Name - Akshat Shah

Roll No - 21BCP322

Sem – III

Digital Electronics and Computer Organization Lab

Course Code – 20CP203P

Department of Computer Science Engineering
School of Technology,
Pandit Deendayal Energy University

Date:12-10-2022

Assignment-7

List of Questions:

Question 1: Write a Verilog code to implement BCD to seven segment display Decoder. Prepare the Truth Table and circuits and verify the same using Test Bench.

About Seven segment display:

- A seven-segment display (SSD) is a form of electronic display device for displaying decimal numbers.
- It has seven segments and we can use it to display 2^7=128 character combinations and we can form combinations of display numerical from 0 to 9.
- Each display unit is represented by a dot point.

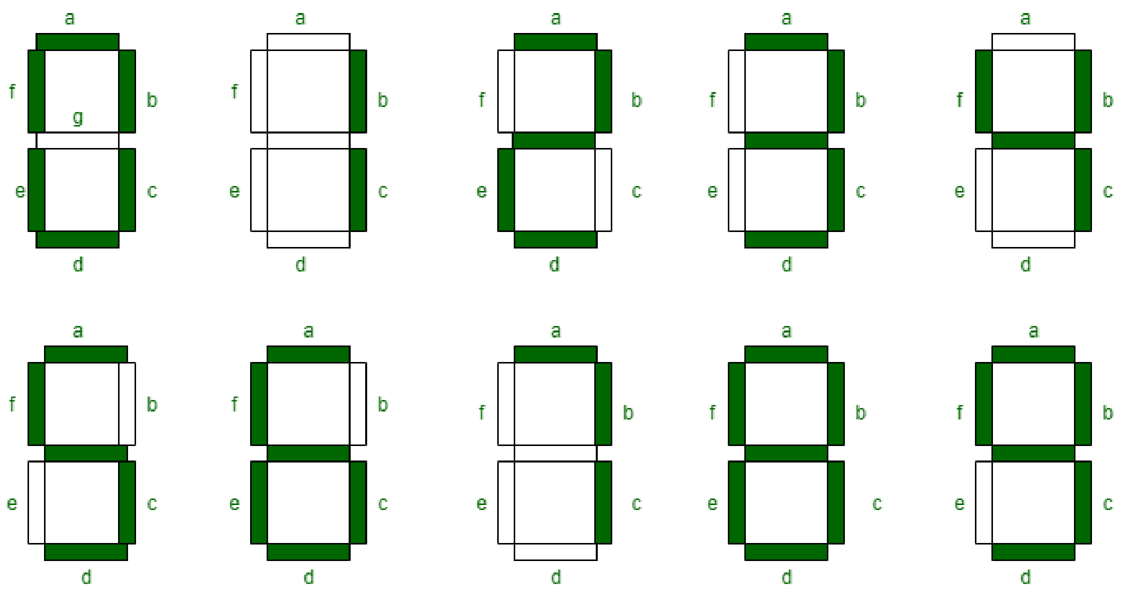


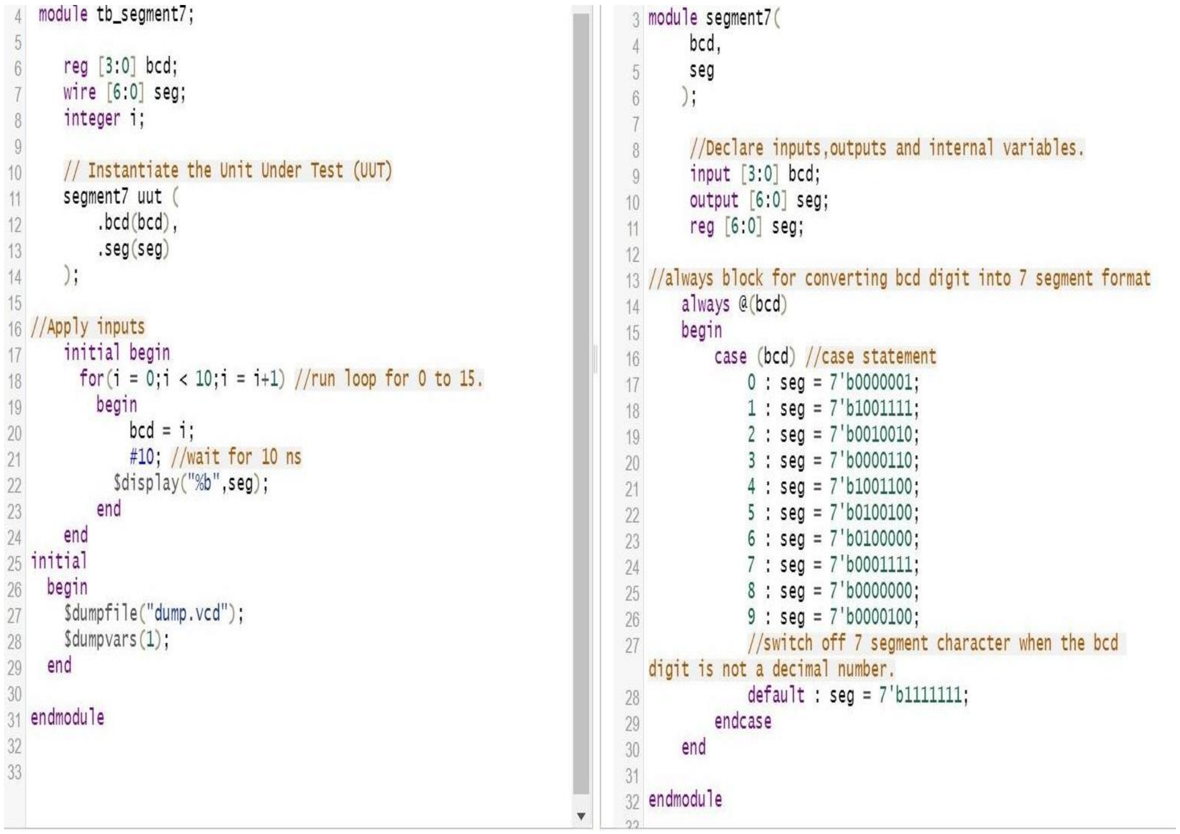
Fig-1: Display of decimal nos using SSD

Applications:

1. Digital clocks
2. Clock radios
3. Calculators
4. Vehicle odometers

Test bench:

```
//Akshat Shah
//21BCP322
```

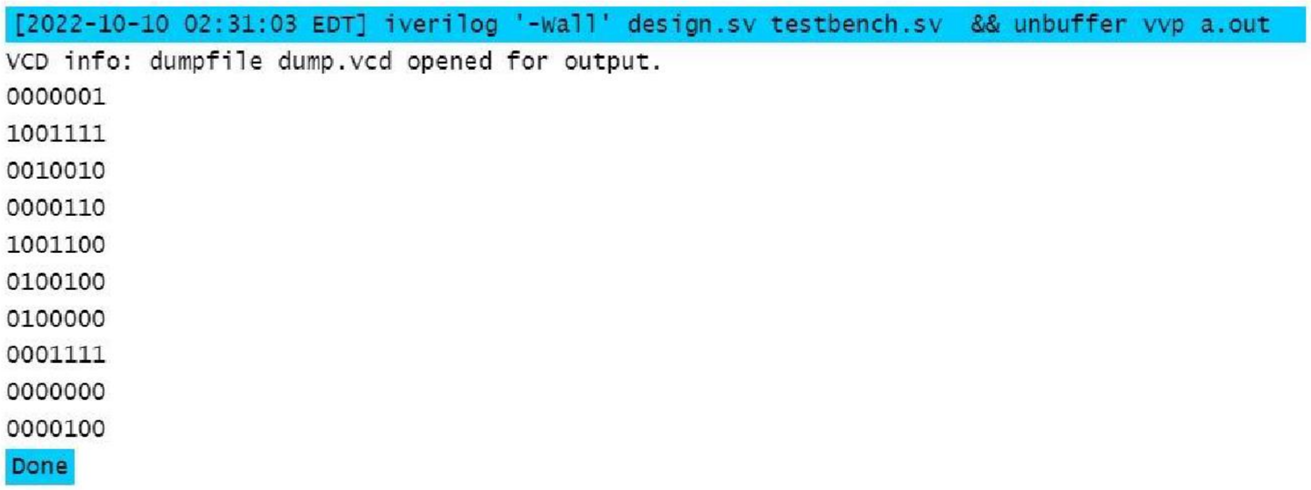


```
module tb_segment7;

   reg [3:0] bcd;
   wire [6:0] seg;
   integer i;

