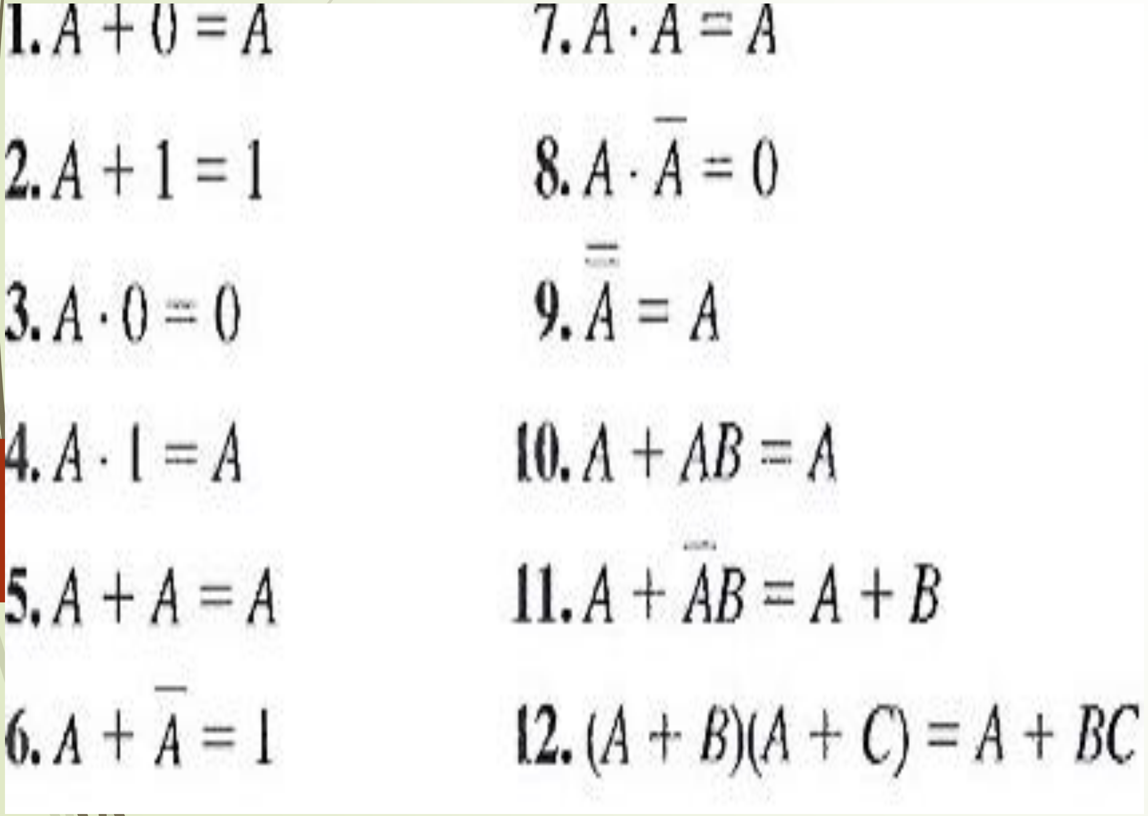
# Digital Electronics1 4th Sem (GE4T) for Honors students

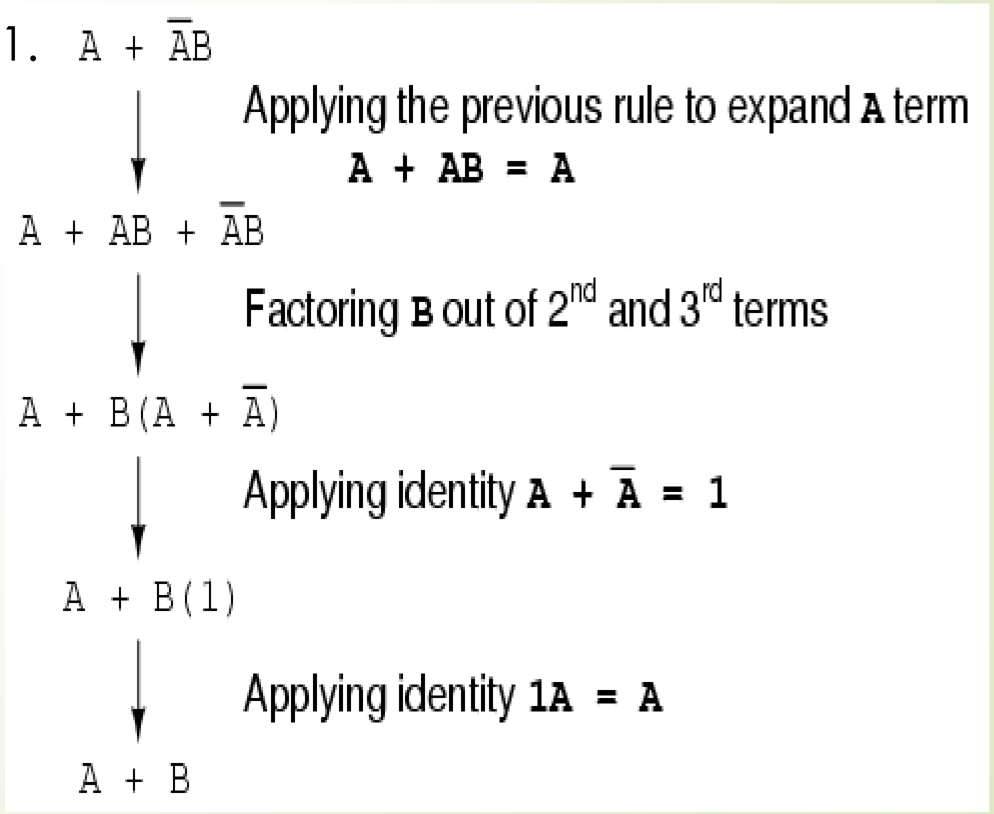
*Mahadeb Pal, Assistant Professor and HOD, Physics.*
SMHGC.

