2	**Boolean Algebra for logic circuits:** Basic Logic variables and logic functions -NOT, AND, NOR, XOR, OR, XNOR, NAND	**7**	**7**

	idealized logic gates and symbols Truth tables, Basic theorems and properties of Boolean algebra, DeMorgan's rules Axiomatic definition of Boolean algebra basic theorems and properties of boolean algebra		

Digital Logic circuits

- Digital Logic circuits operate in the binary mode where each input and output is either 0 or 1.
- The 0 and 1 do not represent actual numbers but instead represent the state of the voltage variable or logic level.
- So in the digital logic field, several other terms are used synonymously with 0 and 1.

LOGIC 1	LOGIC 0
HIGH	LOW
TRUE	FALSE
ON	OFF
+5V	0V

Logic Gates

- The most basic digital devices are called gates.
- Gates got their name from their function of allowing or blocking (gating) the flow of digital information.
- A gate has one or more inputs and produces an output depending on the input(s).
- A gate is called a combinational circuit.
- Three most important gates are: AND, OR, NOT

What is Logic Gate?

- Digital gate is a Digital Device used to perform the logic operation
- Logic gates (or simply gates) are the fundamental building blocks of digital circuitry
- Electronic gates require a power supply.

Gate INPUTS are driven by voltages having two nominal values,
e.g. 0V and 5V representing logic 0 and logic 1 respectively.

The OUTPUT of a gate provides two nominal values of voltage only,
e.g. 0V and 5V representing logic 0 and logic 1 respectively.

- In general, there is only one output to a logic gate except in some special cases.

Q. Define any two characteristics of logic gates

- Logic gates are the logic circuits which act as the basic building blocks of any digital system. It is an **electronic circuit** having one or more than one inputs and only one output.
- The relationship between the input and the output is based on a **"certain logic".**
- Depending on this logic, the gates are named as NOT gate, AND gate, OR, NAND, NOR etc. We have to use different laws, rules and theorems to analyze the digital circuits.
- By connecting the gates, in different ways, we can build circuits **that can perform arithmetic and other functions associated with the human brain.**
- Because they simulate mental processes, gates are often called as **logic circuits**.
- The two important characteristics of logic gates are: **Truth table and Boolean expression.**

Truth table :

- The operation of a logic gate or a logic circuit can be best understood with the help of a table called "Truth Table". The truth table consists of all the possible combinations of the inputs and the corresponding state of output produced by that logic gate or logic circuit.

Boolean expression :

- The relation between the inputs and the outputs of a gate can be expressed mathematically by means of the Boolean Expression.

Classification of Logic Gates

Logic gates

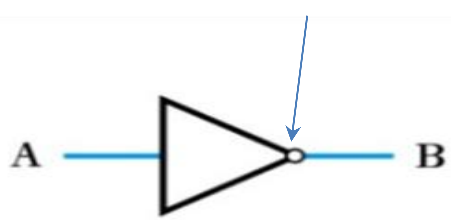
- **Basic**
  - **NOT**
  - **OR**
  - **AND**
- **Universal**
  - **NAND**
  - **NOR**
- **Derived**
  - **EX-OR**
  - **EX-NOR**

Q. Define Basic gates with their symbols and truth table

1) The NOT gate or Inverter is a logic gate having one input (A) and one output (Y). Its symbol and truth table are shown in Fig. below

1) The NOT gate is also known as an "Inverter" because its output is the inverted version or "complement" of its input. This is shown in the truth table of NOT gate.

1) The bubble (o) in the symbol of a NOT gate indicates the inversion operation



(a) Circuit symbol

A	B
0	1
1	0

(b) Truth table

$B = \overline{A}$

(c) Boolean expression

Q. Define AND gate with its symbol and truth table

1) AND is one of the logic operators. It performs the logical multiplication on its inputs.
2) The output is high (Y= 1) if and only if all the inputs to the AND gate are high (1). The output is low (0), if any one or more inputs are low (0). AND gate can have two or more inputs and only one output.
3) The logical symbol of a two input AND gate is as shown below
4) Boolean expression : The expression relating the inputs (A. B) and output (Y) of a gate is called as the expression. The Boolean expression for an AND gate is,

$$Y = A.B$$

where the "dot" between A and B represents multiplication.

(a) Circuit symbol

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

(b) Truth table

$$C = A \cdot B$$

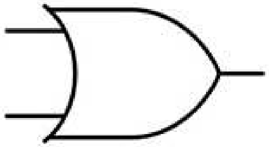
(c) Boolean expression

Q. Define OR gate with its symbol and truth table

1) OR is one of the logic operators. It performs the logical addition on its inputs.
2) The output is high (Y= 1) if any one or all the inputs are high (1). The output is low (0), if both the inputs are low. The logical symbol of a two input OR gate is as shown below
3) Boolean expression : The expression relating the inputs (A. B) and output (Y) of a gate is called as the expression. The Boolean expression for an AND gate is,

$$Y = A + B$$

OR



A	B	Output
0	0	0
1	0	1
0	1	1
1	1	1

Universal Gates

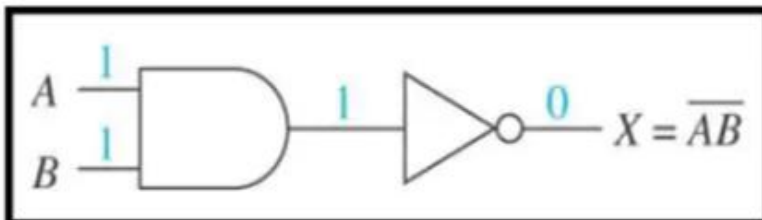
*Q. What are Universal gates ? Explain ***

1. The **NAND and NOR** gates are called as **“Universal gates**” because **it is possible to implement any Boolean expression** with the help of only NAND or NOR gates.

