CSE1003
Digital Logic and Design

Module 2
BOOLEAN ALGEBRA L1

Dr. S. Hemamalini
Professor
School of Electrical Engineering
VIT Chennai

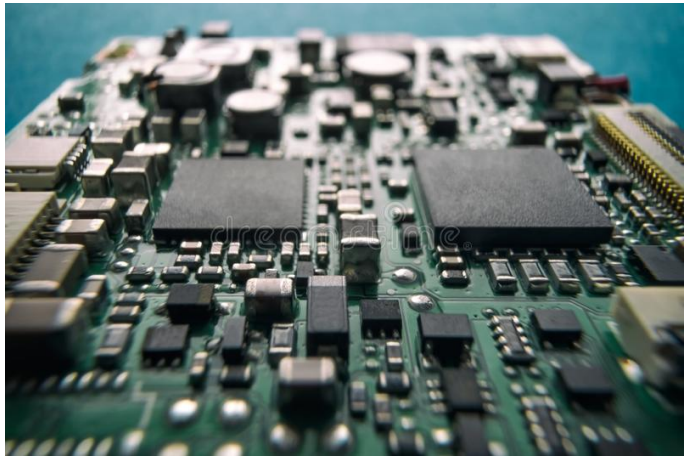
Module 2
BOOLEAN ALGEBRA 8 hrs

- Boolean algebra
- Properties of Boolean algebra
- Boolean functions
- Canonical and Standard forms
- Logic gates - Universal gates
- Karnaugh map - Don’t care conditions
- Tabulation Method

Logic Gates

- The logic gate is the most basic building block of any digital system, including computers.
- Each one of the basic logic gates is a piece of hardware or an electronic circuit that can be used to implement some basic logic expression, also known as Boolean expressions.
- There are three basic logic gates, namely the OR gate, the AND gate and the NOT gate.
- Other logic gates that are derived from these basic gates are the NAND gate, the NOR gate, the EXCLUSIVEOR gate and the EXCLUSIVE-NOR gate.

APPLICATIONS OF LOGIC GATES

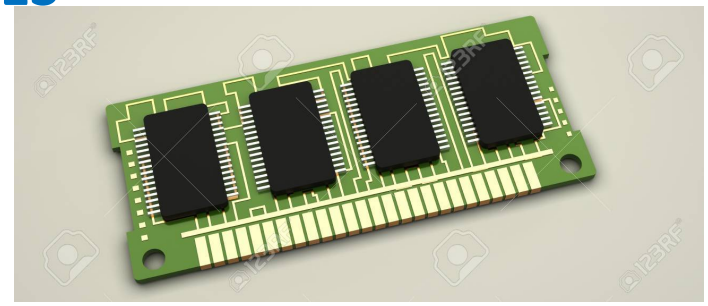


Microprocessors

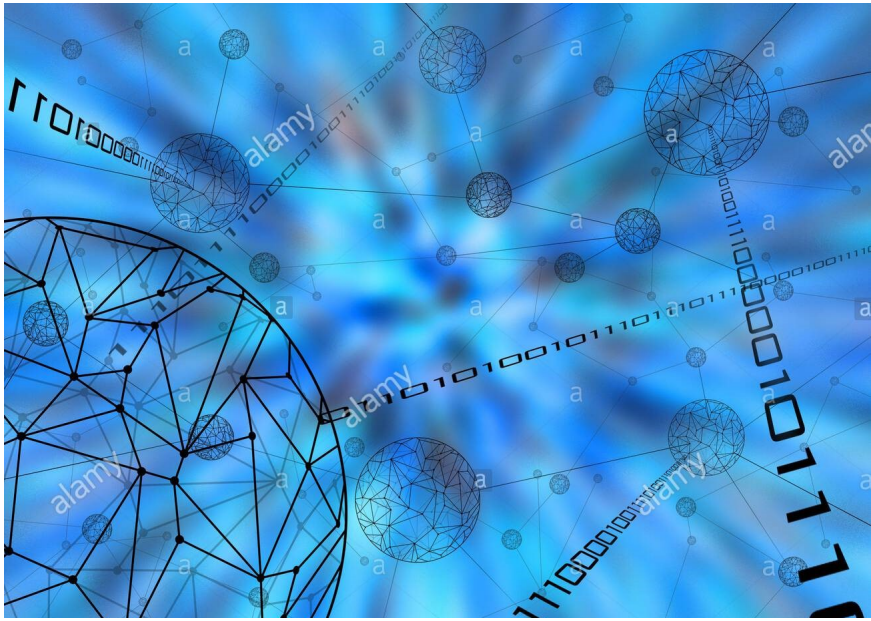
IC

LSI large

Random access memory



Digital Data Transmission and Internet



Flash Memories

Positive and Negative Logic

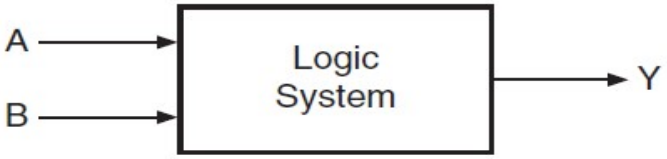
- The binary variables, can have either of the two states, i.e. the logic '0' state or the logic '1' state.
- These logic states in digital systems such as computers, for instance, are represented by two different voltage levels or two different current levels.

1 ON High +5 V

0 OFF Low 0 V

Truth Table

- A truth table lists all possible combinations of input binary variables and the corresponding outputs of a logic system.
- The logic system output can be found from the logic expression, often referred to as the Boolean expression, that relates the output with the inputs of that very logic system.

