ENGG2020 Digital Logic and Systems

Chapter 3: Gate-Level Minimization

The Chinese University of Hong Kong

Canonical and Standard Forms

Minterms: Terms with all variables present, combined with **AND**

For n variables combined with AND, there are 2^n combinations. Each Unique combination is called a minterm.

Maxterms: Terms with all variables present, combined with **OR**

For n variables combined with OR, there are 2^n combinations. Each unique combination is called a maxterm.

Canonical and Standard Forms

Standard Order: Given n variables, we use an n-bit expansion of the index, i , to indicate normal (true) or complement states for the variables.

Minterms: "1" => true; " 0" => complemented.
m_0 (minterm 0): $\bar{x} \cdot \bar{y} \cdot \bar{z}$
m_3 (minterm 3): $\bar{x} \cdot y \cdot z$

Maxterms: "0" => true; "1" => complemented.
M0 (maxterm 0): $x + y + z$
M1 (maxterm 1): $x + \bar{y} + \bar{z}$

m_i is the complemented to M_i

Canonical and Standard Forms

Table 2.3
Minterms and Maxterms for Three Binary Variables

			Minterms		Maxterms	
x	**y**	**z**	**Term**	**Designation**	**Term**	**Designation**
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

Canonical and Standard Forms

Any Boolean function can be expressed as a sum of minterms.

Example: $F = A + \overline{B}C$

Expand the terms with missing variables and collect terms

$$
\begin{aligned}
F &= A + \overline{B}C \\
&= A(B+\overline{B})(C+\overline{C}) + (A+\overline{A})\,\overline{B}C \\
&= ABC + AB\overline{C} + A\overline{B}C + A\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,C
\end{aligned}
$$

Sum of minterms: $F = m_7 + m_6 + m_5 + m_4 + m_1$

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

The complement of a function equals to the sum of those minterms not included in the original function.

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

$$\textit{same as} \quad \overline{F}(A,B,C) = \sum m(0,2,3)$$

Canonical and Standard Forms

Any Boolean function can be expressed as a product of maxterms.

Example:

$$
\begin{aligned}
G &= AB + \overline{A}\,\overline{B} \\
&= (\overline{A} + AB)(\overline{B} + AB) \\
&= (\overline{A} + A)(\overline{A} + B)\ (\overline{B} + A)\ (\overline{B} + B) \\
&= 1 \cdot\ (\overline{A} + B)\ (\overline{B} + A)\ \cdot 1 \\
&= (\overline{A} + B)\ (\overline{B} + A) \\
&= M_2 \cdot M_1
\end{aligned}
$$

$$G(A,B) = \prod\ M(1,2)$$

Product of maxterms:

The complement of a function equals to the product of those maxterms not included in the original function.

$$G(A,B) = \prod M(1,2) = \sum\ m(0,3)$$

$$\textit{same as}\quad \overline{G}(A,B) = \prod M(0,3) = \sum\ m(1,2)$$

Canonical and Standard Forms

Standard Sum-of-Products(SOP) forms:
Equations are written as AND terms summed with OR operators.

$$SOPs: \quad xyz + \overline{x}y\overline{z} + \overline{y}, \quad A\overline{B} + \overline{A}B$$

Standard Product-of-Sums (POS) forms:
Equations are written as OR terms ANDed together.

$$POSs: \quad (x+y+z)(\overline{x}+y+\overline{z})(\overline{y}), \quad (A+\overline{B})(\overline{A}+B)$$

Canonical forms
Sum-of-Minterms or Product-of-Maxterms have one and only one representation.
Examples

Sum-of-minterms: $\quad xyz + \overline{x}y\overline{z} + x\overline{y}z, \quad A\overline{B} + \overline{A}B$

Production-of-maxterm: $\quad (x+y+z)(\overline{x}+y+\overline{z})(x+\overline{y}+z), \quad (A+\overline{B})(\overline{A}+B)$

Mixed forms: not SOP or POS. $(x\overline{y}+z)(\overline{x}+y), \quad AB\overline{C} + C(\overline{A}+B)$

Canonical and Standard Forms

Questions:

Is there only one minimum cost network?

How can we obtain <u>a </u>or <u>the</u> minimum literal expression?

Minimize

$$F(A,B,C) = \sum(0,2,3,4,5,7)$$

$$\begin{aligned}
F &= \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C + A\overline{B}\,\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + ABC \\
&= \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + ABC + A\overline{B}\,\overline{C} + A\overline{B}C \\
&= \overline{A}\,\overline{C}(B + \overline{B}) + BC(A + \overline{A}) + A\overline{B}(\overline{C} + C) \\
&= \overline{A}\,\overline{C} + BC + A\overline{B}
\end{aligned}$$

Pairing F’s terms differently

$$\begin{aligned}
F &= \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C + A\overline{B}\,\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + ABC \\
&= \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + A\overline{B}C + ABC \\
&= \overline{B}\,\overline{C}(\overline{A} + A) + \overline{A}B(C + \overline{C}) + AC(B + \overline{B}) \\
&= \overline{B}\,\overline{C} + \overline{A}B + AC
\end{aligned}$$

