

Practical DF

Name: TANU RAWAT

Enrollment No: D2D1

DIVISION: A

Department : Computer

PRACTICAL 1

Getting familiar with Logisim, studying implementing all basic logic gates. Implement NAND and NOR as universal gates.

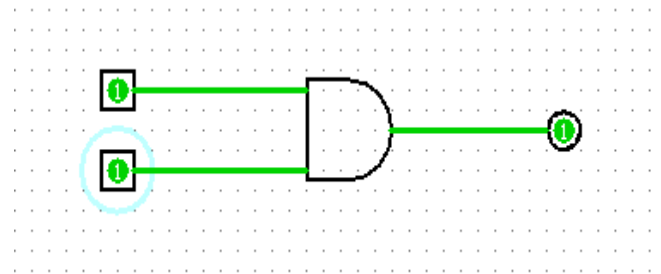
Basic Gate :

AND GATE

- and gate has two or more inputs but only one output
- here if both inputs are 1 then output is 1 else output is 0(zero).
- NOTATION : $c = A * B$

Truth Table

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

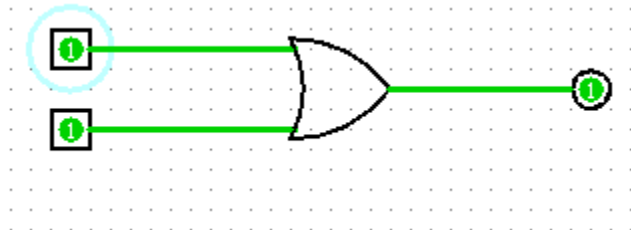


OR GATE

- or gate has two or more inputs but only one output
- here, if both inputs are 0 then the output is 0.
- NOTATION : $A + B$

Truth Table

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1



NOT GATE

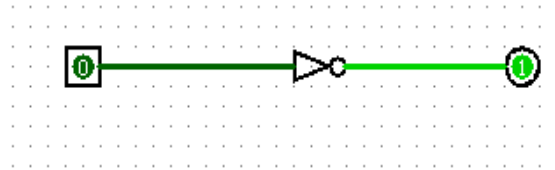
- It is also called an “inverter”.
- Here output is always a complement of input.
- If the input is 1 then the output is 0.

● If the input is 0 then the output is 1.

● NOTATION : $c = A'$

Truth Table

<u>A</u>	<u>C</u>
<u>1</u>	<u>0</u>
<u>0</u>	<u>1</u>

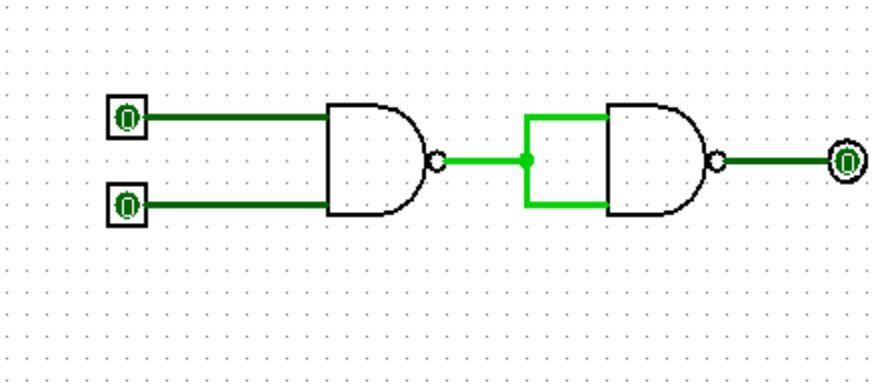


Universal Gate :

Using Nand gate

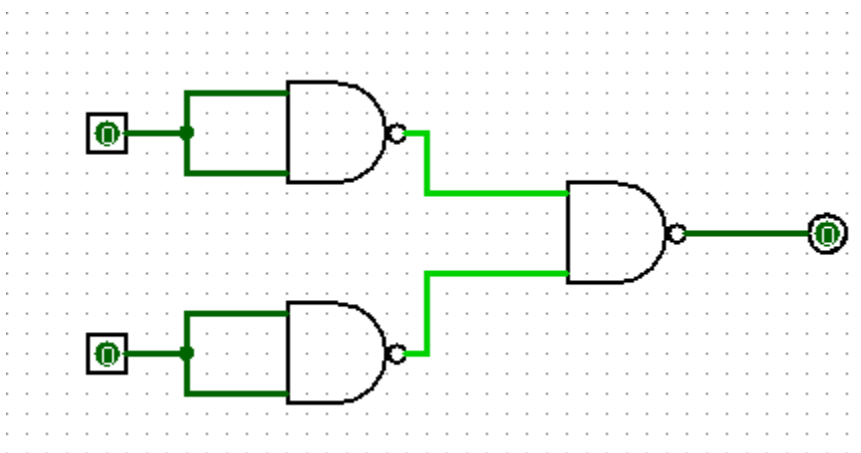
AND GATE :