**Rules of Boolean Algebra** : 12 basic rules that are useful in manipulating and simplifying Boolean expressions. Rules 1 through 9 will be viewed in terms of their application to logic gates. Rules 10 through 12 will be derived in terms of the simpler rules and the laws .



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

10. $A + AB = A(1 + B) = A . I = A$



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

$\downarrow$ Applying the previous rule to expand **A** term **A + AB = A**

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

$\downarrow$ Factoring **B** out of 2nd and 3rd terms

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

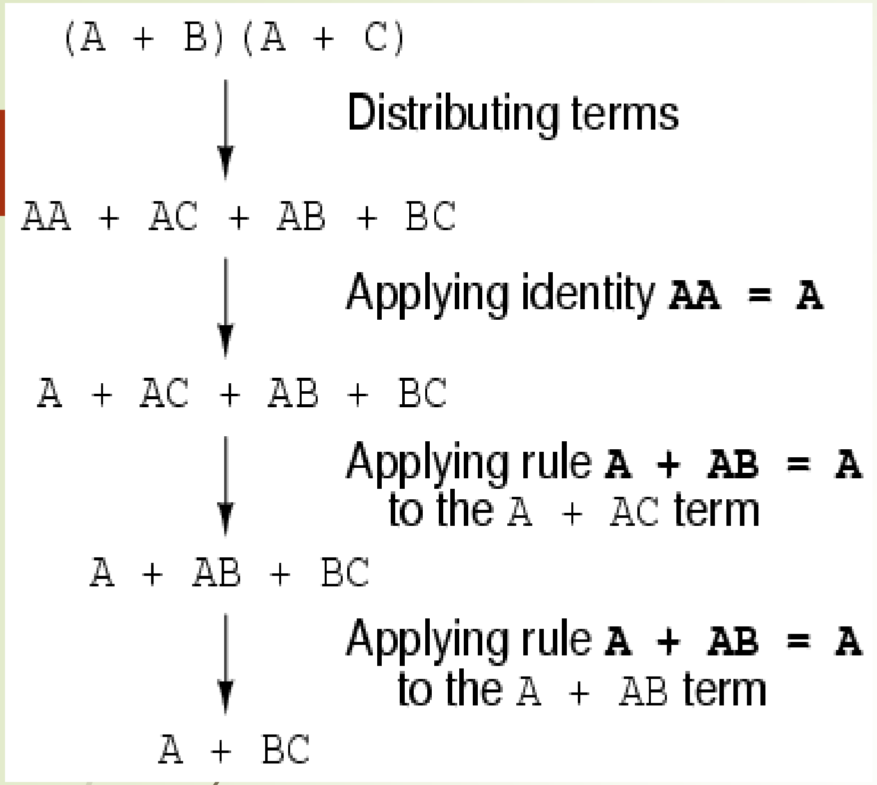
$\downarrow$ Applying identity $\mathbf{A + \overline{A} = 1}$

$A + B(1)$

$\downarrow$ Applying identity **1A = A**

$A + B$

12.



(A + B)(A + C)

↓ Distributing terms

AA + AC + AB + BC

↓ Applying identity **AA = A**

A + AC + AB + BC

↓ Applying rule **A + AB = A** to the A + AC term

A + AB + BC

↓ Applying rule **A + AB = A** to the A + AB term

A + BC

## DEMORGAN'S THEOREMS

One of DeMorgan's theorems is stated as follows:

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

Stated another way,

***The complement of two or more ANDed variables is equivalent to the OR of the complements of the individual variables.***

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.***

Stated another way,

***The complement of two or more ORed variables is equivalent to the AND of the complements of the individual variables,***

## SIMPLIFICATION USING BOOLEAN ALGEBRA

A simplified Boolean expression uses the fewest gates possible to implement a given expression.

