CHAPTER 1

Fundamental of Digital Techniques

1.1 DIGITAL SIGNAL

Digital means binary that is 0 and 1. Signal is nothing but transmission of data.

A digital signal uses discrete values (either 0 or 1) to represents, information in discrete or continuous form.

Discrete information *e.g.* numbers, letters, etc.

Continuous information *e.g.* sound, images, etc.

Digital signal uses only binary data that is (0 or 1). Representation of digital signal is shown in Fig. 1.1.

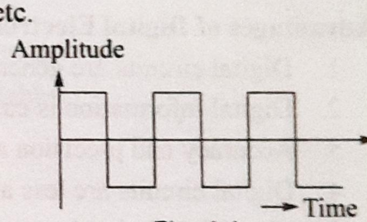


Fig. 1.1

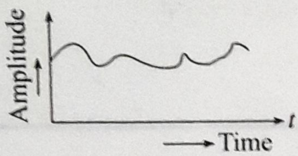
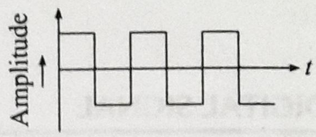
But when we compare digital signal with analog signal then analog signal is varying continuous with time it means all the natural examples are analog signal. Analog signals are continuous in nature.

An analog signal uses some medium to convey the information. If we have information in form of some physical form like sound, light, etc. It can be converted from physical form to electrical signal by using transducer. As all the variation of analog signals are significant.

Hence, any disturbances is equivalent to a change in original signal and so appears noise. That is unwanted source of energy.

1.1.1 Comparison between Analog and Digital Signal

Analog Signal	Digital Signal
1. Analog signals are continuously varying. *e.g.*: Temperature, pressure.	1. Digital signals are discrete based on 0's and 1's *e.g.*: ON/OFF switch.
2. These signals are much more susceptible to noise. Means noise can cause corruption of information.	2. These are based on only two different values either 0 or 1, chances of noise are less in this. 0 = OFF, 1 = ON

3. Analog signals are difficult to transmit.	3. Digital signals are easier to transmi
4. Chances of erros are much more because of continuous varying nature.	4. It offers less error to occur becaus of binary information.
5. It transmits less accurate data.	5. It transmits accurate data.
6. Transmission rate of data is not fast and productivity is less.	6. Transmission rate of data is faste and productivity is better.
7. It is less reliable.	7. It is more reliable.
8. It is represented as: 	8. It is represented as: 

1.1.2 Digital Electronics

It is a system, which processes digital signals. It is a combination of devices designed perform logical information (on physical quantities that are represented in digital fon

Advantages of Digital Electronics

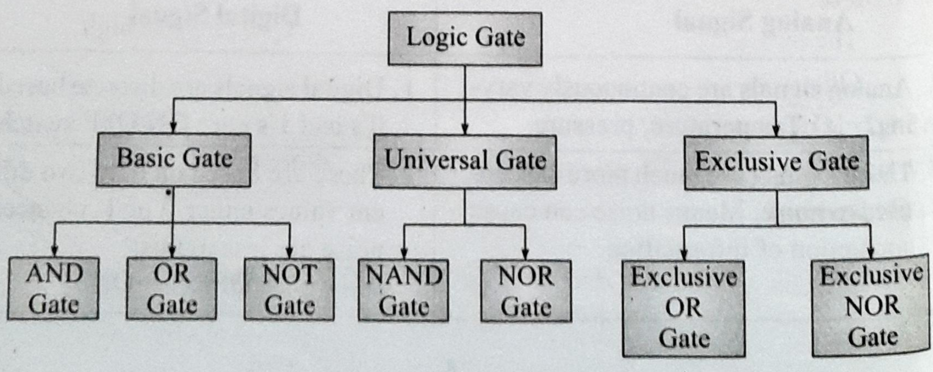
1. Digital circuits are generally easier to design.
2. Digital information is easy to store.
3. Accuracy and precision are greater.
4. Digital circuits are less affected by noise.
5. More digital circuits can be performed on single IC chips.

1.2 LOGIC GATES

Logic Gate is a digital circuit that is able to operate on a number of binary inputs order to perform perticular logic function.

Logic Gates are solid state electronic circuit that are used to perform certa mathematical function, *i.e.* multiplication, addition, inversion by using binary dig (0 and 1). *E.g.*: AND gate for multiply, OR gate for addition.

Three common classification of Logic Gate are popularly used to designed t different circuits and to solve all the mathematical operation.



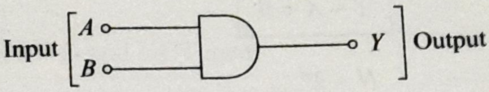
1.2.1 Basic Gate

These are common gate to solve the basic mathematical function like multiplication, addition and inversion operation.

1.2.1.1 AND Gate

The basic function of AND gate is to perform logic multiplication.

Logic Symbol



AND gate takes two or more input and gives a single output.

Inputs are on left hand side.

Output are on right hand side.

Logic Expression

Multiply A, B and equivalent to Y.

$$\boxed{Y = A \cdot B}$$

Truth Table (for 2 input AND gate)

$$N = 2^n$$

where $N =$ No. of combinations.

$n =$ No. of input variables.

when $\boxed{n = 2}$

$$N = 2^2$$

$$\boxed{N = 4}$$

No. of I/P variables = 2 [A, B]

No. of combinations = 4 [00, 01, 10, 11]

A	B	Y = A · B
0	0	0 (Low)
0	1	0 (Low)
1	0	0 (Low)
1	1	1 (High)

0 = Low
1 = High

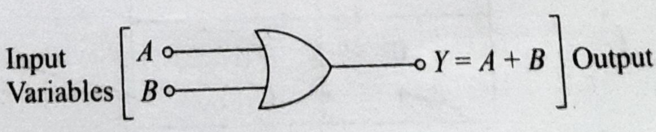
Statement: According to the truth table.

