EEE/INSTR F313
ANALOG AND DIGITAL VLSI DESIGN

DIGITAL ASSIGNMENT

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

PROBLEM 4

SUBMITTED BY

NIRANJAN CHAUDHARI	2020B5A30929P
SUKRITI MATHUR	2020B5A81399P
SAARTHAK VIJAYVARGIYA	2021A3PS1044P

UNDER GUIDANCE OF
DR. ANU GUPTA

Table of Contents

Problem Statement and Specifications

Problem 4.

(a) Design a 3:8 decoder. Design using the basic gates (use minimum type of gates).

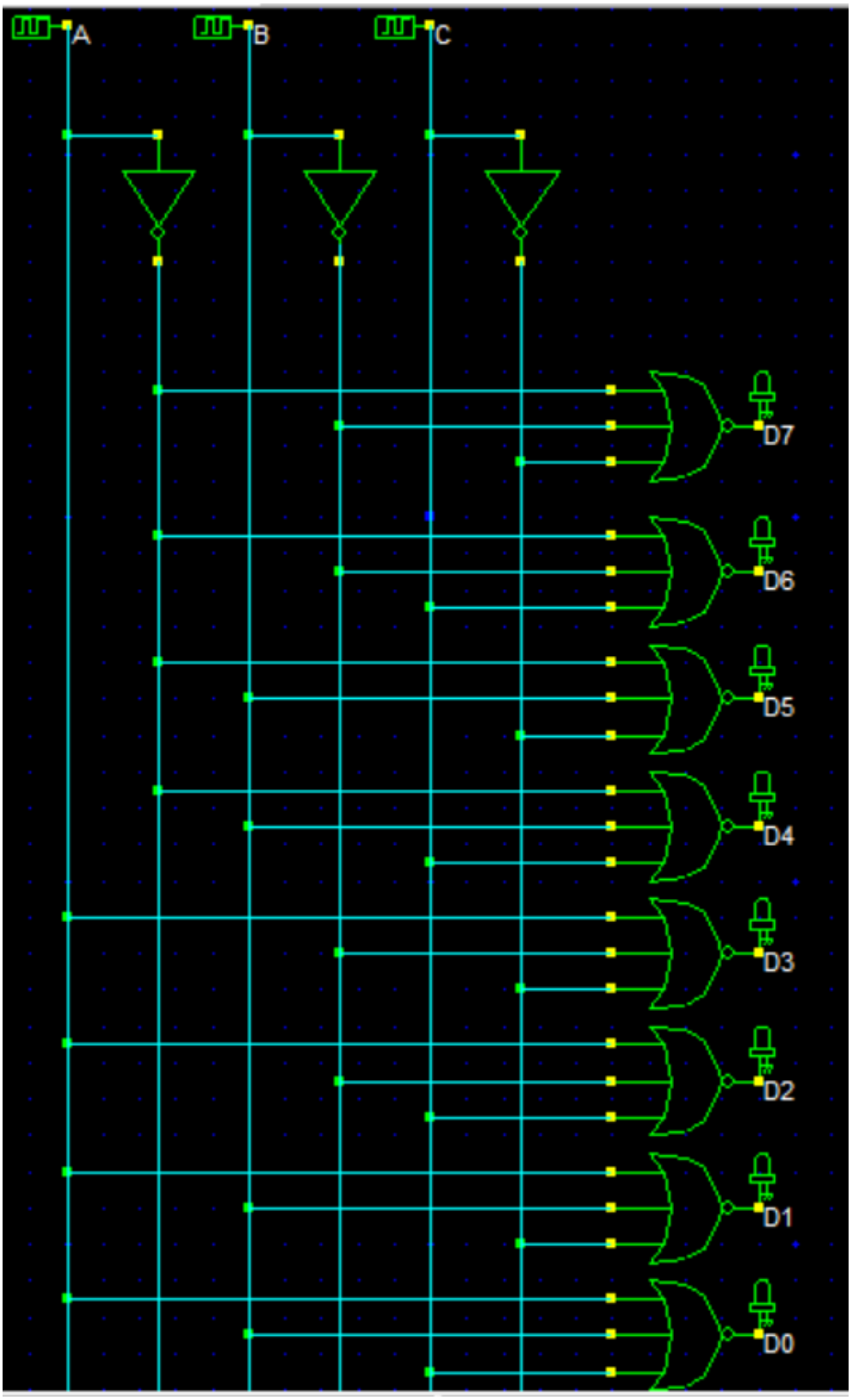
Specifications:

- 180 nm technology is used.
- Vdd is 2V for this technology.
- Body of PMOS is connected to Vdd and the body of NMOS is connected to ground.

Truth Table for 3:8 Decoder

X	Y	Z	D0	D1	D2	D3	D4	D5	D6	D7
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

