EEE 3101: Digital Logic and Circuits

Boolean Algebra

Course Teacher: Nafiz Ahmed Chisty

Associate Professor, Department of EEE & CoE
Head (UG), Department of EEE
Faculty of Engineering
Room# DNG03, Ground Floor, D Building
Email: chisty@aiub.edu
Website: http://engg.aiub.edu/
Website: www.nachisty.com

BOOLEAN OPERATIONS AND EXPRESSIONS

- Boolean algebra is the mathematics of digital systems. A basic knowledge of Boolean algebra is indispensable to the study and analysis of logic circuits. In the last chapter, Boolean operations and expressions in terms of their relationship to NOT, AND, OR, NAND, and NOR gates were introduced.

- Variable, complement, and literal are terms used in Boolean algebra.

- A variable is a symbol (usually an italic uppercase letter) used to represent a logical quantity. Any single variable can have a 1 or a 0 value.

- The complement is the inverse of a variable and is indicated by a bar over the variable (overbar). The complement of the variable A is read as "not A" or "A bar."

- A literal is a variable or the complement of a variable.

Boolean Algebra Operator Precedence

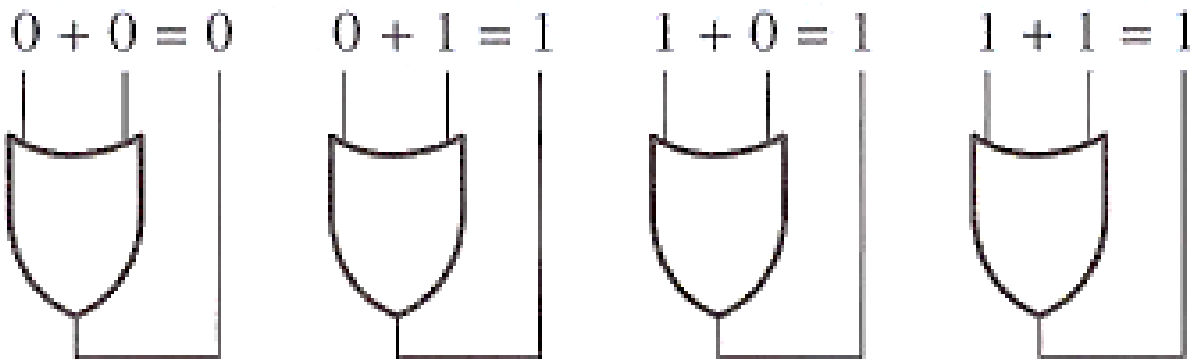
Precedence level	Operator
1	brackets ( )
2	Boolean complement NOT
3	Boolean product AND
4	Boolean sum OR

Note:

Brackets have the highest precedence, i.e., everything inside brackets is evaluated first.

Boolean Addition

Boolean addition is equivalent to the OR operation

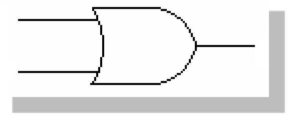


$0 + 0 = 0$

$0 + 1 = 1$

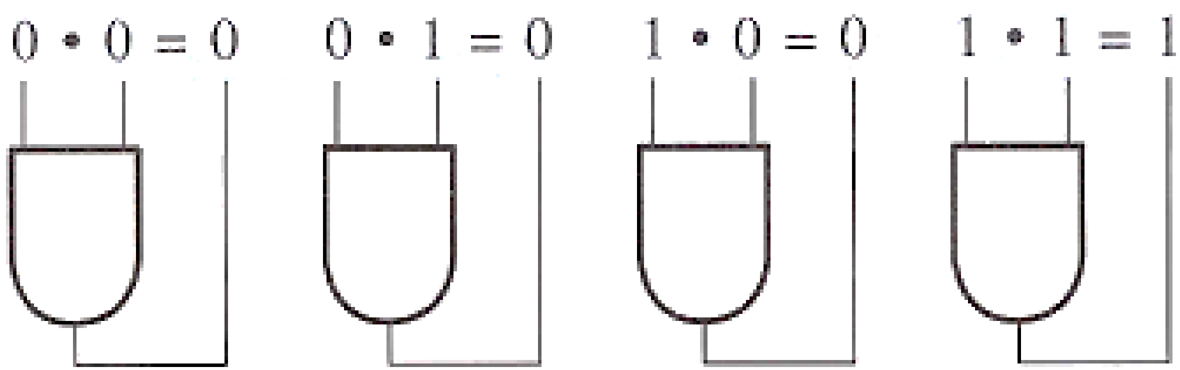
$1 + 0 = 1$

$1 + 1 = 1$



Boolean Multiplication

Boolean multiplication is equivalent to the AND operation

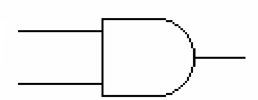


$0 * 0 = 0$

$0 * 1 = 0$

$1 * 0 = 0$

$1 * 1 = 1$



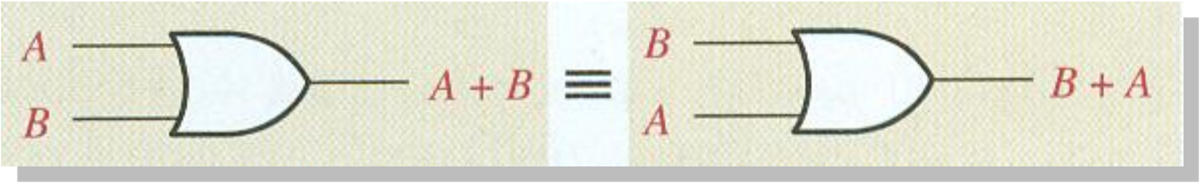
Laws and Rules of Boolean Algebra

Laws of Boolean Algebra

- Commutative Laws
- Associative Laws
- Distributive Law

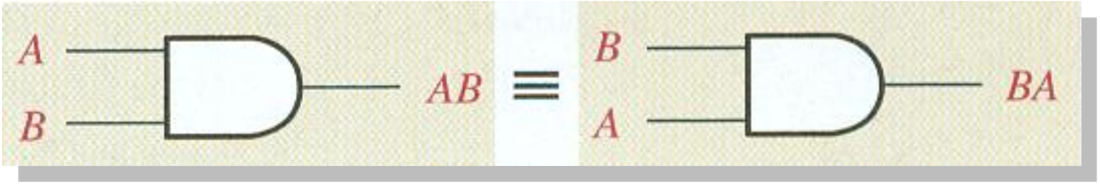
❑Commutative Law of Addition:

A + B = B + A



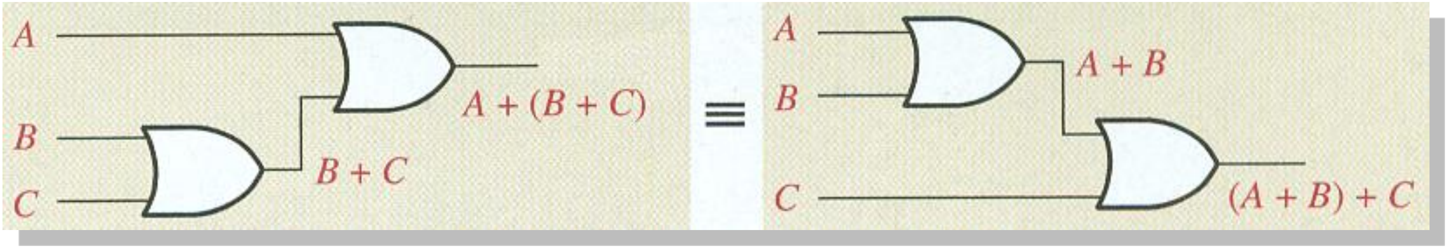
❑Commutative Law of Multiplication:

A * B = B * A



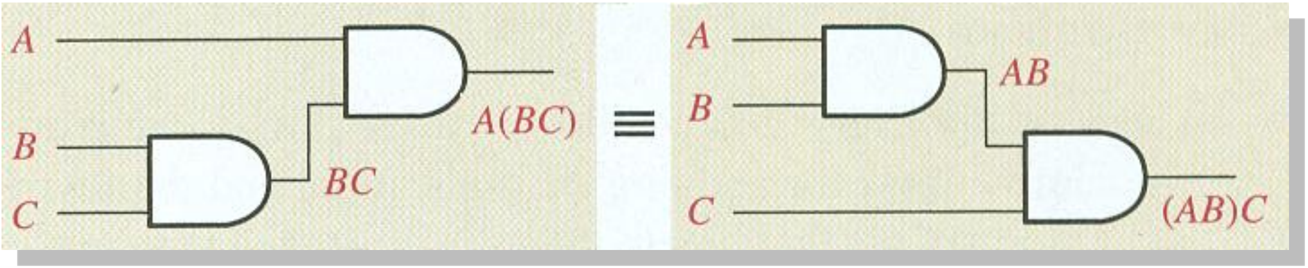
❑ Associative Law of Addition:

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



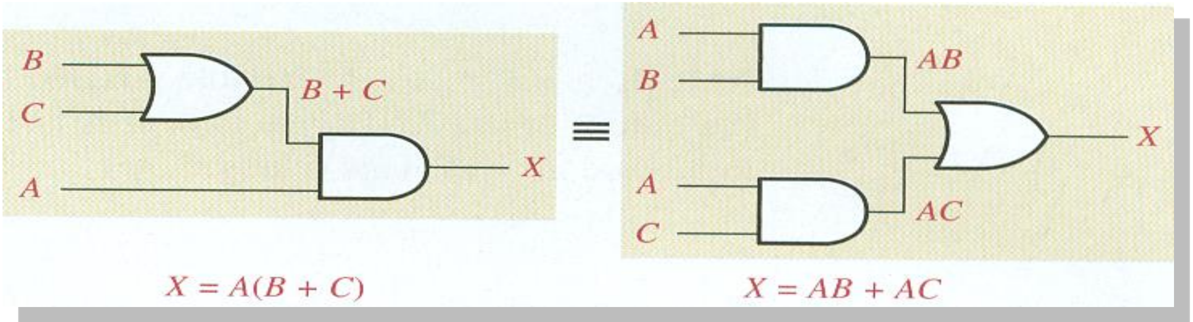
❑ Associative Law of Multiplication:

A * (B * C) = (A * B) * C



❑ Distributive Law:

A(B + C) = AB + AC



Rules of Boolean Algebra

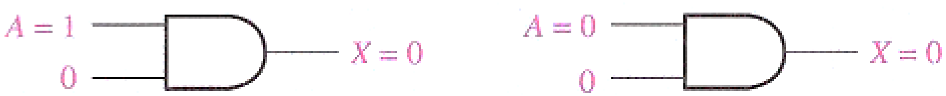
1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

A, B, or C can represent a single variable or a combination of variables.

Rule 1. $A + 0 = A$

Rule 2. $A + 1 = 1$

Rule 3. $A \cdot 0 = 0$



Rule 4. $A \cdot 1 = A$

Rule 5. $A + A = A$

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

OR Truth Table

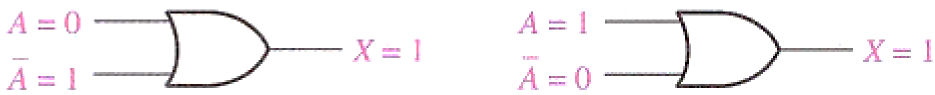
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

AND Truth Table

1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

Rule 6. $A + \overline{A} = 1$



Rule 7. $A \cdot A = A$

Rule 8. $A \cdot \overline{A} = 0$

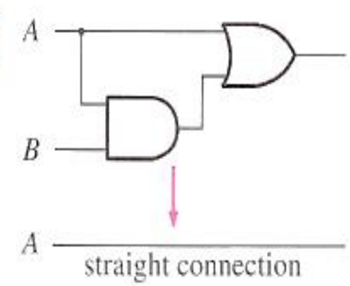
Rule 9. $\overline{\overline{A}} = A$

Rule 10. $A + AB = A$

$A + AB = A(1 + B)$		Factoring (distributive la
	$= A \cdot 1$	Rule 2: $(1 + B) = 1$
	$= A$	Rule 4: $A \cdot 1 = A$

A	B	AB	A + AB
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

equal

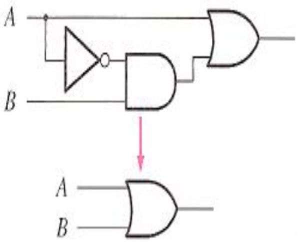


Rule 11. $A + \bar{A}B = A + B$

$A + \bar{A}B = (A + AB) + \bar{A}B$	Rule 10: $A = A + AB$
$= (AA + AB) + \bar{A}B$	Rule 7: $A = AA$
$= AA + AB + A\bar{A} + \bar{A}B$	Rule 8: adding $A\bar{A} = 0$
$= (A + \bar{A})(A + B)$	Factoring
$= 1 \cdot (A + B)$	Rule 6: $A + \bar{A} = 1$
$= A + B$	Rule 4: drop the 1

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

equal



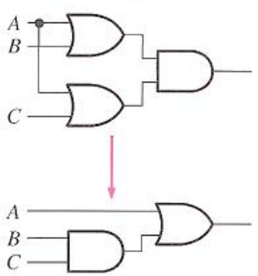
Rule 12. $(A + B)(A + C) = A + BC$

$(A + B)(A + C) = AA + AC + AB + BC$	Distributive law
$= A + AC + AB + BC$	Rule 7: $AA = A$
$= A(1 + C) + AB + BC$	Factoring (distributive law)
$= A \cdot 1 + AB + BC$	Rule 2: $1 + C = 1$
$= A(1 + B) + BC$	Factoring (distributive law)
$= A \cdot 1 + BC$	Rule 2: $1 + B = 1$
$= A + BC$	Rule 4: $A \cdot 1 = A$

1. $A + 0 = A$	7. $A \cdot A = A$
2. $A + 1 = 1$	8. $A \cdot \bar{A} = 0$
3. $A \cdot 0 = 0$	9. $\bar{\bar{A}} = A$
4. $A \cdot 1 = A$	10. $A + AB = A$
5. $A + A = A$	11. $A + \bar{A}B = A + B$
6. $A + \bar{A} = 1$	12. $(A + B)(A + C) = A + BC$

A	B	C	A + B	A + C	(A + B)(A + C)	BC	A + BC
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

equal



DeMorgan's Theorem

DeMorgan's first theorem is stated as follows:

The complement of a product of variables is equal to the sum of the complements of the variables,

$$\overline{XY} = \overline{X} + \overline{Y}$$

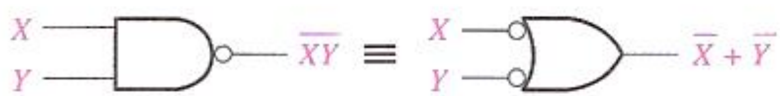
DeMorgan's second theorem is stated as follows:

The complement of a sum of variables is equal to the product of the complements of the variables.