The output of AND gate is high only when all the inputs are high. Otherwise, output is Low.

1.2.1.2 OR Gate

The basic function of OR gate is to perform logic addition.

Logic Symbol:



Similar to AND gate, OR gate takes two or more inputs and gives a sing output.

Inputs are given on left hand side.

Outputs are given on right hand side.

Logic Expression:

Add A, B and equivalent to Y.

$$\boxed{Y = A + B}$$

Truth Table:

$N = 2^n$

$n = 2\ (A, B)$

$N = 4\ (00, 01, 10, 11)$

A	B	$Y = A + B$
0	0	0 (Low)
0	1	1 (High)
1	0	1 (High)
1	1	1 (High)

Statement: The output of OR gate is high when any of the inputs are high.

The output of OR gate is low only, when all of the inputs are low.

1.2.1.3 NOT Gate

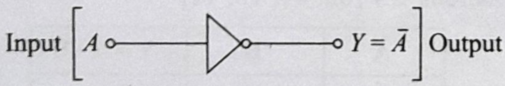
It perform the function of inversion or complement.

Function: To change one logic level to opposite level.

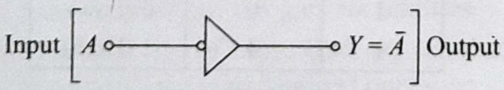
$0 \rightarrow 1$

$1 \rightarrow 0$

Logic Symbol:



OR



0 = Bubble will invert the level.

Logic Expression:

$$\boxed{Y = \bar{A}}$$

Truth Table:

A	$Y = \bar{A}$
0	1 (High)
1	0 (Low)

Statement: A high level to an inverter produces a low level and a low level to an inverter produces a high level.

1.2.2 Universal Gate

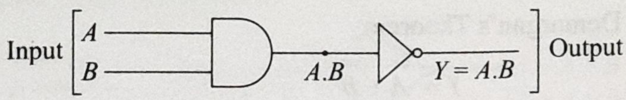
Universal Gate are those gate which can be used to construct an AND gate, an OR gate, an inverter or any combination of these function.

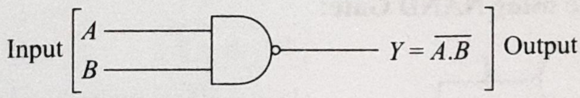
1.2.2.1 NAND Gate

It is the combination of AND and NOT gate.

$$\boxed{\text{NAND} = \text{AND} + \text{NOT}}$$

Logic Symbol:





Logic Expression:

$$\boxed{Y = \overline{A \cdot B}}$$

Complement of AND gate output expression.

Truth Table:

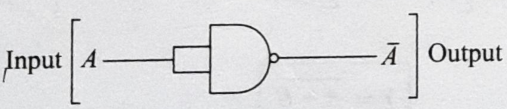
A	B	$A \cdot B$	$Y = \overline{A \cdot B}$
0	0	0	1 (High)
0	1	0	1 (High)
1	0	0	1 (High)
1	1	1	0 (Low)

Statement: The output of NAND gate is low when all the inputs are high.

Otherwise, output is high.

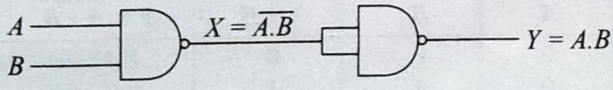
Universal Property of NAND Gate:

(*i*) **NAND Gate as an Inverter:**



$$\overline{A \cdot A} = \bar{A}$$

(*ii*) **AND Gate using NAND Gate:**

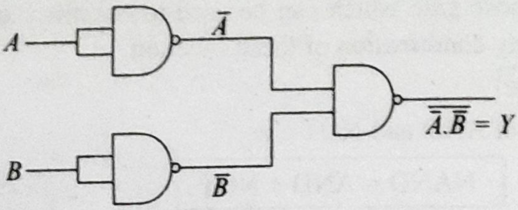


$$X = \overline{A \cdot B}$$

$$Y = \overline{\overline{A \cdot B}}$$

$$\boxed{Y = A \cdot B}$$

(*iii*) **OR Gate using NAND Gate:**



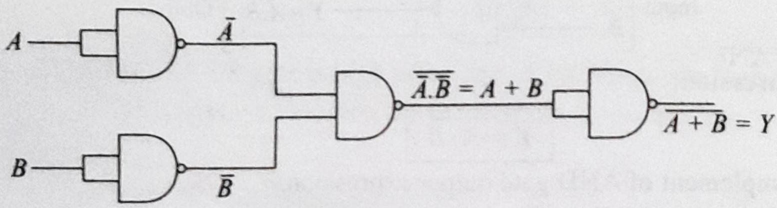
$$Y = \overline{\bar{A} \cdot \bar{B}}$$

Using Demorgan's Theorem:

$$Y = \bar{\bar{A}} + \bar{\bar{B}}$$

$$\boxed{Y = A + B}$$

(*iv*) **NOR Gate using NAND Gate:**



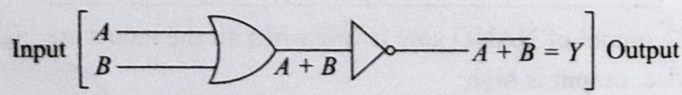
$$\boxed{Y = \overline{A + B}}$$

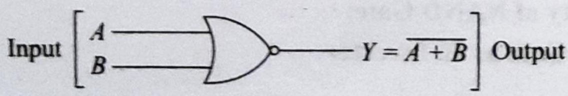
1.2.2.2 NOR Gate OR + NOT

The basic function of NOR gate is to perform logical addition and then producin complemented output.

$$\boxed{\text{NOR} = \text{OR} + \text{NOT}}$$

Logic Symbol:





Logic Expression:

$$\boxed{Y = \overline{A + B}}$$

Complement of OR gate output expression.

