Digital Design

CS223

Lab3

Erkin Aydın

ID: 22002956

31/10/2022

Part b) of Preliminary Report:

1-to-2-Decoder module:

```
module one_to_two_dc(
input logic i, e,
output logic y1, y0);
	assign y0 = (~i) && e;
	assign y1 = i && e;
endmodule
```

1-to-2-Decoder testbench:

```
module one_to_two_dc_tb();
	logic i, e, y1, y0;
	one_to_two_dc dc1(i, e, y1, y0);
	initial begin
		assign e = 0;
		assign i = 0; #30;
		assign i = 1; #30;
		assign e = 1;
		assign i = 0; #30;
		assign i = 1; #30;
	end
endmodule
```

Part c) of Preliminary Report:

2-to-4-Decoder module:

```
module two_to_four_dc(
input logic i1, i0, e,
output logic y3, y2, y1, y0);
        logic d1, d0;
        one_to_two_dc dc1(i1, e, d1, d0);
        one_to_two_dc dc2(i0, d0, y1, y0);
        one_to_two_dc dc3(i0, d1, y3, y2);
endmodule
```

2-to-4-Decoder testbench:

```
module two_to_four_dc_tb();
        logic i1, i0, e;
        logic y3, y2, y1, y0;
        two_to_four_dc dc(i1, i0, e, y3, y2, y1, y0);
        initial begin
                assign e = 0;
                assign i1 = 0;
                assign i0 = 0; #90;
                assign i0 = 1; #90;
                assign i1 = 1;
                assign i0 = 0; #90;
                assign i0 = 1; #90;
                assign e = 1;
                assign i1 = 0;
```

```
        assign i0 = 0; #90;
        assign i0 = 1; #90;
        assign i1 = 1;
        assign i0 = 0; #90;
        assign i0 = 1; #90;
    end
endmodule
```

Part d) of Preliminary Report:

2-to-1-multiplexer module:

```
module two_to_one_mux(i1, i0, s, d);
        input i1, i0, s;
        output d;
        assign d= ~s&i0 | s&i1;
endmodule
```

Part e) of Preliminary Report:

4-to-1-multiplexer module:

```
module four_to_one_mux(i3, i2, i1, i0, s1, s0, z);
	input i3, i2, i1, i0, s1, s0;
	output z;
	wire d1, d0;
	two_to_one_mux m1(i3, i2, s0, d1); // instance1
	two_to_one_mux m2(i1, i0, s0, d0); // instance2
	two_to_one_mux m3(d1, d0, s1, z); // instance3
endmodule
```

4-to-1-multiplexer testbench:

```
module four_to_one_mux_tb();
	reg[3:0] i;
	reg[1:0] s;
	logic d;
	four_to_one_mux m4(i[3], i[2], i[1], i[0], s[1], s[0], d);
	initial begin
		//s0 = s1 = 0;
		//i2 = 0; i3 = 0;
		assign s = 2'b00;
		assign i = 4'b0000; #10;
		assign i = 4'b0001; #10;
		assign i = 4'b0010; #10;
		assign i = 4'b0011; #10;
		assign i = 4'b0100; #10;
```