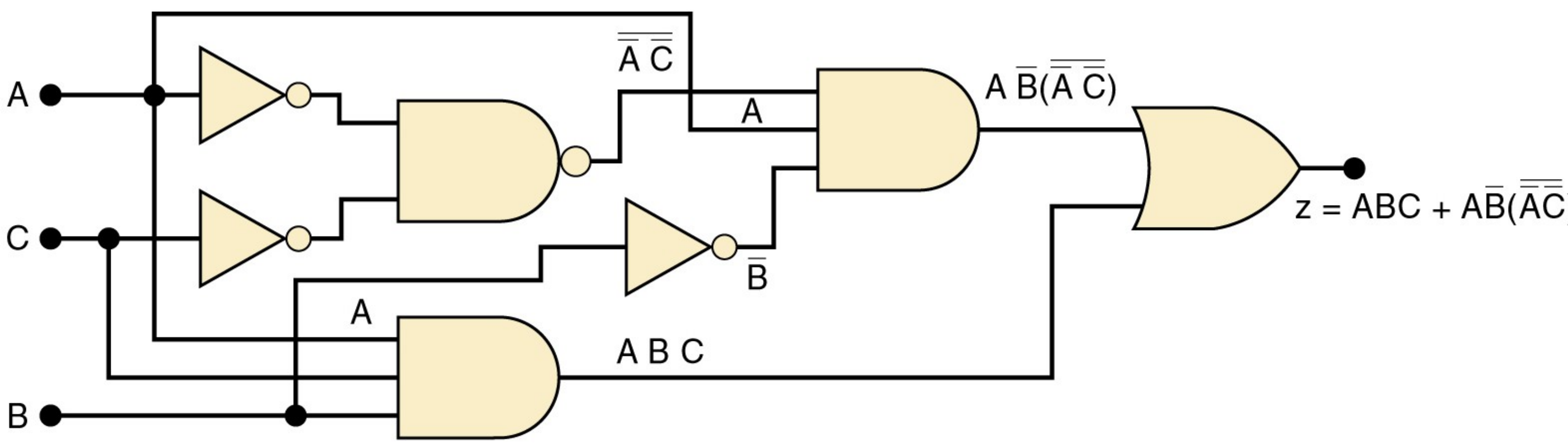
Both have the same numbers of literals and terms!

Simplifying the Logic Design

Simplify the logic circuit shown.

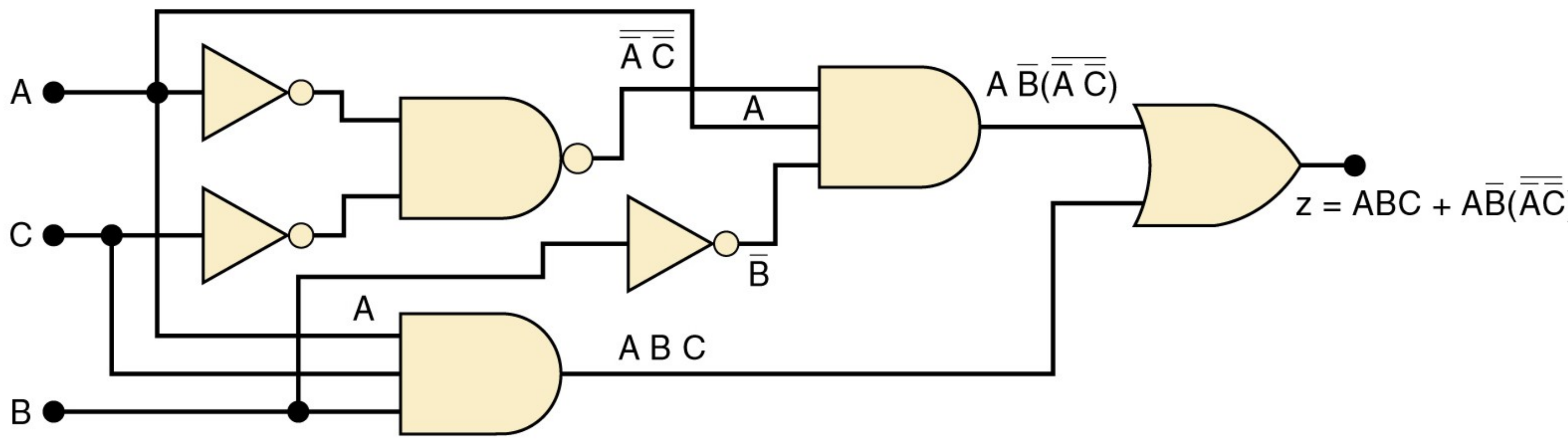
The first step is to determine the expression for the output:

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



$$
\begin{aligned}
z &= ABC + A\overline{B}(\overline{\overline{A}} + \overline{\overline{C}}) && \text{[theorem (17)]} \\
&= ABC + A\overline{B}(A + C) && \text{[cancel double inversions]} \\
&= ABC + A\overline{B}A + A\overline{B}C && \text{[multiply out]} \\
&= ABC + A\overline{B} + A\overline{B}C && [A \cdot A = A]
\end{aligned}
$$

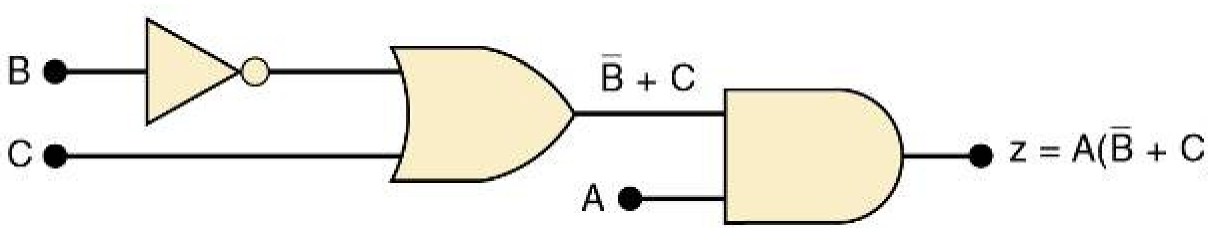
Simplifying the Logic Design



Factoring—the first & third terms above have ***AC*** in common, which can be factored out:

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

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



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

Simplifying the Logic Design

To solve any logic design problem:

Interpret the problem and set up its truth table.

Write the **AND** (product) term for each case where output = 1.