Truth Table:

A	B	$A+B$	$Y = \overline{A+B}$
0	0	0	1 (High)
0	1	1	0 (Low)
1	0	1	0 (Low)
1	1	1	0 (Low)

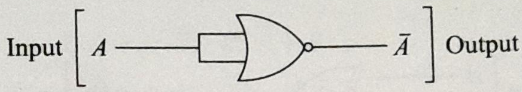
Statement: The output of NOR gate is low, when any of its inputs are high.

and

The output is high, when all of its inputs are low.

Universal Property of NOR Gate:

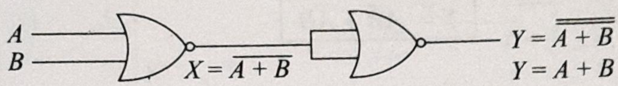
(*i*) **NOR Gate used as Inverter:**



$$Y = \overline{A \cdot A}$$

$$\boxed{Y = \overline{A}}$$

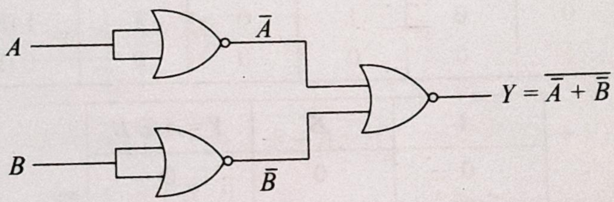
(*ii*) **OR Gate using NOR Gate:**



$$Y = \overline{\overline{A + B}}$$ [∵ Bar is cancel out to Bar]

$$\boxed{Y = A + B}$$

(*iii*) **AND Gate using NOR Gate:**

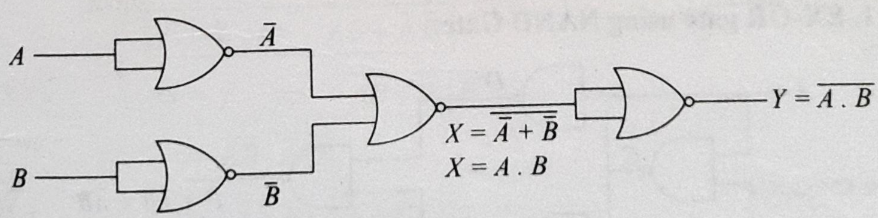


$$Y = \overline{\overline{A} + \overline{B}}$$

$$= \overline{\overline{A}} \cdot \overline{\overline{B}}$$ [∵ Bar (–) can change sign]

$$\boxed{Y = A \cdot B}$$

(*iv*) **NAND Gate using NOR Gate:**



$$X = \overline{\overline{A} + \overline{B}}$$

$$= A \cdot B$$

$$\boxed{Y = \overline{A \cdot B}}$$

This is expression of NAND Gate.

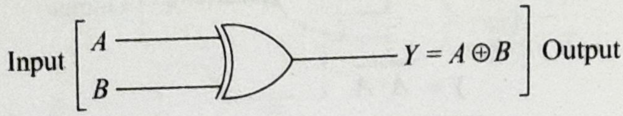
1.2.3 Exclusive Gate

These gate are different from basic and universal gate but using the properties of ba gates only.

1.2.3.1 Exclusive–OR Gate

Exclusive–OR gate is abbreviated as Ex–OR gate, XOR gate.

Logical Symbol:



Logic Expression:

$$Y = A \oplus B \Rightarrow A \text{ XOR } B.$$

$$\boxed{Y = A\bar{B} + \bar{A}B}$$

Truth Table:

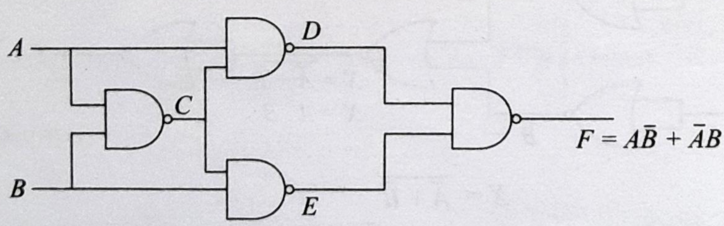
A	B	$\bar{A}$	$\bar{B}$	$\bar{A}B$	$A\bar{B}$	$Y = A\bar{B} + \bar{A}B$
0	0	1	1	0	0	0 (Low)
0	1	1	0	1	0	1 (High)
1	0	0	1	0	1	1 (High)
1	1	0	0	0	0	0 (Low)

A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Statement: The output is high only when the two inputs are at opposite level either 0, 1 and 1, 0.