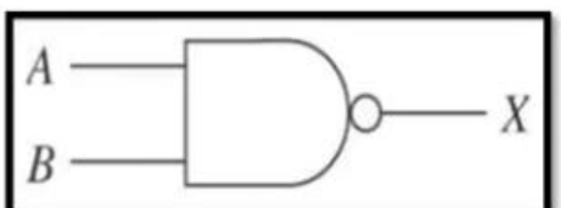
1. Therefore **a user can build any logical circuit** with the help of **only NAND or only NOR gates**.

1. This is a great advantage because a user will have to make a stock of only NAND or NOR gates ICs with him.

What happens when you add a NOT to an AND gate?



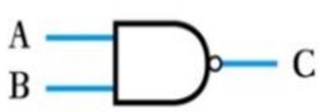
"Not AND" = NAND



A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

1) The term NAND can be split as NOT-AND which means that the NAND operation can be implemented with the combination of an AND gate and a NOT gate i.e. inverter.
2) Thus a NAND gate is equivalent to an AND gate followed by an inverter as shown in Fig. below
3) The truth table of a two input NAND gate is shown below which shows that the output is low (0) if and only if both the inputs are high (1) simultaneously. For all other input combinations the output voltage will be high (1).
4) A NAND gate is called as "Universal Gate" because we can construct AND, OR and NOT gates using only NAND gates.
5)

- **The NAND gate**



(a) Circuit symbol

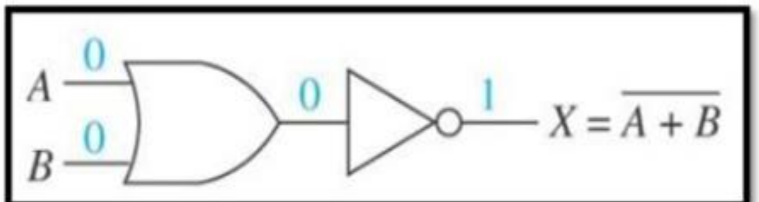
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

(b) Truth table

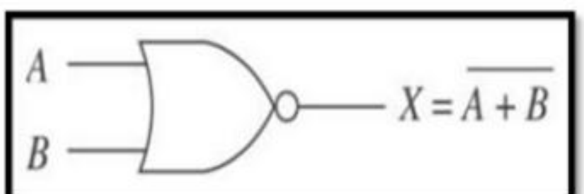
$$C = \overline{A \cdot B}$$

(c) Boolean expression

What happens when you add a NOT to an OR gate?



"Not OR" = NOR



A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

1) This is another universal gate.
2) The word NOR can be split as NOT-OR which means that a NOR operation can be implemented with the combination of an OR gate and a NOT gate, i.e. inverter. Thus a NOR gate is equivalent to an OR gate followed by an inverter as shown in Fig.
3) The symbol of a two input NOR gate is shown in Fig. 3.10.1(a), where a bubble (o) on the output side represents inversion. The truth table of a two input NOR gate is shown in
4) Fig. shows that "the output of a NOR gate is high (1) if and only if all its inputs are low (0) simultaneously". The output of a two input NOR gate is low (0) if any one or all the inputs are at high (1) level.

(a) Circuit symbol

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

(b) Truth table

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

(c) Boolean expression

EX-OR and EX-NOR Gates :

- EX-OR and EX-NOR gates are special type of gates. They can be used for applications such as half adder, full adder and subtractors.
- These gates are also called as Derived gates

1. The exclusive-OR gate is abbreviated as EX-OR gate or sometimes as X-OR gate.
2. An EX-OR gate having two input terminals and one output terminal is shown in Fig.
3. The truth table of a two input EX-OR gate is hown in Fig, which shows that, when both the inputs are at identical logic levels (A = B), the output is low (0) i.e. Y = 0 for A = B = 0 or A = B = 1, and the output is high (1) when A and B are not equal.

- **The Exclusive OR gate**
  - Exclusive OR gate are true if either input is true but not both.
  - The **Symbol** and **Truth Table** is shown below.
  - **Output will be high for different inputs**

(a) Circuit symbol

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

(b) Truth table

$$C = A \oplus B$$

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

(c) Boolean expression

1) The word EX-NOR is a short form of exclusive-NOR. Exclusive-NOR means NOT-exclusive OR, so EX-NOR gate is equivalent to an EX-OR gate followed by a NOT gate.
2) The symbol for a two input EX-NOR gate is as shown in Fig. and its truth table is given in Fig. which shows that the output of an EX-NOR gate is high (1) if both the inputs are identical (A = B) and the output is low (0) if the inputs are not identical (A #B).
3) The Boolean expression for an EX-NOR gate is given by,

• The Exclusive NOR gate

- The **Symbol** and **Truth Table** is shown below.
- **Output will be high for same inputs**