Schematic of 3:8 Decoder in DSCH3



Verilog Code We Developed

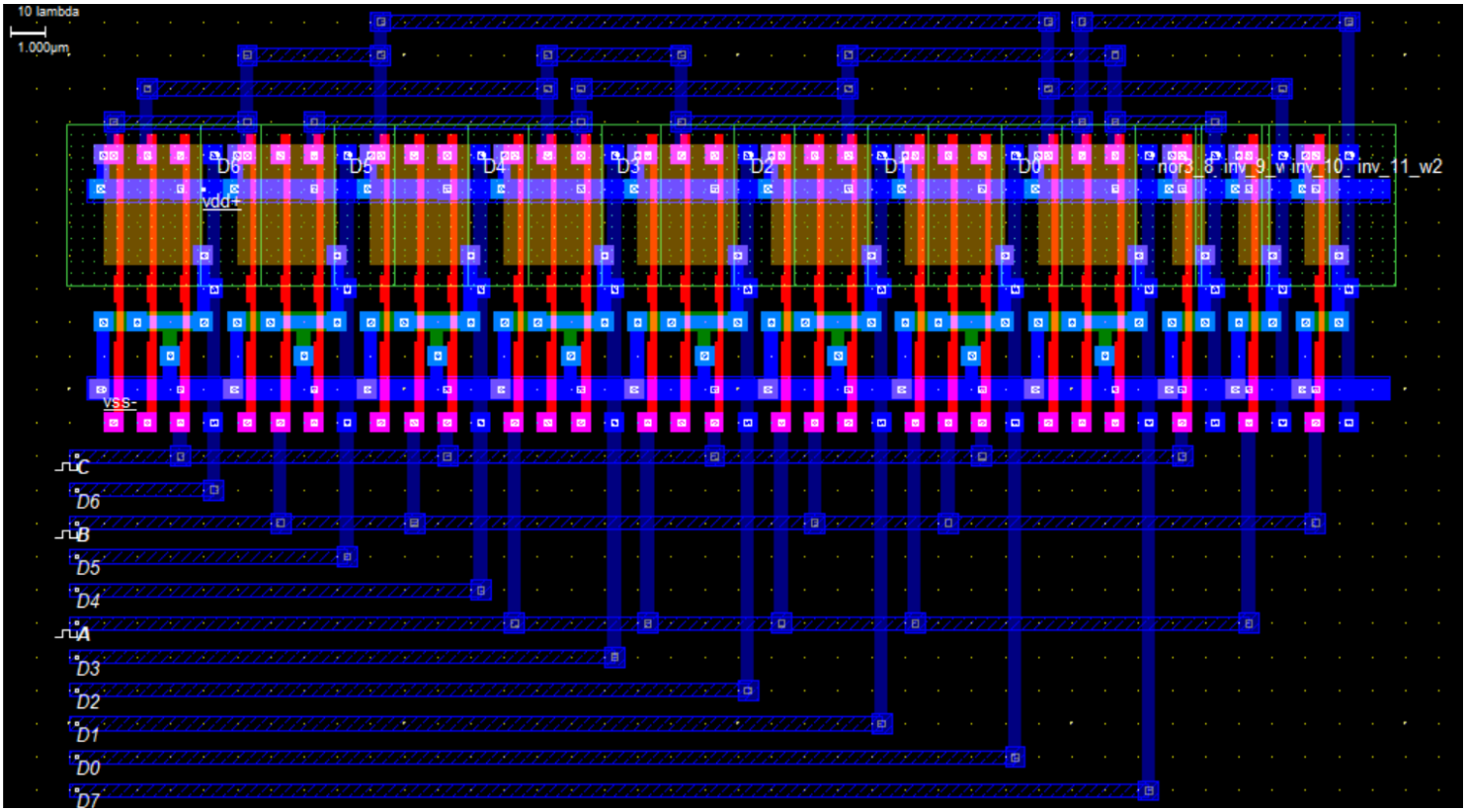
```
module decoder_3to8( A, B, C, D0, D1, D2, D3, D4, D5, D6, D7);
 input A, B, C;
 output D0, D1, D2, D3, D4, D5, D6, D7;
 wire An, Bn, Cn;
 not inv_1(An, A);
 not inv_2(Bn, B);
 not inv_3(Cn, C);
 nor3 nor3_0(D0, A, B, C);
 nor3 nor3_1(D1, A, B, Cn);
 nor3 nor3_2(D2, A, Bn, C);
 nor3 nor3_3(D3, A, Bn, Cn);
 nor3 nor3_4(D4, An, B, C);
 nor3 nor3_5(D5, An, B, Cn);
 nor3 nor3_6(D6, An, Bn, C);
 nor3 nor3_7(D7, An, Bn, Cn);
endmodule

module not(res, inp);
input inp;
output res;
always begin
 res <= ~inp;
end
endmodule

module nor3(res, a, b, c);
input a, b, c;
output res;
always begin
 res <= ~(a || b || c);
end
endmodule
```

Verilog Code Generated by DSCH

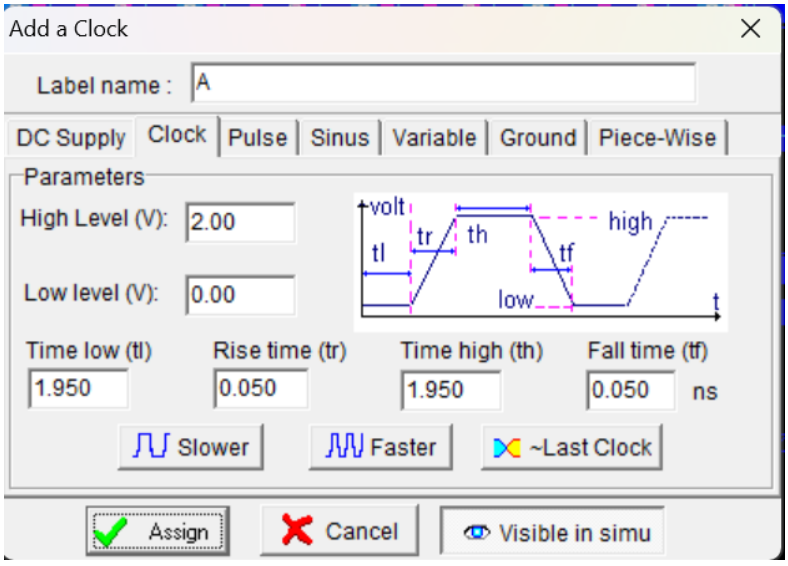
```
module decoder_gates( A,B,C,D4,D7,D6,D5,D0,
 D1,D2,D3);
 input A,B,C;
 output D4,D7,D6,D5,D0,D1,D2,D3;
 nor #(13) nor3_1(D6,w1,w2,C);
 nor #(13) nor3_2(D5,w1,B,w6);
 nor #(13) nor3_3(D4,w1,B,C);
 nor #(13) nor3_4(D3,A,w2,w6);
 nor #(13) nor3_5(D2,A,w2,C);
 nor #(13) nor3_6(D1,A,B,w6);
 nor #(13) nor3_7(D0,A,B,C);
 nor #(13) nor3_8(D7,w1,w2,w6);
 not #(31) inv_9(w6,C);
 not #(31) inv_10(w1,A);
 not #(31) inv_11(w2,B);
endmodule

// Simulation parameters in Verilog Format
always
#2000 A=~A;
#1000 B=~B;
#500 C=~C;

// Simulation parameters
// A CLK 20.000 20.000
// B CLK 10.000 10.000
// C CLK 5.000 5.000
```

Microwind Layout

By compiling the verilog files generated in DSCH, and keeping the transistor specifications in mind, i.e., width PMOS = 36λ and width NMOS = 6λ, we got the circuit as follows.



Here A is set with details as follows:



Similarly, B is set with: tl = th = 0.950 ns

C is set with: tl = th = 0.450 ns

Transistors Specifications

There are a total of 54 transistors used for this design.
Using Logical effort delay model for the 3 input NOR gate, (Wp/Wn) = 6
We take L=Lmin = 180 nm for all transistors.

MOSFET	W (nm)	W/L
PMOS	3240	18
NMOS	540	3

As in Microwind, because of software limitation to 90nm technology, 1λ = 100 nm, hence while making circuit by compiling verilog code, we provided the following parameters.

MOSFET	W (nm)	L (nm)	W/L
PMOS	3600	200	18
NMOS	600	200	3

For inputs:
Frequency of A: ¼ * Frequency of C
Frequency of B: ½ * Frequency of C