*Notation: $A * B$*



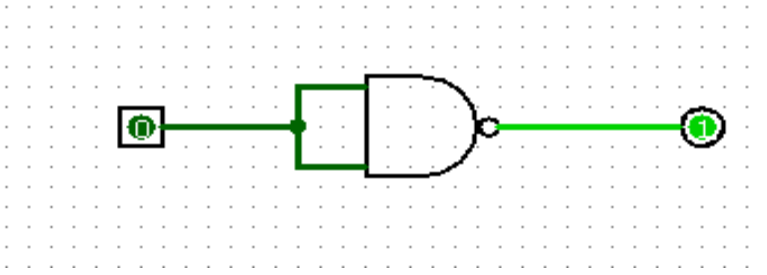
OR GATE :

Notation : $A + B$



NOT GATE :

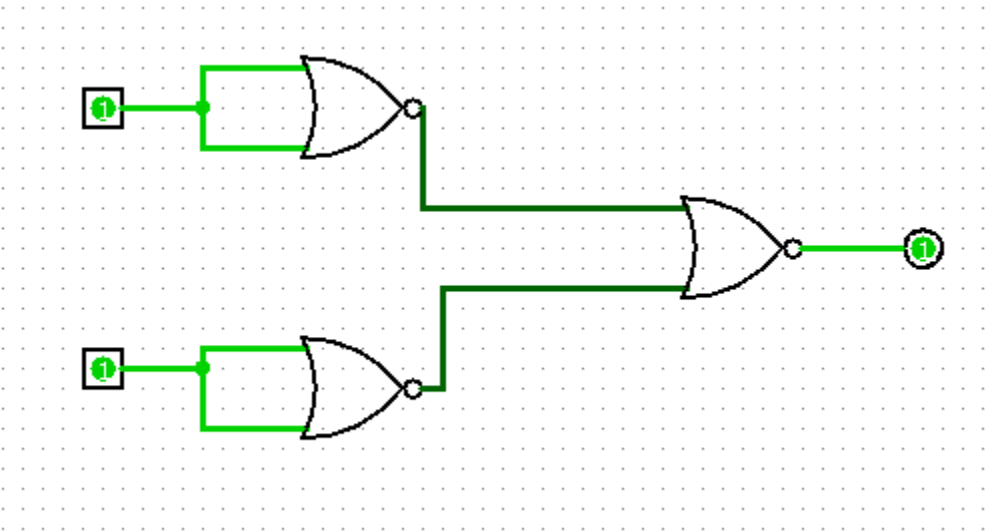
Notation: A'



Using Nor gate

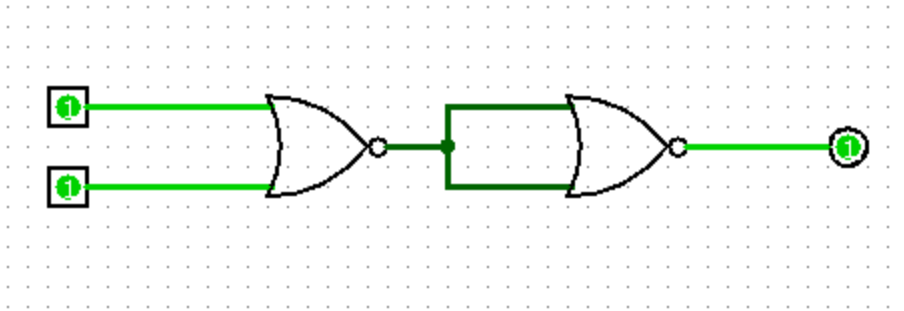
AND GATE :

*Notation: $A * B$*



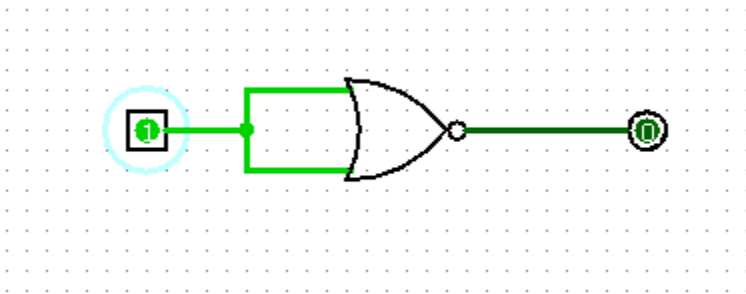
OR GATE :

Notation : $A + B$



NOT GATE :

Notation: A'



PRACTICAL 2

implement half and full adder using logic gates.

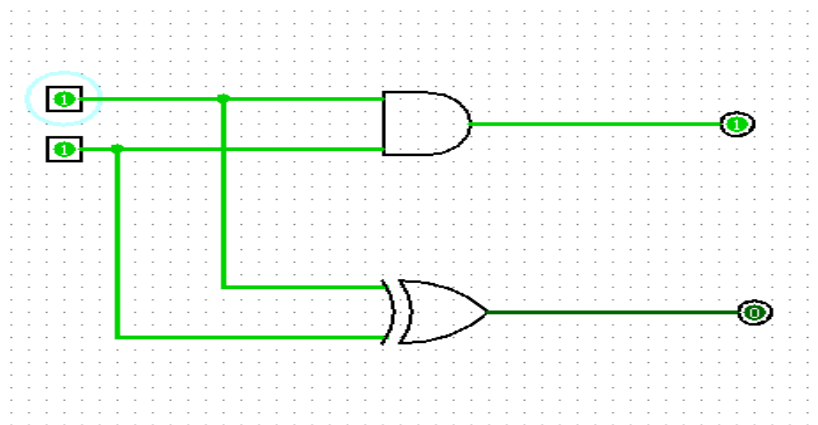
HALF ADDER

- A half adder is a combinational circuit with two i/p & two o/p.
- It adds the inputs and produces the sum(s) and the carry(c) bits.
- The sum (s) is the X-OR therefore, $S = AB' + BA' = A \text{ XOR } B$
- The carry (c) is the AND therefore, $C = AB$