(a) Circuit symbol

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

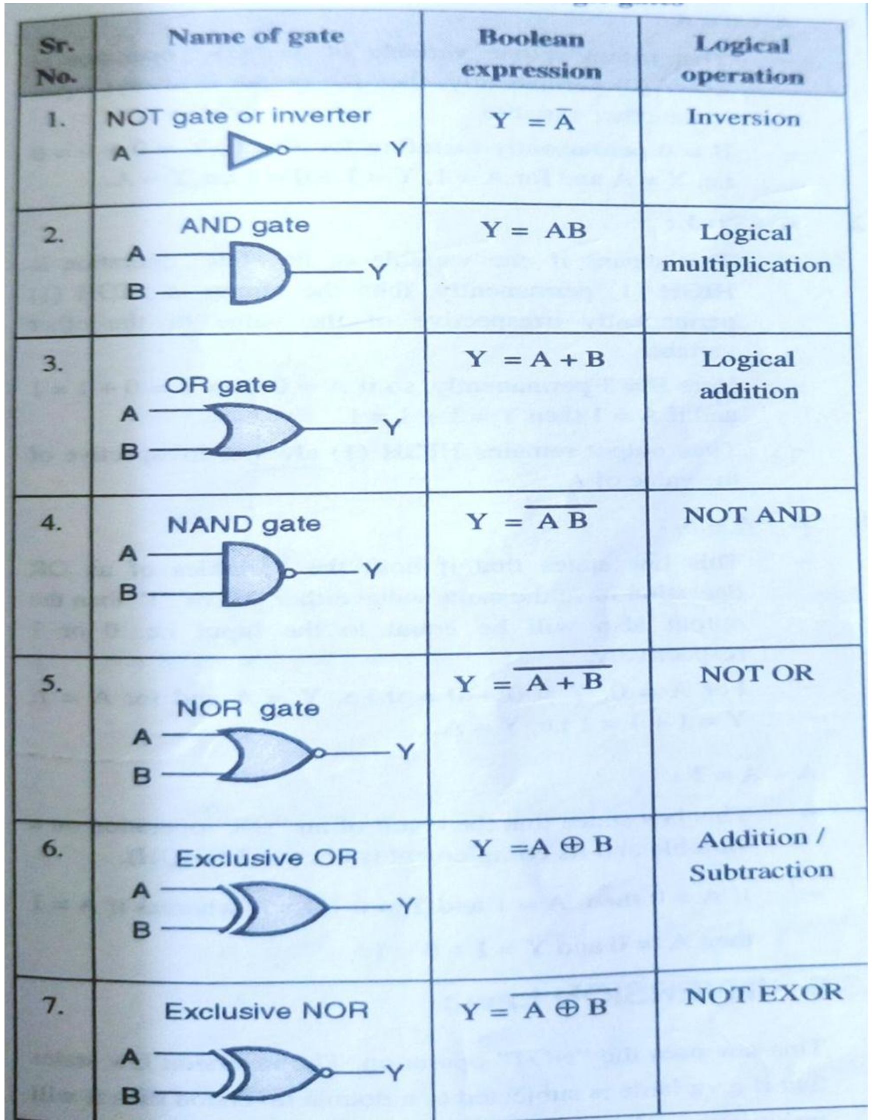
(b) Truth table

$$\overline{A}\,\overline{B}+A\,B$$

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

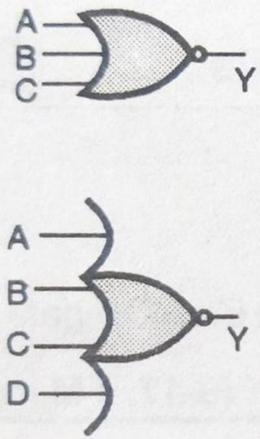
(c) Boolean expression

Q. Draw symbols and logic expressions for all gates



Sr. No.	Name of gate	Boolean expression	Logical operation
1.	NOT gate or inverter	$Y = \overline{A}$	Inversion
2.	AND gate	$Y = AB$	Logical multiplication
3.	OR gate	$Y = A + B$	Logical addition
4.	NAND gate	$Y = \overline{AB}$	NOT AND
5.	NOR gate	$Y = \overline{A + B}$	NOT OR
6.	Exclusive OR	$Y = A \oplus B$	Addition / Subtraction
7.	Exclusive NOR	$Y = \overline{A \oplus B}$	NOT EXOR

Symbols :



(a)

Truth table

Inputs			Output
A	B	C	Y
0	0	0	1
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	0

Output is low if at least one input is 1

(b) Three input NOR gate

(B-472) **Fig. 3.14.4 : Symbols, Boolean equations and truth table for multiple input NOR gate**

Boolean equations :

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

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

NAND Gate as Universal gate

- *NOT gate using NAND gate*

1. NOT gate (Inverter) using NAND gate :

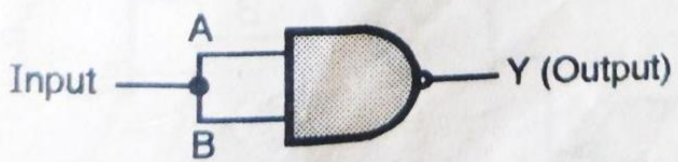
- Fig. 3.15.1(a) shows the realization of a NOT gate (inverter) using a two input NAND gate.
- As both the inputs of the NAND are connected together, we can write that,

$$\text{Input} = A = B = A$$

- So output is given by,

$$Y = \overline{A \cdot B} = \overline{A \cdot A} \qquad \ldots\text{since } A = B = A$$

$$\text{But } A \cdot A = A \qquad \therefore \quad Y = \bar{A}$$

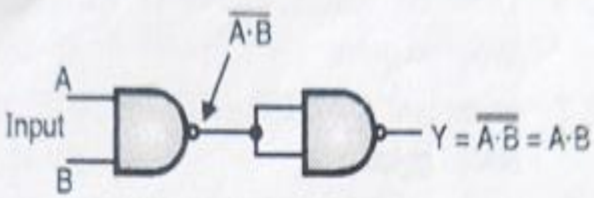


(B-505) **Fig. 3.15.1(a) : NOT using NAND**

2. AND gate using NAND :

$$Y = A \cdot B \quad \text{...AND gate}$$

$$Y = \overline{\overline{A \cdot B}} \quad \text{...Double inversion}$$



(B-506) **Fig. 3.15.1(b) : AND gate using NAND**

$$\therefore\ Y = A \cdot B = \overline{\overline{A \cdot B}} \quad \text{...(3.15.1)}$$

- Equation (3.15.1) can be realized using only NAND gates as shown in Fig. 3.15.1(b).