Two Important designing of EX-OR gate:

1. **EX-OR gate using NAND Gate:**



$$C = \overline{A \cdot B}$$

$$D = \overline{A \cdot C} \Rightarrow \overline{A \cdot \overline{A \cdot B}} \Rightarrow \bar{A} + A \cdot B$$

$$E = \overline{B \cdot C} \Rightarrow \overline{B \cdot \overline{A \cdot B}} \Rightarrow \bar{B} + A \cdot B$$

$$F = \overline{D \cdot E} \Rightarrow \overline{(\bar{A} + AB) \cdot (\bar{B} + AB)}$$

$$\Rightarrow (\overline{\bar{A} + A \cdot B}) + (\overline{\bar{B} + A \cdot B})$$

$$\Rightarrow (\bar{\bar{A}} \cdot \overline{A \cdot B}) + (\bar{\bar{B}} \cdot \overline{A \cdot B})$$

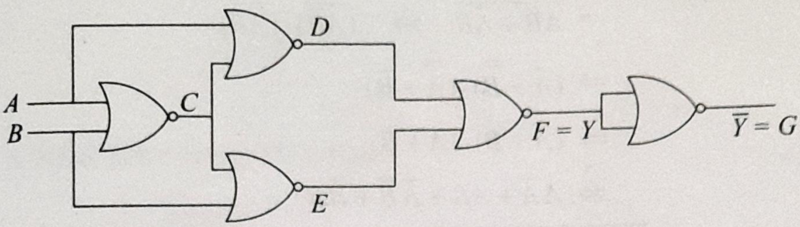
$$\Rightarrow [A \cdot (\overline{A \cdot B})] + [B \cdot (\overline{A \cdot B})]$$

$$\Rightarrow [A \cdot (\bar{A} + \bar{B})] + [B \cdot (\bar{A} + \bar{B})]$$

$$\Rightarrow A \cdot \bar{A} + A \cdot \bar{B} + B \cdot \bar{A} + B \cdot \bar{B} \qquad [A \cdot \bar{A} = 0, B \cdot \bar{B} = 0]$$

$$\Rightarrow A \cdot \bar{B} + \bar{A} \cdot B \quad \Rightarrow \quad A \oplus B$$

2. EX-OR gate using only NOR Gate:



$$\boxed{C = \overline{A + B}}$$

$$D = \overline{A + C} \quad \Rightarrow \quad \overline{A + \overline{A + B}} \quad \Rightarrow \quad \bar{A} \cdot (A + B)$$

$$\Rightarrow A \cdot \bar{A} + \bar{A}B$$

$$\boxed{D = \bar{A}B}$$

$$E = \overline{B + C} \quad \Rightarrow \quad \overline{B + \overline{A + B}} \quad \Rightarrow \quad \bar{B} \cdot (A + B)$$

$$\Rightarrow A \cdot \bar{B} + B \cdot \bar{B}$$

$$\boxed{E = A\bar{B}}$$

$$F = Y = \overline{D + E} \quad \Rightarrow \quad \overline{\bar{A}B + A\bar{B}} \quad \Rightarrow \quad \overline{\bar{A}B} \cdot \overline{A\bar{B}}$$

$$\Rightarrow (\bar{\bar{A}} + \bar{B}) \cdot (\bar{A} + \bar{\bar{B}}) \quad \Rightarrow \quad (A + \bar{B}) \cdot (\bar{A} + B)$$

$$\Rightarrow A \cdot \bar{A} + \bar{A}\bar{B} + B\bar{B} + AB$$

$$\boxed{F = Y \quad \Rightarrow \quad AB + \bar{A}\bar{B}}$$

$$G = \bar{Y} \quad \Rightarrow \quad \overline{AB + \bar{A} \cdot \bar{B}}$$

$$\Rightarrow \overline{AB} \cdot \overline{\bar{A} \cdot \bar{B}}$$

$$\Rightarrow (\bar{A} + \bar{B}) \cdot (\bar{\bar{A}} + \bar{\bar{B}})$$

$$\Rightarrow (\bar{A} + \bar{B}) \cdot (A + B)$$

$$\Rightarrow A\bar{A} + A\bar{B} + \bar{A}B + B \cdot \bar{B} \qquad [A \cdot \bar{A} = 0, B \cdot \bar{B} = 0]$$

$$G \Rightarrow A \cdot \bar{B} + \bar{A}B$$

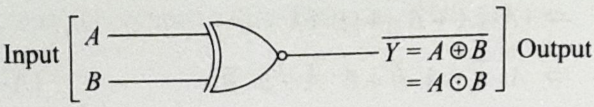
$$\boxed{G \Rightarrow A \oplus B}$$

1.2.3.2 EX-NOR Gate

The complement of EX-OR function is Exclusive-NOR gate.

It is abbreviated as X NOR.

Logic Symbol:



Logic Expression:

$$Y = \overline{A \oplus B}$$

$$= \overline{A\bar{B} + \bar{A}B} \quad \Rightarrow \quad (\overline{A\bar{B}}) \cdot (\overline{\bar{A}B})$$

$$\Rightarrow (\bar{A} + \bar{\bar{B}}) \cdot (\bar{\bar{A}} + \bar{B})$$

$$\Rightarrow (\bar{A} + B) \cdot (A + \bar{B})$$

$$\Rightarrow A\bar{A} + AB + \bar{A}\bar{B} + B\bar{B}$$

$$\boxed{Y \Rightarrow AB + \bar{A}\bar{B}}$$

$$\boxed{Y = A \odot B}$$

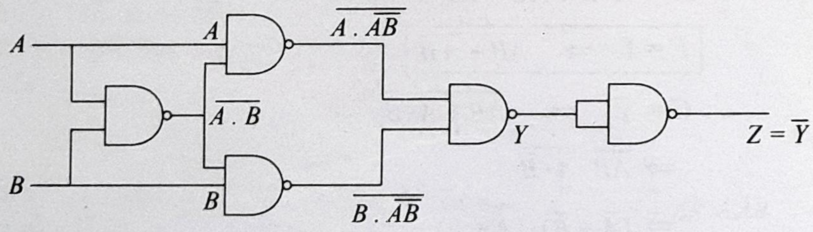
Truth Table:

A	B	$A \cdot \bar{B} + \bar{A} \cdot B$	$Y = AB + \bar{A}\bar{B}$
0	0	0	1 (High)
0	1	1	0 (Low)
1	0	1	0 (Low)
1	1	0	1 (High)

Statement: The output is high only, when the two inputs are alike (00, 11).

