* Digital Electronics and Logic Design (DELD) - Practical Number - 1.

Name:- Kaustubh Shrikant Kabra
Class:- Second Year Engineering.
Div:- A Roll Number:-
Batch:-
Department:- Computer Department
College:- AISSMS's IOIT

Title:- Content to bridge the Gap, Logic Gates.

Aim:-
To realize Logic Gates and there working.

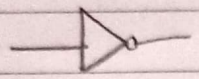
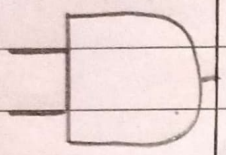
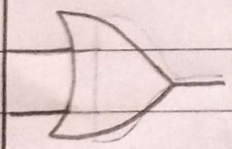
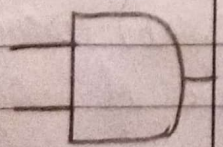
Objective:-
To study the types of gate present.

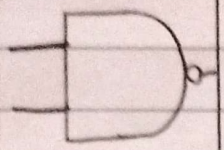
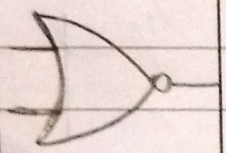
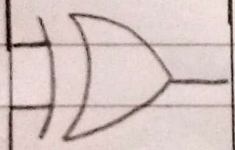
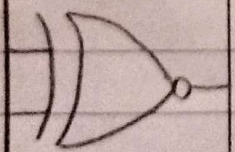
Theory:-

Logic Gates:-
A logic gate is an electronic (device) circuit which make logical decision based on combination of digital / electronical signals present on its input.

Types of Logic Gates:-

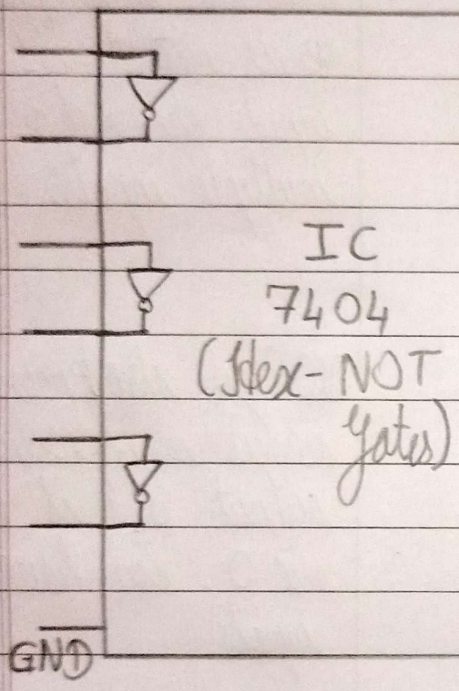
① NOT ② AND ③ OR ④ NAND ⑤ NOR ⑥ EX-OR ⑦ EX-NOR

Gate Name	Gate Symbol	Truth Table	Description
NOT		Input A | Output 0 | 1 1 | 0	Also known as Inverter Gate.
AND		Input A | Input B | Output 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	Output is 1 only if all inputs are 1. AND gates can have two, three, or more inputs.
OR		Input A | Input B | Output 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1	Output is 1 if either of the input are 1. Output is 0 only if all inputs are 0. Can have more than 2 inputs.
NAND		Input A | Input B | Output 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	NAND gate also known as NOT-AND as it is opposite of AND. As its output is 1 if all inputs are not 1. Can have more than one inputs.

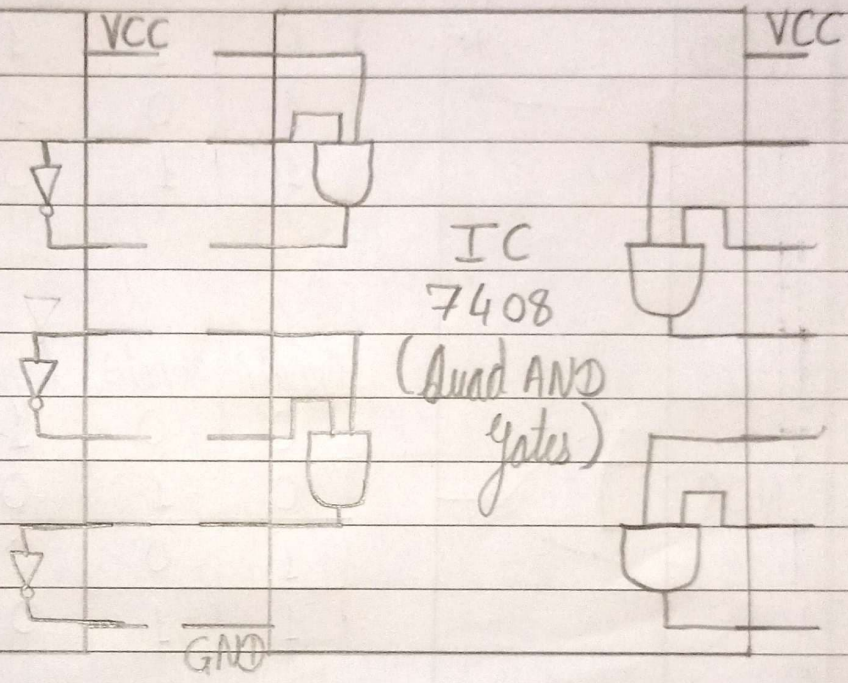
Gate Name	Gate Symbol	Truth Table	Description
NAND		Input A | Input B | Output 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	NAND gate also known as NOT-AND as it is opposite of AND. As its output is 1 if all inputs are not 1. Can have multiple inputs.
NOR		Input A | Input B | Output 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 0	NOR gate also know as NOT-OR as it is opposite of OR, its output is 1 if all inputs are 0. Can have multiple inputs.
EX-OR		Input A | Input B | Output 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	EX-OR also known as Exclusive OR gate, its output is only 1 if either of inputs are 1, but 0 when both are same or all inputs are same.
EX-NOR		Input A | Input B | Output 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	EX-NOR also known as Exclusive NOR gate, its output is 1 only if all inputs are same. Can have multiple inputs.

ICs of Logic Gates:-

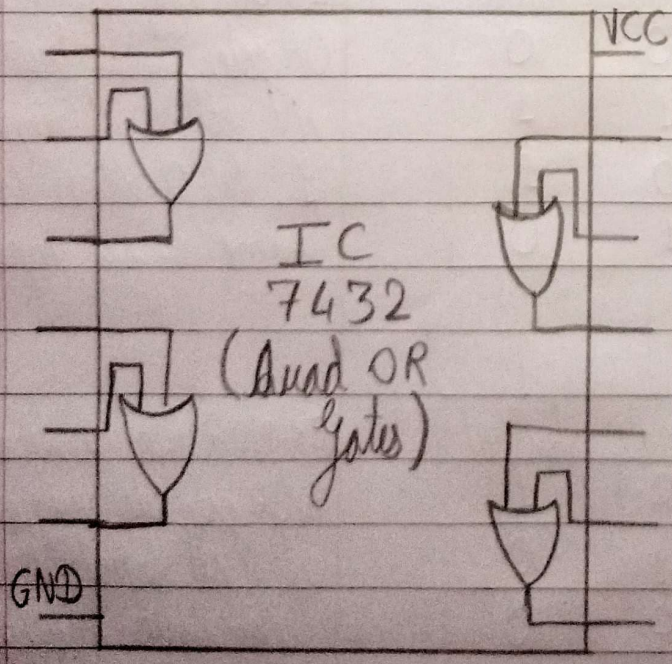
1) NOT (IC-7404)



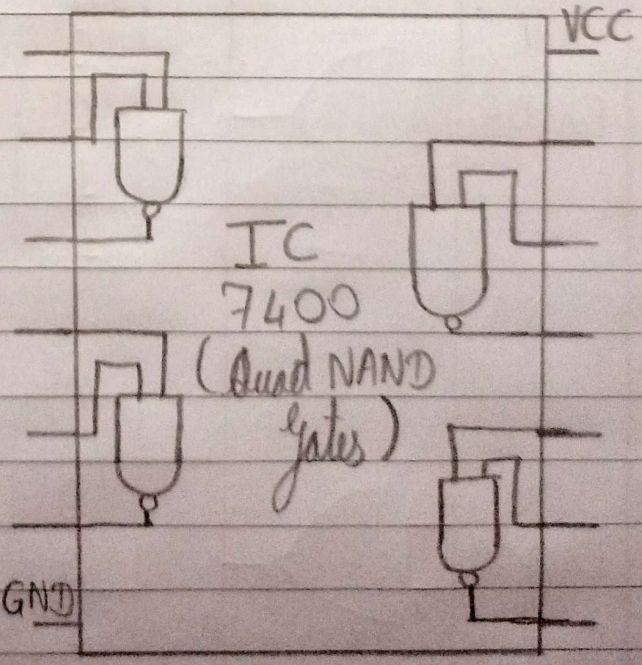
2) AND (IC 7408)



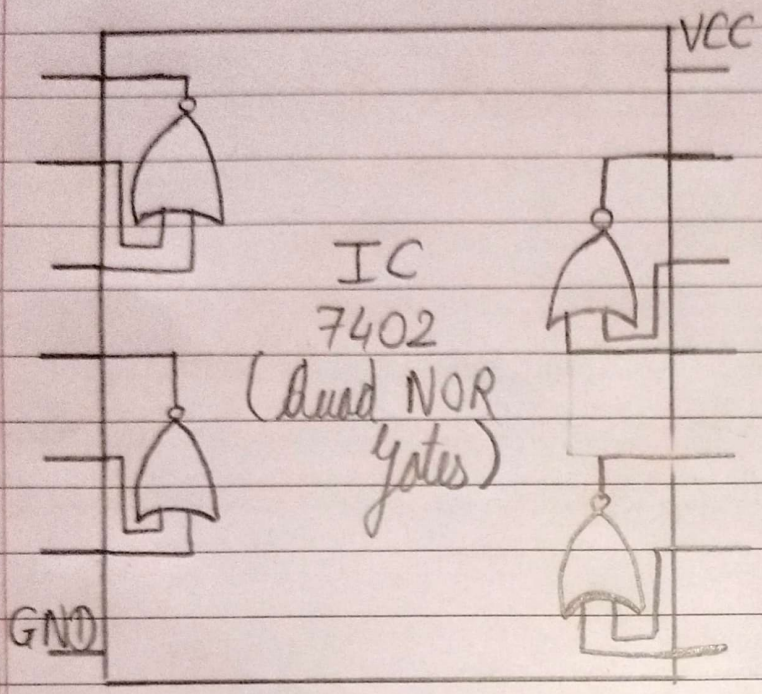
3) OR (IC-7432)



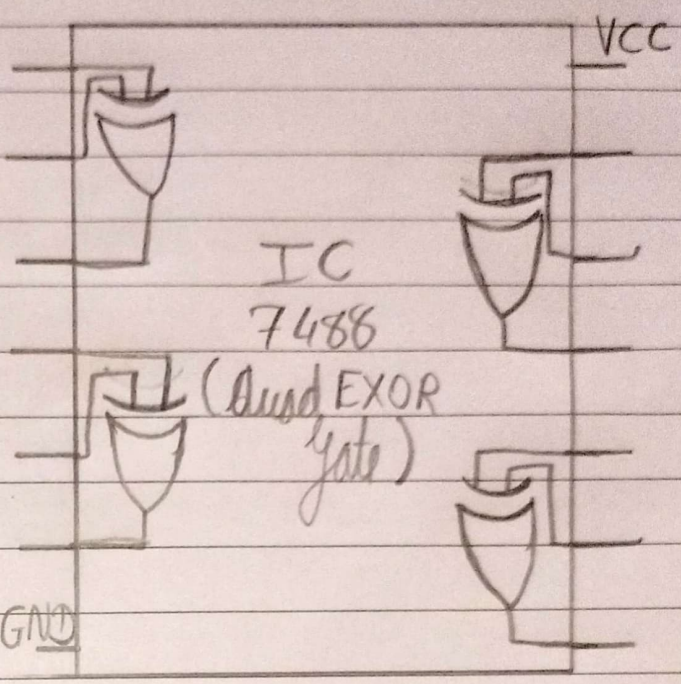
4) NAND (IC-7400)



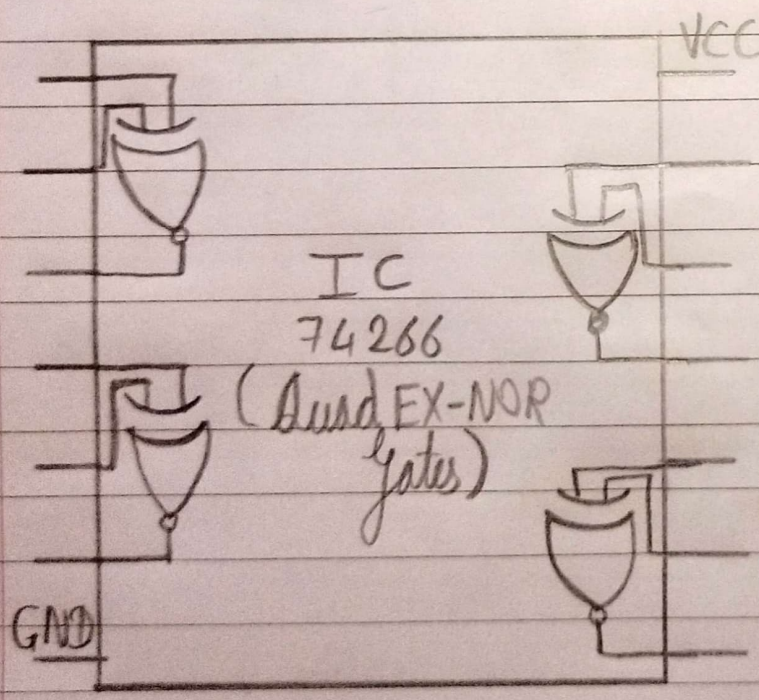
5) NOR (IC-7402)



6) EX-OR (IC-7486)



7) EX-NOR (IC-74266)



Conclusion:-

Hence we have studied the Basic Logic Gate.

* Digital Electronics and Logic Design (DELD) - Practical Number - 2

Name:- Kaustubh Shrikant Kabra.
Class:- Second Year Engineering.
Div:- A Roll Number:-
Batch:-
Department:- Computer Engineering Department
College:- AISSMS's IOIT.

Title:-
Full Adder and Full Subtractor

Aim:-
To realize full adder and full subtractor using
a) Basic gates
b) Universal gates.

Objective:-
Understanding the working of full adder and full subtractor by only using a) Basic Gates
b) Universal Gates.

Theory:-
a) Full Adder:-
The half adder does not take the carry bit from its previous stage into account. This carry bit from its previous stage is called carry-in bit. A combinational logic circuit that adds two data bits, x and y and a carry-in bit, Cin, is called a full

adder. The Boolean functions describing the full-adder are:

$$S = (x \oplus y) \oplus C_{in} \qquad C = xy + C_{in}(x \oplus y)$$

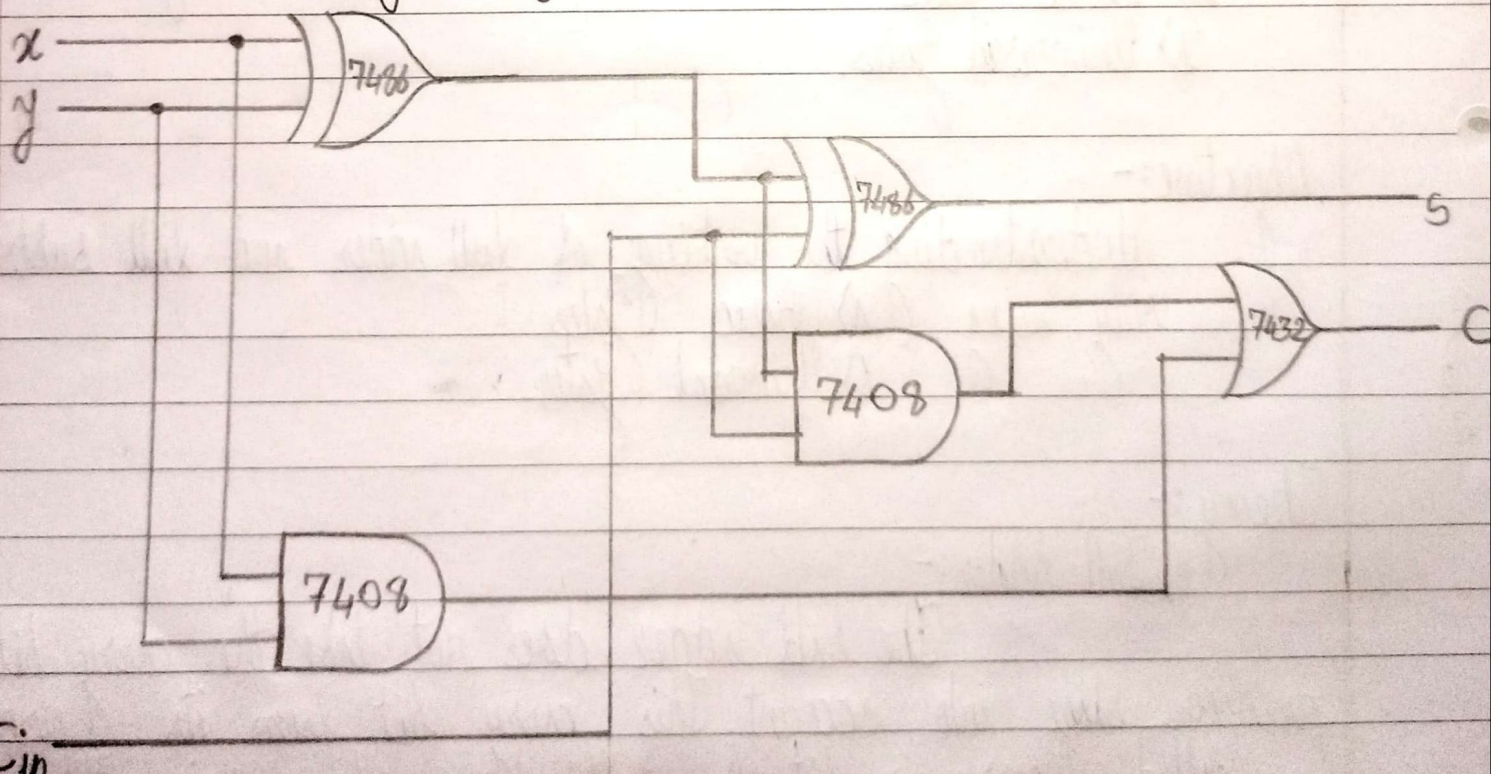
b) Full Subtractor:-

Subtracting two single-bit binary values, B, C_{in} from a single-bit value A produces a difference bit D and a borrow out Br bit. This is called full subtraction. The boolean functions describing the full-subtracter are:

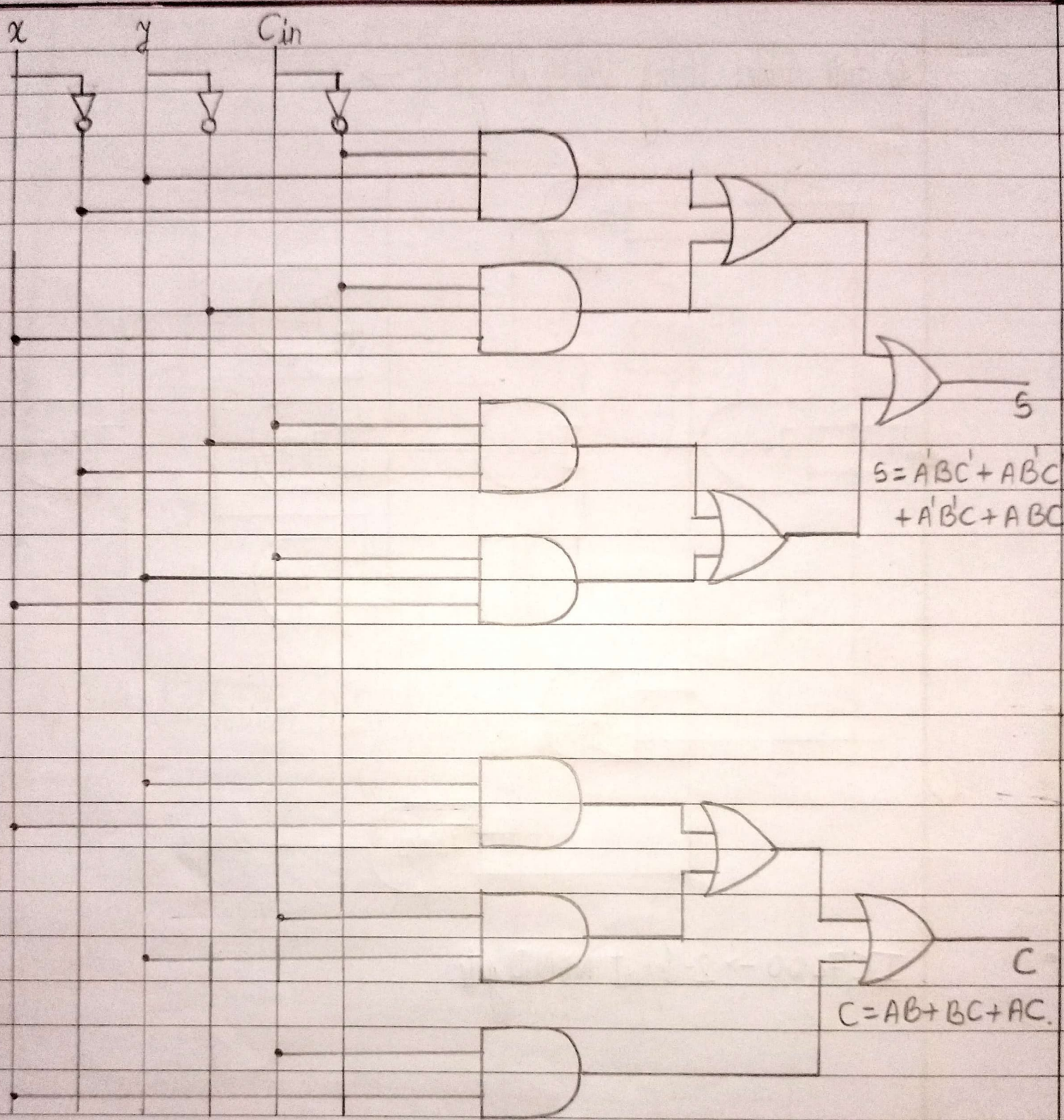
$$D = (A \oplus B) \oplus C_{in} \qquad Br = A'B + A'(C_{in}) + B(C_{in})$$

Logic Diagram:-

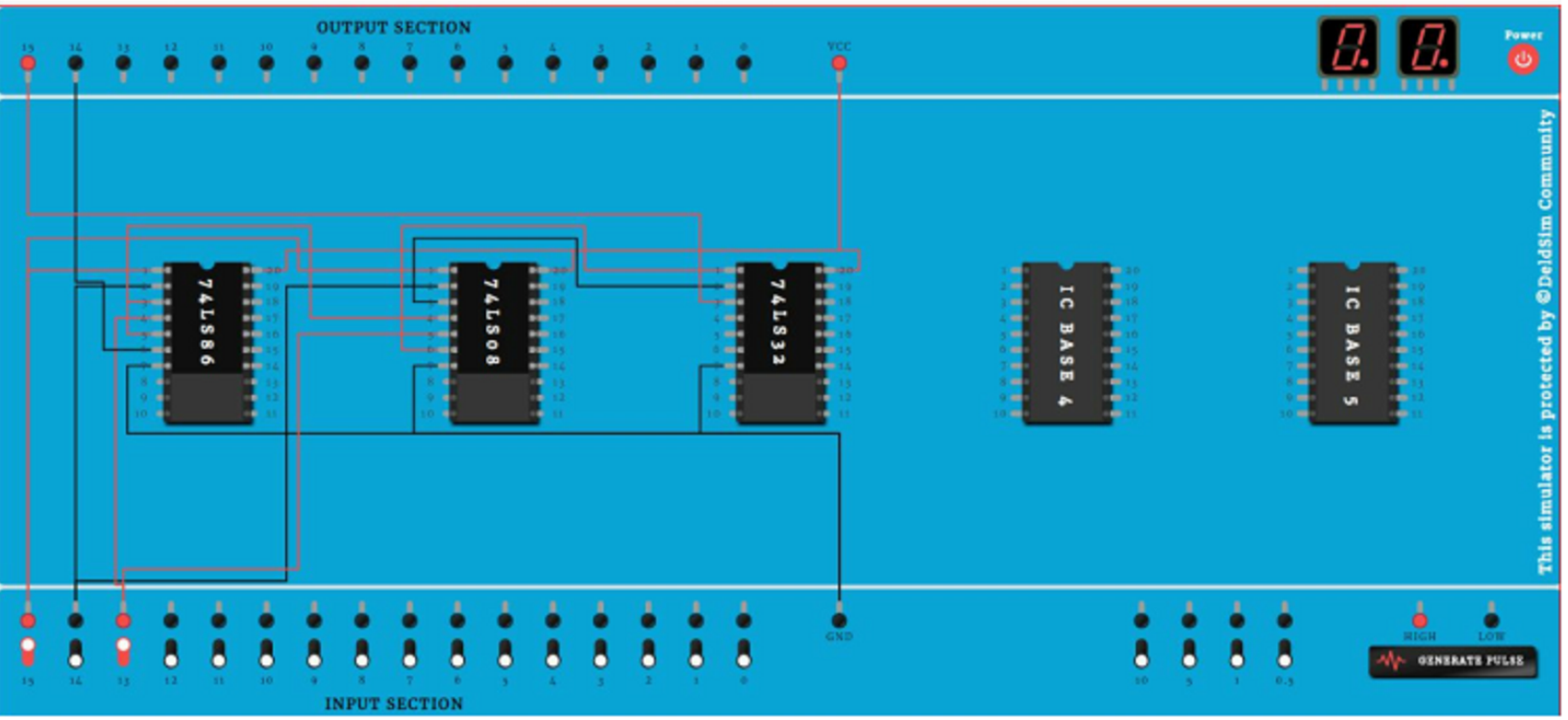
a) Full Adder Using Basic Gates →



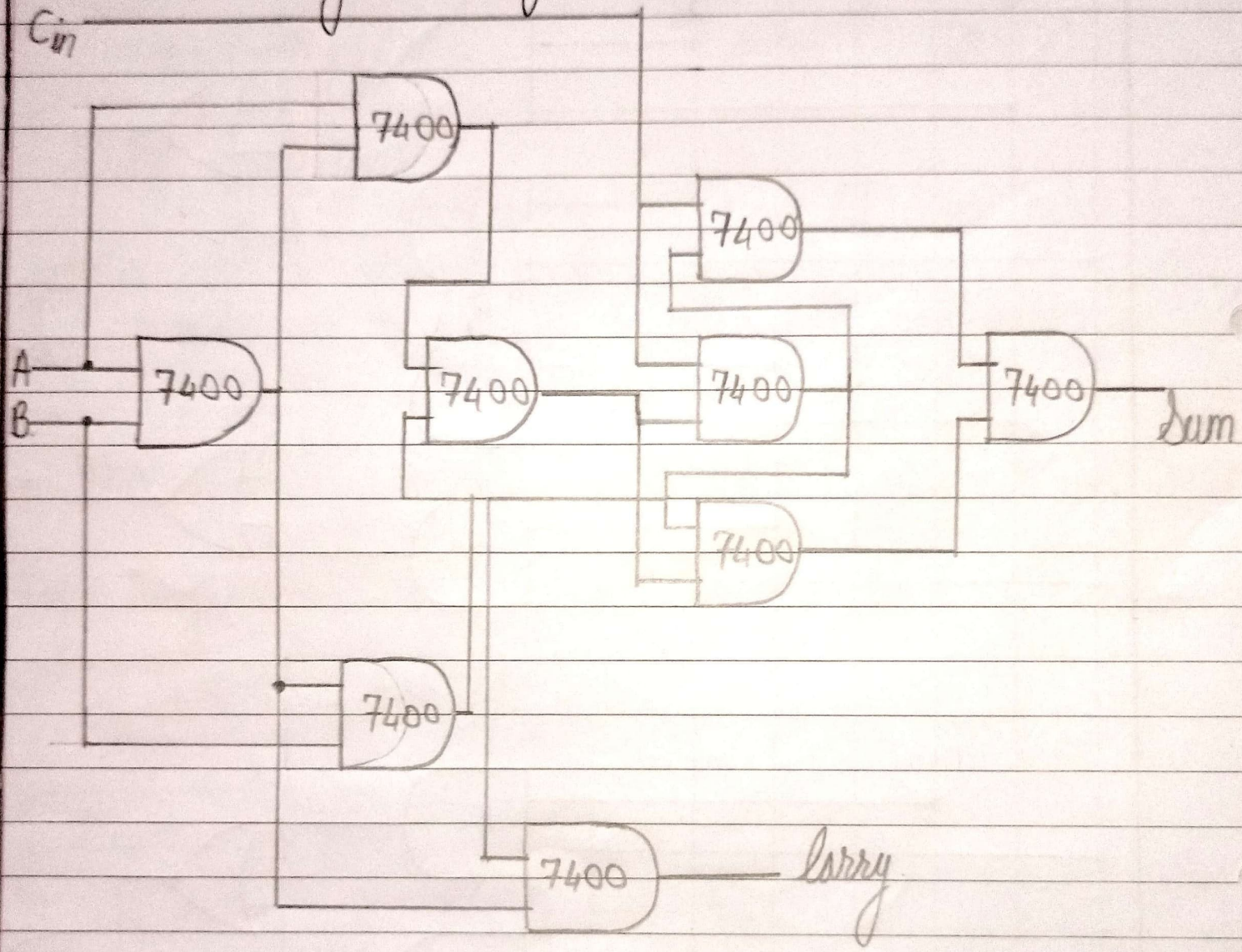
IC 7486 → 2-Input XOR gate, IC 7408 → 2-Input AND gate
IC 7432 → 2-Input OR gate.


x
y
Cin
S
S = A'BC' + AB'C'
+ A'B'C + ABC
C
C = AB + BC + AC.

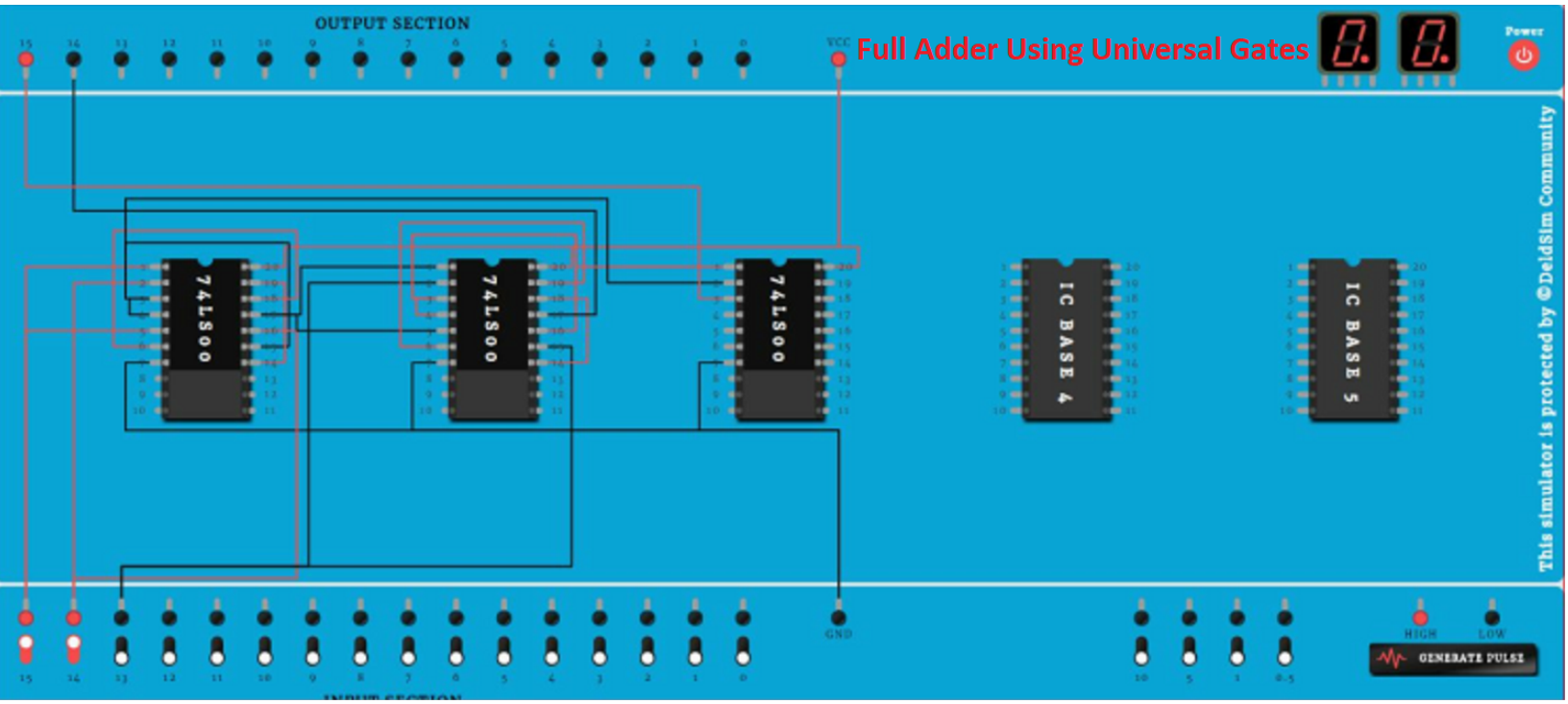
Full Adder Using Basic Gates



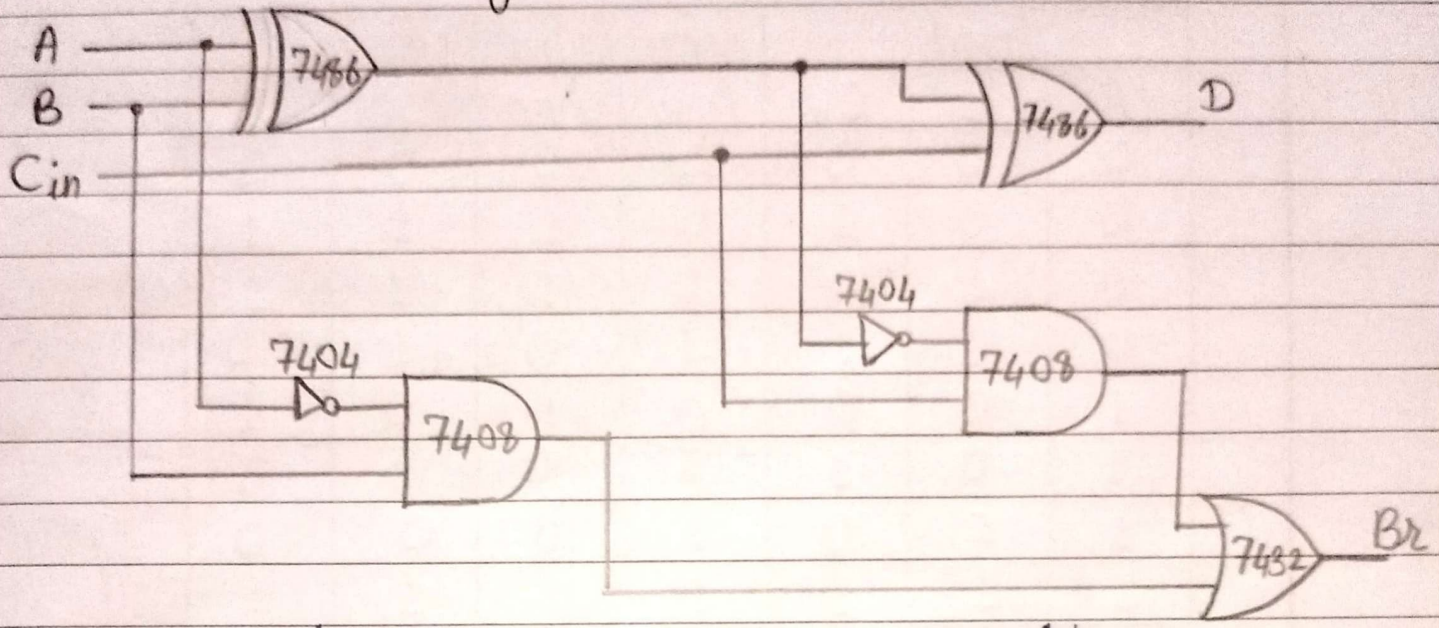
b) Full Adder Using Universal Gates →



IC7400 → 2-Input NAND gate.