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

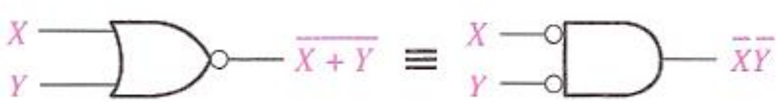
Remember:

"Break the bar, change the sign"



NAND Negative-OR

Inputs		Output	
X	Y	$\overline{XY}$	$\overline{X} + \overline{Y}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0



NOR Negative-AND

Inputs		Output	
X	Y	$\overline{X + Y}$	$\overline{X}\,\overline{Y}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Domain of a Boolean Expression: The domain of a general Boolean expression is the set of variables contained in the expression in either complemented or complemented form.

For example, the domain of the expressing A'B+ AB'C is the set of variables A, B, C and the domain of the expression ABC' + CD'E + B'CD' is the set of variables A, B, C, D, E.

Standard Forms of Boolean Expressions

All Boolean expressions, regardless of their form, can be converted into either of two standard forms: the sum-of-products form or the product-of-sums form. Standardization makes the evaluation, simplification, and implementation of Boolean expressions much more systematic and easier.

Sum of Products (SOP)= minterm

Product of Sums (POS)=maxterm

Minterms and Maxterms

- Each variable in a Boolean expression is a literal
- Boolean variables can appear in normal (x) or complement form (x')
- **Each AND combination of terms is a minterm**
- **Each OR combination of terms is a maxterm**

For example:
Minterms

x	y	z	Minterm	
0	0	0	x'y'z'	m_0
0	0	1	x'y'z	m_1
…				
1	0	0	xy'z'	m_4
…				
1	1	1	xyz	m_7

For example:
Maxterms

x	y	z	Maxterm	
0	0	0	x+y+z	M_0
0	0	1	x+y+z'	M_1
…				
1	0	0	x'+y+z	M_4
…				
1	1	1	x'+y'+z'	M_7

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

The sum-of-product (SOP) form

When two or more product terms are summed by Boolean addition. the resulting expression is a sum-of-products (SOP). Some examples are

$$AB + ABC$$

$$ABC + CDE + \bar{B}C\bar{D}$$

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

Also, an SOP expression can contain a single-variable term, as in $A + \bar{A}\bar{B}C + BC\bar{D}$.

an SOP expression can have the term $\bar{A}\bar{B}\bar{C}$ but not $\overline{ABC}$.

Conversion of a General Expression to SOP Form

$$A(B + CD) = AB + ACD$$

Convert each of the following Boolean expressions to SOP form:

(a) $AB + B(CD + EF)$ (b) $(A + B)(B + C + D)$ (c) $\overline{(\overline{A + B}) + C}$

Solution

(a) $AB + B(CD + EF) = AB + BCD + BEF$

(b) $(A + B)(B + C + D) = AB + AC + AD + BB + BC + BD$

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

The product of sum (POS) form

When two or more sum terms are multiplied, the resulting expression is a product-of-sums (POS). Some examples are

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

A POS expression can contain a single-variable term, as in $\overline{A}(A + \overline{B} + C)(\overline{B} + \overline{C} + D)$ have the term $\overline{A} + \overline{B} + \overline{C}$ but not $\overline{A + B + C}$.

The Standard SOP Form

A *standard SOP expression* is one in which *all* the variables in the domain appear in each product term in the expression. For example, $A\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + ABC\overline{D}$ is a standard SOP expression. Standard SOP expressions are important in constructing truth tables,

Converting Product Terms to Standard SOP

a nonstandard SOP expression is converted into standard form using Boolean algebra rule 6 ($A + \overline{A} = 1$) from Table 4–1: A variable added to its complement equals 1.

EXAMPLE

Convert the following Boolean expression into standard SOP form:

$$A\overline{B}C + \overline{A}\,\overline{B} + AB\overline{C}D$$

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

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

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

$$= \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$$

$$A\overline{B}C + \overline{A}\,\overline{B} + AB\overline{C}D = A\overline{B}CD + A\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + AB\overline{C}D$$

Binary Representation of a Standard Product Term

Remember, a product term is implemented with an AND gate whose output is 1 only if each of its inputs is 1. Inverters are used to produce the complements of the variables as required.

$$A\overline{B}C\overline{D} = 1\cdot\overline{0}\cdot1\cdot\overline{0} = 1\cdot1\cdot1\cdot1 = 1$$

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

The Standard POS Form

A standard POS expression is one in which *all* the variables in the domain appear in each sum term in the expression. For example,

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

Converting a Sum Term to Standard POS

: Add to each nonstandard term.

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

EXAMPLE

Convert the following Boolean expression into standard POS form:

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

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

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

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

Binary Representation of a Standard Sum Term

is implemented with an OR gate whose output is 0 only if each of its inputs is 0. Inverters are used to produce the complements of the variables as required.

$$A + \overline{B} + C + \overline{D} = 0 + \overline{1} + 0 + \overline{1} = 0 + 0 + 0 + 0 = 0$$

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

Converting Standard SOP to Standard POS

EXAMPLE

Convert the following SOP expression to an equivalent POS expression:

$$\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + ABC$$

$$\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + ABC$$

The evaluation is as follows:

$$000 + 010 + 011 + 101 + 111$$

Since there are three variables in the domain of this expression, there are a total of eight (2^3) possible combinations. The SOP expression contains five of these combinations, so the POS must contain the other three which are 001, 100, and 110

Remember, these are the binary values that make the sum term 0. The equivalent POS expression is

$$(A + B + \bar{C})(\bar{A} + B + C)(\bar{A} + \bar{B} + C)$$

BOOLEAN EXPRESSIONS AND TRUTH TABLES

All standard Boolean expressions can be easily converted into truth table format using binary values for each term in the expression. The truth table is a common way of presenting, in a concise format, the logical operation of a circuit. Also. standard SOP or POS expressions can be determined from a truth table. You will find truth tables in data sheets and other literature related to the operation of digital circuits.

Converting SOP Expressions to Truth Table Format

EXAMPLE

Develop a truth table for the standard SOP expression $\overline{A}\,\overline{B}C + A\overline{B}\,\overline{C} + ABC$.

INPUTS			OUTPUT	
A	B	C	X	PRODUCT TERM
0	0	0	0	
0	0	1	1	$\overline{A}\,\overline{B}C$
0	1	0	0	
0	1	1	0	
1	0	0	1	$A\overline{B}\,\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	1	ABC

Converting POS Expressions to Truth Table Format

EXAMPLE

Determine the truth table for the following standard POS expression:

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

INPUTS			OUTPUT	
A	B	C	X	SUM TERM
0	0	0	0	$(A + B + C)$
0	0	1	1	
0	1	0	0	$(A + \overline{B} + C)$
0	1	1	0	$(A + \overline{B} + \overline{C})$
1	0	0	1	
1	0	1	0	$(\overline{A} + B + \overline{C})$
1	1	0	0	$(\overline{A} + \overline{B} + C)$
1	1	1	1	

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

Determining Standard Expressions from a Truth Table

EXAMPLE

From the truth table in Table 4–8, determine the standard SOP expression and the equivalent standard POS expression.

