Chapter No:4
Boolean Algebra and logic Simplification

CHAPTER OUTLINE

- Boolean Operations and Expressions
- Laws and Rules of Boolean Algebra
- DeMorgan's Theorems
- Boolean Analysis of Logic Circuits
- Logic Simplification Using Boolean Algebra
- Standard Forms of Boolean Expressions
- Boolean Expressions and Truth Tables
- The Karnaugh Map
- Karnaugh Map SOP Minimization
- Karnaugh Map POS Minimization

LOGIC CIRCUITS OR LOGIC GATES :

The operation of different logic gates and complex circuits can be described and analyzed using Boolean Algebra

Boolean algebra:

is a mathematical tool that allows us to describe the relationship between a logic circuit's output and its inputs as an algebraic equation (Boolean Expression).

Other techniques used in the analysis , synthesis and documentation of logic system and circuits are :

- *Truth table*
- *Schematic symbol*

- *Boolean algebra is a mathematical tool*
- *Truth tables are data organizational tool*
- *Schematic symbol s are drawing tools*
- *Timing diagrams are graphing tools*
- *Language is universal description tool.*

Boolean Operations and Expressions

Boolean Addition

In Boolean algebra, a **variable** is a symbol used to represent an action, a condition, or data. A single variable can only have a value of 1 or 0.

The **complement** represents the inverse of a variable and is indicated with an overbar. Thus, the complement of A is $\overline{A}$.

A **literal** is a variable or its complement.

Addition is equivalent to the OR operation. The sum term is 1 if one or more if the literals are 1. The sum term is zero only if each literal is 0.

Example Determine the values of A, B, and C that make the sum term of the expression $\overline{A} + B + \overline{C} = 0$?

Solution Each literal must = 0; therefore $A = 1$, $B = 0$ and $C = 1$.

Boolean Multiplication

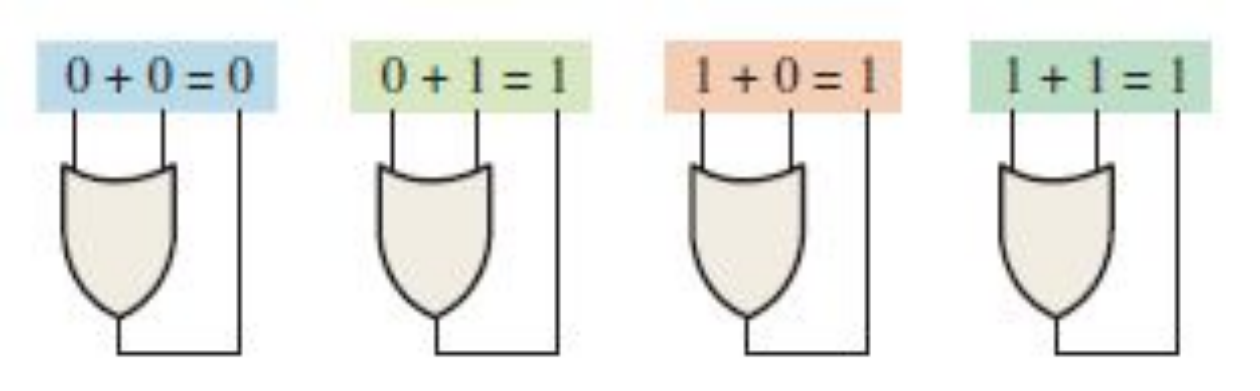
In Boolean algebra, multiplication is equivalent to the AND operation. The product of literals forms a product term. The product term will be 1 only if all of the literals are 1.

Example What are the values of the A, B and C if the product term of $A \cdot \overline{B} \cdot \overline{C} = 1$?

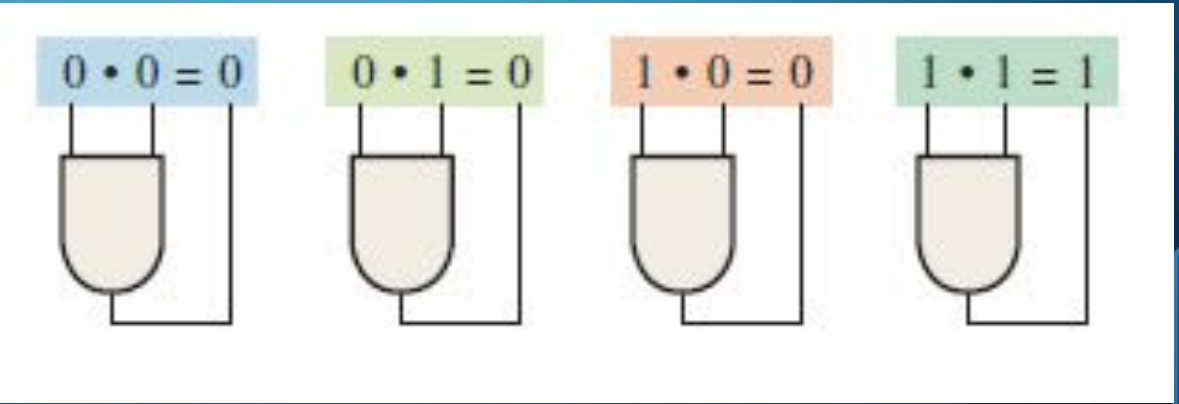
Solution Each literal must = 1; therefore $A = 1$, $B = 0$ and $C = 0$.

Boolean Operation and Expressions

Boolean Addition



Boolean Multiplication

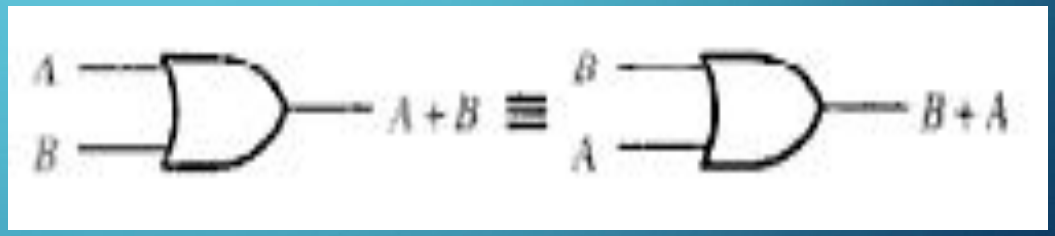


LAWS OF BOOLEAN ALGEBRA

Commutative law :

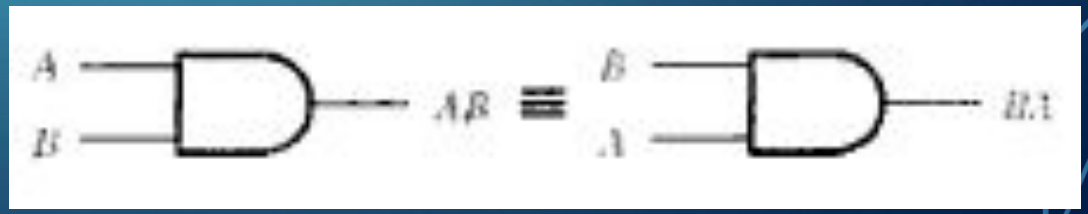
The commutative law of addition for two variables is written as

$$A + B = B + A$$



The commutative law of multiplication for two variables is written as

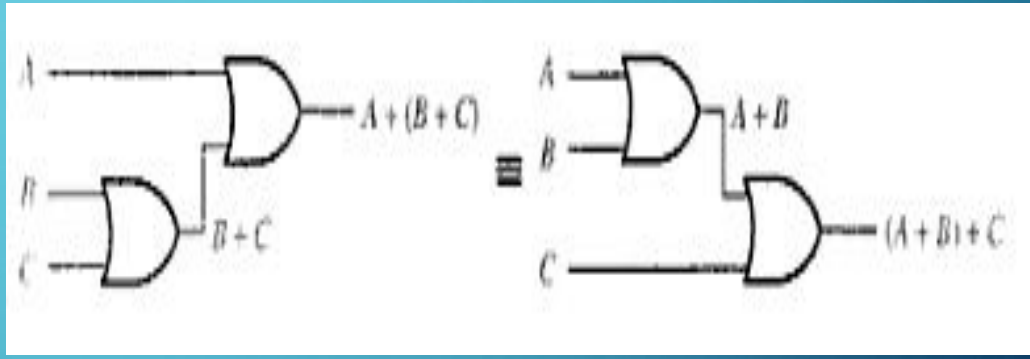
$$AB = BA$$



Associative law

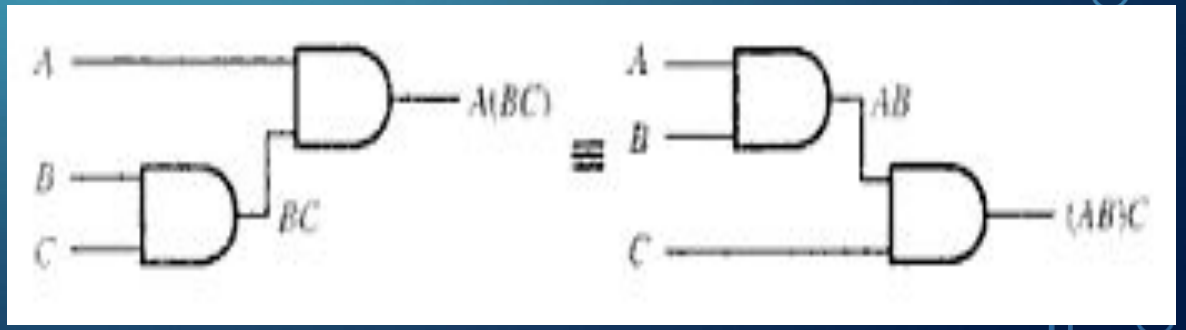
The associated law of addition for three variables is written as

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



The associated law of multiplication for three variables is written as

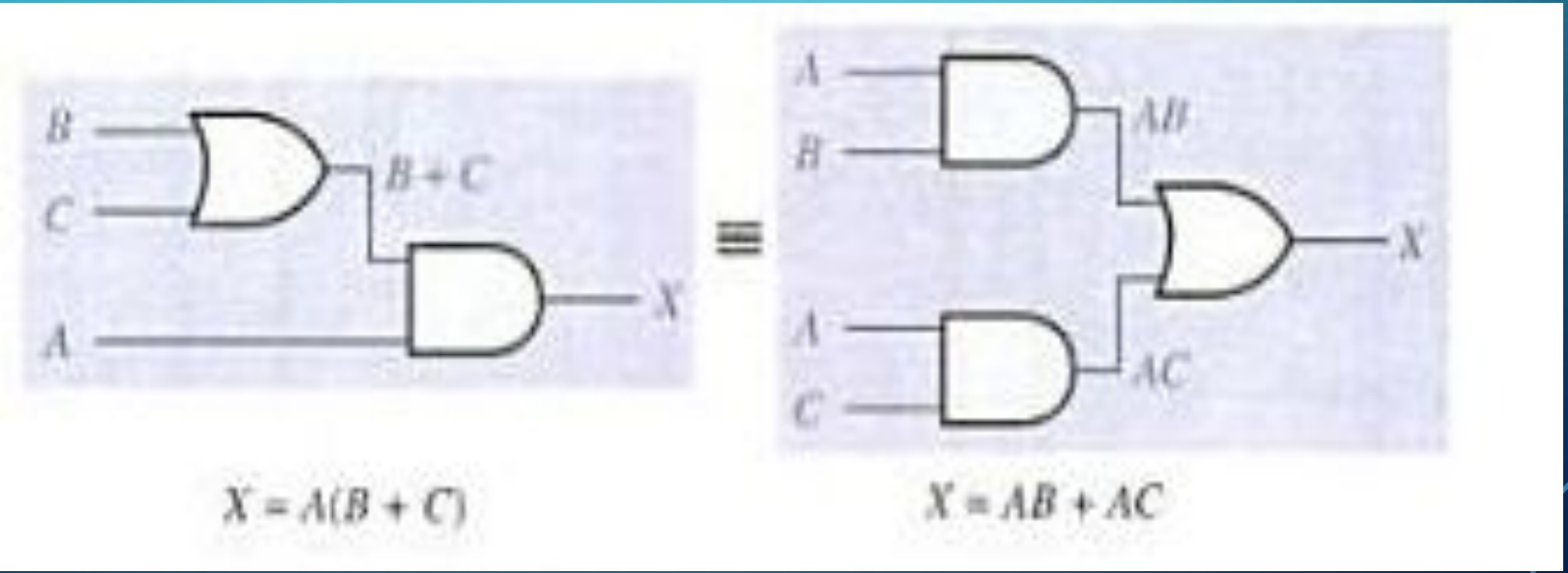
$$A(BC) = (AB)C$$



Distributive law

The distributive law is written for three variables as follows:

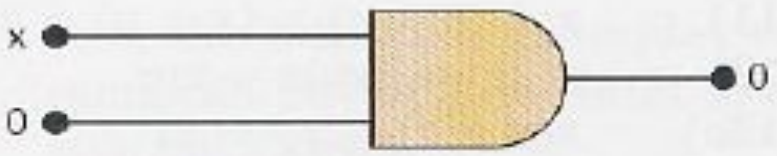
$$A(B + C) = AB + AC$$



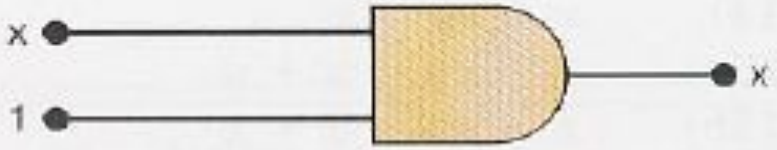
RULES OF BOOLEAN ALGEBRA

Boolean algebra can be used to help analyze a logic circuit and express its operation

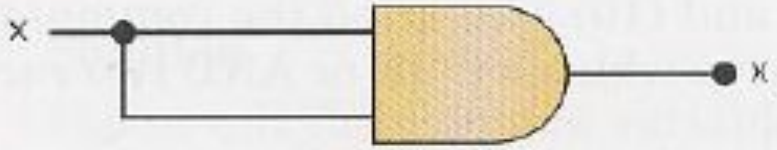
Single Variable theorem :



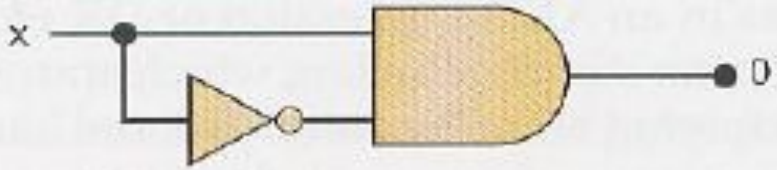
(1) $x \cdot 0 = 0$



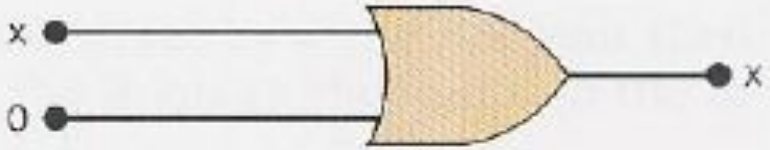
(2) $x \cdot 1 = x$



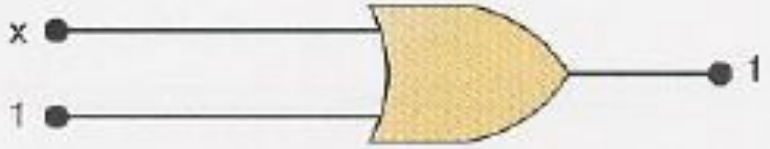
(3) $x \cdot x = x$



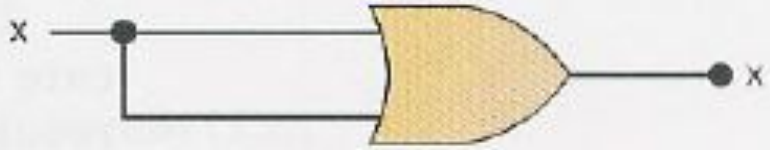
(4) $x \cdot \bar{x} = 0$



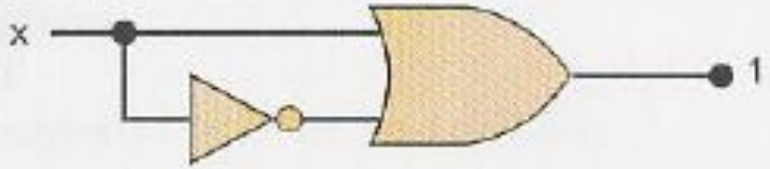
(5) $x + 0 = x$



(6) $x + 1 = 1$



(7) $x + x = x$



(8) $x + \bar{x} = 1$

MULTIVARIABLE THEOREM

It involve more than one variables.

Commutative Law:

$$(9) \qquad x + y = y + x$$
$$(10) \qquad x \cdot y = y \cdot x$$

Associative Law:

$$(11) \qquad x + (y + z) = (x + y) + z = x + y + z$$
$$(12) \qquad x(yz) = (xy)z = xyz$$

Distributive Law:

$$(13a) \qquad x(y + z) = xy + xz$$
$$(13b) \qquad (w + x)(y + z) = wy + xy + wz + xz$$

Others

$$(14) \qquad x + xy = x$$
$$(15a) \qquad x + \bar{x}y = x + y$$
$$(15b) \qquad \bar{x} + xy = \bar{x} + y$$

EXAMPLES:

Simplify the expression $y = A\overline{B}D + A\overline{B}\,\overline{D}$.