Example

Using Boolean algebra techniques, simplify this expression: AB + A(B + C) + B(B + C)

Solution

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

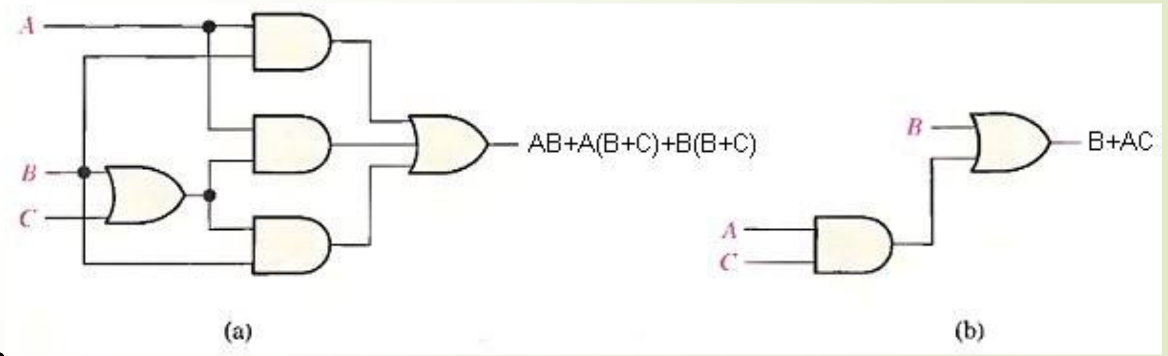
Step 4: Apply rule 10 (B + BC = B) to the last two terms.

AB+AC+BC+B=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.



Exercise

Simplify the Boolean expressions: 1- AB + A(B + C) + B(B + C).

2 [AB( C + BD) + A B]C

3 ABC + ABC + A B C + ABC + ABC

# Standard and Canonical Forms

***<u>STANDARD FORMS OF BOOLEAN EXPRESSIONS</u>***

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.

*<u>The Sum-of-Products (SOP) Form</u>*

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

$AB + ABC$

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

$AB + BCD + AC$

Also, an SOP expression can contain a single-variable term, as in

$A + AB\overline{C} + BC\overline{D}$.

**In an SOP expression a single overbar cannot extend over more than one variable.**

<u>Example</u>

Convert each of the following Boolean expressions to SOP form:

(a) $AB + B(CD + EF)$

(b) $(A + B)(B + C + D)$

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

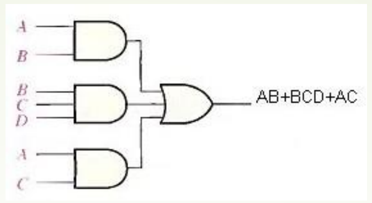


Fig.(4-18) Implementation of the SOP expression AB + BCD + AC.

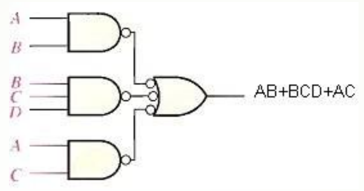


Fig.(4-19) This NAND/NAND implementation is equivalent to the AND/OR in figure above.

*__The Standard SOP Form__*

So far, you have seen SOP expressions in which some of the product terms do not contain all of the variables in the domain of the expression. For example, the expression $\overline{A}B\overline{C} + A\overline{B}D + AB\overline{C}\overline{D}$ has a domain made up of the variables A, B, C. and D. However, notice that the complete set of variables in the domain is not represented in the first two terms of the expression; that is, D or $\overline{D}$ is missing from the first term and C or $\overline{C}$ is missing from the second term.

A standard SOP expression is one in which all the variables in the domain appear in each product term in the expression. For example, $\overline{A}BC\overline{D} + AB\overline{C}D + A\overline{B}CD$ is a standard SOP expression.

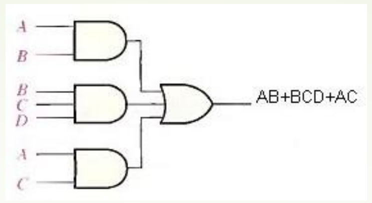


Fig.(4-18) Implementation of the SOP expression AB + BCD + AC.

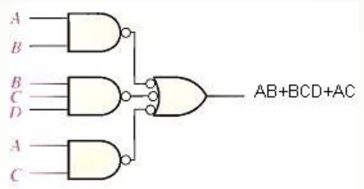


Fig.(4-19) This NAND/NAND implementation is equivalent to the AND/OR in figure above.

*The Standard SOP Form*

So far, you have seen SOP expressions in which some of the product terms do not contain all of the variables in the domain of the expression. For example, the expression $\overline{A}B\overline{C} + A\overline{B}D + AB\overline{C}\overline{D}$ has a domain made up of the variables A, B, C. and D. However, notice that the complete set of variables in the domain is not represented in the first two terms of the expression; that is, D or $\overline{D}$ is missing from the first term and C or $\overline{C}$ is missing from the second term.

A standard SOP expression is one in which all the variables in the domain appear in each product term in the expression. For example, $\overline{A}BC\overline{D} + AB\overline{C}D + A\overline{B}CD$ is a standard SOP expression.