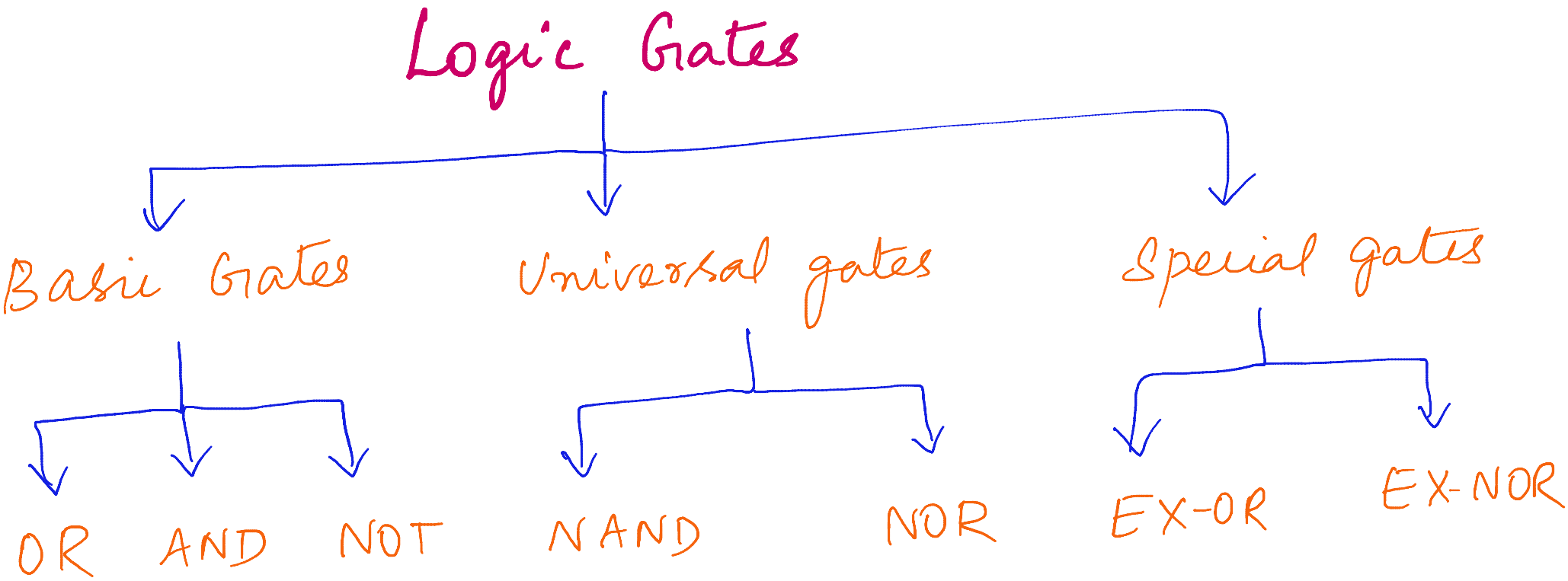


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Two-input logic system

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

Truth table of a three-input logic system

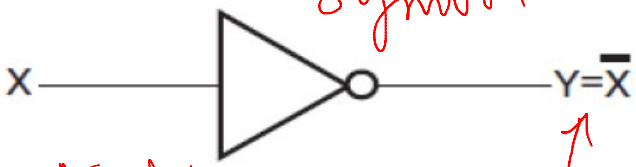

Logic Gates
Basic Gates
Universal gates
Special gates
OR
AND
NOT
NAND
NOR
EX-OR
EX-NOR

NOT Gate

Inverter gate

Inverted

Symbol



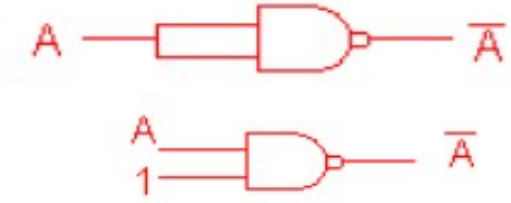
X'

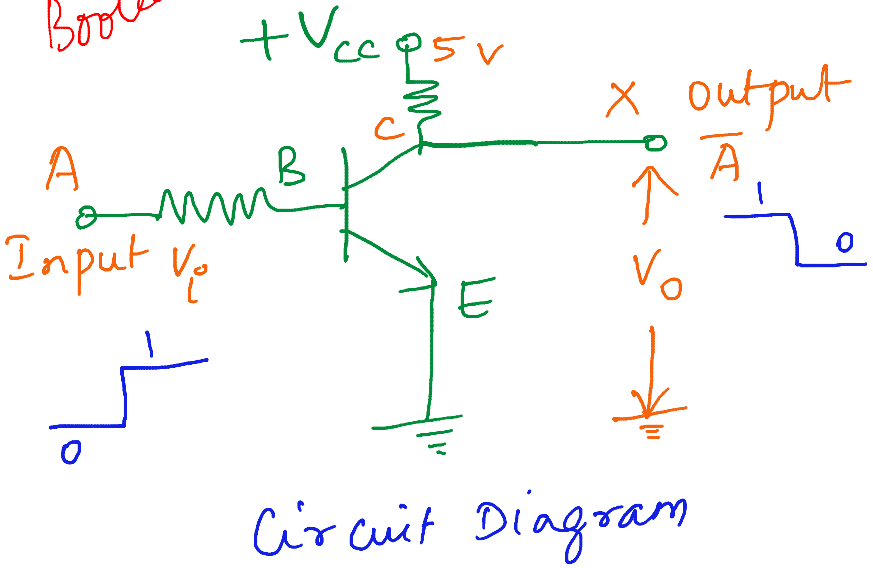
Boolean form.