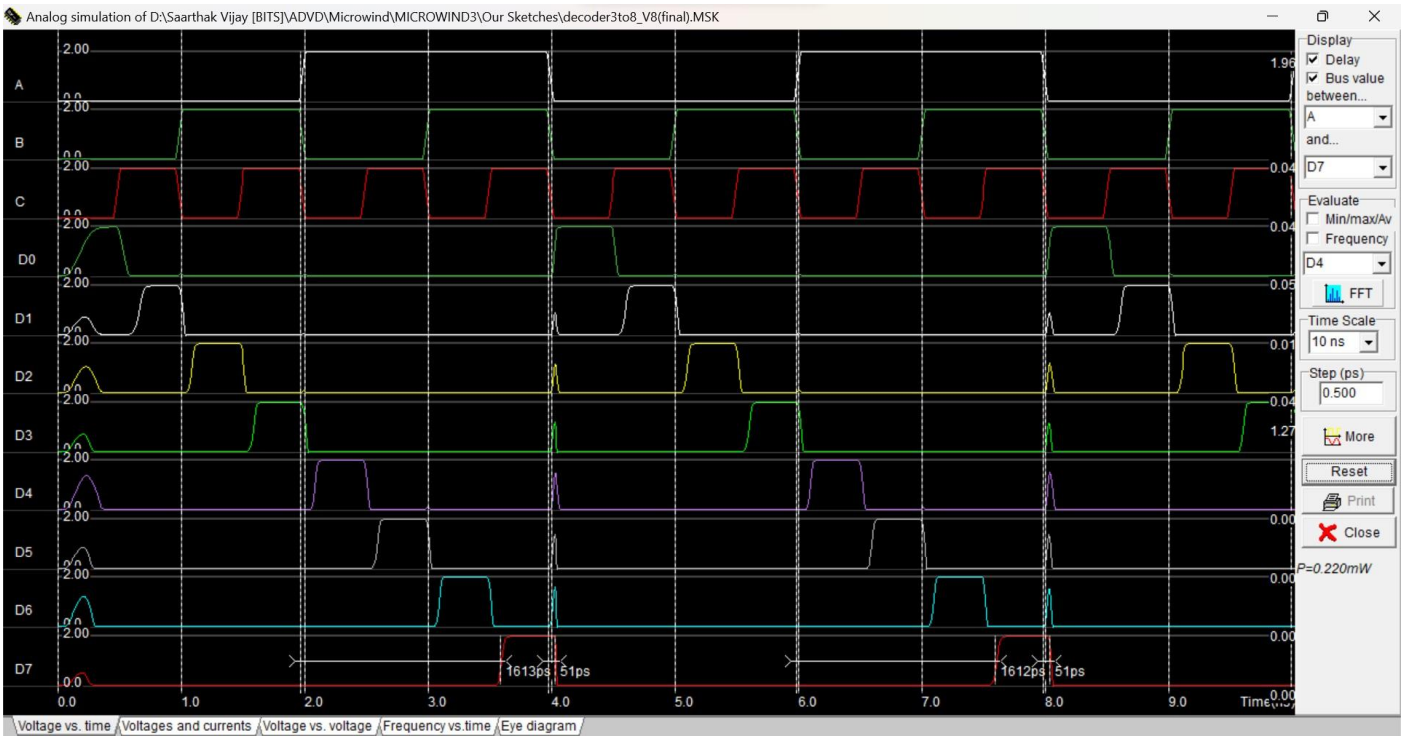
Results

The layout was simulated for different frequencies:

For frequency = 1000MHz (tl + th + tr + tf = 1 ns for C)

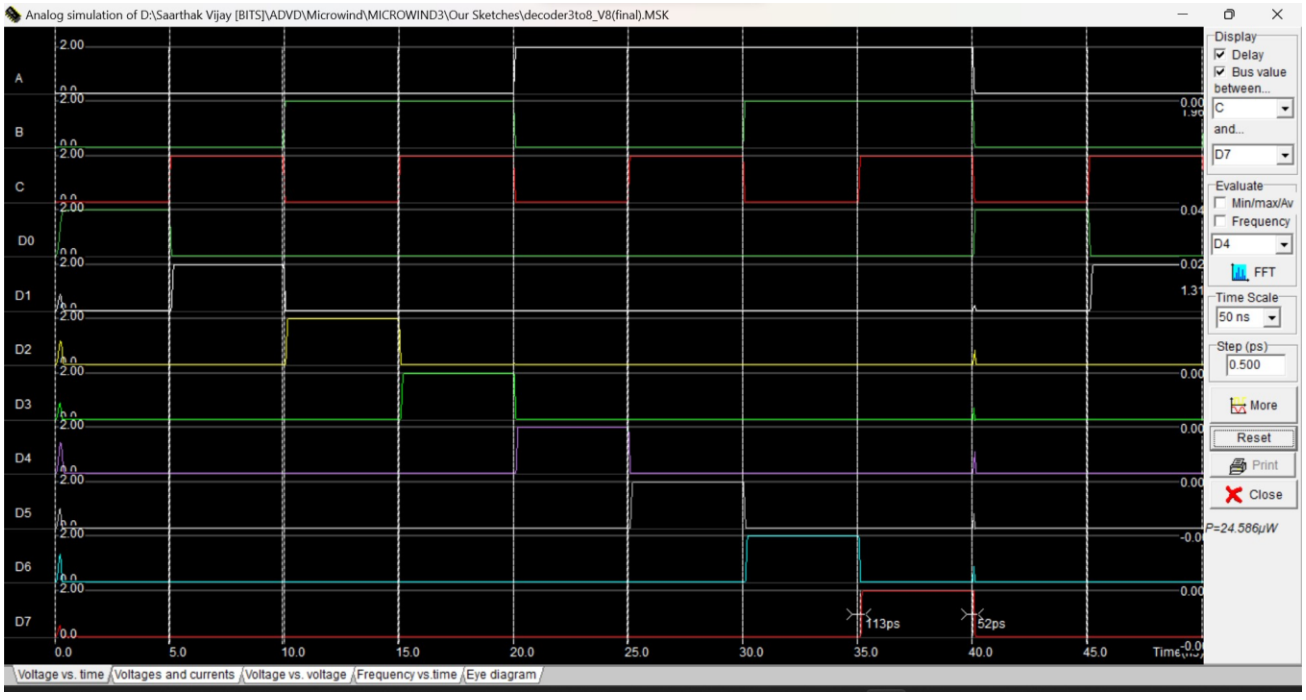
Power = 0.245 mW

Output	Worst Case Delay (ps)
D0	32
D1	184
D2	108
D3	107
D4	101
D5	105
D6	103
D7	113



For frequency = 100MHz (tl + th + tr + tf = 10 ns for C)
Power = 24.586 µW

Output	Worst Case Delay (ps)
D0	34
D1	110
D2	109
D3	108
D4	102
D5	105
D6	103
D7	113



Innovation

We saw that the circuit made by compiling the verilog code was taking 54 MOSFETs and we were able to reduce the no. of transistors involved, hence we started the circuit from scratch and designed a new and better circuit.

We designed an alternate circuit for 3:8 decoder directly in the Microwind. It has following specifications:

MOSFET	W (nm)	L (nm)	W/L
PMOS	600	200	3
NMOS	600	200	3

We deviated from our logical effort aspects because it was gaining better results in terms of power, delay and area.

