<u>Boolean Algebra</u>

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

Rule in Boolean Algebra

Following are the important rules used in Boolean algebra.

- Variable used can have only two values. Binary 1 for HIGH and Binary 0 for LOW.
- Complement of a variable is represented by an overbar (-). Thus, complement of variable B is represented as $\overline{B}$. Thus if B = 0 then $\overline{B}$ = 1 and B = 1 then $\overline{B}$ = 0.
- 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.
- Logical ANDing of the two or more variable is represented by writing a dot between them such as A.B.C. Sometime the dot may be omitted like ABC.

Boolean Function

- A Boolean Function is described by an algebraic expression called Boolean expression which consists of binary variables, the constants 0 and 1, and the logic operation symbols.

- The function F is equal to 1 if x is 1 or if both y' and z are equal to 1, F is equal to 0 otherwise. Saying y'=1 is equivalent to saying that y=0.

- So F is equal to 1 if x=1 or if yz=01.

Truth Table

- The relationship between a function in a truth table and its binary variables can be represented in a truth table.
- There are eight possible combination for assigning bits to the three variables x, y and z.

Logic Diagrams and Expressions

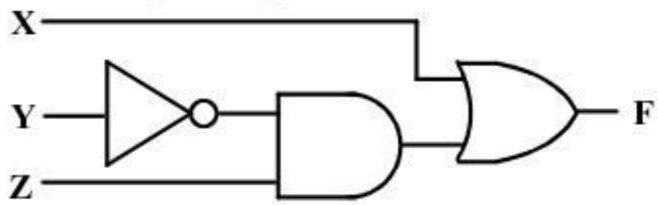
Truth Table

X Y Z	$F = X + \overline{Y} \cdot Z$
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

Logic Equation

$$F = X + \overline{Y}\,Z$$

Logic Diagram



- **Boolean equations, truth tables and logic diagrams describe the same function!**

Boolean Laws

There are six types of Boolean Laws.

Commutative law

Any binary operation which satisfies the following expression is referred to as commutative operation.

(i) A.B = B. A (ii) A + B = B + A

Commutative law states that changing the sequence of the variables does not have 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 irrelevant as their effect is the same.

(i) (A.B).C = A.(B.C) (ii) (A + B) + C = A + (B + C)

Distributive law

Distributive law states the following condition.

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

AND law

These laws use the AND operation. Therefore they are called as AND laws.

(i) $A.0 = 0$ (ii) $A.1 = A$

(iii) $A.A = A$ (iv) $A.\overline{A} = 0$

OR law

These laws use the OR operation. Therefore they are called as OR laws.

(i) $A + 0 = A$ (ii) $A + 1 = 1$

(iii) $A + A = A$ (iv) $A + \overline{A} = 1$

INVERSION law

This law uses the NOT operation. The inversion law states that double inversion of a variable results in the original variable itself.

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

1. Boolean Algebra simplification rules

1. $A+\overline{A}=1$	2. $A+A=A$
3. $A\cdot A=A$	4. $A\cdot\overline{A}=0$
5. $A\cdot(B+C)=A\cdot B+A\cdot C$	6. $A+0=A$
7. $A+1=1$	8. $A\cdot 1=A$
9. $A\cdot 0=0$	10. $A\cdot B=B\cdot A$
11. $A+B=B+A$	12. $B\cdot(A+\overline{A})=B$
13. $A+A\cdot B=A$	14. $A\cdot(A+B)=A$
15. $A+\overline{A}\cdot B=A+B$	16. $A\cdot(\overline{A}+B)=A\cdot B$
17. $\overline{A+B}=\overline{A}\cdot\overline{B}$	18. $\overline{A\cdot B}=\overline{A}+\overline{B}$

De Morgan’s Theoram

De Morgan has suggested two theorems which are extremely useful in Boolean Algebra. The two theorems are discussed below.

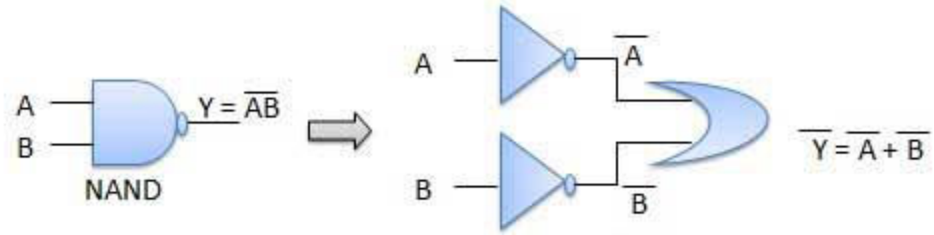
Theorem 1

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

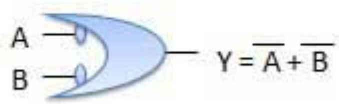
NAND = Bubbled OR

- The left hand side (LHS) of this theorem represents a NAND gate with inputs A and B, whereas the right hand side (RHS) of the theorem represents an OR gate with inverted inputs.

- This OR gate is called as **Bubbled OR.**



NAND $\equiv$ Bubbled OR



Bubbled OR

Table showing verification of the De Morgan's first theorem −

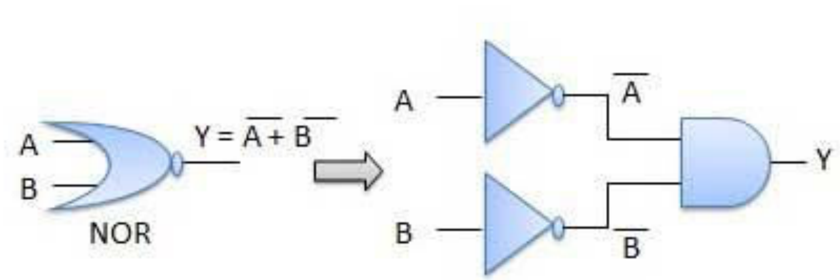
A	B	$\overline{AB}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

Theorem 2

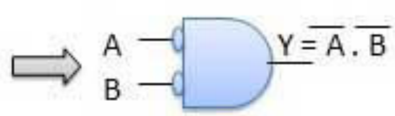
$$\overline{A + B} = \overline{A} . \overline{B}$$

NOR = Bubbled AND

- The LHS of this theorem represents a NOR gate with inputs A and B, whereas the RHS represents an AND gate with inverted inputs.
- This AND gate is called as **Bubbled AND**.



NOR $\equiv$ Bubbled AND



Bubbled AND

Table showing verification of the De Morgan's second 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