INPUTS			OUTPUT
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

$011 \longrightarrow \overline{A}BC$

$100 \longrightarrow A\overline{B}\,\overline{C}$

$110 \longrightarrow AB\overline{C}$

$111 \longrightarrow ABC$

$000 \longrightarrow A + B + C$

$001 \longrightarrow A + B + \overline{C}$

$010 \longrightarrow A + \overline{B} + C$

$101 \longrightarrow \overline{A} + B + \overline{C}$

SOP expression for the output X is

$$X = \overline{A}BC + A\overline{B}\,\overline{C} + AB\overline{C} + ABC$$

The resulting standard POS expression for the output X is

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

The Karnaugh Map

A Karnaugh map is similar to a truth table

A Karnaugh map provides a systematic method for simplifying Boolean expressions and, if properly used, will produce the simplest SOP or POS expression possible, known as the minimum expression.

The effectiveness of algebraic simplification depends on your familiarity with all the laws, rules, and theorems of Boolean algebra and on your ability to apply them. The Karnaugh map, on the other hand, provides a "cookbook" method for simplification.

The 3-Variable Karnaugh Map

The 3-variable Karnaugh map is an array of eight cells

AB \ C	0	1
00	000	001
01	010	011
11	110	111
10	100	101

AB \ C	0	1
00	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$
01	$\bar{A}B\bar{C}$	$\bar{A}BC$
11	$AB\bar{C}$	ABC
10	$A\bar{B}\bar{C}$	$A\bar{B}C$

The 4-Variable Karnaugh Map

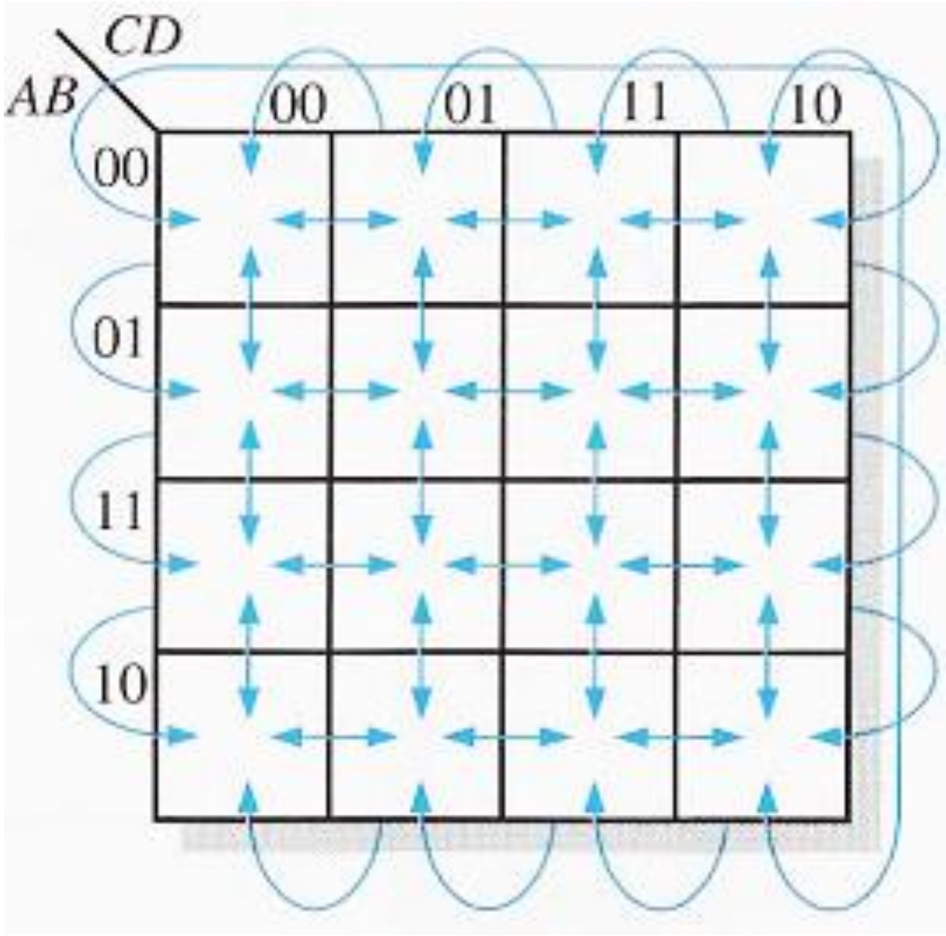
The 4-variable Karnaugh map is an array of sixteen cells

AB \ CD	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

AB \ CD	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}C\bar{D}$
01	$\bar{A}B\bar{C}\bar{D}$	$\bar{A}B\bar{C}D$	$\bar{A}BCD$	$\bar{A}BC\bar{D}$
11	$AB\bar{C}\bar{D}$	$AB\bar{C}D$	$ABCD$	$ABC\bar{D}$
10	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}C\bar{D}$

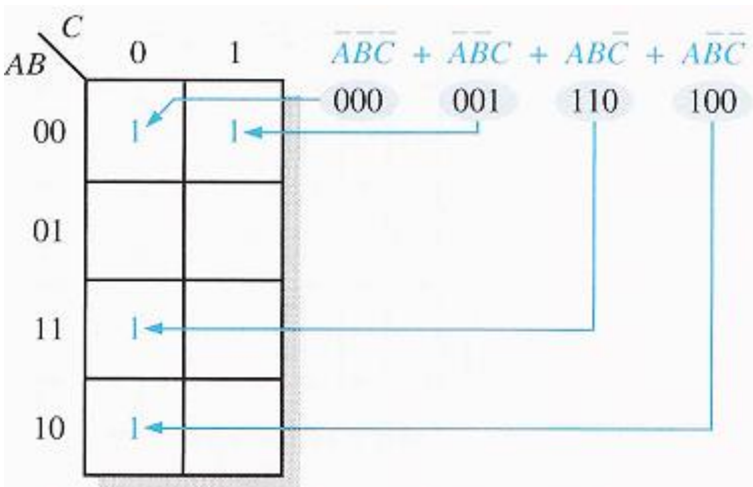
Cell Adjacency

Cells that differ by only one variable are adjacent.



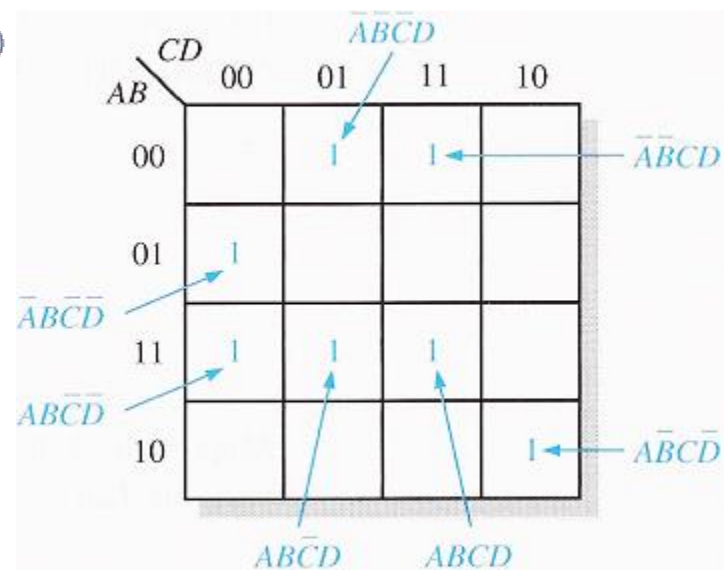
KARNAUGH MAP SOP MINIMIZATION

Mapping a Standard SOP Expression



$$\bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + AB\bar{C}D + ABCD + AB\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + A\bar{B}C\bar{D}$$

0011 0100 1101 1111 1100 0001 10



Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

Mapping a Nonstandard SOP Expression

A Boolean expression must first be in standard form before you use a Karnaugh map.