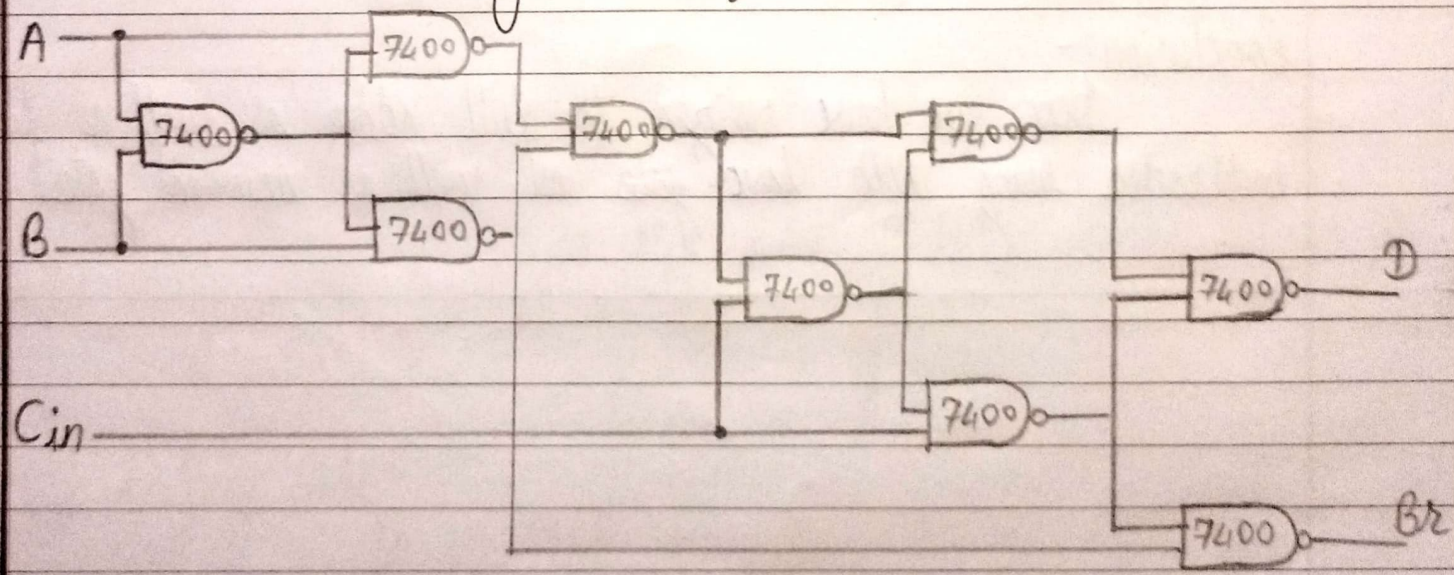

OUTPUT SECTION
Full Adder Using Universal Gates
Power
VCC
GND
74LS00
74LS00
74LS00
IC BASE 4
IC BASE 5
HIGH
LOW
GENERATE PULSE
This simulator is protected by ©DeldSim Community

c) Full Substractor Using Basic Gates →



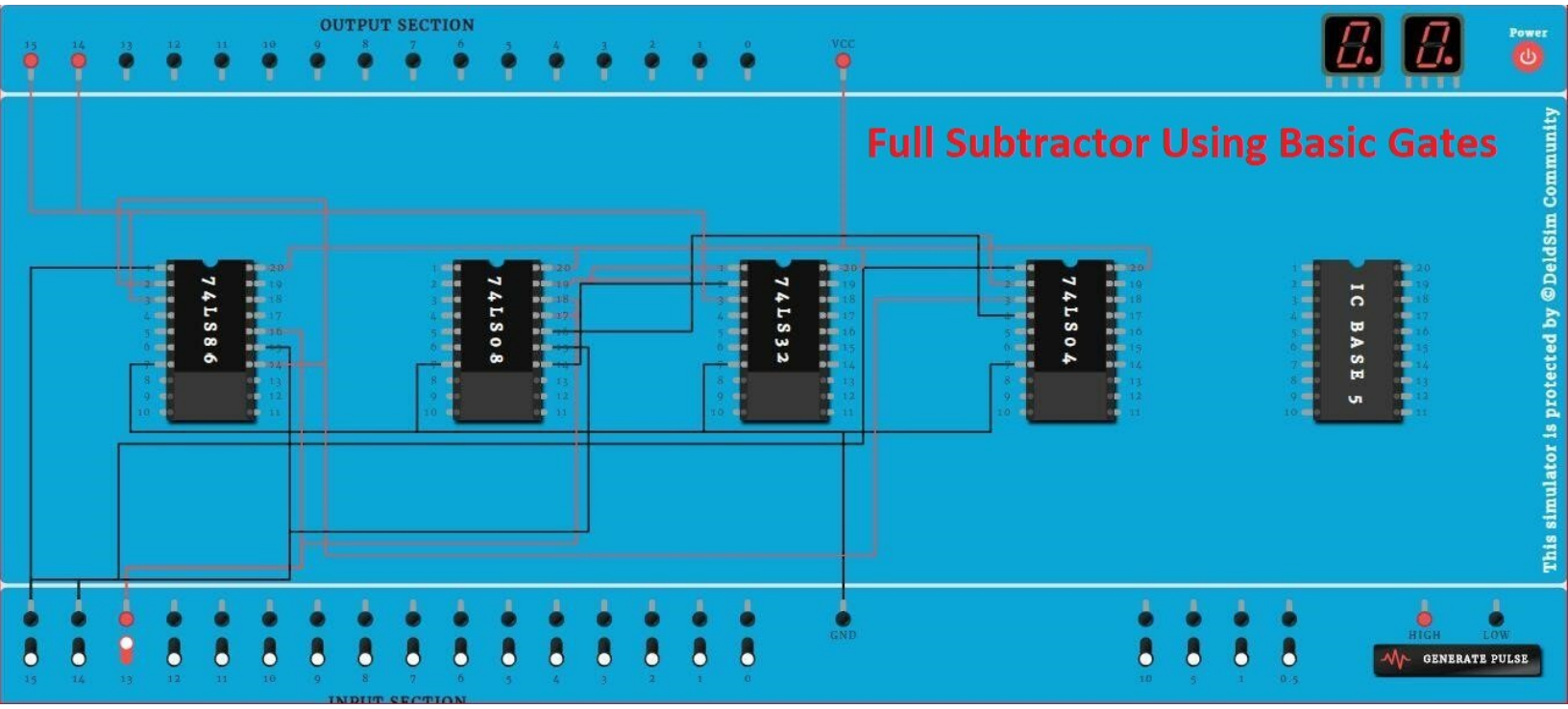
IC7486 → 2 Input XOR gate , IC7404 → NOT Gate
IC7408 → 2-Input AND gate , IC7432 → 2-Input OR gate.

d) Full Substractor Using Universal Gates →

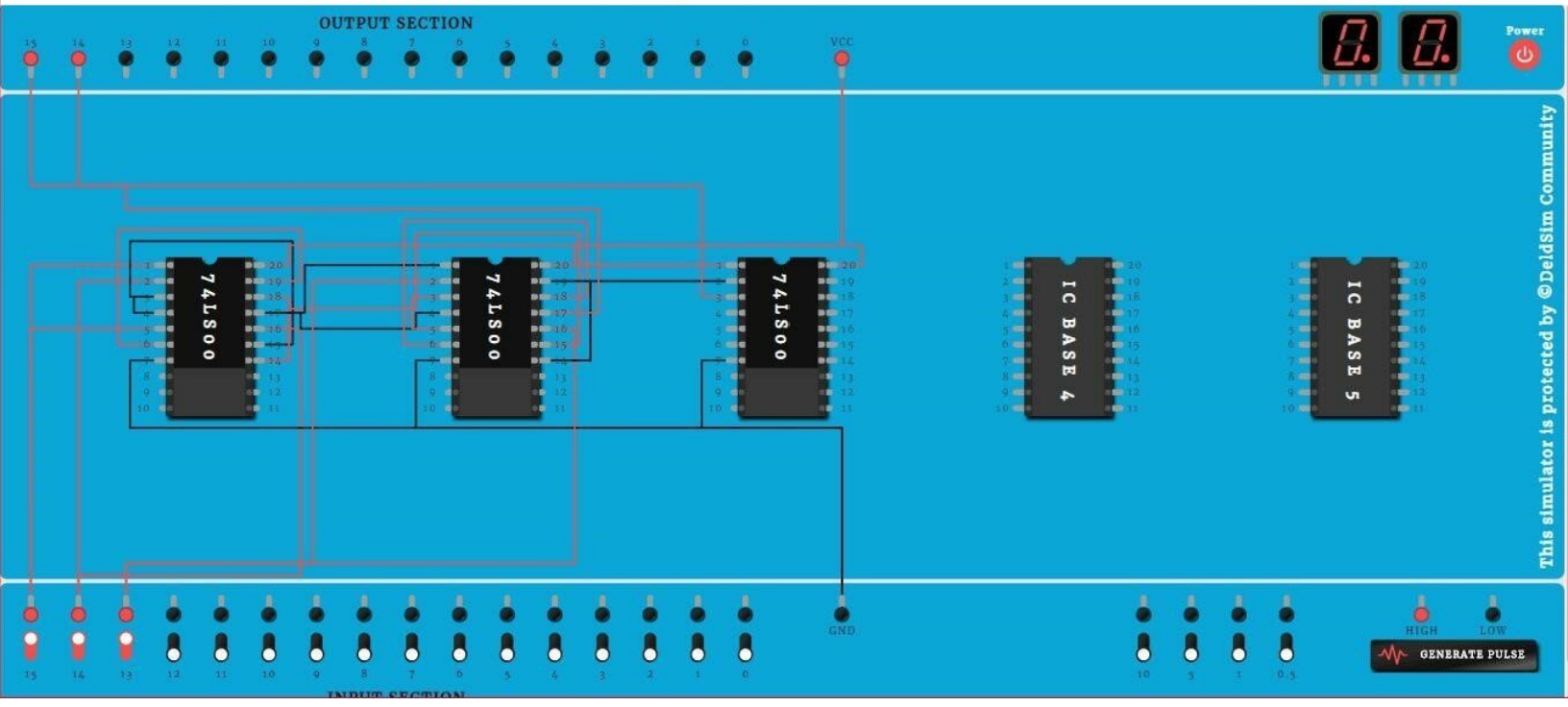


IC7400 → 2-Input NAND gate.

Full Subtractor Using Basic Gates



Full Subtractor Using Universal Gates



Truth Table :-

Input			Adder Output		Subtractor Output	
A	B	Cin	Carry	Sum	Br	D
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	1	1	1
0	1	1	1	0	1	0
1	0	0	0	1	0	1
1	0	1	1	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

Outcomes :-

Thus we have learned how to design and how it works using both basic gates and universal gates.

Conclusion :-

Hence we have realized the full adder as well as full subtractor using both basic gates as well as universal gates.

Digital Electronics and Logic Design (DELD) - Practical Number - 23

Name:- Kaustubh Shrikant Kabra
Class:- Second Year Engineering
Div:- A Roll Number:-
Batch:-
Department:- Computer Engineering Department.
College:- AISSMS's IOIT.

Title:-
Design and Implement code converters a) Binary to Gray
b) BCD to Excess-3.

Aim:-
To Design and understand the code converters:- a) Binary to Gray
b) BCD to Excess-3

Theory:-
a) Binary Number-
A number represented by the digits '0' and '1' is called binary number. Example:- i) 001
ii) 100
iii) 011

b) Gray code-
In gray code, bit patterns for two consecutive numbers differ in only one bit position. It's application is it is use in which the normal sequence of binary number generated by the hardware may produce an error during transition from one number to another.
Example:- 011 (Binary) → 010 (gray)

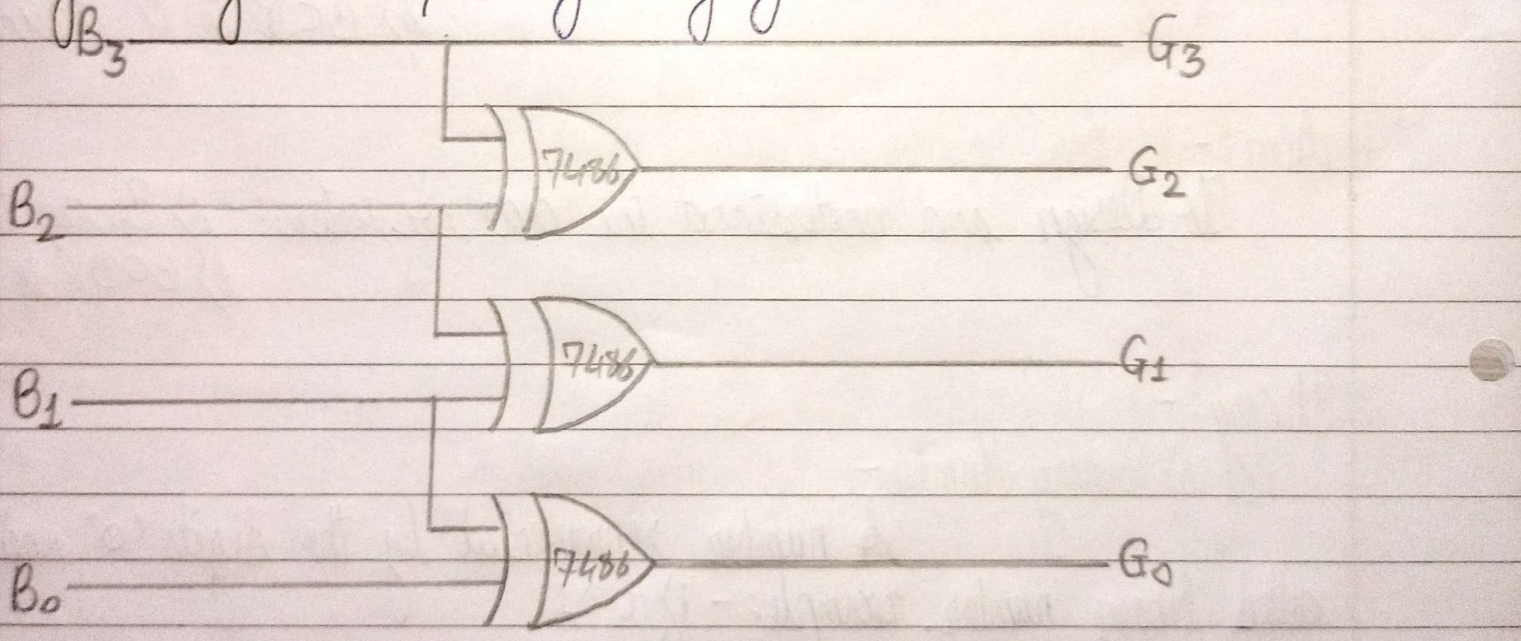
c) Binary coded Decimal Number (BCD) -

BCD is a class of encoding of a decimal number where each digit is represented by a fixed number of bits, usually four or eight. Four bit BCD can be represent 16 numbers i.e. from 0(0000) to 15(1111).

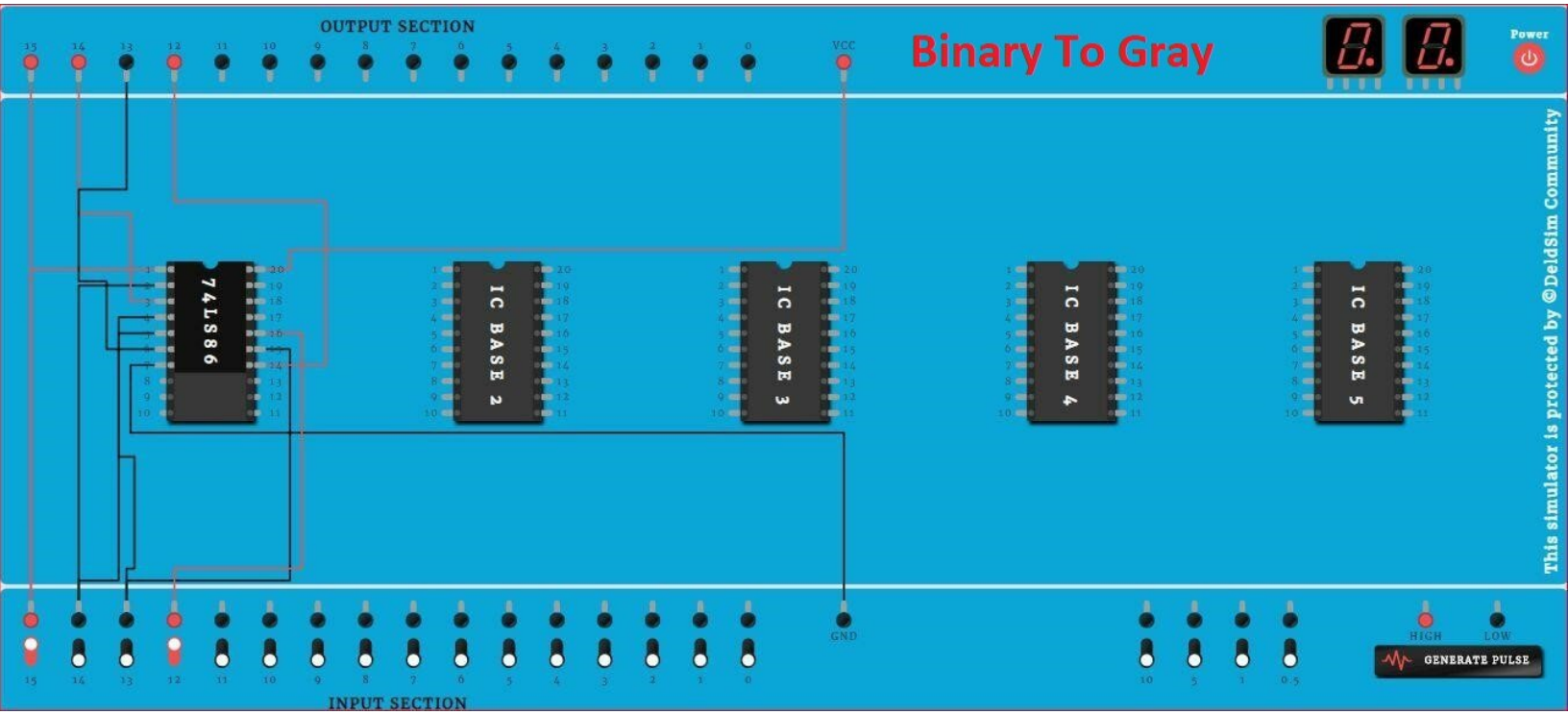
d) Excess - 3 →

It is a non-weighted code used to express code to express decimal number. It is particularly significant for arithmetic operations as it overcomes shortcoming encounters.

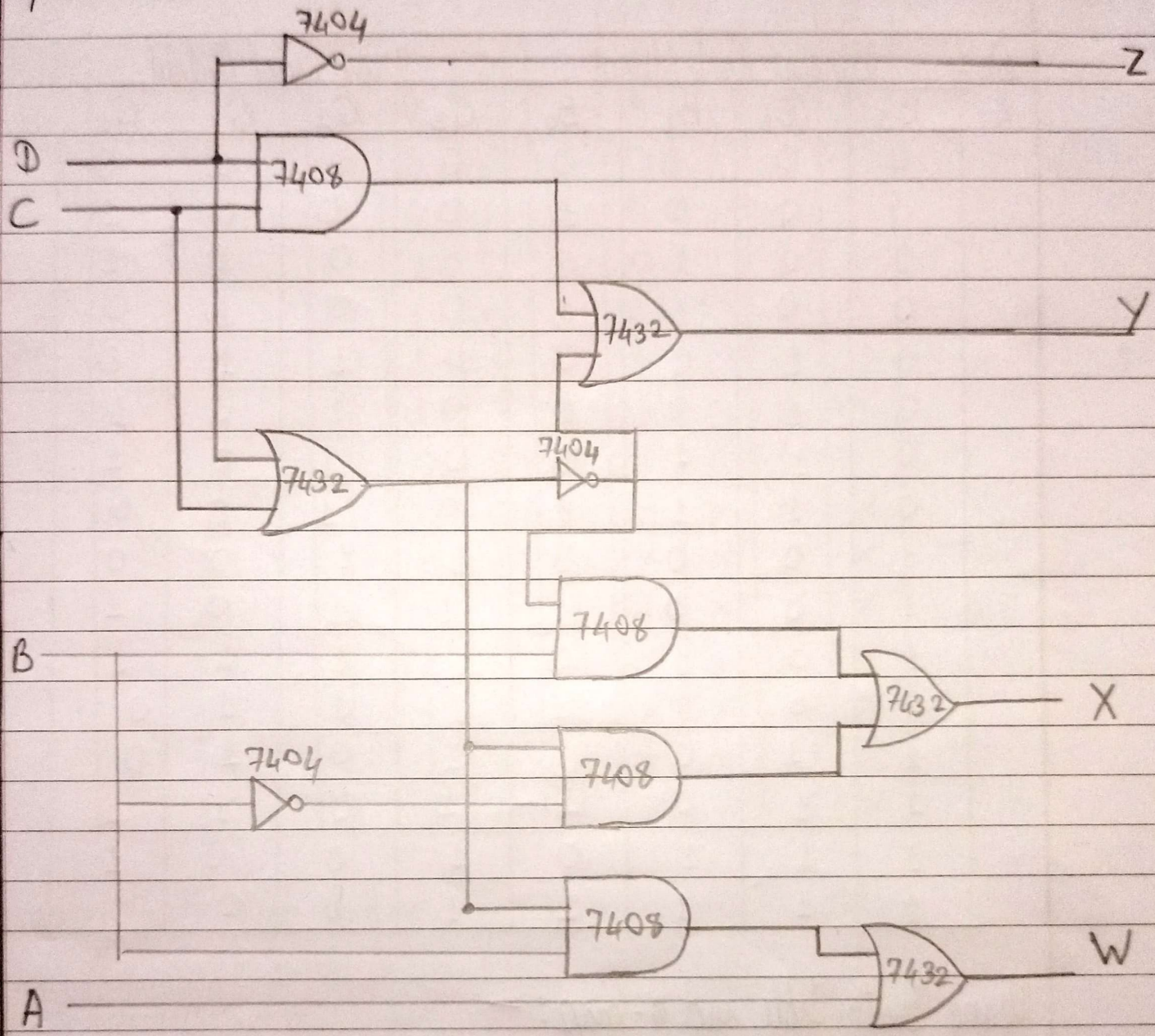
Logic Diagram: → a) Binary to Gray :-



IC 7486 → 2-Input XOR gate.


OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
Binary To Gray
Power
74LS86
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
10 5 1 0.5
INPUT SECTION

b) BCD to Excess 3:-



IC 7404 → NOT gate.
IC 7408 → 2-Input AND gate
IC 7432 → 2-Input OR gate.

OUTPUT SECTION
BCD To Excess 3
Power
VCC
74LS08
74LS32
74LS04
IC BASE 4
IC BASE 5
GND
INPUT SECTION
HIGH
LOW
GENERATE PULSE
This simulator is protected by ©DeldSim Community

Truth Table :-

a) →

Binary Code/Input				Gray Code/Output			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Using Truth Table and K-map :-

$G_3 = B_3$

$G_2 = B_2 \oplus B_3$ / B_2 XOR B_3

$G_1 = B_1 \oplus B_2$ / B_1 XOR B_3.

$G_0 = B_0 \oplus B_1$ / B_0 XOR B_1.

b) →

Decimal	BCD				Excess-3			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	x	x	x	x
10	1	0	1	0	x	x	x	x
11	1	0	1	1	x	x	x	x
12	1	1	0	0	x	x	x	x
13	1	1	0	1	x	x	x	x
14	1	1	1	0	x	x	x	x
15	1	1	1	1	x	x	x	x

Using Truth table and K-map: -

$W = A + BC + BD$

$X = B'C + B'D + BC'D'$

$Y = CD + C'D'$

$Z = D'$

Outcome :-

From the experiment we learnt to design and analyze binary to Gray and BCD to Excess-3 code convertor.

Conclusion :-

Hence, we have designed and implemented i) binary to gray and ii) BCD to Excess-3.

* Digital Electronics and Logic Design (DELD) - Practical Number - 4

Name:- Kaustubh Shrikant Kabra.
Class:- Second Year Engineering.
Div:- A Roll Number:-
Batch:-
Department:- Computer Department.
College:- AISSMS's IOIT.

Title:-
4-bit Binary Adder (IC-7483).

Aim:-
Design and Realization of BCD Adder using 4-bit binary Adder (IC-7483).

Objective:-
Student will be able to realize the BCD adder using binary adder and logic gates.

Theory:-

Adder:-
An adder is digital circuit that perform addition of number. In many computer and other kind of processor adder are used in arithmetic logic Unit (ALU).

Binary Adder:-
Binary Adder are arithmetic circuits in the form of half-adder and full adder used to add together two binary digits.
It is very common combinational logic circuit which can be constructed

using logic circuit just a few basic logic gates allowing it to add together two or more binary number is the Binary Adder.

BCD Adder:-

A BCD Adder is a circuit that add two BCD digits and produce a sum digit also in BCD.

To implement BCD Adder we require:- ① 4-bit binary adder, 2 if sum greater than 9.
② Logic circuit to detect sum greater than 9.

Truth Table:-

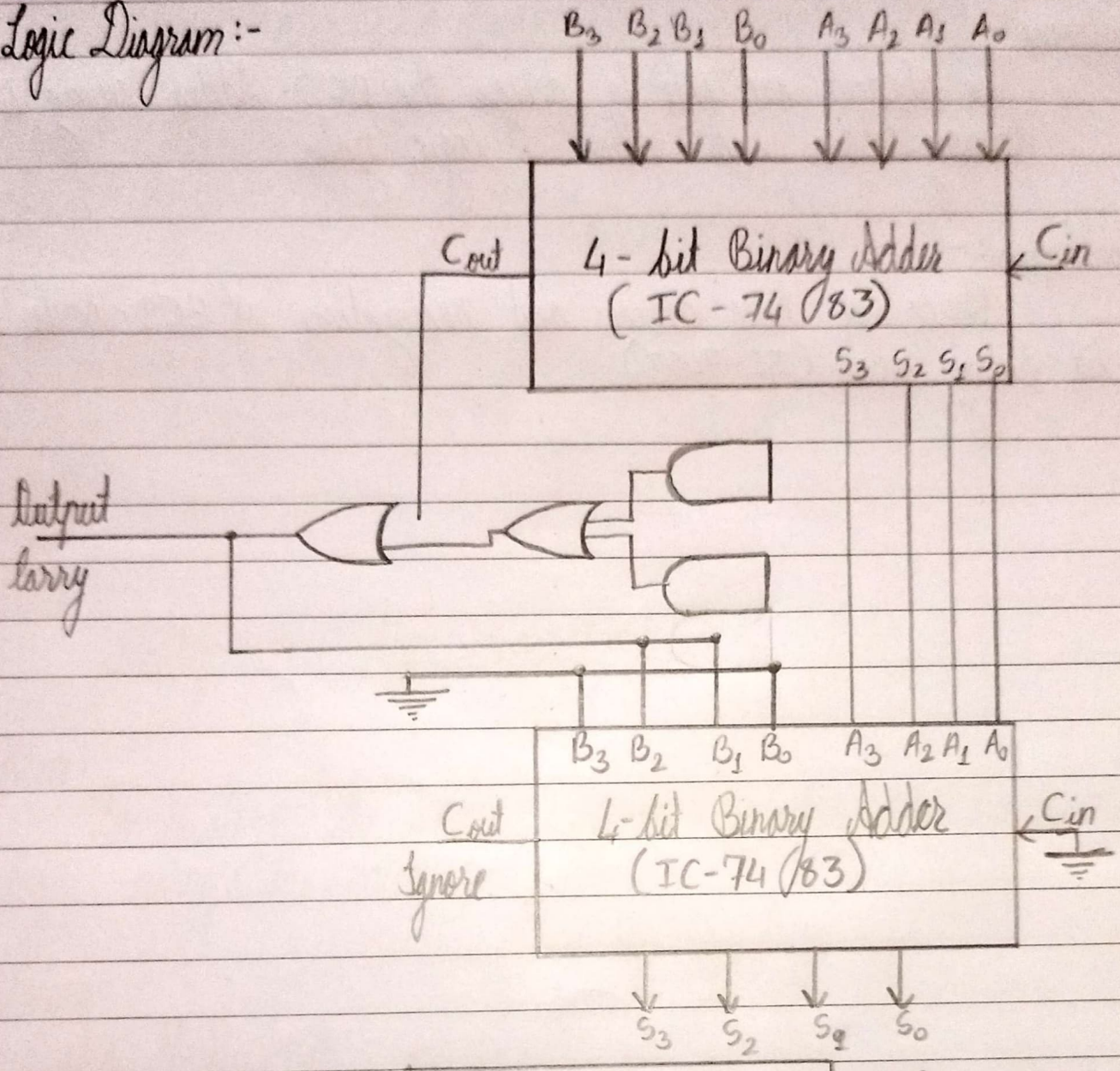
Input				Output
A (S_3)	B (S_2)	C (S_1)	D (S_0)	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Simplification of the truth table using K-map.