Factor out the common variables $A\overline{B}$ using theorem (13):

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

Using theorem (8), the term in parentheses is equivalent to 1. Thus,

$$\begin{aligned} y &= A\overline{B} \cdot 1 \\ &= A\overline{B} \qquad \text{[using theorem (2)]} \end{aligned}$$

Simplify $z = (\overline{A} + B)(A + B)$.

The expression can be expanded by multiplying out the terms [theorem (13)]:

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

Invoking theorem (4), the term $\overline{A} \cdot A = 0$. Also, $B \cdot B = B$ [theorem (3)]:

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

Factoring out the variable B [theorem (13)], we have

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

Finally, using theorems (2) and (6),

$$z = B$$

DEMORGAN 'S THEOREM

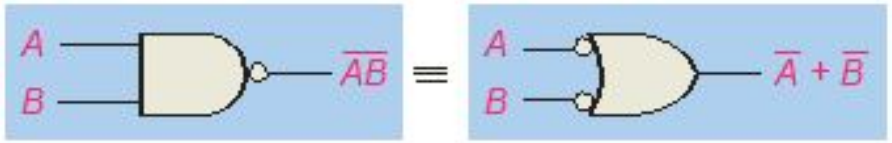
De Margan 's theorem are extremely useful in simplifying expressions in which a product or sum of variables is inverted.

DeMorgan's 1st Theorem

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

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

Applying DeMorgan's first theorem to gates:



NAND Negative-OR

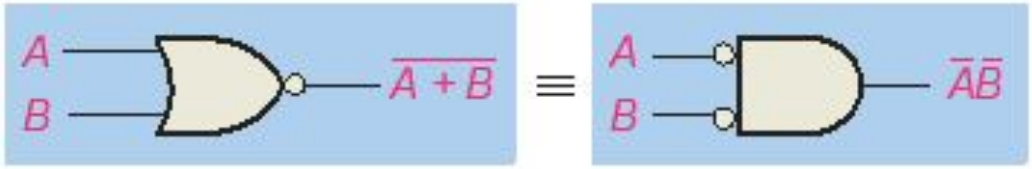
Inputs		Output	
A	B	$\overline{AB}$	$\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

DeMorgan's 2nd Theorem

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

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

Applying DeMorgan's second theorem to gates:



NOR Negative-AND

Inputs		Output	
A	B	$\overline{A + B}$	$\overline{A}\overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Example Apply DeMorgan's theorem to remove the overbar covering both terms from the expression $X = \overline{\overline{C} + D}$.

Solution To apply DeMorgan's theorem to the expression, you can break the overbar covering both terms and change the sign between the terms. This results in $X = \overline{\overline{C}} \cdot \overline{D}$. Deleting the double bar gives $X = C \cdot \overline{D}$.

DEMORGAN 'S THEOREM

Apply DeMargan theorem:

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

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

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

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

EXAMPLE

Simplify the expression $z = \overline{(\overline{A} + C) \cdot (B + \overline{D})}$ to one having only single variables inverted.

Using theorem (17), and treating $(\overline{A} + C)$ as x and $(B + \overline{D})$ as y, we have

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

$$\begin{aligned} z &= \overline{(\overline{A} + C)} + \overline{(B + \overline{D})} \\ &= (\overline{\overline{A}} \cdot \overline{C}) + \overline{B} \cdot \overline{\overline{D}} \end{aligned}$$

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

EXAMPLE

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

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

$$\omega = \overline{(A + BC) \cdot (D + EF)}$$
$$= \overline{(A + BC)} + \overline{(D + EF)}$$
$$= (\overline{A} \cdot \overline{BC}) + (\overline{D} \cdot \overline{EF})$$
$$= [\overline{A} \cdot (\overline{B} + \overline{C})] + [\overline{D} \cdot (\overline{E} + \overline{F})]$$
$$= \overline{A}\,\overline{B} + \overline{A}\,\overline{C} + \overline{D}\,\overline{E} + \overline{D}\,\overline{F}$$

DeMorgan's theorems are easily extended to more than two variables.

$$\overline{x + y + z} = \overline{x} \cdot \overline{y} \cdot \overline{z}$$
$$\overline{x \cdot y \cdot z} = \overline{x} + \overline{y} + \overline{z}$$

$$x = \overline{\overline{AB} \cdot \overline{CD} \cdot \overline{EF}}$$
$$= \overline{\overline{AB}} + \overline{\overline{CD}} + \overline{\overline{EF}}$$
$$= AB + CD + EF$$

Simplify $x = ACD + \overline{A}BCD.$

Solution

Factoring out the common variables CD, we have

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

Utilizing theorem (15a), we can replace $A + \overline{A}B$ by $A + B$, so

$$\begin{aligned} x &= CD(A + B) \\ &= ACD + BCD \end{aligned}$$

Using *Boolean algebra*, simplify the following *Boolean* expression:

A'.B'.C + (A + B + C') + A'.B'.C'.D

$$x = (M + N)(\overline{M} + P)(\overline{N} + \overline{P})$$

$$X = MP\overline{N} + N\overline{MP}$$

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

$$Z = B\overline{C}$$

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

Step 1: Identify the terms to which you can apply DeMorgan's theorems, and think of each term as a single variable. Let $\overline{A + B\overline{\overline{C}}} = X$ and $D(\overline{E + \overline{\overline{F}}}) = Y$.

Step 2: Since $\overline{X + Y} = \overline{X}\,\overline{Y}$,

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

Step 3: Use rule 9 ($\overline{\overline{A}} = A$) to cancel the double bars over the left term (this is not part of DeMorgan's theorem).

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

Step 4: Apply DeMorgan's theorem to the second term.

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

Step 5: Use rule 9 ($\overline{\overline{A}} = A$) to cancel the double bars over the $E + \overline{F}$ part of the term.

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

Apply DeMorgan's theorems to each of the following expressions:

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

(b) $\overline{ABC + DEF}$

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

(a) Let $A + B + C = X$ and $D = Y$. The expression $\overline{(A + B + C)D}$ is of the form $\overline{XY} = \overline{X} + \overline{Y}$ and can be rewritten as

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

Next, apply DeMorgan's theorem to the term $\overline{A + B + C}$.

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

(b) Let $ABC = X$ and $DEF = Y$. The expression $\overline{ABC + DEF}$ is of the form $\overline{X + Y} = \overline{X}\,\overline{Y}$ and can be rewritten as

$$\overline{ABC + DEF} = (\overline{ABC})(\overline{DEF})$$

Next, apply DeMorgan's theorem to each of the terms $\overline{ABC}$ and $\overline{DEF}$.

$$(\overline{ABC})(\overline{DEF}) = (\overline{A} + \overline{B} + \overline{C})(\overline{D} + \overline{E} + \overline{F})$$

(c) Let $A\overline{B} = X$, $\overline{C}D = Y$, and $EF = Z$. The expression $\overline{A\overline{B} + \overline{C}D + EF}$ is of the form $\overline{X + Y + Z} = \overline{X}\,\overline{Y}\,\overline{Z}$ and can be rewritten as

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

Next, apply DeMorgan's theorem to each of the terms $\overline{A\overline{B}}$, $\overline{\overline{C}D}$, and $\overline{EF}$.

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

Related Problem

Apply DeMorgan's theorems to the expression $\overline{\overline{ABC} + D + E}$.

Apply DeMorgan's theorems to each expression:

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

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

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

Solution

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

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

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

SIMPLIFICATION USING BOOLEAN ALGEBRA

Using Boolean algebra techniques, simplify this expression:

$$AB + A(B + C) + B(B + C)$$

The following is not necessarily the only approach.

Step 1: Apply the distributive law to the second and third terms in the expression, as follows:

$$AB + AB + AC + BB + BC$$

Step 2: Apply rule 7 ($BB = B$) to the fourth term.

$$AB + AB + AC + B + BC$$

Step 3: Apply rule 5 ($AB + AB = AB$) to the first two terms.

$$AB + AC + B + BC$$

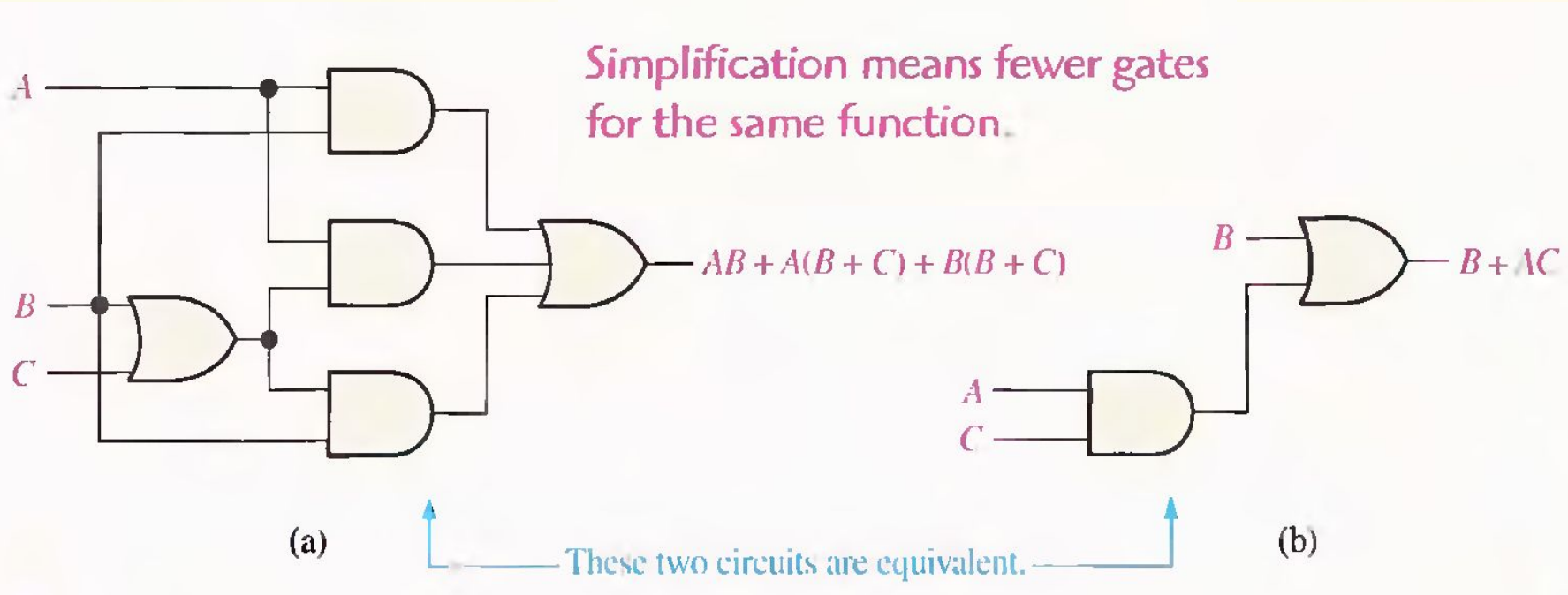
Step 4: Apply rule 10 ($B + BC = B$) to the last two terms.

$$AB + AC + B$$

Step 5: Apply rule 10 ($AB + B = B$) to the first and third terms.

$$B + AC$$

At this point the expression is simplified as much as possible. Once you gain experience in applying Boolean algebra, you can often combine many individual steps.



EXAMPLE

Simplify the following Boolean expression:

$$[A\overline{B}(C + BD) + \overline{A}\,\overline{B}]C$$

Step 1: Apply the distributive law to the terms within the brackets.

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

Step 2: Apply rule 8 ($\overline{B}B = 0$) to the second term within the parentheses.

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

Step 3: Apply rule 3 ($A \cdot 0 \cdot D = 0$) to the second term within the parentheses.

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

Step 4: Apply rule 1 (drop the 0) within the parentheses.

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

EXAMPLE

Step 5: Apply the distributive law.

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

Step 6: Apply rule 7 ($CC = C$) to the first term.

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

Step 7: Factor out $\overline{B}C$.

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

Step 8: Apply rule 6 ($A + \overline{A} = 1$).

$$\overline{B}C \cdot 1$$

Step 9: Apply rule 4 (drop the 1).

$$\overline{B}C$$

Example

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

$$A\overline{B}$$

Simplify the Boolean expression $[AB(C + \overline{BD}) + \overline{AB}]CD$.

$$CD$$

Simplify the following Boolean expression:

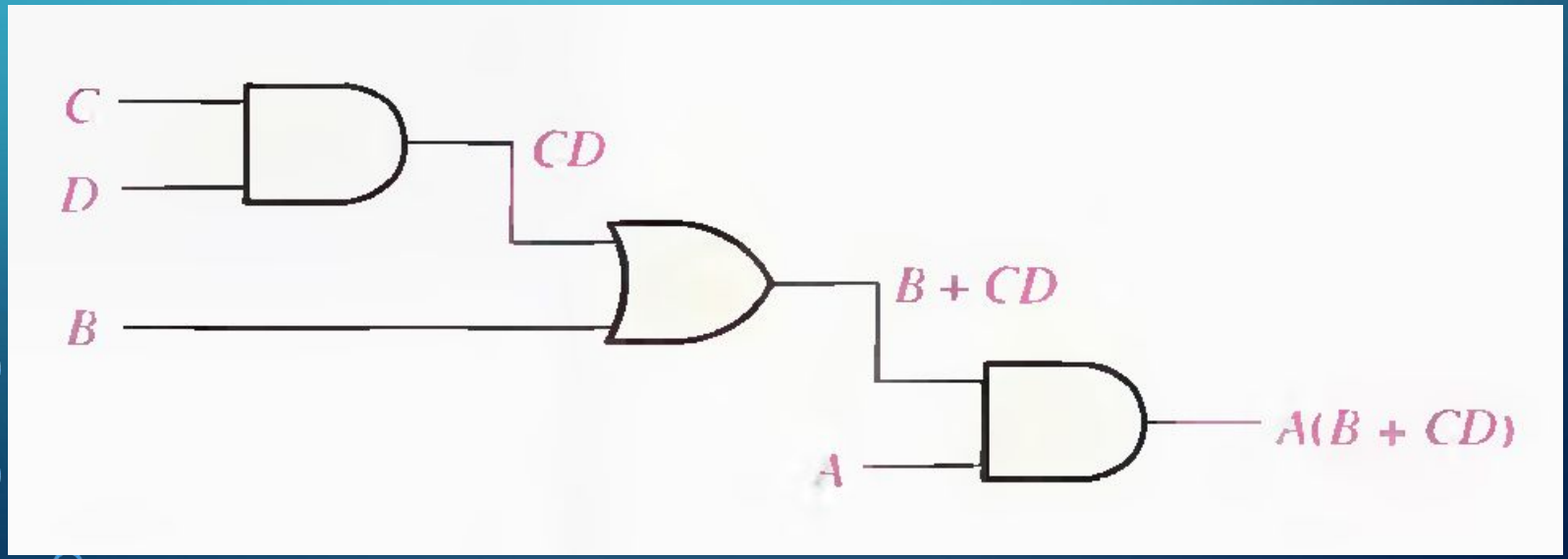
$$\overline{A}BC + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C + ABC$$

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

BOOLEAN ANALYSIS OF LOGIC CIRCUITS

Combinational logic circuits can be analyzed by writing the expression for each gate and combining the expressions according to the rules for Boolean algebra.

Boolean Expression for a Logic Circuit



Constructing a Truth Table for a Logic Circuit

A truth table shows the output for all possible values of the input variables can be developed. For four inputs A , B, C & D , sixteen(2^4) combination of values are possible.

Evaluating the Expression To evaluate the expression $A(B + CD)$, first find the values of the variables that make the expression equal to 1, using the rules for Boolean addition and multiplication. In this case, the expression equals 1 only if $A = 1$ and $B + CD = 1$ because

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

Now determine when the $B + CD$ term equals 1. The term $B + CD = 1$ if either $B = 1$ or $CD = 1$ or if both B and CD equal 1 because

$$B + CD = 1 + 0 = 1$$
$$B + CD = 0 + 1 = 1$$
$$B + CD = 1 + 1 = 1$$

The term $CD = 1$ only if $C = 1$ and $D = 1$.

To summarize, the expression $A(B + CD) = 1$ when $A = 1$ and $B = 1$ regardless of the values of C and D or when $A = 1$ and $C = 1$ and $D = 1$ regardless of the value of B. The expression $A(B + CD) = 0$ for all other value combinations of the variables.

Truth Table

To summarize the expression
A (B +CD) = 1
When A=1 and B = 1 regardless of the values of C and D
OR
When A=1 and C= 1 and D = 1 regardless of the value of B.

The expression
A (B +CD) = 0 for all other value Combination s

INPUTS				OUTPUT
A	B	C	D	A(B + CD)
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	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

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, ABCD + ABCD + ABCD

Converting Product Terms to Standard SOP

Converting Product Terms to Standard SOP Each product term in an SOP expression that does not contain all the variables in the domain can be expanded to standard form to include all variables in the domain and their complements.
As stated in the following steps:

Step 1. Multiply each nonstandard product term by a term made up of the sum of a missing variable and its complement ($A + A' = 1$) . This results in two product terms. As you know, you can multiply anything by 1 without changing its value.

Step 2. Repeat Step 1 until all resulting product terms contain all variables in the domain in either complemented or uncomplemented form. In converting a product term to standard form, the number of product terms is doubled for each missing variable.