Two Important designing of EX-NOR gate:

1. X NOR Gate Using NAND Gate:



$$Y = \overline{\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}}$$

$$= \overline{\overline{A \cdot \overline{AB}}} + \overline{\overline{B \cdot \overline{AB}}}$$

$$= A \cdot \overline{AB} + B \cdot \overline{AB}$$

$$= A \cdot (\bar{A} + \bar{B}) + B \cdot (\bar{A} + \bar{B})$$

$$= A\cdot\bar{A} + A\bar{B} + B\bar{A} + B\bar{B}$$

$$= A\bar{B} + \bar{A}B$$

$$Z = \bar{Y}$$

$$= \overline{A\bar{B} + \bar{A}B}$$

$$= \overline{A\cdot\bar{B}}\cdot\overline{\bar{A}\cdot B}$$

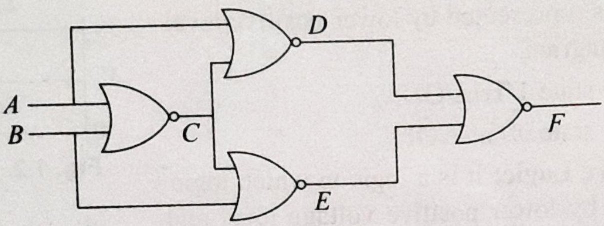
$$= (\bar{A} + \bar{\bar{B}})\cdot(\bar{\bar{A}} + \bar{B})$$

$$= (\bar{A} + B)(A + \bar{B}) \Rightarrow A\bar{A} + AB + B\bar{B} + \bar{A}\bar{B}$$

$$\boxed{Z \Rightarrow AB + \bar{A}\bar{B}}$$

$$\boxed{Z = A \odot B}$$

2. X NOR Gate using NOR Gate:



$$\boxed{C = \overline{A + B}}$$

$$D = \overline{A + C} \Rightarrow \overline{A + \overline{A + B}} \Rightarrow \bar{A}\cdot(A + B)$$

$$\Rightarrow A\cdot\bar{A} + \bar{A}B \Rightarrow \bar{A}B$$

$$\boxed{D = \bar{A}B}$$

$$E = \overline{B + C} \Rightarrow \overline{B + \overline{A + B}} \Rightarrow \bar{B}\cdot(A + B)$$

$$\Rightarrow A\bar{B} + B\bar{B} \Rightarrow A\bar{B}$$

$$\boxed{E = A\bar{B}}$$

$$F = \overline{D + E} \Rightarrow \overline{\bar{A}B + A\bar{B}}$$

$$\Rightarrow \overline{\bar{A}B}\cdot\overline{A\bar{B}}$$

$$\Rightarrow (\bar{\bar{A}} + \bar{B})\cdot(\bar{A} + \bar{\bar{B}})$$

$$\Rightarrow (A + \bar{B})\cdot(\bar{A} + B)$$

$$\Rightarrow A\bar{A} + \bar{A}\bar{B} + B\bar{B} + AB$$

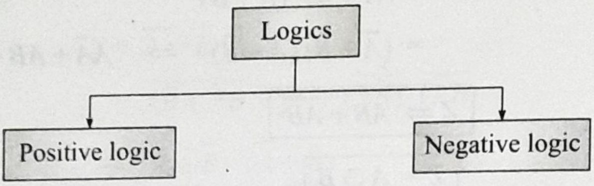
$$\boxed{F \Rightarrow AB + \bar{A}\bar{B}}$$

$$\boxed{F = A \odot B}$$

1.3 POSITIVE LOGIC AND NEGATIVE LOGIC

Digital electronics is based on binary '0' and '1' and practically the binary values a represented by voltage levels. Under normal operating conditions, the voltage appli to any input terminal of a logic circuit is restricted to have one of the two nomin values.

Need of Logics: For identification of associativity of logic variables with electric signals.



(*i*) **Positive Logic:** It is a logic in which logic '1' is represented by higher positive voltage level.

Logic '0' is represented by lower positive level that is shown in diagram.

Logic '1' = state 1/True/ON.

Logic '0' = state 0/False/OFF.

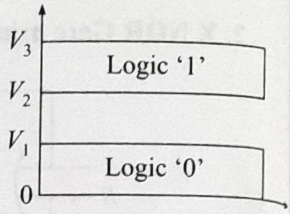


Fig. 1.2. Positive Logic

(*ii*) **Negative Logic:** It is a logic in which logic '1' is represented by lower positive voltage level and Logic '0' is represented by high positive voltage level.

State 1/True/ON = Logic '0'

State 0/False/OFF = Logic '1'

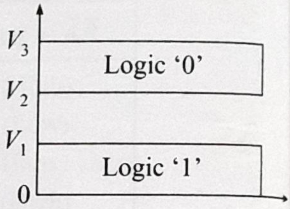


Fig. 1.3. Negative Logic

An electronic logic gate can perform different operations depends upon logic.

For example:

A gate can work as AND gate – for Positive Logic

OR gate – For Negative Logic.

1.4 BOOLEAN ALGEBRA

It is one tool to reduce logical expressions and mathematics of logical expression. The reduced expression can then be implemented with a smaller, simpler circuit, which in turn saves the prices of unnecessary gates, reduces the number of gates needed and reduces the power and the amount of space required by those gates.

In Boolean algebra there are two levels:

Binary 1 – High level.

Binary 0 – Low level.