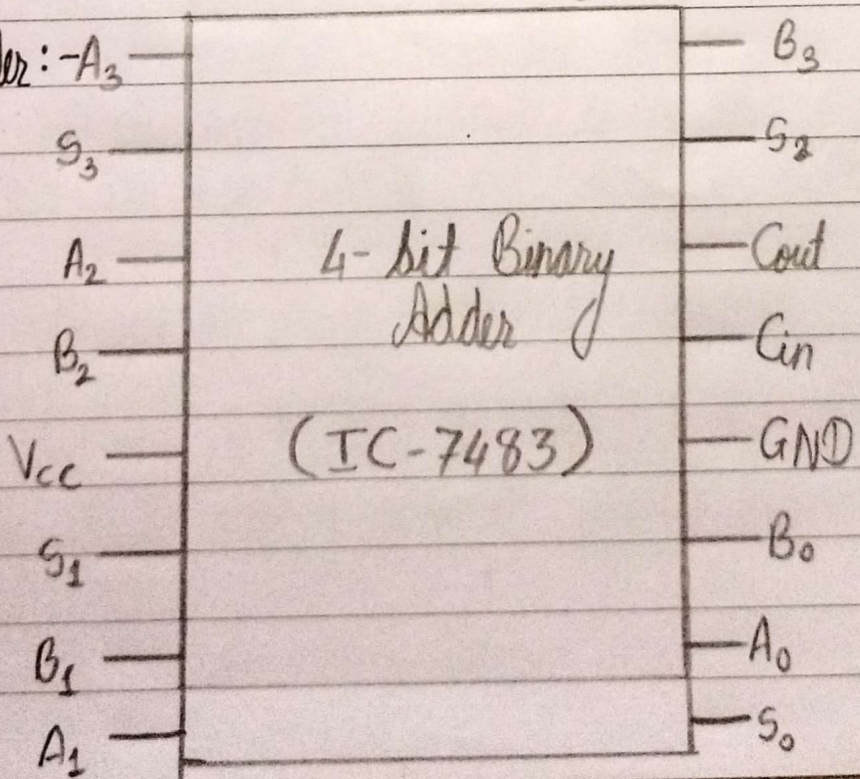
S_3S_2 \ S_1S_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

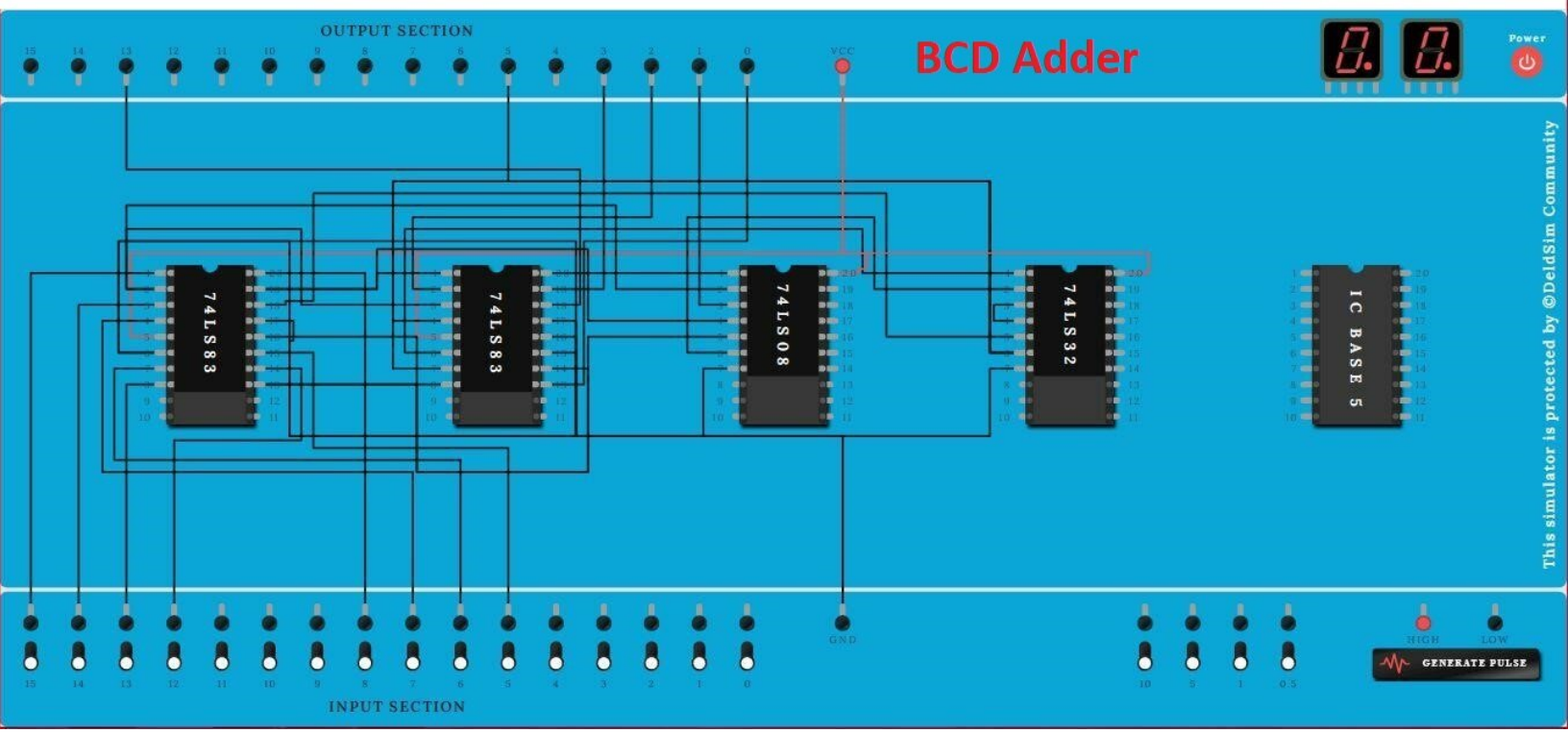
$$Y = S_3S_2 + S_2S_1$$

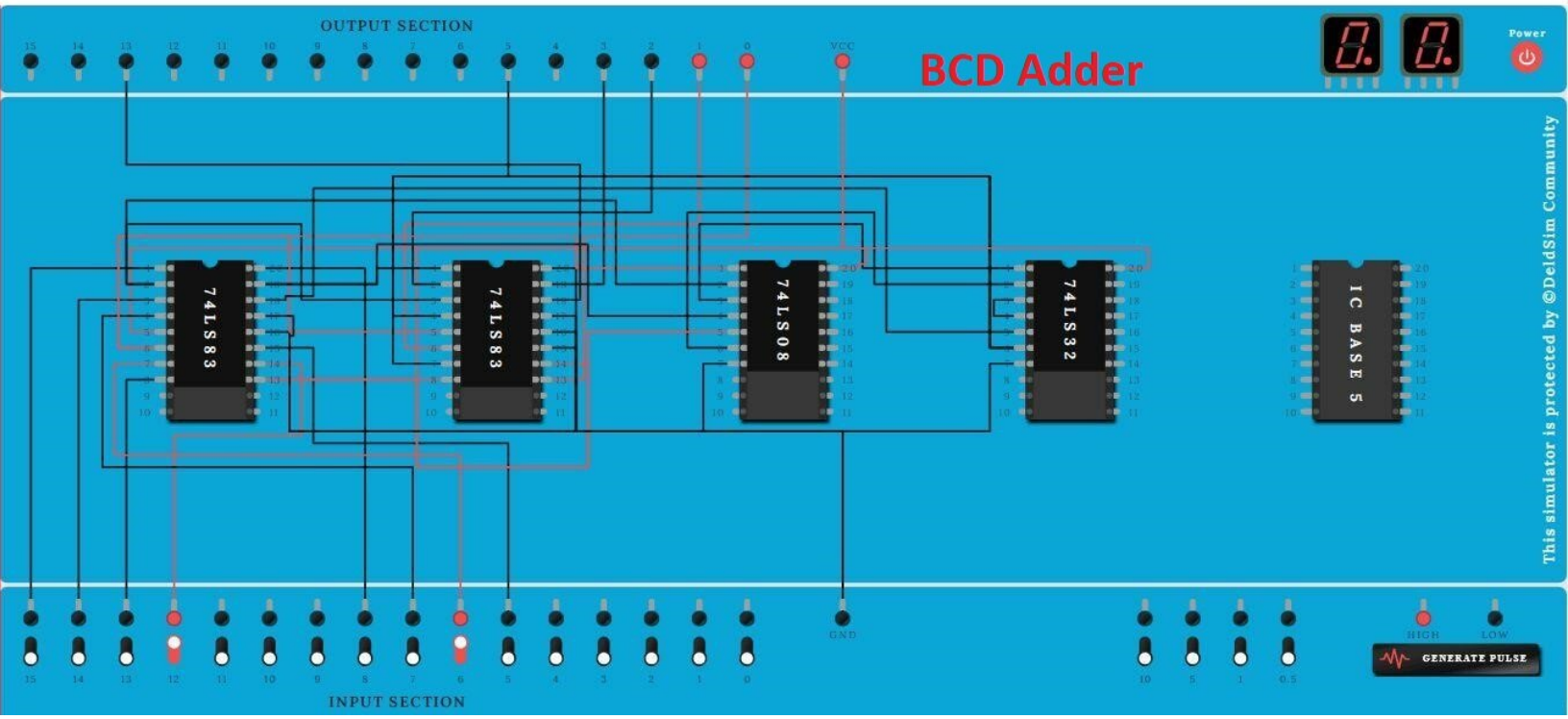
Logic Diagram :-

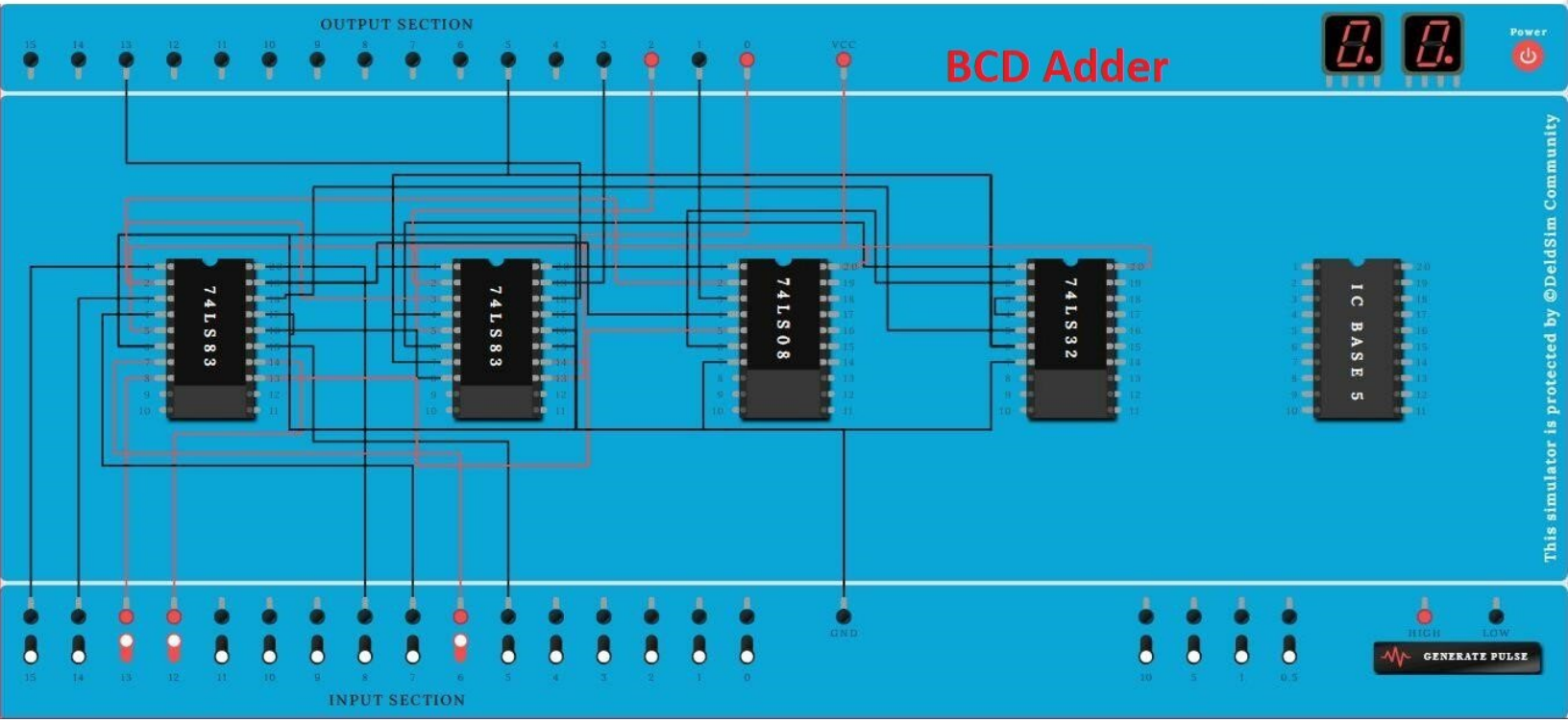


4-bit Binary Adder :-




OUTPUT SECTION
BCD Adder
Power
VCC
74LS83
74LS83
74LS08
74LS32
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
BCD Adder
Power
74LS83
74LS83
74LS08
74LS32
IC BASE 5
VCC
GND
This simulator is protected by ©DeldSim Community
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
BCD Adder
Power
74LS83
74LS83
74LS08
74LS32
IC BASE 5
VCC
GND
This simulator is protected by ©DeldSim Community
HIGH
LOW
GENERATE PULSE
INPUT SECTION

Outcome :-

The student are able to design the BCD-Adder using 4-bit Binary Adder and other logic gates.

Conclusion :-

Hence we have design and realization of BCD-Adder using 4-bit binary adder (IC-7483).

Digital Electronics and Logic Design (DELD) - Practical Number - 05.

Name:- Kaustubh Shrikant Kabra
Class:- Second Year Engineering.
Div:- A Roll Number:-
Batch:-
Department:- Computer Engineering Department
College:- AISSMS's IOIT

Title:-
Multiplexer And Demultiplexer.

Aim:-
To realize the boolean expression for suitable combinational logic using the mux and demux.

Theory:-
a) Multiplexer:-
A multiplexer is also known as data selector, it is a device that selects between several analog or digital input that has select lines. The select lines are used to select the line to send the output.
In this experiment we are using Mux 74151/74153.

b) De-Multiplexer:-
A Demux is a device that takes a single input line and routes it to one of several digital output lines. Also called as Data Distributor.
In this experiment we are using Demux 74154/74138.

Experiment:-

a) Solve the boolean equation using 4:1 multiplexer.

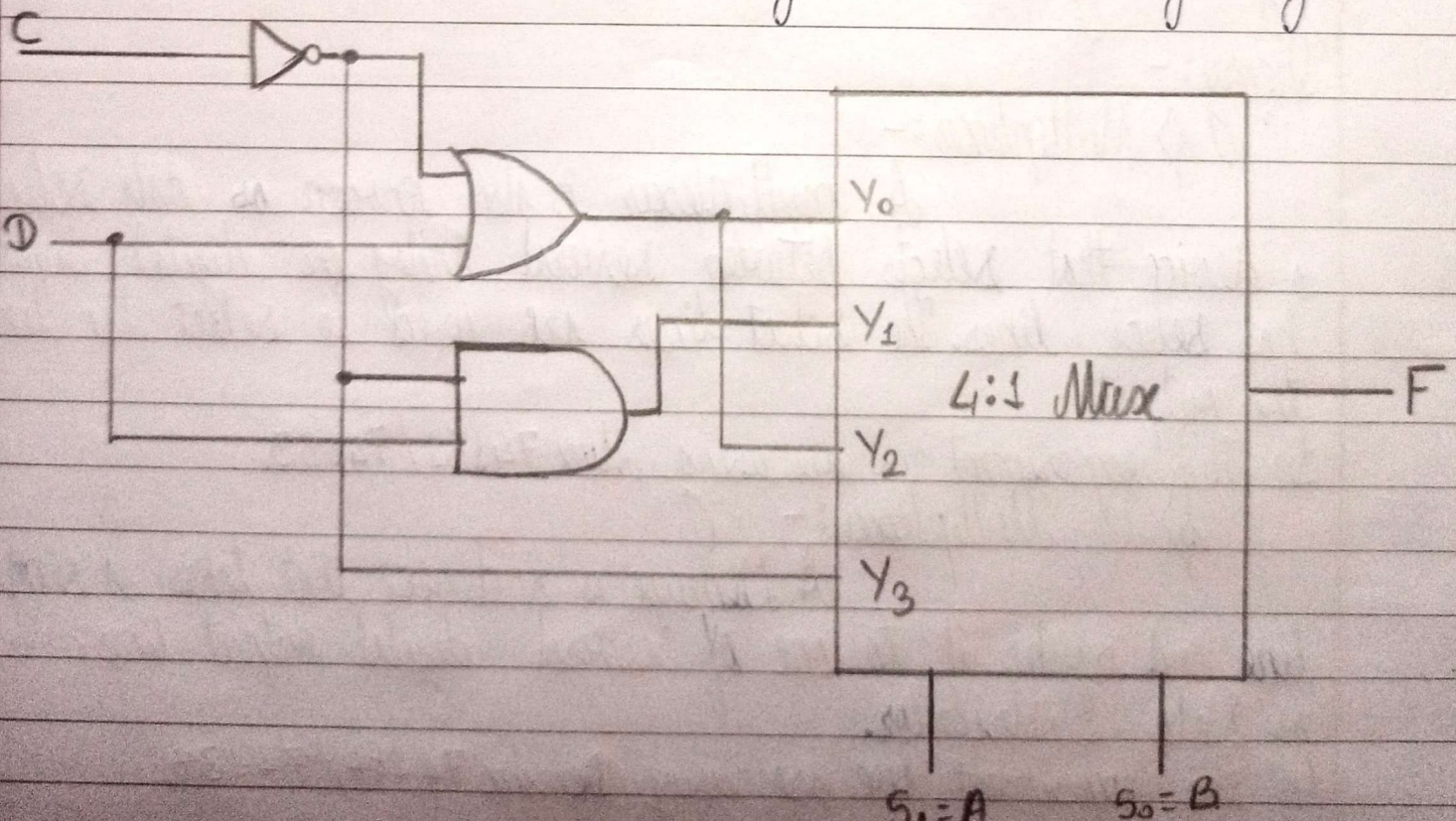
$F(A, B, C, D) = \Sigma m(1, 4, 5, 7, 9, 12, 13)$.

Using K-map :-

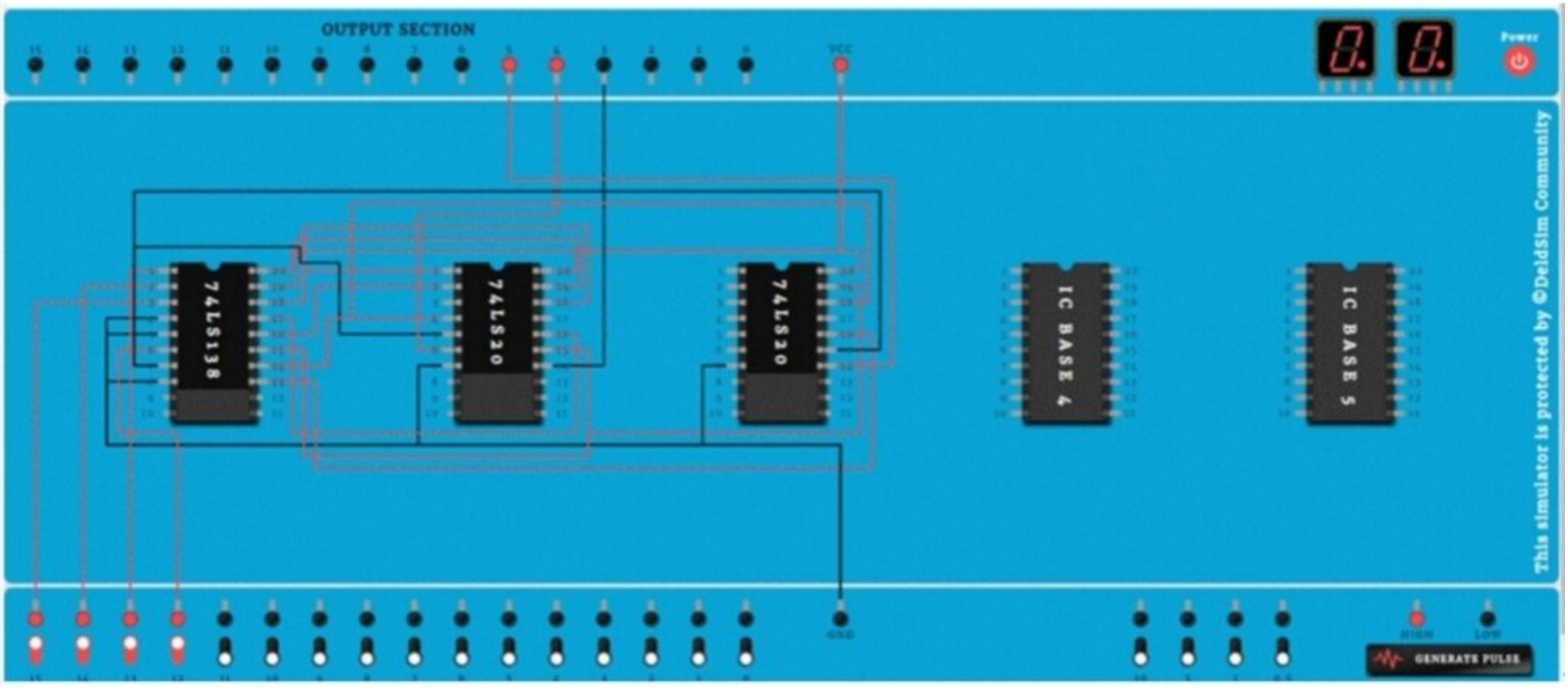
CD \ AB	00	01	11	10
00		1	1	
01	1	1	1	1
11		1		
10				

S_1	S_0	Y	
0	0	C'D	Y_0
0	1	C'+D	Y_1
1	0	C'D	Y_2
1	1	C'	Y_3

Using the truth table logic diagram is:-

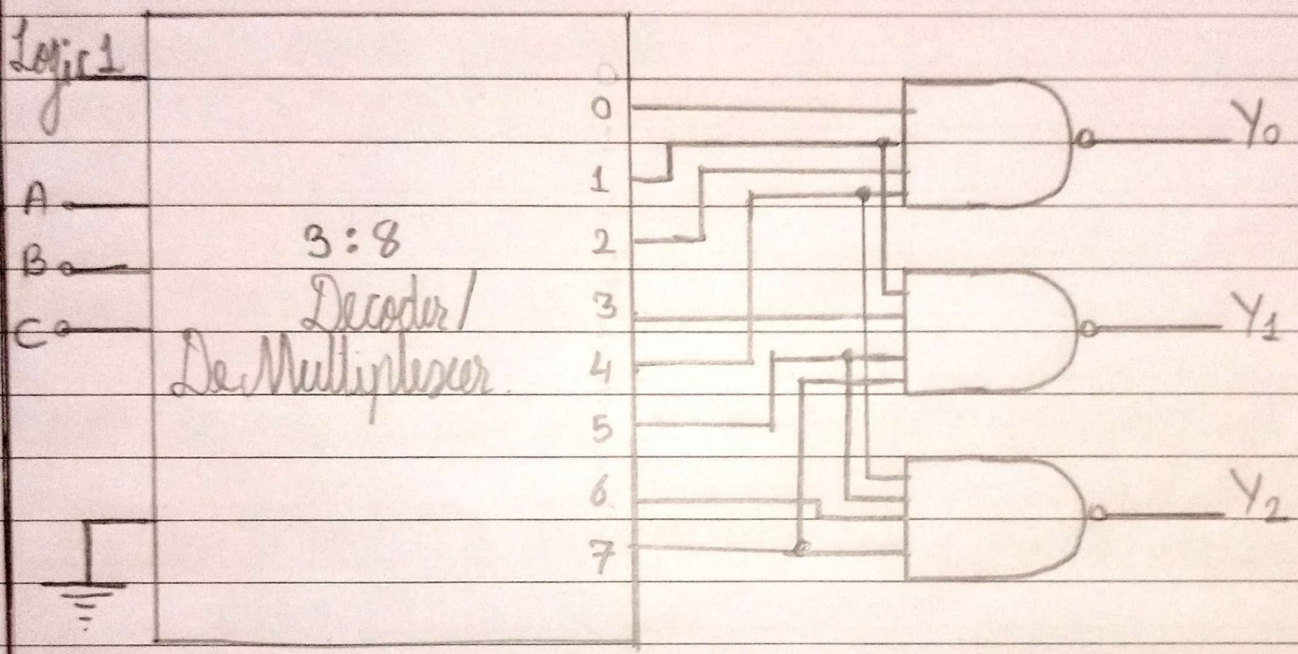


Mux and Demux



6) Implement the following function using 3:8 line decoder.

$Y_0(A,B,C) = \Sigma m(0,1,2,4)$
$Y_1(A,B,C) = \Sigma m(1,3,5,7)$
$Y_2(A,B,C) = \Sigma m(4,5,6,7)$



Outcomes:-

The student are able to realize the boolean equation using logic circuit mux and demux.

Conclusion:-

Hence, we have realized the expression for suitable combinational circuit, i.e. Mux and De Mux.

Digital Electronics and Logic Design (DELD) - Practical Number - 6

Name :- Kaustubh Shrikant Kabra
Class :- Second Year Engineering
Div :- A Roll Number :-
Batch :-
Department :- Computer Department
College :- AISSMS's IOIT.

Title :- Comparators.

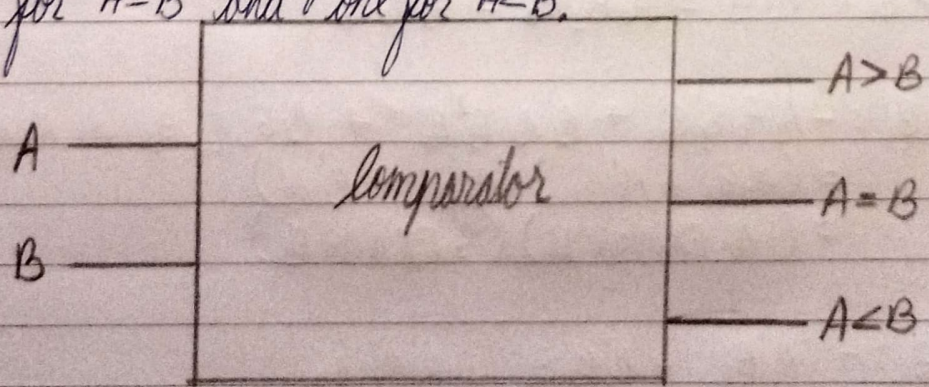
Aim :- To verify the truth-table of two bit comparators using logic gates.

Objective :- To study and verify the truth table of two bit comparators using logic gates.

Theory :-

A digital comparator is a combinational circuit that compares two digital or binary number in order to find out whether one binary number is equal, less than or greater than other binary number.

We logically design a circuit for which we will have two inputs one for A and other for B and have three output terminal. One for A > B, one for A = B and one for A < B.



Truth Table :-

Input				Output		
A_1	A_0	B_1	B_0	$A<B$	$A=B$	$A>B$
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

$A > B :- A_1 B_1' + A_0 B_1' B_0' + A_1 A_0 B_0'$

$A = B :- A_1' A_0 B_1' B_0' + A_1' A_0 B_1' B_0 + A_1 B_0 A_0 B_1 + A_1 A_0' B_1 B_0'$

$= (A_0 B_0 + A_0' B_0')(A_1 B_1 + A_1' B_1')$

$= (A_0 \text{ EX-NOR } B_0)(A_1 \text{ EX-NOR } B_1)$.

$A < B :- A_1' B_1 + A_0' B_1 B_0 + A_1' A_0' B_0$.

Logic Diagram :-

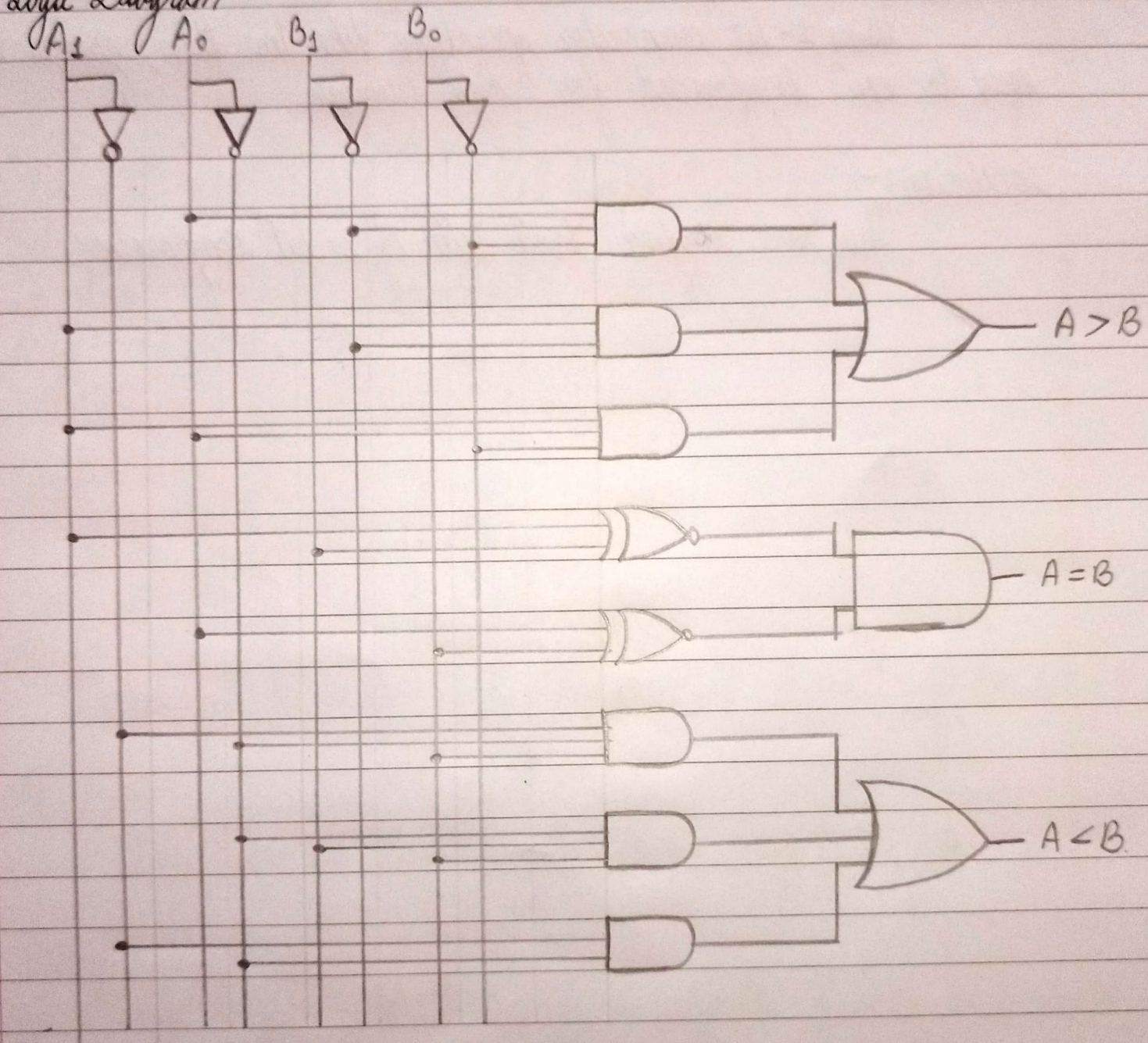
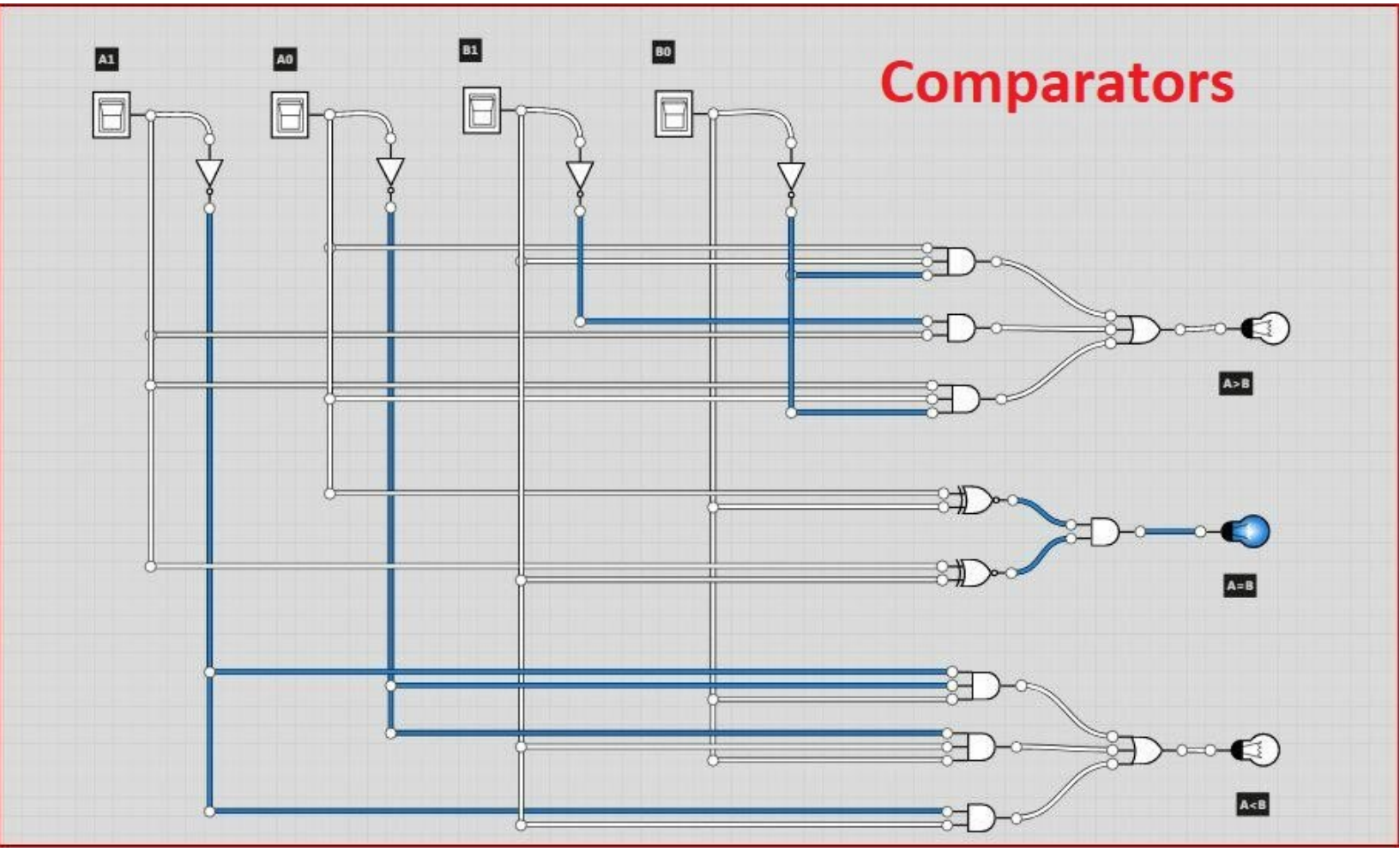
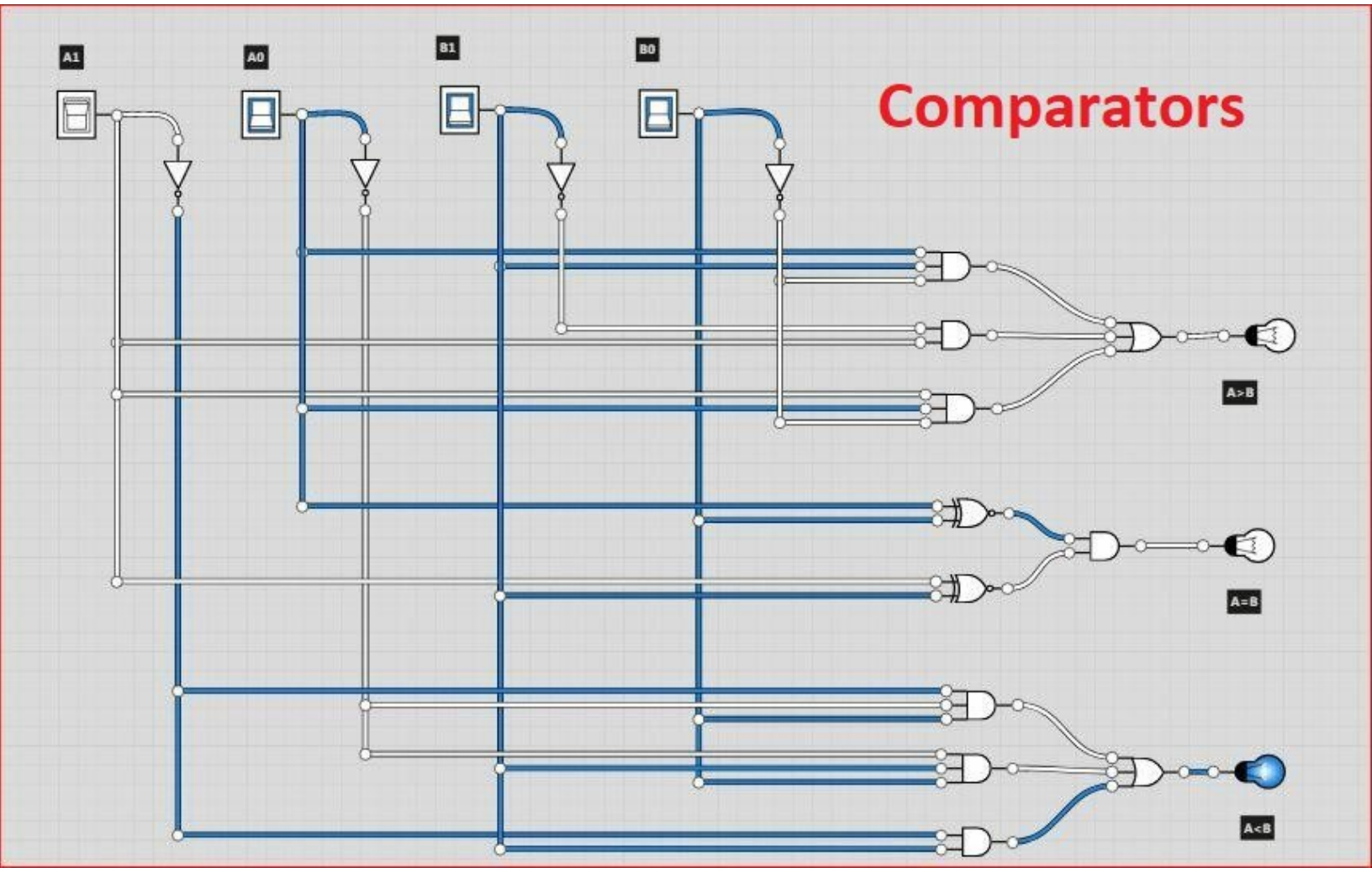
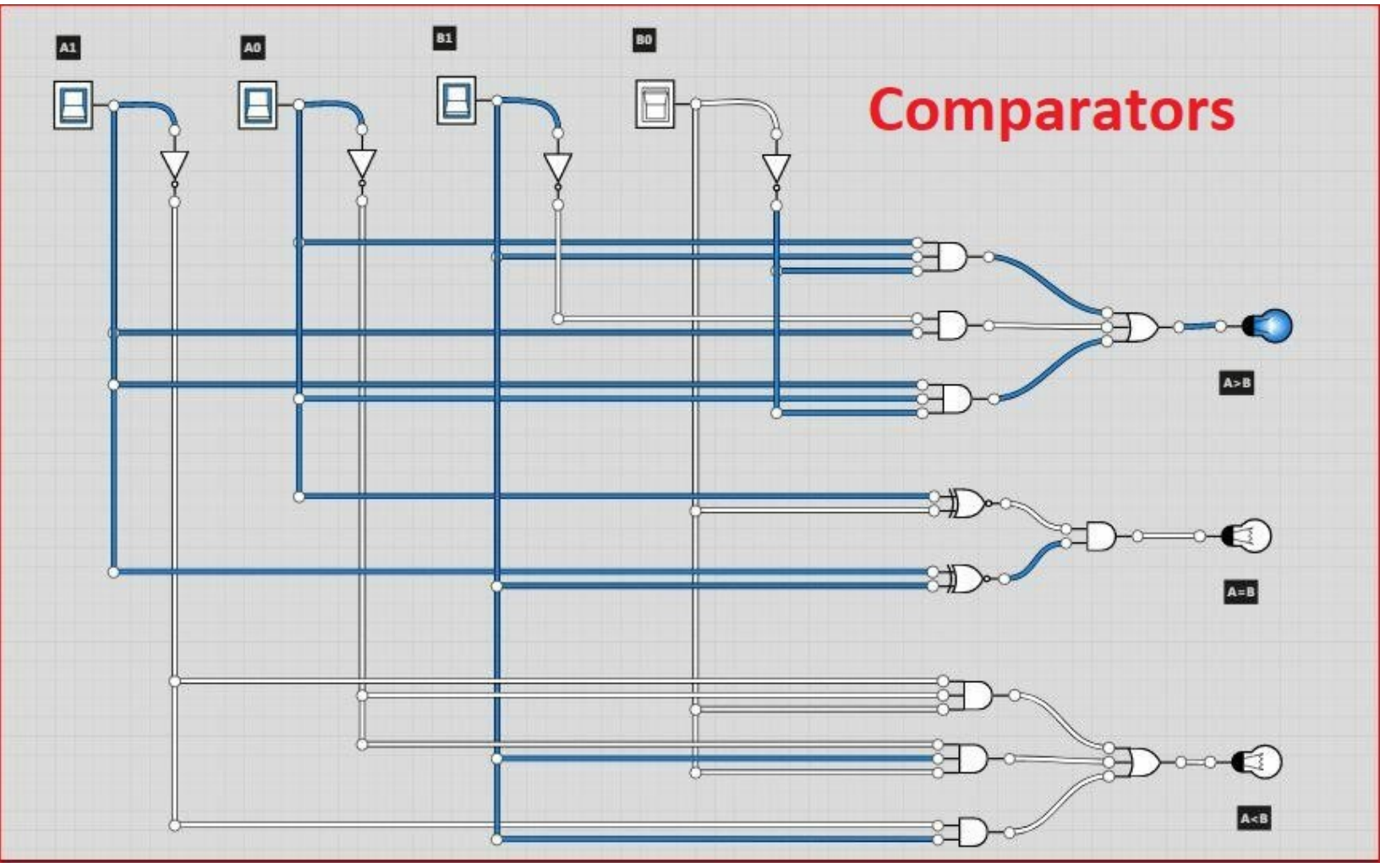


fig :- 2-bit comparator.