Truth Table

Input		Output	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuit :



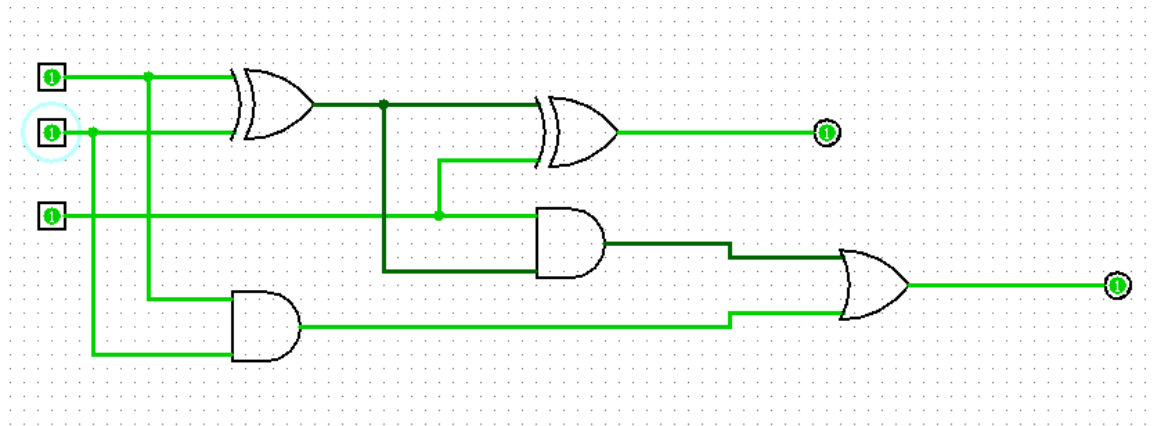
FULL ADDER

- A full adder is a combinational circuit that adds two bits and a carry and outputs a sum bit and a carry it.
- The full adder adds the bits A and B and the carry from the previous column called the carry-in cin and output the sum bit s and the carry bit called the carry out cout.
- The variable S gives the value of the least significant bit of the sum.
- $S = A'B'Cin + A'BCin + AB'Cin + ABCin$
 $A \text{ XOR } B \text{ XOR } Cin$
- $Carry \text{ Cout} = A'BCin + AB'Cin + ABCin$
 $AB + (A \text{ XOR } B)Cin$

TRUTH TABLE

Input			Output	
A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Circuit :



PRACTICAL 3

implement half and full subtractors using logic gates

HALF SUBTRACTOR

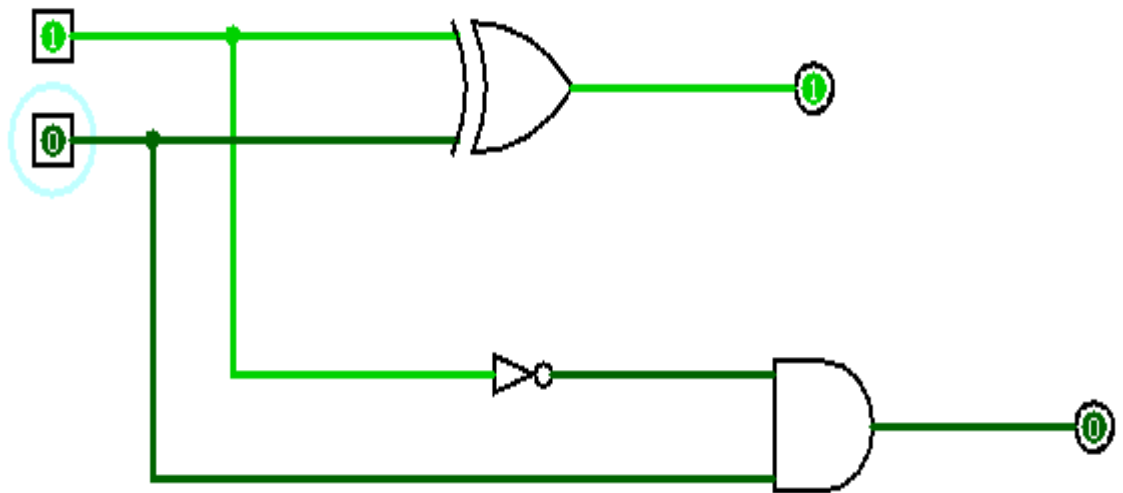
- a half subtractor is a combinational circuit that subtracts one bit from the other and produces the difference.
- It also has an output to specify if a 1 has been borrowed.
- Difference $D = AB' + BA' = A \text{ XOR } B$
- Borrow $B = A'B$