Numerical Expansion of a Nonstandard Product Term

EXAMPLE

Map the following SOP expression on a Karnaugh map: $\overline{A} + A\overline{B} + AB\overline{C}$.

$\overline{A}$	$+ A\overline{B}$	$+ AB\overline{C}$
000	100	110
001	101	
010		
011		

AB \ C	0	1
00	1	1
01	1	1
11	1	
10	1	1

$$\overline{B}\,\overline{C} + A\overline{B} + AB\overline{C} + A\overline{B}C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + A\overline{B}CD$$

$\overline{B}\,\overline{C}$	$A\overline{B}$	$+ AB\overline{C}$	$+ A\overline{B}C\overline{D}$	$+ \overline{A}\,\overline{B}\,\overline{C}D$	$+ A\overline{B}CD$
0000	1000	1100	1010	0001	1011
0001	1001	1101			
1000	1010				
1001	1011				

AB \ CD	00	01	11	10
00	1	1		
01				
11	1	1		
10	1	1	1	1

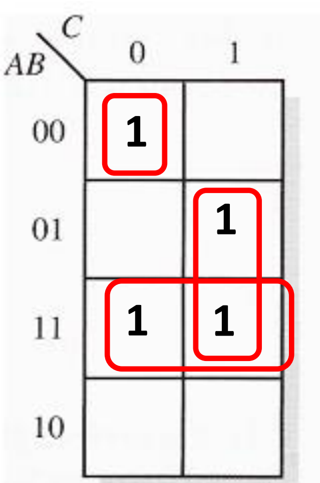
Karnaugh Map Simplification of SOP Expressions

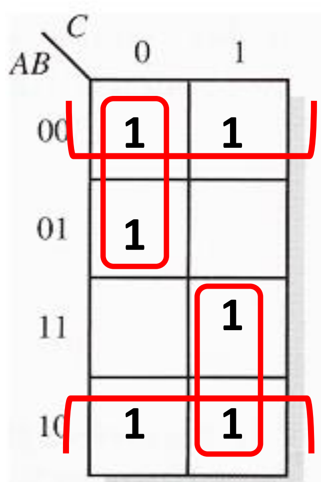
a minimum SOP expression is obtained by ***grouping* the 1s**

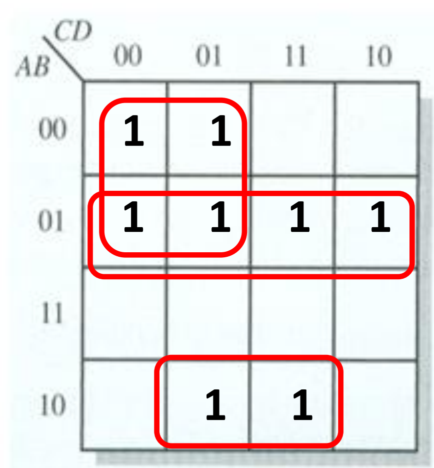
<u>**Grouping the 1s:**</u> You can group 1s on the Karnaugh map according to the following rules by enclosing those adjacent cells containing Is.

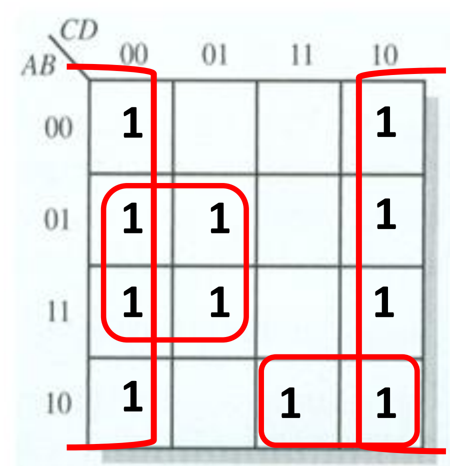
The goal is to maximize the size of the groups and to minimize the number of groups.

1. A group must contain either 1, 2, 4, 8, or 16 cells, which are all powers of two. In the case of a 3-variable map, $2^3 = 8$ cells is the maximum group.
2. Each cell in a group must be adjacent to one or more cells in that same group, but all cells in the group do not have to be adjacent to each other.
3. Always include the largest possible number of 1s in a group in accordance with rule 1.
4. Each 1 on the map must be included in at least one group. The 1s already in a group can be included in another group as long as the overlapping groups include non-common 1s.









Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

Na

Determining the Minimum SOP Expression from the Map: When all the 1s representing the standard product terms in an expression are properly mapped and grouped, the process of determining the resulting minimum SOP expression begins. The following rules are applied to find the minimum product terms and the minimum SOP expression:

1. Group the cells that have 1s. Each group of cells containing 1s creates one product term composed of all variables that occur in only one form (either un-complemented or complemented) within the group. Variables that occur both un-complemented and complemented within the group are eliminated. These are called contradictory variables.
2. Determine the minimum product term for each group.

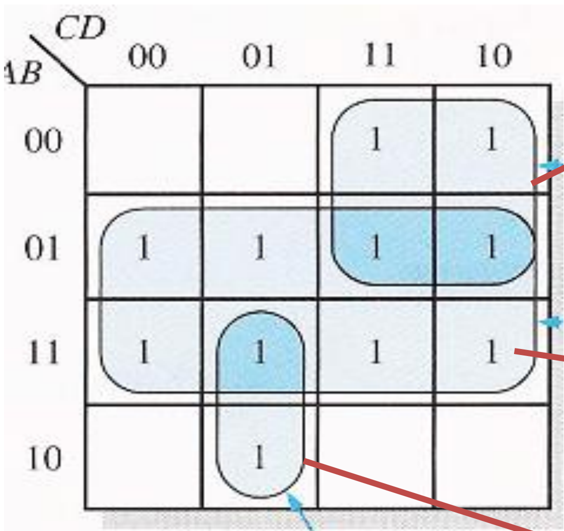
   a. For a 3-variable map:

   (1) A 1-cell group yields a 3-variable product term
   (2) A 2-cell group yields a 2-variable product term
   (3) A 4-cell group yields a 1-variable term
   (4) An 8-cell group yields a value of 1 for the expression

   b. For a 4-variable map:

   (1) A 1-cell group yields a 4-variable product term
   (2) A 2-cell group yields a 3-variable product term
   (3) A 4-cell group yields a 2-variable product term
   (4) An 8-cell group yields a 1-variable term
   (5) A 16-cell group yields a value of 1 for the expression
3. When all the minimum product terms are derived from the Karnaugh map, they are summed to form the minimum SOP expression.