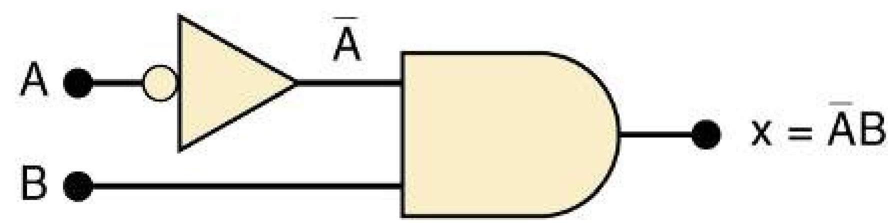
Combine the terms in SOP form.

Simplify the output expression if possible.

Implement the circuit for the final, simplified expression.

Circuit that produces a 1 output only for the A = 0, B = 1 condition.

A	B	x
0	0	0
0	1	1
1	0	0
1	1	0



Simplifying the Logic Design

Design a logic circuit with three inputs, A, B, and C. Output to be HIGH only when a majority inputs are HIGH.

AND terms for each case where output is 1.

Truth table.

A	*B*	*C*	*x*	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\rightarrow \bar{A}BC$
1	0	0	0	
1	0	1	1	$\rightarrow A\bar{B}C$
1	1	0	1	$\rightarrow AB\bar{C}$
1	1	1	1	$\rightarrow ABC$

SOP expression for the output:

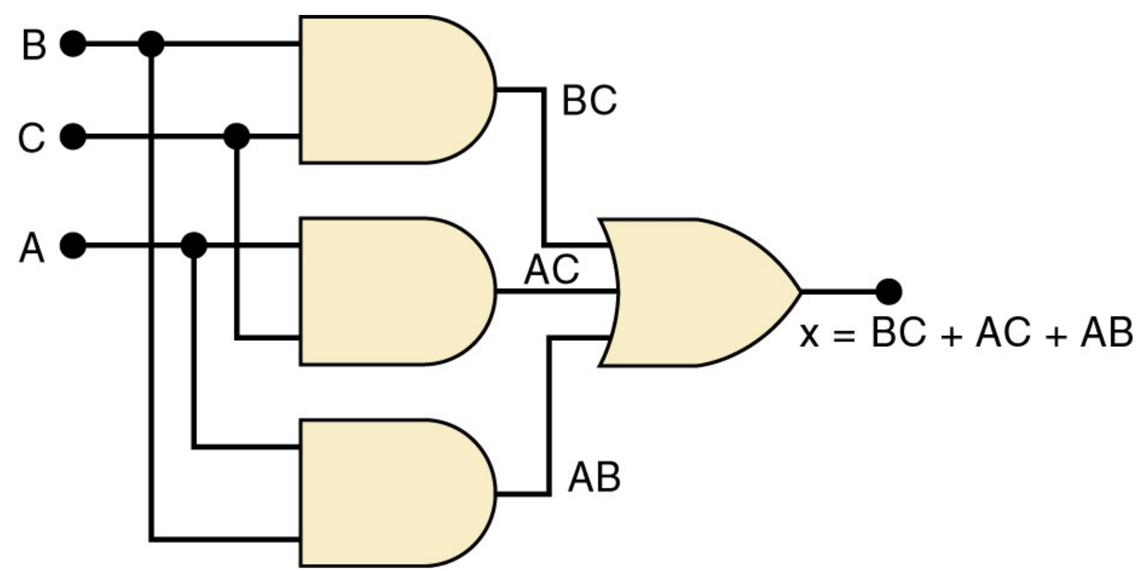
$$x = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

Simplifying the Logic Design

Design a logic circuit with three inputs, A, B, and C. Output to be HIGH only when a majority inputs are HIGH.

Simplified output expression:

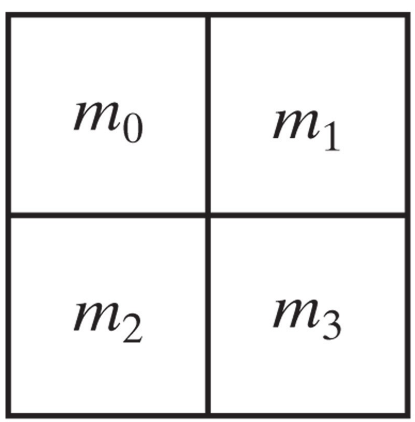
$$x = BC + AC + AB$$



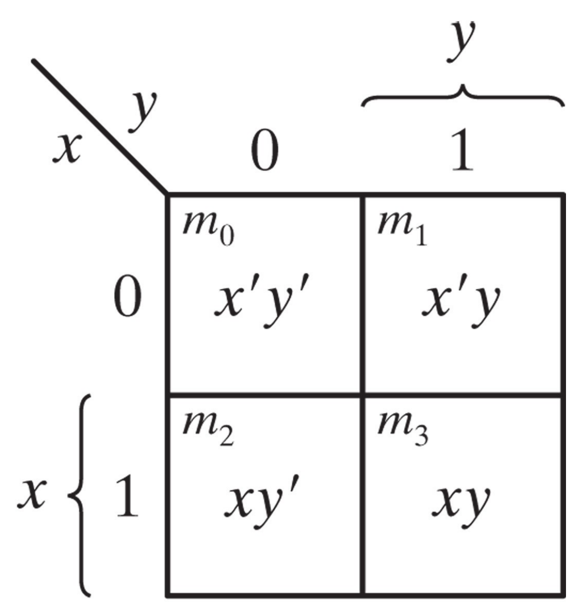
Karnaugh Map Method

A graphical method of simplifying logic equations or truth tables—also called a **K map**.

Theoretically can be used for any number of input variables—practically limited to 5 or 6 variables.