Truth Table

Input		output	
A	B	d	b
0	0	0	0
0	1	1	1
1	0	1	0

1	1	0	0
---	---	---	---

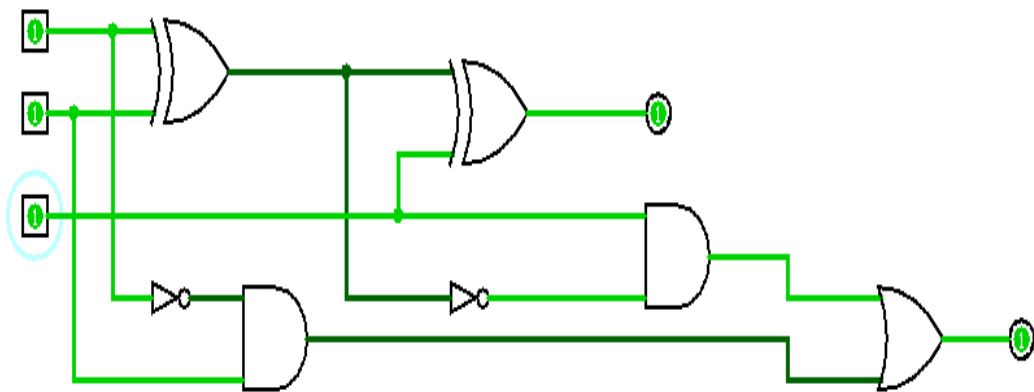
Circuit :



-

FULL SUBTRACTOR

- The half subtractor can be used only for LSB subtraction.
- If there is a borrow during the subtraction of the LSBs, it affects the subtraction in the next higher column; the subtrahend bit is subtracted from the minuend bit, considering the borrow from that column used for the subtraction in the preceding column.
- Such a subtraction is performed by a full subtractor.
- Difference $D = A'B'bin + A'Bbin + AB'bin' + ABbin$
 $= A \text{ XOR } B \text{ XOR } bin$
- Borrow $b = A'B'bin + A'Bbin' + A'Bbin + ABbin$
 $= A'B + (A \text{ XOR } B)' bin$



Truth Table :

Input			output	
A	B	bin	d	b
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0

1	1	0	0	0
1	1	1	1	1

PRACTICAL 4

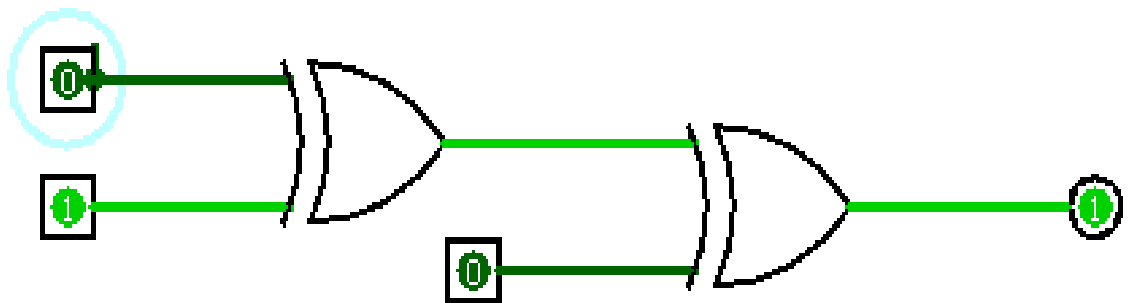
perform parity checker

Parity checker

- the parity checker is needed to detect errors in communication and also in the memory storage devices parity checker is used for testing.
- $F = A'B'C + A'BC' + ABC + AB'C' = A \text{ XOR } B \text{ XOR } C$

Truth table :

Input			Output parity bit (f)
A	B	C	
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



PRACTICAL 5

study and implement Multiplexer and Demultiplexer

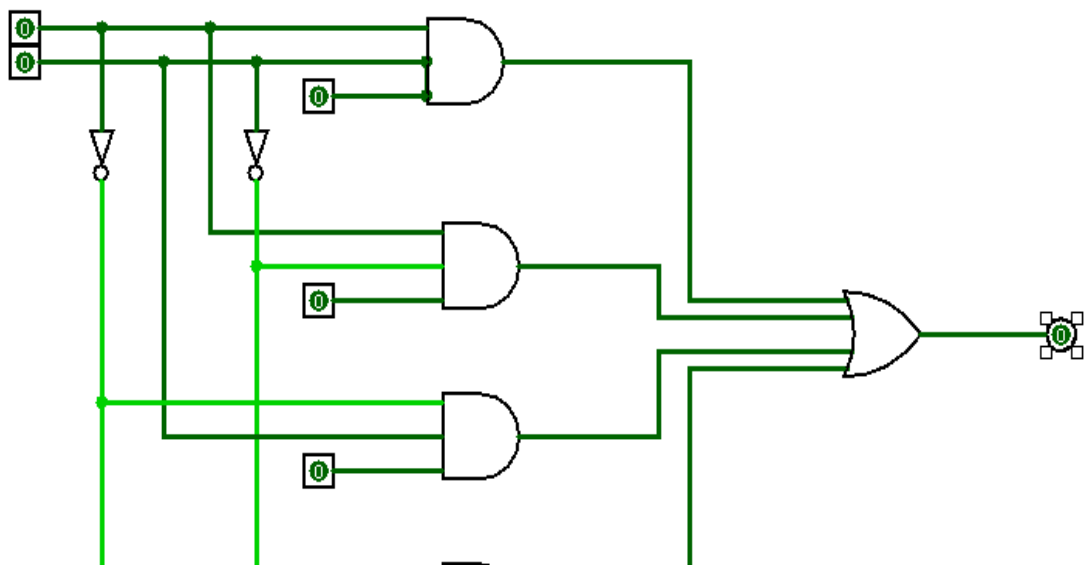
Multiplexer

- it is a combinational circuit that has a maximum of 2^n data inputs, 'n' selection lines, and a single output line. One of these data inputs will be connected to the output based on the values of the selection lines.