Basic Laws of Boolean Algebra:

(*i*) Commutative Law

(*ii*) Associative Law

(*iii*) Distributive Law.

Explanation:

Commutative Law

$A + B = B + A$ → Commutative law of addition (OR Logic)

$A \, . \, B = B \, . \, A$ → Commutative law of multiplication (AND Logic)

Associative Law

Associative law of addition → $A + (B + C) = (A + B) + C$

Associative law of multiplication → $A\,(BC) = (AB)\,C$

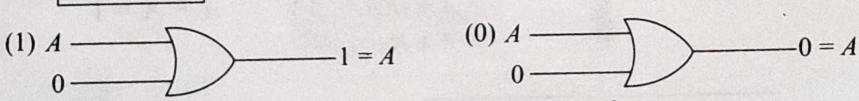
Distributive Law

$A\,(B + C) = AB + AC$ → OR function

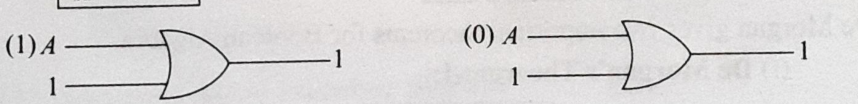
$(A + B)C = AC + BC$ → OR function

1.4.1 Basic Rules of Boolean Algebra

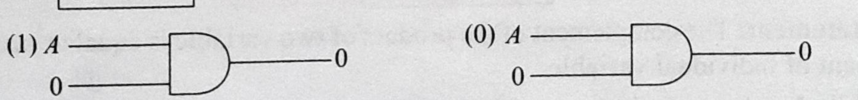
1. $\boxed{A + 0 = A}$



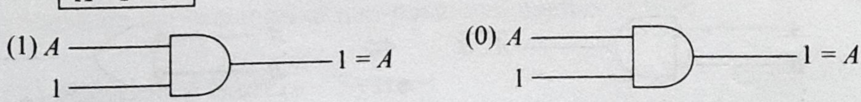
2. $\boxed{A + 1 = 1}$



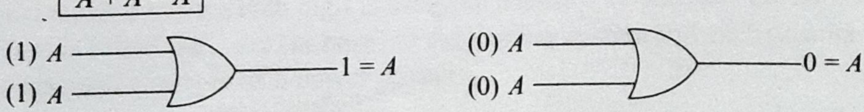
3. $\boxed{A \cdot 0 = 0}$



4. $\boxed{A \cdot 1 = A}$



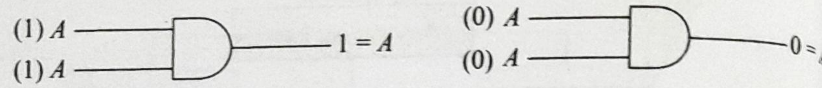
5. $\boxed{A + A = A}$



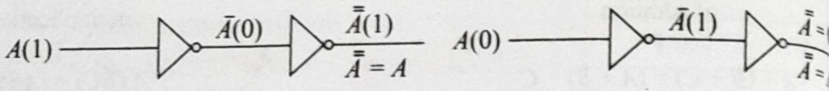
6. $\boxed{A + \bar{A} = 1}$

7. $\boxed{A \cdot \bar{A} = 0}$

8. $\boxed{A \cdot A = A}$



9. $\boxed{\bar{\bar{A}} = A}$



10. (*i*) $\boxed{A + AB = A}$

Proof:

$$
\begin{aligned}
&A + AB \\
&= A(1 + B) && \text{As } 1 + B = 1 \\
&= A(1) && A \cdot 1 = A \\
&= A
\end{aligned}
$$

(*ii*) $\boxed{A + \bar{A}B = A + B.}$

Proof:

$$
\begin{aligned}
&A + \bar{A}B \\
&= A + AB + \bar{A}B && \text{As } A + AB = A \\
&= A + B(A + \bar{A}) && A + \bar{A} = 1 \\
&= A + B.
\end{aligned}
$$

1.4.2 De Morgan's Theorem

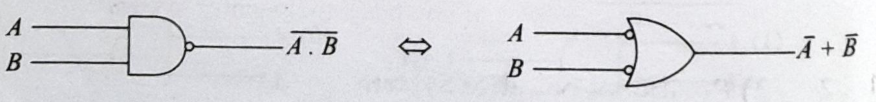
De Morgan gives two important theorems for Boolean Algebra.

(*i*) **De Morgan's Theorem-I:**

$$\boxed{\overline{AB} = \bar{A} + \bar{B}}$$

Statement: The complement of the product of two variable is equal to sum of compl ment of individual variable.

Gate Implementation:



Truth Table:

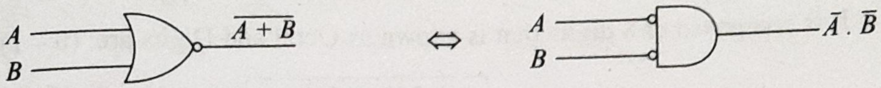
A	B	$\overline{AB}$	$\bar{A}+\bar{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

(*ii*) **De Morgan's Theorem-II:**

$$\boxed{\overline{A+B} = \bar{A} \cdot \bar{B}}$$

Statement: The complement of the sum of two variables is equal to product of complement of individual variables.

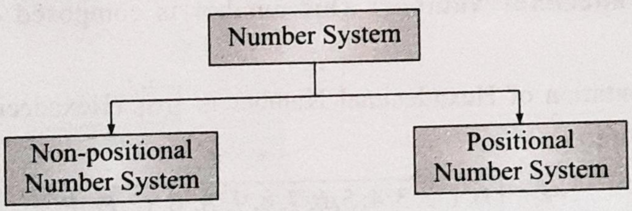
Gate Implementation:



Truth Table:

A	B	$\overline{A+B}$	$\bar{A} \cdot \bar{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

1.5 NUMBER SYSTEM



1.5.1 (1) Non-Positional Number System

It is the Roman numerals number system, where same digits are assigned a unique alphabetic symbol (I, V, L, C etc.) and the value of the number is given by the "sum or difference" of digits.