Comparators
A1
A0
B1
B0
A>B
A=B
A<B


A1
A0
B1
B0
Comparators
A>B
A=B
A<B


A1
A0
B1
B0
Comparators
A>B
A=B
A<B

Outcomes:-

Using 2-bit comparator operations like less than, greater than and equal to can be performed for 2 digit number.

Conclusion:-

Thus we verified truth table of 2-bit comparators.

* Digital Electronics and Logic Design (DELD) - Practical Number - 7

Name:- Kaustubh Shrikant Kabra
Class:- Second Year Engineering
Div:- A Roll Number:-
Batch:-
Department:- Computer Department
College:- AISSMS's IOIT

Title:-
Parity Generator and Checker.

Aim:-
Design and Implement parity generator and checker using EX-OR gate.

Objective:-
Learn even/odd parity generator/checker using EX-OR gate.

Theory:-
Parity - A term used to specify the member of one's in a digital word as odd/even.

Even Parity Generator -
Will produce a logic 1 at its output if the data word contain an odd number of ones. If the data contain an even number of ones then the output of the parity generator will be low.

Parity Checker -
At the reciving end a logic circuit is used to check the parity

of receiving information, and determines whether error is included in the message or not.

Even bit Parity code-
Total number of ones in parity code is even.

Odd bit Parity code-
Total number of one's in parity code word is odd.

Truth Table :-

Ⓐ Parity Generator :-

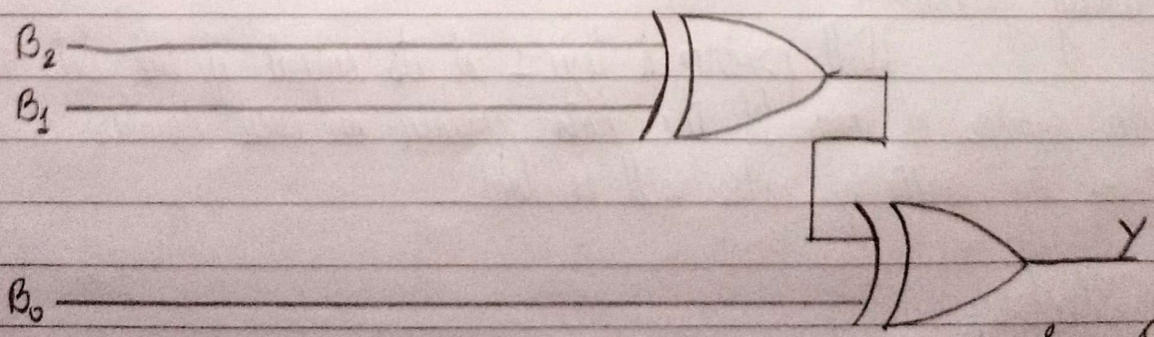
① Even Parity Generator -

Input			Output
B_2	B_1	B_0	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Expression for output using k-map

B_2 \ B_1B_0	00	01	11	10
0	0	[1]	0	[1]
1	[1]	0	[1]	0

$f = B_2 \text{ (EX-OR) } B_1 \text{ (EX-OR) } B_0$



- Even Parity Generator

OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
Even Parity Generator
Power
74LS86
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
10 5 1 0.5
INPUT SECTION

② Odd Parity Generator :-

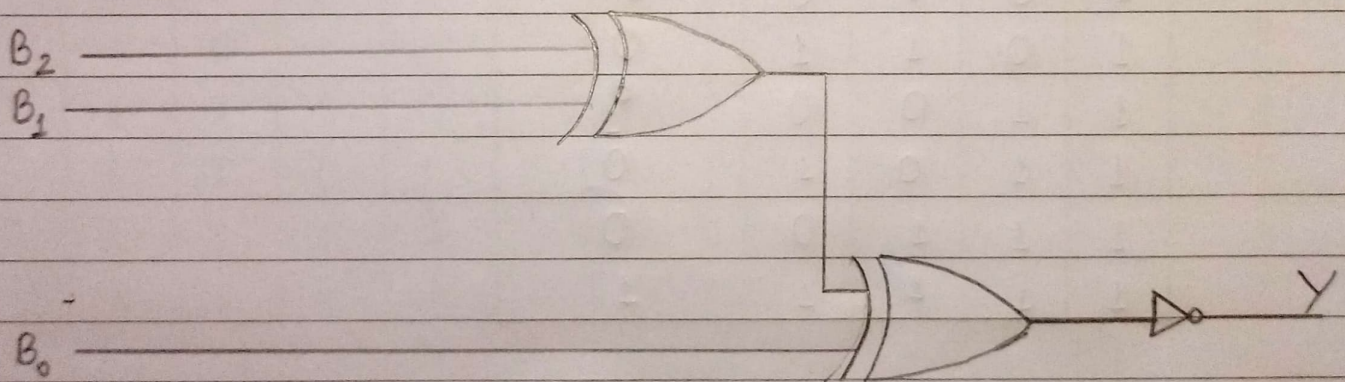
Input			Output
B_2	B_1	B_0	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Expression using k-map.

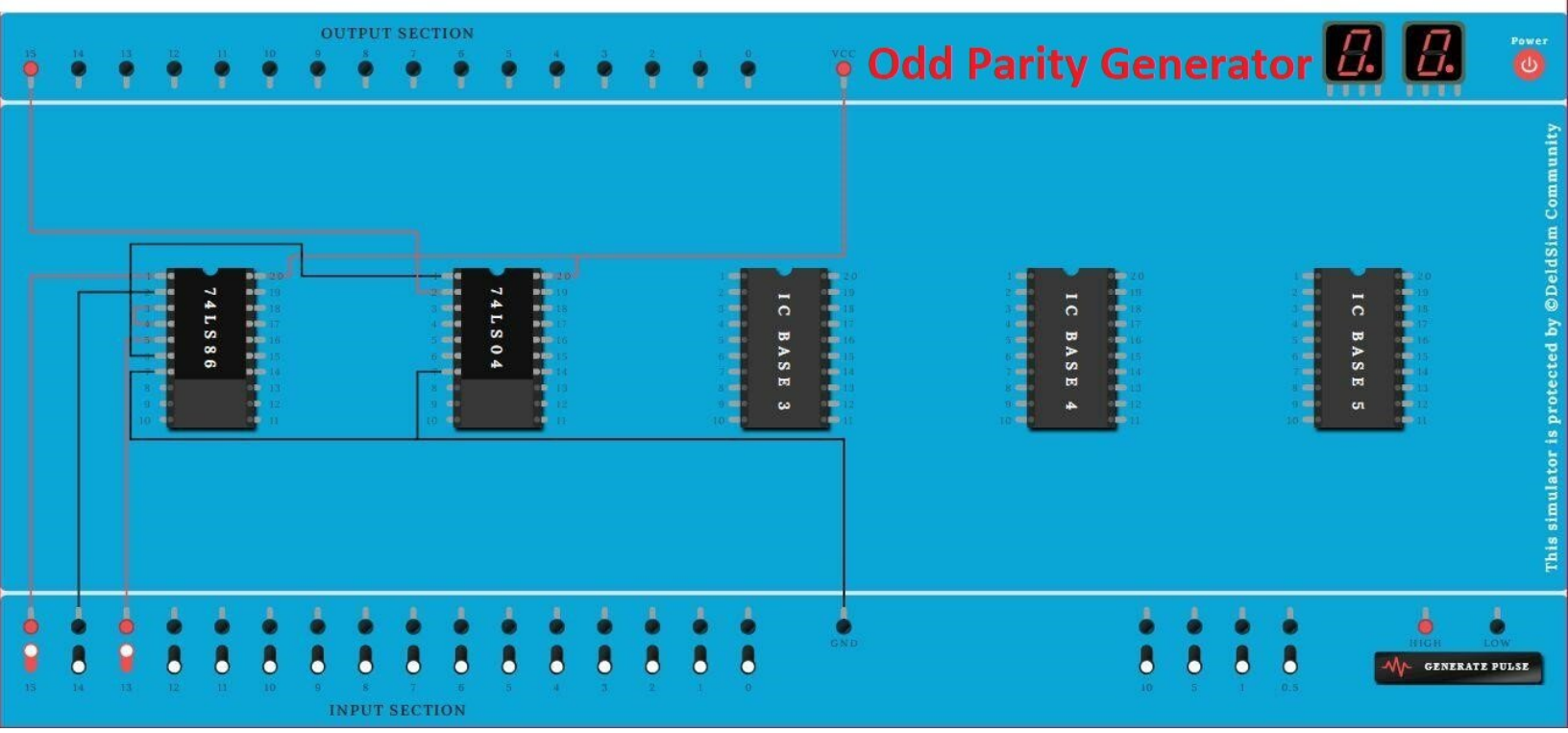
B_2 \ B_1B_0	00	01	11	10
0	1	0	1	0
1	0	1	0	1

$$Y = B_2 (EX\text{-}NOR) B_1 (EX\text{-}NOR) B_0$$

$$= B_2 (EX\text{-}OR) B_1 (EX\text{-}OR) B_0$$



- Odd Parity Generator


OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
Odd Parity Generator
Power
74LS86
74LS04
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
10 5 1 0.5
INPUT SECTION

(B) Parity Checker :-

[0 → Error
1 → Not Error]

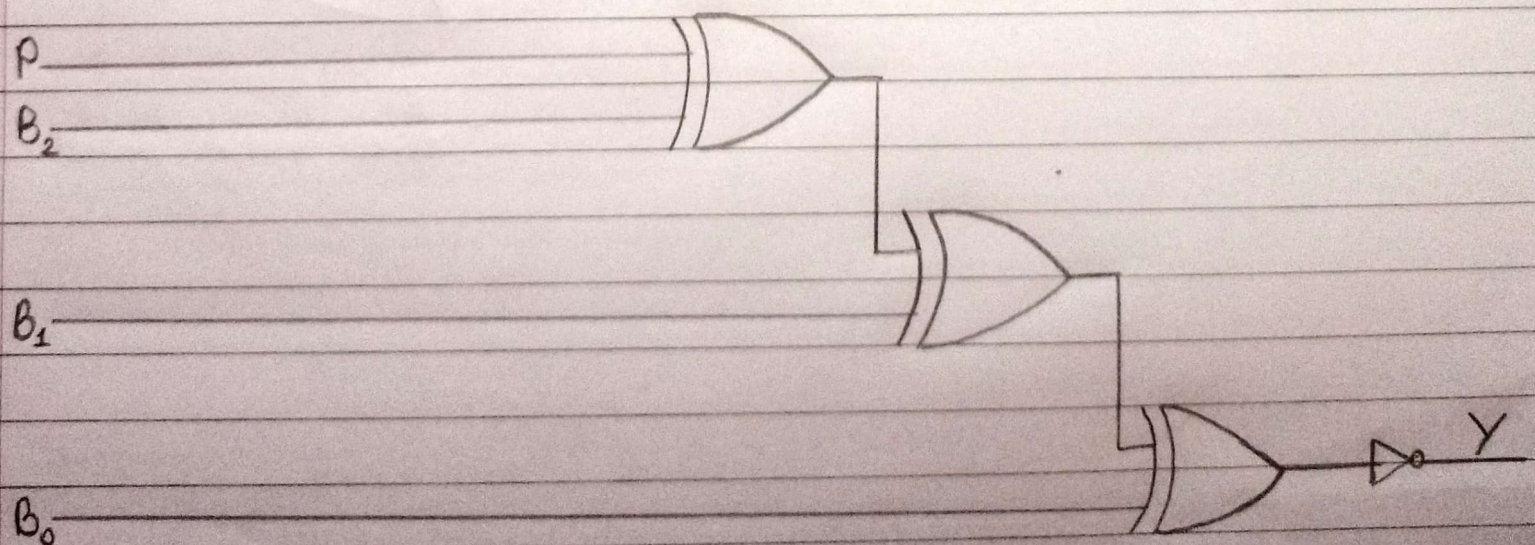
(1) Odd Parity Checker :-

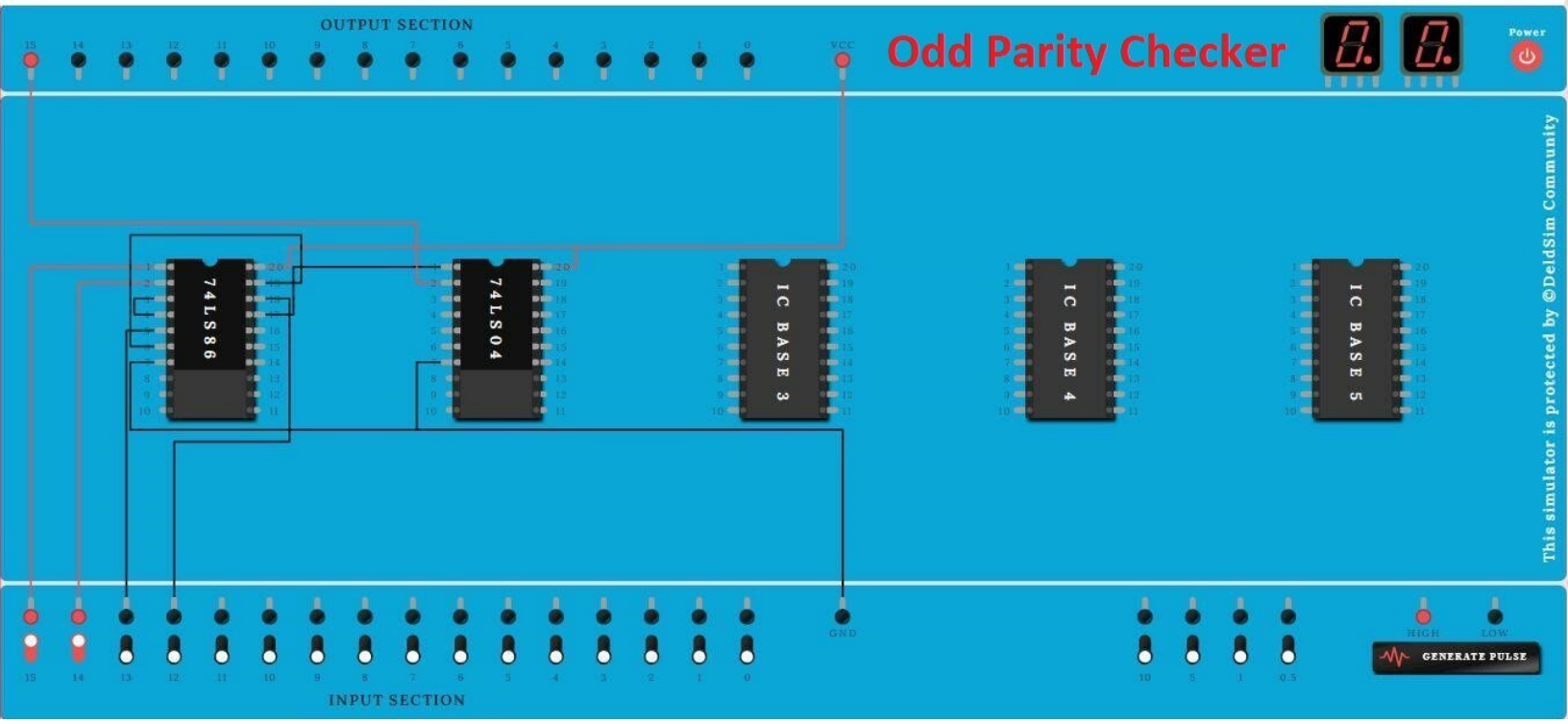
Input				Output
Parity(P)	B_2	B_1	B_0	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Expression using k-map

PB_2 \ B_1B_0	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	1	0	1	0
10	0	1	0	1

$Y = P\,(EX\text{-}OR)\,B_2\,(EX\text{-}OR)\,B_1\,(EX\text{-}OR)\,B_0$




OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
Odd Parity Checker
Power
74LS86
74LS04
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
10 5 1 0.5
INPUT SECTION

② Even Parity Checker -

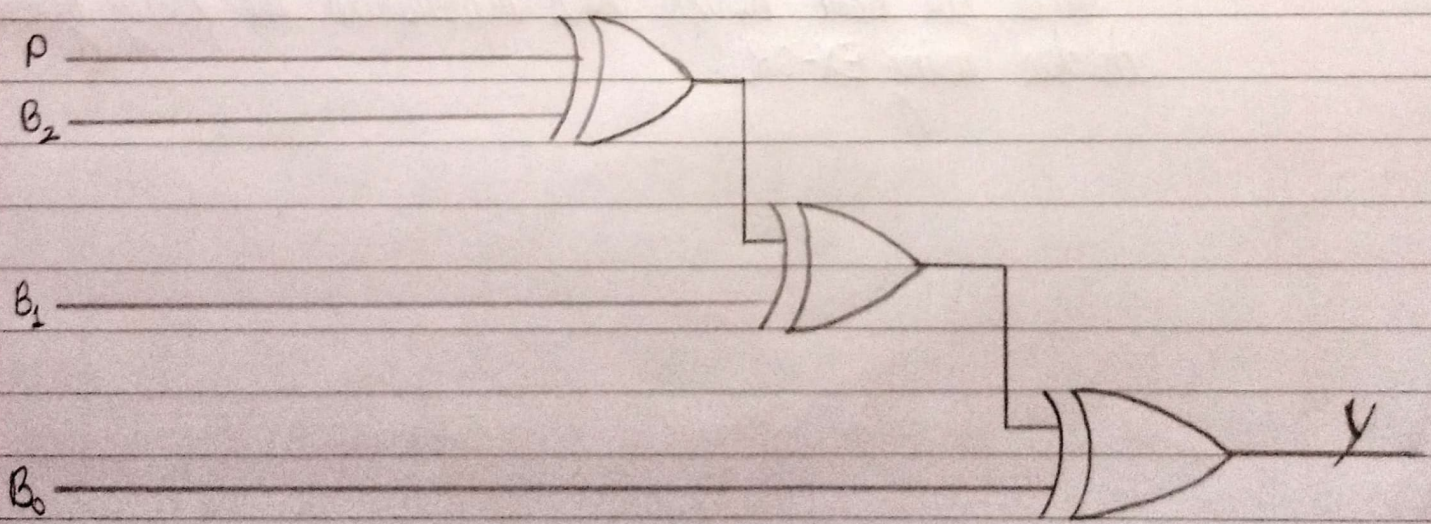
[0 → Error
1 → Not Error]

Input				Output
Parity (P)	B_2	B_1	B_0	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Expression using k-map.

PB_2 \ B_1B_0	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	0	1	0	1
10	1	0	1	0

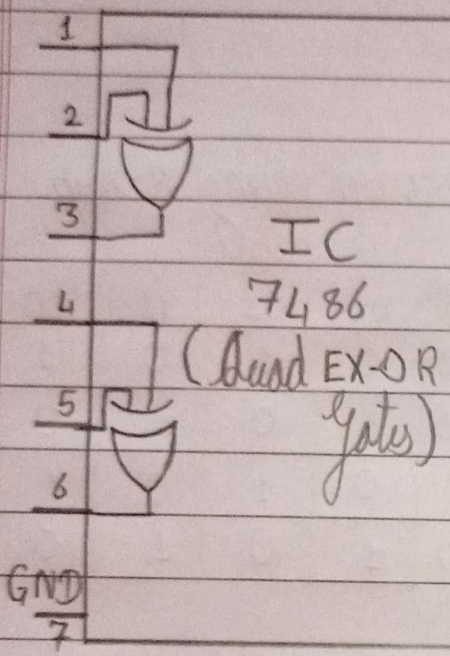
$Y = P \text{ (EX-OR) } B_2 \text{ (EX-OR) } B_1 \text{ (EX-OR) } B_0$



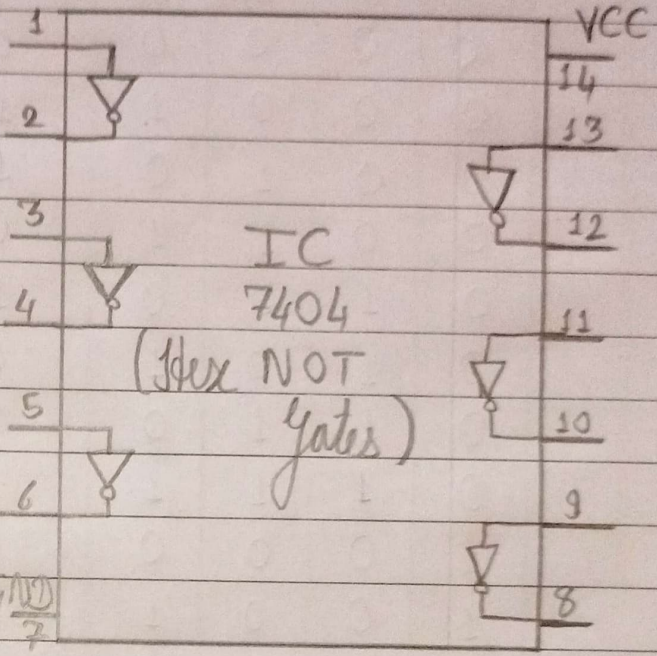
OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
Even Parity Checker
Power
74LS86
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
INPUT SECTION
10 5 1 0.5
HIGH
LOW
GENERATE PULSE

* IC's Used:-

1) EX-OR Gate - (IC 7486)



2) NOT Gate - (IC 7404)



Outcomes:-

Thus we study parity generator and checker and their truth tables.

Conclusion:-

Hence, we have design and implemented the parity generator and checker using EX-OR.

* Digital Electronics and Logic Design (DELD) - Practical Number - 8

Name :- Kaustubh Shrikant Kabra.
Class :- Second Year Engineering
Div :- A Roll Number :-
Batch :-
Department :- Computer Department
College :- AISSMS's IOIT.

Aim :-

Design and Realization : Flip-Flop Conversion.

Title :-

Flip-Flop Conversion.

Objective :-

To convert one flip-flop into another type of flip-flop :-

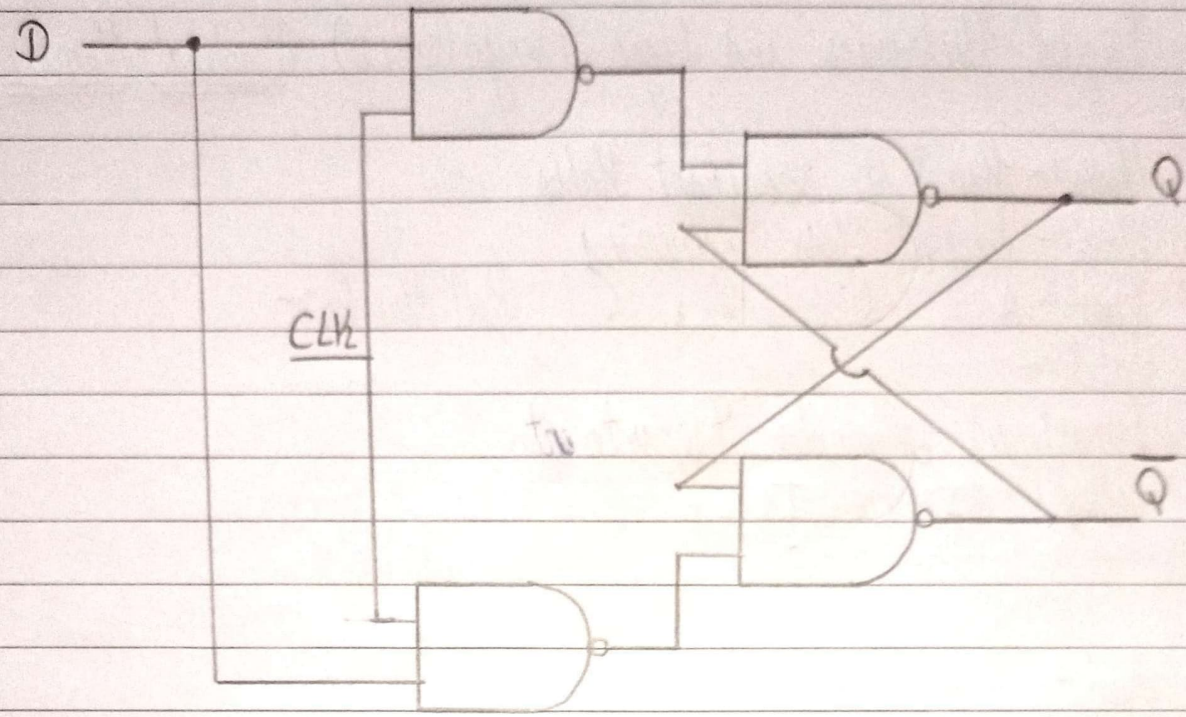
1. D-flip flop to T-flip flop.
2. JK-flip flop to D-flip flop.

Theory :-

D-Flip Flop -

It is a modified Set-Reset flip-flop with addition of an inverter to prevent the S and R input being at same logic level.

D-flip flop is by far the most important of clocked flip flop as it ensures that input S and R never equal at same time. It is constructed from a gated SR-flip flop with an inverter added between the S and R input to allow for a single D (input) Data.

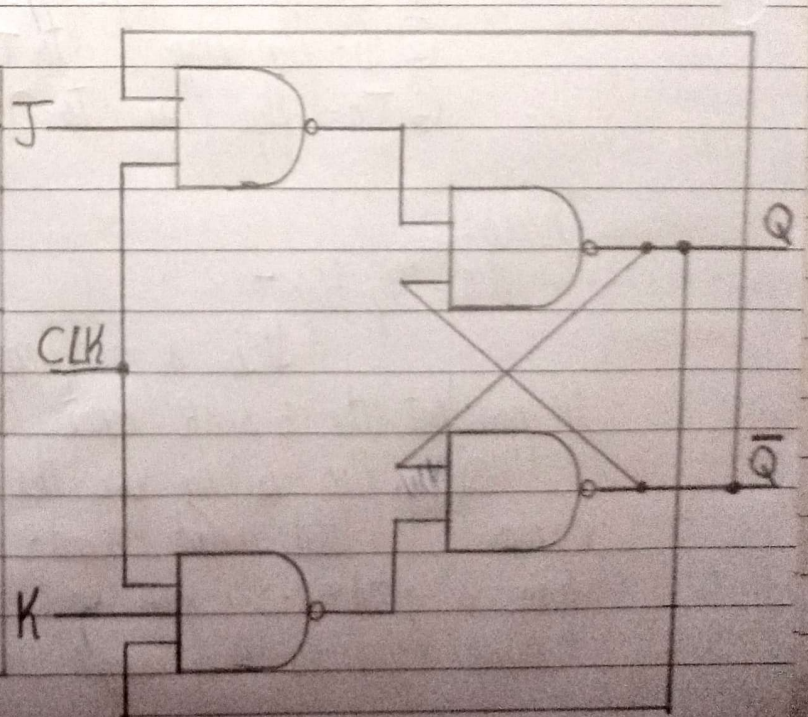


J-K Flip-Flop:-

JK flip flop is similar to SR flip flop but there is no change in state when J and K input are both Low. JK flip flop is considered to be a universal flip flop circuit.

JK flip flop has no invalid or forbidden input states of the SR latch even when S and R are high.

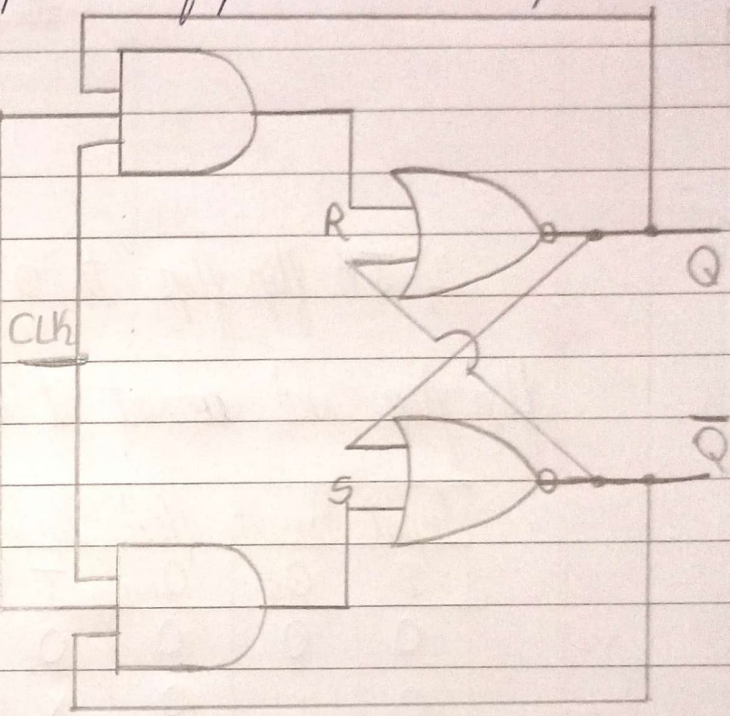
Clock	Input		Output		Description
	J	K	Q	$\overline{Q}$	
x	0	0	1	0	Memory NC
x	0	0	0	1	
↓	0	1	1	0	Reset
x	0	1	0	1	
↓	1	0	0	1	Set
x	1	0	1	0	
↓	1	1	0	1	Toggle
↓	1	1	1	0	



T- Flip Flop:-

To avoid the intermediate state occurence in SR flip flop, T flip flop is used. This flip flop work as toggle switch. The next output state is changed with the complement of present state output. This is called toggling.