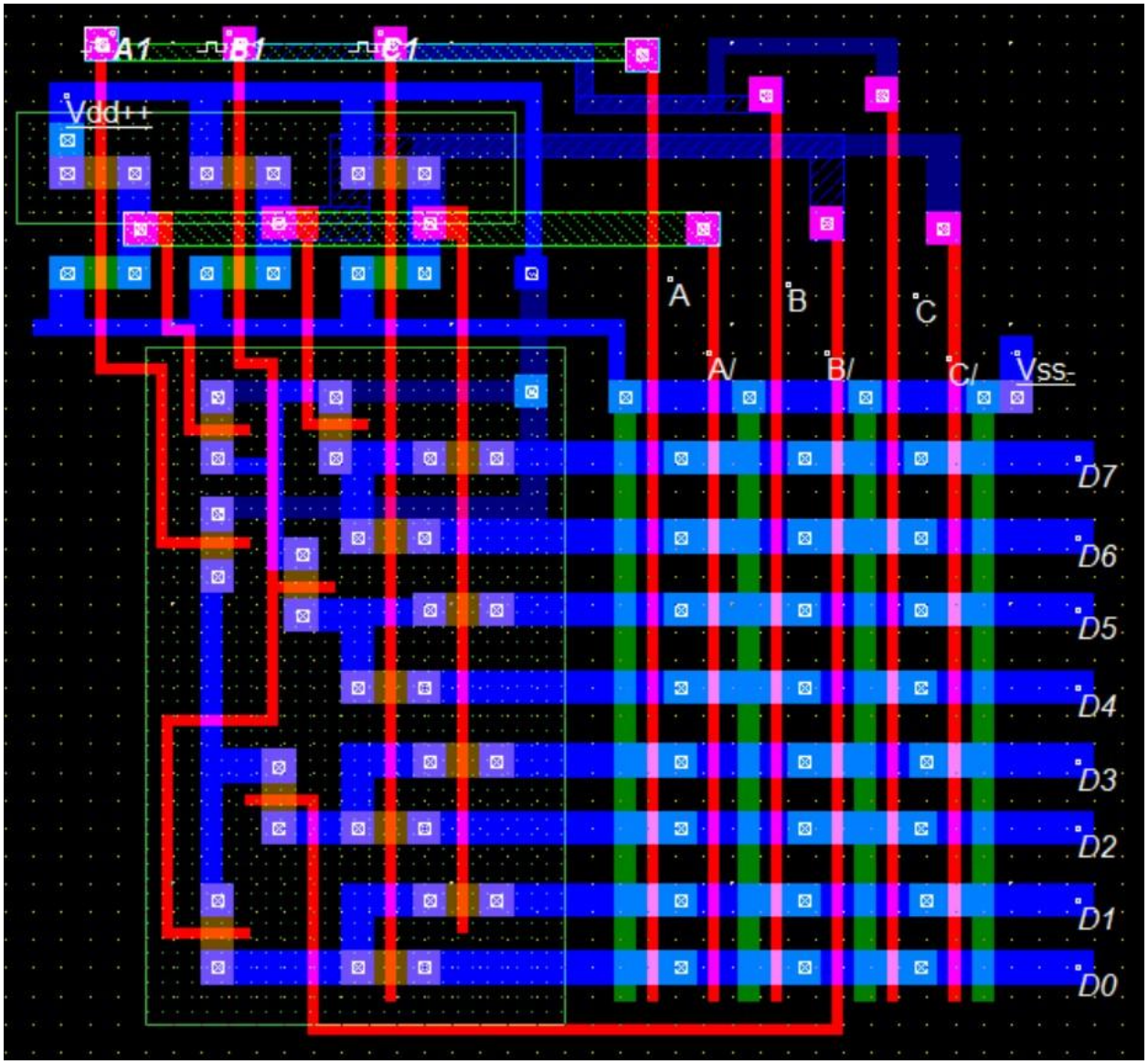
Salient features:

1) It has only 44 MOSFETs instead of 54.
2) It has less power consumption.

Power at frequency	Original	Innovated
1000 MHz	245 μW	92.236 μW
100 MHz	24.586 μW	10.007 μW

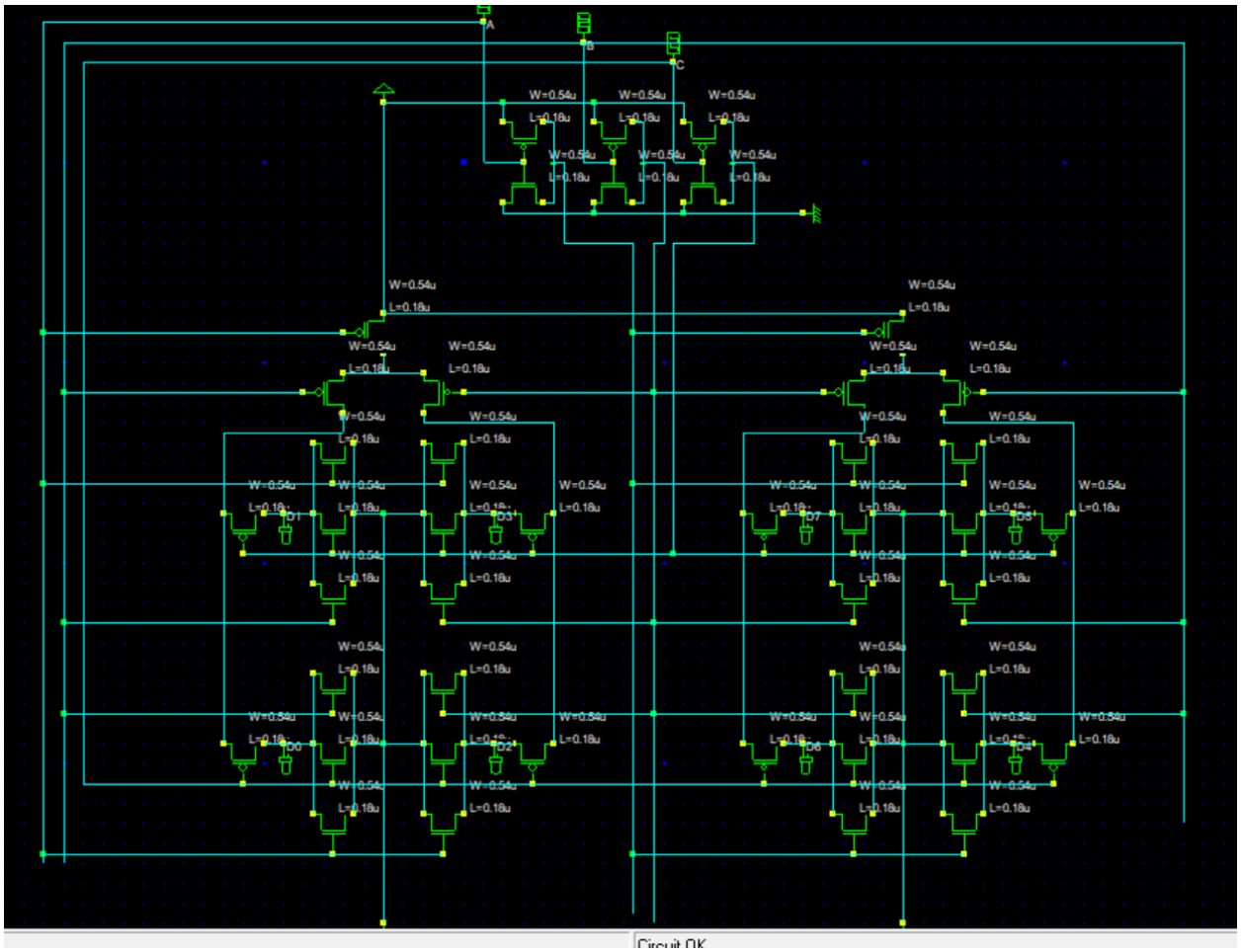
3) It has less delay as compared to the original circuit (Please refer to the Worst case delay tables).
4) It takes less area, as we have reduced the number of horizontal and vertical lines.