Truth table

Input A	Output $\bar{A}$
0	1
1	0

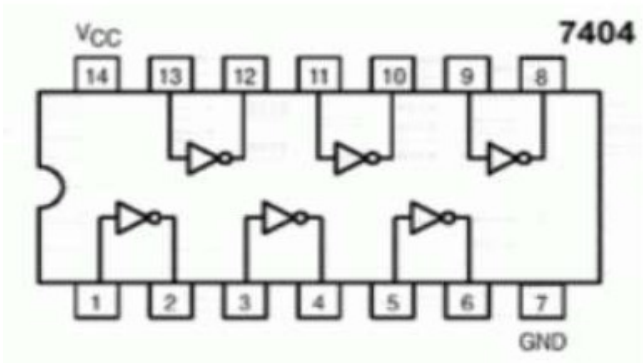
A NOT gate or *inverter* outputs a logic level that's the opposite (complement) of the input logic level.



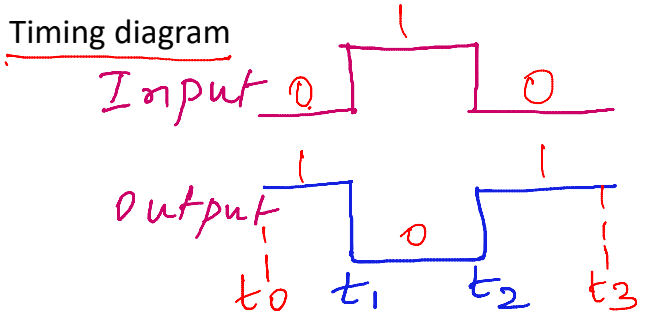


Circuit Diagram

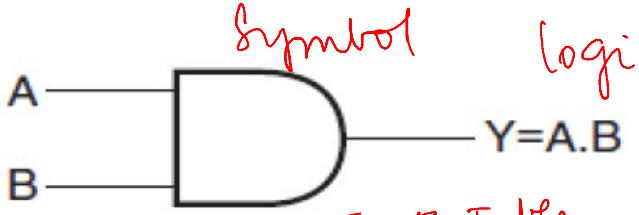
PIN DIAGRAM:



Timing diagram



AND GATE

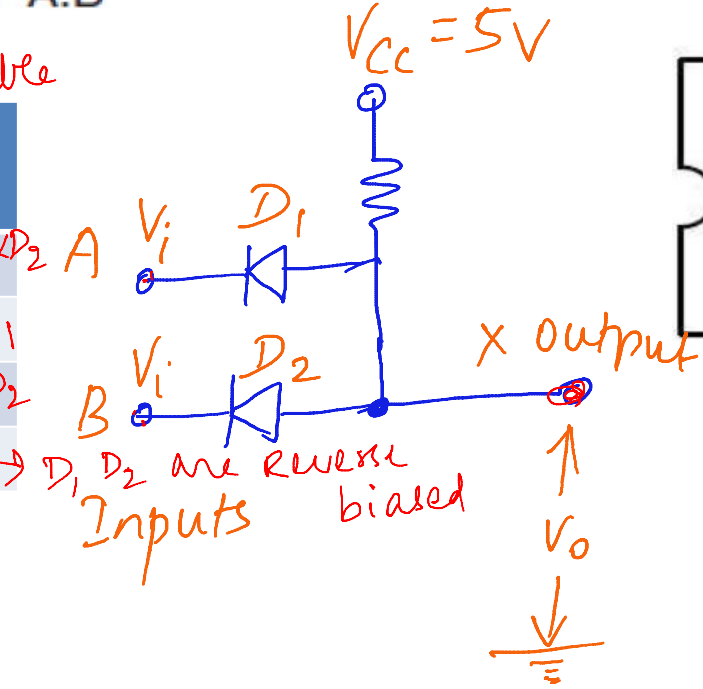


Logical Multiplication

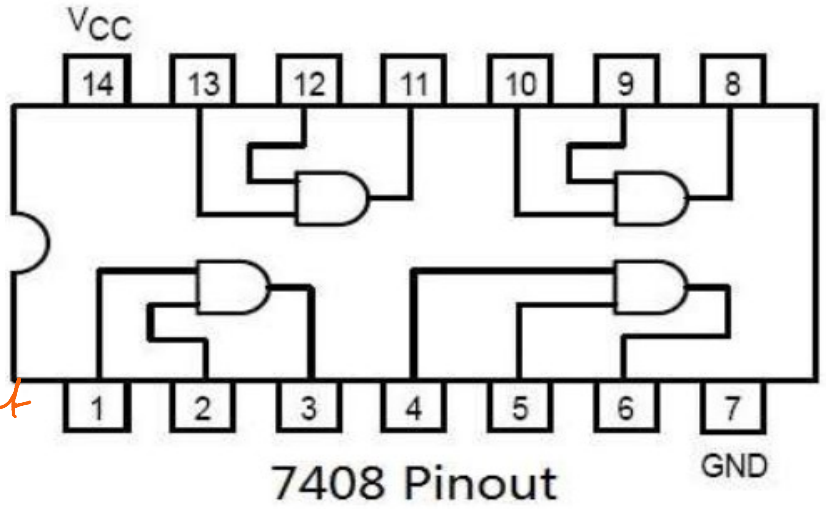
Truth Table

Input A	Input B	Output A.B
0	0	0
0	1	0
1	0	0
1	1	1

The output of an AND gate is HIGH only when both inputs are HIGH.