*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 SOP to include all variables in the domain and their complements. As stated in the following steps, 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.

Step 1. Multiply each nonstandard product term by a term made up of the sum of a missing variable and its complement. 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.

Example

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 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:

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

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

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

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

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:

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

*The Product-of-Sums (POS) Form*

A sum term was defined before as a term consisting of the sum (Boolean addition) of literals (variables or their complements). When two or more sum terms are multiplied, the resulting expression is a product-of-sums (POS). Some examples are

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

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

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

A POS expression can contain a single-variable term, as in

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

In a POS expression, a single overbar cannot extend over more than one variable; however, more than one variable in a term can have an overbar. For example, a POS expression can have the term $\overline{A} + \overline{B} + \overline{C}$ but not $\overline{A + B + C}$.

Implementation of a POS Expression simply requires ANDing the outputs of two or more OR gates. A sum term is produced by an OR operation and the product of two or more sum terms is produced by an AND operation. Fig.(4-

20) shows for the expression (A + B)(B + C + D)(A + C). The output X of the AND gate equals the POS expression.

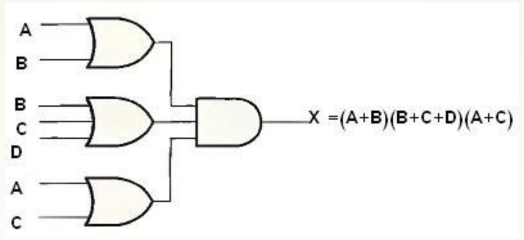


Fig.(4-20)

*The Standard POS Form*

So far, you have seen POS expressions in which some of the sum terms do not contain all of the variables in the domain of the expression. For example, the expression

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

has a domain made up of the variables A, B, C, and D. Notice that the complete set of variables in the domain is not represented in e first two terms of the expression; that is, D or $\overline{D}$ is missing from the first term and C or $\overline{C}$ is missing from the second term.

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

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

is a standard POS expression. Any nonstandard POS expression (referred to simply as POS) can be converted to the standard form using Boolean algebra.

*Converting a Sum Term to Standard POS*

Each sum term in a POS 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, a nonstandard POS expression is converted into standard form using Boolean algebra rule 8 ($A\overline{A} = 0$) from Table 4-1:

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 from Table 4-1: $A + BC = (A + B)(A + C)$

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

Example

Convert the following Boolean expression into standard POS form:

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

Solution

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

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

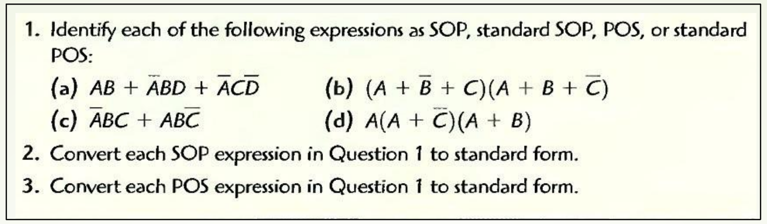
The second term, $\overline{B} + C + \overline{D}$, is missing variable A or $\overline{A}$, so add $A\overline{A}$ and apply rule 12 as follows:

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

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

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

Examples:-



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

***__CANONICAL FORMS OF BOOLEAN EXPRESSIONS__***

| | |
|---|---|
| With one variable | $x$ & $\bar{x}$. |
| With two variables | $\bar{x}\,\bar{y}$, $x\,\bar{y}$, $\bar{x}\,y$ and $x\,y$. |
| With three variables | $\bar{x}\,\bar{y}\,\bar{z}$, $\bar{x}\,\bar{y}\,z$, $\bar{x}\,y\,\bar{z}$, $\bar{x}\,y\,z$, $x\,\bar{y}\,\bar{z}$, $x\,\bar{y}\,z$, $x\,y\,\bar{z}$ & $x\,y\,z$. |

These eight AND terms are called minterms.

n variables can be combined to form $2^n$ minterms.

| x | y | z | minterm | designation | maxterm | designation |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | x y z | $m_0$ | x+y+z | |
| 0 | 0 | 1 | x y z | $m_1$ | $x+y+\bar{z}$ | |
| 0 | 1 | 0 | x y z | $m_2$ | $x+\bar{y}+z$ | |
| 0 | 1 | 1 | x y z | $m_3$ | $x+\bar{y}+\bar{z}$ | |
| 1 | 0 | 0 | x y z | $m_4$ | x+y+z | |
| 1 | 0 | 1 | x y z | $m_5$ | x+y+z | |
| 1 | 1 | 0 | x y z | $m_6$ | x+y+z | |

**Note that each maxterm is the complement of its corresponding minterm and vice versa.**

For example the function F

| x | y | z | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

$F = \bar{x}\,\bar{y}\,z + x\,\bar{y}\,\bar{z} + x\,y\,z$

$F = m_1 + m_4 + m_7$

Any Boolean function can be expressed as a sum of minterms (sum of products **SOP**) or product of maxterms (product of sums **POS**).