	Previous		Next	
T	Q	$\bar{Q}$	Q	$\bar{Q}$
0	0	1	0	1
0	1	0	1	0
1	0	1	1	0
1	1	0	0	1



* Conversion:-

1) D flip flop to T flip flop:-

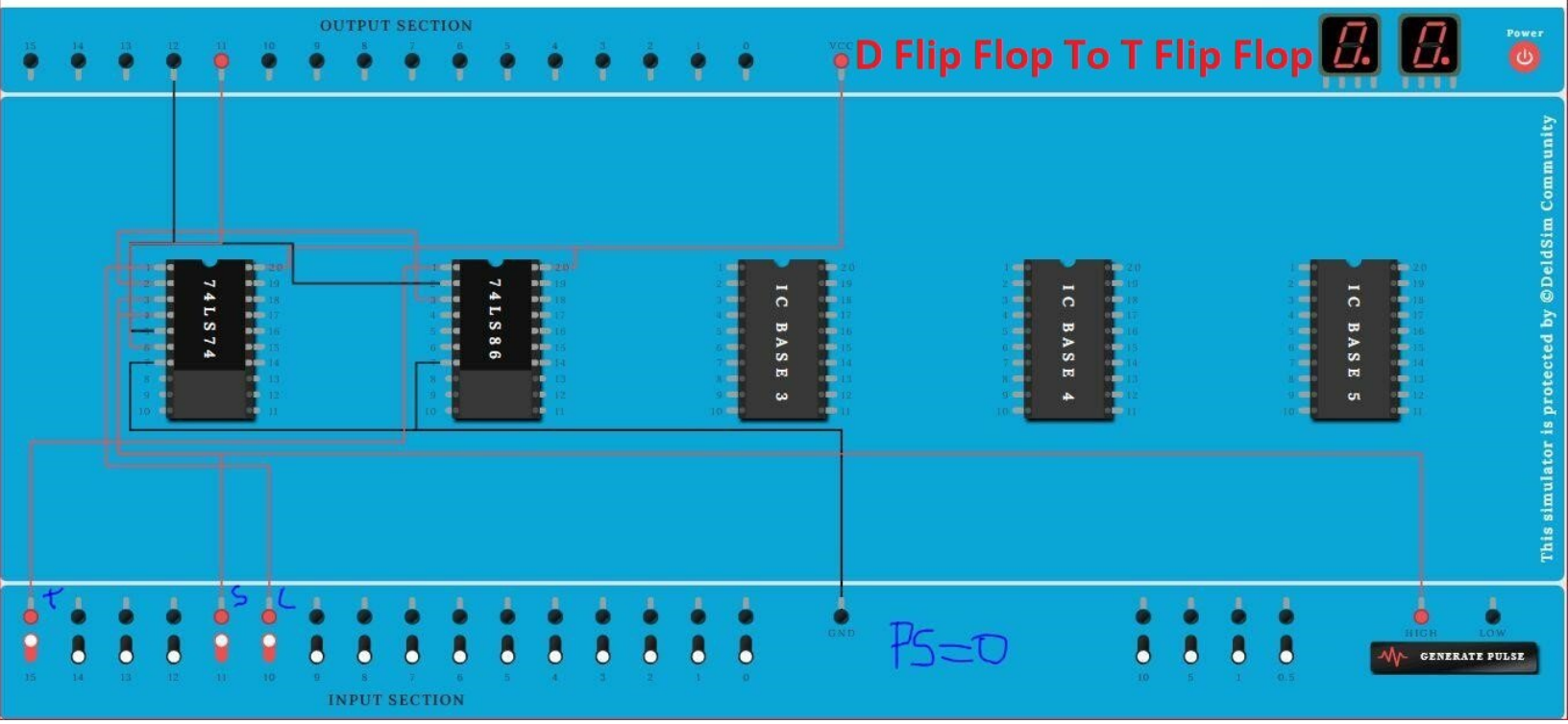
Compare the truth table of both D and T flip flop and make excitation table.

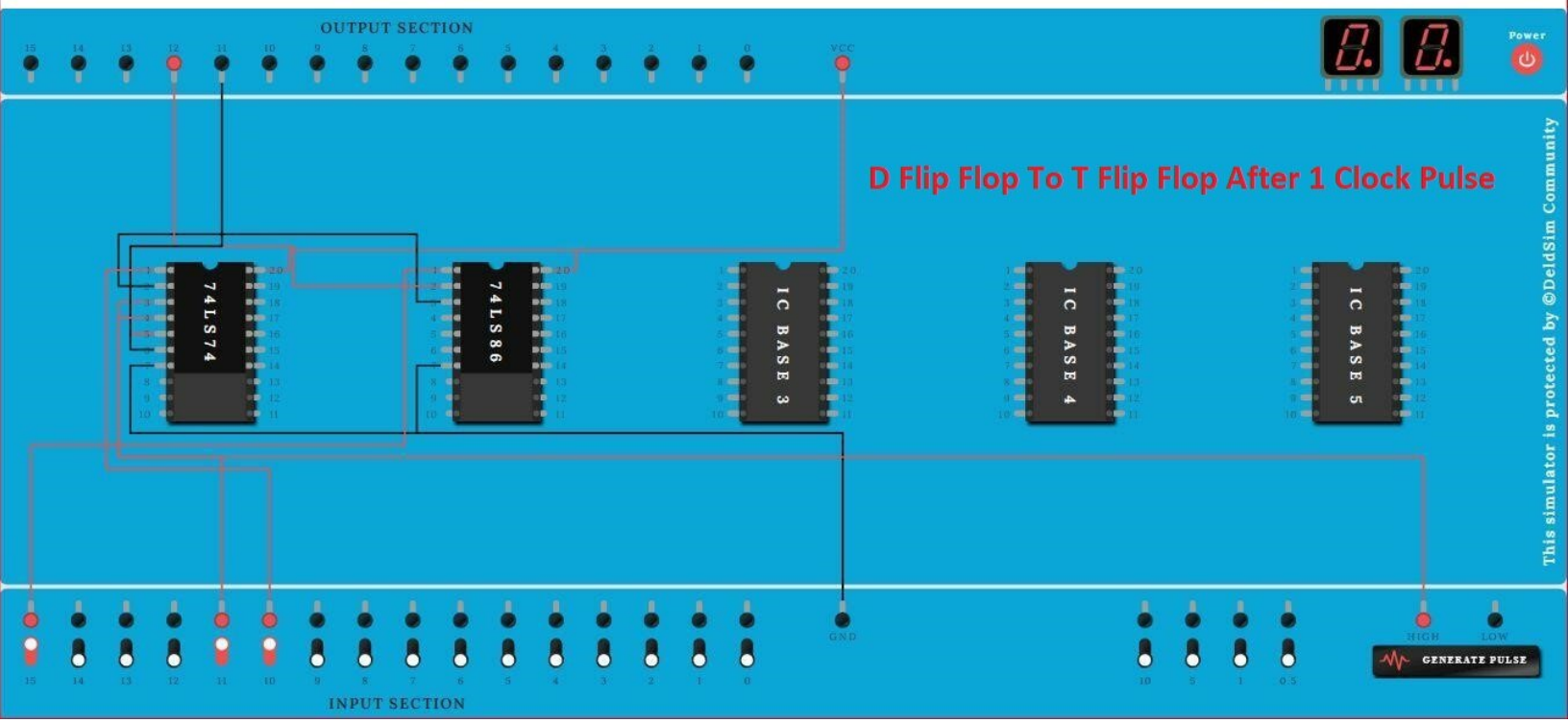
Input	Present S	Next S	FF input
T	Q_n	Q_{n+1}	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

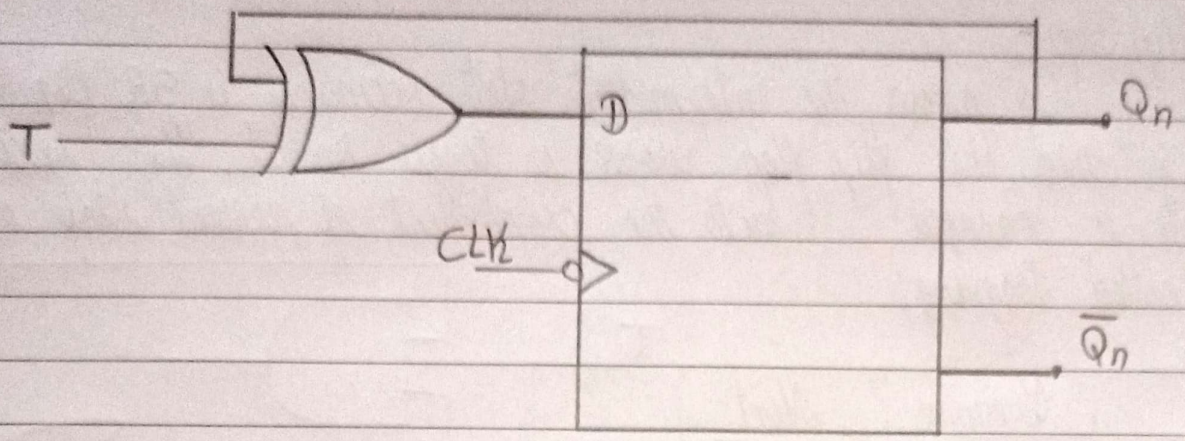
Using K-map for D.

T \ Q_n	0	1
0	0	1
1	1	0

$D = \bar{T}Q_n + T\bar{Q}_n = T \oplus Q_n$


OUTPUT SECTION
D Flip Flop To T Flip Flop
Power
74LS74
74LS86
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
T
S
C
PS=0
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
D Flip Flop To T Flip Flop After 1 Clock Pulse
Power
74LS74
74LS86
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION



2) JK flip flop to D flip flop.-

Compare the truth table of both flip flop and convert it into excitation table.

Input D	Present Q_n	Next Q_{n+1}	Flip Flop Input J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	1	X	0

Using K map for expression of JK.

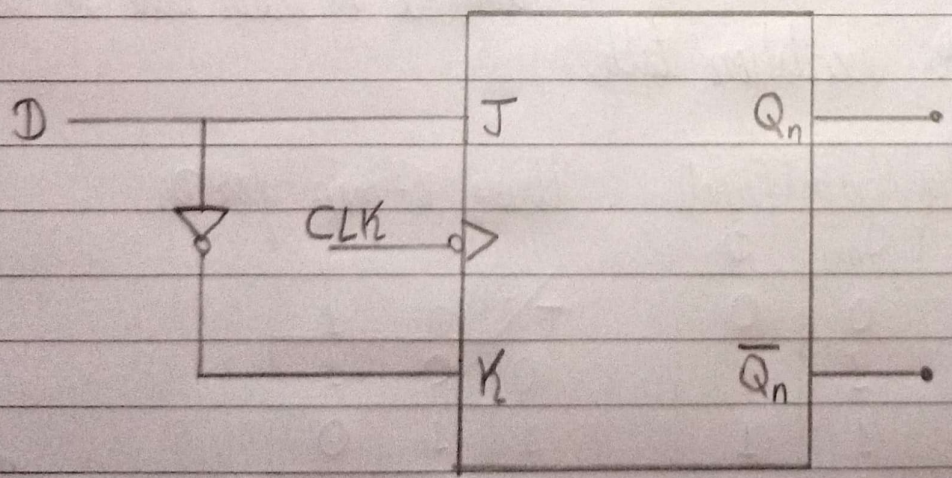
for J

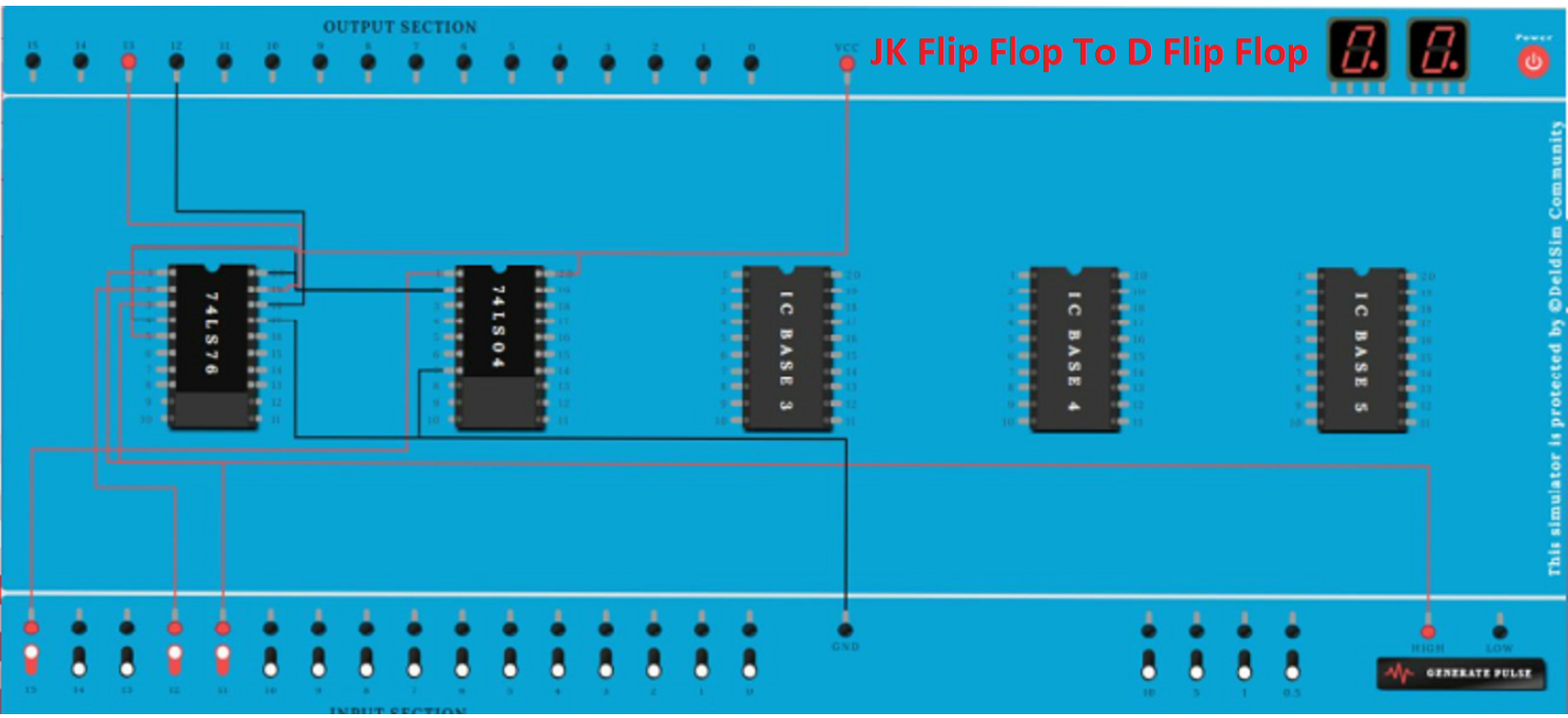
D \ Q_n	0	1
0	0	X
1	1	X

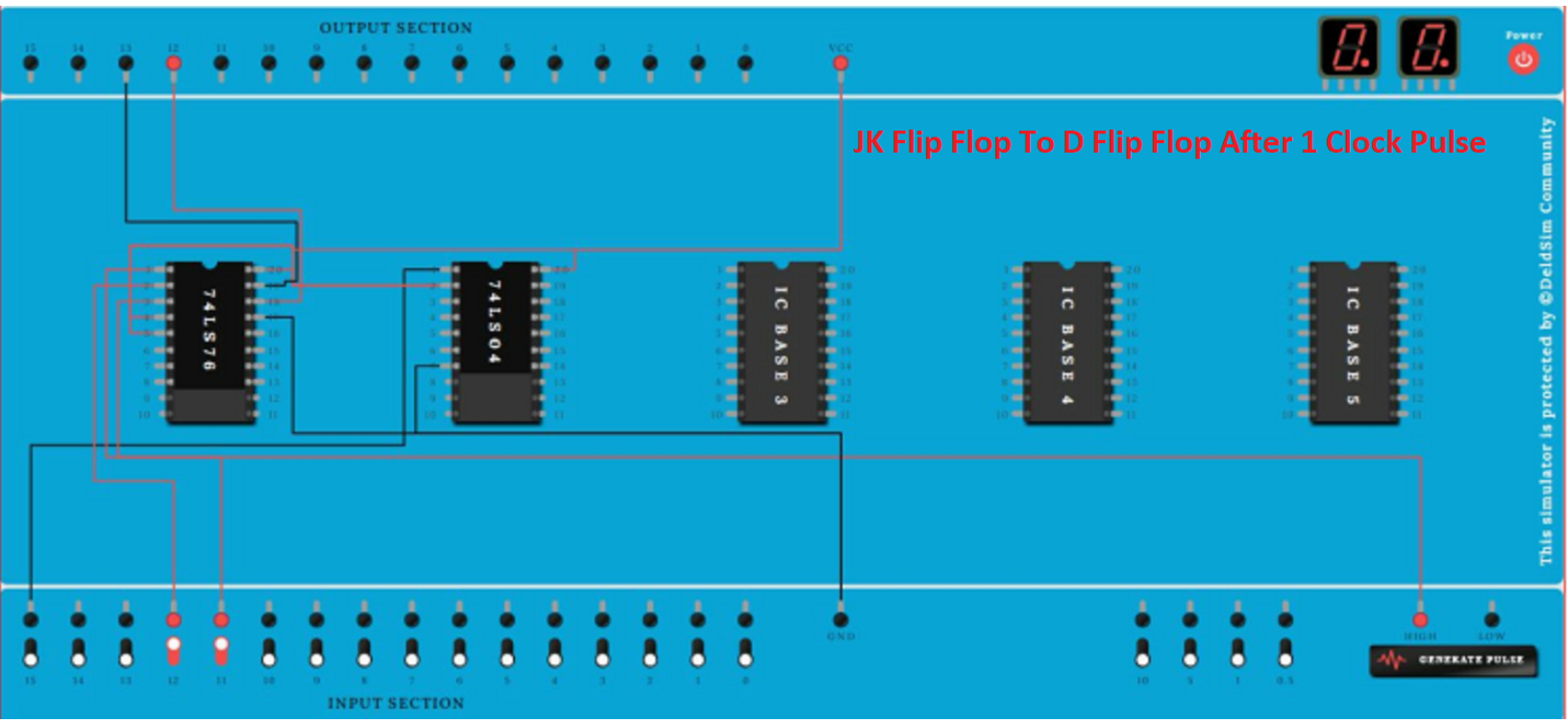
for K

D \ Q_n	0	1
0	X	1
1	X	0

$J = D$ $\quad K = \overline{D}$

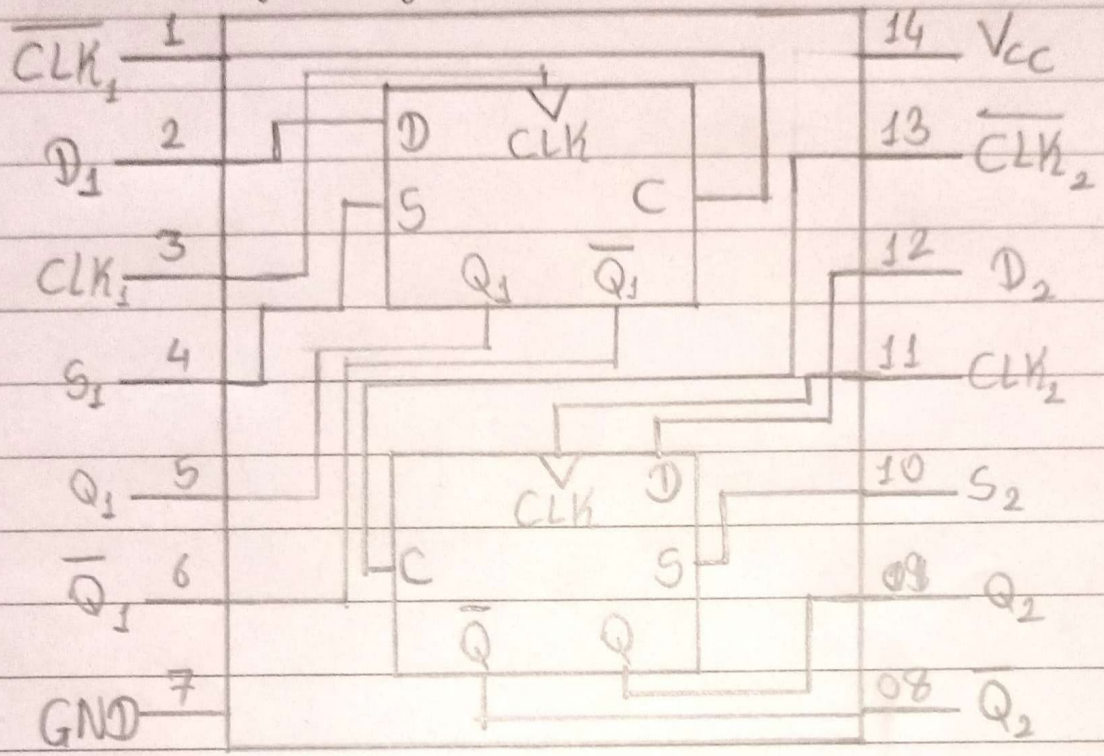



OUTPUT SECTION
JK Flip Flop To D Flip Flop
74LS76
74LS04
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE


OUTPUT SECTION
JK Flip Flop To D Flip Flop After 1 Clock Pulse
74LS76
74LS04
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

IC's Used -

① Dual Positive Edge trigered IC 7474 (D flip flop) -



Conclusion:-

Hence, we have successfully converted the flip flop from one type to another type.

* Digital Electronics and Logic Design (DELD) - Practical Number - 9

Name :- Kaustubh Shrikant Kabra.
Class :- Second Year Engineering.
Div :- A Roll Number :-
Batch :-
Department :- Computer Department.
College :- AISSMS's IOIT.

Title :-
Ripple Counter

Aim :-
Design of 2 bit and 3-bit ripple counter using MS-JK flip flops.

Objective :-
To design 2-bit and 3-bit ripple counter with help of MS-JK flip-flops.

Theory :-
Asynchronous counter / Ripple Counter :-
A digital counter is set of flip-flop
An ripple counter uses T flip flop to perform a counting function.
In ripple counter, first flip flops is clocked with clock pul
and then each successive flip-flop is clocked by Q and $\bar{Q}$.

Types :-
① Up Counter
② Down Counter.

Truth Table and Logic Diagrams :-

A) 2-bit Ripple counters

① Up counter →

Counting State	Output	
	Q_A	Q_B
0	0	0
1	0	1
2	1	0
3	1	1

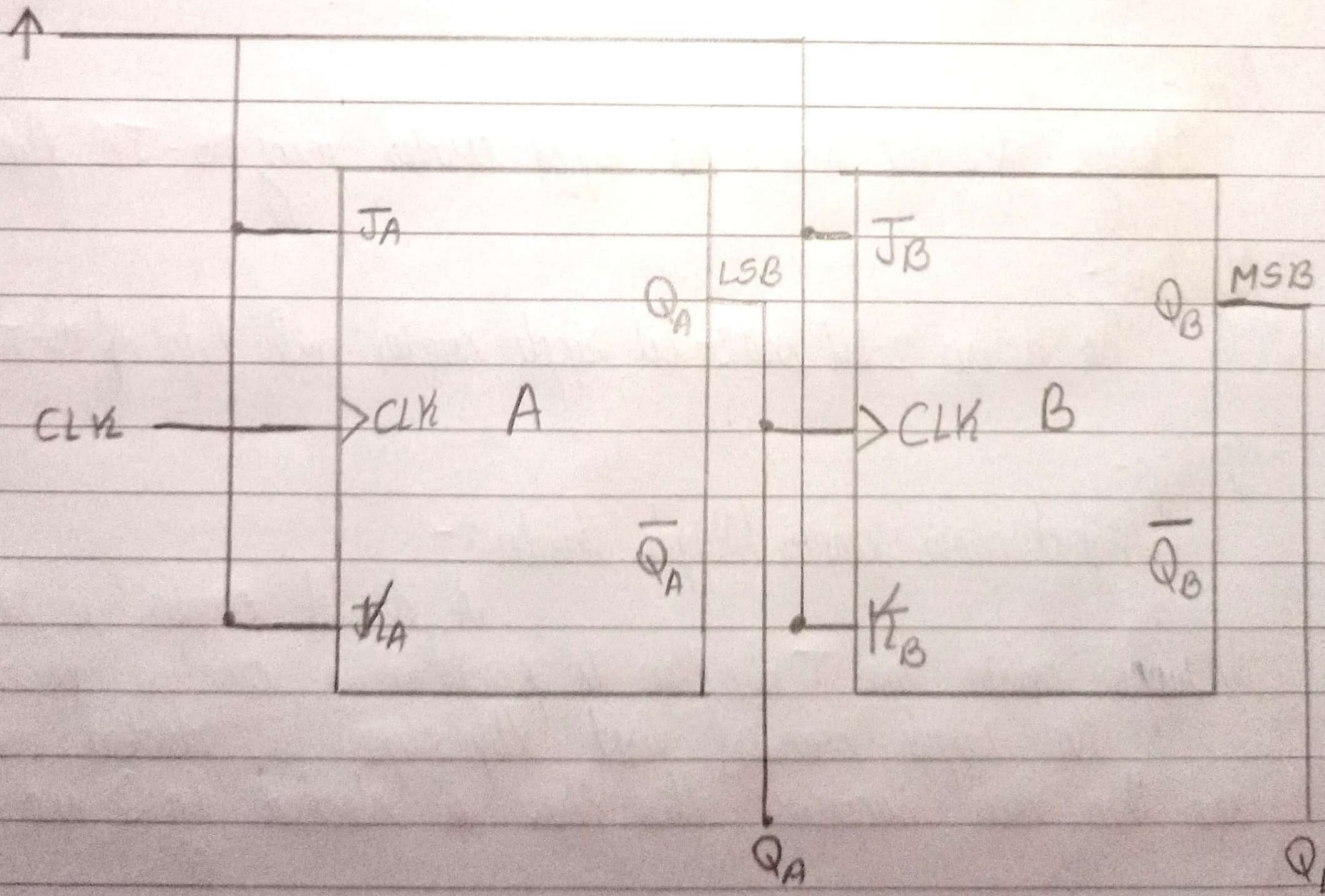
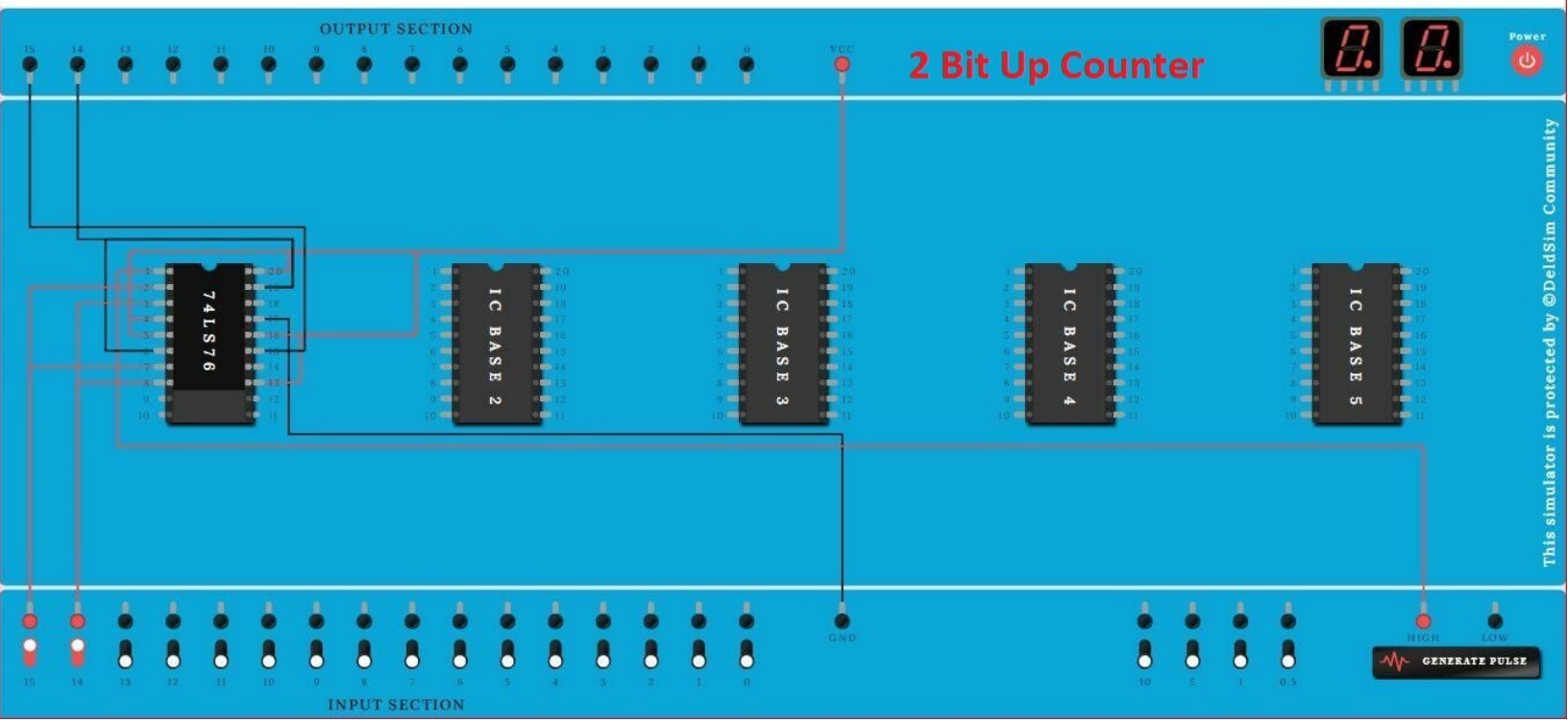
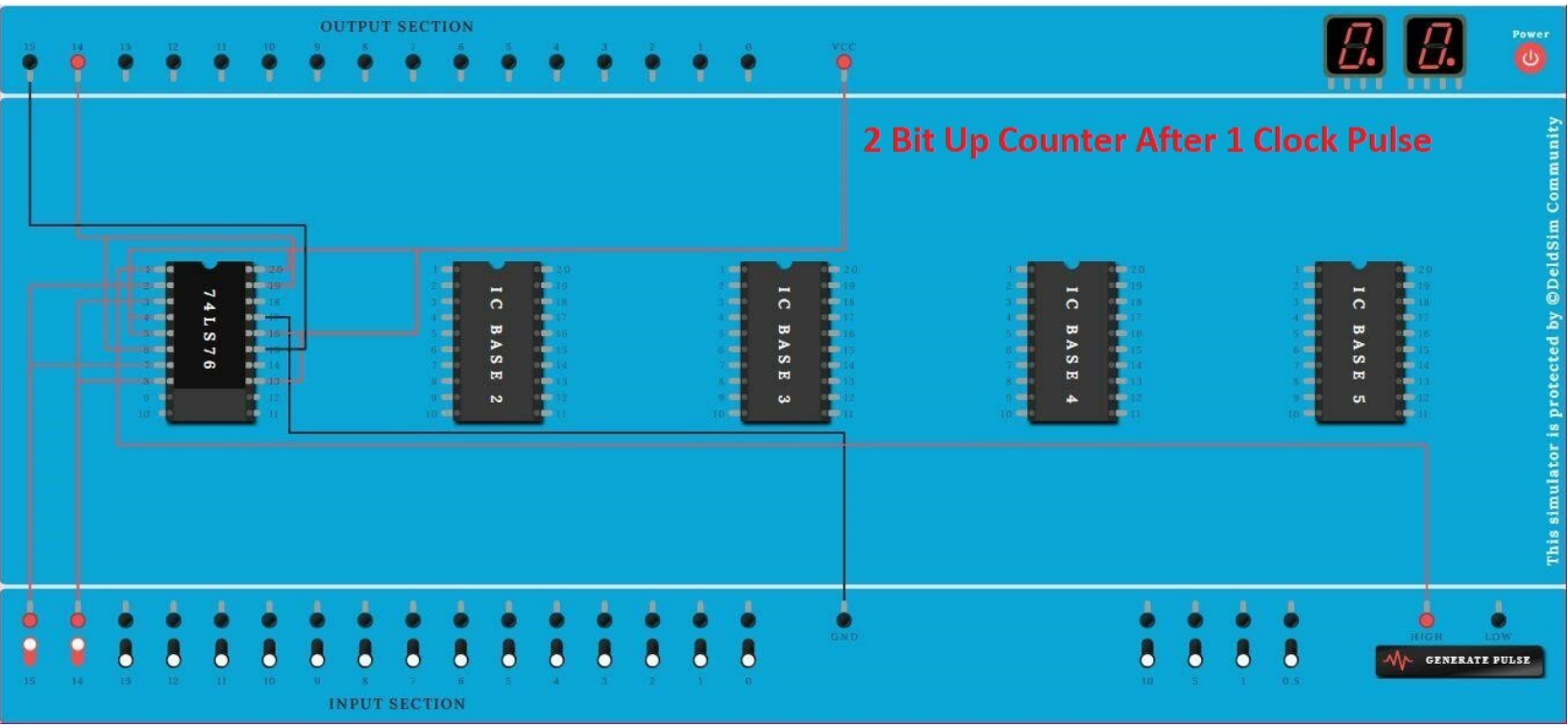


fig:- 2 bit Up counter.


OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
2 Bit Up Counter
Power
74LS76
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
10 5 1 0.5
INPUT SECTION


OUTPUT SECTION
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
VCC
2 Bit Up Counter After 1 Clock Pulse
Power
74LS76
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
10 5 1 0.5
INPUT SECTION

② Down counter →

Counter State	Output Q_A	Q_B
3	1	1
2	1	0
1	0	1
0	0	0

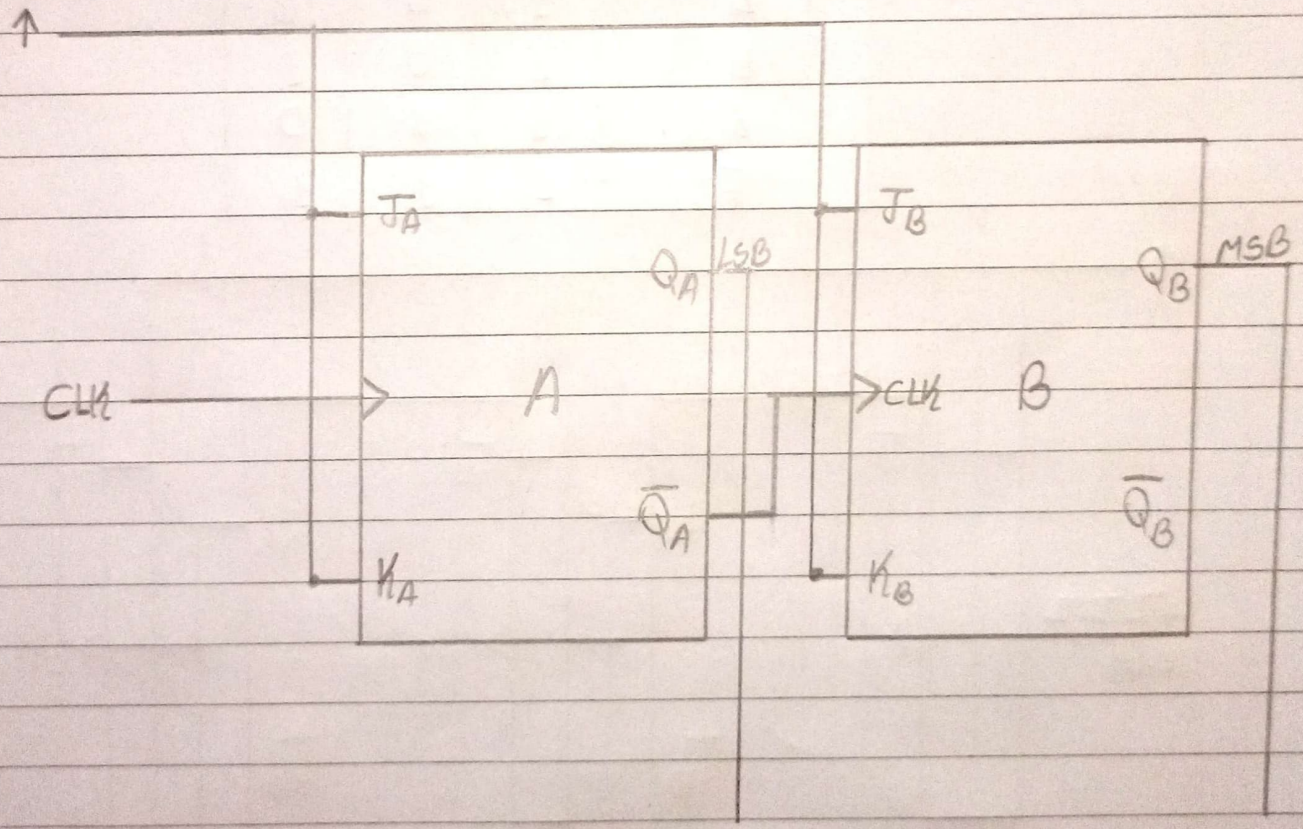
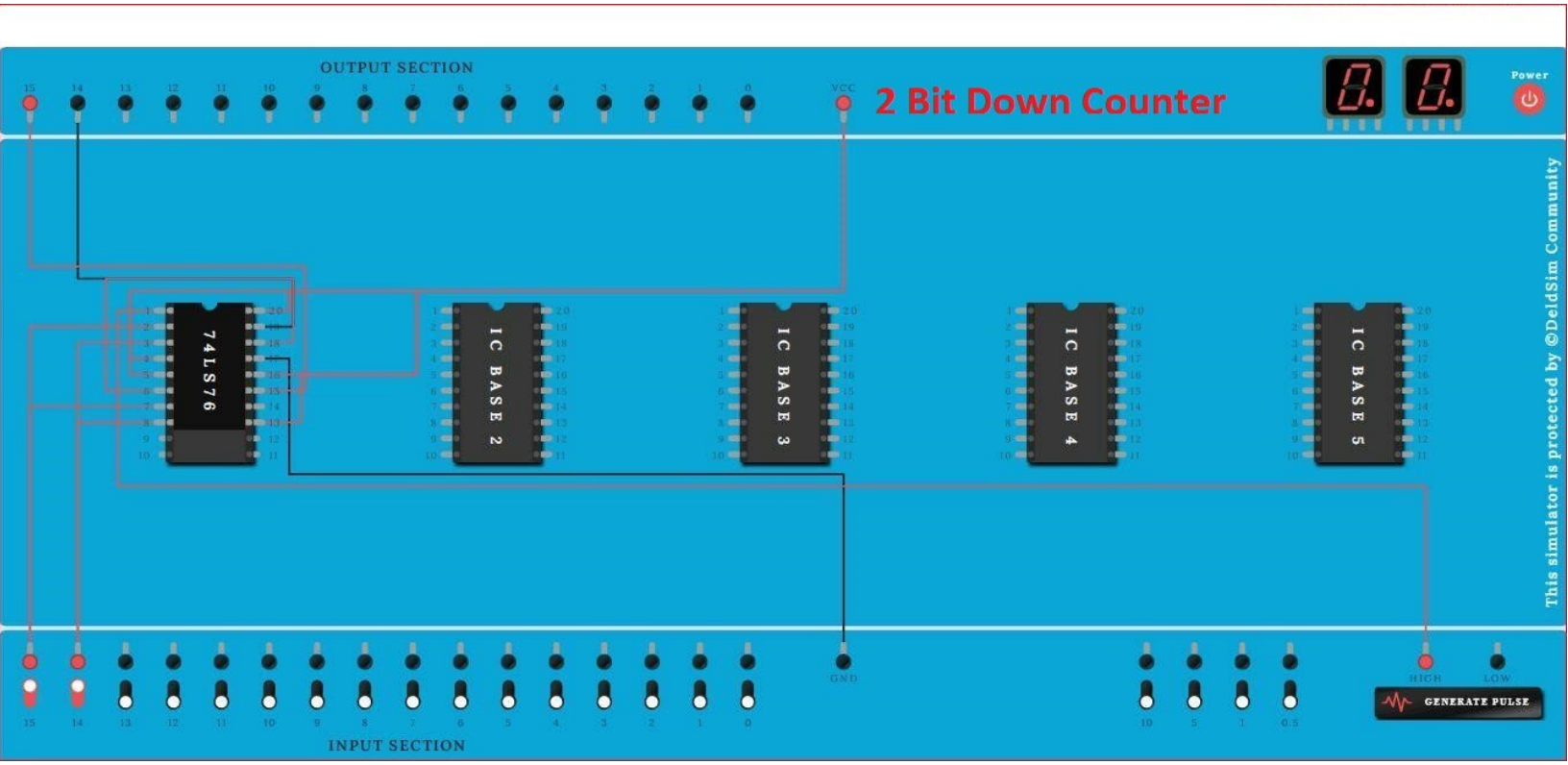
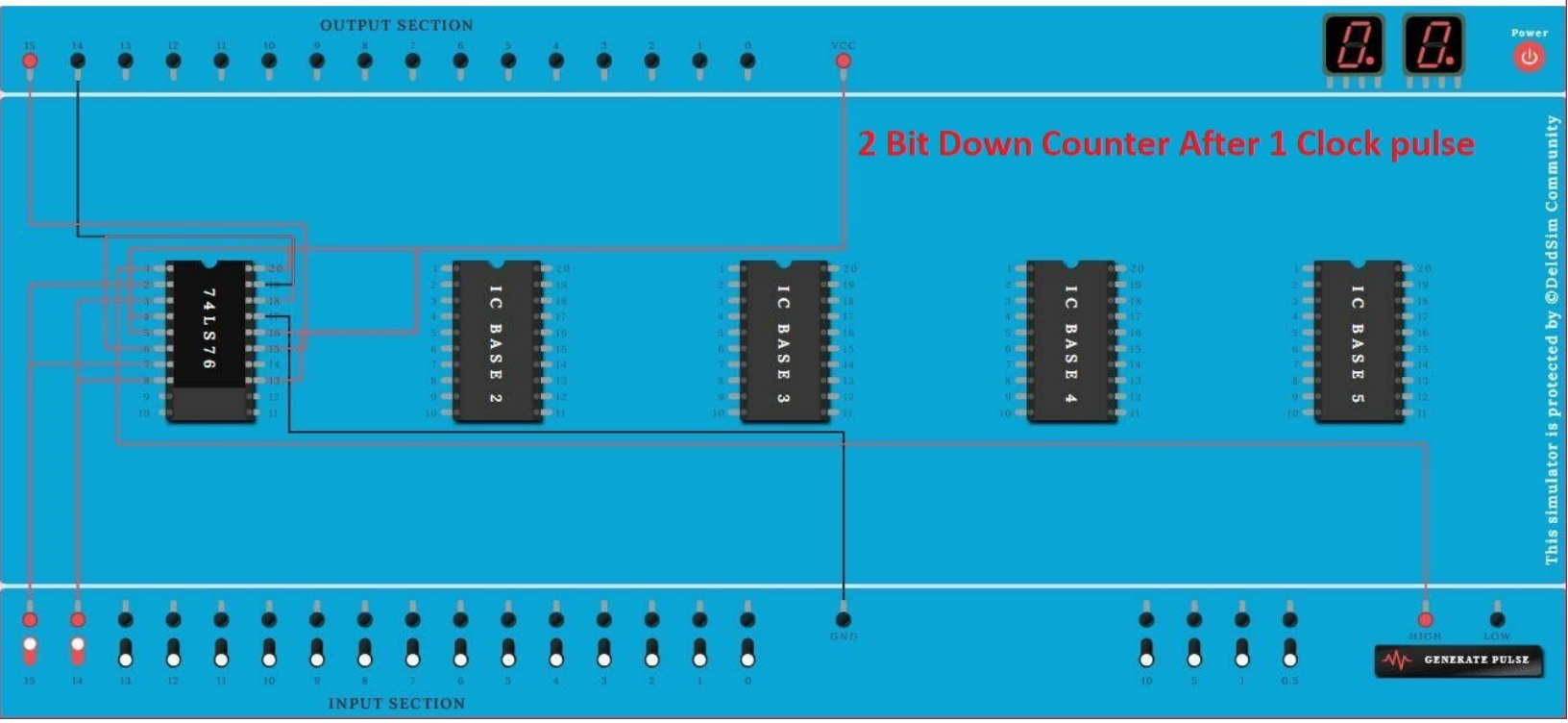


fig :- 2-bit Down counter.


OUTPUT SECTION
2 Bit Down Counter
Power
74LS76
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
VCC
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION
This simulator is protected by ©DeldSim Community


OUTPUT SECTION
2 Bit Down Counter After 1 Clock pulse
Power
74LS76
IC BASE 2
IC BASE 3
IC BASE 4
IC BASE 5
VCC
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION
This simulator is protected by ©DeldSim Community

B) 3-bit Ripple counters

① Up counter :-

Counter State	Output		
	Q_A	Q_B	Q_C
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1.

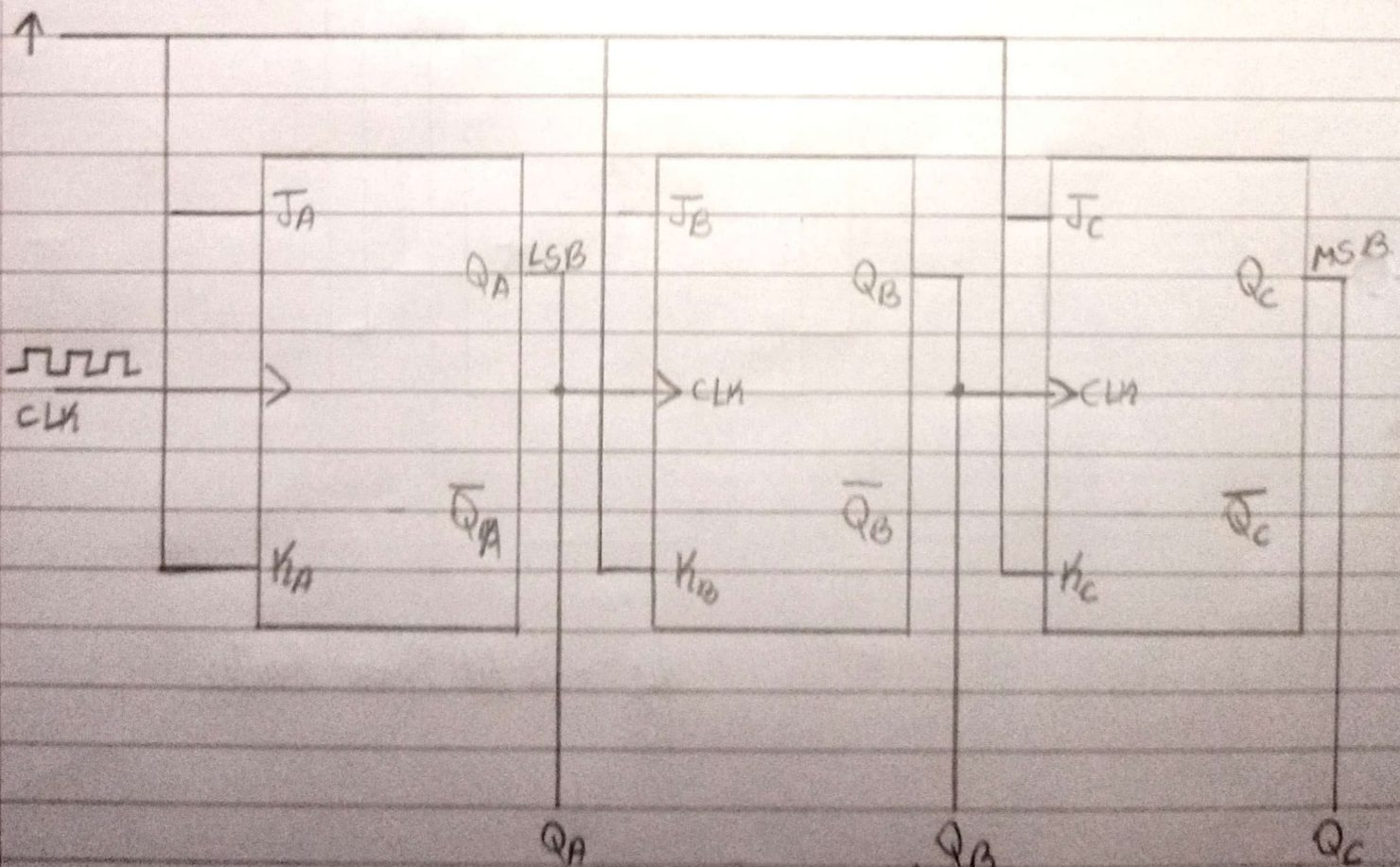
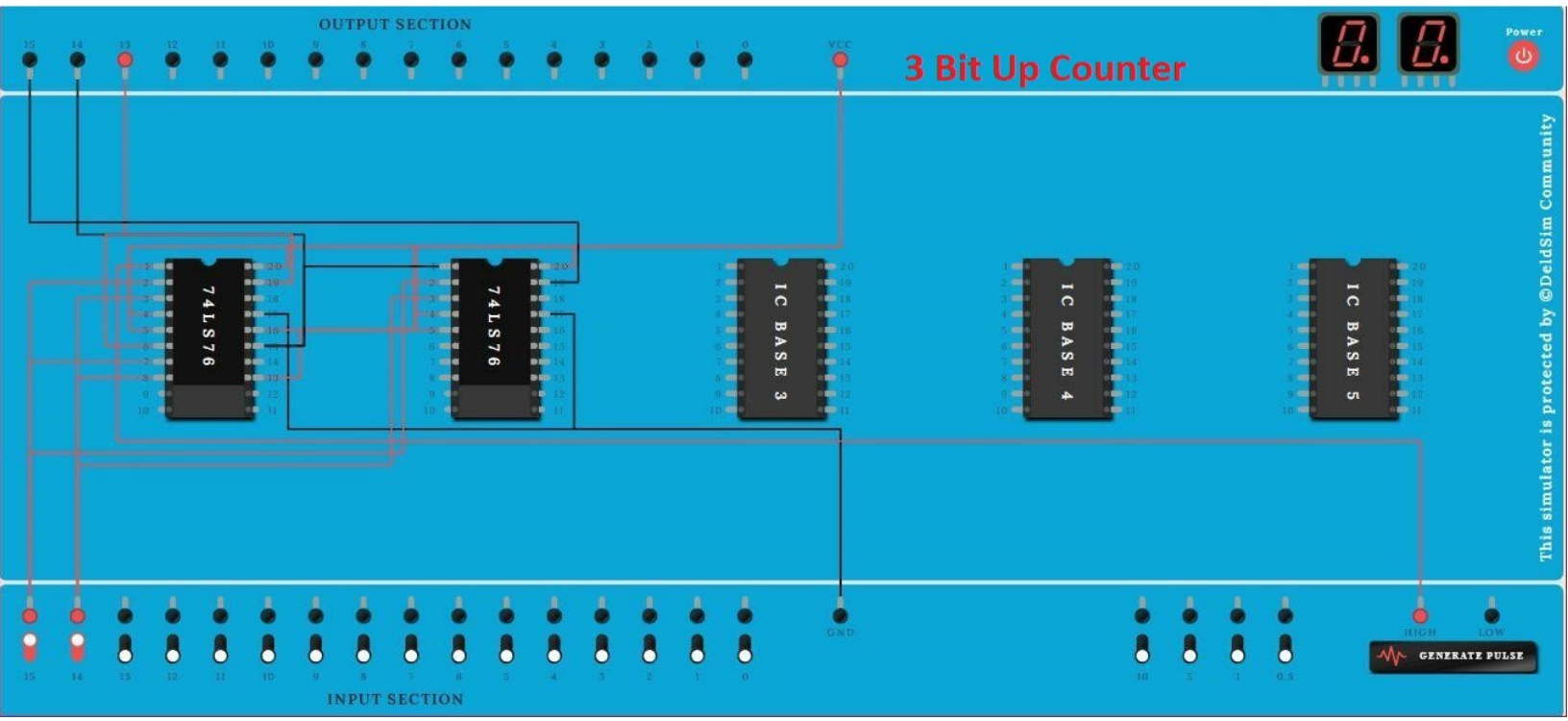
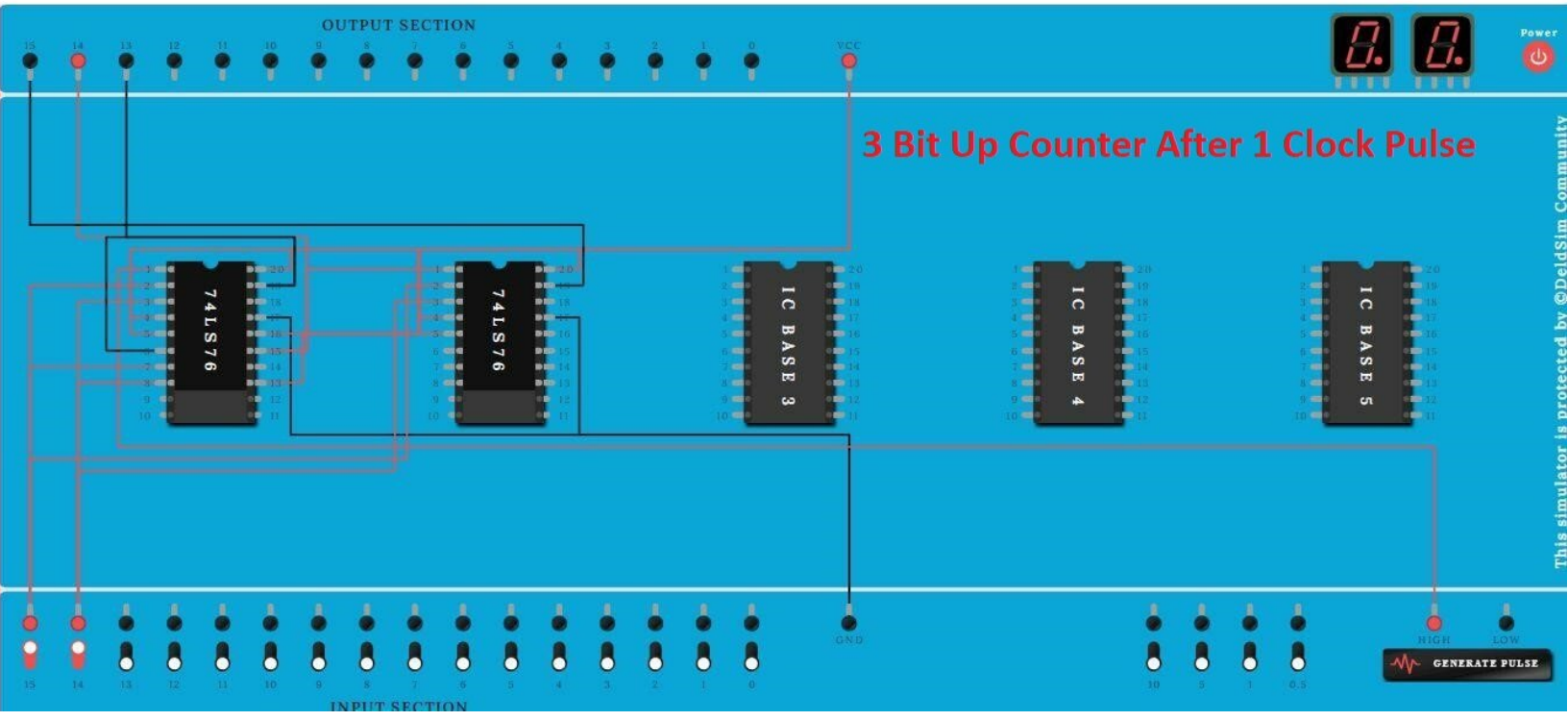


fig :- 3-bit Up counter


OUTPUT SECTION
3 Bit Up Counter
Power
74LS76
74LS76
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
3 Bit Up Counter After 1 Clock Pulse
Power
74LS76
74LS76
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

② Down counter :->

Counter State	Output		
	Q_A	Q_B	Q_C
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0

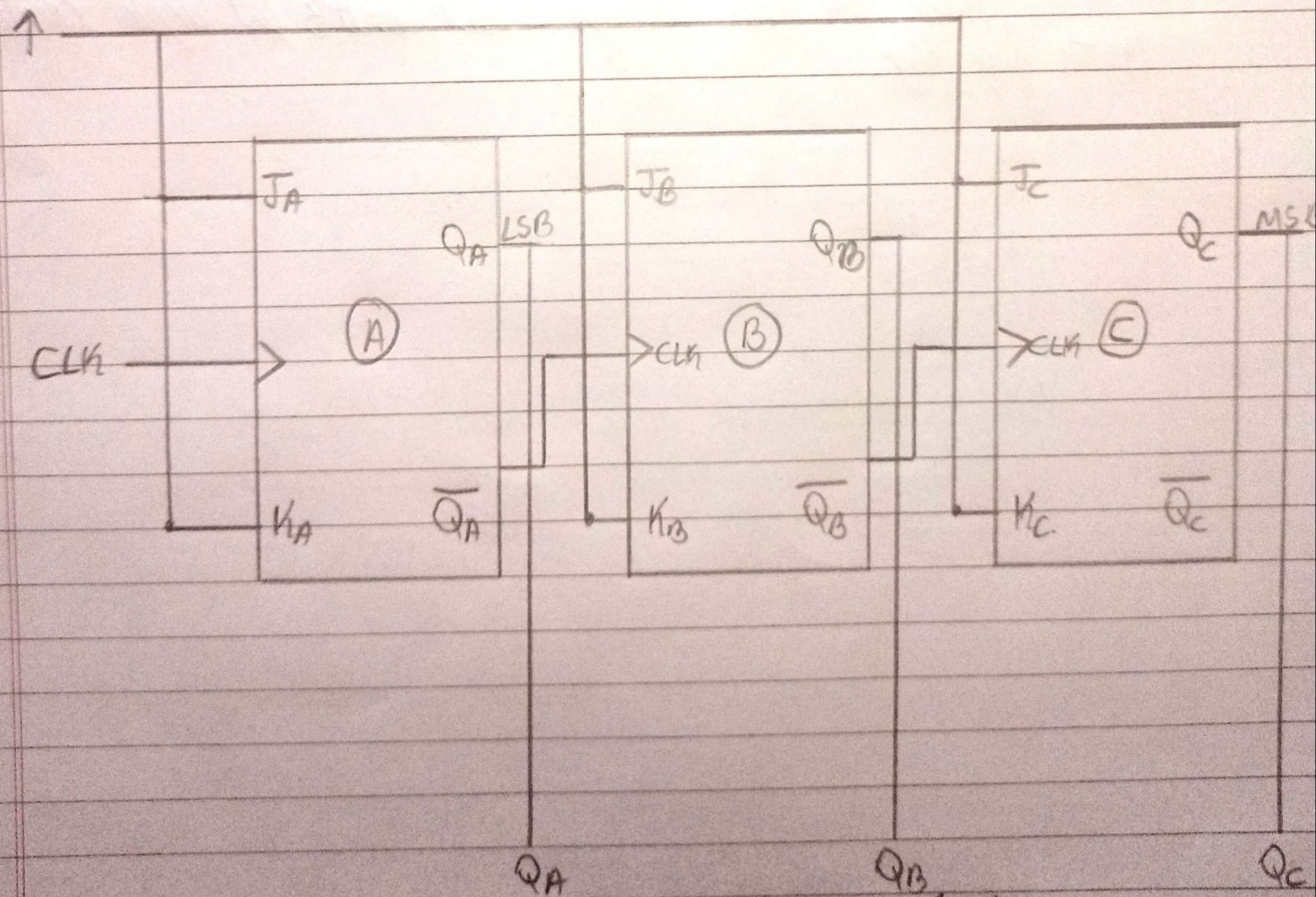
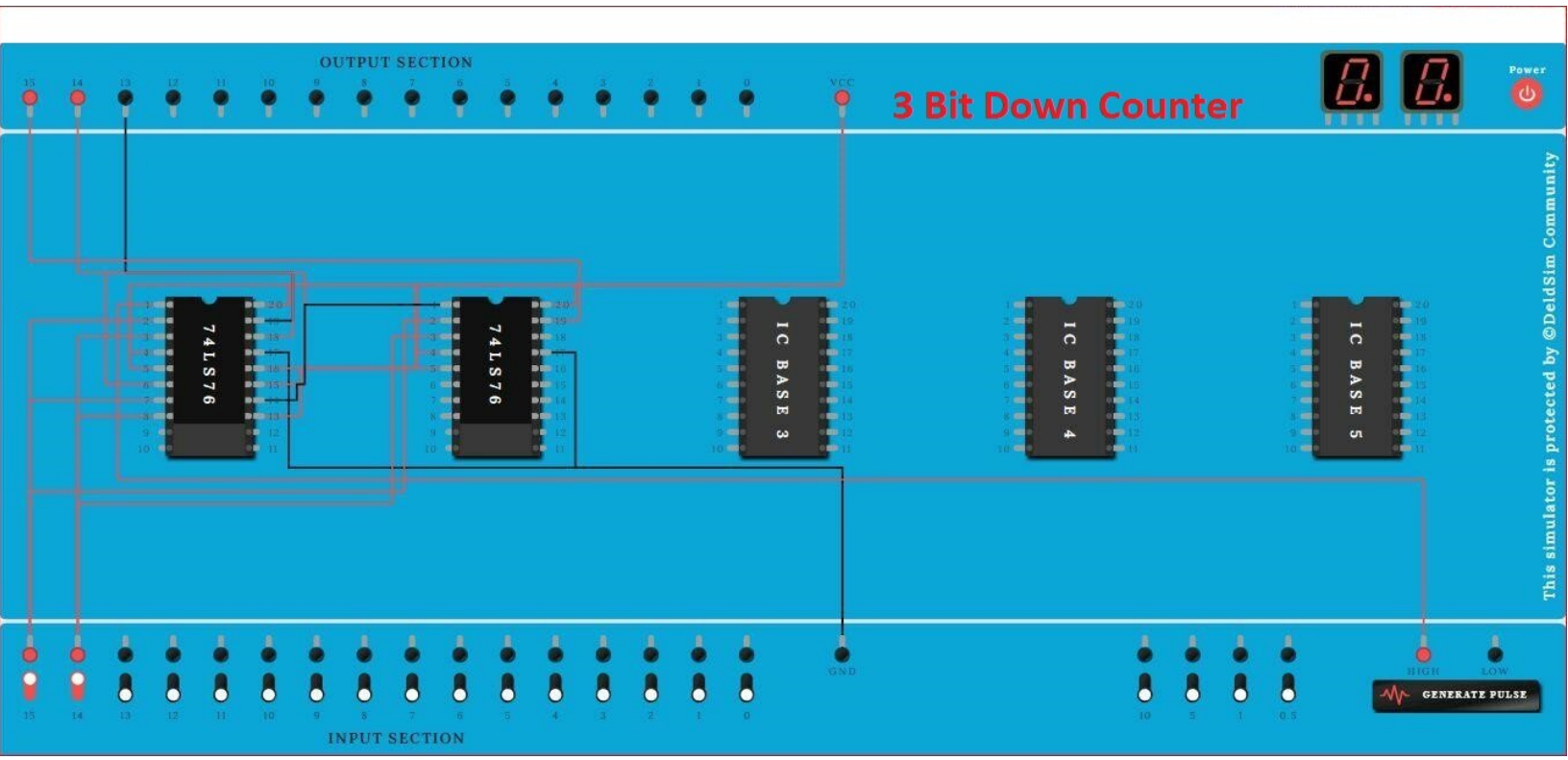
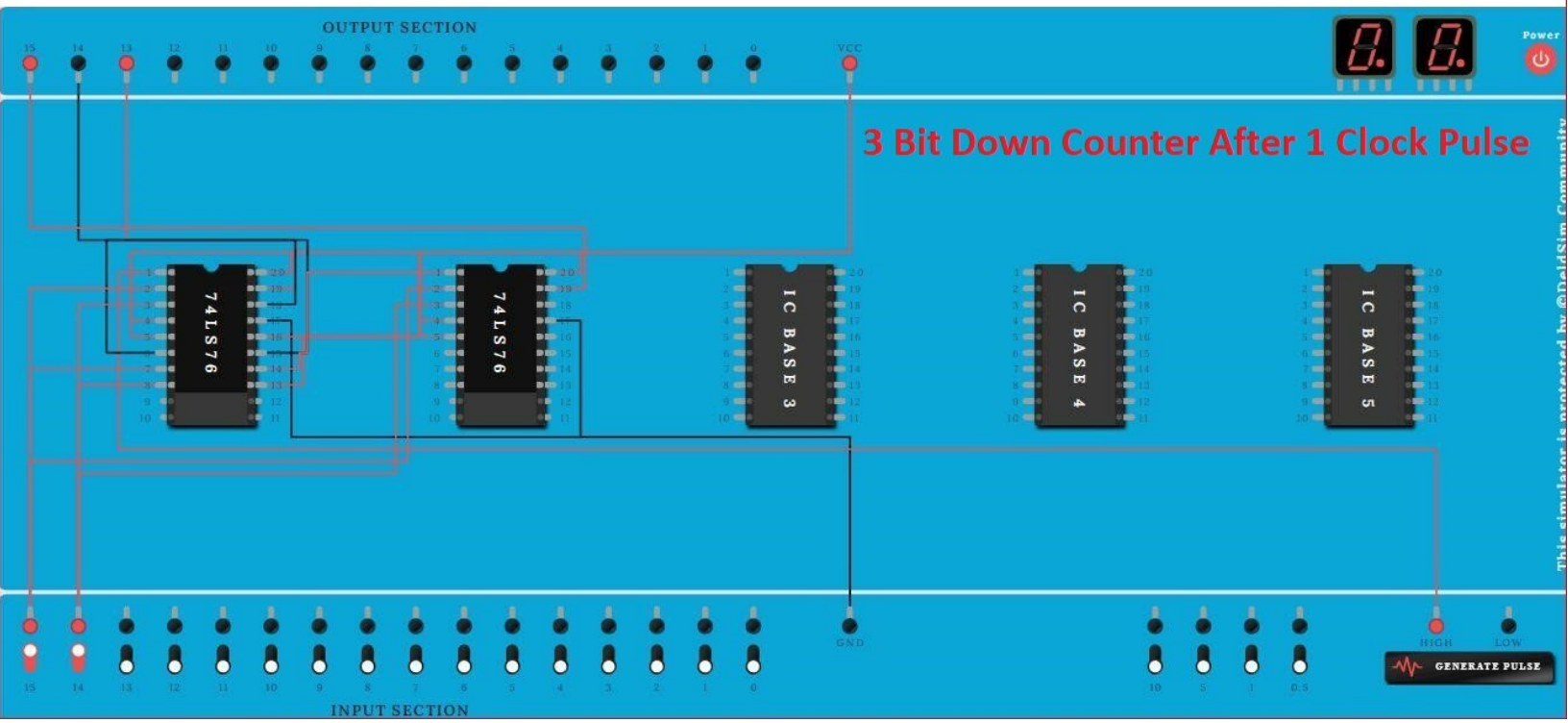


fig:- 3-bit Down counter.


OUTPUT SECTION
3 Bit Down Counter
Power
74LS76
74LS76
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
3 Bit Down Counter After 1 Clock Pulse
Power
74LS76
74LS76
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

Uses:-

1. Counting devices
2. Pulse counter
3. Used for dividing frequency

Outcome:-

Thus we implemented up and down ripple counter using IC 7476.

Conclusion:-

Hence, we have design 2 bit and 3 bit ripple counter using MSJK flip flop.

* Digital Electronics and Logic Design (DELD) - Practical Number - 10

Name :- Koustubh Shrikant Habra
Class :- Second Year Engineering
Div :- A Roll Number :-
Batch :-
Department :- Computer Department.
College :- AISSMS's IOIT.

Title :-
Synchronous counter.

Aim :-
Design of synchronous 3-bit up and down counter using MS-JK flip flops.

Objective :-
To design and understand 3-bit synchronous up down counter.

Theory :-
Synchronous counter -
In this counter, all the flip flop recives the external clock pulse simultaneously.

Example :- Ring counter and Johnson counter.

The gate propogation delay at reset time with not be present or we may say will not occur.

