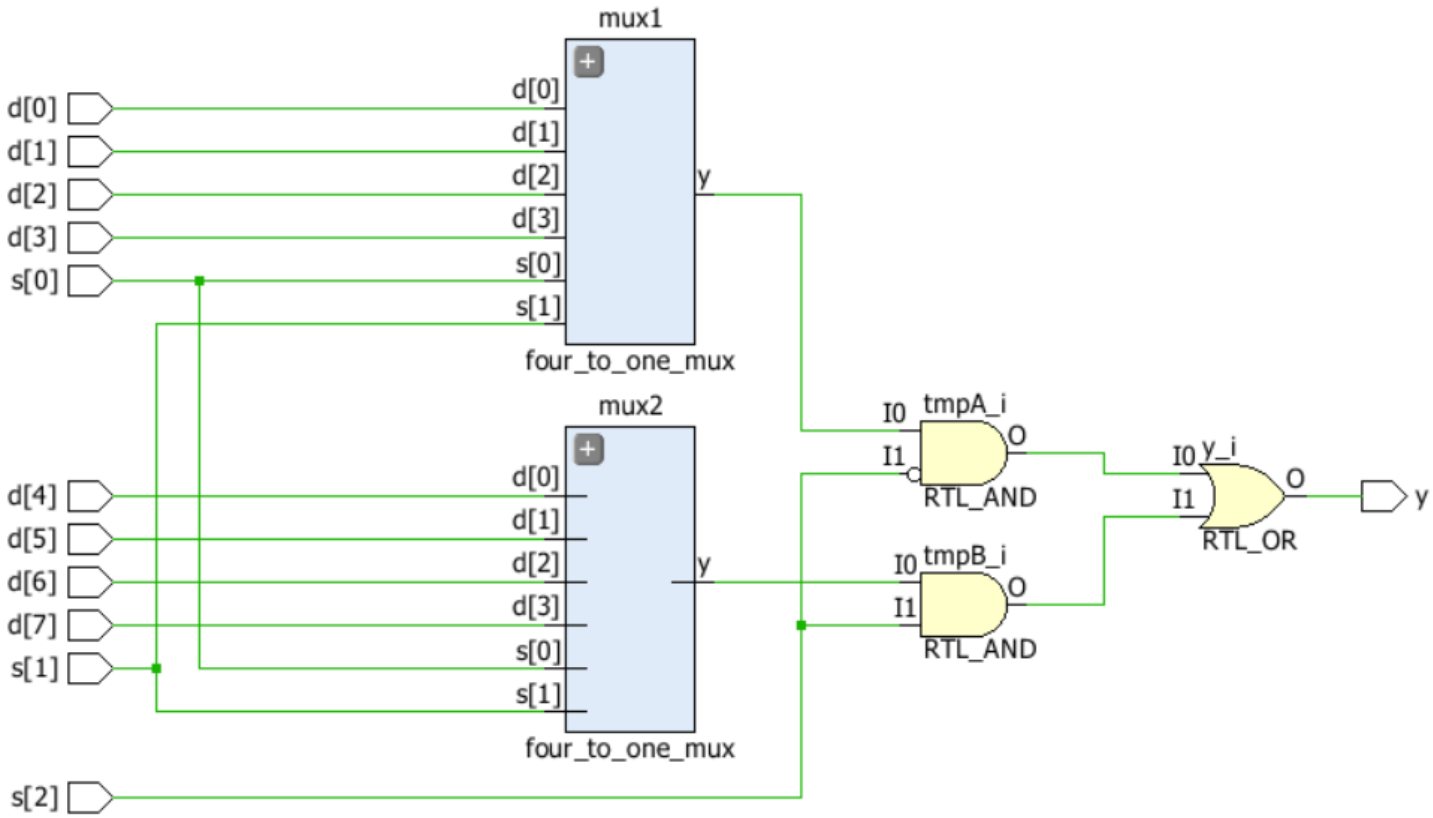
(f) Structural SystemVerilog Module for 8-to-1 Multiplexer (Using two 4-to-1 Multiplexers, two AND, an OR and a NOT gate)

```
module eight_to_one_mux(input d[7:0], s[2:0], output y);
    logic m[1:0];

    four_to_one_mux mux1(d[3:0], s[1:0], m[0]);
    four_to_one_mux mux2(d[7:4], s[1:0], m[1]);

    logic tmpA, tmpB, tmpC;
    not(tmpC, s[2]);
    and(tmpA, m[0], tmpC);
    and(tmpB, m[1], s[2]);
    or(y, tmpA, tmpB);
endmodule
```

(f) Block Diagram of 8-to-1 Multiplexer



FUNCTION F

STRUCTURAL MODULE BLOCK DIAGRAM

(g) Structural SystemVerilog Module for Function F (Using an 8-to-1 and a NOT gate)

```
module function_f(input a, b, c, d, output y);
    logic m[7:0];

    assign m = {d, d, ~d, ~d, 0, d, d, 0};

    eight_to_one_mux mux1(m, {a, b, c}, y);
endmodule
```

(g) Block Diagram for Function F