Microwind Diagram:

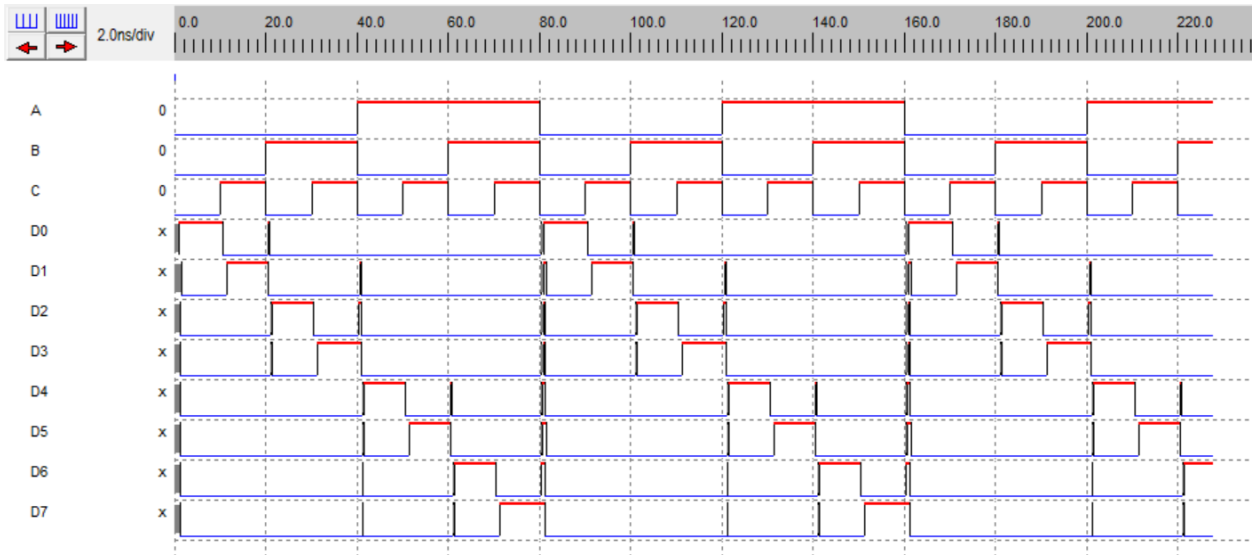


Basic idea behind this circuit was, from D0 to D3, only one PMOS of A is sufficient inside the pull up circuit. Similarly one PMOS of A' for D4 to D7. PMOS A is connected to one PMOS of B for D0,D1, also PMOS A is connected to one PMOS of B' for D2,D3.
Also all the NMOS are connected in a linear fashion to reduce the area.

DSCH3 Schematic:



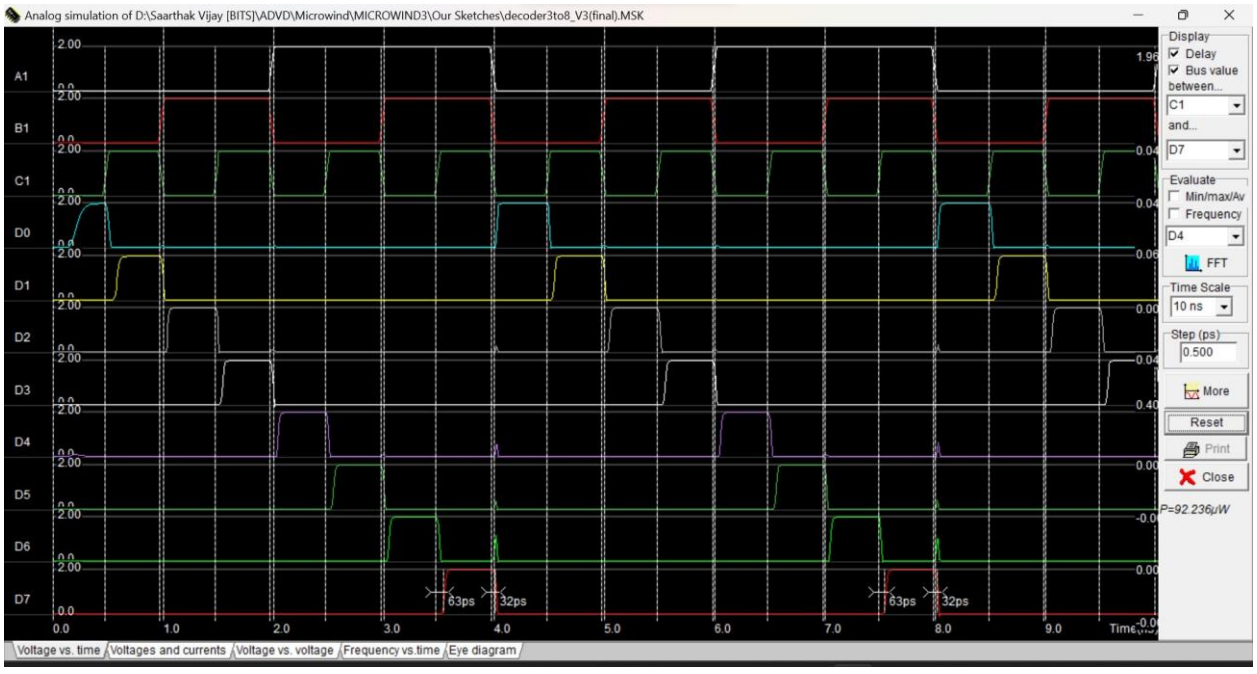
Timing Diagram in DSCH:



The layout of microwind was simulated for different frequencies:

- For frequency = 1000MHz
  Power = 92.236 µW

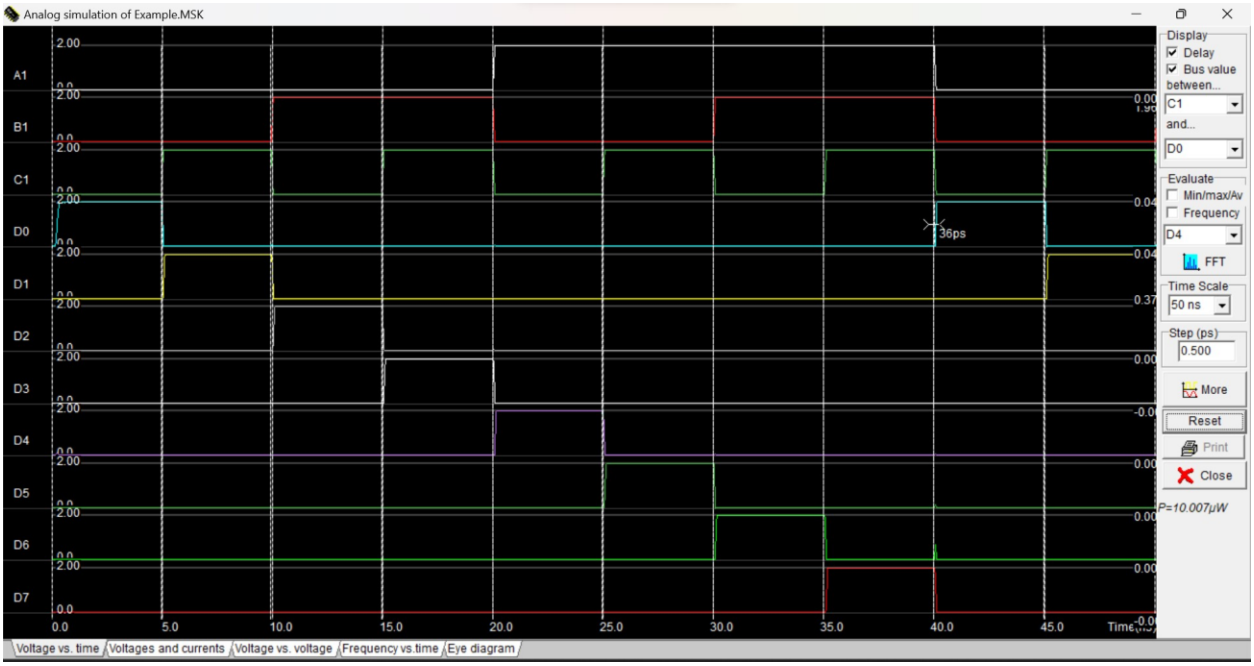
Output	Worst Case Delay (ps) [New]	Worst Case Delay (ps) [Original circuit]
D0	35	32
D1	106	184
D2	61	108
D3	62	107
D4	60	101
D5	63	105
D6	61	103
D7	63	113



We can observe that delays are very less in the new circuit.

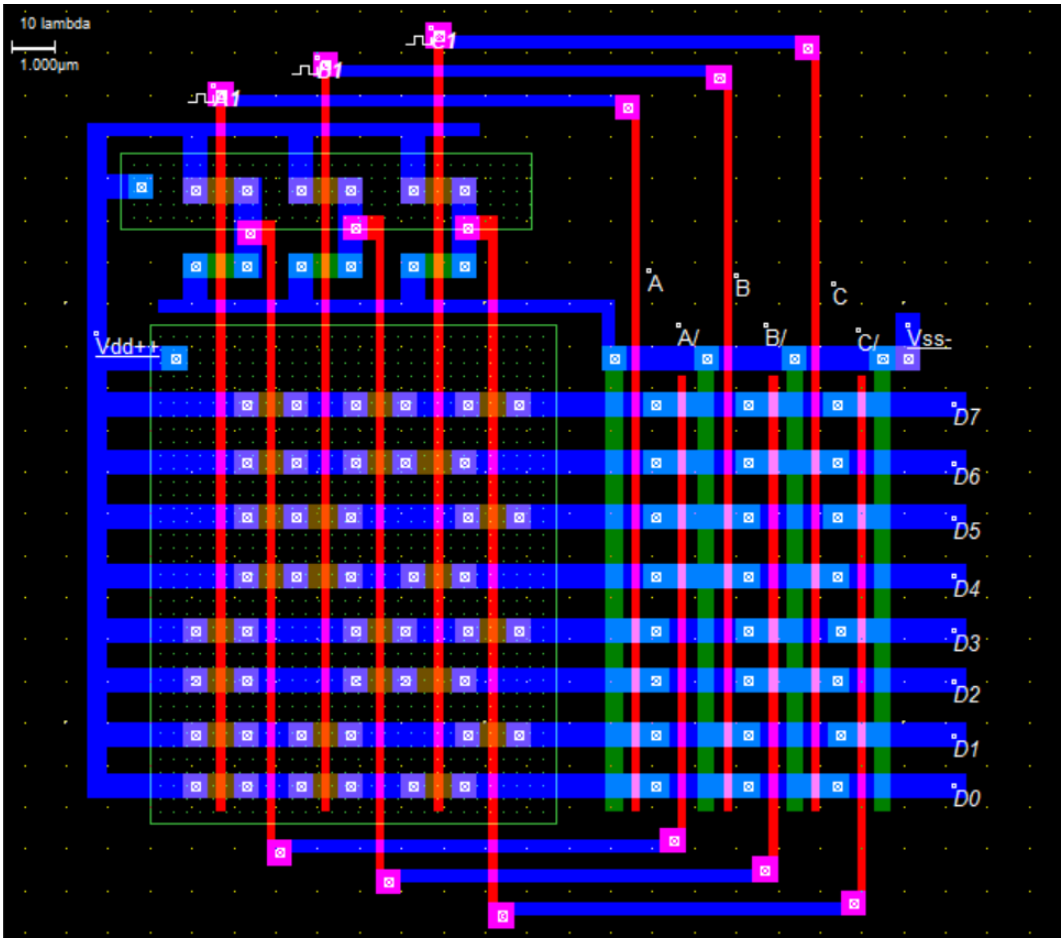
- For frequency = 100MHz
  Power = 10.007 µW

Output	Worst Case Delay (ps) [New]	Worst Case Delay (ps) [Original circuit]
D0	36	34
D1	62	110
D2	62	109
D3	63	108
D4	63	102
D5	63	105
D6	62	103
D7	63	113

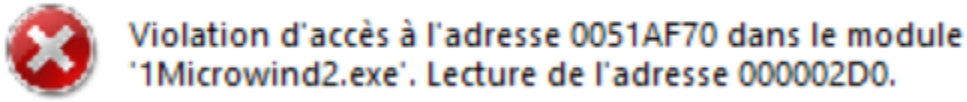


Challenges Faced:

- The design process began by breaking down the given problem into smaller problems which could be combined to solve the original problem.
- Our initial approach after surveying the literature available on the internet was to use Pass Transistor Logic (PTL) for the design. PTL is known for using minimum number of transistors, consuming very less power and less silicon area. The drawback of PTL is that voltage degradation causes problems in driving the next stage and output can be distorted. Hence, PTL implementation demands very good optimization, which was achieved in the circuit design.
- After deriving the circuit from verilog, we came up with a new design in which area was minimized. The diagram is given below:



- After making this circuit, we got an idea of minimizing the area by the process of layering and reducing the MOSFETs but due to Microwind's unfriendliness, the whole circuit has to be redrawn and modified.
- For DSCH2 or DSCH3, while copying or moving the elements, some address error came through the software which was not correctable by Windows, because of which the whole program had to be "Force Stopped" due to which any unsaved work would get lost.



Problem Statement and Specifications

Problem 4.

(b) Design a synchronous sequential circuit has one input and an output. The output, 'O' will go low and remain so if input is high for two consecutive inputs followed by low for two consecutive inputs.