Convert the following Boolean expression into standard SOP form:

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

Solution The domain of this SOP expression is A, B, C, D. Take one term at a time. The first term, $A\overline{B}C$, is missing variable D or $\overline{D}$, so multiply the first term by $D + \overline{D}$ as follows:

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

In this case, two standard product terms are the result.

The second term, $\overline{A}\,\overline{B}$, is missing variables C or $\overline{C}$ and D or $\overline{D}$, so first multiply the second term by $C + \overline{C}$ as follows:

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

The two resulting terms are missing variable D or $\overline{D}$, so multiply both terms by $D + \overline{D}$ as follows:

$$\begin{aligned}\overline{A}\,\overline{B} &= \overline{A}\,\overline{B}C + \overline{A}\,\overline{B}\,\overline{C} = \overline{A}\,\overline{B}C(D + \overline{D}) + \overline{A}\,\overline{B}\,\overline{C}(D + \overline{D}) \\ &= \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\end{aligned}$$

In this case, four standard product terms are the result.

The third term, $AB\overline{C}D$, is already in standard form. The complete standard SOP form of the original expression is as follows:

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

Converting a Sum Term to Standard POS

As stated in the following steps, a non-stanard POS expression is converted into standard form using Boolean algebra rule 8 (A . A = 0) : A variable multiplied by its complement equals O.

Step 1. Add to each nonstandard product term a term made up of the product of the missing variable and its complement. This results in two sum terms. As you know, you can add 0 to anything without changing its value.

Step 2. Apply rule 12 : A + BC = (A + B)(A + C)

Step 3. Repeat Step 1 until all resulting sum terms contain all variables in the domain in either complemented or uncomplemented form.

Convert the following Boolean expression into standard POS form:

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

Solution The domain of this POS expression is A, B, C, D. Take one term at a time. The first term, $A + \overline{B} + C$, is missing variable D or $\overline{D}$, so add $D\overline{D}$ and apply rule 12 as follows:

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

The second term, $\overline{B} + C + \overline{D}$, is missing variable A or $\overline{A}$, so add $A\overline{A}$ and apply rule 12 as follows:

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

The third term, $A + \overline{B} + \overline{C} + D$, is already in standard form. The standard POS form of the original expression is as follows:

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

Converting Standard SOP to Standard POS

Convert the following SOP expression to an equivalent POS expression:

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

Solution

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

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

1. Identify each of the following expressions as SOP, standard SOP, POS, or standard POS:
   (a) $AB + \overline{A}BD + \overline{A}C\overline{D}$ (b) $(A + \overline{B} + C)(A + B + \overline{C})$
   (c) $\overline{A}BC + AB\overline{C}$ (d) $(A + \overline{C})(A + B)$
2. Convert each SOP expression in Question 1 to standard form.
3. Convert each POS expression in Question 1 to standard form.

Boolean Expressions and Truth Tables

Converting SOP Expressions to Truth Table Format

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

TABLE 4–6

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

Create a truth table for the standard SOP expression $ABC + ABC$.

Converting POS Expressions to Truth Table Format

Determine the truth table for the following standard POS expression:

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

TABLE 4–7

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	

Develop a truth table for the following standard POS expression:

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

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

TABLE 4–8

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

Solution

There are four 1s in the output column and the corresponding binary values are 011, 100, 110, and 111. Convert these binary values to product terms as follows:

$$011 \longrightarrow \overline{A}BC$$
$$100 \longrightarrow A\overline{B}\,\overline{C}$$
$$110 \longrightarrow AB\overline{C}$$
$$111 \longrightarrow ABC$$

The resulting standard SOP expression for the output X is

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

For the POS expression, the output is 0 for binary values 000, 001, 010, and 101. Convert these binary values to sum terms as follows:

$$000 \longrightarrow A + B + C$$
$$001 \longrightarrow A + B + \overline{C}$$
$$010 \longrightarrow A + \overline{B} + C$$
$$101 \longrightarrow \overline{A} + B + \overline{C}$$

The resulting standard POS expression for the output X is

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

BINARY FUNCTION EXPRESSION

- So far have seen to possible ways
  - Binary equations
  - Truth tables
- What other ways are there?

STANDARD FORMS

- Facilitate simplification
- Result in more desirable implementations
- Standard Forms rely on two type of terms
  - *Product Terms* – Terms that are ANDed together
    - XYZ
    - (A+B)(C+D)(A+D)
  - *Sum Terms* – Terms that are ORed together
    - X+Y+Z

Binary Representation of a Standard Product Term

An SOP expression is equal to 1 only if one or more of the product terms in the expression is equal to 1.

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

Binary Representation of a Standard Sum Term

A POS expression is equal to 0 only if one or more of the sum terms in the expression is equal to 0.

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

MINTERMS

- Boolean Functions can be defined by truth tables. In a Boolean function, a product term in which all the variables appear is called a minterm of the function.
- Minterms specify the function as an OR of the minterms (product terms).
- For 2 variables have 4 minterms
  - X'Y' X'Y XY' XY
- In general, if a function has n variables there are 2^n minterms
- The subscript on the minterm is the decimal of the binary

SUM-OF-PRODUCTS

- Starting with the minterm specification of a function

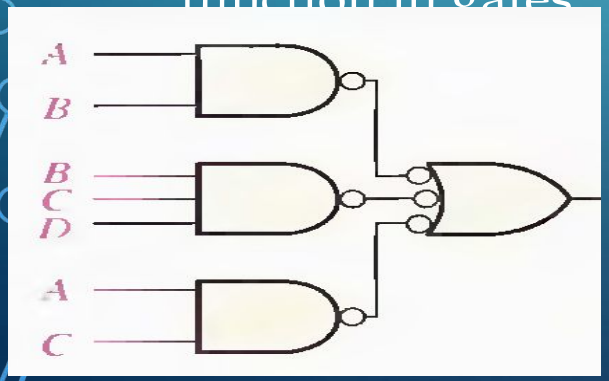
  $F(X,Y,Z) = \sum m(2,3,4,7)$

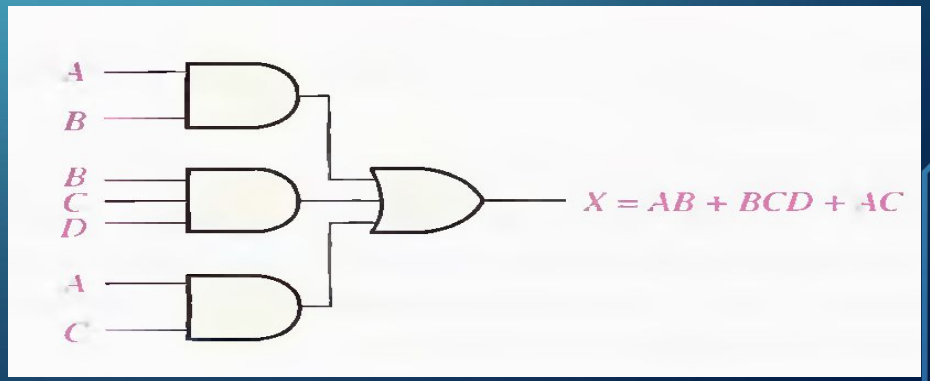
  $= X'YZ' + X'YZ + XY'Z' + XYZ$

- Each minterm represents a product term and then we sum them to generate the function.
- This form is called *sum-of-products*.
- Even when in minimal form it is still the sum-of-products.

  $F = AB + CD + CE$

  - The sum-of-products form is a 2 level implementation of the function in gates





SOP Given a Table of Combinations

- What is the SOP form for the following 3 input / 1 output digital device?

S	A	B	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Computing the SOP (2)

S	A	B	f	minterm name
0	1	0	1	m_2
0	1	1	1	m_3
1	0	1	1	m_5
1	1	1	1	m_7

This SOP has 4 minterms:

f = S'AB' + S'AB + SA'B + SAB

Canonical (Standard) SOP

Boolean functions can use shorthand notation when in SOP form:

- f = S'AB' + S'AB + SA'B + SAB

$$f(S,A,B) = \Sigma(m_2, m_3, m_5, m_7)$$

or

$$f(S,A,B) = \Sigma m(2,3,5,7)$$

Canonical SOP Example

$f(x_1,x_2,x_3) = \Sigma m(1,4,5,6)$

minterm	x_1	x_2	x_3	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

x1'x2'x3 + x1x2'x3' + x1x2'x3 + x1x2x3'

MAXTERMS

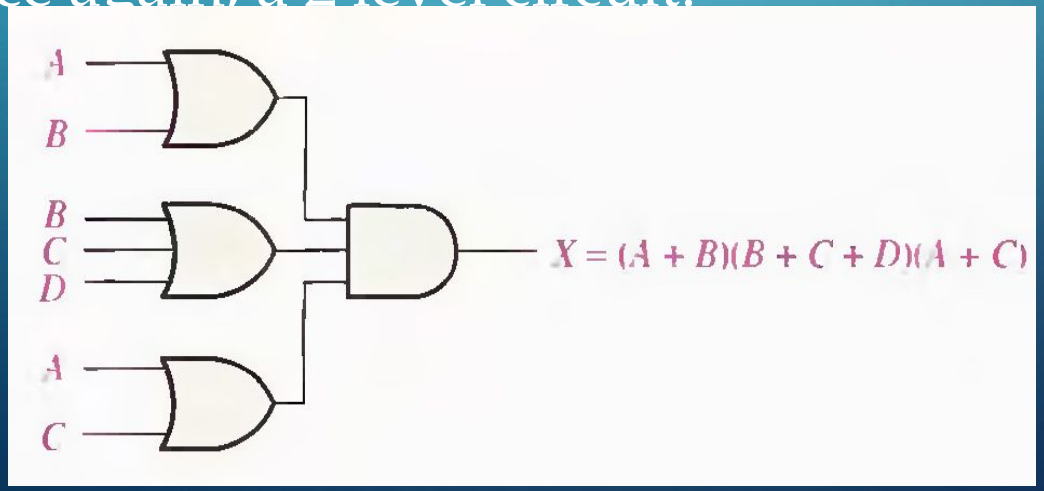
- A sum term that contains all the variables in complemented or un-complemented form is called a maxterm.
- For n variables , 2^n maxterms.

PRODUCT-OF-SUMS

- This form arises from the Maxterm representation of a function.

$$F(X,Y,Z) = \prod M\,(2,3,4,6) = (X+Y'+Z)(X+Y'+Z')(X'+Y+Z)(X'+Y'+Z)$$

- And now have F in a product-of-sums form
- Once again, a 2 level circuit.



Computing the POS

- Identify rows with "0" on output (f = 0)

A	B	f
0	0	0
0	1	1
1	0	0
1	1	0

- Represent the input for each 0 row as a *maxterm*

Canonical POS Example

- $f(x_1,x_2,x_3) = \Pi(M_0,M_2,M_3,M_7) = \Pi M(0,2,3,7)$

maxterm	x_1	x_2	x_3	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

- f = (x1+x2+x3)(x1+x2'+x3)(x1+x2'+x3')(x1'+x2'+x3')

Example: 3 Way Light Control

- L(A,B,C) = Σ*m*(5,6)

or

L(A,B,C) = Π*M*(0,1,2,3,4,7)

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

- SOP:
  - L = (A B' C)+(A B C')
- POS: (A+B+C)(A+B+C') (A+B'+C)(A+B'+C')(A'+B+C)(A'+B'+C')
  - L =

INTERCHANGE CANONICAL SOP AND POS

- For the same Boolean eqn
  - Canonical SOP form is complementary to its canonical POS form
  - Use missing terms to interchange Σ and Π

- Examples
  - $F(A,B,C) = \Sigma\, m(0,1,4,6,7)$

  Can be re-expressed by

  - $F(A,B,C) = \Pi\, M(2,3,5)$

    Where 2, 3, 5 are the missing minterms in the canonical SOP form

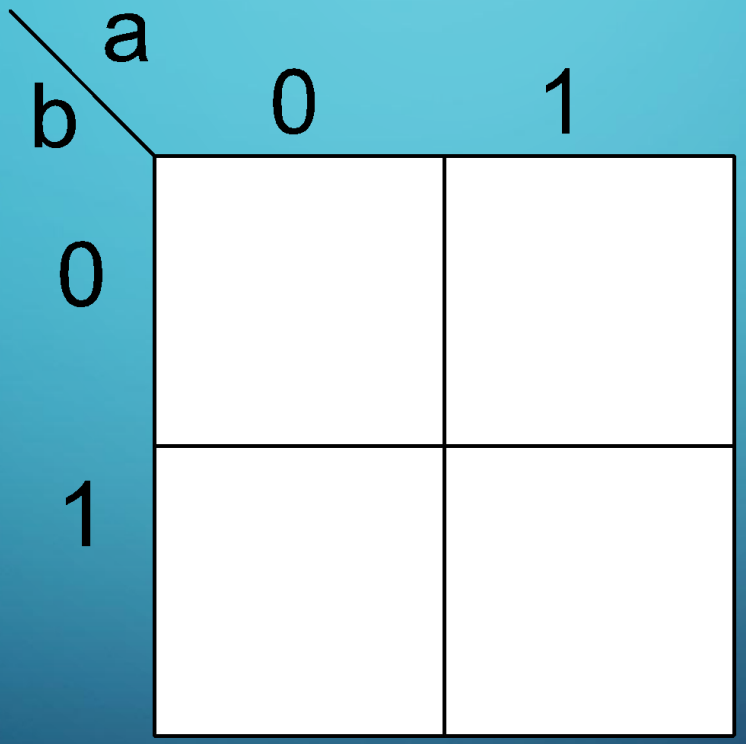
WHAT ARE KARNAUGH MAPS?

- Karnaugh maps provide an alternative way of simplifying logic circuits.
- Instead of using Boolean algebra simplification techniques, you can transfer logic values from a Boolean statement or a truth table into a Karnaugh map.
- The arrangement of 0's and 1's within the map helps you to visualize the logic relationships between the variables and leads directly to a simplified Boolean statement.

KARNAUGH MAPS (K-MAPS)

- An *n*-variable K-map has 2^n cells with each cell corresponding to an *n*-variable truth table value.
- K-map cells are labeled with the corresponding truth-table row.
- K-map cells are arranged such that adjacent cells correspond to truth rows that differ in only one bit position (*logical adjacency*).