Truth Table :

Input		output
S1	S0	Y
0	0	Y1
0	1	Y2
1	0	Y3
1	1	Y4

Circuit :



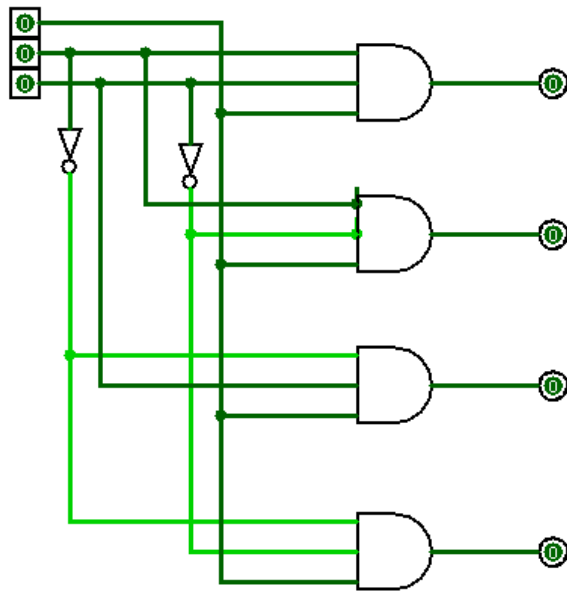
Demultiplexer

- a demultiplexer performs the reverse operation of a multiplexer; it takes a single input and distributes it over several outputs.
- Demux is a 1 to N device.

Truth table :

input		output			
S1	S0	Y1	Y2	Y3	Y4
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

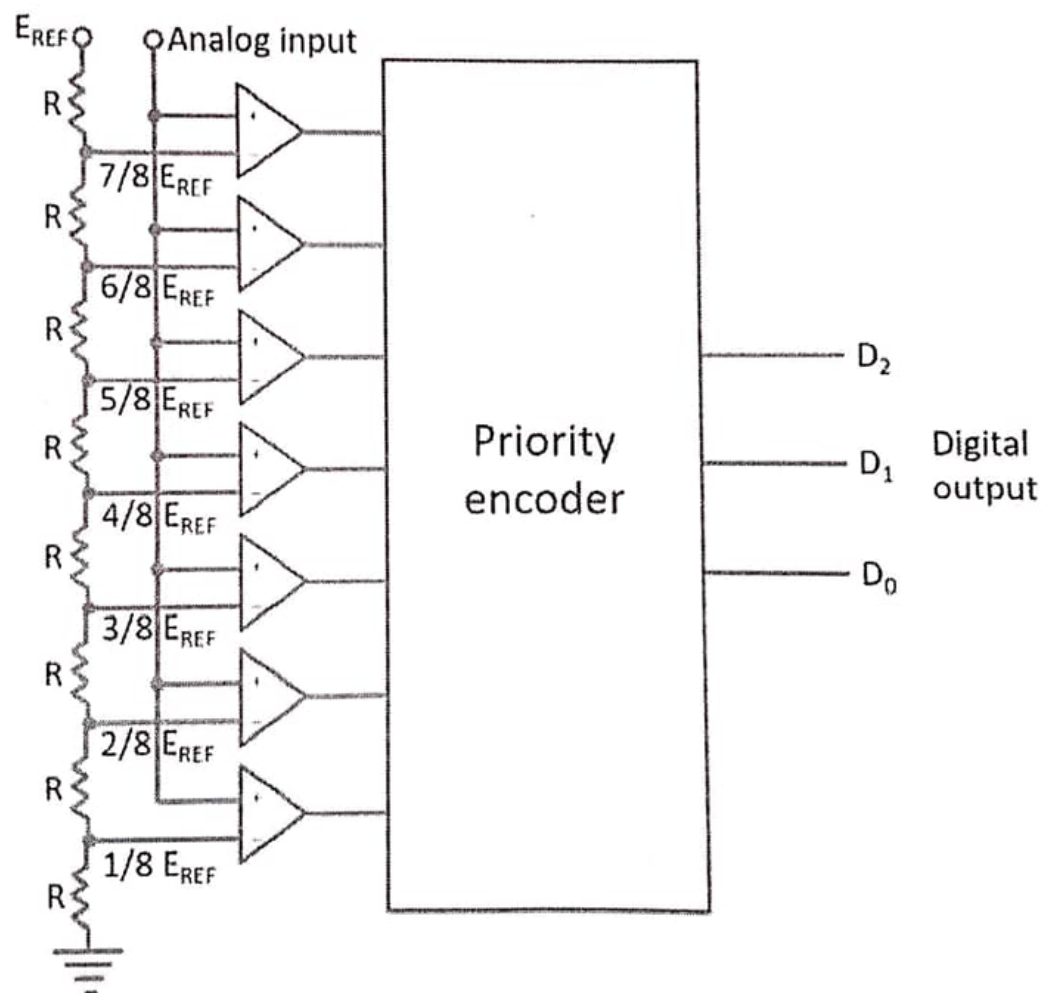
Circuit :