Example: $I(1)$, $II(1 + 1)$, $IV(5 - 1 = 4)$, $XX(10 + 10)$.

So, these are the examples of non-positional number system.

1.5.2 (2) Positional Number System

In positional number system each digit is assigned a unique symbol and the number of symbols used is called the radix (or base) of the number system and each position in a number is assigned a weight as a power of radix.

An the basis of power of Radix there are different types of number system ar

(*i*) Binary Number

(*ii*) Octal Number

(*iii*) Decimal Number

(*iv*) Hexadecimal Number.

(*i*) **Binary Number:** It is composed of only two digits either '0' or '1' and the are known as Bits.

Representation of Binary Number is $\boxed{2}$

Values under the Binary Number are $\boxed{0, 1}$

Binary weights $2^x = 2^{-2}\ 2^{-1}\ 2^0\ 2^1\ 2^2 \ldots$

(*ii*) **Octal Number:** Representation of octal number is $\boxed{8}$

It is composed of 8 digits that is known as Octal and Digits are: (0 – 1)

$$\boxed{0, 1, 2, 3, 4, 5, 6, 7}$$

It can represent 3 bit binary Number.

Octal weights $8^x = 8^{-2}\ 8^{-1}\ 8^0\ 8^1\ 8^2 \ldots$

(*iii*) **Decimal Number:** It is composed of ten digits. Representation of Decim number is $\boxed{10}$

Due to these ten digits it is known as Decimal and Digits are: (0 – 9)

$$\boxed{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}$$

Decimal weights $10^x = 10^{-2}\ 10^{-1}\ 10^0\ 10^1\ 10^2 \ldots$

(*iv*) **Hexadecimal Number:** This number is composed of 16 digits an characters.

Representation of Hexadecimal Number is $\boxed{16}$ (Hexadecimal Number are 0–9, $A - F$ (10 – 15)

$$\boxed{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}$$

Hexadecimal number system is convenient in digital because each hexadecimal represents, a 4-bit binary Number.

1.5.3 Rules for Conversions of Number System

1. **To Convert:**

$$\left.\begin{array}{l}\text{Binary (2)}\\ \text{Octal (8)}\\ \text{Hexadecimal (16)}\end{array}\right] \longrightarrow \text{Decimal (10)}$$

For this conversion of number system **multiplication with weight method** is used or **sum of weight method**.

2. **To Convert:**

$$\text{Decimal (10)} \longrightarrow \left[\begin{array}{l}\text{Binary (2)}\\ \text{Octal (8)}\\ \text{Hexadecimal (16)}\end{array}\right.$$

We have to use the **division method by radix** for conversion of number system.

Explanations of Conversion:

1. Binary to Decimal 2. Octal to Decimal 3. Hexadecimal to Decimal	Sum of Weight Method
4. Decimal to Binary 5. Decimal to Octal 6. Decimal to Hexadecimal	Division Method by Radix

1.5.4 Conversions with Example

1. **Binary to Decimal:** (2 to 10)

It means values given in the binary number have to convert in the decimal values.

Example:

$$(0\ 1\ 0\ 1\ 1)_2 = (?)_{10}.$$

Solution:

$$\begin{array}{ccccc}(0 & 1 & 0 & 1 & 1)_2 = (\ )_{10}.\\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow\\ 4 & 3 & 2 & 1 & 0\end{array}$$

$$\begin{aligned}\text{Sum of Weight Method} &= 2^0 \times 1 + 2^1 \times 1 + 2^2 \times 0 + 2^3 \times 1 + 2^4 \times 0\\ &= 1 \times 1 + 2 \times 1 + 4 \times 0 + 8 \times 1 + 16 \times 0\\ &= 1 + 2 + 8\\ &= 11\end{aligned}$$

$$\boxed{(01011)_2 = (11)_{10}}$$

2. **Octal to Decimal**: (8 to 10)

It means values given in the octal number have to convert in the decimal values.

Example:

$$(4\ 5\ 7)_8 = (\ )_{10}.$$

Solution:

$$\begin{array}{ccc}(4 & 5 & 7)_8 = (\ )_{10}.\\ \uparrow & \uparrow & \uparrow\\ 2 & 1 & 0\end{array}$$

$$\left.\begin{aligned}&= 8^0 \times 7 + 8^1 \times 5 + 8^2 \times 4\\ &= 7 \times 1 + 8 \times 5 + 64 \times 4\\ &= 7 + 40 + 256\\ &= 303\end{aligned}\right]\ \text{Sum of weight Method}$$

$$\text{Answer} = (457)_8 = (303)_{10}$$

3. **Hexadecimal to Decimal:** (16 to 10)

It means values given in the hexadecimal number have to convert in the d... mal values.

Example: $(A\ 7\ B)_{16} = (\ )_{10}$

Solution:

$$(A\ 7\ B)_{16} = (\ )_{10}.$$
$$\uparrow\ \uparrow\ \uparrow$$
$$2\ \ 1\ \ 0$$
$$= (10\ 7\ 11)_{16} = (\ )_{10}.$$
$$\uparrow\ \uparrow\ \uparrow$$
$$2\ \ 1\ \ 0$$
$$= 16^0 \times 11 + 16^1 \times 7 + 16^2 \times 10$$
$$= 11 + 112 + 256 \times 10$$
$$= 123 + 2560$$
$$= 2683$$

Sum of weight Meth...