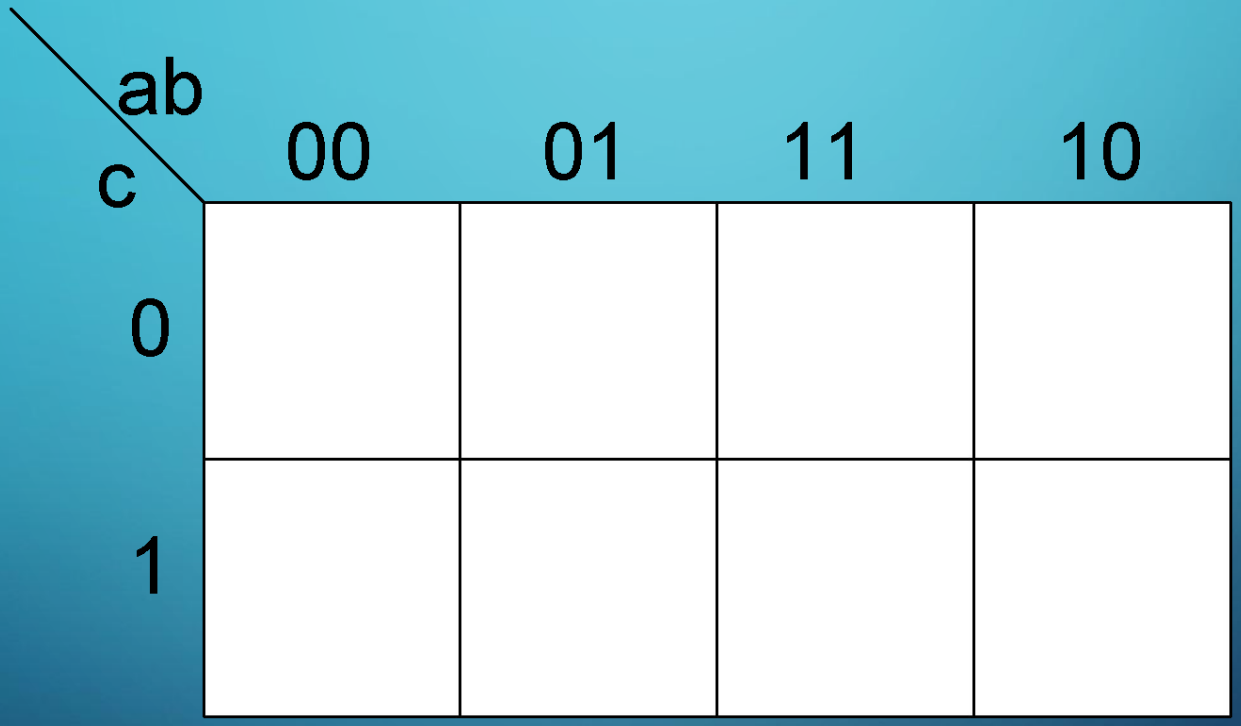
TWO-VARIABLE K-MAP



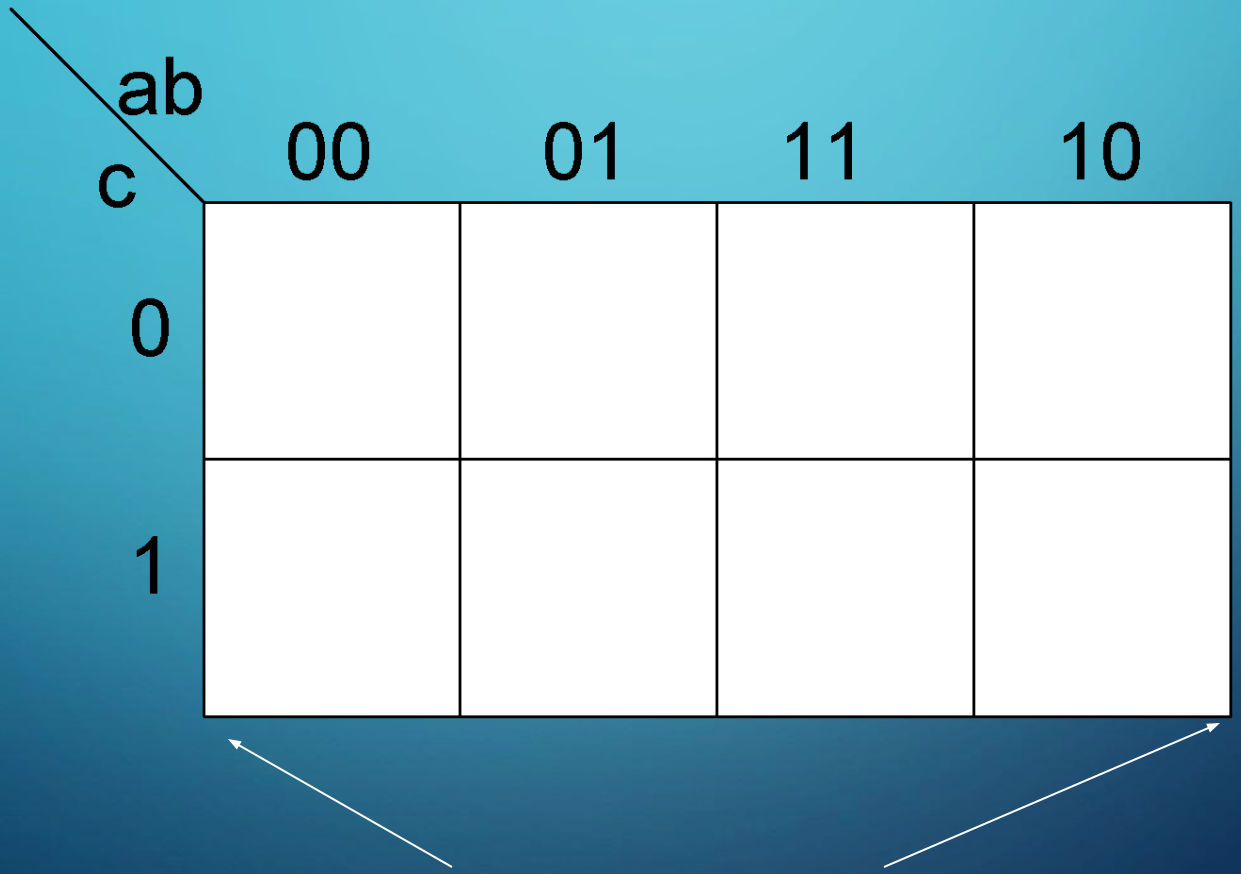
THREE-VARIABLE K-MAP

c \ ab	00	01	11	10
0	m_0	m_2	m_6	m_4
1	m_1	m_3	m_7	m_5

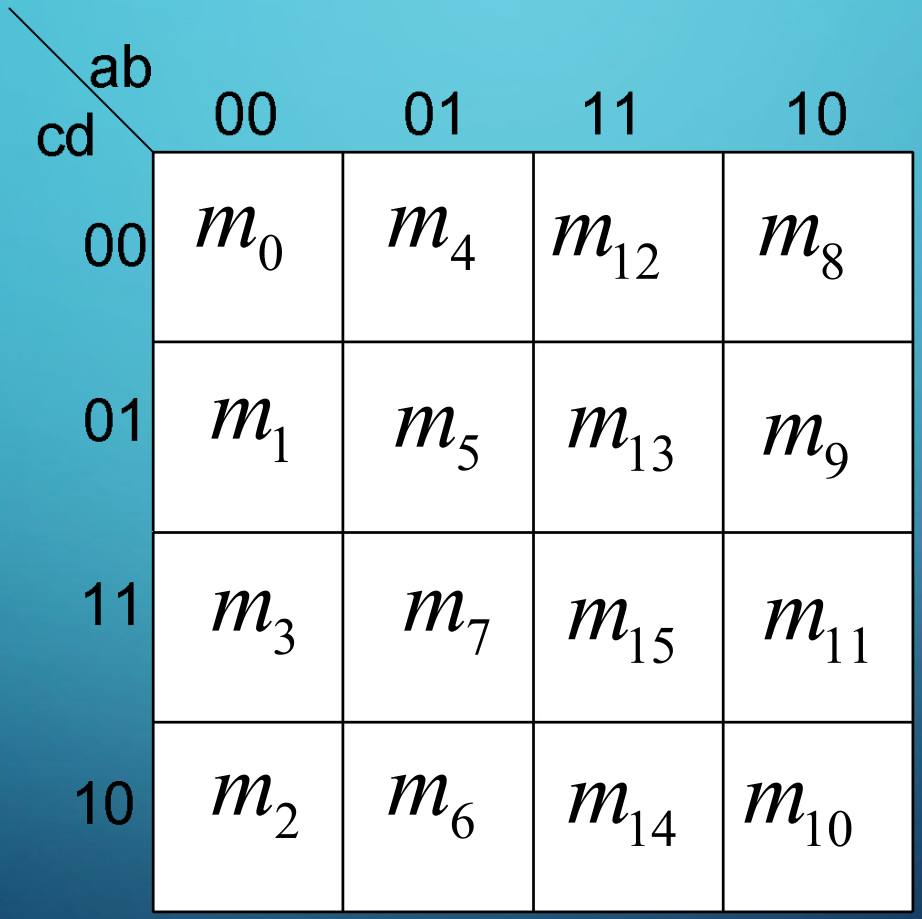
THREE-VARIABLE K-MAP



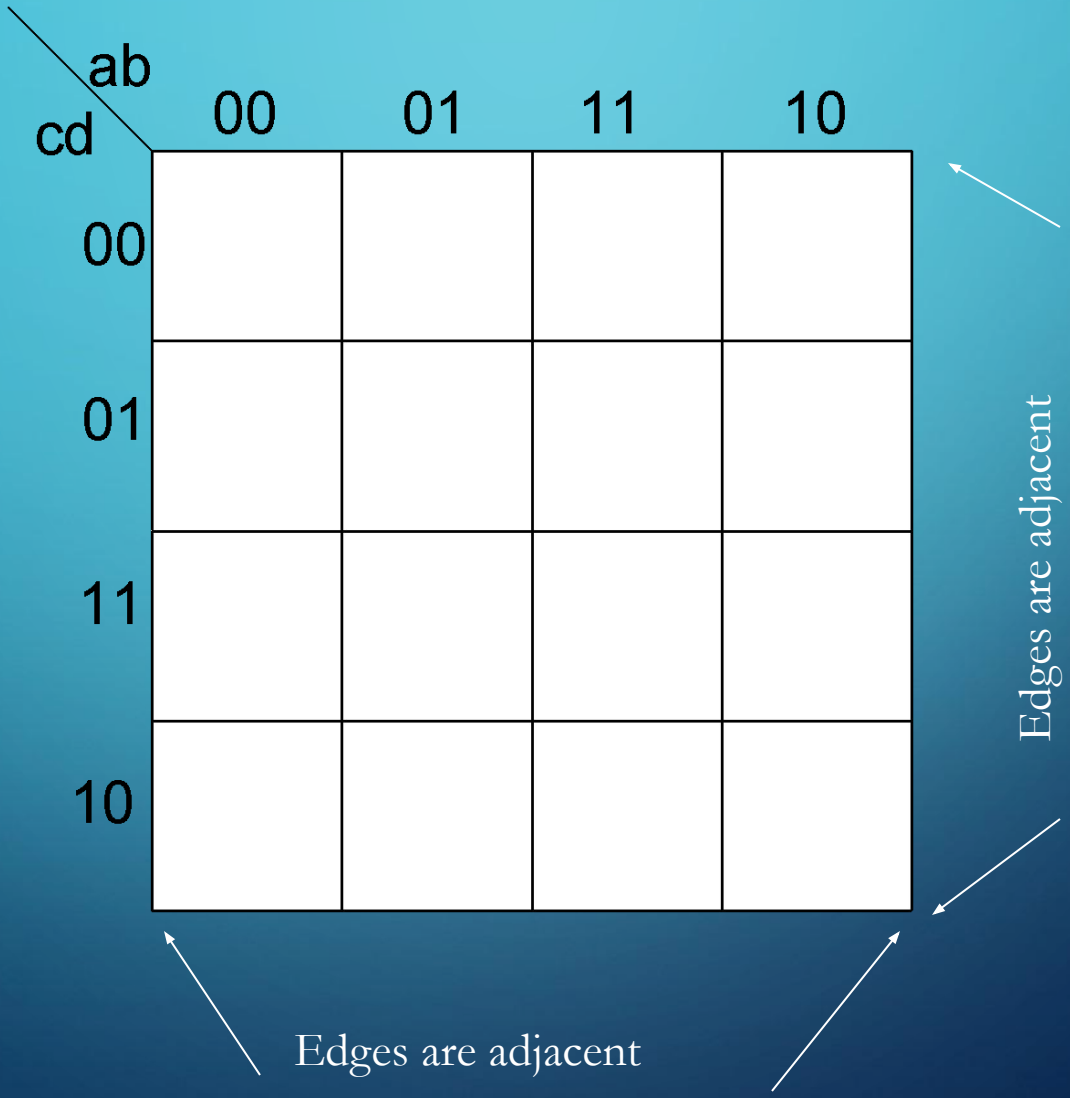
THREE-VARIABLE K-MAP



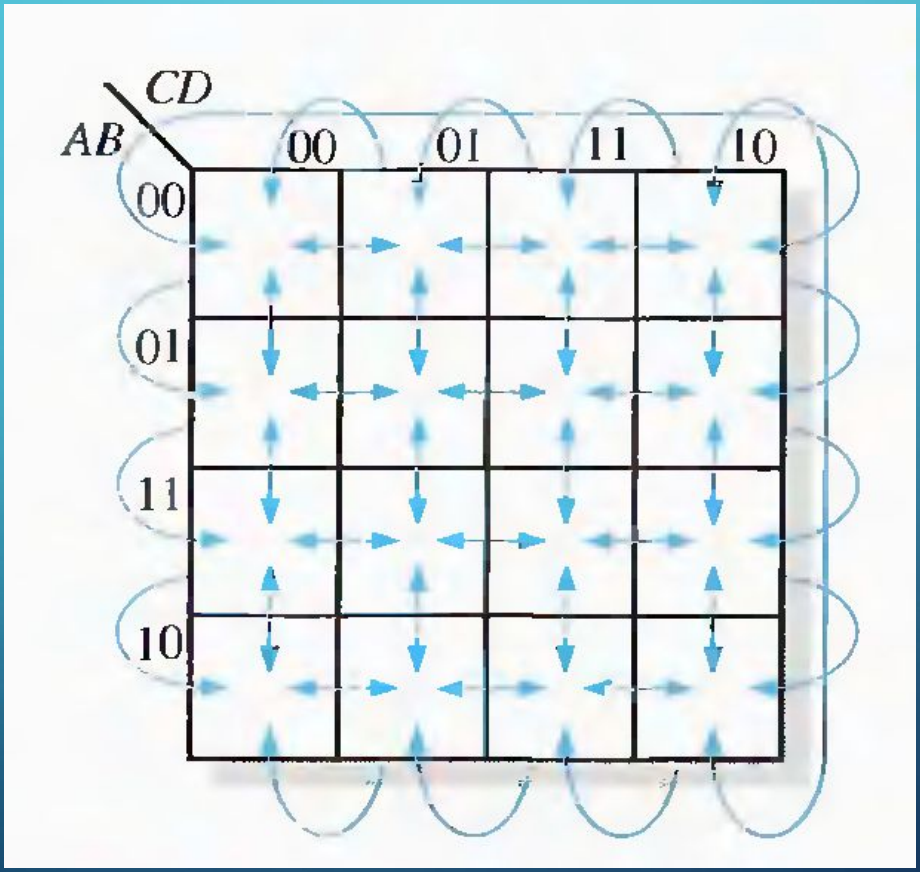
FOUR-VARIABLE K-MAP



FOUR-VARIABLE K-MAP

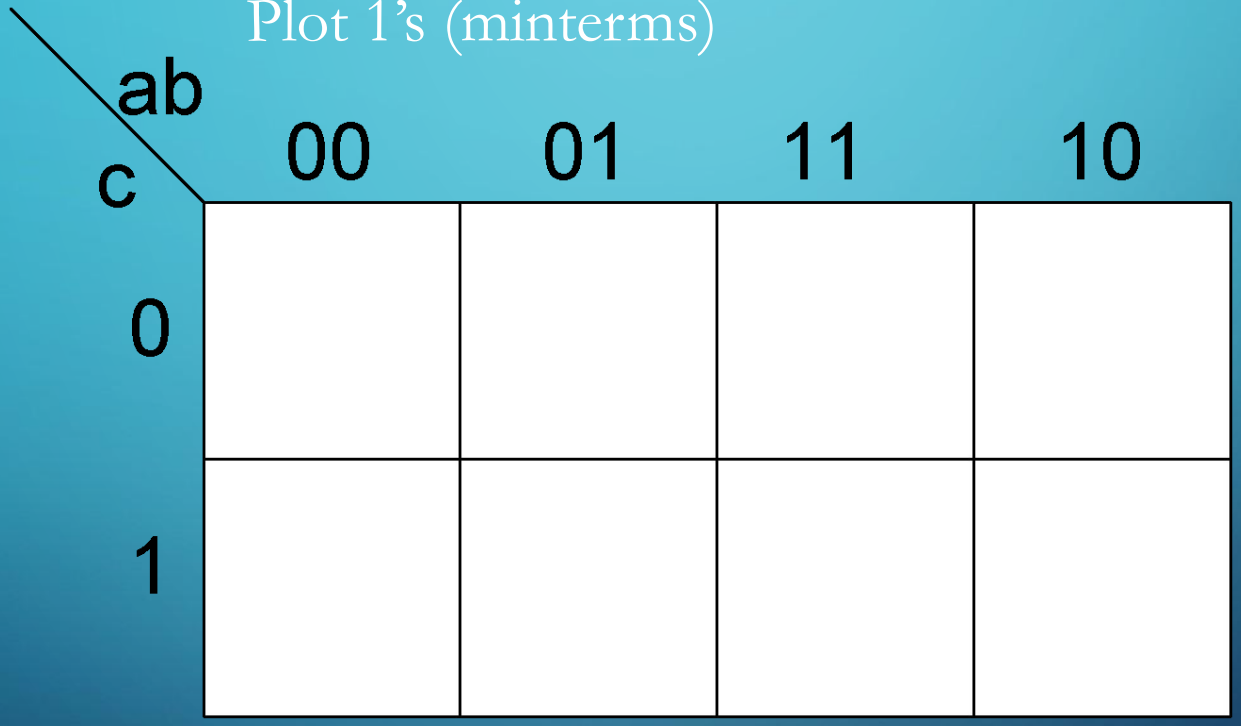


FOUR-VARIABLE K-MAP



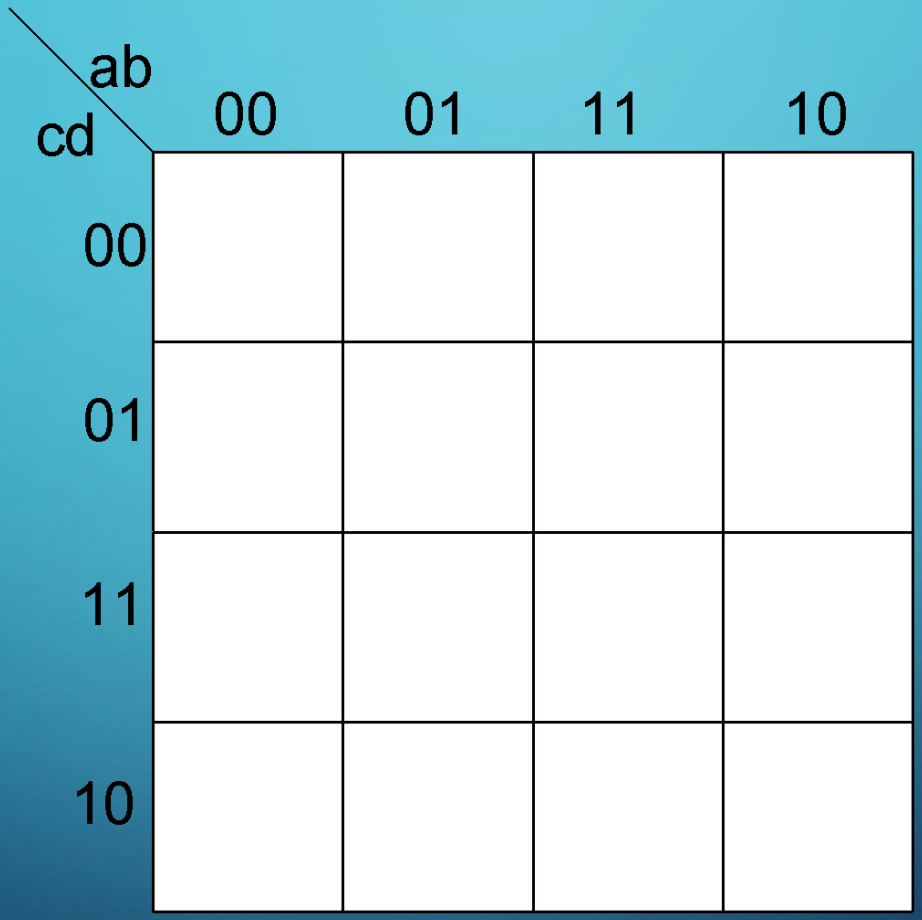
THREE-VARIABLE K-MAP EXAMPLE

Plot 1's (minterms)



