BOOLEAN ALGEBRA:

- Basic Definitions
- Axiomatic Definition of Boolean Algebra
- Basic Theorems and properties of Boolean Algebra
- Boolean Functions
- Canonical and Standard Forms, Other Logic Operations
- Digital Logic Gates
- Integrated Circuits

Boolean Algebra: Boolean algebra, like any other deductive mathematical system, may be defined with aset of elements, a set of operators, and a number of unproved axioms or postulates. A *set* of elements is anycollection of objects having a common property. If **S** is a set and ***x*** and ***y*** are certain objects, then **x Î S**denotes that ***x*** is a member of the set **S**, and ***y*** **ÏS** denotes that ***y*** is not an element of **S**. A set with adenumerable number of elements is specified by braces: **A** = {1,2,3,4}, *i.e.* the elements of set **A** are thenumbers 1, 2, 3, and 4. A *binary operator* defined on a set S of elements is a rule that assigns to each pair ofelements from S a unique element from S._ Example: In *a***b*=*c*, we say that * is a binary operator if it specifies a rule for finding *c* from the pair (*a*,*b*)and also if *a*, *b*, *c* Î S.

CLOSURE: The Boolean system is *closed* with respect to a binary operator if for every pair of Boolean values,it produces a Boolean result. For example, logical AND is closed in the Boolean system because it accepts only Boolean operands and produces only Boolean results.
_ A set *S* is closed with respect to a binary operator if, for every pair of elements of *S*, the binary operator specifies a rule for obtaining a unique element of *S*.
For example, the set of natural numbers N = {1, 2, 3, 4, … 9} is closed with respect to the binary operator plus (+) by the rule of arithmetic addition, since for any *a*, *b* Î N we obtain a unique *c* Î N by the operation *a* + *b* = c.

ASSOCIATIVE LAW:A binary operator * on a set *S* is said to be associative whenever (*x* * *y*) * *z* = *x* * (*y* * *z*) for all *x*, *y*, *z* Î S, forall Boolean values x, y and z.

COMMUTATIVE LAW:
A binary operator * on a set *S* is said to be commutative whenever x * *y* = *y* * *x* for all *x*, *y*, *z* ϵ

IDENTITY ELEMENT:
A set ***S*** is said to have an identity element with respect to a binary operation * on S if there exists an element *e* **ϵ** S with the property *e* * *x* = *x* * *e* = *x* for every *x* ϵ S

BASIC IDENTITIES OF BOOLEAN ALGEBRA
Postulate 1 (Definition): A Boolean algebra is a closed algebraic system containing a set *K* of two or more elements and the two operators · and + which refer to logical AND and logical OR

- $x + 0 = x$

- $x \cdot 0 = 0$

- $x + 1 = 1$
- $x \cdot 1 = 1$
- $x + x = x$
- $x \cdot x = x$
- $x + x' = x$
- $x \cdot x' = 0$
- $x + y = y + x$
- $xy = yx$
- $x + (y + z) = (x + y) + z$
- $x (yz) = (xy) z$
- $x (y + z) = xy + xz$
- $x + yz = (x + y)(x + z)$
- $(x + y)' = x' y'$
- $(xy)' = x' + y'$
- $(x')' = x$

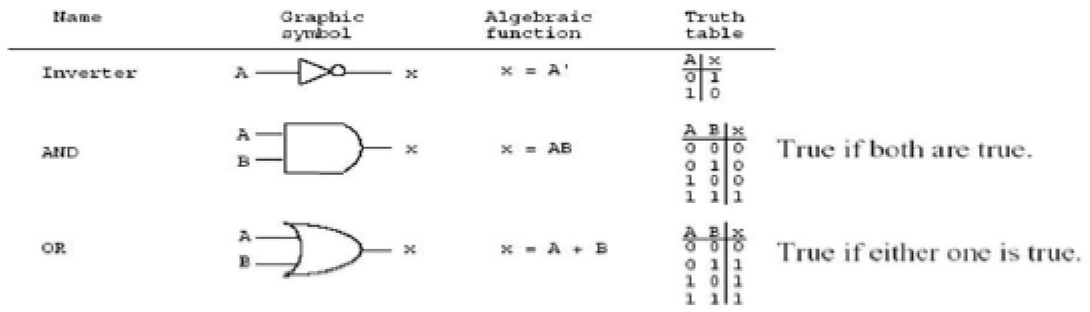
DeMorgan's Theorem
(a) : $(a + b)' = a'b'$
(b) : $(ab)' = a' + b'$
Generalized DeMorgan's Theorem
(a) : $(a + b + \dots z)' = a'b' \dots z'$
(b) : $(a.b \dots z)' = a' + b' + \dots z$,,