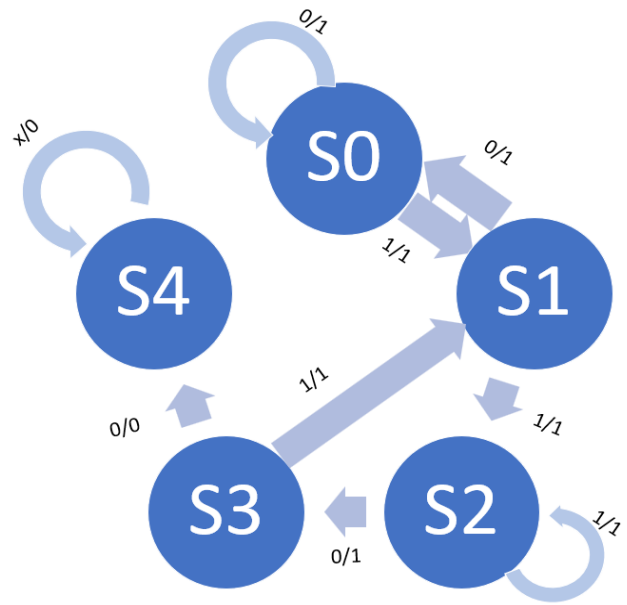
Specifications:

The objective is to detect "1100" input sequence. We are using Mealy machine for detection.

States Declaration:

State	Represents	Binary Coding
S0	start	000
S1	1	001
S2	11	010
S3	110	011
S4	1100	100

State Diagram:



State Table:

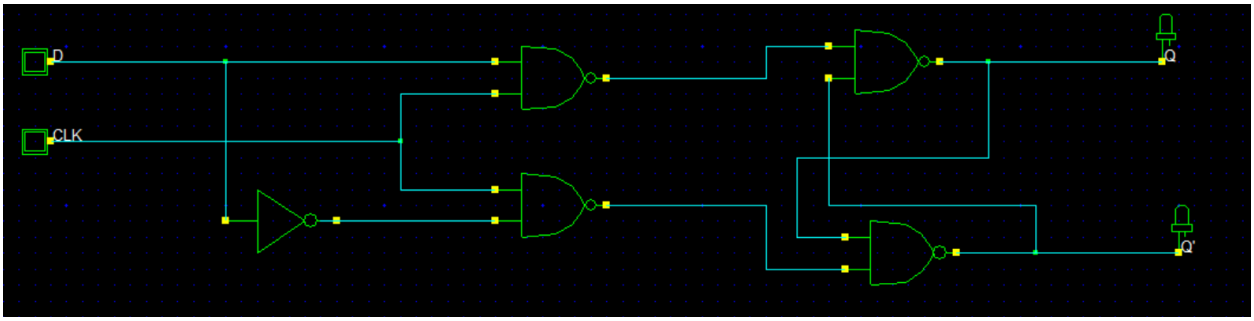
Q2	Q1	Q0	X	$Q2^+$	$Q1^+$	$Q0^+$	Y
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	0	0	1
0	0	1	1	0	1	0	1
0	1	0	0	0	1	1	1
0	1	0	1	0	1	0	1
0	1	1	0	1	0	0	0
0	1	1	1	0	0	1	1
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	x	x	x	x
1	0	1	1	x	x	x	x
1	1	0	0	x	x	x	x
1	1	0	1	x	x	x	x
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	x

We are using D flip-flops for the design.
Solving for expressions:

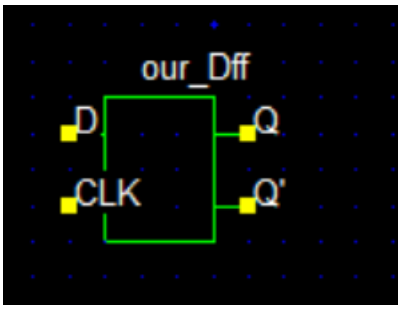
- D2 = Q2 + Q1Q0X’
- D1 = Q1Q0’ + Q1’Q0X
- D0 = Q1Q0’X’ + Q1Q0X +Q2’Q1’Q0’X
- Y = Q2’(Q1’+Q0’+X)

Schematic in DSCH3

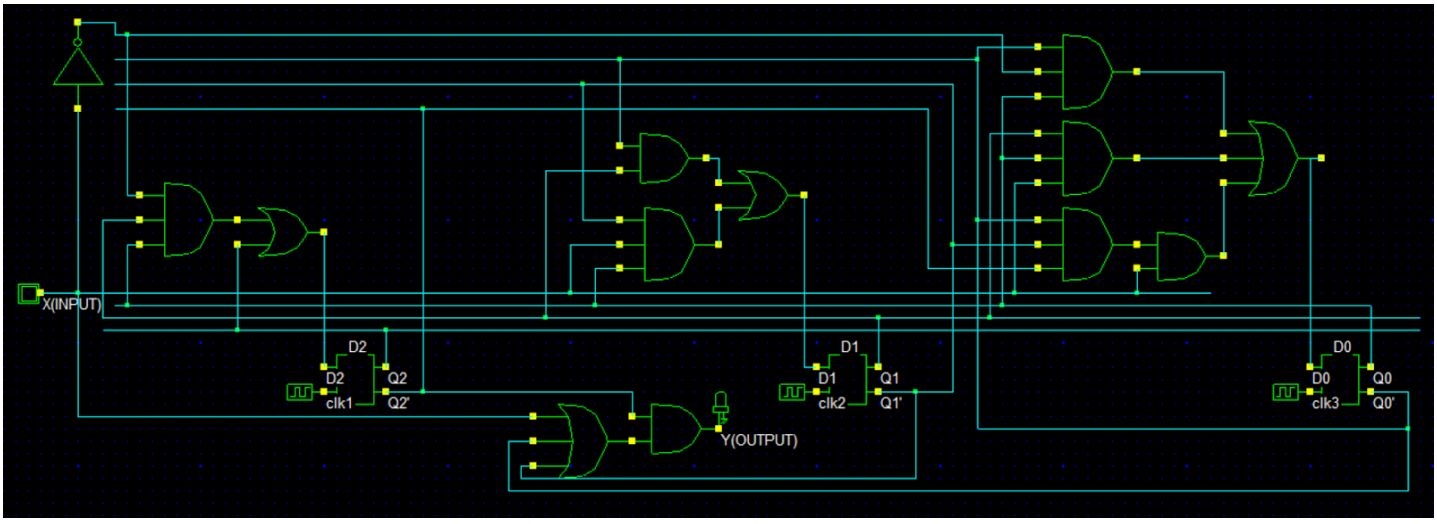
For this purpose, we designed our own edge-triggered D Flip-Flop in DSCH and named it as a symbol to make the bigger circuit.