   // Instantiate the Unit Under Test (UUT)
   segment7 uut (
       .bcd(bcd),
       .seg(seg)
   );

//Apply inputs
   initial begin
     for(i = 0;i < 10;i = i+1) //run loop for 0 to 15.
       begin
           bcd = i;
           #10; //wait for 10 ns
         $display("%b",seg);
       end
   end
initial
  begin
   $dumpfile("dump.vcd");
   $dumpvars(1);
  end

endmodule
```

```
module segment7(
     bcd,
     seg
    );

     //Declare inputs,outputs and internal variables.
     input [3:0] bcd;
     output [6:0] seg;
     reg [6:0] seg;

//always block for converting bcd digit into 7 segment format
    always @(bcd)
    begin
        case (bcd) //case statement
            0 : seg = 7'b0000001;
            1 : seg = 7'b1001111;
            2 : seg = 7'b0010010;
            3 : seg = 7'b0000110;
            4 : seg = 7'b1001100;
            5 : seg = 7'b0100100;
            6 : seg = 7'b0100000;
            7 : seg = 7'b0001111;
            8 : seg = 7'b0000000;
            9 : seg = 7'b0000100;
            //switch off 7 segment character when the bcd
digit is not a decimal number.
            default : seg = 7'b1111111;
        endcase
    end

endmodule
```