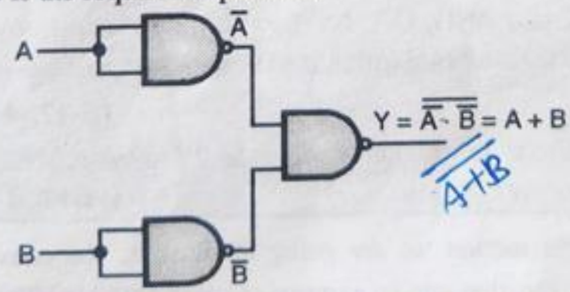
3. OR gate using NAND :

$$Y = A + B \quad \text{... OR gate}$$

$$Y = \overline{\overline{A + B}} \quad \text{... Double inversion}$$

$$\therefore\ Y = \overline{\bar{A} \cdot \bar{B}} \quad \text{... De Morgan's law}$$

This is the required expression.



(B-507) **Fig. 3.15.1(c) : OR gate using NAND gates**

- So $Y = A + B = \overline{\bar{A} \cdot \bar{B}}$ and Fig. 3.15.1(c) shows the realization.

NOR Gate as Universal gate

1. NOT gate (inverter) using NOR :

- Fig. 3.15.2(a) shows the realization of a NOT gate using only NOR gate.
- As both the inputs of the NOR gate are connected together, we can write that,

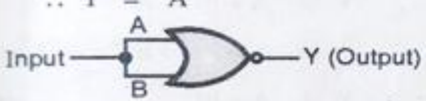
$$A = B = A$$

- So output of NOR is given by,

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

- But A + A = A

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



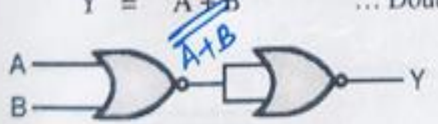
(B-511) **Fig. 3.15.2(a) : NOT gate using NOR gate**

- This is the Boolean expression of an inverter. So Fig. 3.15.2(a) indeed represents an inverter.

2. OR using NOR :

$$Y = A + B \qquad \text{... OR gate}$$

$$Y = \overline{\overline{A + B}} \qquad \text{... Double inversion}$$



(B-512) **Fig. 3.15.2(b) : OR using only NOR gates**

This is the required expression.

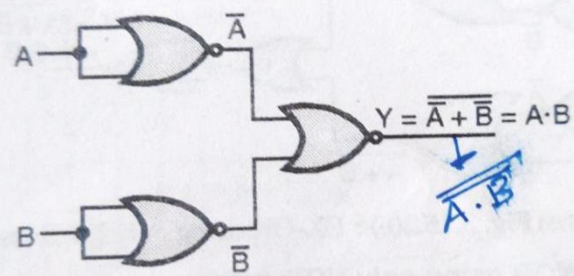
- Fig. 3.15.2(b) shows the realization of OR gate using only NOR gates.

3. AND using only NOR gates :

$$Y = A \cdot B \qquad \text{... AND gate}$$

$$Y = \overline{\overline{A \cdot B}} \qquad \text{...Double inversion}$$

- This is the required expression, Fig. 3.15.2(c) shows the realization of AND gate using NOR gates only.



(B-513) **Fig. 3.15.2(c) : AND gate using only NOR gates**

NAND gate using only NOR gates :

- Boolean expression for a NAND gate is,

$$Y = \overline{A \cdot B} = \bar{A} + \bar{B} \qquad \text{...using De Morgan's theorem}$$

- Take double inversion of RHS to get,

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

- This is the required expression. Fig. 3.15.2(d) shows the realization of NAND gate using only NOR gates.

Boolean Laws

*Q. State any four Boolean laws used to reduce Boolean Expression ****

1. Commutative law.
2. Associative law.
3. Distributive law.
4. OR law.
5. AND law
6. INVERSION law

- Boolean Algebra is used to analyze and simplify the digital (logic) circuits. Since it uses only the binary numbers i.e. 0 and 1 it is also called as "Binary Algebra", or "Logical Algebra".
- It was invented by George Boole in the year 1854.

Rules in boolean algebra :

- There are some rules to be followed while using a Boolean algebra, these are:

1. Variables used can have only two values. Binary 1 for HIGH and Binary 0 for LOW.

2. Complement of a variable is represented by a overbar (-). Thus complement of variable B is represented as B. Thus if

B=0 then B = 1 and if B = 1 then B = 0.

3. ORing of the variables is represented by a plus (+) sign between them. For example ORing of A, B, C is represented as A + B + C.

4. Logical ANDing of the two or more variables is represented by writing a dot between them such as A B C.D.E. Sometimes the dot may be omitted like ABCDE.

Commutative Law:

- Any binary operation which satisfies the following expression is known as commutative operation :

1. A.B=B.A
2. A+B=B+A

- Thus the commutative law states that we can change the sequence of the variables (inputs) without having any effect on the output of a logic circuit.

Associative Law:

- This law states that the order in which the logic operations are performed is not at all important ultimate outcome is the same.

i.e. $(A.B)\,C = A\,(B.C)$ and

$$(A+B) + C = A + (B+C)$$

Distributive Law:

- The distributive law states that,

$$A\,(B+C) = AB + AC$$

AND Laws :

- These laws are related to the AND operation therefore they are called as "AND" laws. The AND laws are as follows:

1. **A.0=0:**

That means if one input of an AND operation is permanently kept at the LOW (0) level, then the output is zero irrespective of the other variable.