Which was converted into this symbol:



DSCH3 Schematic:



Verilog Code We Developed

```
module nand(out, in1, in2);
 input in1, in2;
 output out;
always @(posedge CLK)
 out = ~(in1 && in2);
end
endmodule
```

```
module and3(out, in1, in2, in3);
 input in1, in2, in3;
 output out;
always
 out <= (in1 && in2 && in3);
end
endmodule
```

```
module dff( D, CLK, Q, Qn);

input D, CLK;
 output Q, Qn;
 wire w1, w2;

initial begin
 Q <= 0;
 Qo <= 1;
End

 nand nand_1(w1, D, CLK);
 nand nand_2(w2, ~D, CLK);
 nand nand_3(Q, w1, Qo);
 nand nand_4(Qo, w2, Q);

endmodule
```

```
module sequential( cl11, clk2, clk3, X, Y);

input X, clk1, clk2, clk3;
 output Y;
 wire d0, d1, d2, q0, q1, q2, q0n, q1n, q2n,
w1, w2,

 dff dff_1(d0, clk1, q0, q0n);
 dff dff_2(d1, clk2, q1, q1n);
 dff dff_3(d2, clk3, q2, q2n);

 and3 a1(w1, q1, q2, ~X);
 and3 a2(w2, ~q1, q0, X);
 and3 a3(w3, ~q1, ~q0, X);
 and3 a4(w5, q1, q0, X);
 and3 a5(w6, q1, ~q0, ~X);

always
 w4 = ~q2 && w3;
 d0 = w4 || w5 || w6;
 d1= (q1 && ~q0) || w2;
 d2 = w1 || q2;
 Y = ~q2 && (~q1 || ~q0 || X);
end
endmodule
```

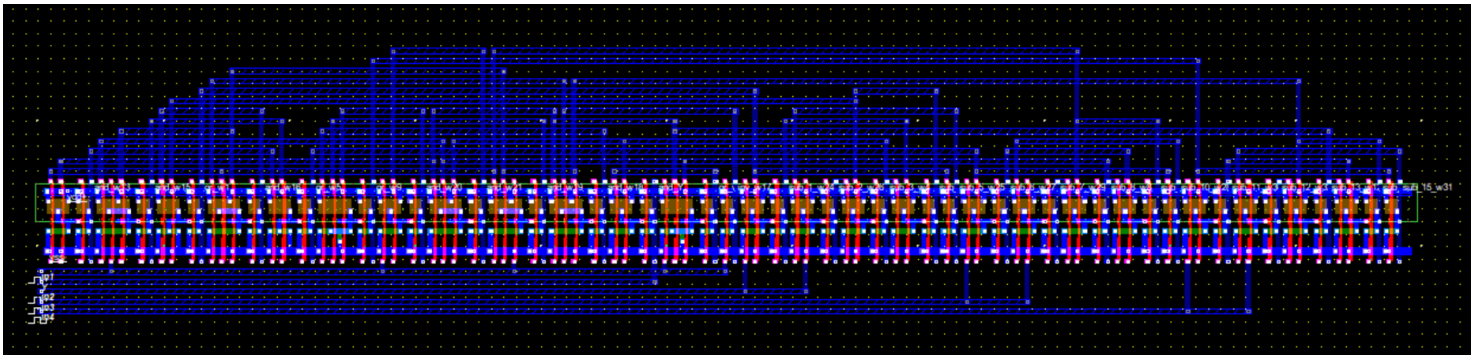
Verilog Code by DSCH

```
module ques2( XINPUT,clk3,clk1,clk2,YOUTPUT);
 input XINPUT,clk3,clk1,clk2;
 output YOUTPUT;
 wire w24,w25,w26,w27,w28,w29,w30,w31;
 wire w32;
 and #(16) and(w13,w7,w12);
 and #(16) and(w15,w8,XINPUT,w11);
 or #(23) or(w1,w16,w3);
 and #(16) and(w16,w17,w7,w11);
 or #(23) or(w5,w13,w15);
 or #(26) or(w9,w18,w19,w20);
 and #(16) and(w20,XINPUT,w21);
 and #(16) and(w21,w12,w8,w4);
 and #(16) and(w19,w7,w11,XINPUT);
 and #(16) and(w18,w12,w17,w11);
 and #(16) and(YOUTPUT,w23,w4);
 or #(19) or(w23,XINPUT,w12,w8);
 not #(17) inv(w17,XINPUT);
 and #(15) sub_1(w24,clk1,w1);
 and #(15) sub_2(w26,w25,clk1);
 and #(34) sub_3(w4,w26,w3);
 and #(27) sub_4(w3,w4,w24);
 not #(14) sub_5(w25,w1);
 and #(15) sub_6(w27,clk2,w5);
 and #(15) sub_7(w29,w28,clk2);
 and #(41) sub_8(w8,w29,w7);
 and #(41) sub_9(w7,w8,w27);
 not #(14) sub_10(w28,w5);
```

```
 and #(15) sub_11(w30,clk3,w9);
 and #(15) sub_12(w32,w31,clk3);
 and #(48) sub_13(w12,w32,w11);
 and #(48) sub_14(w11,w12,w30);
 not #(14) sub_15(w31,w9);
endmodule

// Simulation parameters in Verilog Format
always
#1000 X(INPUT)=~X(INPUT);
#1000 clk3=~clk3;
#1000 clk1=~clk1;
#1000 clk2=~clk2;

// Simulation parameters
// X(INPUT) CLK 10 10
// clk3 CLK 10.000 10.000
// clk1 CLK 10.00 10.00
// clk2 CLK 10.000 10.000
```

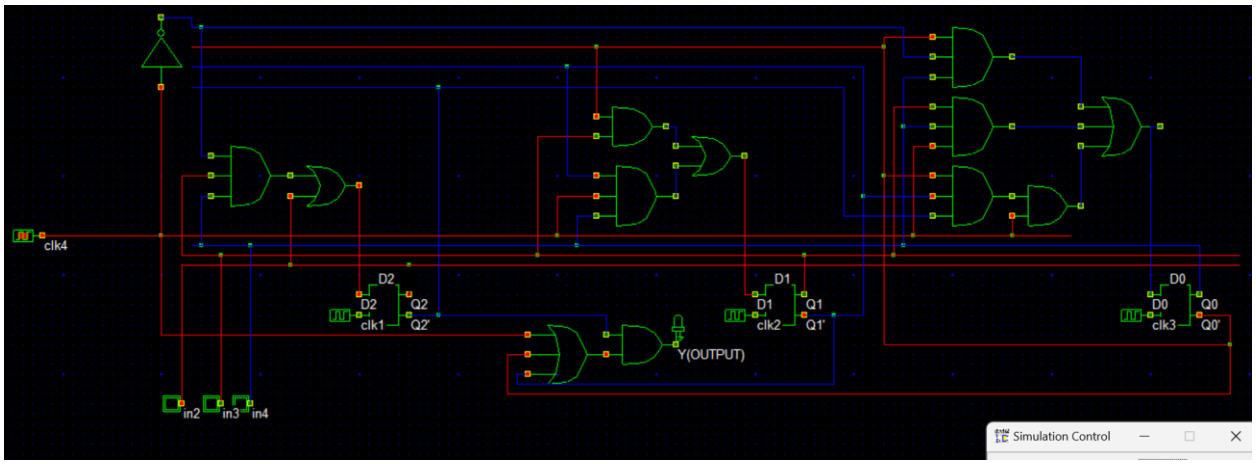
Microwind Circuit:

After inserting the verilog code in Microwind, we got the circuit given in the below picture:



Results:

The resulting simulation gave desired results, and the LED was switched OFF after detecting ‘1100’ sequence.



Conclusion:

Although designing a circuit in our notebooks with the use of the logic gates seems to be easy, the practical implementation of the logic is too challenging which we were able to learn through this assignment.

This assignment taught us:

1) Challenges faced by the fab engineers for optimizing the circuit area, propagation delay, power consumption.
2) Design rules and color-coding schemes.
3) Verilog code and its use.
4) Controlling capacitances and W/L values.