EXAMPLE



ABCD
0011
0010
0111
0110

A' C

ABCD
1101
1001

A C'D

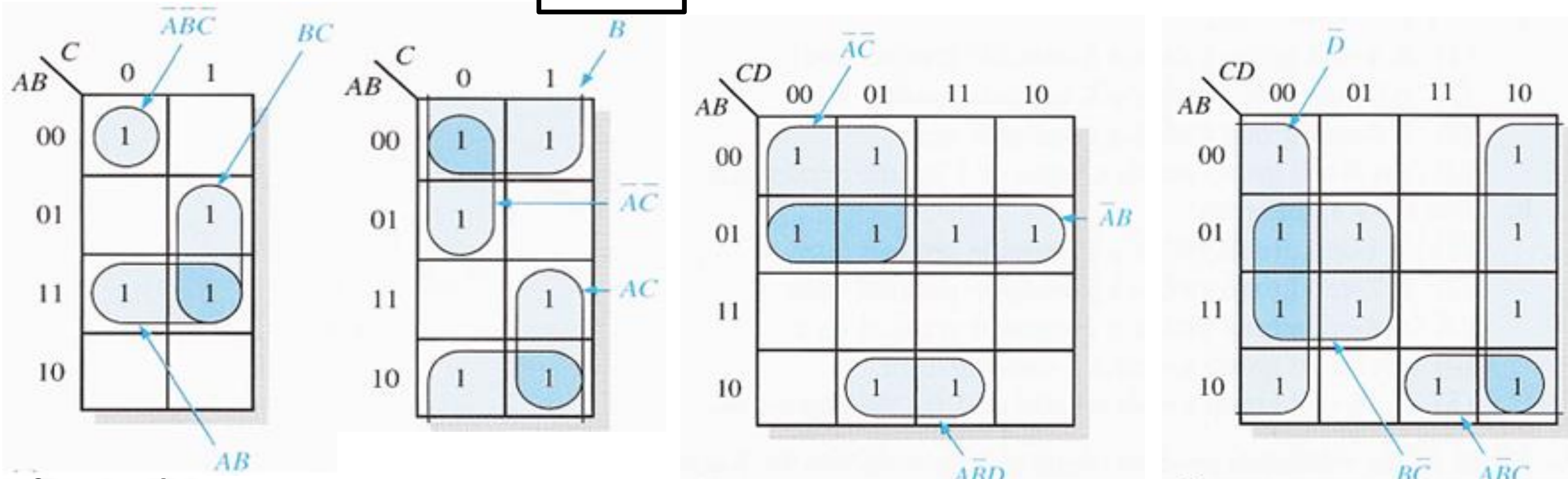
ABCD
0100
0101
0111
0110
1100
1101
1111
1110

B

the sum of these product terms:

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

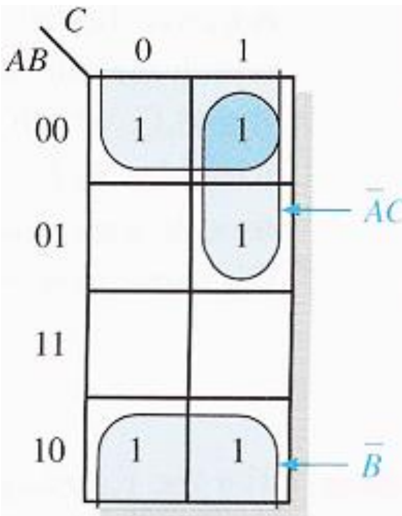
EXAMPLE



EXAMPLE

Use a Karnaugh map to minimize the following standard SOP expression

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



The binary values of the expression are

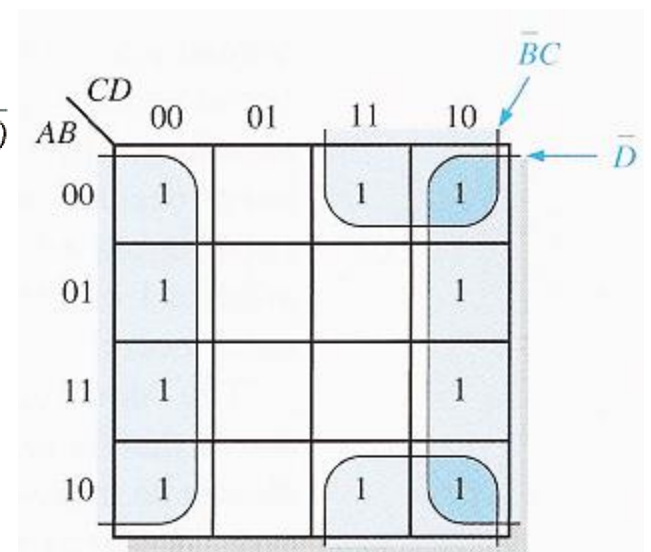
$$101 + 011 + 011 + 000 + 100$$

The resulting minimum SOP expression is

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

EXAMPLE

$$\overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}CD + A\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + \overline{A}BC\overline{D} + ABC\overline{D} + A\overline{B}C\overline{D}$$



The resulting minimum SOP expression is

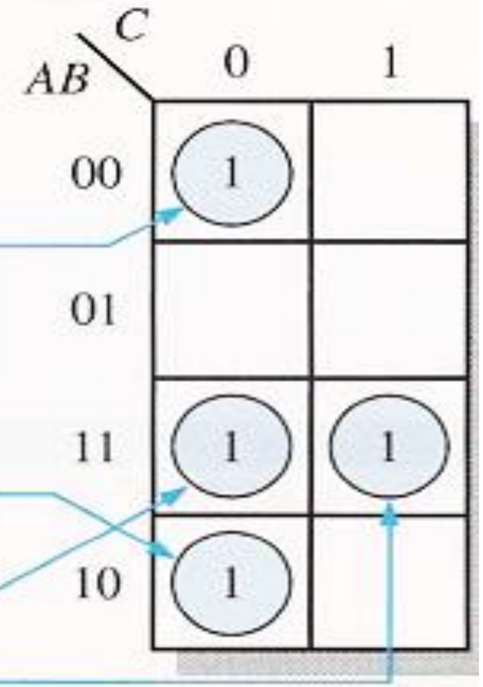
$$\overline{D} + \overline{B}C$$

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

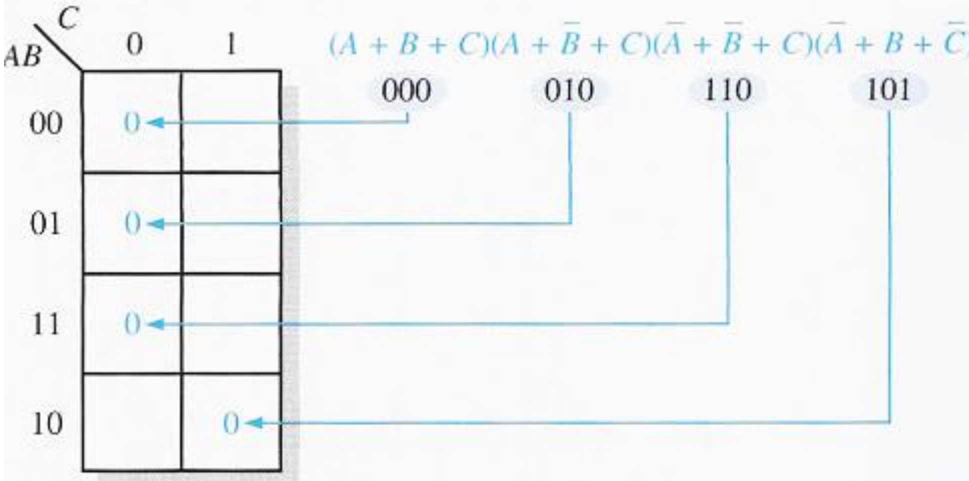
Mapping Directly from a Truth Table

$X = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C} + ABC$

Inputs			Output
A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1



KARNAUGH MAP POS MINIMIZATION



Karnaugh Map Simplification of POS Expressions

The process for minimizing a POS expression is basically the same as for an SOP

EXAMPLE

Use a Karnaugh map to minimize the following standard POS expression:

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

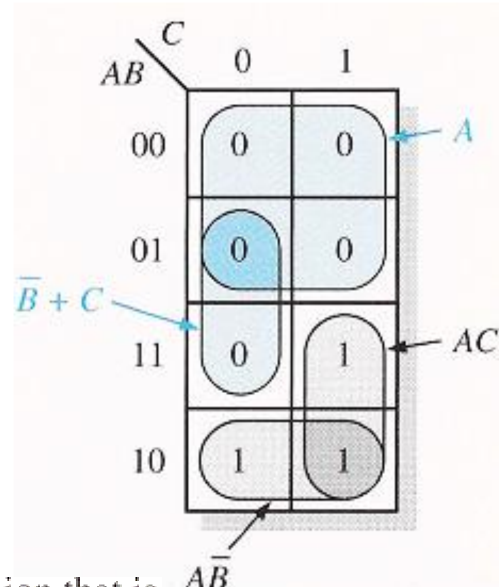
Also, derive the equivalent SOP expression.