$$F(a,b,c) = \sum m(1,3,5,6)$$

FOUR-VARIABLE K-MAP EXAMPLE



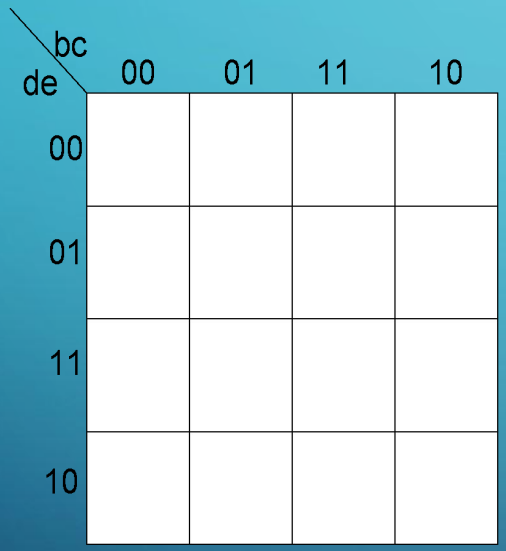
$$F(a,b,c,d) = \sum m(0,2,9,12,14)$$

FIVE VARIABLE K-MAP

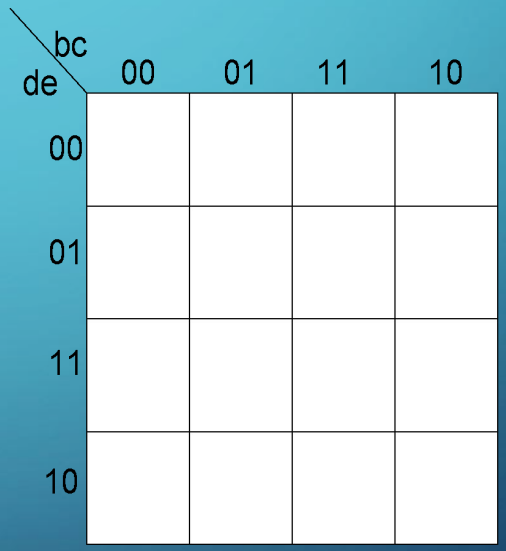
Use two four variable K-maps

USE TWO FOUR-VARIABLE K-MAPS

A=0 map



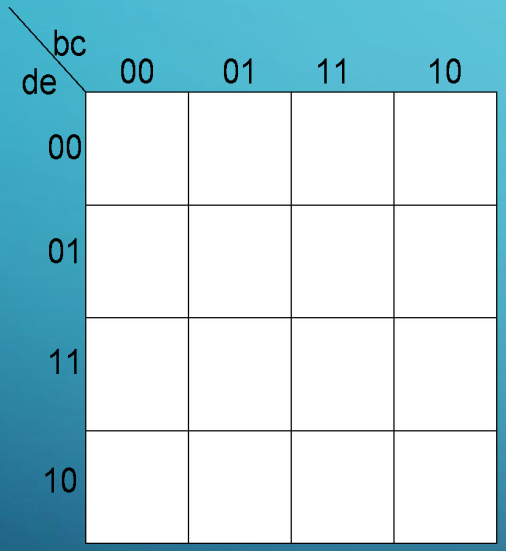
A=1 map



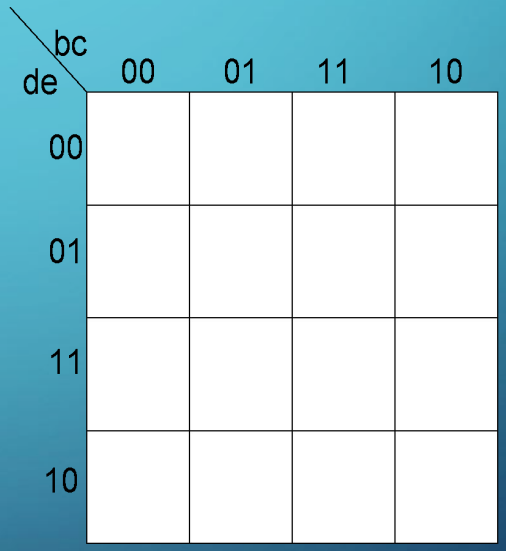
$$F(a,b,c,d,e) = \sum m(5,7,13,15,21,23,29,31)$$

USE TWO FOUR-VARIABLE K-MAPS

A=0 map



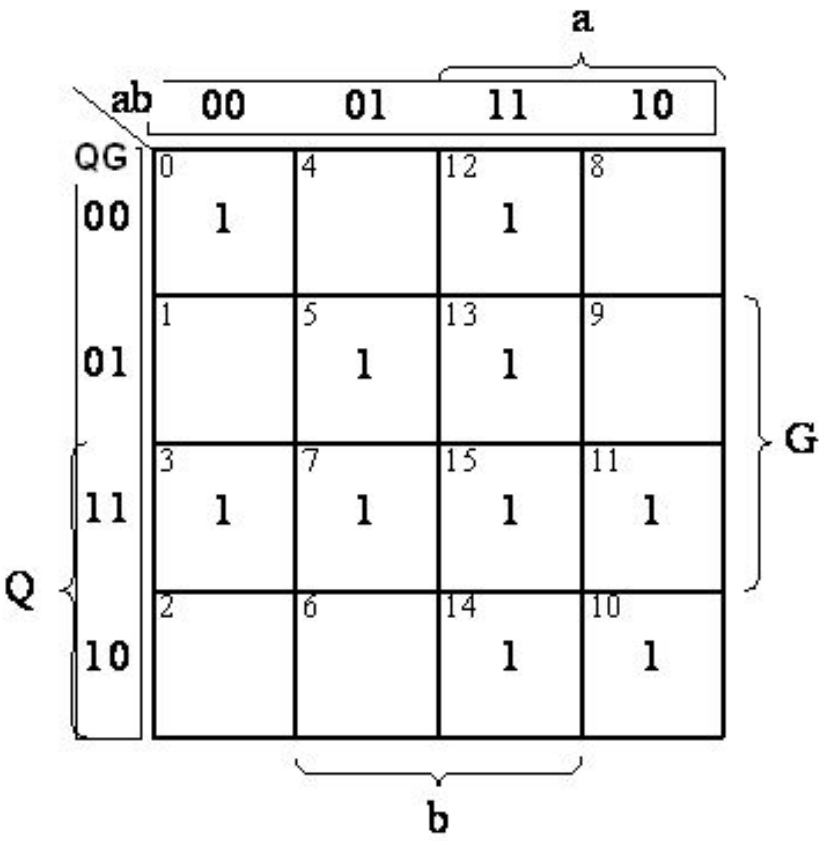
A=1 map



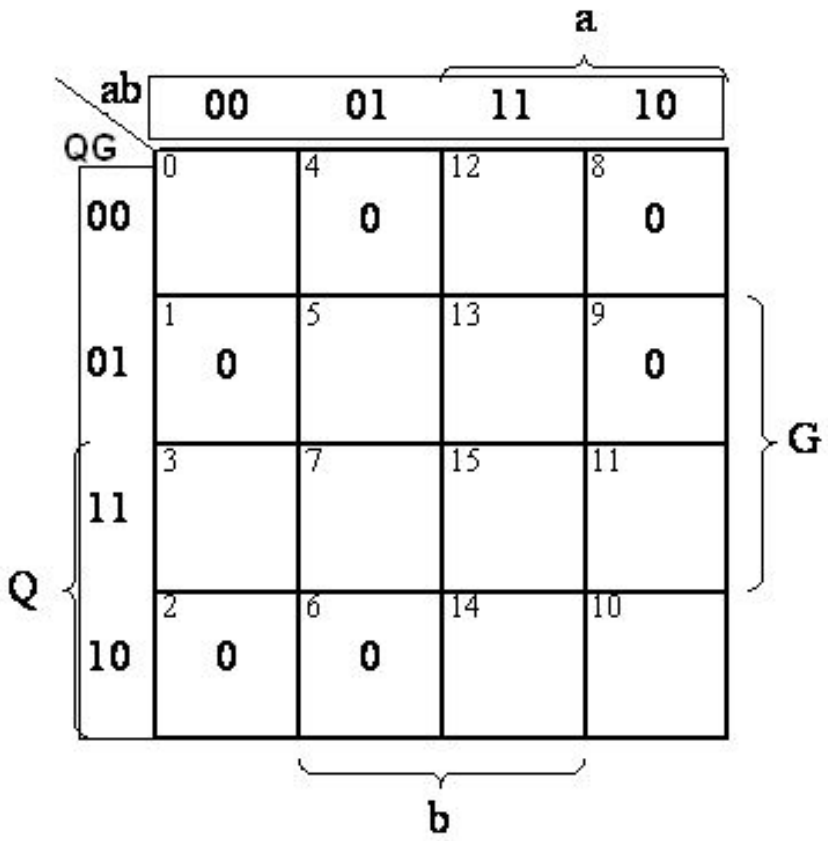
$$F(a,b,c,d,e) = \sum m(5,7,13,15,21,23,29,31)$$

FOUR VARIABLE EXAMPLE (A) MINTERM FORM. (B) MAXTERM FORM.

$f(a,b,Q,G) = \sum m(0,3,5,7,10,11,12,13,14,15) = \prod M(1,2,4,6,8,9)$



(a)

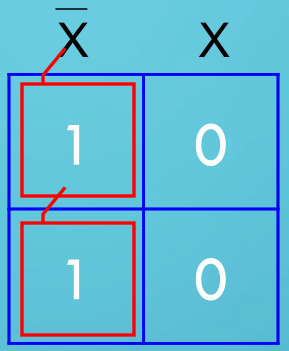


(b)

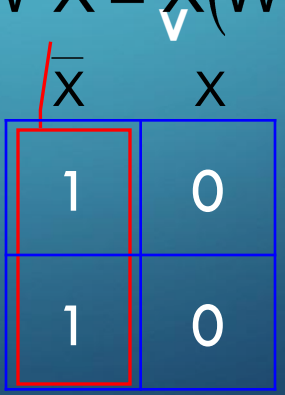
ADJACENT CELLS = SIMPLIFICATION

$\overline{W}\,\overline{X}$

$W\,\overline{X}$



$$\overline{W}\,\overline{X} + W\,\overline{X} = \overline{X}\left(\overline{W} + W\right) = \overline{X}$$

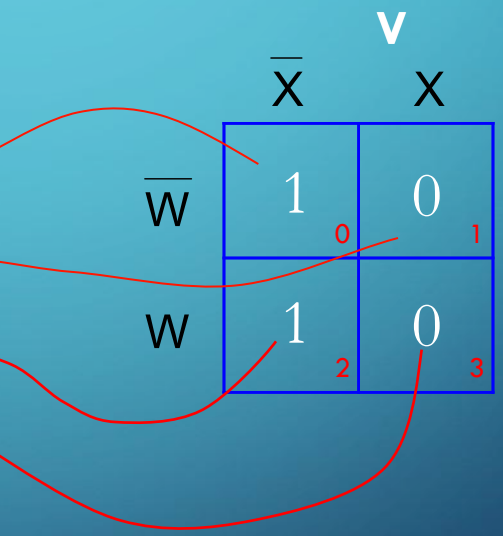


$\overline{X}$

TRUTH TABLE TO K-MAP MAPPING

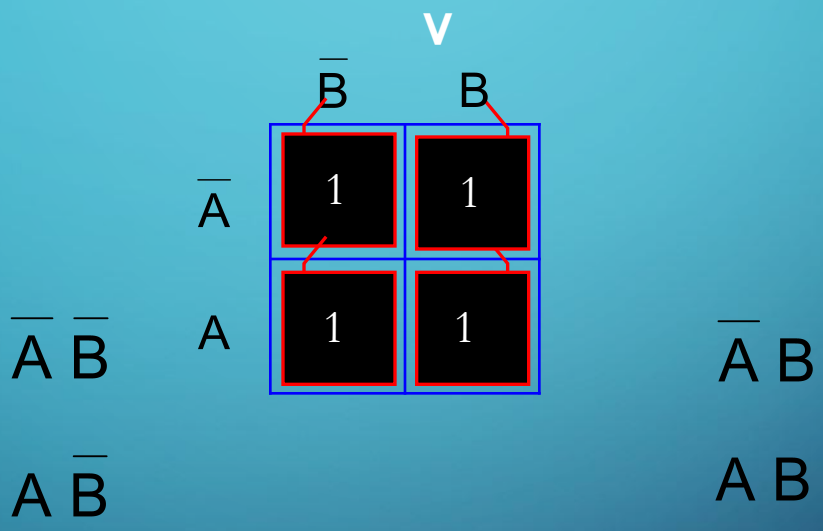
Two Variable K-Map

	W	X	F_{WX}
Minterm – 0	0	0	1
Minterm – 1	0	1	0
Minterm – 2	1	0	1
Minterm – 3	1	1	0



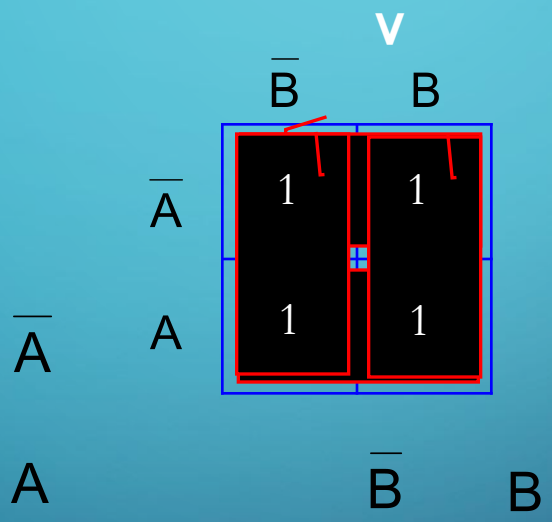
TWO VARIABLE K-MAP GROUPINGS

Groups of One – 4



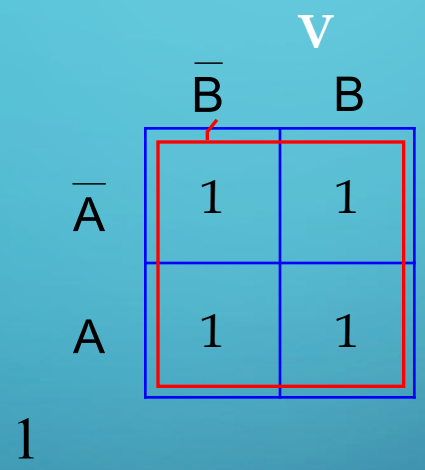
TWO VARIABLE K-MAP GROUPINGS

Groups of Two – 4



TWO VARIABLE K-MAP GROUPINGS

Group of Four – 1



K-MAP SIMPLIFICATION PROCESS

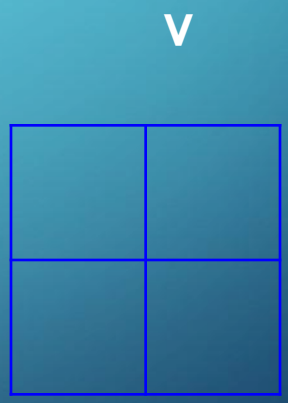
1. Construct a label for the K-Map. Place 1s in cells corresponding to the 1s in the truth table. Place 0s in the other cells.
2. Identify and group all isolated 1's. Isolated 1's are ones that cannot be grouped with any other one, or can only be grouped with one other adjacent one.
3. Group any hex.
4. Group any octet, even if it contains some 1s already grouped but not enclosed in a hex.
5. Group any quad, even if it contains some 1s already grouped but not enclosed in a hex or octet.
6. Group any pair, even if it contains some 1s already grouped but not enclosed in a hex, octet, or quad.
7. OR together all terms to generate the SOP equation.

EXAMPLE :VARIABLE K-MAP

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_1.

J	K	F_1
0	0	1
0	1	1
1	0	0
1	1	0



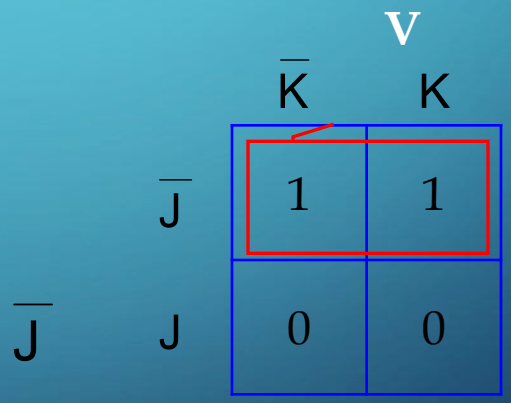
EXAMPLE: VARIABLE K-MAP

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_1.