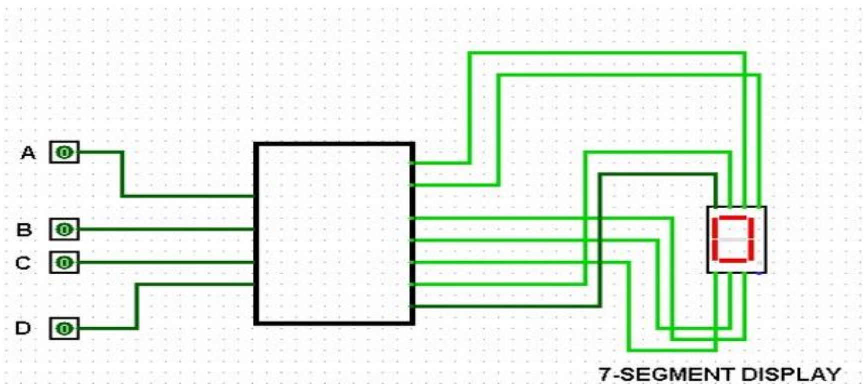
Fig-1.1 Verilog Code for SSD

Output:

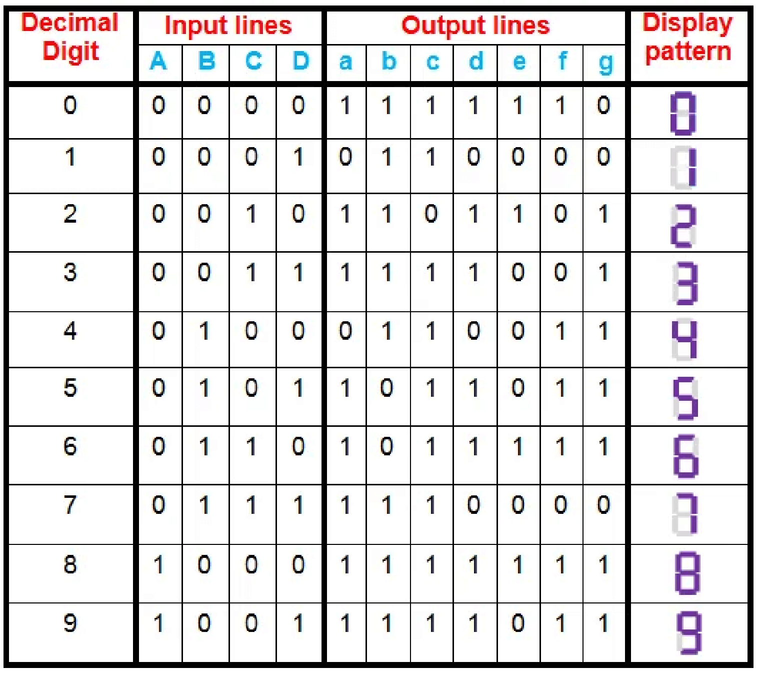


```
[2022-10-10 02:31:03 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
VCD info: dumpfile dump.vcd opened for output.
0000001
1001111
0010010
0000110
1001100
0100100
0100000
0001111
0000000
0000100
Done
```