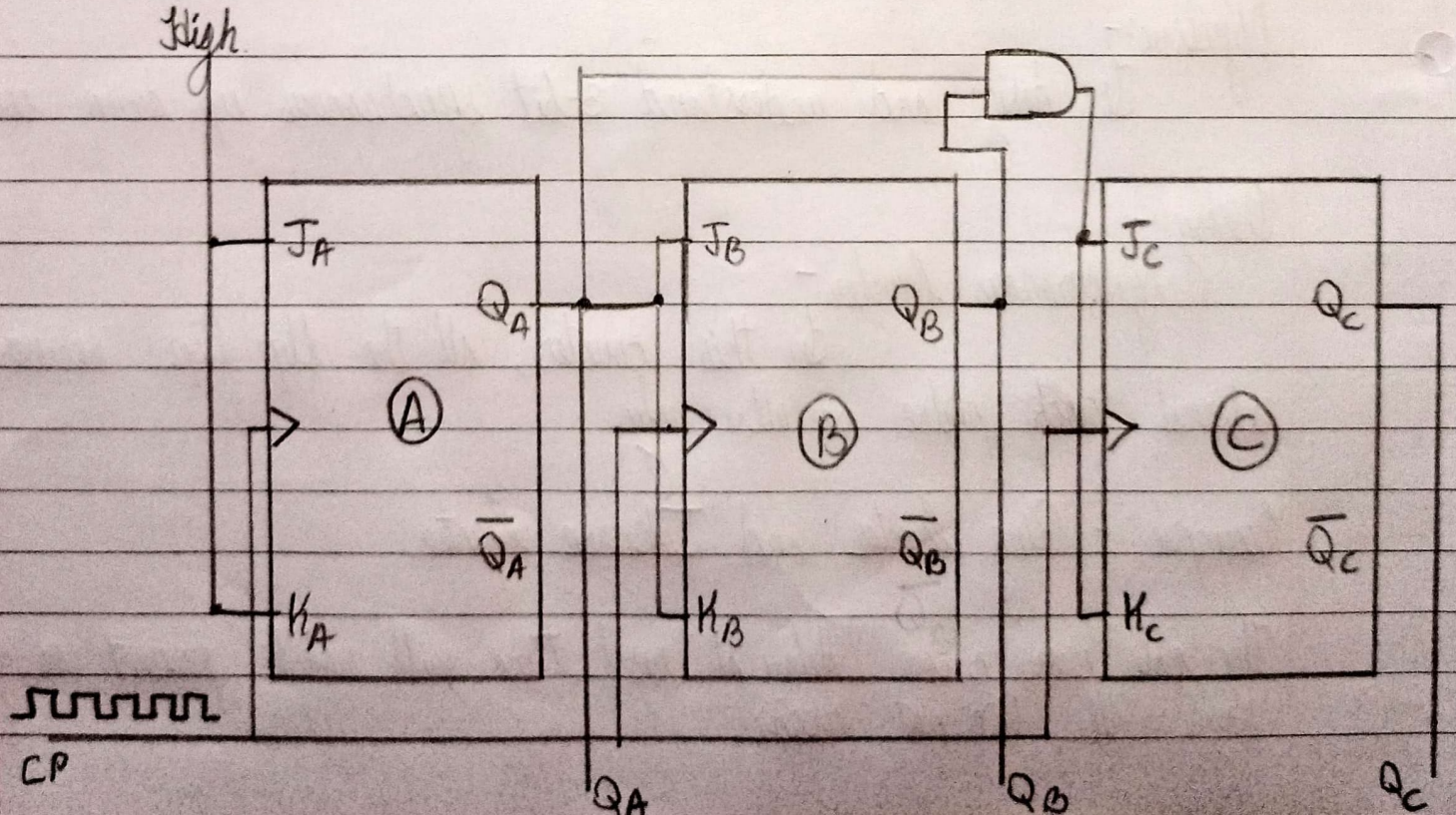
Classification of Synchronous Counter :-

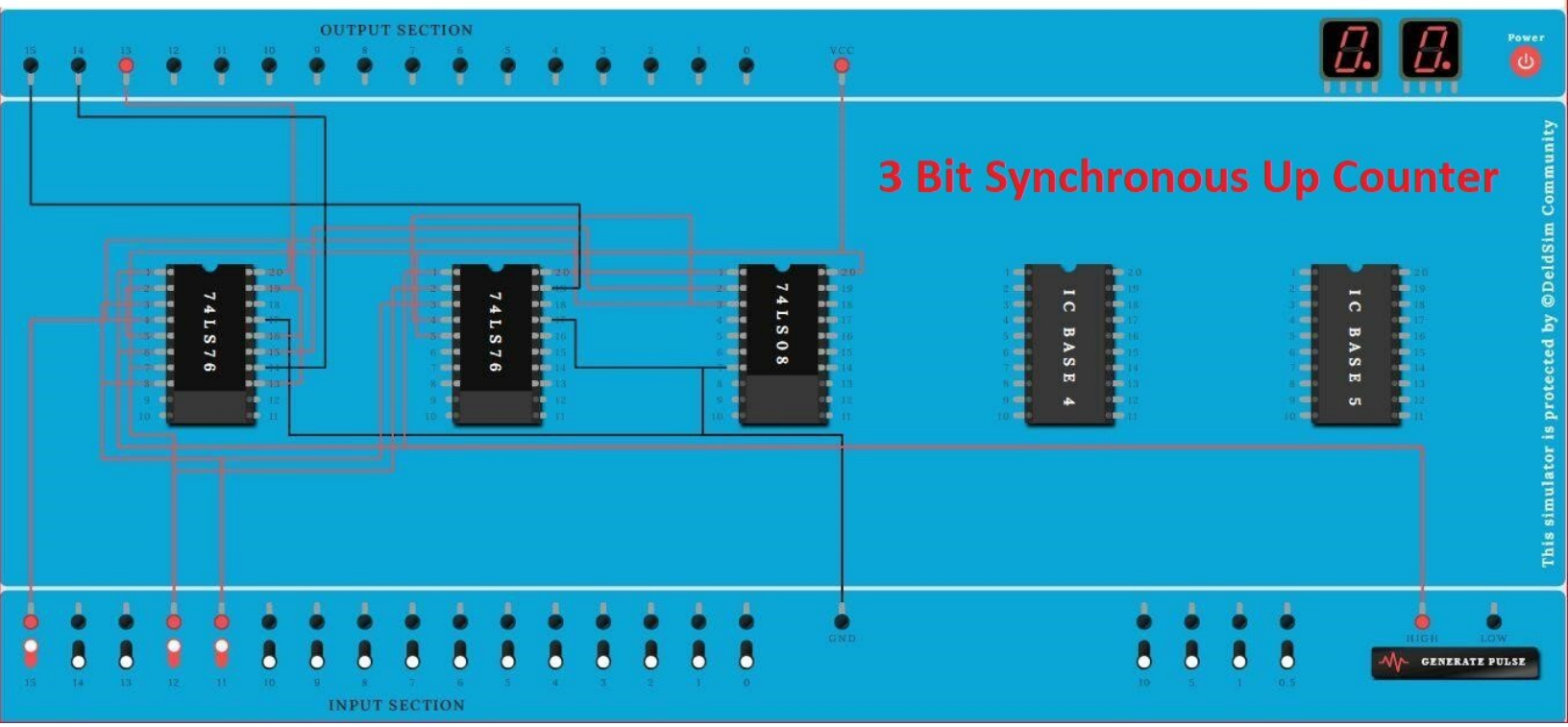
1. Up counter
2. Down counter
3. Up-down counter

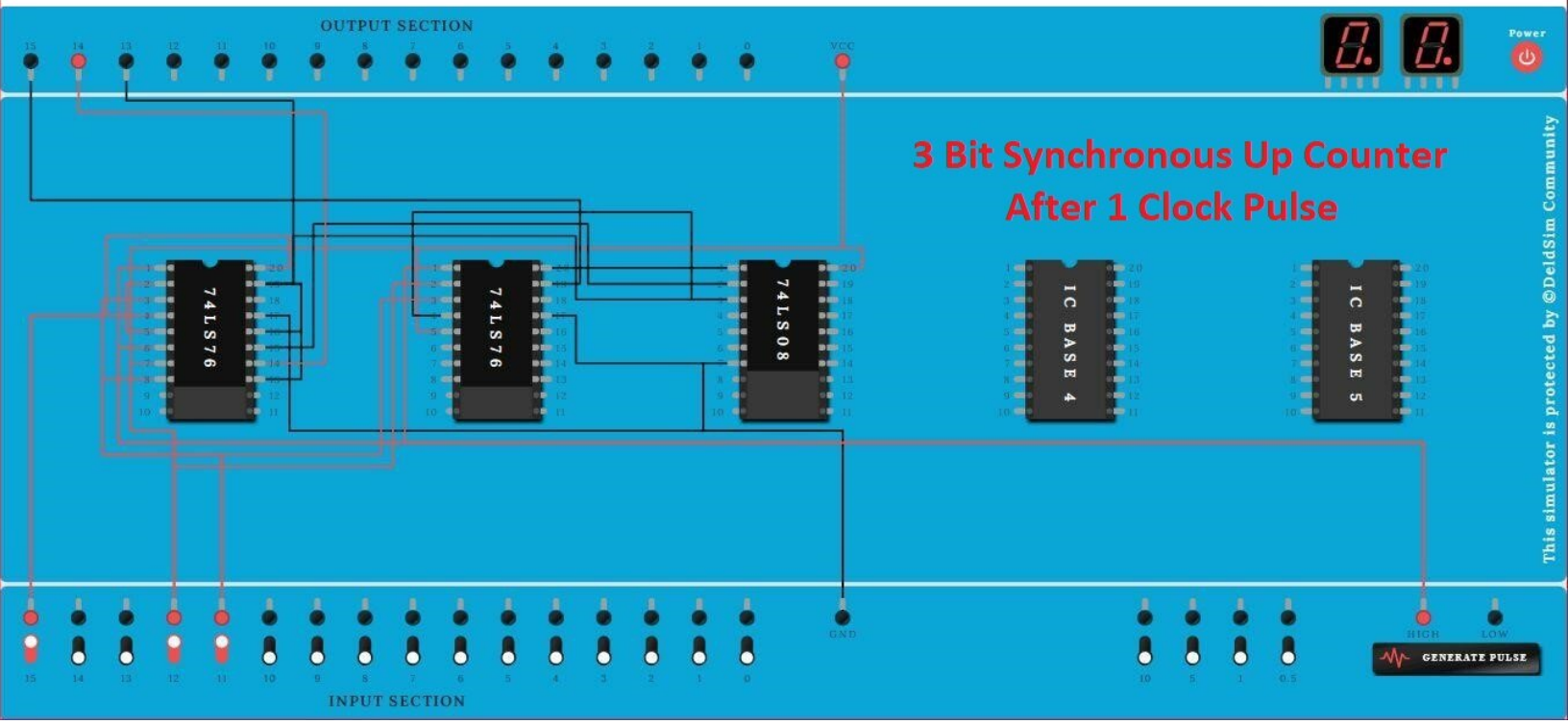
Logic Diagram and Truth Table :-

1) Up counter →

CP	Q_C	Q_B	Q_A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

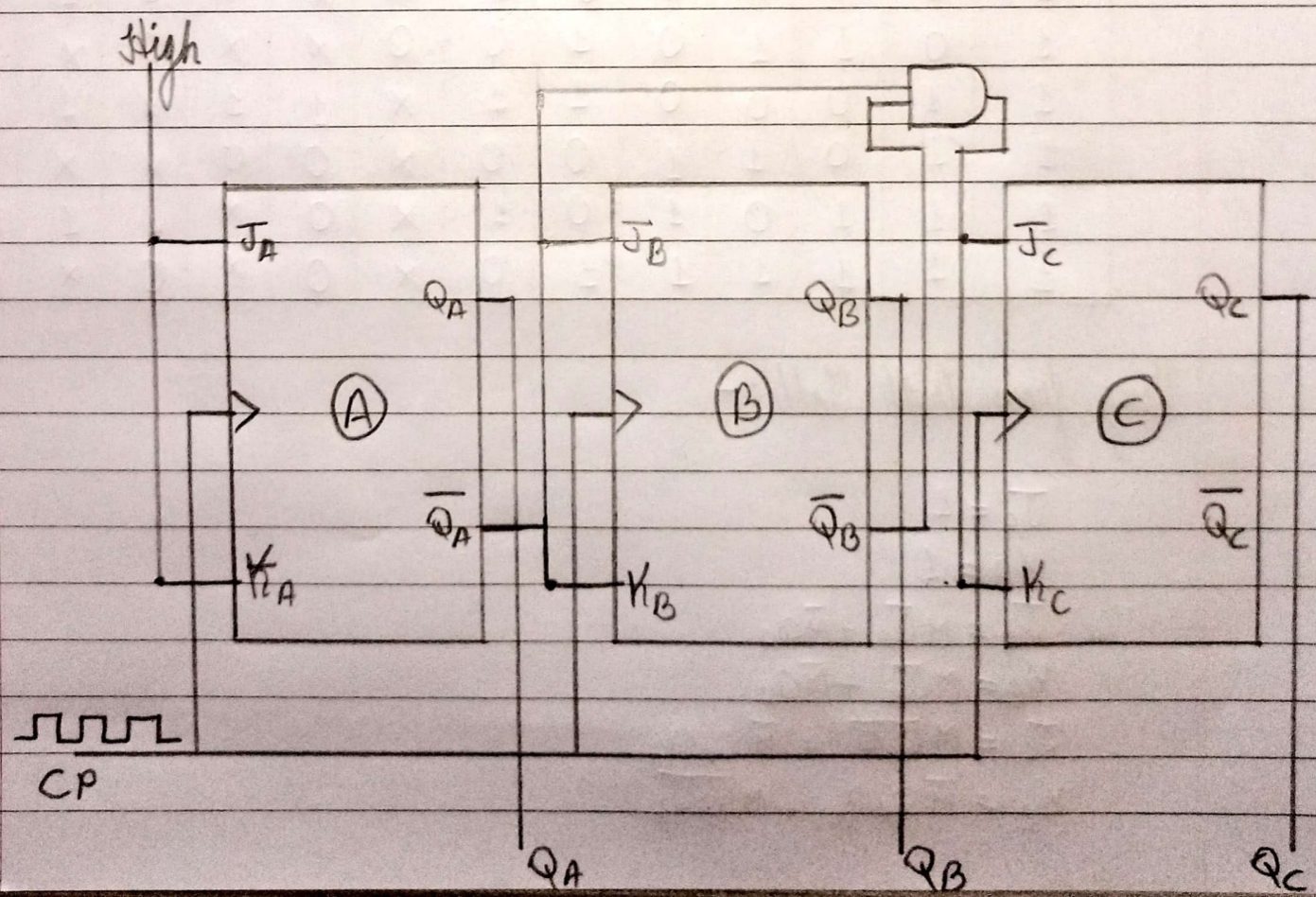


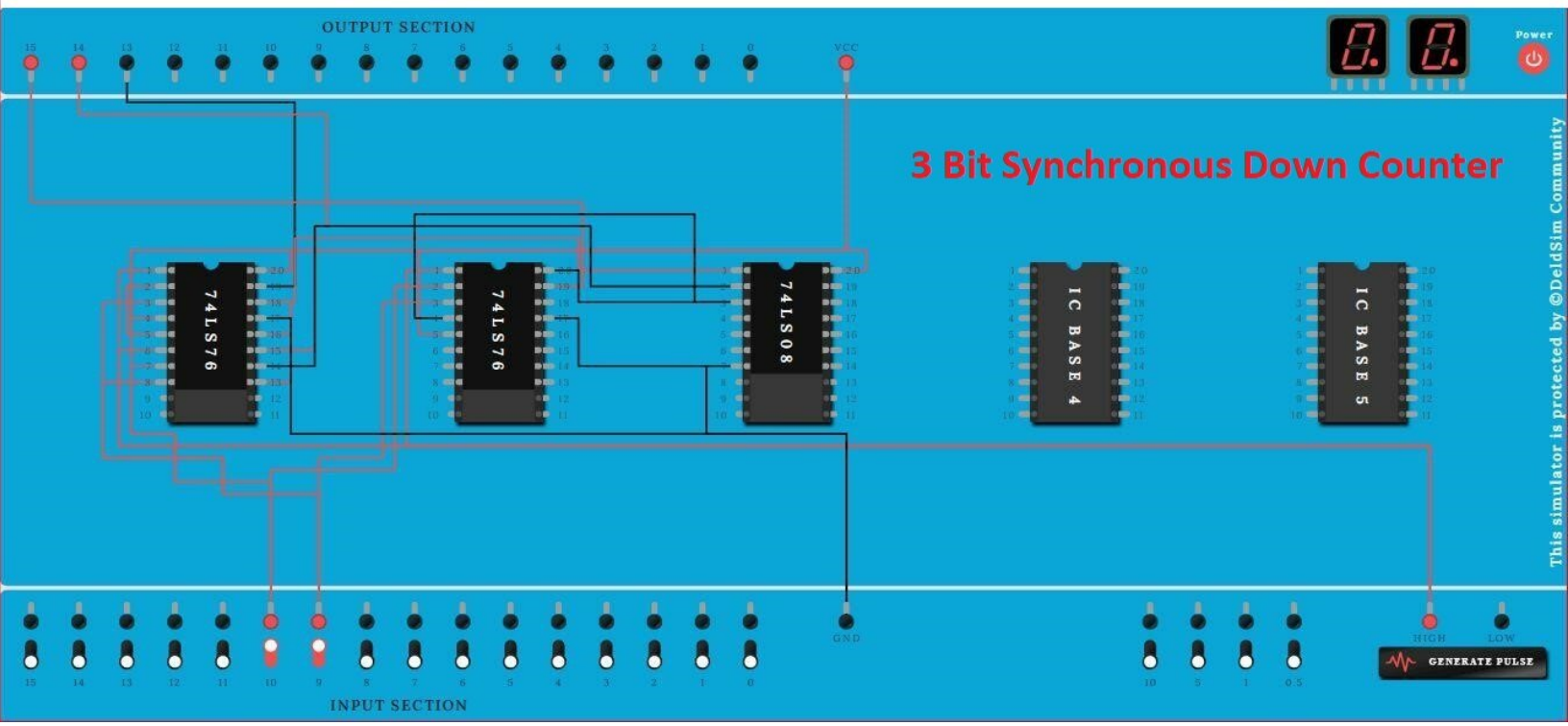

OUTPUT SECTION
VCC
Power
3 Bit Synchronous Up Counter
74LS76
74LS76
74LS08
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

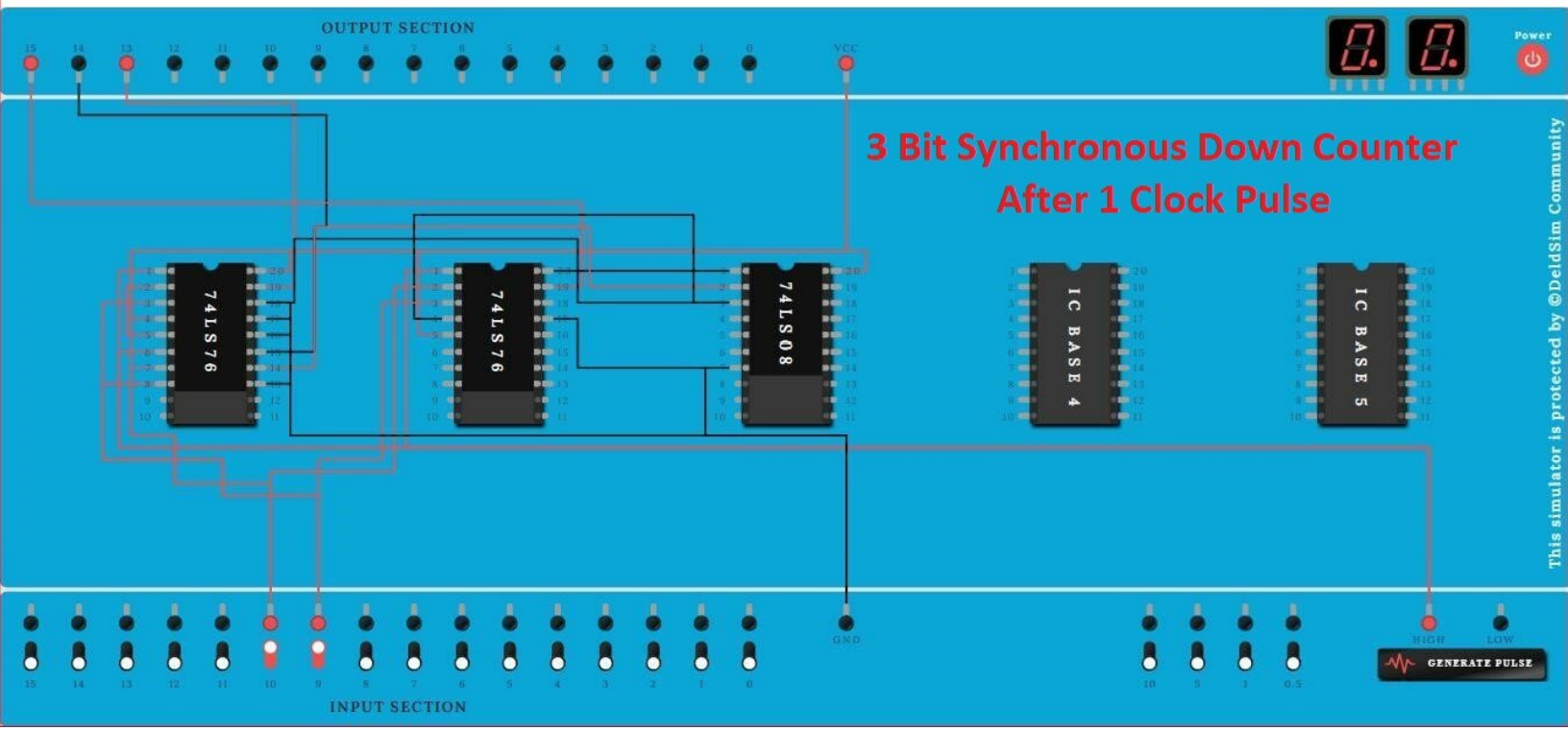

OUTPUT SECTION
VCC
Power
3 Bit Synchronous Up Counter
After 1 Clock Pulse
74LS76
74LS76
74LS08
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

2) Down counter →

CP	Q_C	Q_B	Q_A
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0




OUTPUT SECTION
VCC
Power
3 Bit Synchronous Down Counter
74LS76
74LS76
74LS08
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
VCC
Power
3 Bit Synchronous Down Counter
After 1 Clock Pulse
74LS76
74LS76
74LS08
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

3) Up-Down counter:-

Control Input	Present			Next			Input from flip flop					
M	Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	0	1	0	X	0	X	1	X
0	0	0	1	0	1	0	0	X	1	X	X	1
0	0	1	0	0	1	1	0	X	X	0	1	X
0	0	1	1	1	0	0	1	X	X	1	X	1
0	1	0	0	1	0	1	X	0	1	X	1	X
0	1	0	1	1	1	0	X	0	0	X	X	1
0	1	1	0	1	1	1	X	0	X	0	1	X
0	1	1	1	0	0	0	X	1	X	1	X	1
1	0	0	0	1	1	1	1	X	1	X	1	X
1	0	0	1	0	0	0	0	X	0	X	X	1
1	0	1	0	0	0	1	0	X	X	1	1	X
1	0	1	1	0	1	0	0	X	X	0	X	1
1	1	0	0	0	1	1	X	1	1	X	1	X
1	1	0	1	1	0	0	X	0	0	X	X	1
1	1	1	0	1	0	1	X	0	X	1	1	X
1	1	1	1	1	1	0	X	0	X	0	X	1

Using from truth Table:-

$J_A = 1$

$K_A = 1$

$J_B = M\bar{Q}_A + \bar{M}Q_A$

$K_B = M\bar{Q}_A + \bar{M}Q_A$

$J_C = M\overline{Q_A Q_B} + \bar{M}Q_A Q_B$

$K_C = \bar{M}Q_A Q_B + M\overline{Q_A Q_B}$.

Uses:-

1) Counting device
2) Count No. clock pulse.
3) Digital voltmeter
4) Used in digital triangular wave generator.

IC's used:-

1) Dual Master Slave JK flip flop (IC 7476)
2) AND gate (IC 7408)
3) OR gate (IC 7432).

Outcome:-

The up, down and up-down are successfully implemented, the counter are studied and output are checked. The truth table is verified.

Conclusion:-

Hence we have design 3 bit synchronous couter.

* Digital Electronics and Logic Design (DELD) - Practical Number - 11

Name :- Kaustubh Shrikant Kabra
Class :- Second Year Engineering
Div :- A Roll Number :-
Batch :-
Department :- Computer Department
College :- AISSMS's IOIT.

Title :-
Decade Counter

Aim :-
Realization of Mod-N counter using Decade Counter (IC7490).

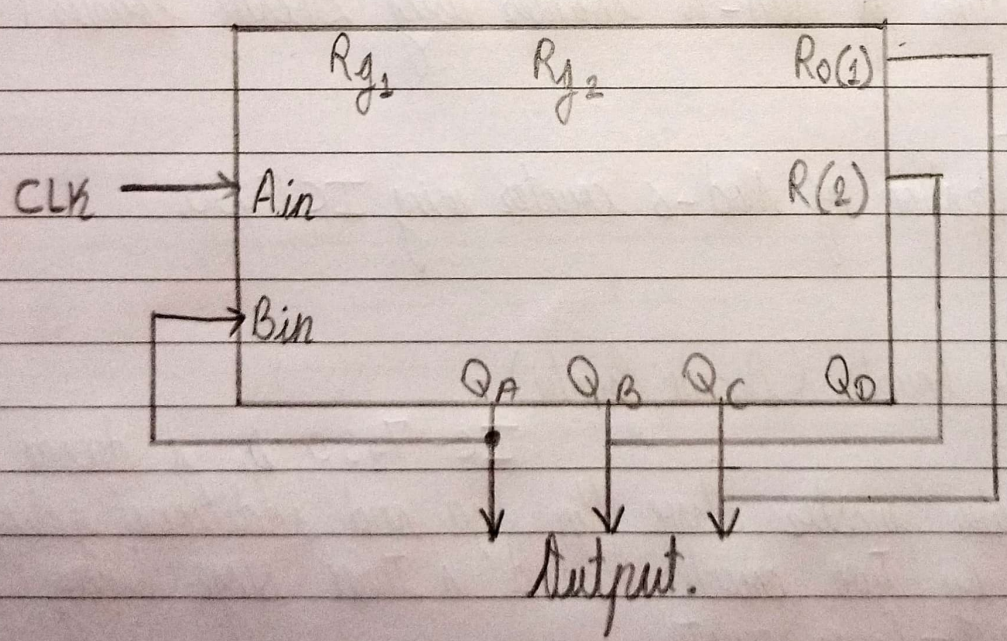
Objective :-
Realization of Mod-6 counter using IC 7490.

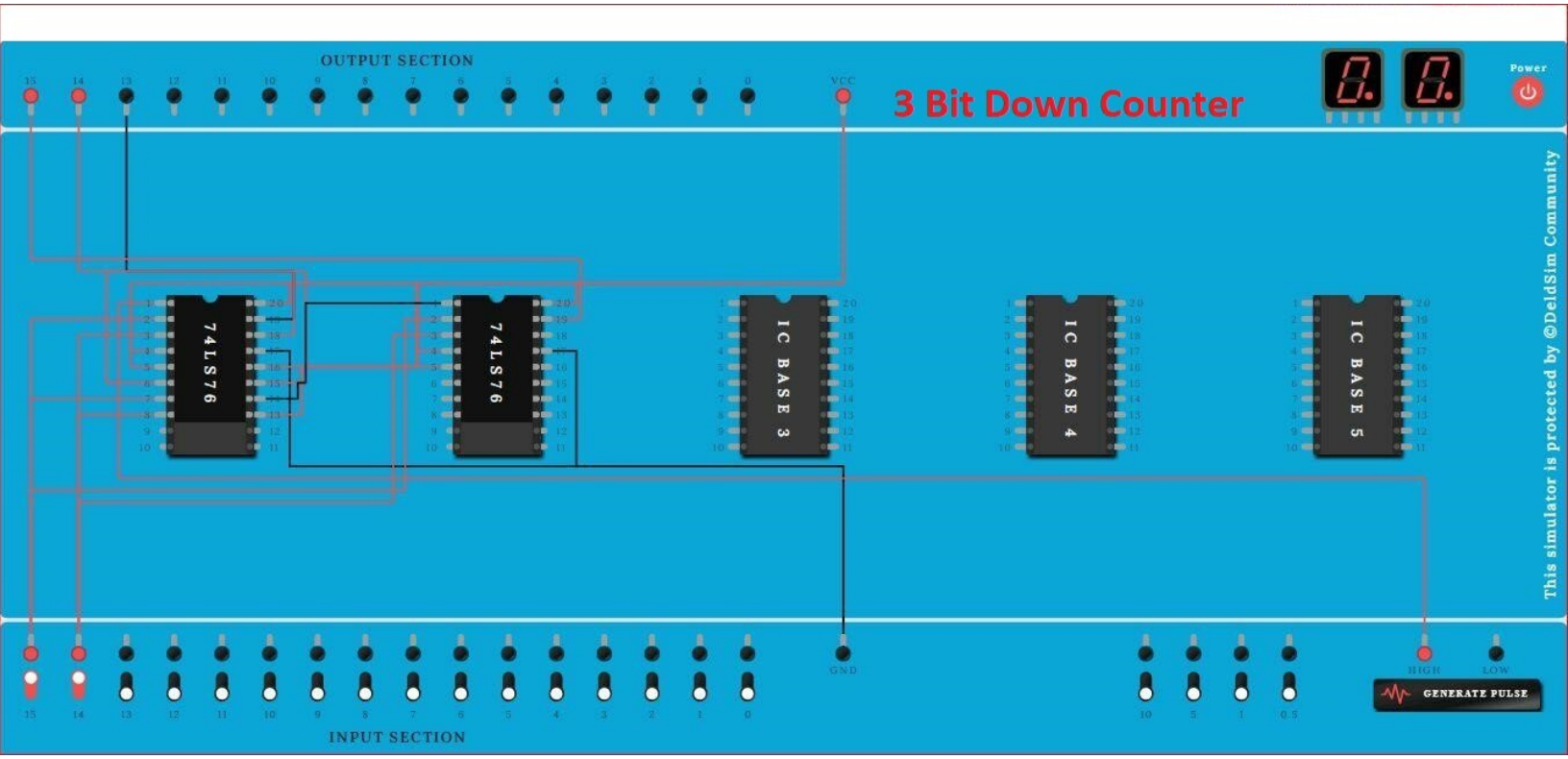
Theory :-
Ripple Counter (Decade Counter) -
IC-7490 is a decade counter, It contain four master slave flip flop and additional gating to provide a divide by two counter and a three stage binary counter which provide a ÷ by 5 counter.

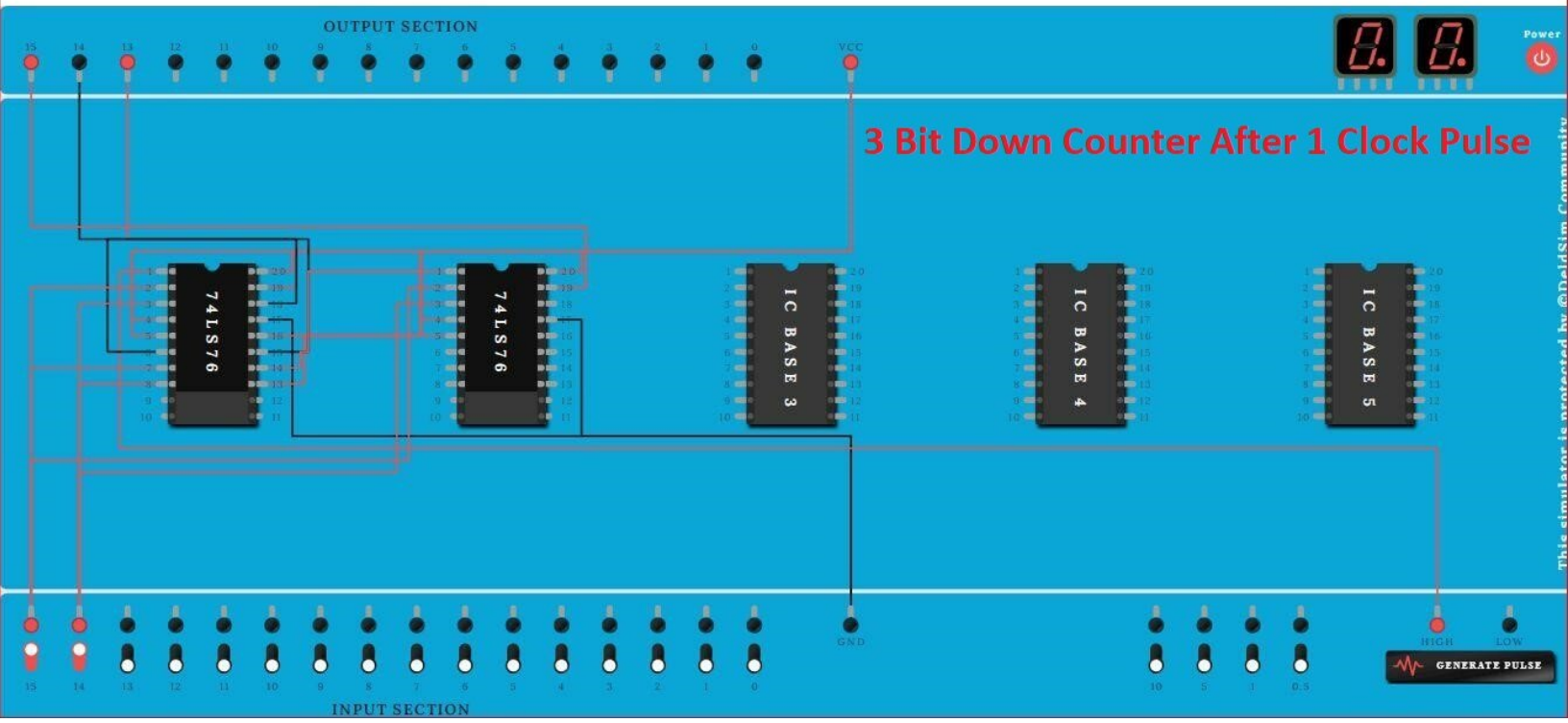
Mod-6 Counter:-

Mod counter are cascaded counter circuits which count to a set modulus value before resting Mod-6.

CLK Pulse	Q_A	Q_B	Q_C
Initial=0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	Reset to Zero		

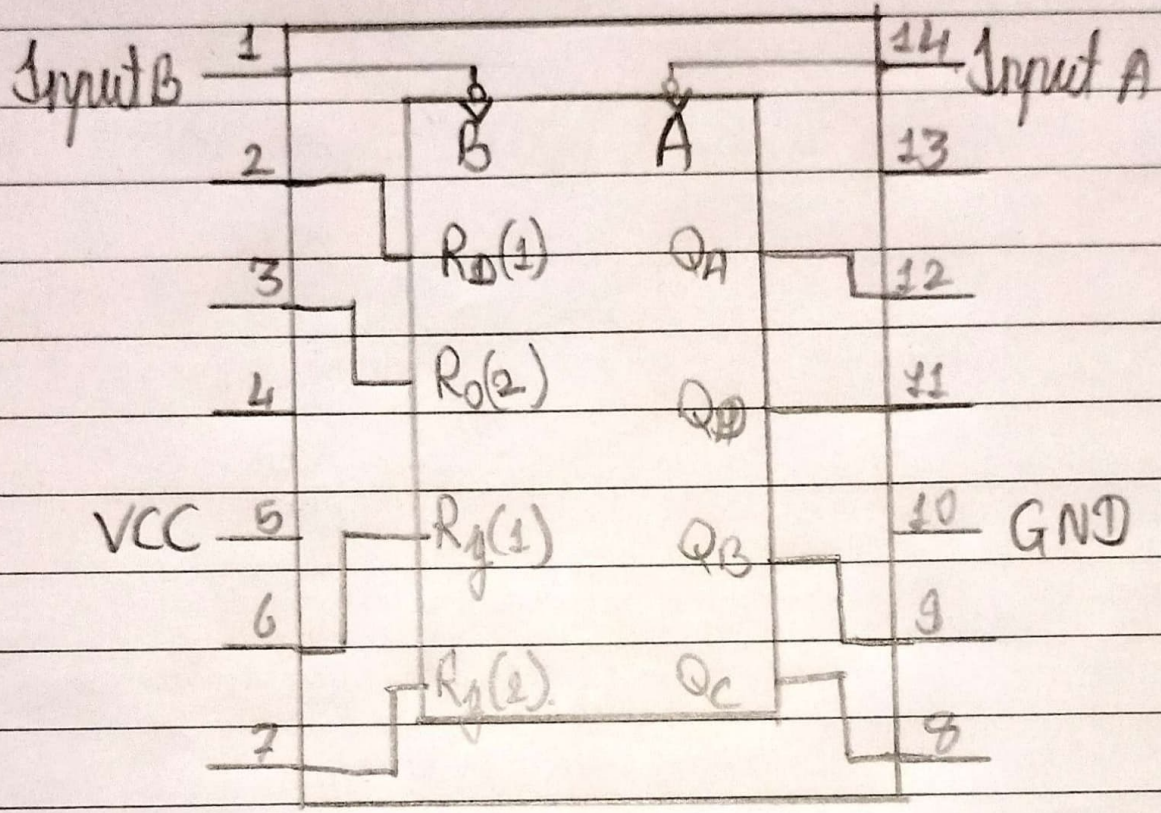



OUTPUT SECTION
VCC
3 Bit Down Counter
Power
74LS76
74LS76
IC BASE 3
IC BASE 4
IC BASE 5
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
VCC
3 Bit Down Counter After 1 Clock Pulse
Power
74LS76
74LS76
IC BASE 3
IC BASE 4
IC BASE 5
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

IC used:-

① IC 7490 [Decade counter]



Conclusion:-

Hence, we have realized a Mod-6 counter using IC 7490.