Solution The combinations of binary values of the expression are

$$(0 + 0 + 0)\ (0 + 0 + 1)\ (0 + 1 + 0)\ (0 + 1 + 1)\ (1 + 1 + 0)$$

resulting minimum POS expression is

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

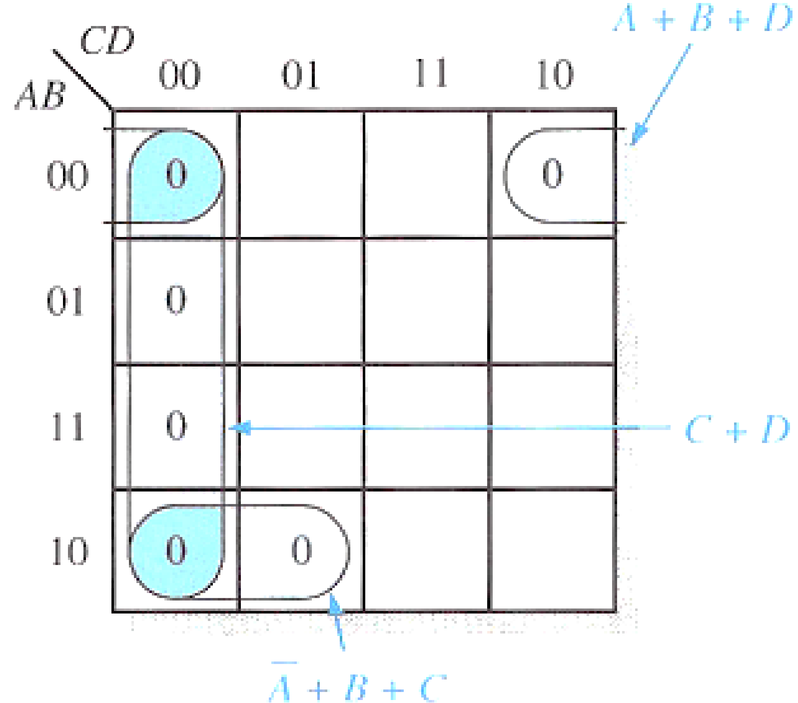


Grouping the 1s as shown by the gray areas yields an SOP expression that is equivalent to grouping the 0s.

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

Nafiz A. Chisty| Head, Dept of EEE; Associate Professor, Dept. of EEE & CoE, FE

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



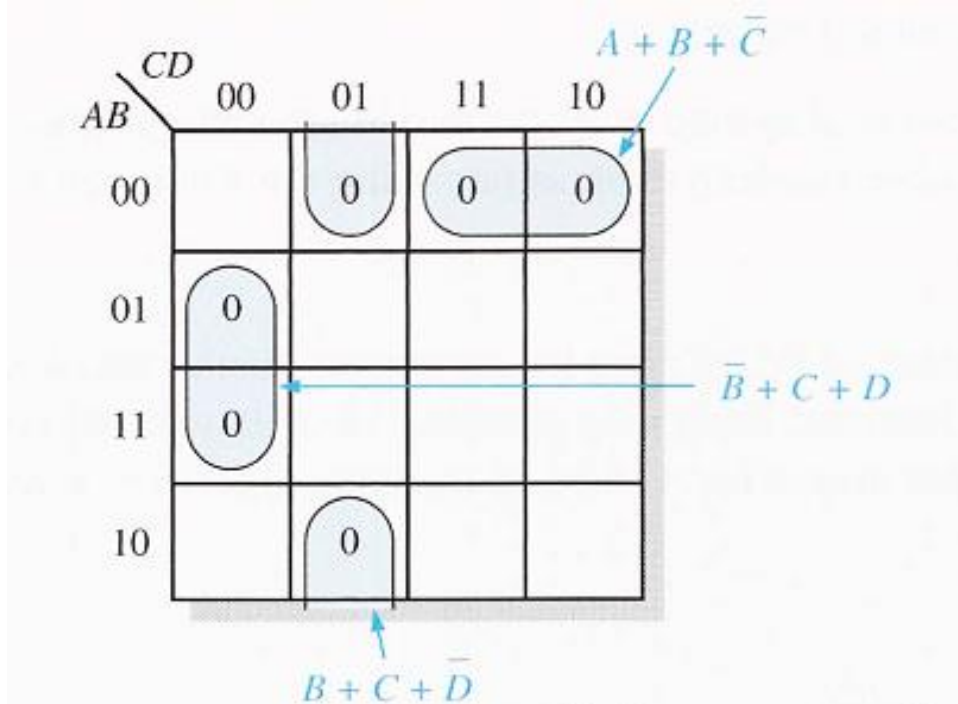
$$(C + D)\,(A + B + D)\,(\overline{A} + B + C)$$

Converting Between POS and SOP Using the Karnaugh Map

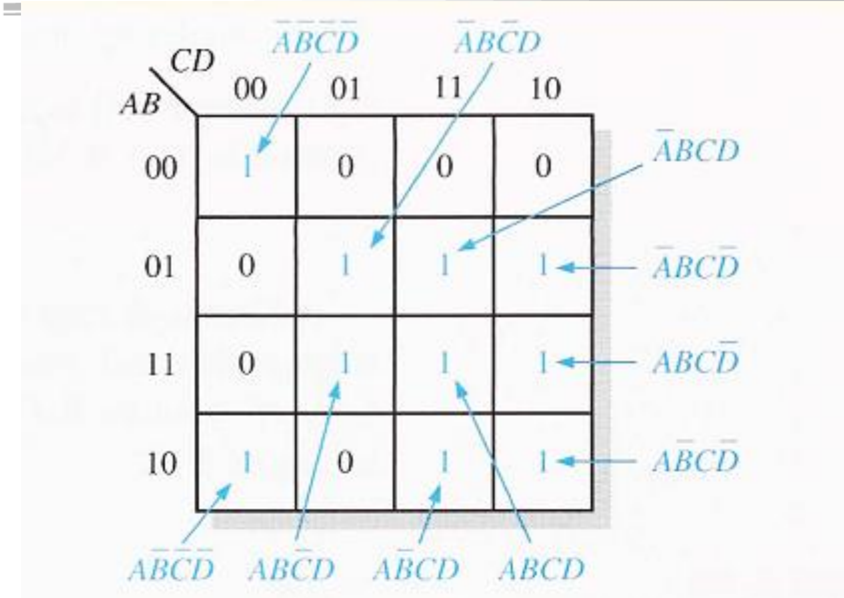
EXAMPLE

Using a Karnaugh map, convert the following standard POS expression into a minimum POS expression, a standard SOP expression, and a minimum SOP expression.