Solution:

J	K	F_1
0	0	1
0	1	1
1	0	0
1	1	0

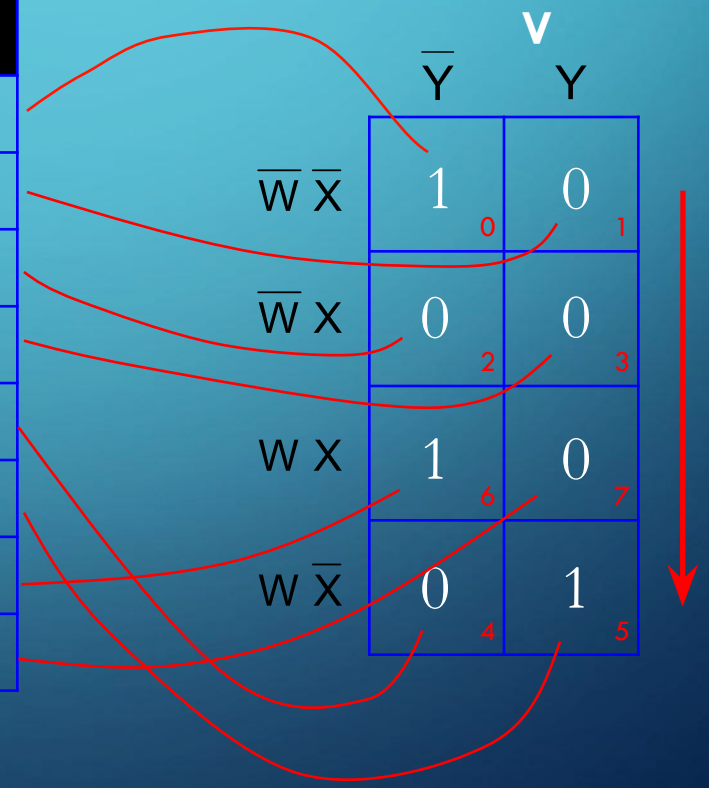


$$F_1 = \overline{J}$$

TRUTH TABLE TO K-MAP MAPPING

Three Variable K-Map

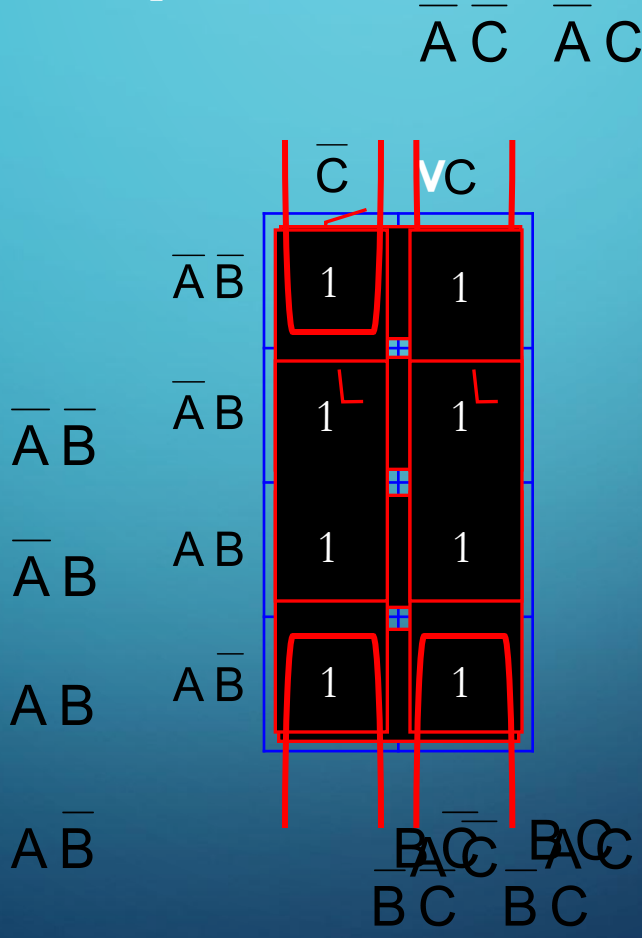
	W	X	Y	F_{WXY}
Minterm – 0	0	0	0	1
Minterm – 1	0	0	1	0
Minterm – 2	0	1	0	0
Minterm – 3	0	1	1	0
Minterm – 4	1	0	0	0
Minterm – 5	1	0	1	1
Minterm – 6	1	1	0	1
Minterm – 7	1	1	1	0



Only one variable changes for every row change

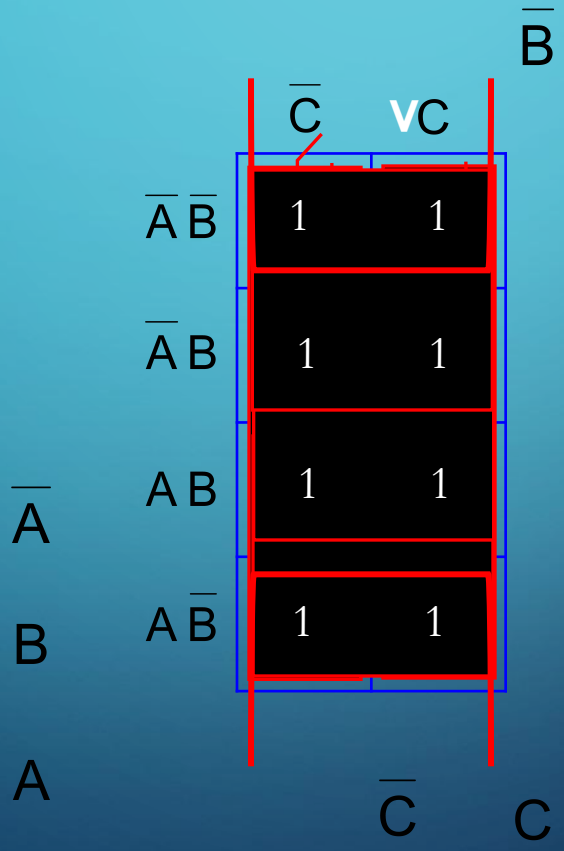
THREE VARIABLE K-MAP GROUPINGS

Groups of One – 8 (not shown)
Groups of Two – 12



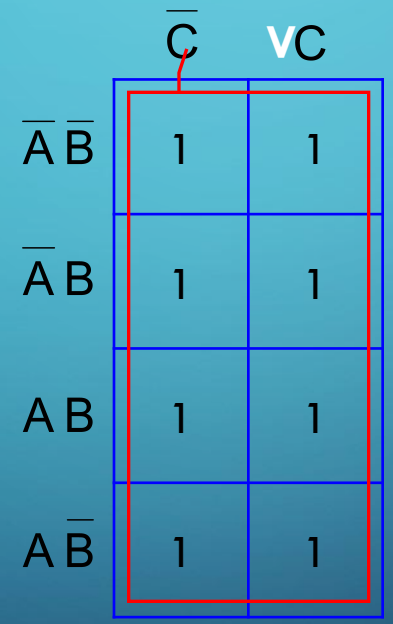
THREE VARIABLE K-MAP GROUPINGS

Groups of Four – 6



THREE VARIABLE K-MAP GROUPINGS

Group of Eight - 1



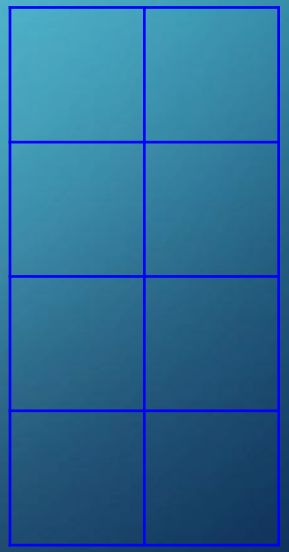
1

EXAMPLE: VARIABLE K-MAP

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_2.

E	F	G	F_2
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0



TRUTH TABLE TO K-MAP MAPPING

Four Variable K-Map

	W	X	Y	Z	F_{WXYZ}
Minterm – 0	0	0	0	0	0
Minterm – 1	0	0	0	1	1
Minterm – 2	0	0	1	0	1
Minterm – 3	0	0	1	1	0
Minterm – 4	0	1	0	0	1
Minterm – 5	0	1	0	1	1
Minterm – 6	0	1	1	0	0
Minterm – 7	0	1	1	1	1
Minterm – 8	1	0	0	0	0
Minterm – 9	1	0	0	1	0
Minterm – 10	1	0	1	0	1
Minterm – 11	1	0	1	1	0
Minterm – 12	1	1	0	0	1
Minterm – 13	1	1	0	1	0
Minterm – 14	1	1	1	0	1
Minterm – 15	1	1	1	1	1

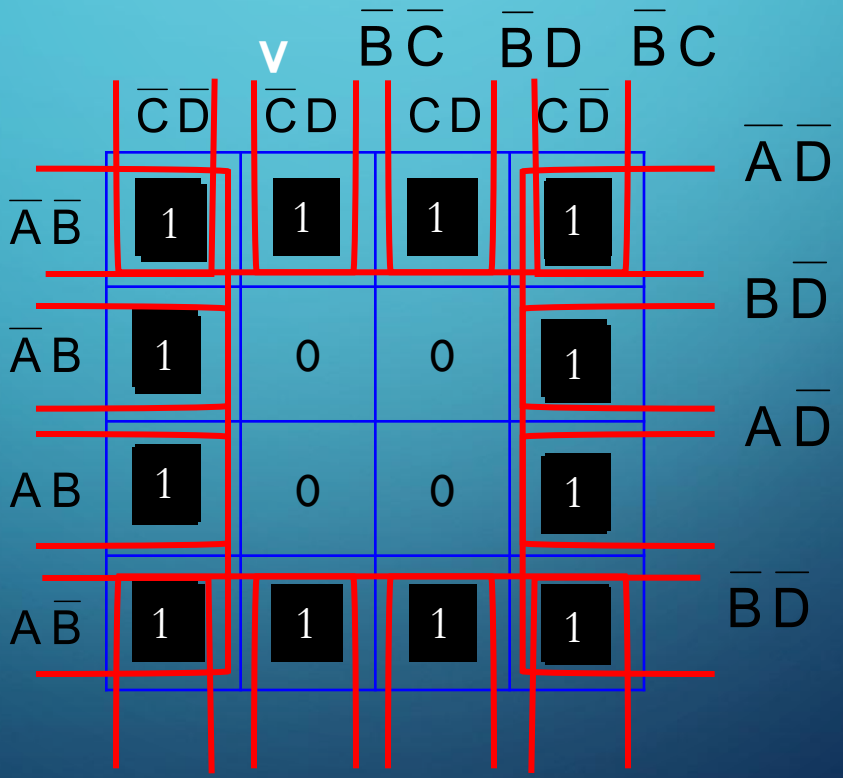
Only one variable changes for every column change

	$\bar{Y}\bar{Z}$	$\bar{Y}Z$	YZ	$Y\bar{Z}$
$\bar{W}\bar{X}$	0 (0)	1 (1)	0 (3)	1 (2)
$\bar{W}X$	1 (4)	1 (5)	1 (7)	0 (6)
WX	1 (12)	0 (13)	1 (15)	1 (14)
$W\bar{X}$	0 (8)	0 (9)	0 (11)	1 (10)

Only one variable changes for every row change

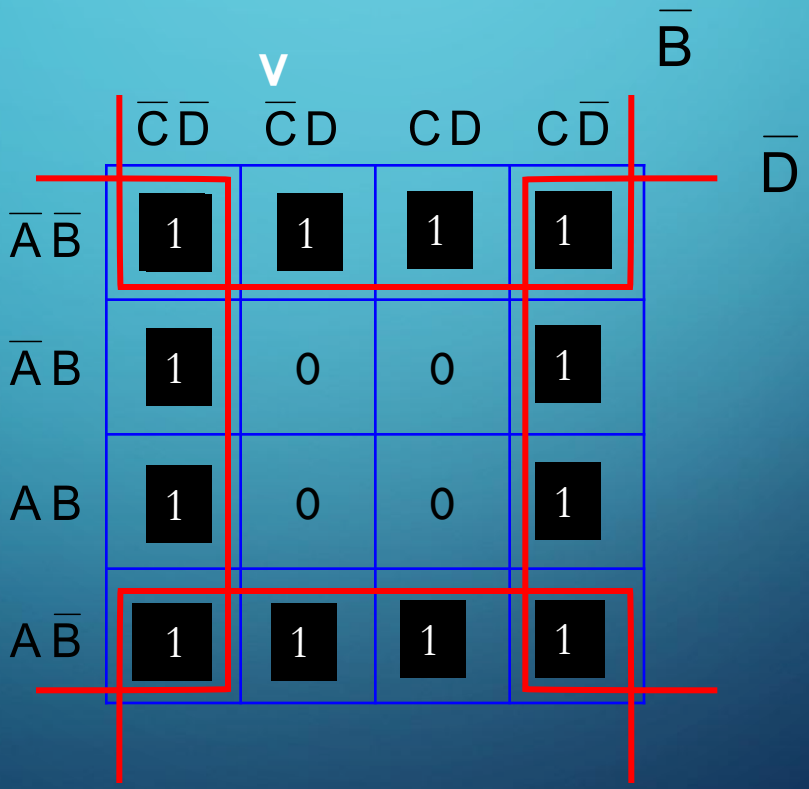
FOUR VARIABLE K-MAP GROUPINGS

Groups of One – 16 (not shown)
Groups of Two – 32 (not shown)
Groups of Four – 24 (seven shown)



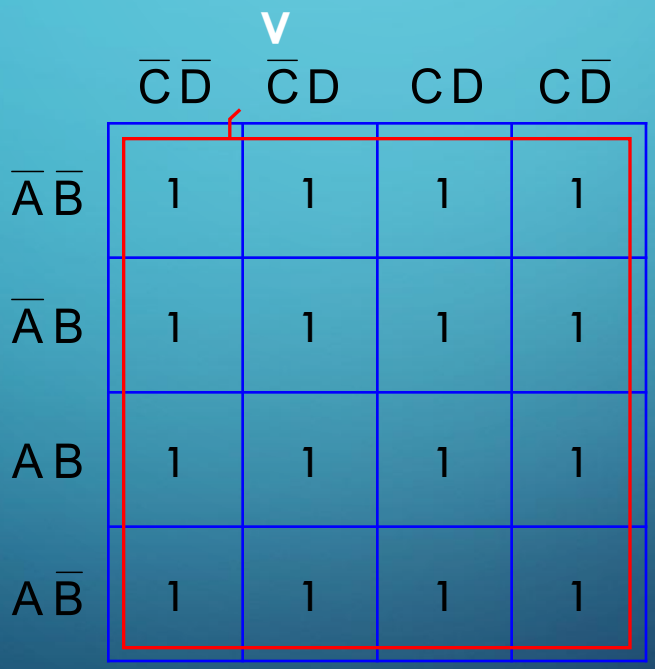
FOUR VARIABLE K-MAP GROUPINGS

Groups of Eight – 8 (two shown)



FOUR VARIABLE K-MAP GROUPINGS

Group of Sixteen – 1



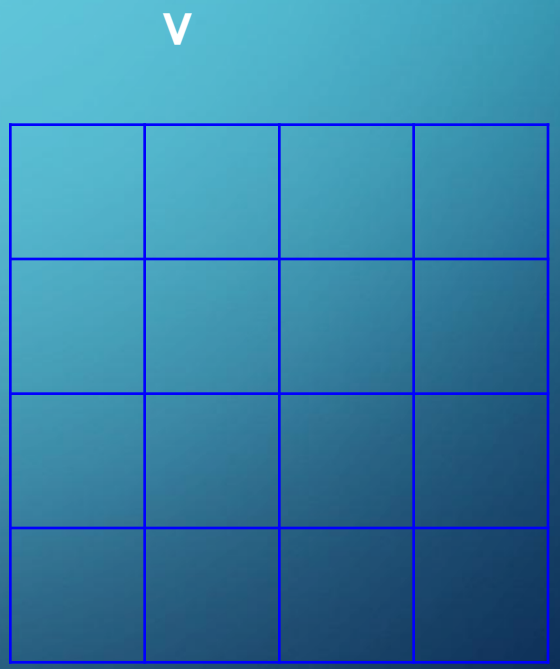
1

EXAMPLE : 4 VARIABLE K-MAP

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_3.

R	S	T	U	F_3
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1



EXAMPLE : DON'T CARE CONDITIONS

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_4. Be sure to take advantage of the *don't care* conditions.

Solution:

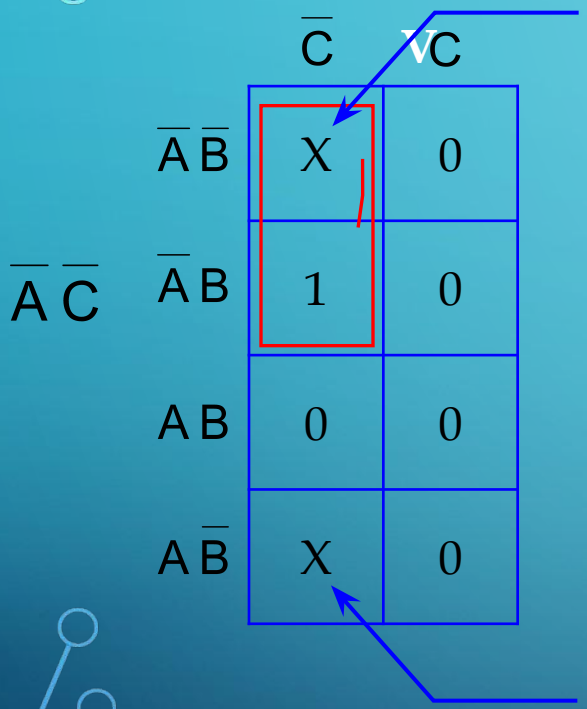
R	S	T	U	F_4
0	0	0	0	X
0	0	0	1	0
0	0	1	0	1
0	0	1	1	X
0	1	0	0	0
0	1	0	1	X
0	1	1	0	X
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	X
1	1	0	0	X
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$$F_4 = \bar{R}T + R\bar{S}$$

DON'T CARE CONDITIONS

- A *don't care* condition, marked by (X) in the truth table, indicates a condition where the design doesn't care if the output is a (0) or a (1).
- A *don't care* condition can be treated as a (0) or a (1) in a K-Map.
- Treating a *don't care* as a (0) means that you do not need to group it.
- Treating a *don't care* as a (1) allows you to make a grouping larger, resulting in a simpler term in the

Don't Care Conditions



This *don't care* condition was treated as a (1). This allowed the grouping of a single one to become a grouping of two, resulting in a simpler term.