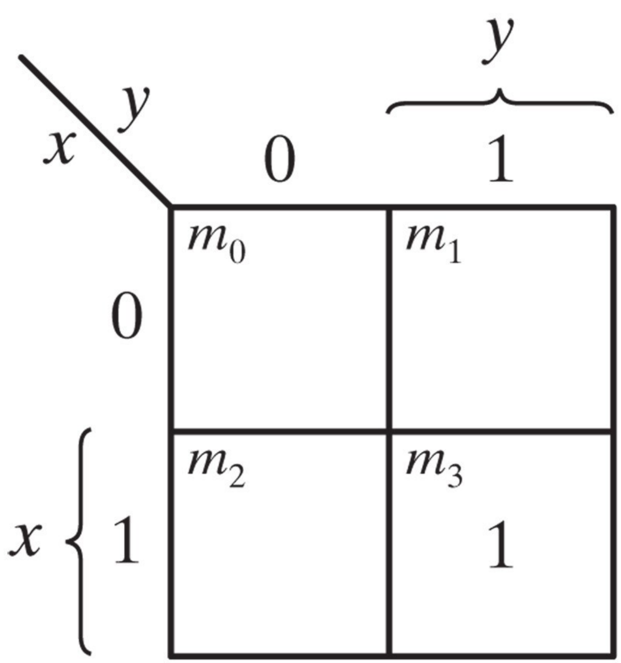
(a)



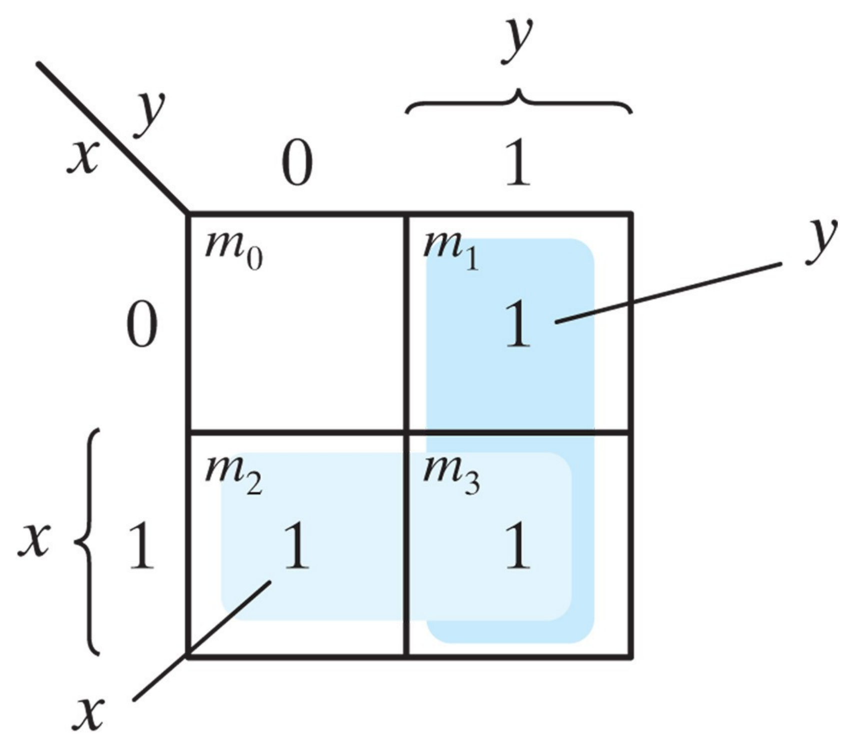
(b)