Answer $(A\ 7\ B)_{16} = (2683)_{10}$

4. **Decimal to Binary:** (10 to 2)

Now, values are given in the decimal no. and have to convert in the bin... values. Using division by radix method.

Example: $(25)_{10} = (?)_2$

Solution: Now divide 25 by 2.

2	25
2	12 – 1
2	6 – 0
2	3 – 0
2	1 – 1
	0 – 1

Divide by Radix Meth...

For result with values from bottom to top.

$$= 11001$$

Answer: $(25)_{10} = (11001)_2$

5. **Decimal to Octal:** (10 to 8)

Now, values given in the decimal number have to convert in the octal values using the division by radix method.

Example: $(89)_{10} = (?)_8$

Solution: Divide 89 by 8.

8	89
8	11 – 1
8	1 – 3
	0 – 1

$= 131$

Answer: $(89)_{10} = (131)_8$

6. **Decimal to Hexadecimal:** (10 to 16)

In this values given in the decimal values have to convert in the hexadecim... values.

Example: $(289)_{10} = (?)_{16}$

Solution: Divide 289 by 16

$$\begin{array}{r|l} 16 & 289 \\ \hline 16 & 18 - 1 \\ \hline 16 & 1 - 2 \\ \hline & 0 - 1 \end{array} \Bigg\uparrow \; = 121$$

Answere: $(289)_{10} = (121)_{16}$

Example: $(678)_{10} = (?)_{16}$

Solution: Divide 678 by 16.

$$\begin{array}{r|l} 16 & 678 \\ \hline 16 & 42 - 2 \\ \hline 16 & 2 - 10 \\ \hline & 0 - 2 \end{array} \Bigg\uparrow \; = \begin{array}{ccc} 2 & 10 & 2 \\ \uparrow & \uparrow & \uparrow \\ 2 & A & 2 \end{array} \Big] \text{ Hexadecimal value.}$$

Answer: $(678)_{10} = (2A2)_{16}$

1.5.5 Fractional Part Conversion

(*i*) Binary to Decimal Number
(*ii*) Octal to Decimal Number
(*iii*) Hexadecimal to Decimal Number

] All are same as previous method.

(*i*) **Binary to Decimal Number (2 to 10):** The method used for conversion for fractional part is same that is sum of weight method.

Example: $(101.101)_2 = (?)_{10}$

Solution:

$$\begin{array}{cccccc} (1 & 0 & 1 . & 1 & 0 & 1)_2 = (?)_{10} \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ 2 & 1 & 0 & -1 & -2 & -3 \end{array}$$

$$\left.\begin{aligned} &= 2^0 \times 1 + 2^1 \times 0 + 2^2 \times 1 + 2^{-1} \times 1 + 2^{-2} \times 0 + 2^{-3} \times 1 \\ &= 1 \times 1 + 2 \times 0 + 4 \times 1 + \frac{1}{2} \times 1 + \frac{1}{4} \times 0 + \frac{1}{8} \times 1 \\ &= 1 + 0 + 4 + \frac{1}{2} + 0 + \frac{1}{8} \\ &= 5 + \frac{1}{2} + \frac{1}{8} \\ &= 5 + \frac{01}{2} + \frac{1}{8} \end{aligned}\right] \text{Sum of weight Method.}$$

$$= \frac{40 + 4 + 1}{8} \Rightarrow \frac{45}{8} \Rightarrow 5.625$$

Answer: $(101.101)_2 = (5.625)_{10}$

Some Unsolved Example:

(*i*) $(47.65)_8 = (?)_{10}$

(ii) $(A2.C7)_{16} = (?)_{10}$

(iii) Decimal to Binary Number

(iv) Decimal to Octal Number

(v) Decimal to Hexadecimal Number

(ii) **Decimal to Binary Number:** The method used for the conversi fractional part is same in these case that is division by Radix.

Examples: $(18.8125)_{10} = (?)_2$.

Solution:

$$\underbrace{18}_{\text{Division Method}} \, . \, \underbrace{8125}_{\text{By Multiply method}}$$

2	18
2	9 – 0
2	4 – 1
2	2 – 0
2	1 – 0
	0 – 1

$$\begin{array}{r} .8125 \\ \times\ 2 \\ \hline 1 \leftarrow 1).6250 \\ \times\ 2 \\ 1 \leftarrow 1).2500 \\ \times\ 2 \\ 1 \leftarrow 0).5000 \\ \times\ 2 \\ 1 \leftarrow 1).0000 \end{array}$$

Multiply will 2 till the result will not give zero after point and if the value is repeating then write complement (–) one the repeated values.

$(.1101)_2$

Answer: $(18.8125)_{10} = (10010.1101)_2$

Some Unsolved Example:

(i) $(14.765)_{10} = ()_8$

(ii) $(27.745)_{10} = ()_{16}$

1.5.6 Some Different Conversion

(i) Binary to Octal Number

(ii) Binary to Hexadecimal Number.

For these two method we will use the same methods used previously.

(i) **Binary to Octal Number (2 to 8):** For this first convert (2 to 10) binar decimal number and then convert the (10 to 8) decimal values in octal number.

For 2 to 10] use sum of weight method.

10 to 8] use division by Radix method.

And this is same for the second conversion.

Some unsolved Examples are:

(i) $(10110)_2 = (?)_8$

(ii) $(476)_8 = (?)_2$

(iii) $(10111)_2 = (?)_{16}$

(iv) $(ABC)_{16} = (?)_{16}$

(v) $(452)_8 = (?)_{16}$