* Digital Electronics and Logic Design (DELD) - Practical Number - 12

Name :- Kaustubh Shrikant Kabra.
Class :- Second Year Engineering
Div :- A Roll Number :-
Batch :-
Department :- Computer Department
College :- AISSMS's IOIT.

Title :-
Sequence Generator.

Aim :-
Design and implement sequence generator using D flip-flop.

Objectives :-
Design sequence generator for :-
① Odd Sequence
② Even Sequence

Theory :-
For the design of sequence generator particular sequence can be determined as follows.
No. of flip-flop required to generate particular sequence can be determined as follows :-
① Find the number of 1's in the sequence
② Find the number of 0's in the sequence

③ Take the maximum out of two.

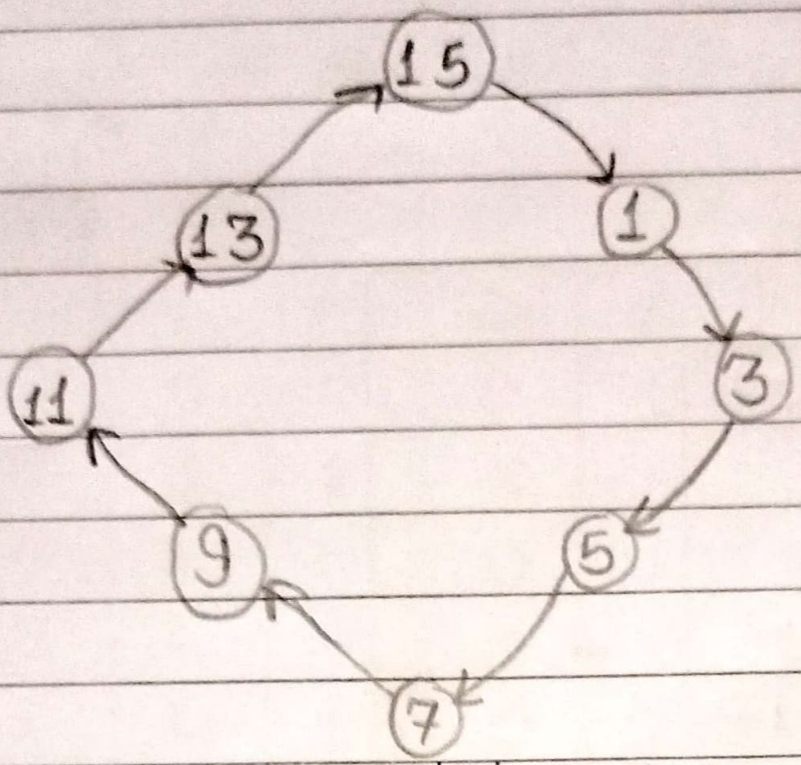
④ If N is the required no. of flip-flop, choose minimum value of 'n' to satisfy equation given below:

$$Max(0's, 1's) \leq 2^{n-1}$$

Truth Table :-

1) Sequence Generator for Odd Number (4-bit)

Present State				Next State			
Q_A	Q_B	Q_C	Q_D	Q_{A+1}	Q_{B+1}	Q_{C+1}	Q_{D+1}
0	0	0	0	X	X	X	X
0	0	0	1	0	0	1	1
0	0	1	0	X	X	X	0
0	0	1	1	0	1	0	1
0	1	0	0	X	X	X	X
0	1	0	1	0	1	1	1
0	1	1	0	X	X	X	X
0	1	1	1	1	0	0	1
1	0	0	0	X	X	X	X
1	0	0	1	1	0	1	1
1	0	1	0	X	X	X	X
1	0	1	1	1	1	0	1
1	1	0	0	X	X	X	X
1	1	0	1	1	1	1	1
1	1	1	0	X	X	X	X
1	1	1	1	0	0	0	1



Simplification using truth table and kmaps :-

$$D_A = Q_A \overline{Q}_C + Q_A \overline{Q}_B + \overline{Q}_A Q_B Q_C$$

$$D_B = Q_B \overline{Q}_C + \overline{Q}_B Q_C$$

$$D_C = \overline{Q}_C$$

$$D_D = 1.$$

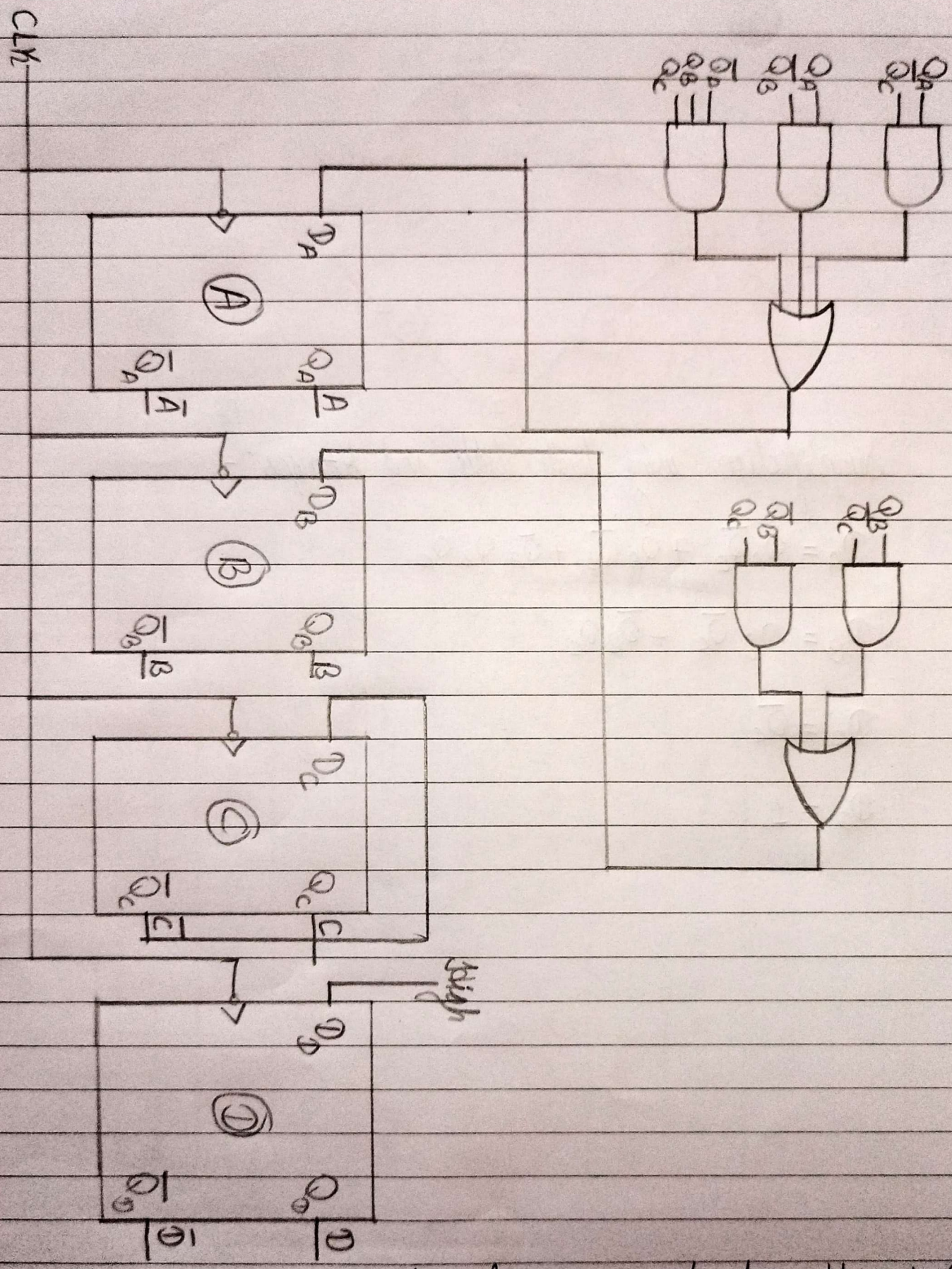
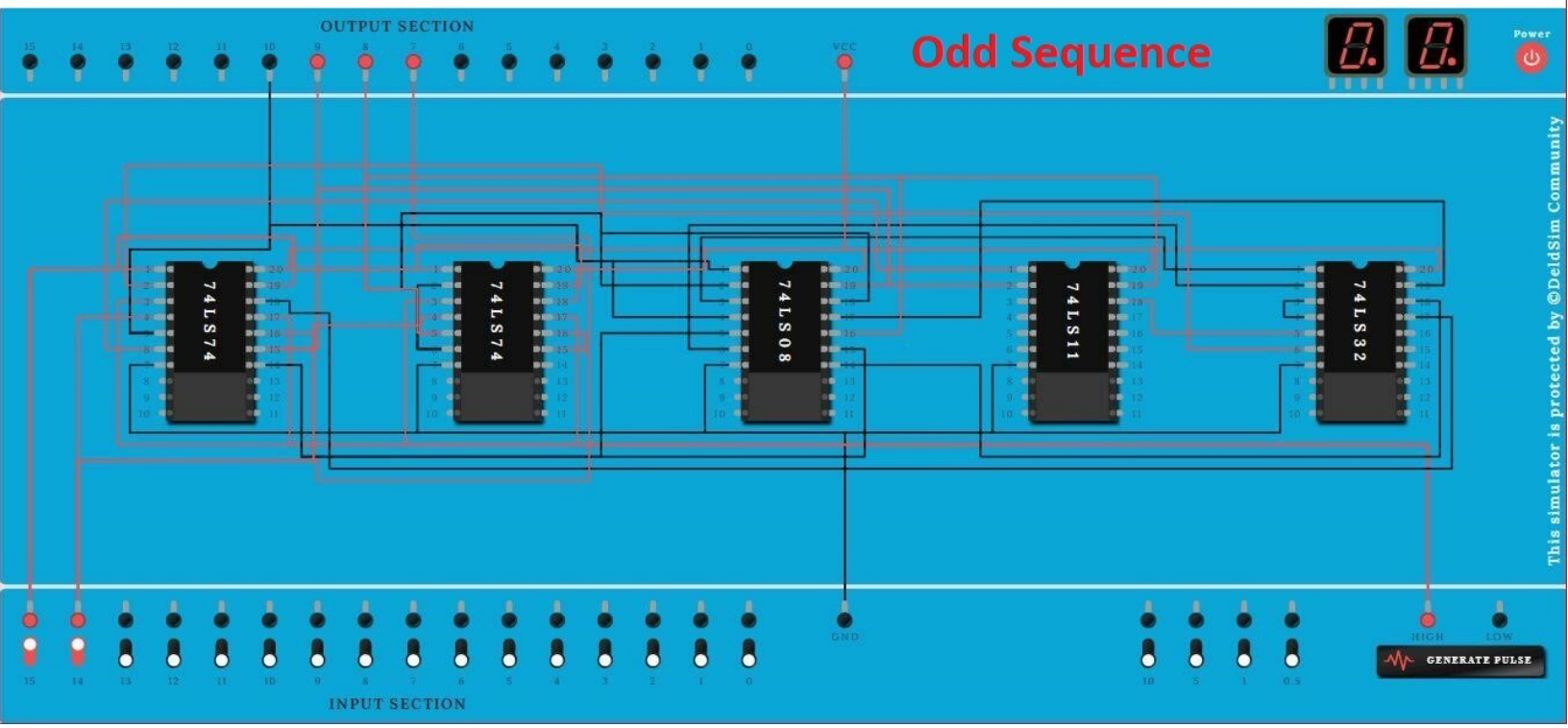
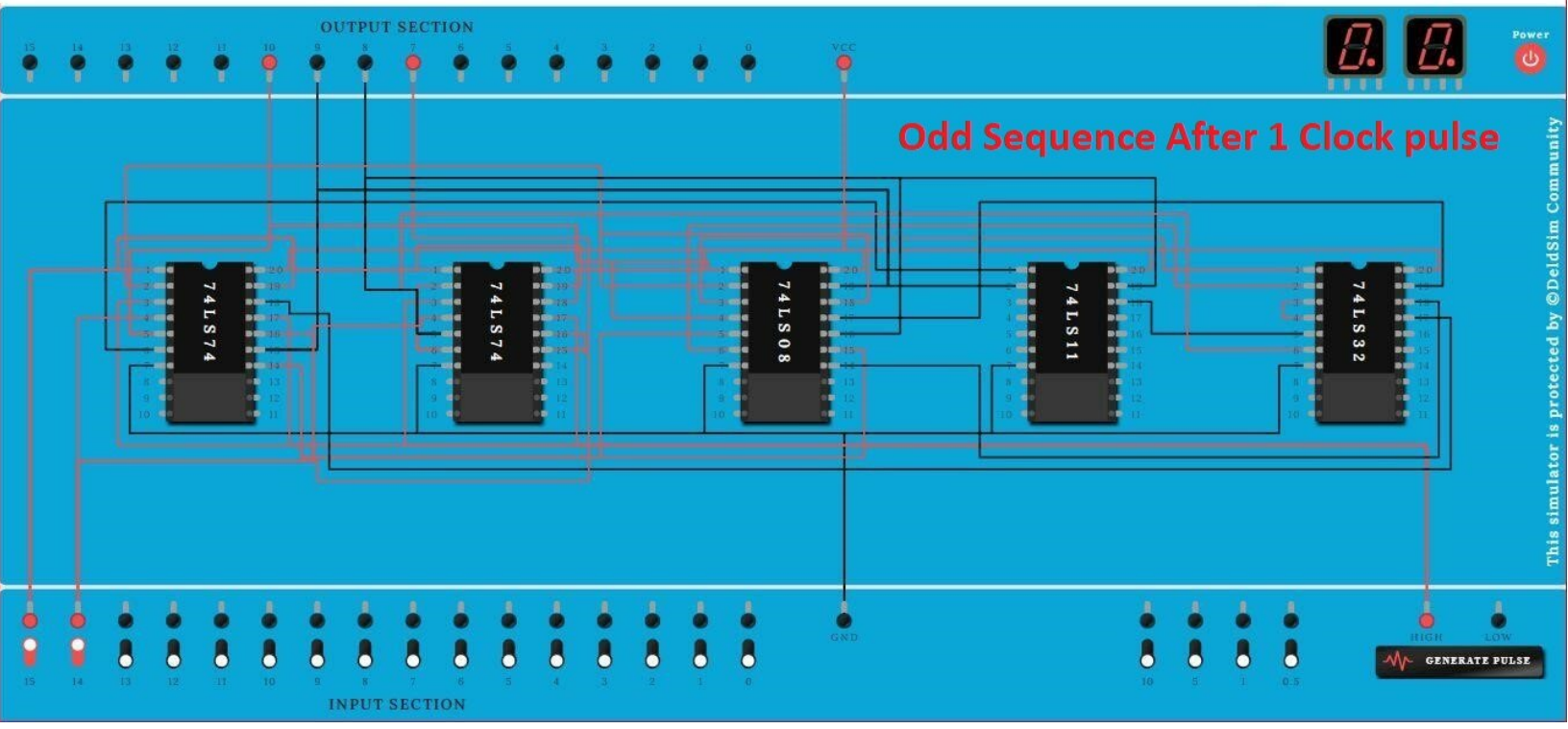


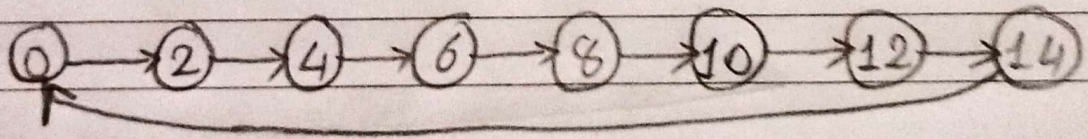
fig:- Sequence generator for odd number (4-bit).


OUTPUT SECTION
Odd Sequence
Power
VCC
74LS74
74LS74
74LS08
74LS11
74LS32
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION
This simulator is protected by ©DeldSim Community


OUTPUT SECTION
Odd Sequence After 1 Clock pulse
Power
VCC
74LS74
74LS74
74LS08
74LS11
74LS32
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION
This simulator is protected by ©DeldSim Community

2) Sequence Generator for even number (4-bit)-

Present State				Next State			
Q_A	Q_B	Q_C	Q_D	Q_{A+1}	Q_{B+1}	Q_{C+1}	Q_{D+1}
0	0	0	0	0	0	1	0
0	0	0	1	X	X	X	X
0	0	1	0	0	1	0	0
0	0	1	1	X	X	X	X
0	1	0	0	0	1	1	0
0	1	0	1	X	X	X	X
0	1	1	0	1	0	0	0
0	1	1	1	X	X	X	X
1	0	0	0	1	0	1	0
1	0	0	1	X	X	X	X
1	0	1	0	1	1	0	0
1	0	1	1	X	X	X	X
1	1	0	0	1	1	1	0
1	1	0	1	X	X	X	X
1	1	1	0	0	0	0	0
1	1	1	1	X	X	X	X



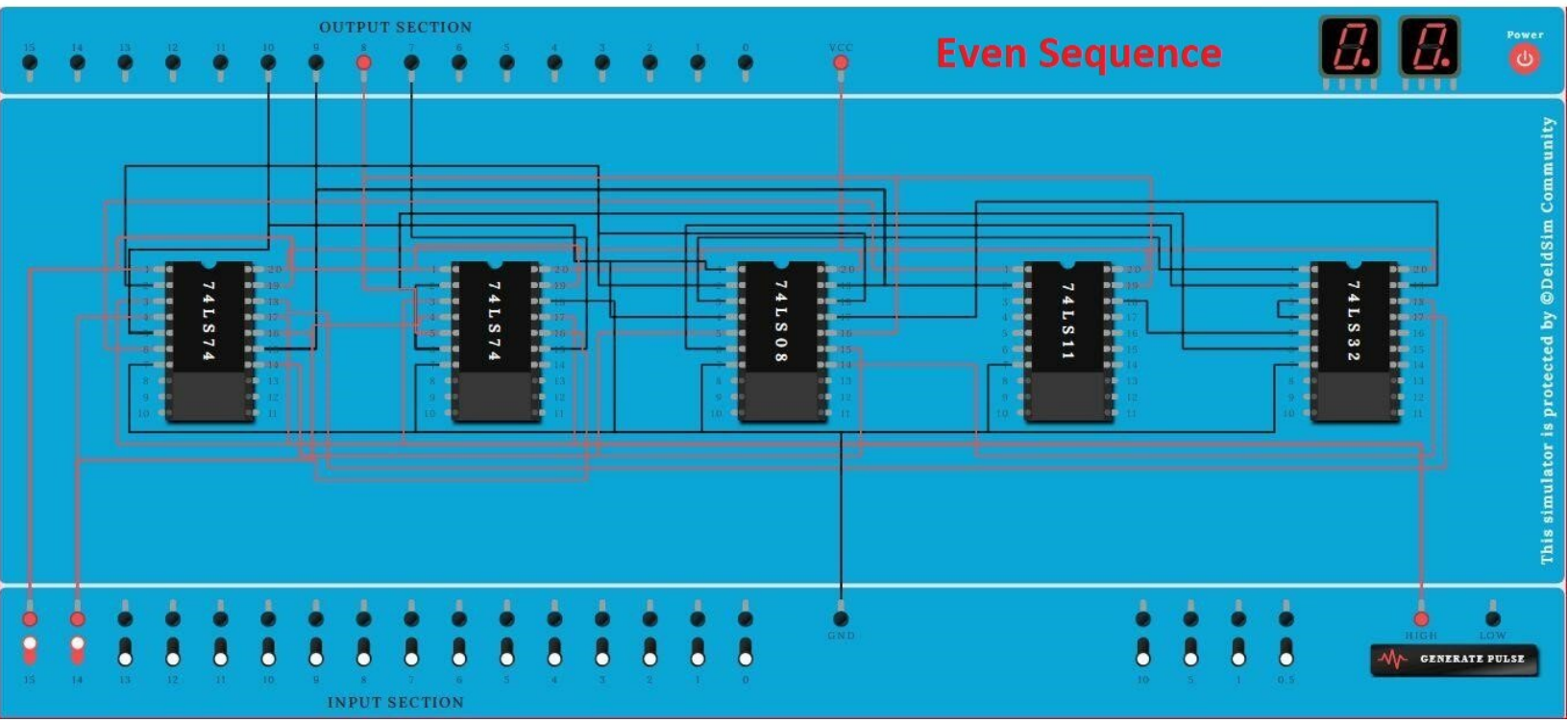
Using truth table and kmap for simplification.

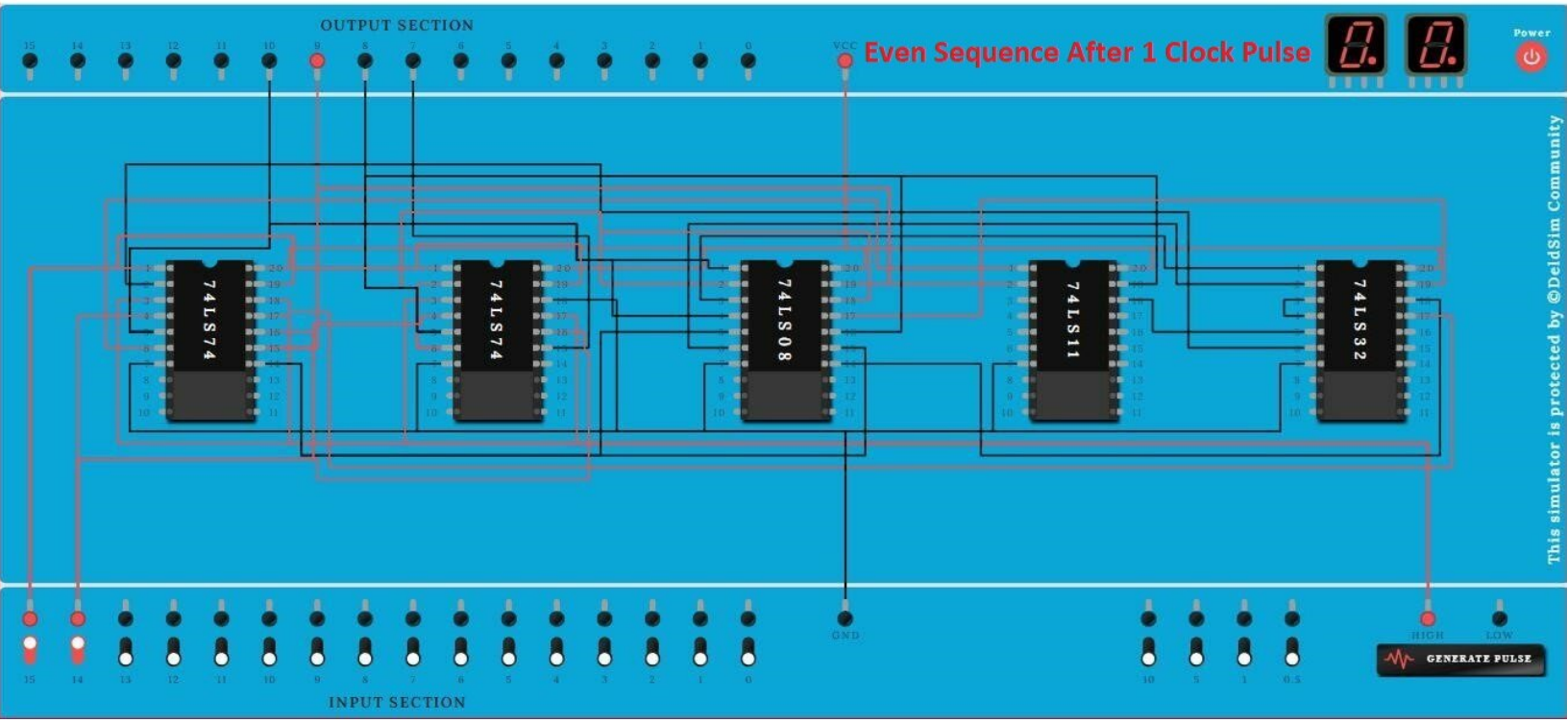
$D_A = Q_A\bar{Q}_C + Q_A\bar{Q}_B + \bar{Q}_A Q_B Q_C$

$D_B = Q_B\bar{Q}_C + \bar{Q}_B Q_C$

$D_C = \bar{Q}_C$

$D_D = 0$


OUTPUT SECTION
Even Sequence
Power
VCC
74LS74
74LS74
74LS08
74LS11
74LS32
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION


OUTPUT SECTION
Even Sequence After 1 Clock Pulse
Power
VCC
74LS74
74LS74
74LS08
74LS11
74LS32
This simulator is protected by ©DeldSim Community
GND
HIGH
LOW
GENERATE PULSE
INPUT SECTION

Conclusion:-

Hence, we have design and implimented the sequence generator using D flip flop.

* Digital Electronics and Logic Design (DELD) - Practical Number - 13

Name :- Kaustubh Shrikant Kabra.
Class :- Second Year Engineering.
Div :- A Roll Number :-
Batch :-
Department :- Computer Department
College :- AISSMS's IOIT.

Title :-
Shift Registers.

Aim :-
Study of Shift Registers [SISO, SIPO, PISO, PIPO]

Objective :-
The study the working of shift registers.

Theory :-
Mode of operation of a shift register: The various mode in which a shift register can operate are as follows :-

1. Serial input Serial Output [SISO]
2. Serial Input Parallel Output [SIPO]
3. Parallel Input Serial Output [PISO]
4. Parallel Input Parallel Output [PIPO]

1) Serial In Serial Out [SISO] :-

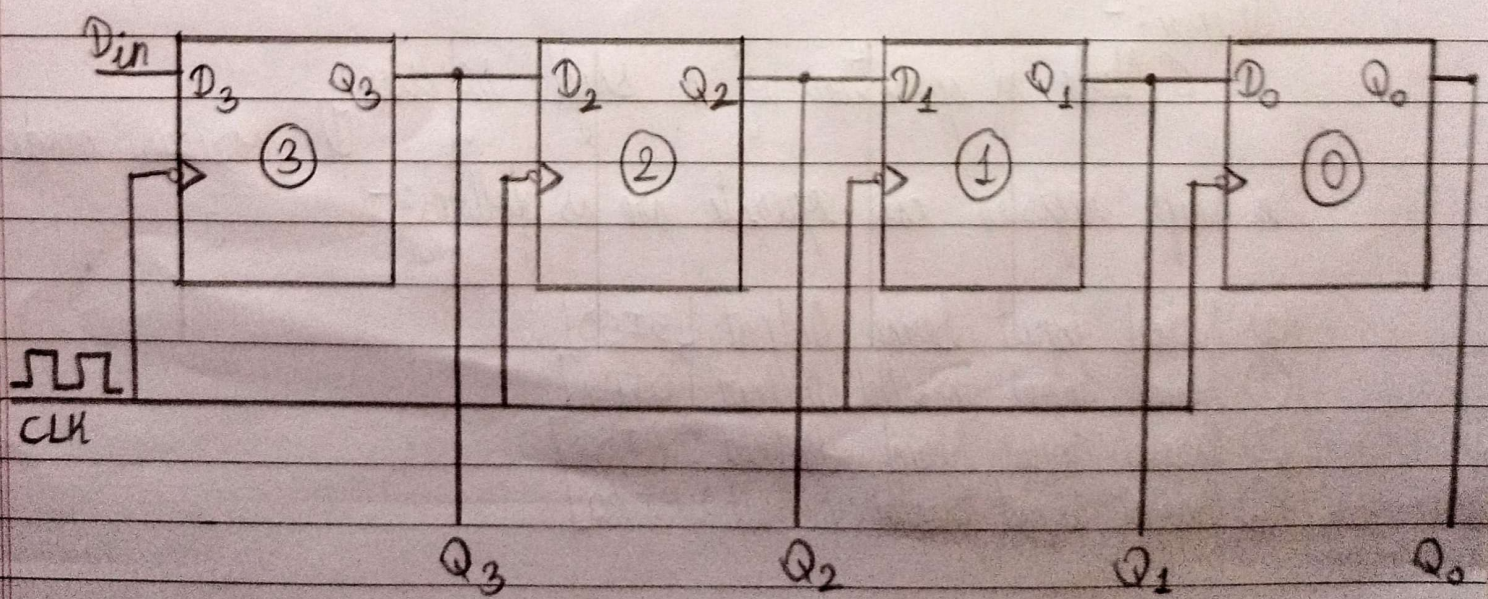
C.P	Q_3	Q_2	Q_1	Q_0	D_{in}
Initially	0	0	0	0	1
↓1	0	0	0	1	1
↓2	0	0	1	1	1
↓3	0	1	1	1	1
↓4	1	1	1	1	1

Shift Left Register

C.P	D_{in}	Q_3	Q_2	Q_1	Q_0
Initially	1	0	0	0	0
↓1	1	1	0	0	0
↓2	1	1	1	0	0
↓3	1	1	1	1	0
↓4	1	1	1	1	1

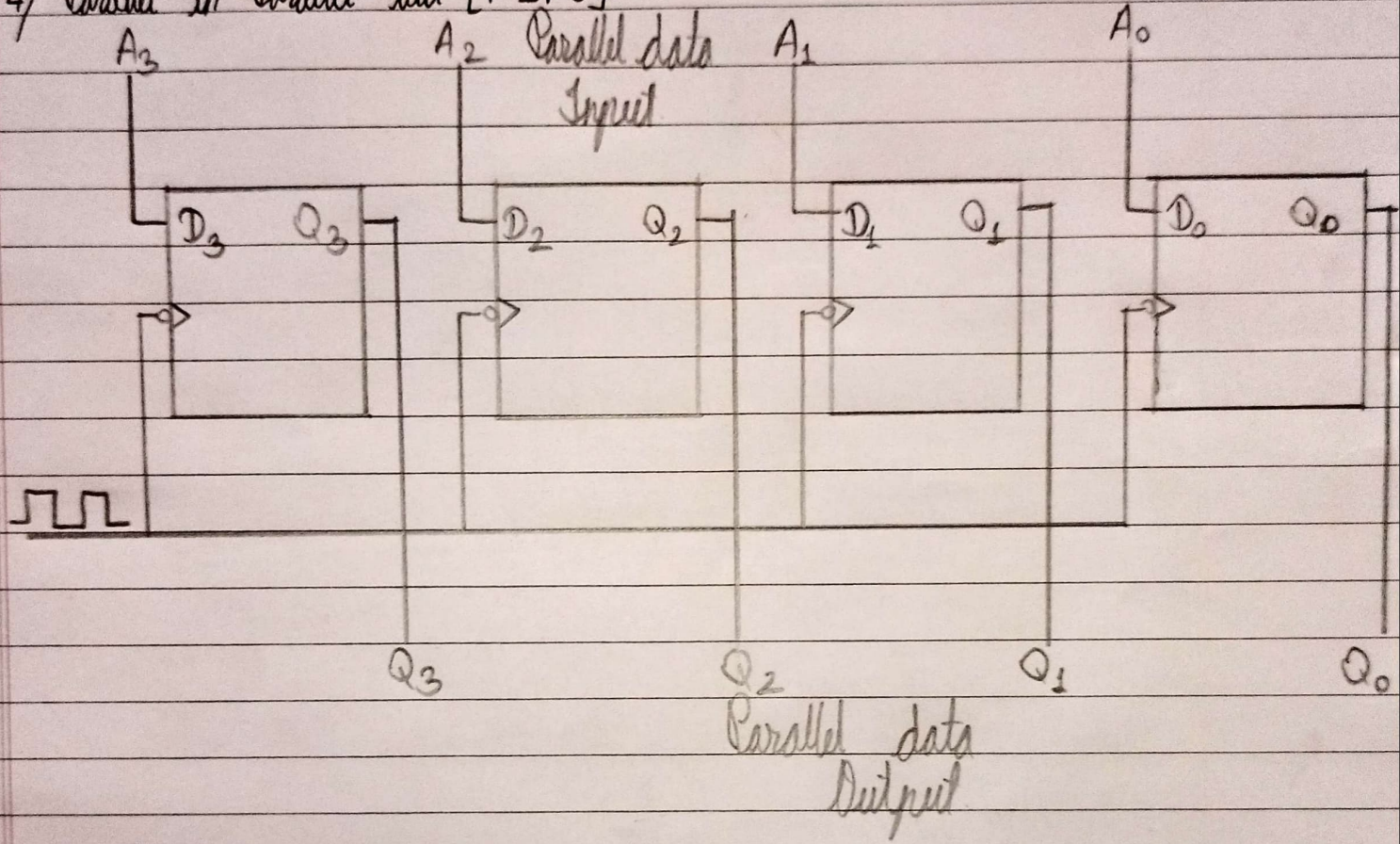
Shift Right Register

2) Serial In Parallel Out [SIPO] :-



CP	Q_3	Q_2	Q_1	Q_0
–	NC	NC	NC	NC
↓	D_3	D_2	D_1	D_0

4) Parallel in Parallel Out [PIPO]:-



Application of Shift Register:-

1. Temporary data storage.
2. Delay line
3. Serial-to Parallel Converter
4. Parallel-to Serial Converter
5. Counter
6. Sequence Generator

Conclusion:-

Hence, we have studied the shift register and its different types.