PRACTICAL 6

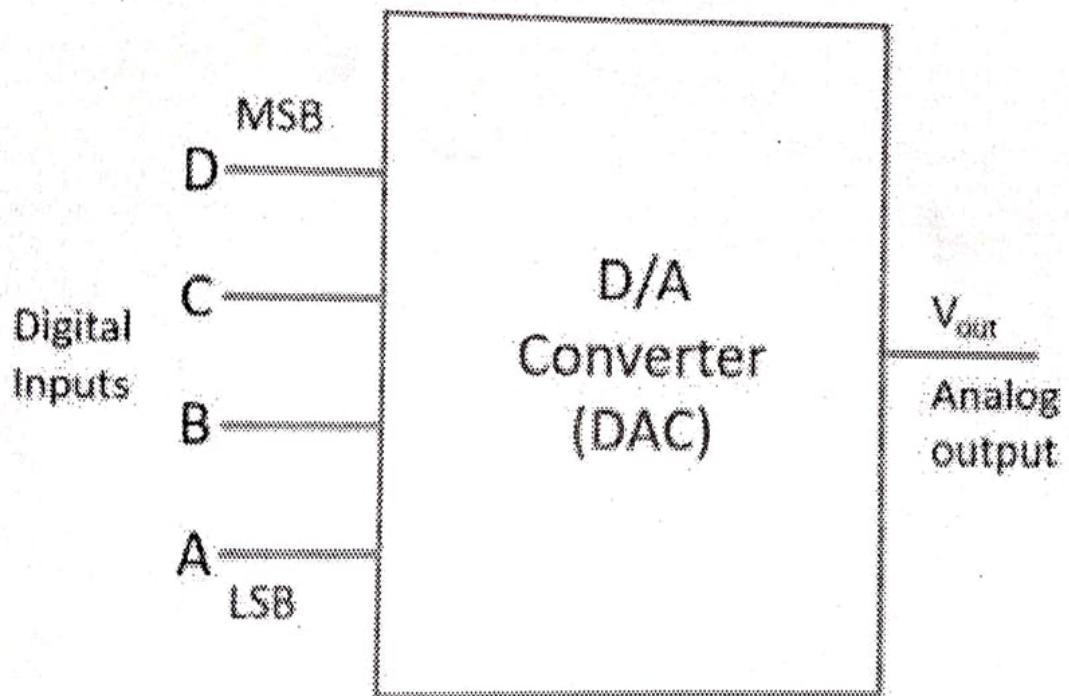
Study and configure the A to D converter and D to A converter.

A to D converter :



D to A converter :

2. D/A Converter :



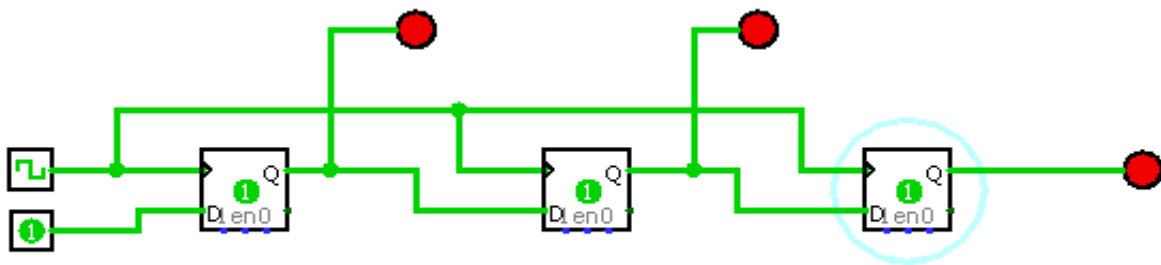
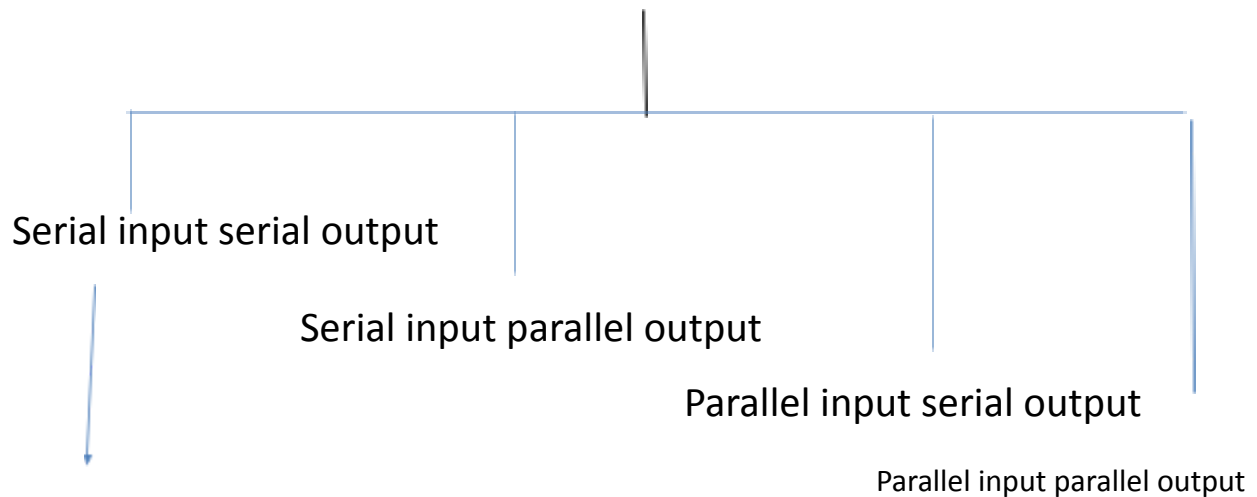
PRACTICAL 7