Karnaugh Map Method

Examples



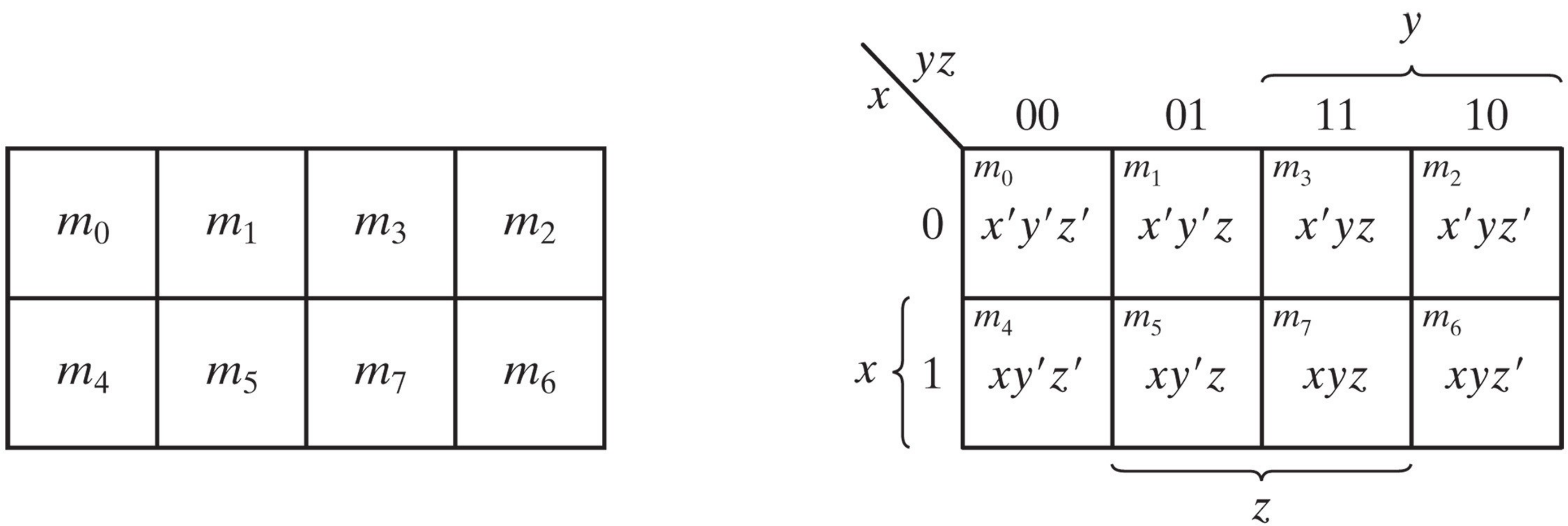
(a) xy



(b) $x + y$

Karnaugh Map Method

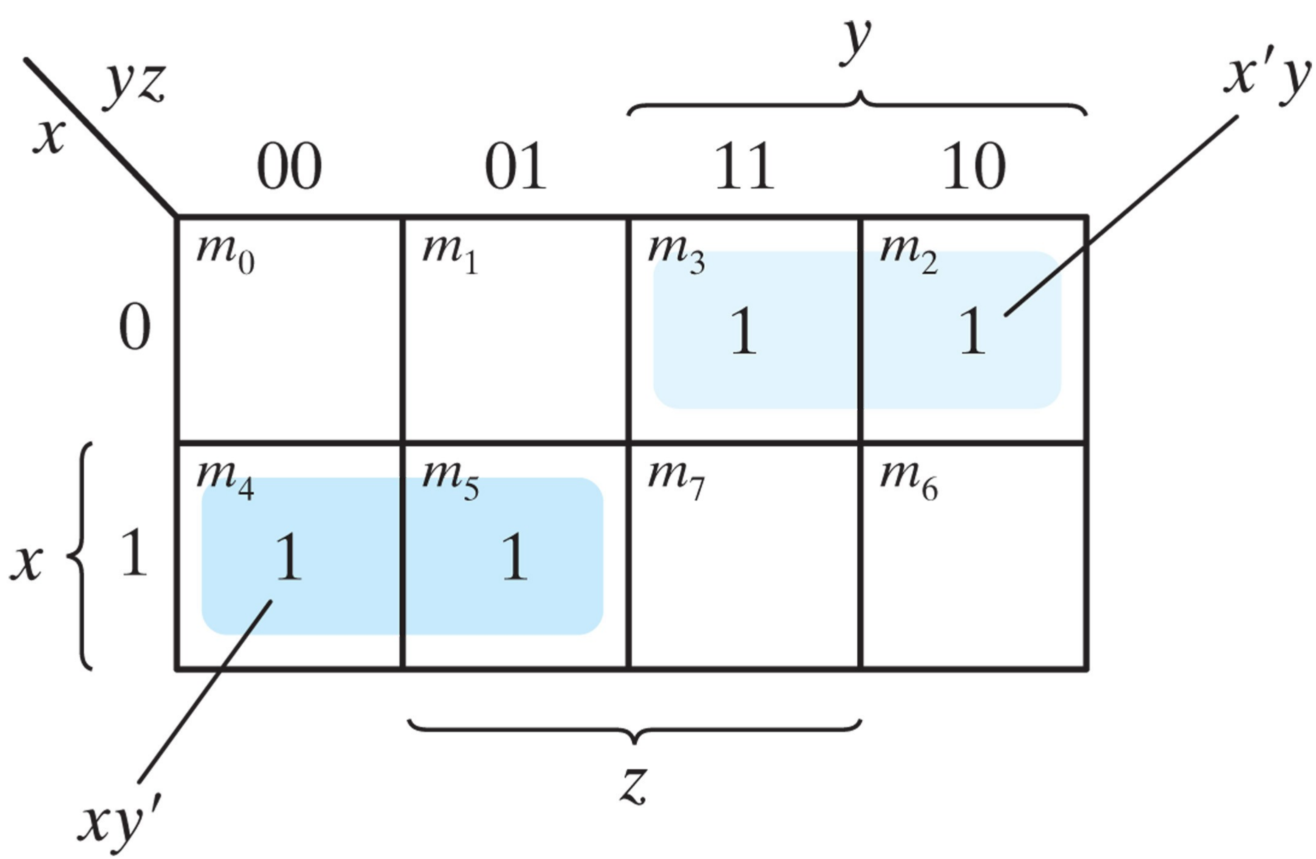
Three-variable map



(a) (b)