$\bar{F} = \bar{x}\,\bar{y}\,\bar{z} + \bar{x}\,y\,\bar{z} + \bar{x}\,y\,z + x\,\bar{y}\,z + x\,y\,\bar{z}$

The complement of $\bar{F} = \bar{\bar{F}} = F$

$F = (x + y + z)\,(x + \bar{y} + z)\,(x + \bar{y} + \bar{z})\,(\bar{x} + y + \bar{z})\,(\bar{x} + \bar{y} + z)$

$F = M_0\,M_2\,M_3\,M_5\,M_6$

Example

Express the Boolean function $F = A + \bar{B}C$ in a sum of minterms (SOP).

Solution

The term A is missing two variables because the domain of F is (A, B, C)

$A = A(B + \bar{B}) = AB + A\bar{B}$ because $B + \bar{B} = 1$

$\bar{B}C$ missing A, so

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

$AB(C + \bar{C}) = ABC + AB\bar{C}$

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

$F = ABC + AB\bar{C} + \underline{A\bar{B}C} + A\bar{B}\bar{C} + \underline{A\bar{B}C} + \bar{A}\bar{B}C$

$\boxed{\text{Because } A + A = A}$

$F = ABC + AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C} + \bar{A}\bar{B}C$

$F = m_7 + m_6 + m_5 + m_4 + m_1$

In short notation

$F(A, B, C) = \sum(1, 4, 5, 6, 7)$

$\bar{F}(A, B, C) = \sum(0, 2, 3)$

**The complement of a function expressed as the sum of minterms equal to the sum of minterms missing from the original function.**

Truth table for $F = A + \bar{B}C$

| | A | B | C | $\bar{B}$ | BC | F |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | 1 | 0 | 0 | 0 | 0 |
| 3 | 0 | 1 | 1 | 0 | 0 | 0 |
| 4 | 1 | 0 | 0 | 1 | 0 | 1 |
| 5 | 1 | 0 | 1 | 1 | 1 | 1 |
| 6 | 1 | 1 | 0 | 0 | 0 | 1 |
| 7 | 1 | 1 | 1 | 0 | 0 | 1 |

Example

Express $F = xy + \overline{x}z$ in a product of maxterms form.

Solution

$F = xy + \overline{x}z = (xy + \overline{x})(xy + z) = (x + \overline{x})(y + \overline{x})(x + z)(y + z)$

remember $x + \overline{x} = 1$

$F = (y + \overline{x})(x + z)(y + z)$

$F = (\overline{x} + y + z\overline{z})(x + y\overline{y} + z\ )(x\overline{x} + y + z)$

$F = \underline{\underline{(\overline{x} + y + z)}}(\overline{x} + y + \overline{z})\underline{(x + y + z)}(x + \overline{y} + z)\underline{(x + y + z)}\underline{\underline{(\overline{x} + y + z)}}$

$F = (\overline{x} + y + z)(\overline{x} + y + \overline{z})(x + y + z)(x + \overline{y} + z)$

$F = M_4 M_5 M_0 M_2$

$F(x, y, z) = \prod(0, 2, 4, 5)$

$\overline{F}(x, y, z) = \prod(1, 3, 6, 7)$

**The complement of a function expressed as the product of maxterms equal to the product of maxterms missing from the original function.**

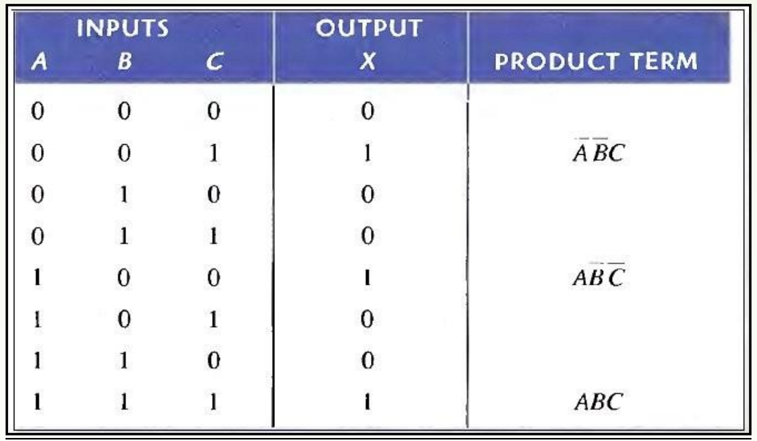
To convert from one canonical form to another, interchange the symbols $\sum$, $\prod$ and list those numbers missing from the original form.