Study and implement a shifter.

SHIFT REGISTER

- It is a group of flip flops connected in series used to store multiple bits of data. the information stored within these registers can be transferred with the help of shift registers. A shift register is a group of flip-flops used to store multiple bits of data.

4 types are there :

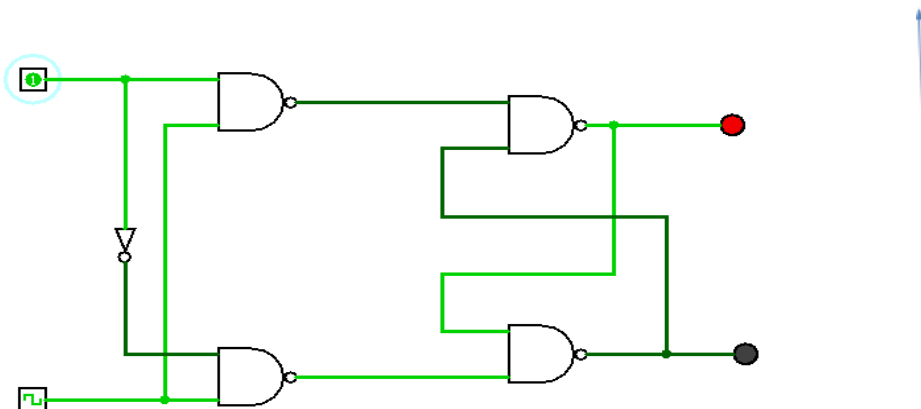
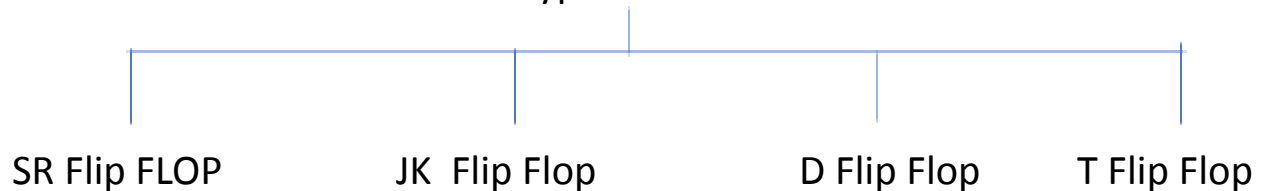


PRACTICAL 8

Study and implement flip flops

A circuit that has two stable states is treated as a **flip-flop**. These stable states are used to store binary data that can be changed by applying varying inputs. Flip-flops are the fundamental building blocks of the digital system. Flip flops and latches are examples of data storage elements. In the sequential logical circuit, the flip flop is the basic storage element.

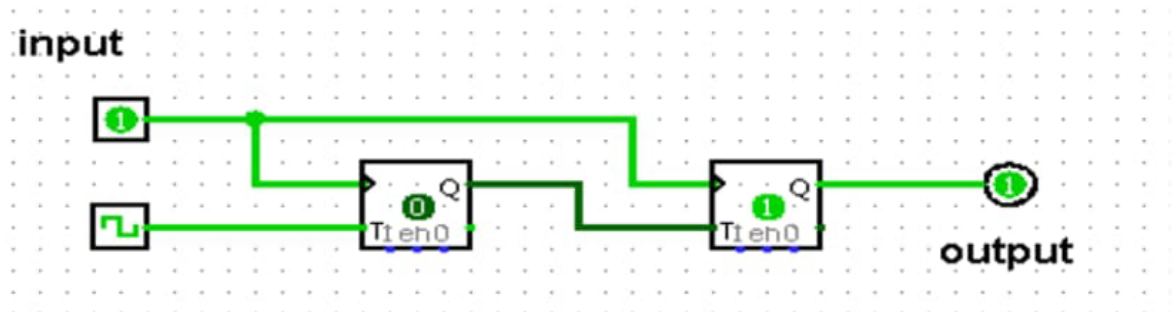
4 types are there :