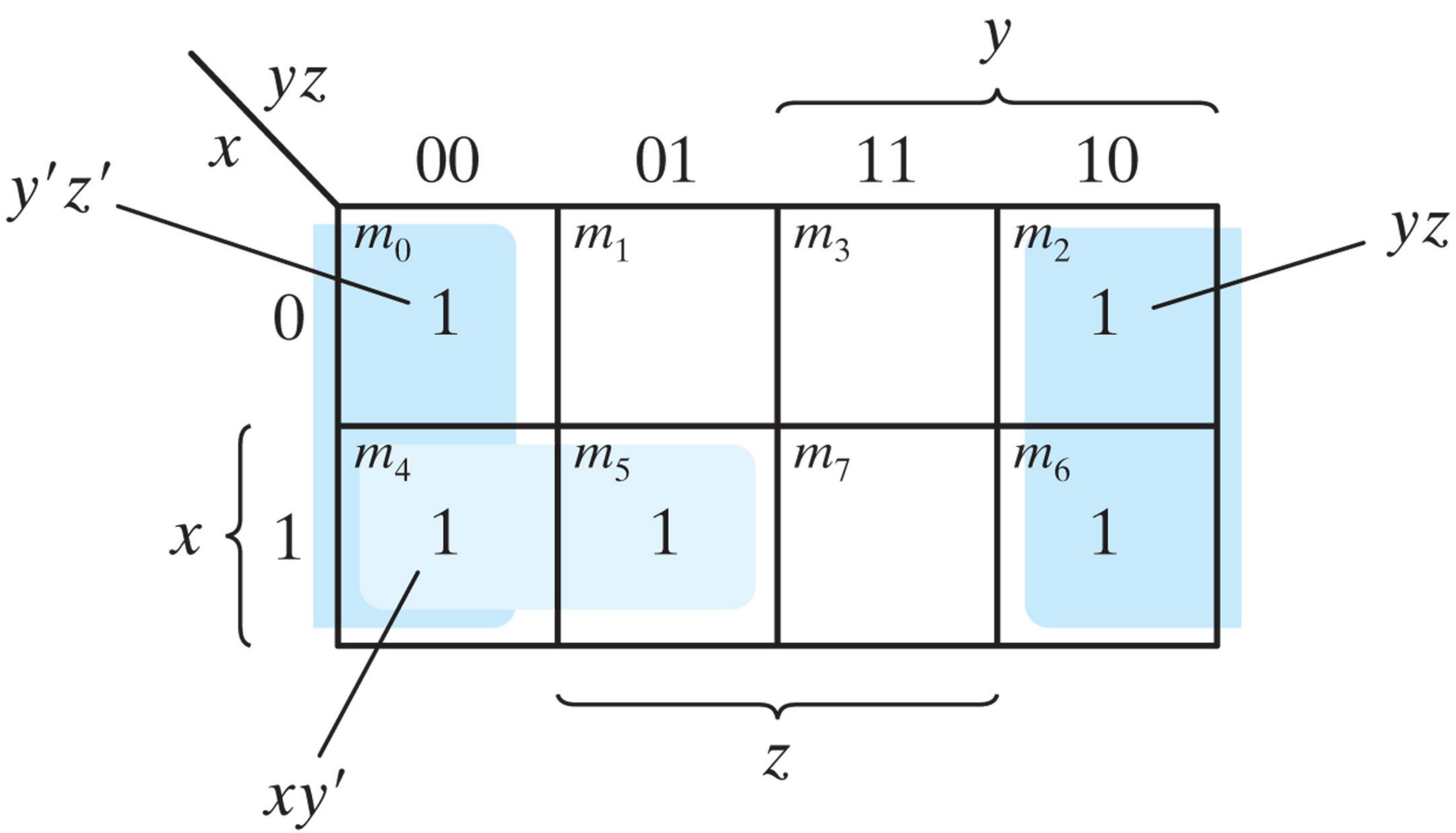
Karnaugh Map Method

Example: $F(x, y, z) = \Sigma(2, 3, 4, 5) = x'y + xy'$



Karnaugh Map Method

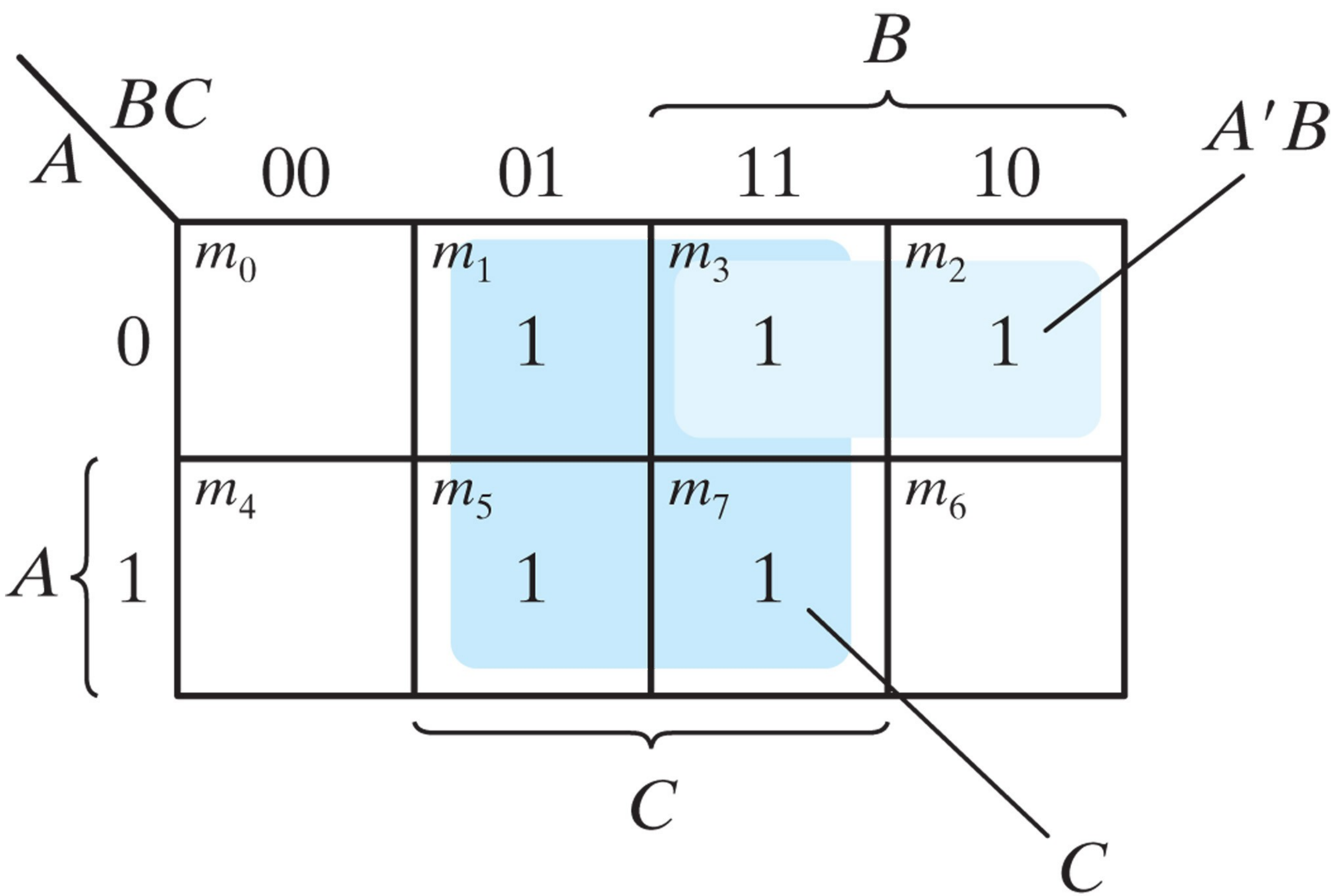
Example: $F(x, y, z) = \Sigma(0, 2, 4, 5, 6) = z' + xy'$