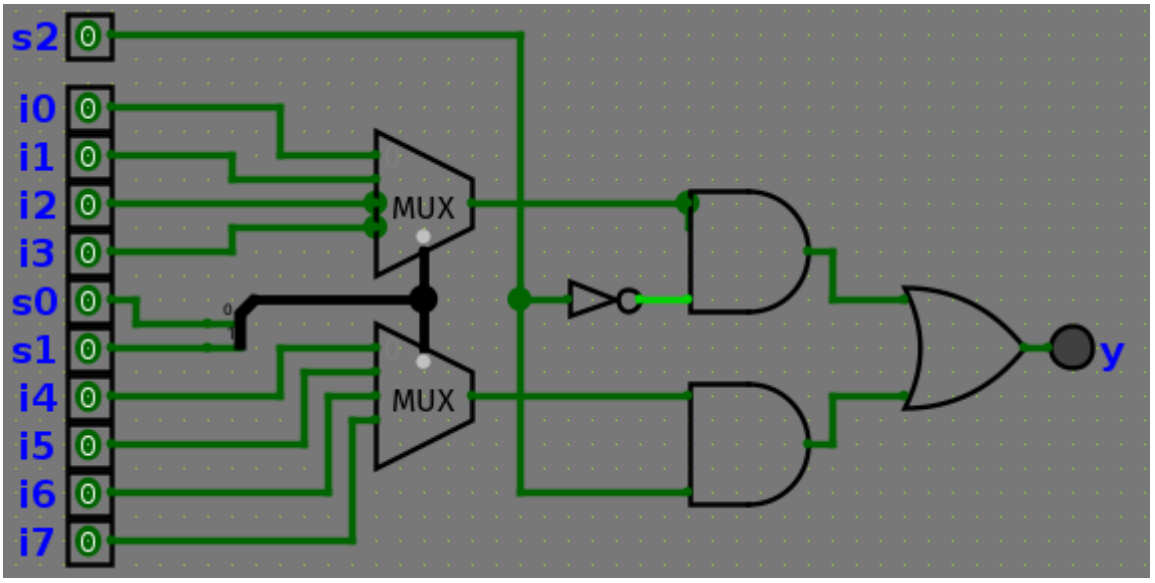
```
assign i = 4'b0101; #10;
assign i = 4'b0110; #10;
assign i = 4'b0111; #10;
assign i = 4'b1000; #10
assign i = 4'b1001; #10;
assign i = 4'b1010; #10;
assign i = 4'b1011; #10;
assign i = 4'b1100; #10;
assign i = 4'b1101; #10;
assign i = 4'b1110; #10;
assign i = 4'b1111; #10;
assign s = 2'b01;
assign i = 4'b0000; #10;
assign i = 4'b0001; #10;
assign i = 4'b0010; #10;
assign i = 4'b0011; #10;
assign i = 4'b0100; #10;
assign i = 4'b0101; #10;
assign i = 4'b0110; #10;
assign i = 4'b0111; #10;
assign i = 4'b1000; #10
assign i = 4'b1001; #10;
assign i = 4'b1010; #10;
assign i = 4'b1011; #10;
assign i = 4'b1100; #10;
assign i = 4'b1101; #10;
assign i = 4'b1110; #10;
```

```
assign i = 4'b1111; #10;
assign s = 2'b10;
assign i = 4'b0000; #10;
assign i = 4'b0001; #10;
assign i = 4'b0010; #10;
assign i = 4'b0011; #10;
assign i = 4'b0100; #10;
assign i = 4'b0101; #10;
assign i = 4'b0110; #10;
assign i = 4'b0111; #10;
assign i = 4'b1000; #10
assign i = 4'b1001; #10;
assign i = 4'b1010; #10;
assign i = 4'b1011; #10;
assign i = 4'b1100; #10;
assign i = 4'b1101; #10;
assign i = 4'b1110; #10;
assign i = 4'b1111; #10;
assign s = 2'b11;
assign i = 4'b0000; #10;
assign i = 4'b0001; #10;
assign i = 4'b0010; #10;
assign i = 4'b0011; #10;
assign i = 4'b0100; #10;
assign i = 4'b0101; #10;
assign i = 4'b0110; #10;
assign i = 4'b0111; #10;
```

```
                assign i = 4'b1000; #10
                assign i = 4'b1001; #10;
                assign i = 4'b1010; #10;
                assign i = 4'b1011; #10;
                assign i = 4'b1100; #10;
                assign i = 4'b1101; #10;
                assign i = 4'b1110; #10;
                assign i = 4'b1111; #10;
                assign s= 2'b00;
                assign i = 2'b0000;
        end
endmodule
```

Part f) of Preliminary Report:

8-to-1-multiplexer Block Diagram:



8-to-1-multiplexer module:

```
module eight_to_one_mux(
input logic i7, i6, i5, i4, i3, i2, i1, i0,
input logic s2, s1, s0,
output logic y);
        logic d1, d0;
        logic and_out1, and_out2;
        logic s2_inv;
        four_to_one_mux mux4_1(i3, i2, i1, i0, s1, s0, d0);
        four_to_one_mux mux4_2(i7, i6, i5, i4, s1, s0, d1);
        not n1(s2_inv, s2);
        and a1(and_out1, d0, s2_inv);
        and a2(and_out2, d1, s2);
        or o1(y, and_out1, and_out2);
```

```
endmodule
```

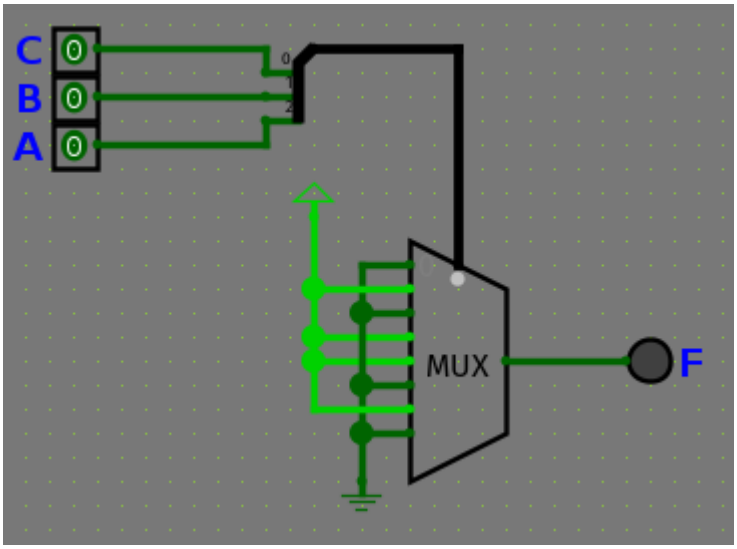
8-to-1-multiplexer testbench:

```
module eight_to_one_mux_tb();
	reg[7:0] i;
	reg[2:0] s;
	logic d;
	eight_to_one_mux mux8(i[7], i[6], i[5], i[4], i[3], i[2], i[1], i[0], s[2], s[1], s[0], d);
	initial begin
		//i2 = 0; i3 = 0;
		assign s = 3'b000;
		assign i = 8'b00000000; #1;
		for(int j = 0; j < 255; j++) begin
			assign i = i + 8'b00000001; #1;
		end
		assign s = 3'b001;
		assign i = 8'b00000000; #1;
		for(int j = 0; j < 256; j++) begin
			assign i = i + 8'b00000001; #1;
		end
		assign s = 3'b010;
		assign i = 8'b00000000; #1;
		for(int j = 0; j < 256; j++) begin
			assign i = i + 8'b00000001; #1;
		end
		assign s = 3'b011;
		assign i = 8'b00000000; #1;
```

```
            for(int j = 0; j < 256; j++) begin
                assign i = i + 8'b00000001; #1;
            end
            assign s = 3'b100;
            assign i = 8'b00000000; #1;
            for(int j = 0; j < 256; j++) begin
                assign i = i + 8'b00000001; #1;
            end
            assign s = 3'b101;
            assign i = 8'b00000000; #1;
            for(int j = 0; j < 256; j++) begin
                assign i = i + 8'b00000001; #1;
            end
            assign s = 3'b110;
            assign i = 8'b00000000; #1;
            for(int j = 0; j < 256; j++) begin
                assign i = i + 8'b00000001; #1;
            end
            assign s = 3'b111;
            assign i = 8'b00000000; #1;
            for(int j = 0; j < 256; j++) begin
                assign i = i + 8'b00000001; #1;
            end
        end
endmodule
```

Part g) of Preliminary Report:

F(A,B,C,D) Block Diagram:



F(A,B,C,D) module:

```
module lab_function(
                input logic a, b, c, d,
                output logic y);

        eight_to_one_mux mux8(0, 1, 0, 1, 1, 0, 1, 0, a, b, c, y);
endmodule
```