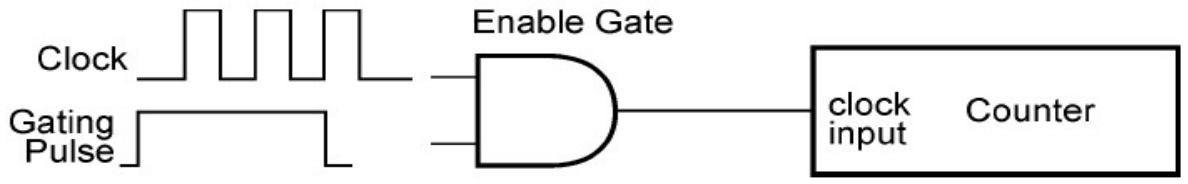
PIN DIAGRAM:



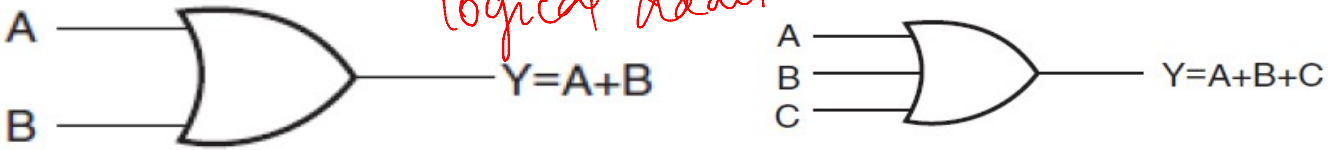
7408 Pinout

Application of an AND gate

- An AND gate is commonly used as an ENABLE or INHIBIT gate to allow or disallow passage of data from one point in the circuit to another.
- One such application of enabling operation, for instance, is in the measurement of the frequency of a pulsed waveform or the width of a given pulse with the help of a counter.
- In the case of frequency measurement, a gating pulse of known width is used to enable the passage of the pulse waveform to the counter's clock input.
- In the case of pulse width measurement, the pulse is used to enable the passage of the clock input to the counter.

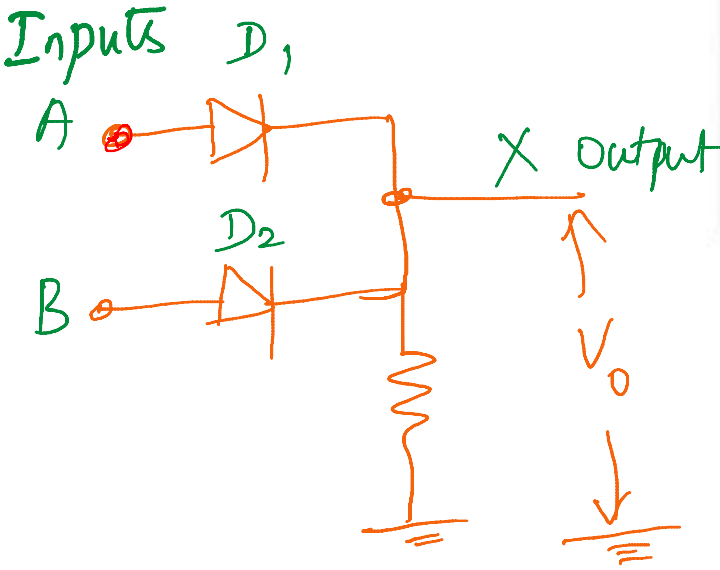


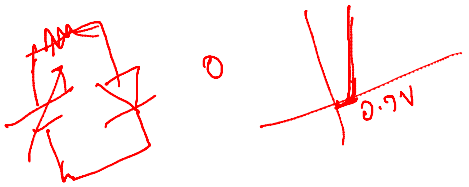
OR GATE



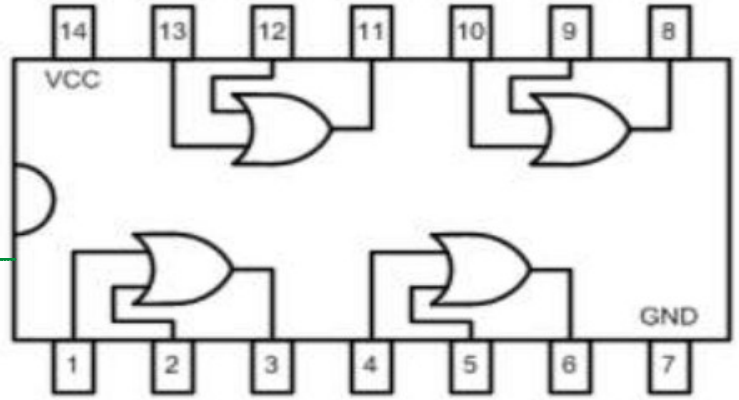
Input A	Input B	Output A+B
0	0	0
0	1	1
1	0	1
1	1	1

The output of an OR gate will go HIGH if one or both inputs goes HIGH. The output only goes LOW when both inputs are LOW.