$Note: y'z' + yz' = z'$

Karnaugh Map Method

Example: ***A'C + A'B + AB'C + BC = C + A'B***



Karnaugh Map Method

Four-variable map

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

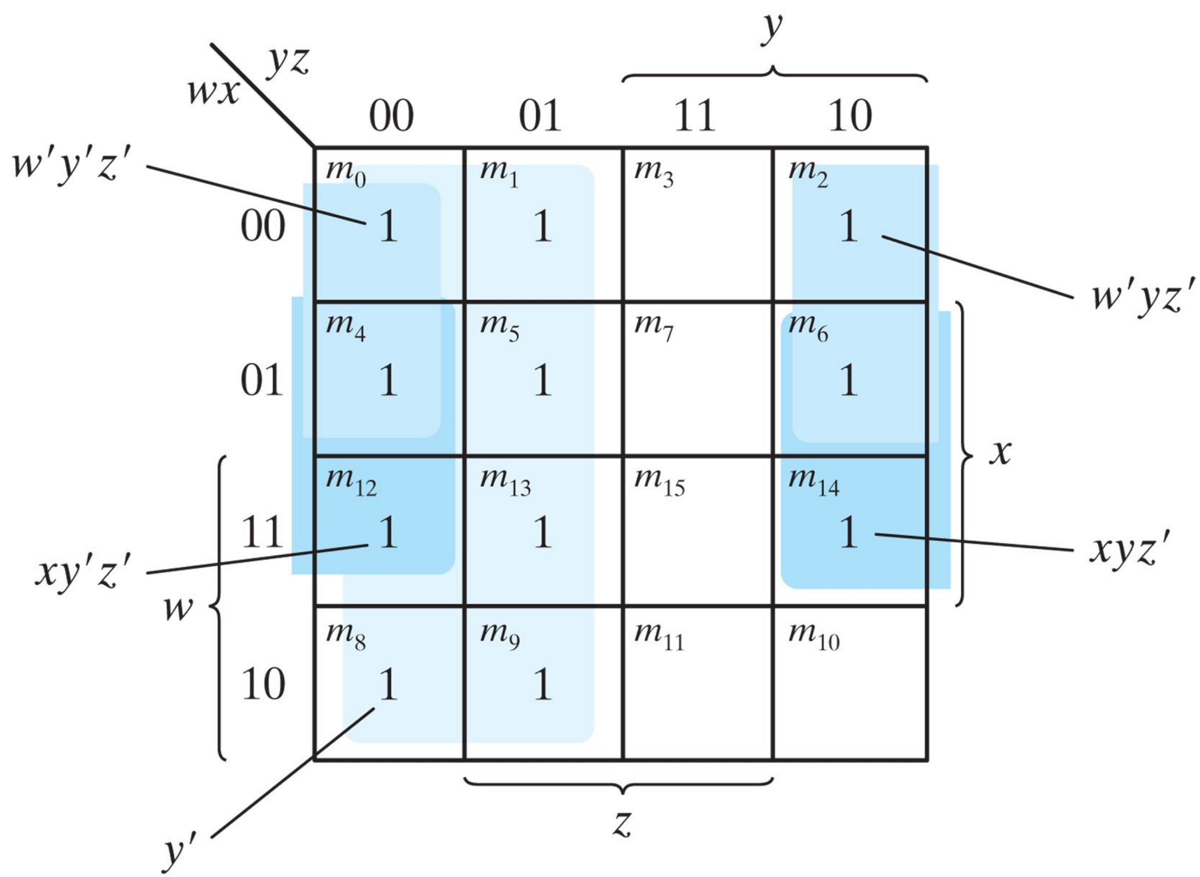
(a)

$wx \backslash yz$	00	01	11	10
00	m_0 $w'x'y'z'$	m_1 $w'x'y'z$	m_3 $w'x'yz$	m_2 $w'x'yz'$
01	m_4 $w'xy'z'$	m_5 $w'xy'z$	m_7 $w'xyz$	m_6 $w'xyz'$
11	m_{12} $wxy'z'$	m_{13} $wxy'z$	m_{15} $wxyz$	m_{14} $wxyz'$
10	m_8 $wx'y'z'$	m_9 $wx'y'z$	m_{11} $wx'yz$	m_{10} $wx'yz'$

(b)

Karnaugh Map Method

Example: ***F*(*w*, *x*, *y*, *z*) = Σ(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14) =**
y' + w'z' + xz'



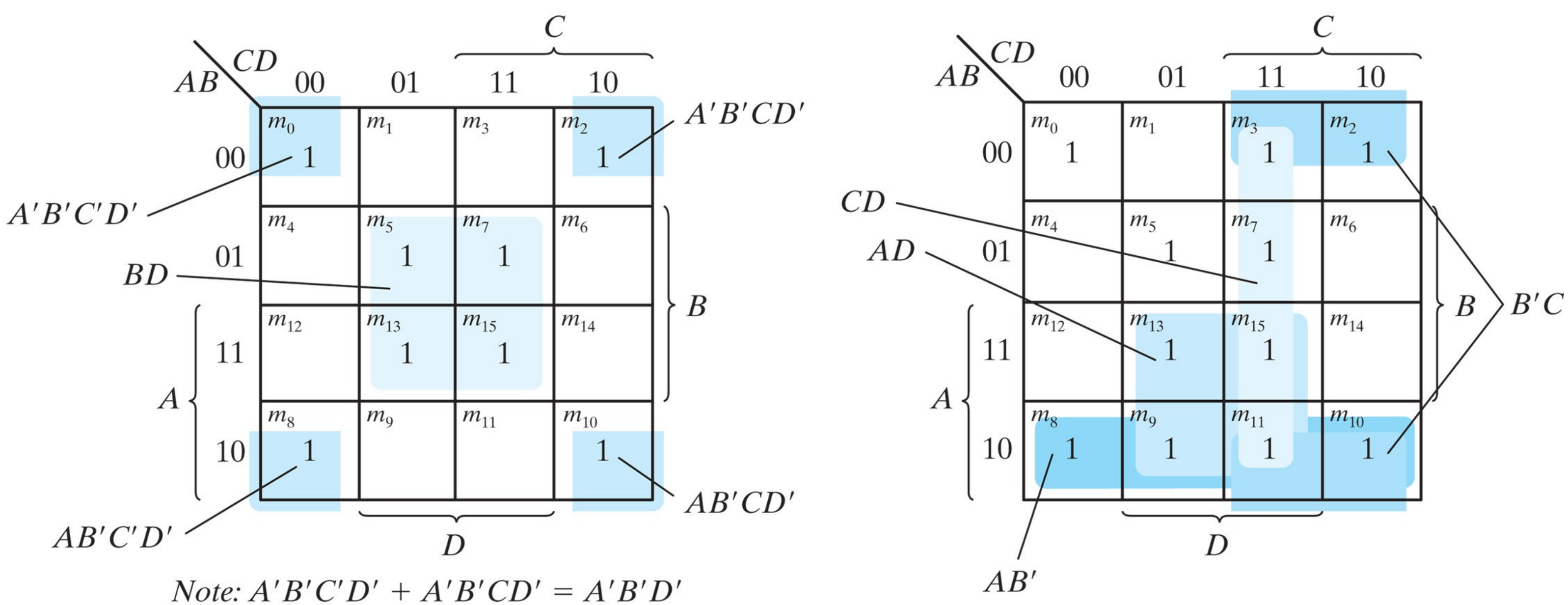
$$Note: w'y'z' + w'yz' = w'z'$$
$$xy'z' + xyz' = xz'$$