2. **A.1 =A**:

- That means if one input of an AND operation is HIGH (1) permanently then the output is always equal to the other variable.

3. **A.A=A**

- That means if both the inputs in an AND operation have the same value either “0" or "1" then the output will also have the same value as that of the input.

4.A.$\overline{A}$=0

This law states that the result of an "AND" operation on a variable (A) and its complement (A) is always LOW (0)

OR laws

- These laws use the OR operation. Therefore they are called as OR laws. The OR laws are as follows:

1. **$A + 0 = A$:**

- That means if one variable of an "OR" operation is LOW (0) permanently, then the output is always equal to the other variable.

2. **$A + 1 = 1$:**

- That means if one variable of an "OR" operation is HIGH (1) permanently, then the output is HIGH (1) permanently irrespective of the value of the other variable.

3. **$A + A = A$:**

- This law states that if both the variables of an OR operation have the same value either "0" or "1" then the output also will be equal to the input i.e. 0 or 1 respectively.

4. **$A + \overline{A} = 1$:**

- This law states that the result of an "OR" operation on a variable and its complement is always 1 (HIGH).

INVERSION Law

- This law uses the “ NOT” operation. The inversion law states that if a variable is subjected to a double inversion then it will result in the original variable itself.

$$\overline{\overline{\mathbf{A}}}=\mathbf{A}$$

Other Important Rules

1. A+AB=A

LHS=A+AB

=A(1+B)

But 1+B=1

∴LHS=A.1=A Since A.1=A

∴ A+AB=A

Hence proved.

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

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

Substitute A = A + AB

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

But $A + \bar{A} = 1$

:: LHS = A + B (1) = A + B

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

Hence proved.

- 3. (A + B) (A + C) = A + BC :

LHS = (A + B) (A + C)
= A .A+A .C+B.A+B.C

But A .A = A and B.A =A.B

:. LHS = A + AC + AB + BC

But A + AB = A

.. LHS = A + AC + BC = A (1 + C) + BC

But (1+ C) = 1

:. LHS = A + BC
.. (A + B) (A + C) = A + BC

...Proved.

Duality theorem.

*Q. State and explain Duality Theorem****

1. According to the duality theorem the following conversions are possible in a given Boolean expression :

1. **Change each AND operation to an OR operation.**
2. **Change each OR operation to an AND operation.**
3. **Complement any 1 or 0 appearing in the expression.**

2. Duality theorem is sometimes useful in creating new expressions from the given Boolean expressions.

3. For example if the given expression is A + 1 = 1 then replace the OR (+) operation by AND ( ) operation and take complement of 1 to write the dual of the given relation as,
A.0 = 0

4. The dual of A.(B+C) = AB + AC is given by,
A+ (B.C) = (A + B). (A + C)

De-Morgan's Theorem

*Q. State and Prove De-Morgans Theorem*******

Theorem 1:

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

1. This theorem states that the complement of the product is equal to the addition of the complement of each term.
2. This can be shown as follows. The left hand side of this theorem represents a NAND gate with inputs A and B whereas The Right Hand Side represents OR gate with inverted inputs.
3. This theorem can be verified by writing a truth table as follows

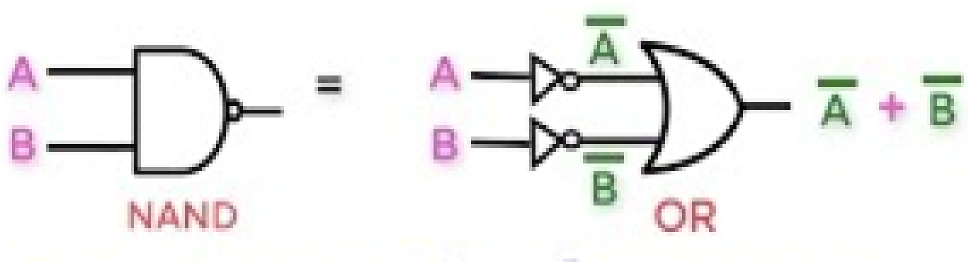


Illustration of De-Morgans First Theorem

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

Truth table

- **Theorem 2:**

$$\boxed{\overline{A+B} = \overline{A}\,.\overline{B}}$$

1. This theorem states that the complement of the addition is equal to the product of the complement of each term.
2. This can be shown as follows. The left hand side of this theorem represents a NOR gate with inputs A and B whereas The Right Hand Side represents AND gate with inverted inputs.
3. This theorem can be verified by writing a truth table as follows

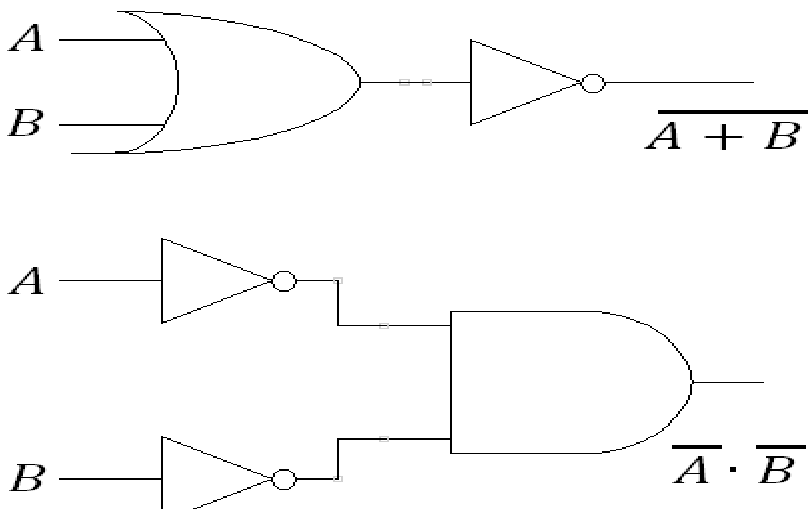


Illustration of De-Morgans First Theorem

A	B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

Truth table

Problem: Simplify the Boolean expression $A + \overline{A} \cdot B$.