AXIOMATIC DEFINITION OF BOOLEAN ALGEBRA:
1. Closure
a. Closure with respect to (wrt) OR (+)
b. Closure with respect to AND (•)
2. Identity
a. Identity element wrt to OR : 0
b. Identity element wrt to AND : 1
3. Commutative Property

a. Commutative Property wrt to OR : $x + y = y + x$
b. Commutative Property wrt to AND : $x \cdot y = y \cdot x$
4. Distributive Property
a. $x \cdot (y + z) = (x\cdot y) + (x\cdot z)$
b. $x + (y\cdot z) = (x + y)(x + z)$
5. Existence of Complement
a. $x + x'' = 1$
b. $x \cdot x'' = 0$

LOGIC GATES

Formal logic: In formal logic, a statement (proposition) is a declarative sentence that is either true(1) or false (0). It is easier to communicate with computers using formal logic.
• Boolean variable: Takes only two values – either true (1) or false (0). They are used as basic units of formal logic.
• Boolean algebra: Deals with binary variables and logic operations operating on those variables.
• Logic diagram: Composed of graphic symbols for logic gates. A simple circuit sketch that represents inputs and outputs of Boolean functions.



Name	Graphic symbol	Algebraic function	Truth table	
Inverter		$x = A'$	A x: 0 1; 1 0	
AND		$x = AB$	A B x: 0 0 0; 0 1 0; 1 0 0; 1 1 1	True if both are true.
OR		$x = A + B$	A B x: 0 0 0; 0 1 1; 1 0 1; 1 1 1	True if either one is true.

INTEGRATED CIRCUIT

An integrated circuit or monolithic integrated circuit (also referred to as an IC, a chip, or a microchip) is a set of electronic circuits on one small plate ("chip") of semiconductor material, normally silicon. This can be made much smaller than a discrete circuit made from independent electronic components. ICs can be made very compact, having up to several billion transistors and other electronic components in an area the size of a human fingernail.
LEVEL OF INTEGRATION
1.SSI : Small Scale Integration
It has less than 100 components(about 10gates)
2.MSI: Medium Scale Integration
It contains less than 500 components or have more than 10 but less than 100 gates.
3.LSI: Large scale integration
Number of components is between 500 and 300000 or have more than 100gates.
4.VLSI:very large scale integration. process of creating an integrated circuit (IC)
by combining thousands of transistors into a single chip.

DIGITAL LOGIC FAMILIES

Transistor–transistor logic (**TTL**) is a class of digital circuits built from bipolar junction transistors (BJT) and resistors. It is called *transistor–transistor logic* because both the logic gating function (e.g., AND) and the amplifying function are performed by transistors (contrast with resistor–transistor logic (RTL) and diode–transistor logic (DTL).
Emitter coupled logic(ECL)
Emitter-coupled logic (ECL) is the fastest logic circuit family available for conventional logic-system design.4 High speed is achieved by operating all bipolar transistors out of saturation, thus avoiding storage-time delays,