$F = M_4 M_5 M_0 M_2 = m_1 + m_3 + m_6 + m_7$

$F(x, y, z) = \prod(0, 2, 4, 5) = \sum(1, 3, 6, 7)$

Example

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



| INPUTS A | B | C | OUTPUT 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*

Reca11 that a POS expression is equal to 0 only if at least one of the sum terms is equal to 0. To construct a truth table from a POS expression, list all the possible combinations of binary values of the variables just as was done for the SOP expression. Next, convert the POS expression to standard form if it is not already. Finally, place a 0 in the output column (X) for each binary value that makes the expression a 0 and place a 1 for all the remaining binary values. This procedure is illustrated in Example below:

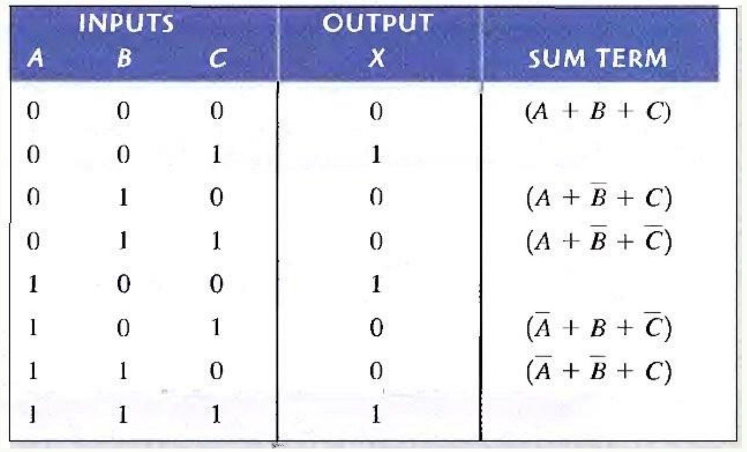
Example

Determine the truth table for the following standard POS expression:

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

Solution

There are three variables in the domain and the eight possible binary values are listed in the left three columns of. The binary values that make the sum terms in the expression equal to 0 are $A+ B + C$: 000; $A + \overline{B} + C$: 010: $A + \overline{B} + \overline{C}$: 011; $\overline{A} + B + \overline{C}$: 10l; and $\overline{A} + \overline{B} + C$: 110. For each of these binary values, place a 0 in the output column as shown in the table. For each of the remaining binary combinations, place a 1 in the output column.



| 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 | |

## 5 KARNAUGH MAP MINIMIZATION

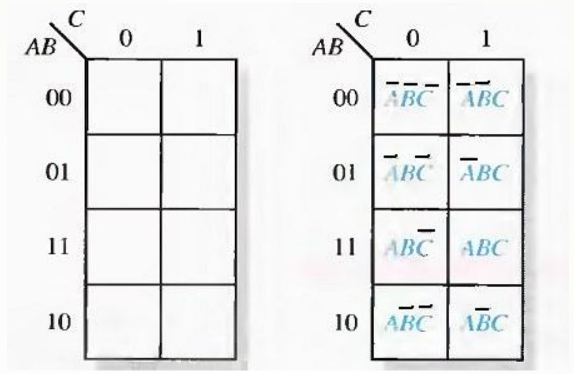
A Karnaugh map provides a systematic method for simplifying Boolean expressions and, if properly used, will produce the simplest SOP or POS expression possible, known as the minimum expression. As you have seen, the effectiveness of algebraic simplification depends on your familiarity with all the laws, 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.

A Karnaugh map is similar to a truth table because it presents all of the possible values of input variables and the resulting output for each value. Instead of being organized into columns and rows like a truth table, the Karnaugh map is an array of cells in which each cell represents a binary value of the input variables. The cells are arranged in a way so that simplification of a given expression is simply a matter of properly grouping the cells. Karnaugh maps can be used for expressions with two, three, four. and five variables. Another method, called the Quine-McClusky method can be used for higher numbers of variables.

The number of cells in a Karnaugh map is equal to the total number of possible input variable combinations as is the number of rows in a truth table. For three variables, the number of cells is $2^3 = 8$. For four variables, the number of cells is $2^4 = 16$.

### ***The 3-Variable Karnaugh Map***

The 3-variable Karnaugh map is an array of eight cells. as shown in Fig.(5-1)(a). In this case, A, B, and C are used for the variables although other letters could be used. Binary values of A and B are along the left side (notice the sequence) and the values of C are across the top. The value of a given cell is the binary values of A and B at the left in the same row combined with the value of C at the top in the same column. For example, the cell in the upper left corner has a binary value of 000 and the cell in the lower right corner has a binary value of 101. Fig.(5-1)( b) shows the standard product terms that are represented by each cell in the Karnaugh map.