There was no advantage in treating this *don't care* condition as a (1), thus it was treated as a (0) and not grouped.

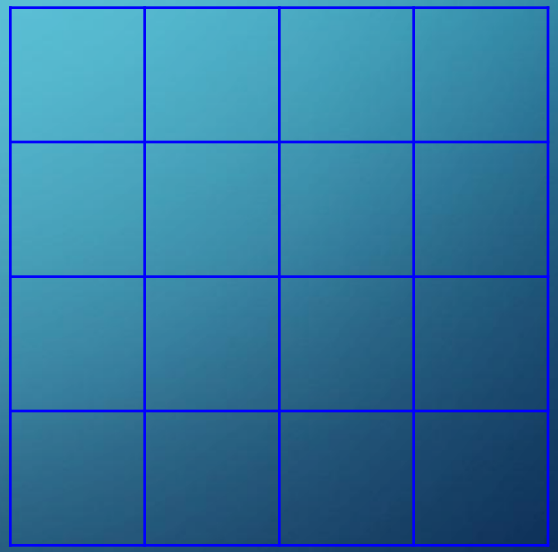
EXAMPLE : DON'T CARE CONDITIONS

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_4. Be sure to take advantage of the *don't care* conditions.

R	S	T	U	F_4
0	0	0	0	X
0	0	0	1	0
0	0	1	0	1
0	0	1	1	X
0	1	0	0	0
0	1	0	1	X
0	1	1	0	X
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	X
1	1	0	0	X
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

V



EXAMPLE : DON'T CARE CONDITIONS

Example:

After labeling and transferring the truth table data into the K-Map, write the simplified sum-of-products (SOP) logic expression for the logic function F_4. Be sure to take advantage of the *don't care* conditions.

Solution:

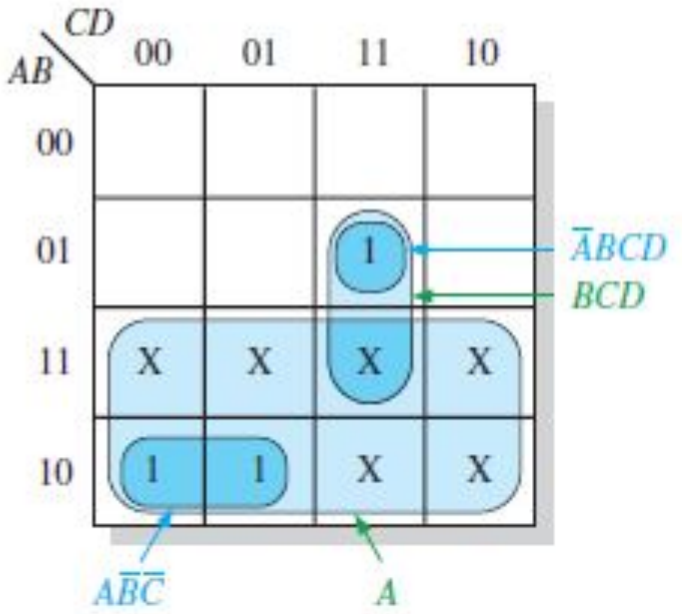
R	S	T	U	F_4
0	0	0	0	X
0	0	0	1	0
0	0	1	0	1
0	0	1	1	X
0	1	0	0	0
0	1	0	1	X
0	1	1	0	X
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	X
1	1	0	0	X
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$$F_4 = \bar{R}T + R\bar{S}$$

Inputs				Output
A	*B*	*C*	*D*	*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

(a) Truth table

Don't cares



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

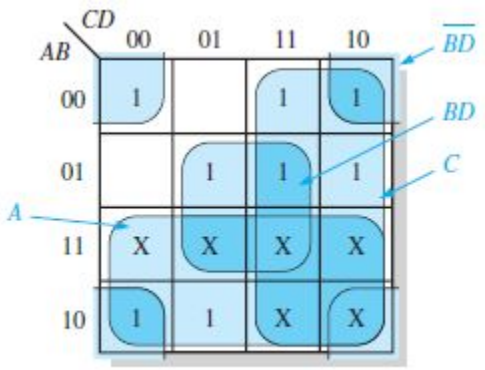
In a 7-segment display, each of the seven segments is activated for various digits. For example, segment *a* is activated for the digits 0, 2, 3, 5, 6, 7, 8, and 9, as illustrated in Figure 4–41. Since each digit can be represented by a BCD code, derive an SOP expression for segment *a* using the variables *ABCD* and then minimize the expression using a Karnaugh map.

FIGURE 4–41 7-segment display.

The expression for segment *a* is

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

Draw the logic diagram for the segment-*a* logic.

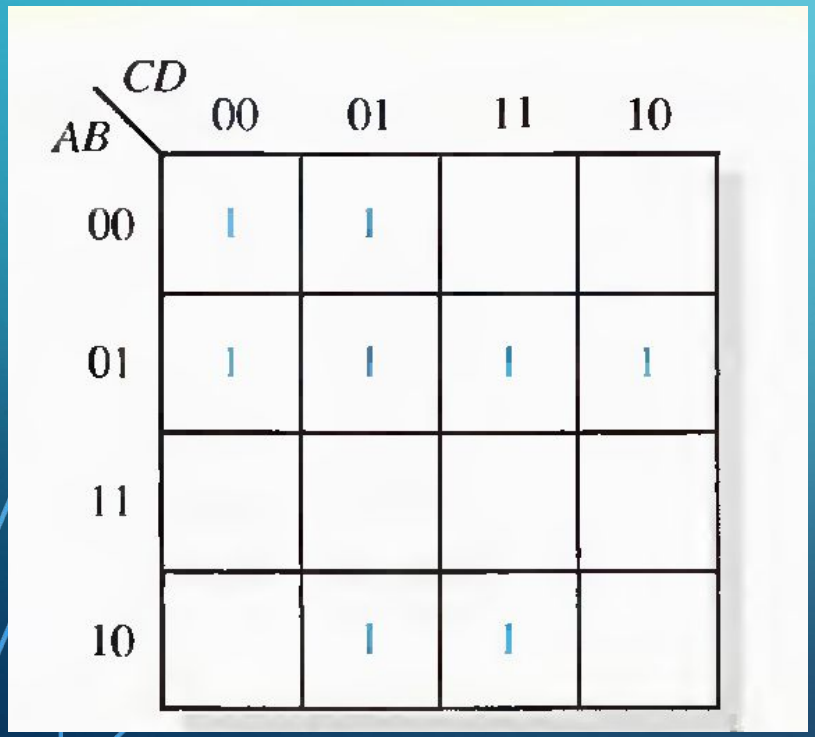


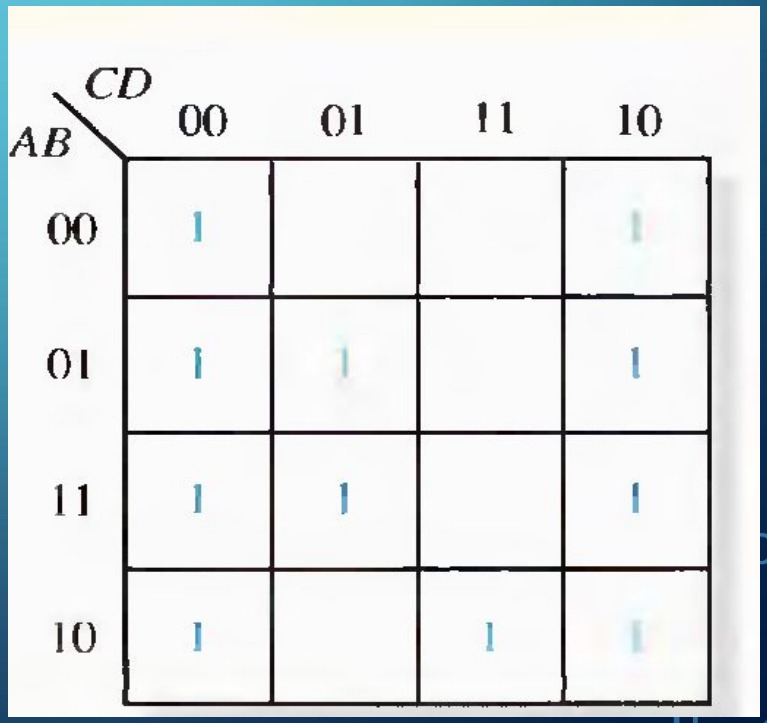
$$a = A + C + BD + \overline{B}\,\overline{D}$$

KARNAUGH MAPS (EXAMPLES)

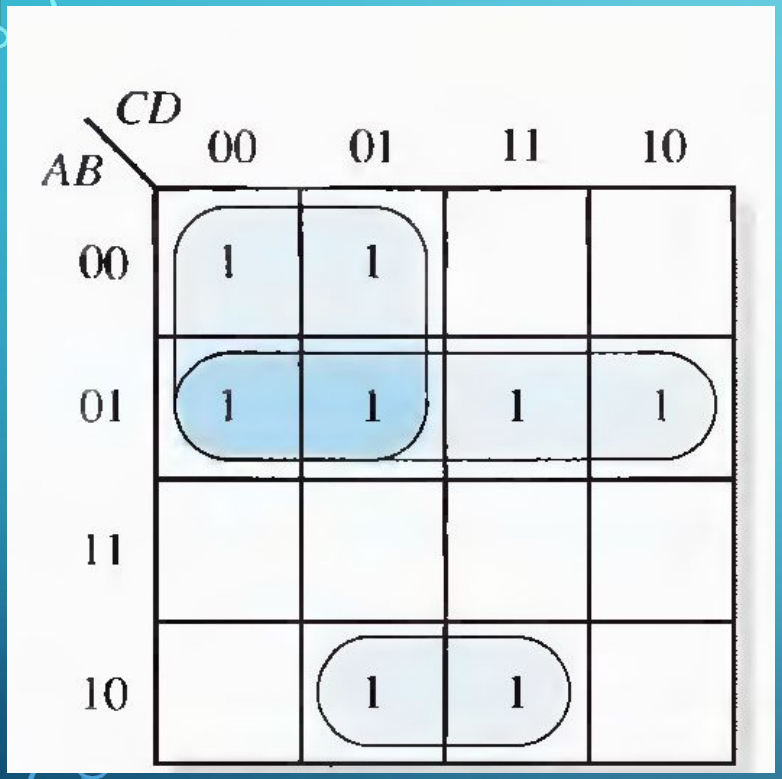
F(A,B,C,D)=Σm(0,1,4,5,6,7,9,11)

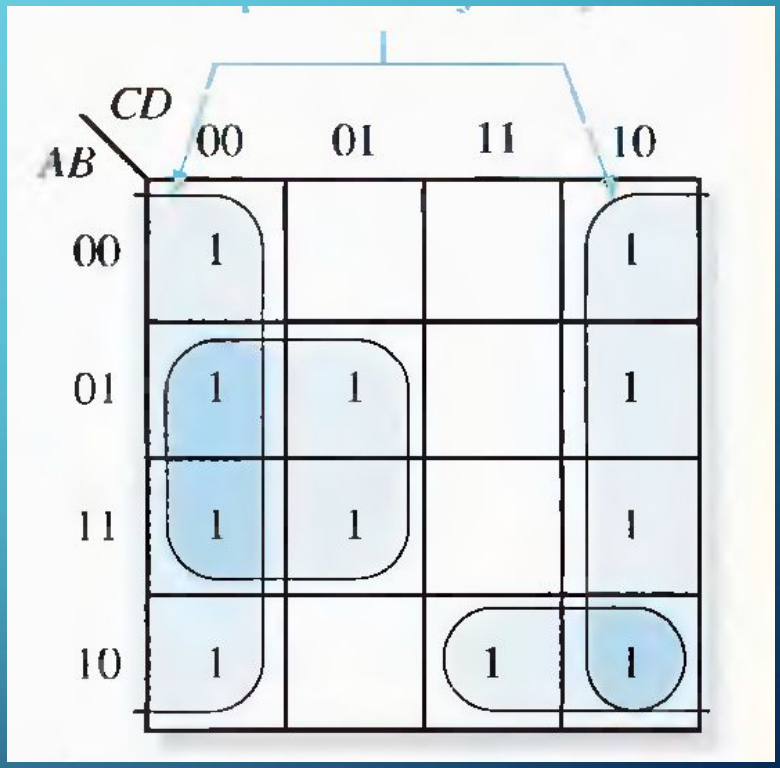
F(A,B,C,D)=Σm(0,2,4,5,6,8,10,11,12,13,14,)





Group the 1s in each of the Karnaugh maps





$X = A'B + A'C' + AB'D$

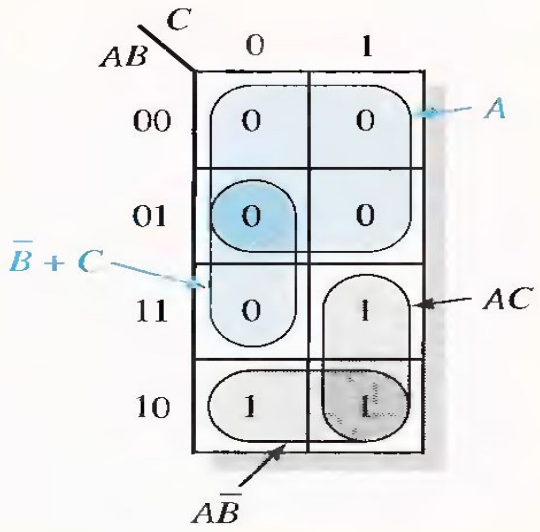
$X = D' + BC' + AB'C$

KARNAUGH MAP POS MINIMIZATION

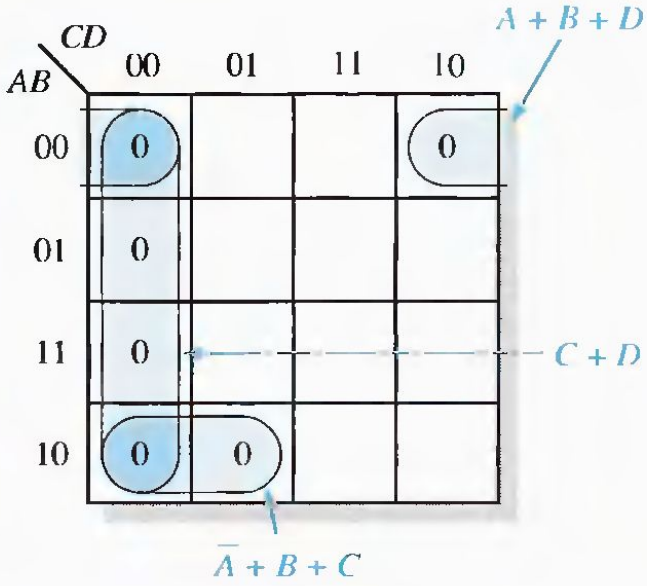
Use a Karnaugh map to minimize the following standard POS expression:

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

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

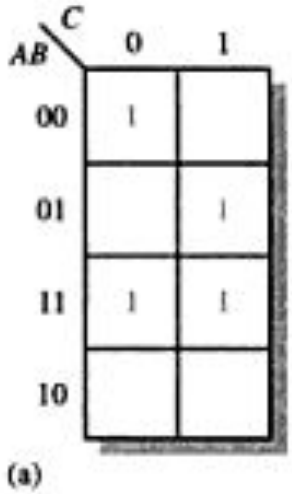


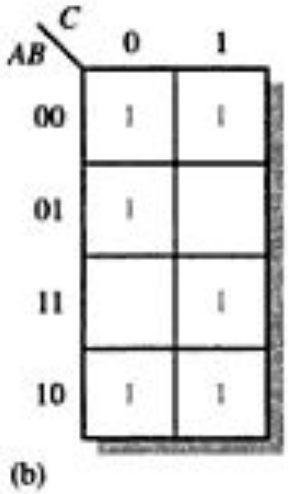
$$A(\overline{B} + C)$$

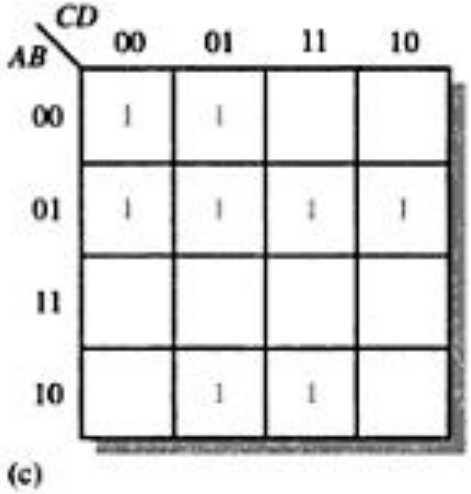


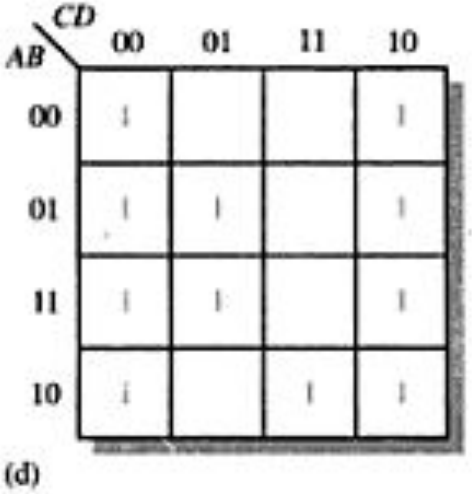
$$(C + D)\,(A + B + D)\,(\overline{A} + B + C)$$