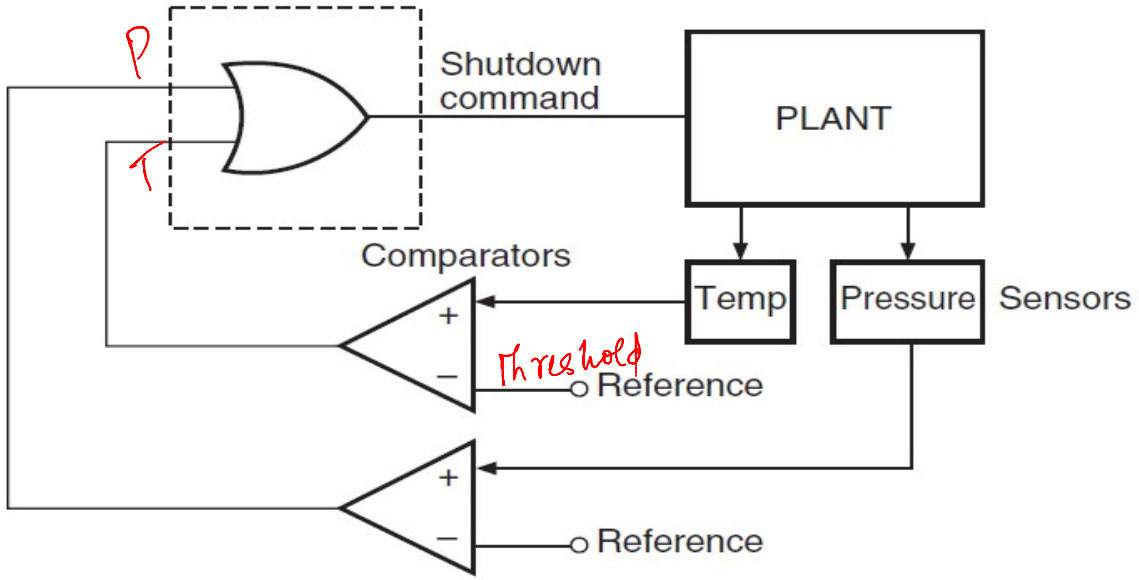
PIN DIAGRAM:



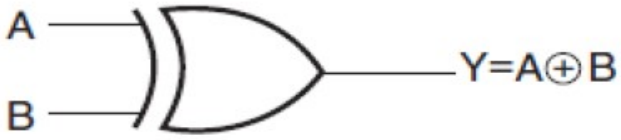
7432 Quad 2 Input OR

Application of an OR gate

- An OR gate can be used in all those situations where the occurrence of any one or more than one event needs to be detected or acted upon.

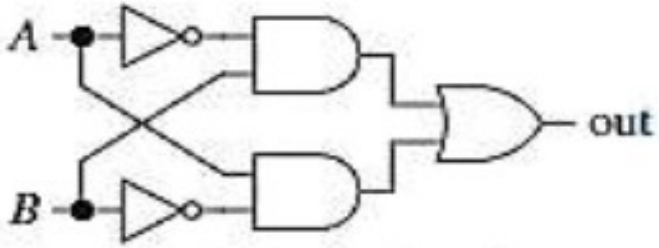


X-OR Gate



Input A	Input B	Output $\bar{A}B + A\bar{B}$
0	0	0
0	1	1
1	0	1
1	1	0

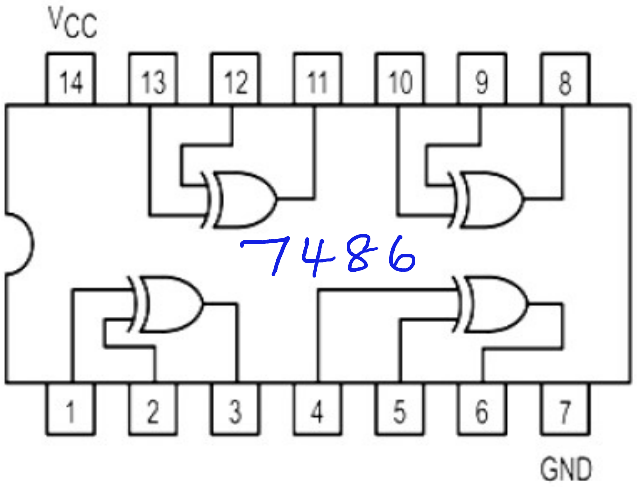
The output of an XOR gate goes HIGH if the inputs are different from each other. XOR gates only come with two inputs.



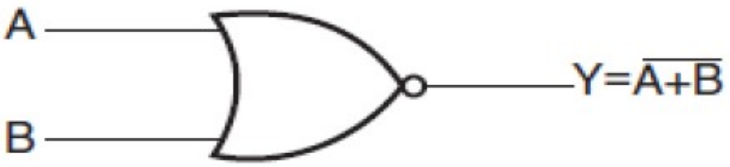
equivalent circuit

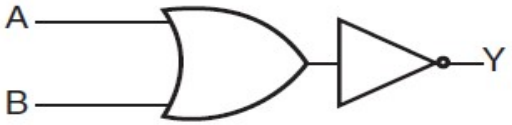
$$Y = (A \oplus B) = \overline{A}B + A\overline{B}$$

PIN DIAGRAM :