Circuit diagram:



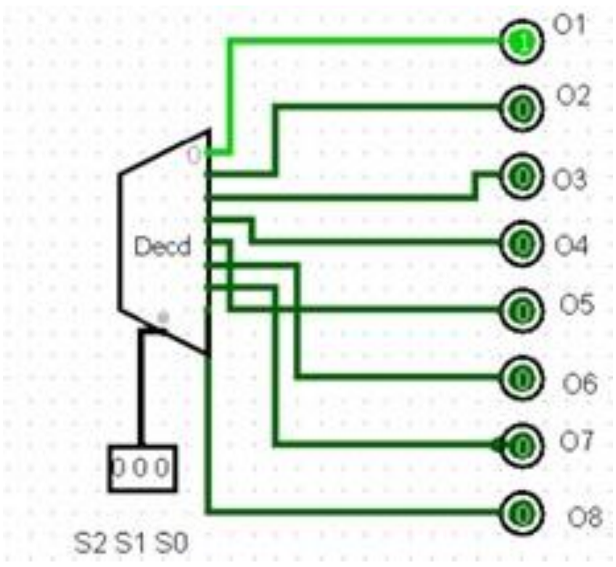
Truth table:



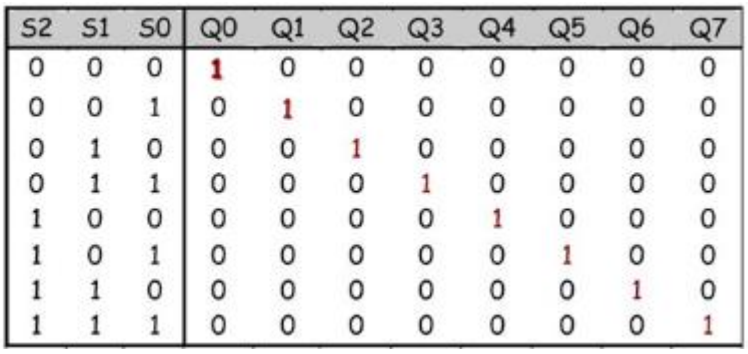
Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	0
1	0	0	0	1	0	1	1	0	0	0	0	1
2	0	0	1	0	1	1	0	1	1	0	1	2
3	0	0	1	1	1	1	1	1	0	0	1	3
4	0	1	0	0	0	1	1	0	0	1	1	4
5	0	1	0	1	1	0	1	1	0	1	1	5
6	0	1	1	0	1	0	1	1	1	1	1	6
7	0	1	1	1	1	1	1	0	0	0	0	7
8	1	0	0	0	1	1	1	1	1	1	1	8
9	1	0	0	1	1	1	1	1	0	1	1	9

2. Design a 3:8 Decoder using Logisim.

Circuit Diagram:



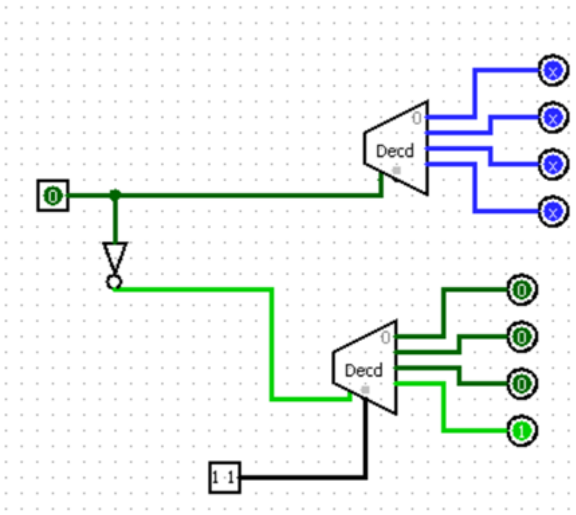
Truth table:



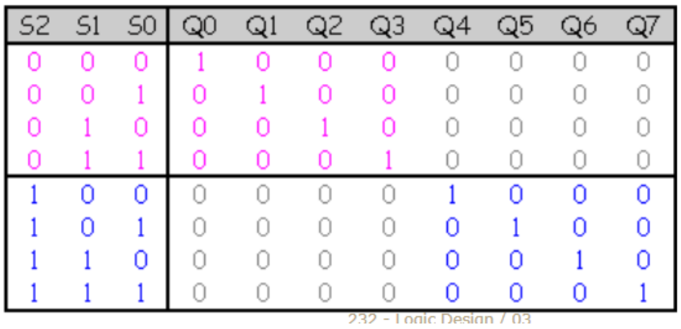
S2	S1	S0	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

3. Design a 3:8 Decoder using two 2:4 Decoder using Logisim.

Circuit diagram:



Truth table:

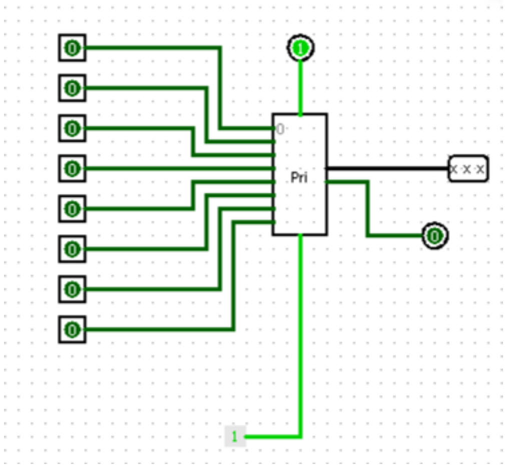


S2	S1	S0	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

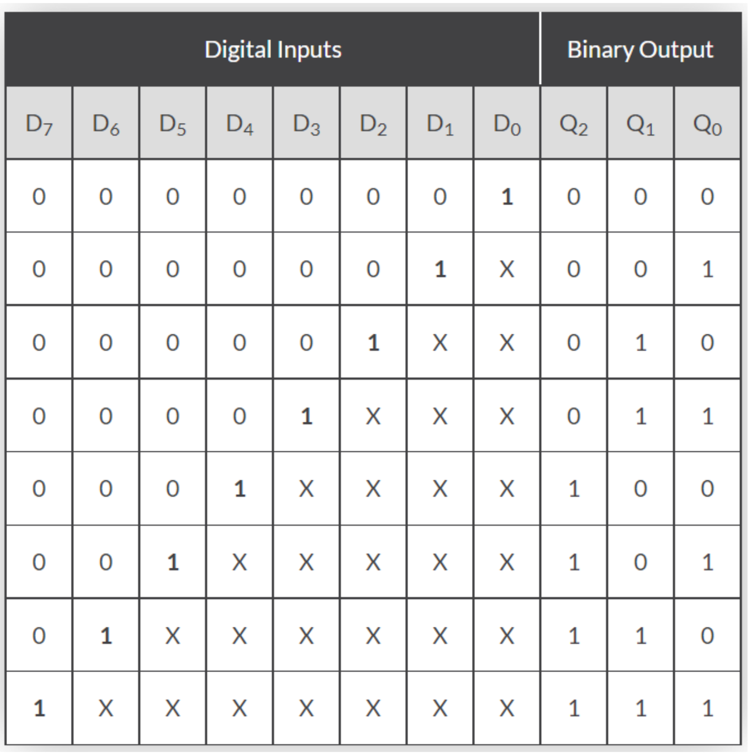
Q0 = S2′ S1′ S0′
Q1 = S2′ S1′ S0
Q2 = S2′ S1 S0′
Q3 = S2′ S1 S0
Q4 = S2 S1′ S0′
Q5 = S2 S1′ S0
Q6 = S2 S1 S0′
Q7 = S2 S1 S0