Solution:

$$A + \overline{A} \cdot B = A + (B \cdot \overline{A}) = (A + B) \cdot (A + \overline{A}) = (A + B) \cdot 1 = A + B$$

Example 8

Problem: Simplify the Boolean expression $A \cdot (A + B)$.

Solution:

$$A \cdot (A + B) = A \cdot A + A \cdot B = A + A \cdot B = A \cdot (1 + B) = A \cdot 1 = A$$

Example 9

Problem: Simplify the Boolean expression $\overline{A} \cdot A \cdot B$.

Solution:

$$\overline{A} \cdot A \cdot B = (\overline{A} \cdot A) \cdot B = 0 \cdot B = 0$$

Example 5

Problem: Simplify the Boolean expression $(A \cdot B) + (\overline{A} \cdot B)$.

Solution:

$$(A \cdot B) + (\overline{A} \cdot B) = B \cdot (A + \overline{A}) = B \cdot 1 = B$$

Example 6

Problem: Simplify the Boolean expression $A \cdot B + \overline{A} \cdot \overline{B} + A \cdot \overline{B} + \overline{A} \cdot B$.

Solution:

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

$$= A \cdot (B + \overline{B}) + \overline{A} \cdot (B + \overline{B}) = A \cdot 1 + \overline{A} \cdot 1 = A + \overline{A} = 1$$

Example 10

Problem: Simplify the Boolean expression $A \cdot (A + \overline{A} \cdot B)$.

Solution:

$$A \cdot (A + \overline{A} \cdot B) = A \cdot A + A \cdot (\overline{A} \cdot B) = A + (A \cdot \overline{A} \cdot B) = A + 0 = A$$

Example 11

Problem: Simplify the Boolean expression $(A \cdot B) + (\overline{A} \cdot \overline{B})$.

Solution:

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

Example 12

Problem: Simplify the Boolean expression $A \cdot (B + \overline{B}) \cdot (C + \overline{C})$.

Solution:

$$A \cdot (B + \overline{B}) \cdot (C + \overline{C}) = A \cdot 1 \cdot 1 = A$$

Example 13

Problem: Simplify the Boolean expression $(A + B) \cdot (A + \overline{B}) \cdot (A + C)$.

Solution:

$$(A + B) \cdot (A + \overline{B}) \cdot (A + C) = (A + (B \cdot \overline{B})) \cdot (A + C) = (A + 0) \cdot (A + C) = A \cdot (A \cdot$$

Example 14

Problem: Simplify the Boolean expression $A \cdot B + A \cdot C + B \cdot C$.

Solution:

$$A \cdot B + A \cdot C + B \cdot C = A \cdot (B + C) + B \cdot C$$

This expression is already in a relatively simplified form. However, if you want to express it in another form, you can use the Distributive Law:

$$A \cdot B + A \cdot C + B \cdot C = A \cdot (B + C) + B \cdot C = (A + B) \cdot (A + C)$$

Example 15

Problem: Simplify the Boolean expression $A \cdot \overline{B} + \overline{A} \cdot B \cdot C$.

Solution:

$$A \cdot \overline{B} + \overline{A} \cdot B \cdot C = (A + \overline{A}) \cdot \overline{B} + \overline{A} \cdot B \cdot C = 1 \cdot \overline{B} + \overline{A} \cdot B \cdot C = \overline{B} + \overline{A} \cdot B \cdot C$$

Example 1: Simplify the Boolean Expression

Expression: $A \cdot \overline{A} + B \cdot (A + \overline{A})$

Solution:

1. Start with the expression: $A \cdot \overline{A} + B \cdot (A + \overline{A})$.
2. Use the **Complement Law**: $A \cdot \overline{A} = 0$, so the expression becomes $0 + B \cdot (A + \overline{A})$.
3. The **Identity Law** states that $0 + X = X$, so $B \cdot (A + \overline{A})$ remains.
4. According to the **Complement Law**, $A + \overline{A} = 1$, thus the expression simplifies to $B \cdot 1$.
5. Finally, by the **Identity Law**: $B \cdot 1 = B$.

Expression: $(A + \overline{B}) \cdot (A + B)$

Solution:

1. Expand using **Distributive Law:**

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

2. Simplify each term:

   - $A \cdot A = A$ (Idempotent Law)
   - $\overline{B} \cdot B = 0$ (Complement Law)

3. The expression now is:

$$A + A \cdot B + \overline{B} \cdot A + 0 = A + A \cdot B + \overline{B} \cdot A$$

4. Factor A out:

$$A \cdot (1 + B + \overline{B}) = A \cdot 1 = A$$

Final Simplified Expression: A

Example 3: Simplify the Boolean Expression

Expression: $\overline{A \cdot B} + A$

Solution:

1. Apply **De Morgan's Theorem** to the first term:

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

2. Substitute this into the expression:

$$(\overline{A} + \overline{B}) + A$$

3. Rearrange the terms:

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

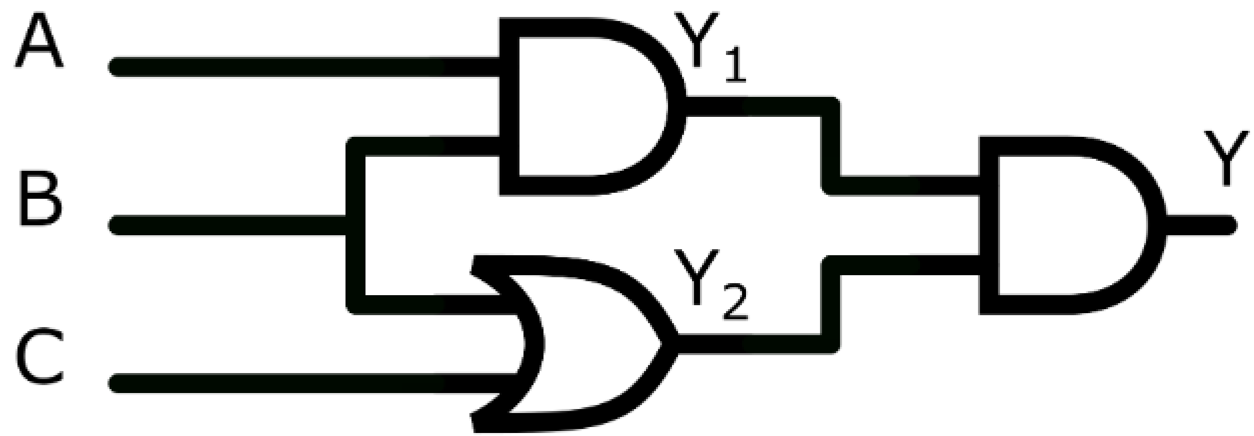
4. Apply the **Complement Law**: $A + \overline{A} = 1$, so the expression becomes:

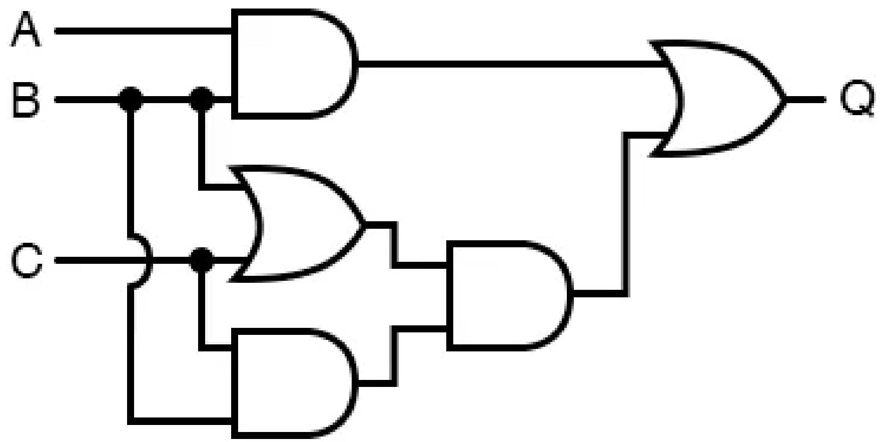
$$1 + \overline{B}$$

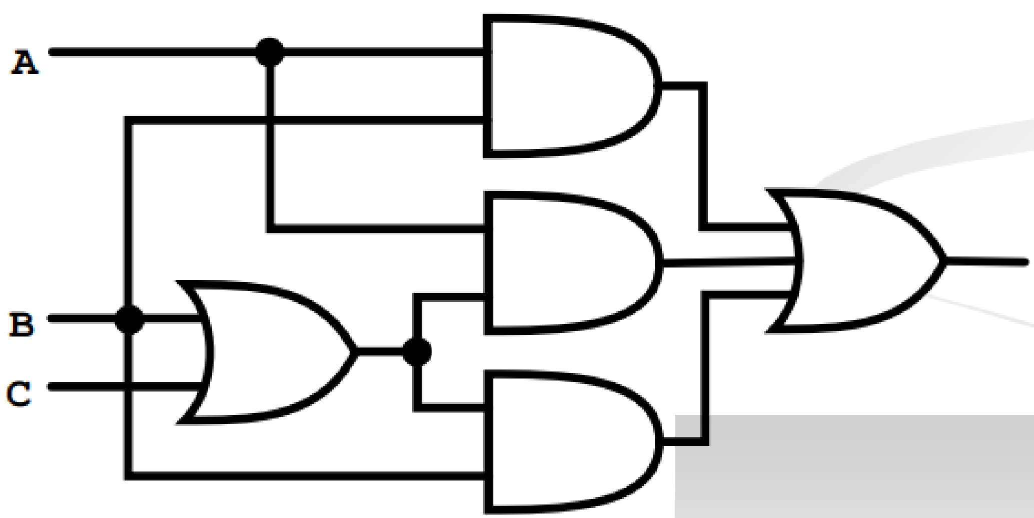
5. The **Domination Law** states $1 + X = 1$.