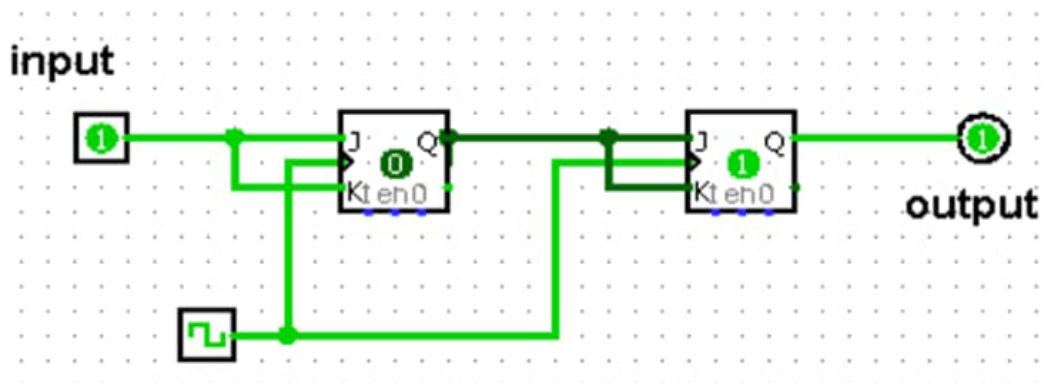
PRACTICAL 9

Study and implement counter.

1. Asynchronous counter :



2. Synchronous counter :

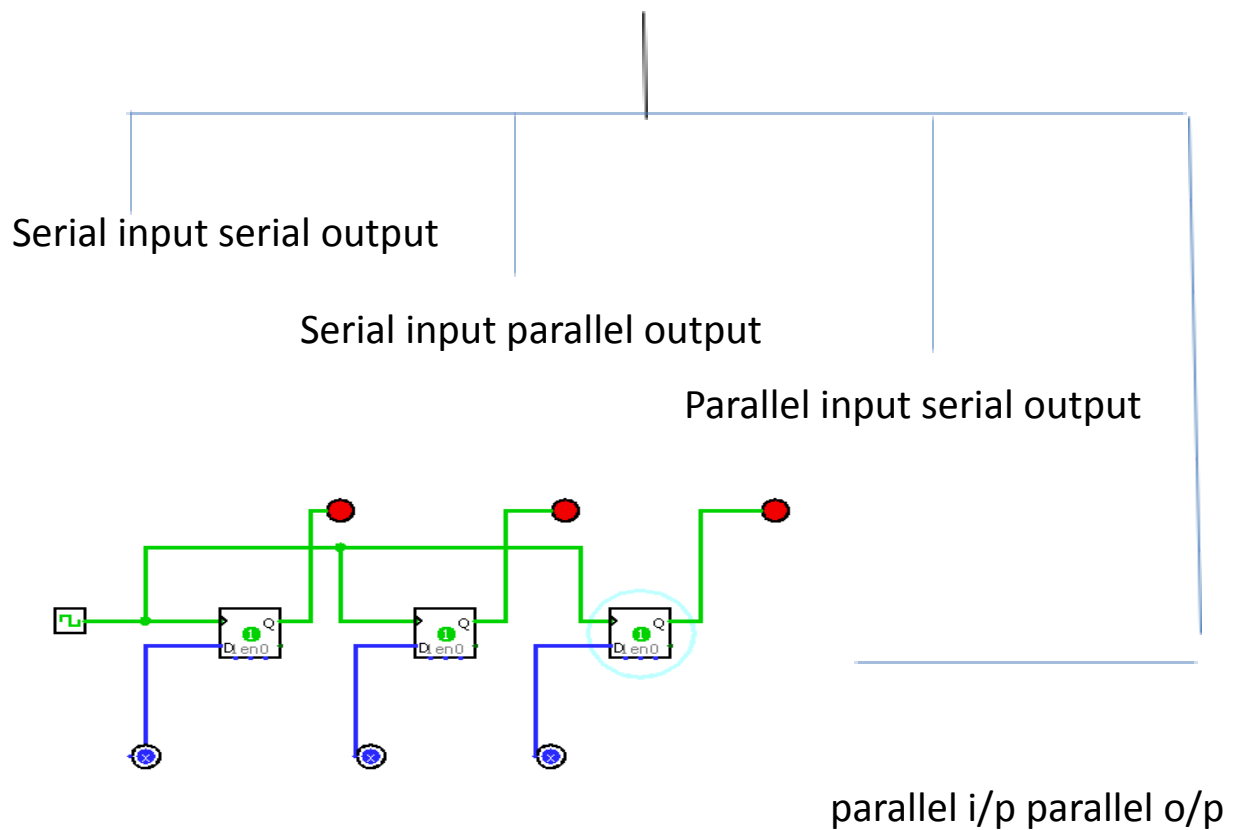


PRACTICAL 10

SHIFT REGISTER

- It is a group of flip flops connected in series used to store multiple bits of data. the information stored within these registers can be transferred with the help of shift registers. A shift register is a group of flip-flops used to store multiple bits of data.

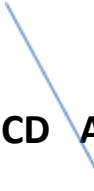
4 types are there :



PRACTICAL 11

Study and implement K – Map for the given function (sop)

$$F = ABC'D + AB'C'D' + AB'CD' + A'B'CD' + A'B'C'D'$$



CD \ AB	A'B'	A'B	AB	AB'
1	1			1
			1	
1	1			1

$$\text{SOP (F)} = B'D' + ABC'D$$