4.Design an 8:3 Priority Encoder using Logisim

Circuit diagram:



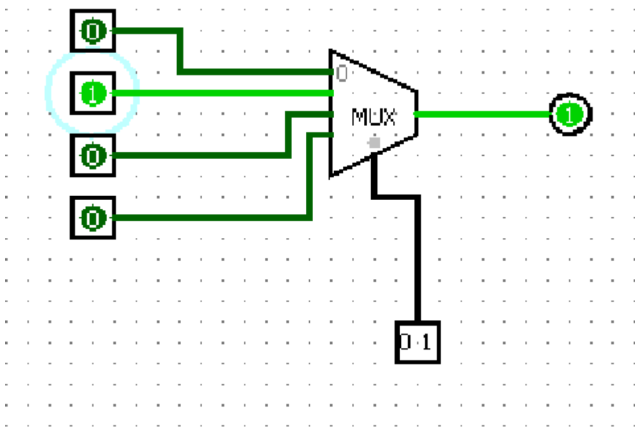
Truth table:



Digital Inputs								Binary Output		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	**1**	0	0	0
0	0	0	0	0	0	**1**	X	0	0	1
0	0	0	0	0	**1**	X	X	0	1	0
0	0	0	0	**1**	X	X	X	0	1	1
0	0	0	**1**	X	X	X	X	1	0	0
0	0	**1**	X	X	X	X	X	1	0	1
0	**1**	X	X	X	X	X	X	1	1	0
1	X	X	X	X	X	X	X	1	1	1

5. Design a 4:1 Multiplexer using Logisim.

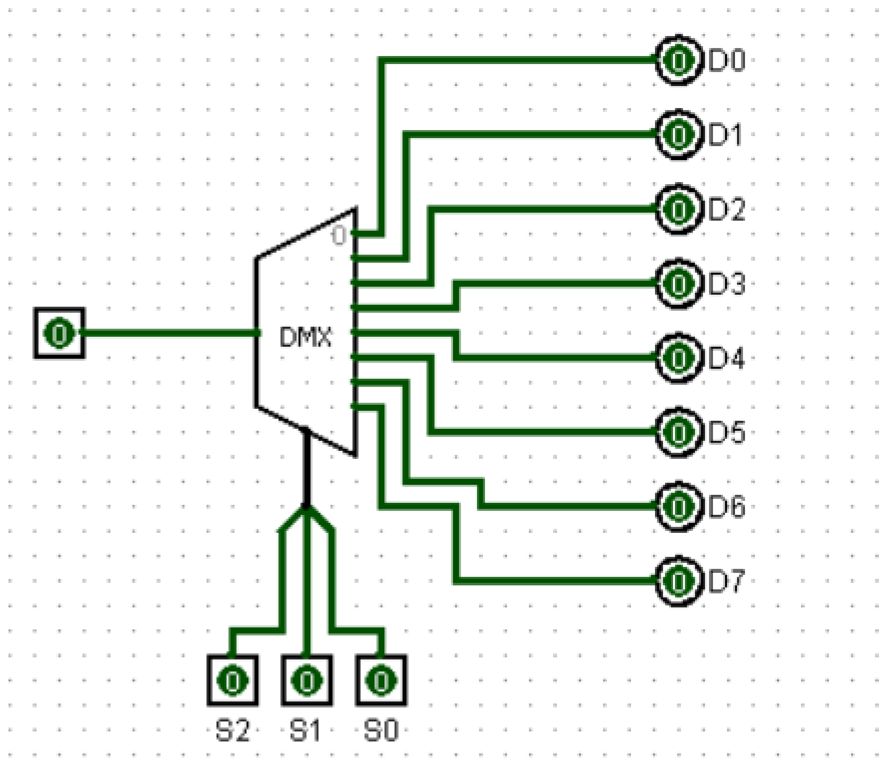
Circuit diagram:



Truth table:

Selection Lines		Output
S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

6. Design a 1:8 Demultiplexer using Logisim.



Truth table:

INPUT			**OUTPUT**							
S2	S1	S0	d7	d6	d5	d4	d3	d2	d1	d0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0