NOR Gate

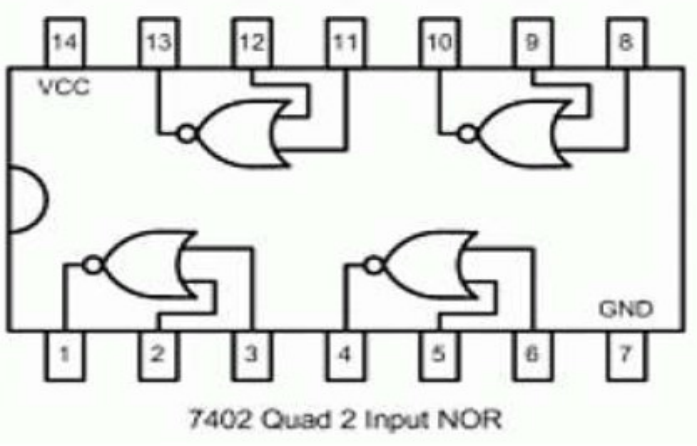




Input A	Input B	Output $\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

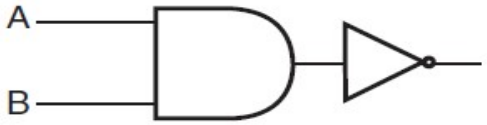
Combines the NOT function with an OR gate; output goes LOW if one or both inputs are LOW, output goes HIGH when both inputs are LOW.

PIN DIAGRAM :

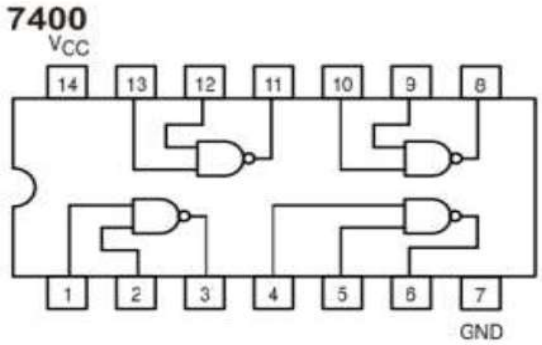


7402 Quad 2 Input NOR

NAND Gate

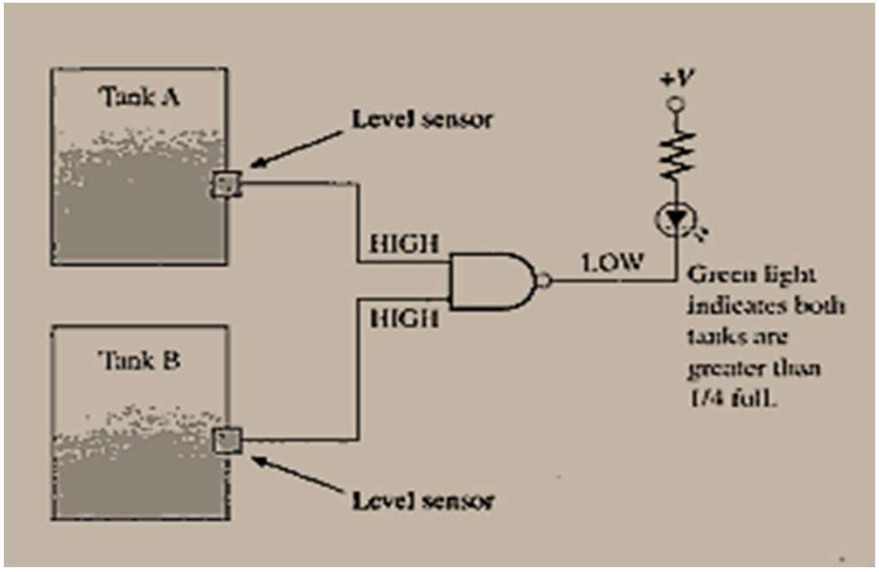


PIN DIAGRAM :

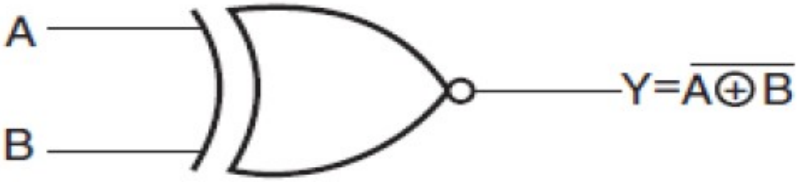


Input A	Input B	Output $\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

Combines the NOT function with an AND gate; output only goes LOW when both inputs are HIGH.



X-NOR Gate



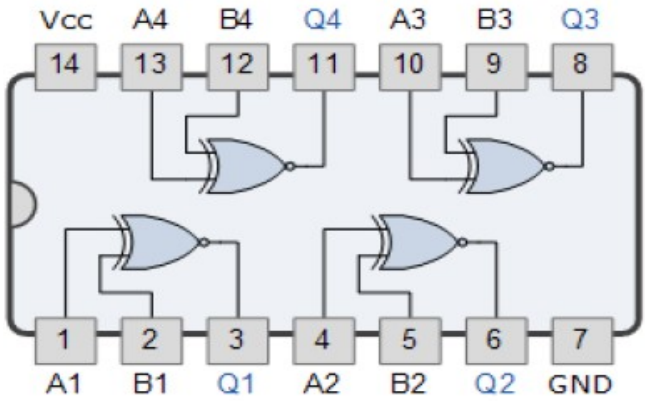
equivalent circuit

Input A	Input B	Output $A.B + \bar{A}.\bar{B}$
0	0	1
0	1	0
1	0	0
1	1	1

$$Y = (\overline{A \oplus B}) = (A.B + \overline{A}.\overline{B})$$

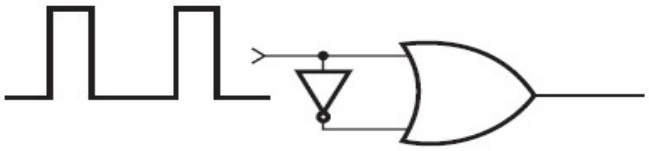
Combines the NOT function with an XOR gate; output goes HIGH if the inputs are the same.