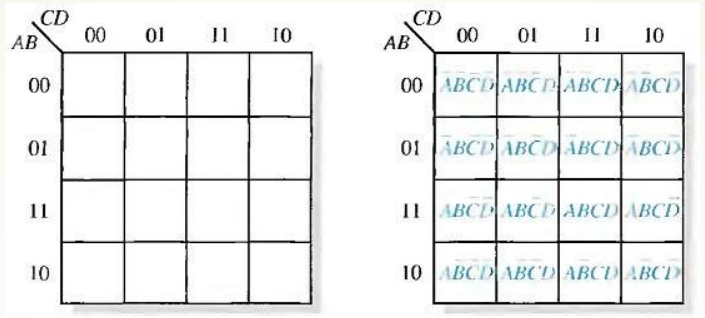
C
AB
0
1
00
01
11
10
$\overline{A}\overline{B}\overline{C}$
$\overline{A}\overline{B}C$
$\overline{A}B\overline{C}$
$\overline{A}BC$
$AB\overline{C}$
$ABC$
$A\overline{B}\overline{C}$
$A\overline{B}C$

(a) (b)

Fig.(5-1) A 3-variable Karnaugh map showing product terms.

***<u>The 4-Variable Karnaugh Map</u>***

The 4-variable Karnaugh map is an array of sixteen cells, as shown in Fig.(5-2)(a). Binary values of A and B are along the left side and the values of C and D are across the top. The value of a given cell is the binary values of A and B at the left in the same row combined with the binary values of C and D at the top in the same column. For example, the cell in the upper right corner has a binary value of 0010 and the cell in the lower right corner has a binary value of 1010. Fig.(5-2)(b) shows the standard product terms that are represented by each cell in the 4-variable Karnaugh map.



(a) (b)

Fig.(5-2) A 4-variable Karnaugh map.

*Cell Adjacency*

The cells in a Karnaugh map are arranged so that there is only a single-variable change between adjacent cells. Adjacency is defined by a single-variable change. In the 3-variable map the 010 cell is adjacent to the 000 cell, the 011 cell, and the 110 cell. The 010 cell is not adjacent to the 001 cell, the 111 cell, the 100 cell, or the 101 cell.

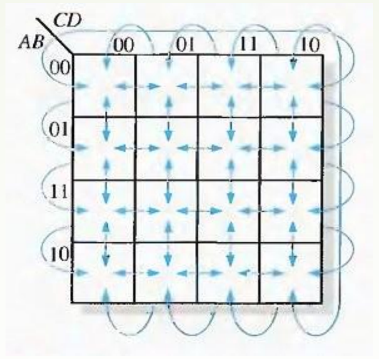


Fig.(5-3) Adjacent cells on a Karnaugh map are those that differ by only one variable. Arrows point between adjacent cells.

***KARNAUGH MAP SOP MINIMIZATION***

For an SOP expression in standard form, a 1 is placed on the Karnaugh map for each product term in the expression. Each 1 is placed in a cell corresponding to the value of a product term. For example, for the product term ABC, a 1 goes in the 10l cell on a 3-variable map.

Example

Map the following standard SOP expression on a Karnaugh map:

see Fig.(5-4).

Example

Map the following standard SOP expression on a Karnaugh map:

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

See Fig.(5-5).

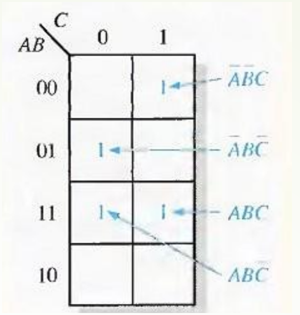


Fig.(5-4)

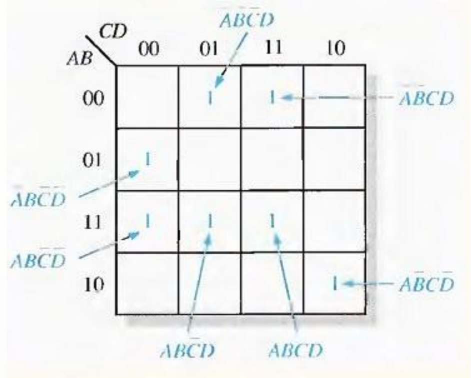

CD
AB
00
01
11
10
1
$\overline{A}\,\overline{B}\,\overline{C}D$
$\overline{A}\,\overline{B}CD$
$\overline{A}B\overline{C}\,\overline{D}$
$AB\overline{C}\,\overline{D}$
$AB\overline{C}D$
$ABCD$
$A\overline{B}C\overline{D}$

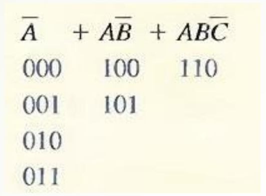
Fig.(5-5)

<u>Example</u>

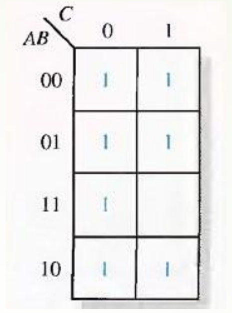
Map the following SOP expression on a Karnaugh map: $\overline{A} + A\overline{B} + AB\overline{C}.$

<u>Solution</u>

The SOP expression is obviously not in standard form because each product term does not have three variables. The first term is missing two variables, the second term is missing one variable, and the third term is standard. First expand the terms numerically as follows:



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



| $AB \backslash C$ | 0 | 1 |
|---|---|---|
| 00 | 1 | 1 |
| 01 | 1 | 1 |
| 11 | 1 | |
| 10 | 1 | 1 |

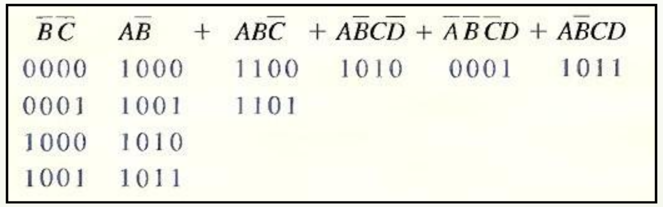
Example

Map the following SOP expression on a Karnaugh map:

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

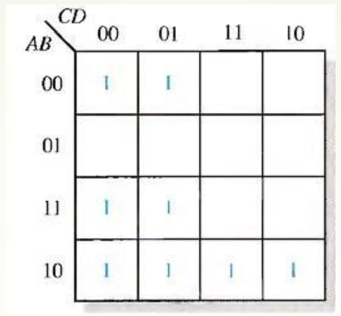
Solution

The SOP expression is obviously not in standard form because each product term does not have four variables.



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

Map each of the resulting binary values by placing a 1 in the appropriate cell of the 4- variable Karnaugh map.



***Karnaugh Map Simplification of SOP Expressions***

Grouping the 1s, you can group 1s on the Karnaugh map according to the following rules by enclosing those adjacent cells containing 1s. The goal is to maximize the size of the

groups and to minimize the number of groups.

- 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.
- Each cell in a group must be adjacent to one or more cells in that same group.
- Always include the largest possible number of 1s in a group in accordance with rule 1.
- 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 noncommon 1s.

Example:

Group the 1s in each of the Karnaugh maps in Fig.(5-6).

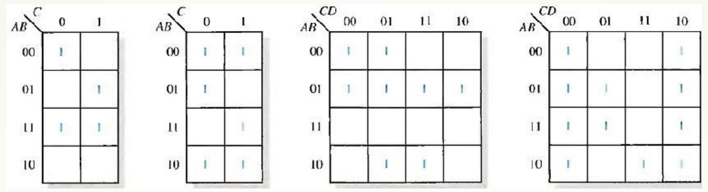


Fig.(5-6)

Solution:

The groupings are shown in Fig.(5-7). In some cases, there may be more than one way to group the 1s to form maximum groupings.

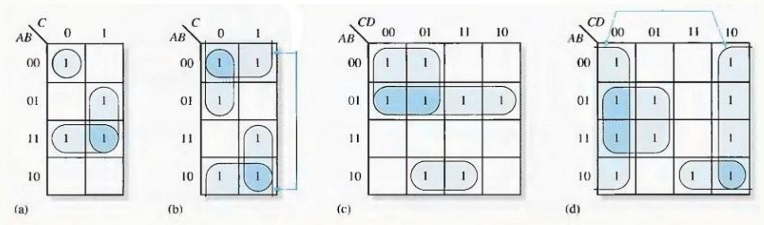


Fig.(5-7)

Determine the minimum product term for each group.

a. For a 3-variable map:

(1) A l-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

Example:

Determine the product terms for each of the Karnaugh maps in Fig.(5-7) and write the resulting minimum SOP expression.

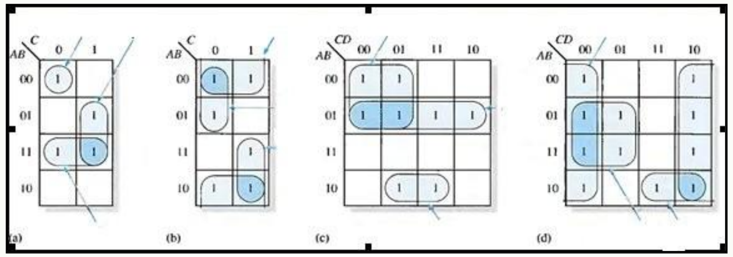


Fig.(5-8)

Solution:

The resulting minimum product term for each group is shown in Fig.(5-8). The minimum SOP expressions for each of the Karnaugh maps in the figure are:

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

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

Example: Use a Karnaugh map to minimize the following standard SOP expression:

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

Example: Use a Karnaugh map to minimize the following SOP expression:

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

**"Don't Care" Conditions**

Sometimes a situation arises in which some input variable combinations are not allowed. For example, recall that in the BCD code there are six invalid combinations: 1010, 1011, 1100, 1101, 1110, and 1111. Since these unallowed 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. Fig.(5-9) shows that for each "don't care" term, an X is placed in the cell. When grouping the 1 s, 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.

The truth table in Fig.(5-9)(a) describes a logic function that has a 1 output only when the BCD code for 7,8, or 9 is present on the inputs. If the "don't cares" are used as 1s, the resulting expression for the function is A + BCD, as indicated in part (b). If the "don't cares" are not used as 1s, the resulting