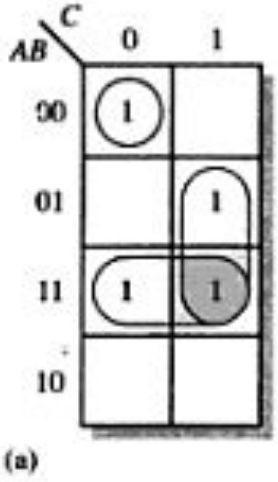
Group the 1s in each of the Karnaugh maps

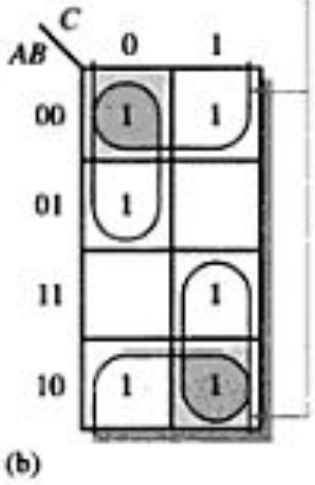


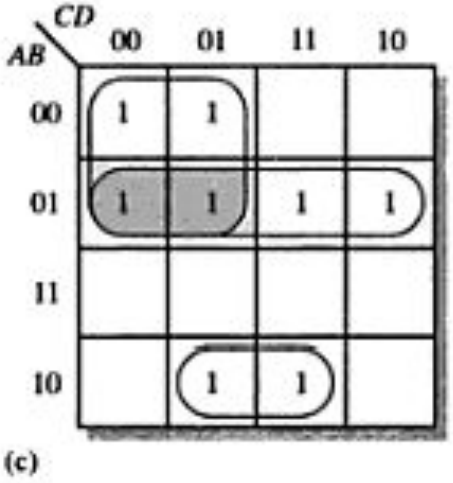


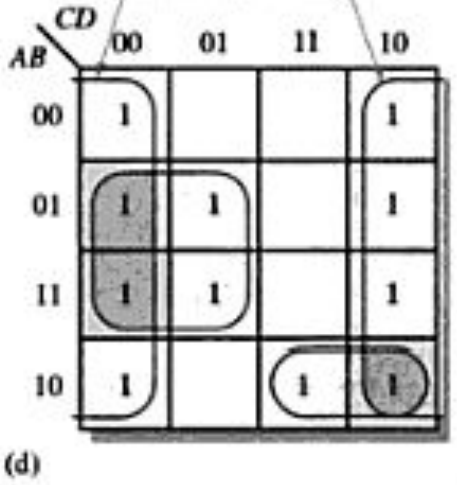


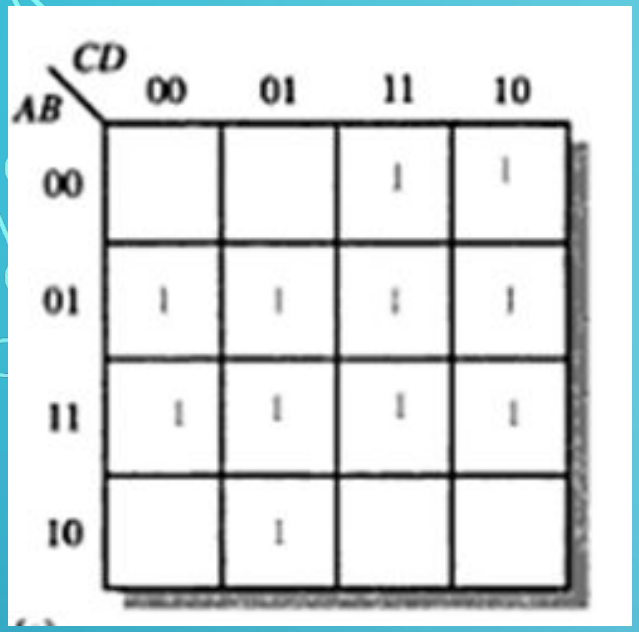


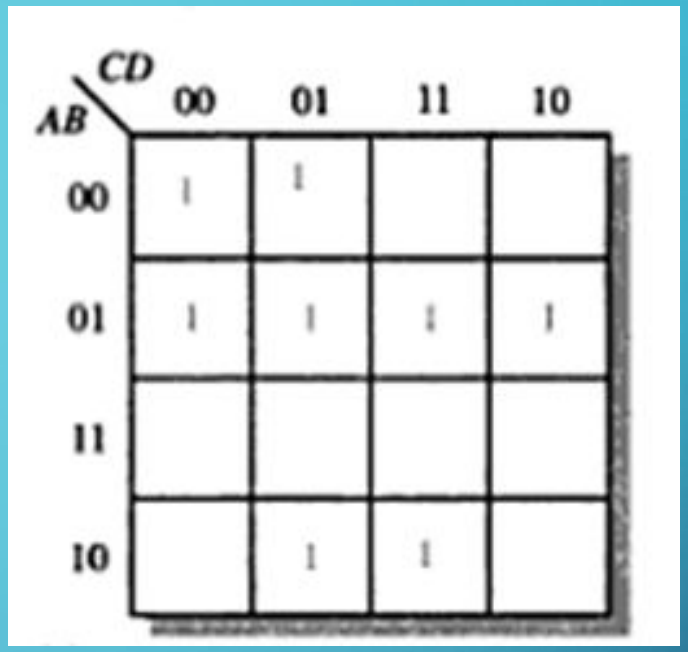


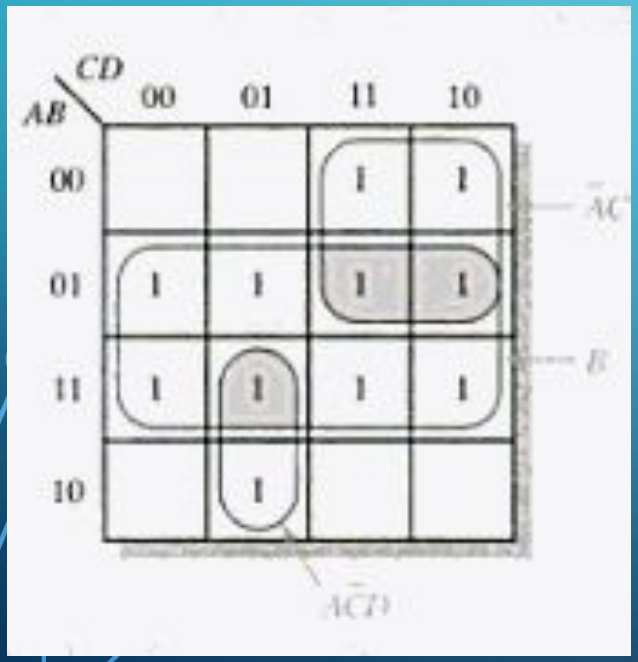




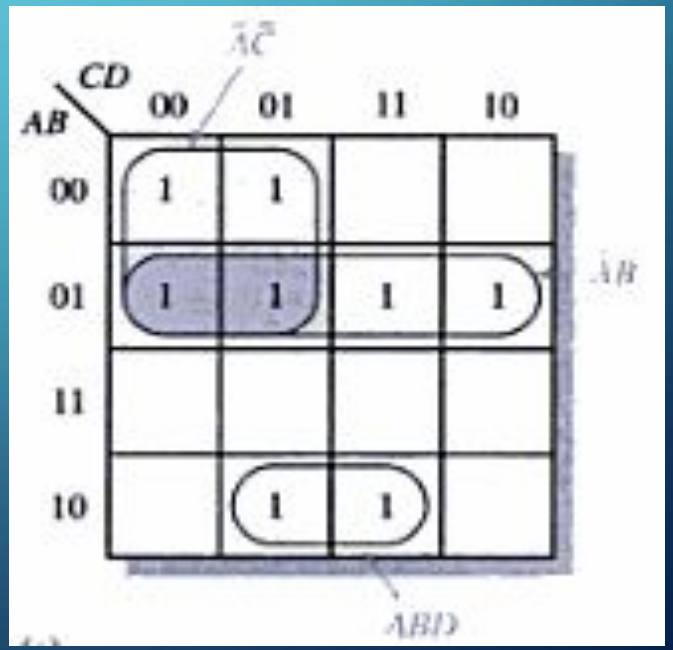








$B + \bar{A}C + A\bar{C}\bar{D}$



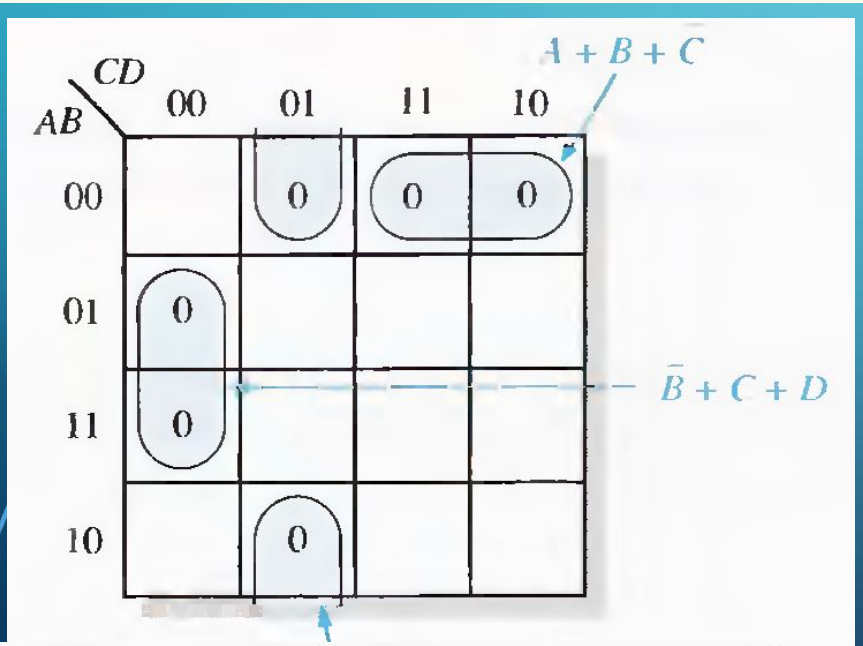
$\bar{A}B + \bar{A}\bar{C} + A\bar{B}\bar{D}$

Converting Between POS and SOP Using the Karnaugh Map

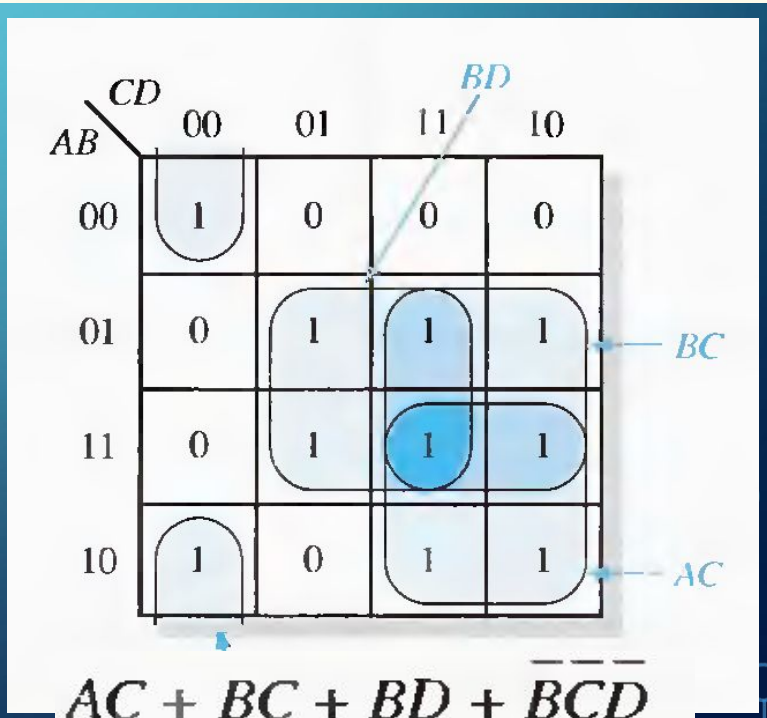
Using a Karnaugh map, convert the following standard POS expression into a minimum POS expression, a standard SOP expression, and a minimum SOP expression.

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

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



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



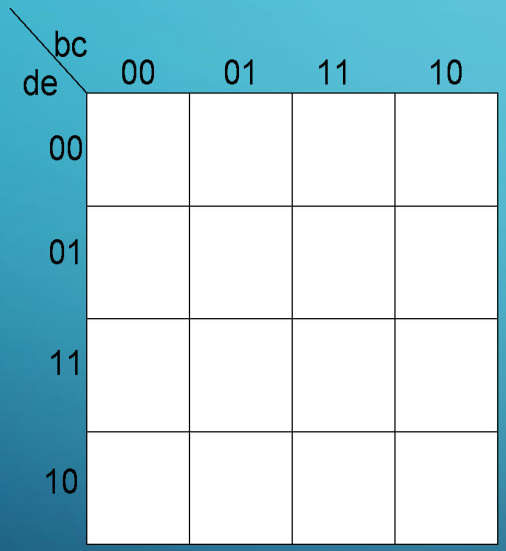
$$AC + BC + BD + \overline{B}\overline{C}\overline{D}$$

FIVE VARIABLE K-MAP

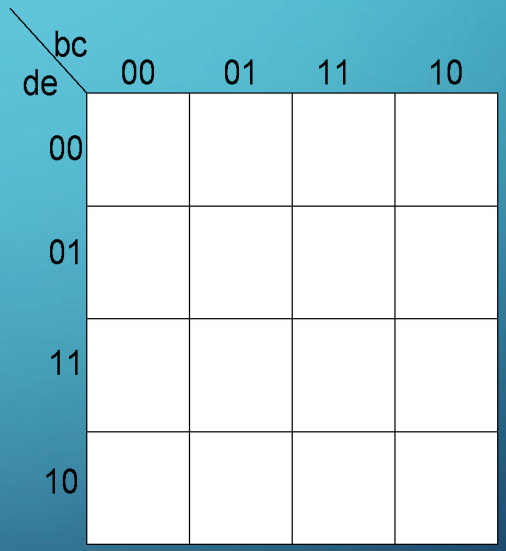
Use two four variable K-maps

USE TWO FOUR-VARIABLE K-MAPS

A=0 map



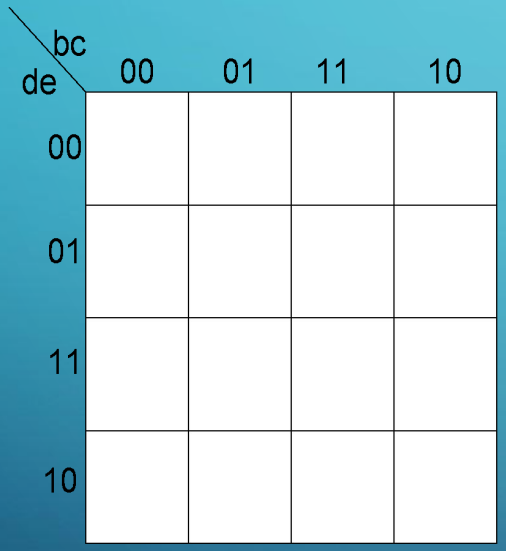
A=1 map



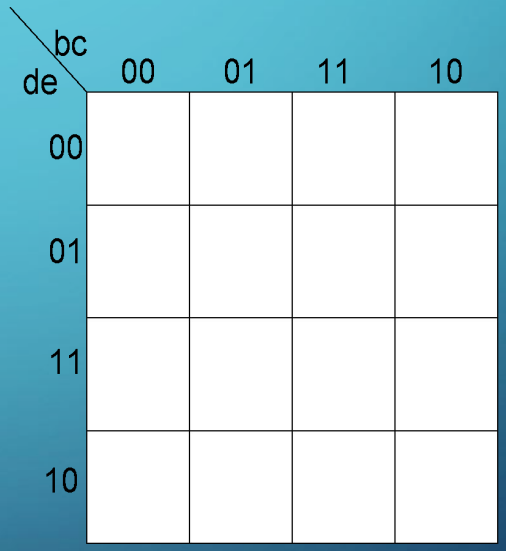
$$F(a,b,c,d,e) = \sum m(5,7,13,15,21,23,29,31)$$

USE TWO FOUR-VARIABLE K-MAPS

A=0 map



A=1 map



$$F(a,b,c,d,e) = \sum m(5,7,13,15,21,23,29,31)$$