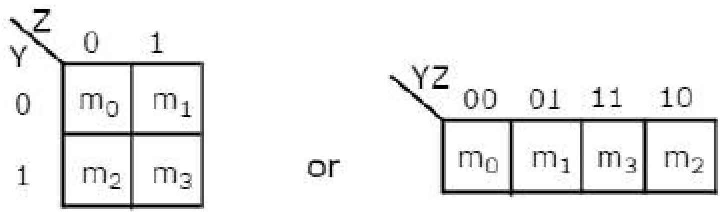
and by keeping the logic signal swings relatively small (about 0.8 V or less), thus reducing the time required to charge and discharge the various load and parasitic capacitances.
Complementary Metal oxide semiconductor (CMOS): Technology for constructing integrated circuits. CMOS technology is used in microprocessors, microcontrollers, static RAM, and other digital logic circuits. CMOS technology is also used for several analog circuits such as image sensors (CMOS sensor), data converters, and highly integrated transceivers for many types of communication.

INTRODUCTION
Minimization of switching functions is to obtain logic circuits with least circuit complexity. This goal is very difficult since how a minimal function relates to the implementation technology is
important. For example, If we are building a logic circuit that uses discrete logic made of small scale Integration ICs(SSIs) like 7400 series, in which basic building block are constructed and are available for use. The goal of minimization would be to reduce the number of ICs and not the logic gates. For example, If we require two 6 and gates and 5 Or gates,we would require 2 AND ICs(each has 4 AND gates) and one OR IC. (4 gates). On the other hand if the same logic could be implemented with only 10 nand gates, we require only 3 ICs. Similarly when we design logic on Programmable device, we may implement the design with certain number of gates and remaining gates may not be used. Whatever may be the criteria of minimization we would be guided by the following:
• Boolean algebra helps us simplify expressions and circuits
• Karnaugh Map: A graphical technique for simplifying a Boolean expression into either
form: minimal sum of products (MSP)
• minimal product of sums (MPS)
• Goal of the simplification.
• There are a minimal number of product/sum terms
• Each term has a minimal number of literals
• Circuit-wise, this leads to a minimal two-level implementation

K-map Simplification
• Imagine a two-variable sum of minterms
• x‟y‟ + x‟y
• Both of these minterms appear in the top row of a Karnaugh map, which means that they both contain the literal x‟

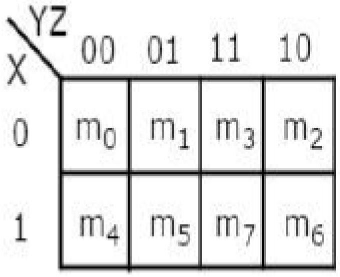
K-Maps for 2 to 4 Variables
K-Map method is most suitable for minimizing Boolean functions of 2 variables to 4variables. Now, let us discuss about the K-Maps for 2 to 4 variables one by one.

2 Variable K-Map
The number of cells in 2 variable K-map is four, since the number of variables is two. The following figure shows **2 variable K-Map**.



• There is only one possibility of grouping 4 adjacent min terms.
• The possible combinations of grouping 2 adjacent min terms are {(m_0, m_1), (m_2, m_3), (m_0, m_2) and (m_1, m_3)}.

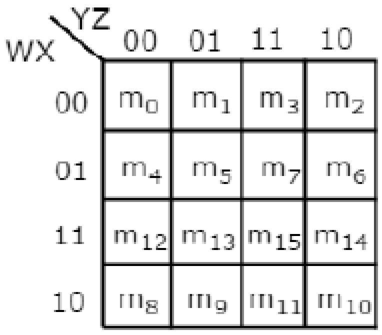
3 Variable K-Map
The number of cells in 3 variable K-map is eight, since the number of variables is three. The following figure shows **3 variable K-Map.**



X \ YZ	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

• There is only one possibility of grouping 8 adjacent min terms.
• The possible combinations of grouping 4 adjacent min terms are {(m_0, m_1, m_3, m_2), (m_4, m_5, m_7, m_6), (m_0, m_1, m_4, m_5), (m_1, m_3, m_5, m_7), (m_3, m_2, m_7, m_6) and (m_2, m_0, m_6, m_4)}.
• The possible combinations of grouping 2 adjacent min terms are {(m_0, m_1), (m_1, m_3), (m_3, m_2), (m_2, m_0), (m_4, m_5), (m_5, m_7), (m_7, m_6), (m_6, m_4), (m_0, m_4), (m_1, m_5), (m_3, m_7) and (m_2, m_6)}.
• If x=0, then 3 variable K-map becomes 2 variable K-map.