PIN DIAGRAM :



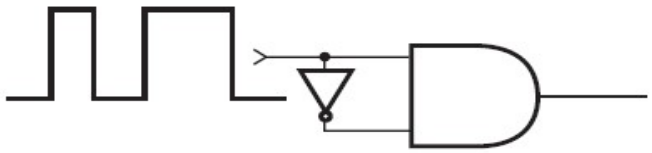
NOT Gate

- *For the logic circuit arrangements of Figs (a) and (b), draw the output waveform.*



(a)

In the case of the OR gate arrangement of Fig. (a), the output will be permanently in logic '1' state as the two inputs can never be in logic '0' state together owing to the presence of the inverter.

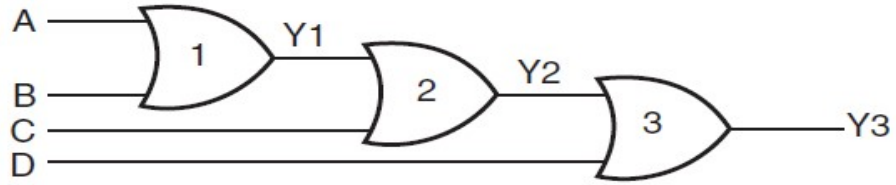


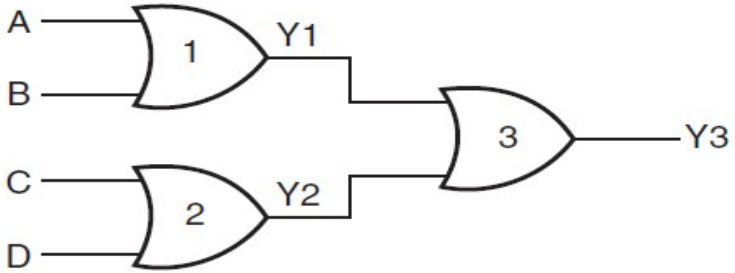
(b)

In the case of the AND gate arrangement of Fig. (b), the output will be permanently in logic '0' state as the two inputs can never be in logic '1' state together owing to the presence of the inverter.

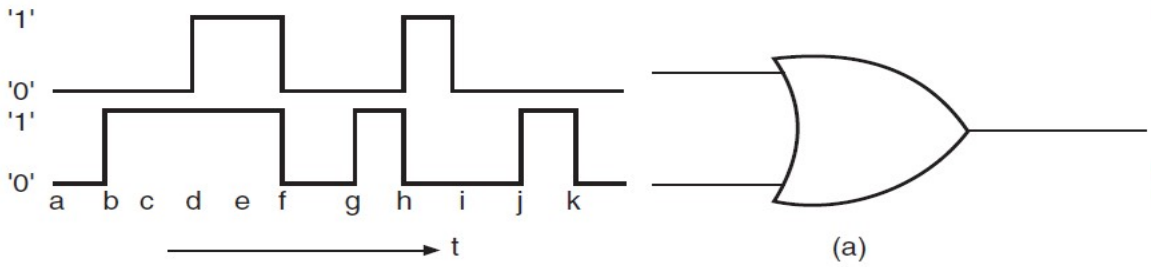
OR GATE

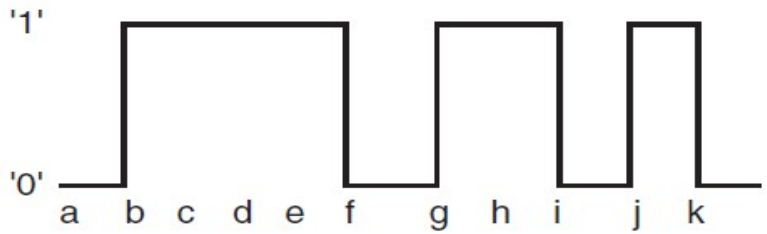
- ***How would you hardware-implement a four-input OR gate using two-input OR gates only?***





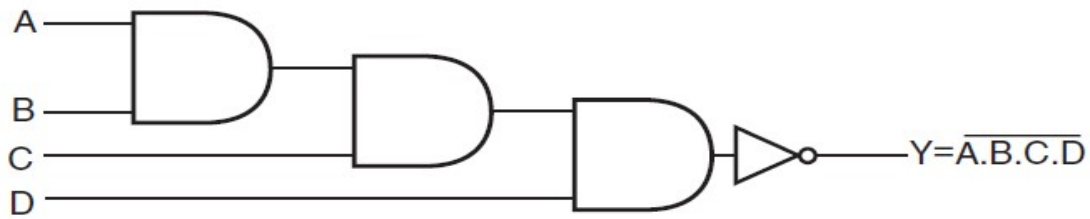
- ***Draw the output waveform for the OR gate and the given pulsed input waveforms of Fig.***



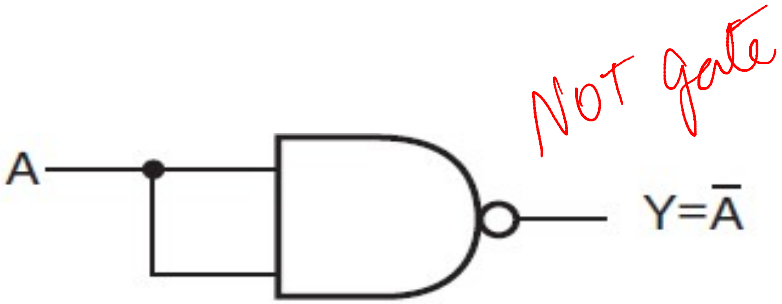


Show the logic arrangements for implementing:

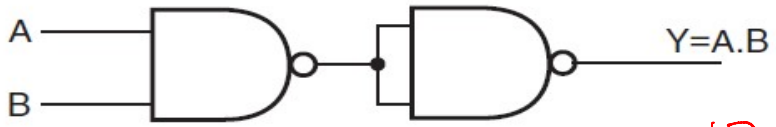
(a) a four-input NAND gate using two-input AND gates and NOT gates



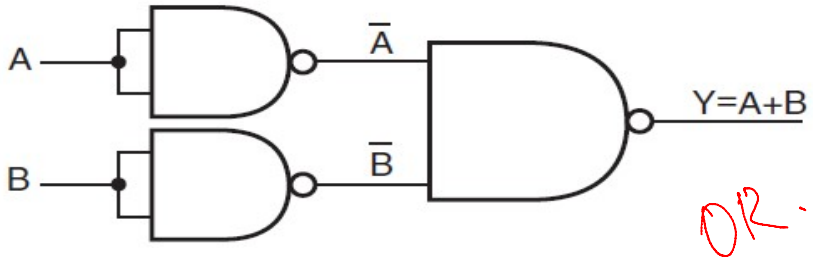
Implementation of basic logic gates using only NAND gates



NOT gate

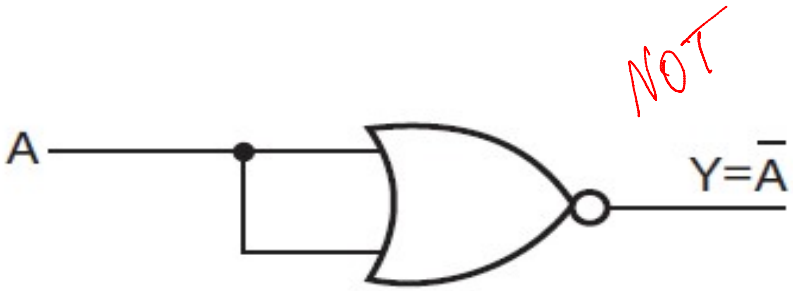


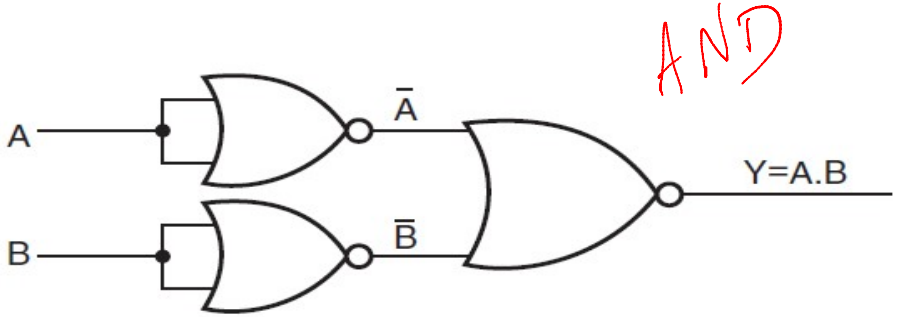
AND

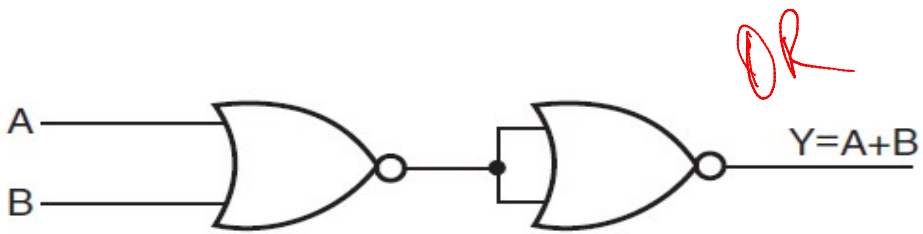


OR.

Implementation of basic logic gates using only NOR gates

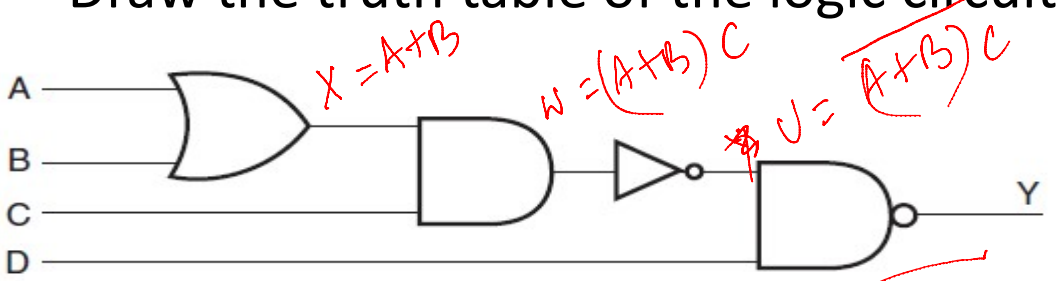






Do it by yourself

- Draw the truth table of the logic circuit shown in



$X = A+B$

$W = (A+B)C$

$U = \overline{(A+B)C}$

$Y = \overline{(\overline{(A+B)C})D}$

$2^4 = 16$

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1