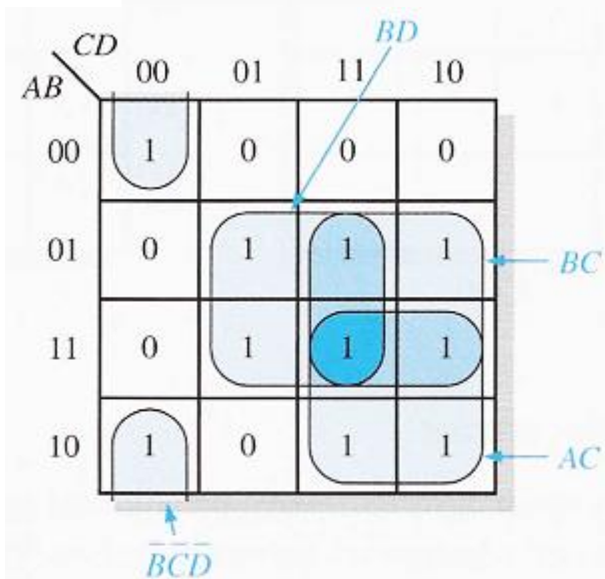
$$(\overline{A} + \overline{B} + C + D)(A + \overline{B} + C + D)(A + B + C + \overline{D})$$
$$(A + B + \overline{C} + \overline{D})(\overline{A} + B + C + \overline{D})(A + B + \overline{C} + D)$$



(a) Minimum POS: $(A + B + C)(\overline{B} + C + D)(B + C + \overline{D})$



(b) Standard SOP:
$\overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}B\overline{C}D + \overline{A}BCD + \overline{A}BC\overline{D} + ABC\overline{D} + A\overline{B}C\overline{D} +$
$A\overline{B}\overline{C}\overline{D} + AB\overline{C}D + A\overline{B}CD + ABCD$



(c) Minimum SOP: $AC + BC + BD + \overline{B}\overline{C}\overline{D}$

Determine the simplifies POS for the following expression

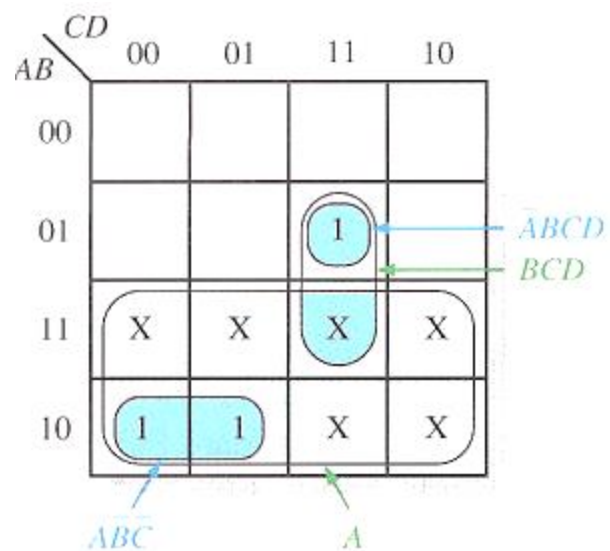
F(A,B,C,D)= ε(0,1,3,5)

"Don't Care" Conditions

Sometimes a situation arises in which some input variable combinations are not allowed. For example, in BCD code there are six invalid combinations: 1010, 1011, 1100, 1101, 1110, and 1111. Since these un-allowed states will never occur in an application involving the BCD code, they can be treated as "don't care" terms with respect to their effect on the output. That is, for these "don't care" terms either a 1 or a 0 may be assigned to the output: it really does not matter since they will never occur.

The "don't care" terms can be used to advantage on the Karnaugh map. Figure below shows that for each "don't care" term, an X is placed in the cell. When grouping the 1s, the Xs can be treated as 1s to make a larger grouping or as 0s if they cannot be used to advantage. The larger a group, the simpler the resulting term will be.

Inputs A B C D	Output Y
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 0 1	0
0 1 1 0	0
0 1 1 1	1
1 0 0 0	1
1 0 0 1	1
1 0 1 0	X
1 0 1 1	X
1 1 0 0	X
1 1 0 1	X
1 1 1 0	X
1 1 1 1	X

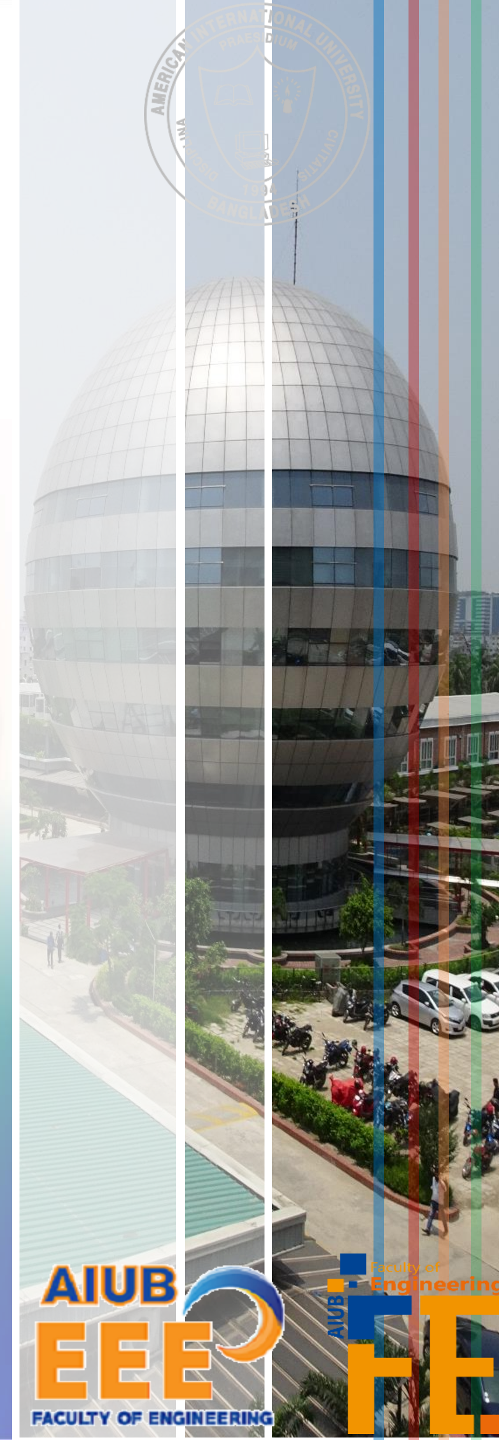


(b) Without "don't cares" $Y = A\bar{B}\bar{C} + \bar{A}BCD$
With "don't cares" $Y = A + BCD$

Reference:

[1] Thomas L. Floyd, "Digital Fundamentals" 11th edition, Prentice Hall.
[2] M. Morris Mano, "Digital Logic & Computer Design" Prentice Hall.

Thanks



	A	B	C	Y
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

	0	1
00	000 0	001 1
01	010 2	011 3
11	110 6	111 7
10	100 4	101 5

	A	B	C	D	Y
0	0	0	0	0	
1	0	0	0	1	
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
10	1	0	1	0	
11	1	0	1	1	
12	1	1	0	0	
13	1	1	0	1	
14	1	1	1	0	
15	1	1	1	1	

	00	01	11	10
00	0000 0	0001 1	0011 3	0010 2
01	0100 4	0101 5	0111 7	0110 6
11	1100 12	1101 13	1111 15	1110 14
10	1000 8	1001 9	1011 11	1010 10