Final Simplified Expression: 1

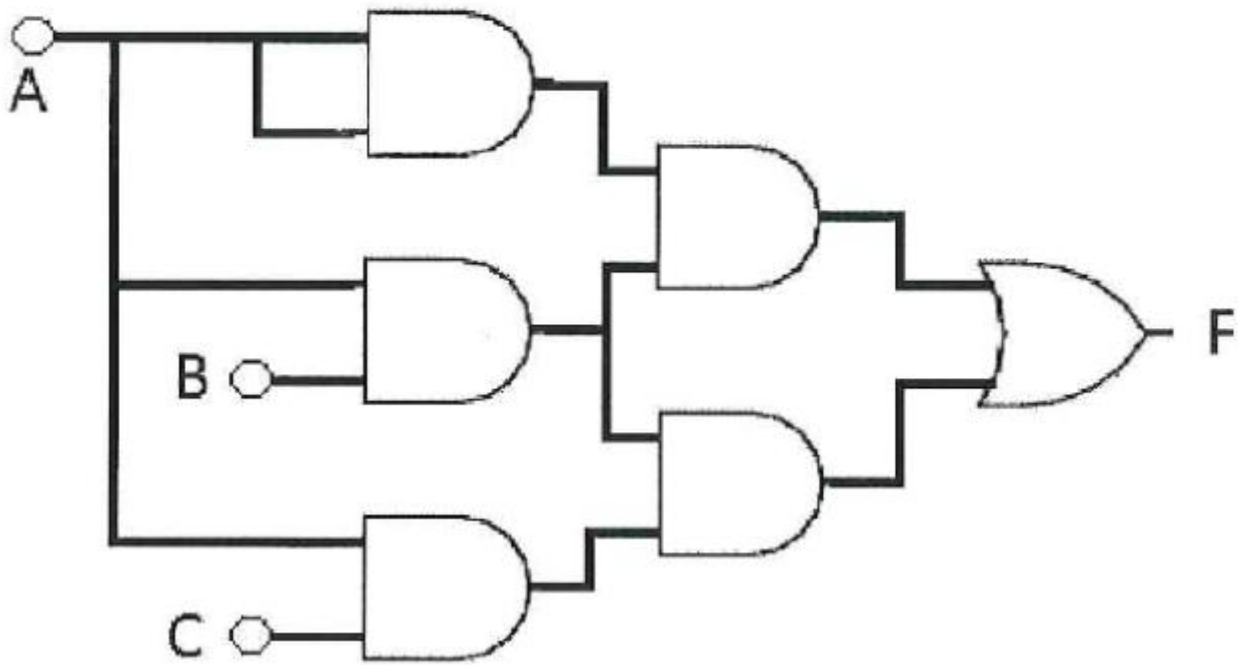
Write the Boolean expression and simplify it



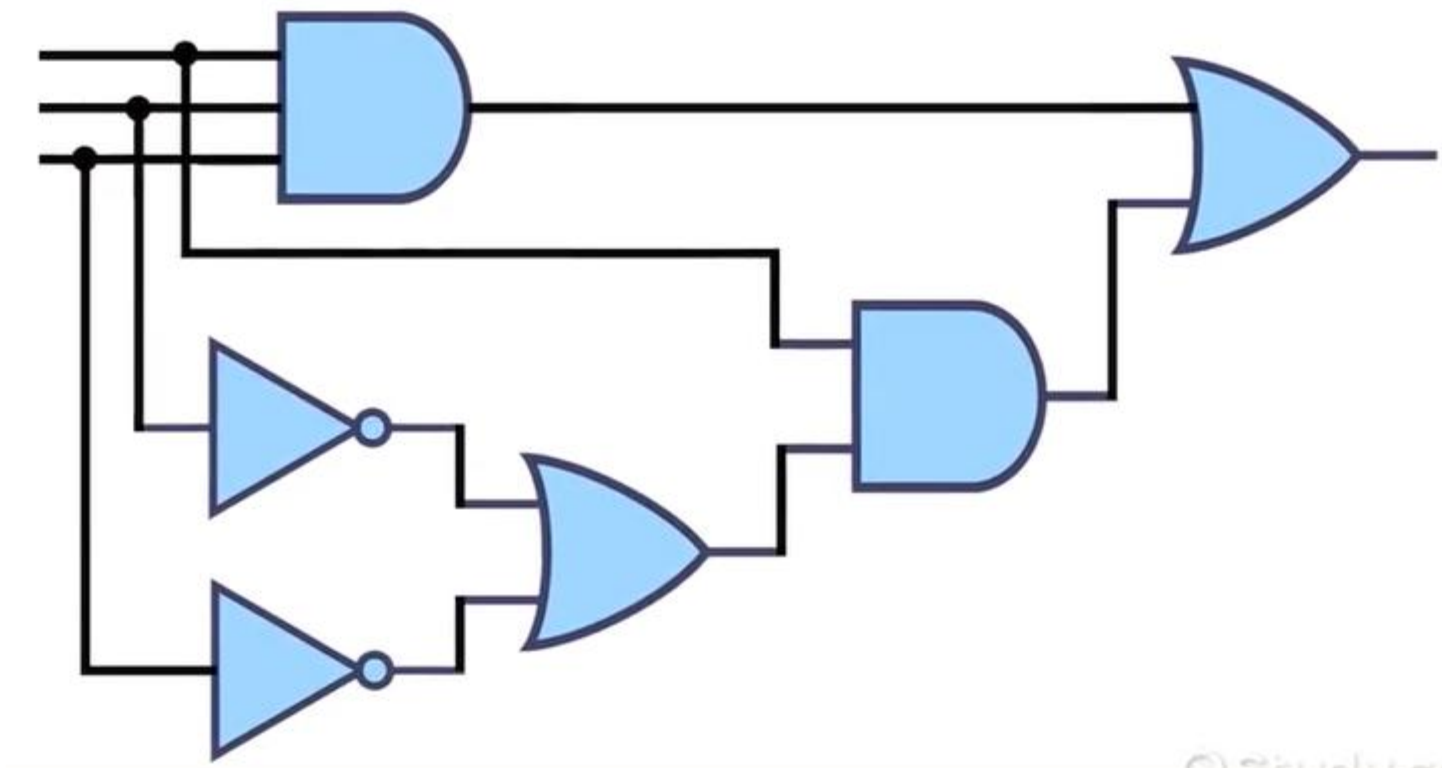

A
B
C
Q


A
B
C

Simplify the following



Simplify the following



Practice Questions

All 7 logic gates- symbol,TT, Boolean Expression
Design gates using NAND
Design gates using NOR
Explain Demorgan's theorem
All boolean algebraic laws
Simplify circuit (Refer last question)
Simplify using Boolean laws