4 Variable K-Map
The number of cells in 4 variable K-map is sixteen, since the number of variables is four. The following figure shows **4 variable K-Map**.

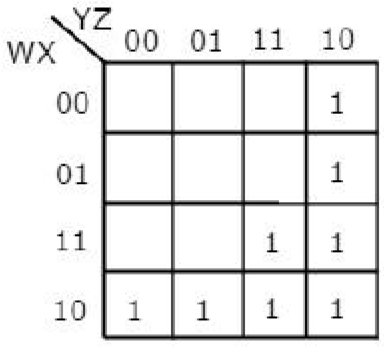


WX \ YZ	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

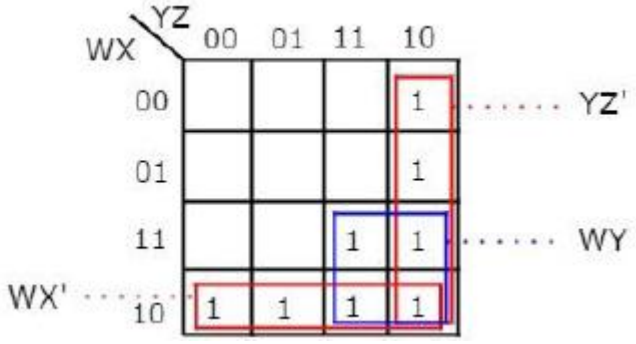
• There is only one possibility of grouping 16 adjacent min terms.
• Let R_1, R_2, R_3 and R_4 represents the min terms of first row, second row, third row and fourth row respectively. Similarly, C_1, C_2, C_3 and C_4 represents the min terms of first column, second column, third column and fourth column respectively. The possible combinations of grouping 8 adjacent min terms are {(R_1, R_2), (R_2, R_3), (R_3, R_4), (R_4, R_1), (C_1, C_2), (C_2, C_3), (C_3, C_4), (C_4, C_1)}.
• If w=0, then 4 variable K-map becomes 3 variable K-map.

Minimization of Boolean Functions using K-Maps
If we consider the combination of inputs for which the Boolean function is „1‟, then we will get the Boolean function, which is in **standard sum of products** form after simplifying the K-map.
Similarly, if we consider the combination of inputs for which the Boolean function is „0‟, then we will get the Boolean function, which is in **standard product of sums** form after simplifying the K-map.

Example
Let us **simplify** the following Boolean function, **f(W, X, Y, Z)= WX'Y' + WY + W'YZ'** using K-map.
The given Boolean function is in sum of products form. It is having 4 variables W, X, Y & Z. So, we require **4 variable K-map**. The **4 variable K-map** with ones corresponding to the given product terms is shown in the following figure.



Here, 1s are placed in the following cells of K-map.
• The cells, which are common to the intersection of Row 4 and columns 1 & 2 are corresponding to the product term, **WX'Y'**.
• The cells, which are common to the intersection of Rows 3 & 4 and columns 3 & 4 are corresponding to the product term, **WY**.
• The cells, which are common to the intersection of Rows 1 & 2 and column 4 are corresponding to the product term, **W'YZ'.** There are no possibilities of grouping either 16 adjacent ones or 8 adjacent ones. There are three possibilities of grouping 4 adjacent ones. After these three groupings, there is no single one left as ungrouped. So, we no need to check for grouping of 2 adjacent ones. The **4 variable K-map** with these three **groupings** is shown in the following figure.



Here, we got three prime implicants WX‟, WY & YZ‟
Therefore, the **simplified Boolean function** is
f= WX' + WY + YZ'