Karnaugh Map Method

A prime implicant is a product term obtained by combining the maximum possible number of adjacent squares in the map.

If a minterm in a square is covered by only one prime implicant, that prime implicant is said to be **essential**.

Example: $F(A, B, C, D) = (0, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$



Note: $A'B'C'D' + A'B'CD' = A'B'D'$
$AB'C'D' + AB'CD' = AB'D'$
$A'B'D' + AB'D' = B'D'$

(a) Essential prime implicants BD and $B'D'$

(b) Prime implicants CD, $B'C$, AD, and AB'

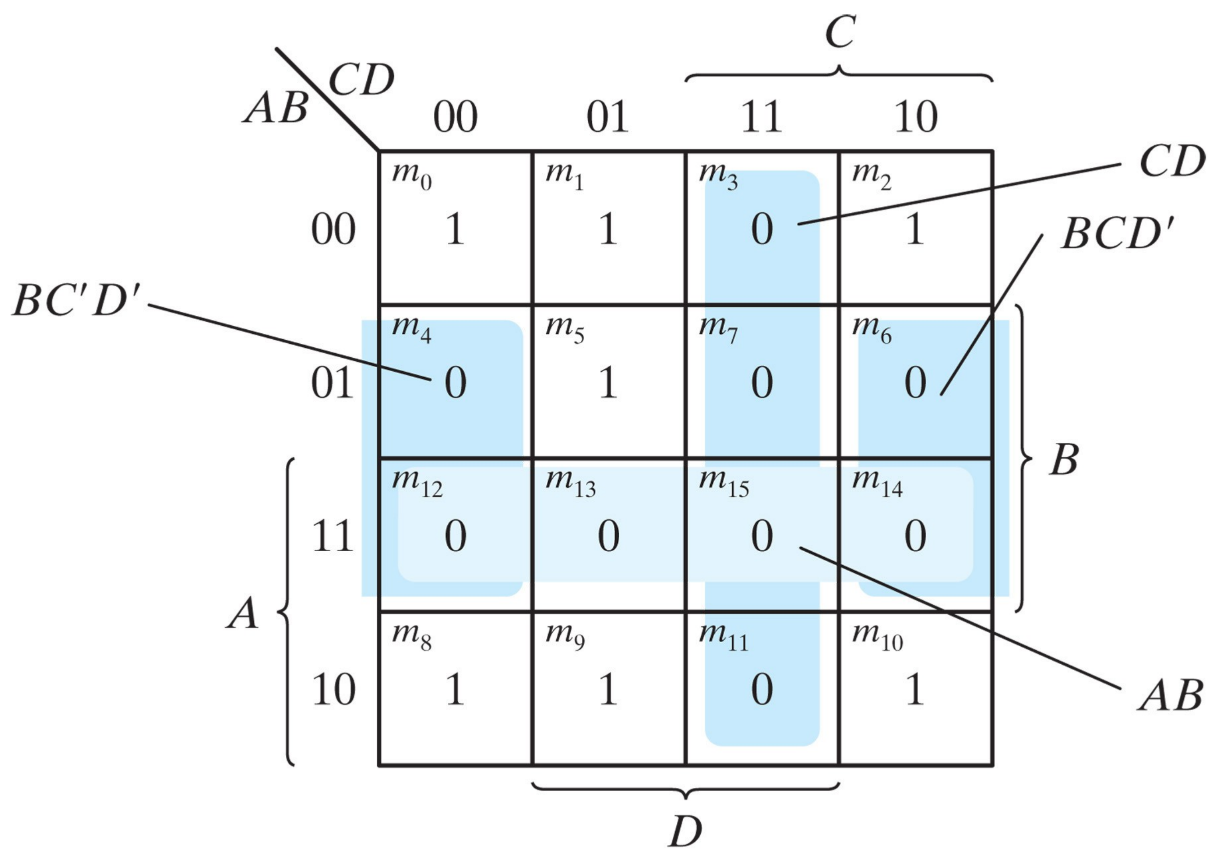
Karnaugh Map Method

Example: $F(A, B, C, D) = (0, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$

$$\begin{aligned} F &= BD + B'D' + CD + AD \\ &= BD + B'D' + CD + AB' \\ &= BD + B'D' + B'C + AD \\ &= BD + B'D' + B'C + AB' \end{aligned}$$

Karnaugh Map Method

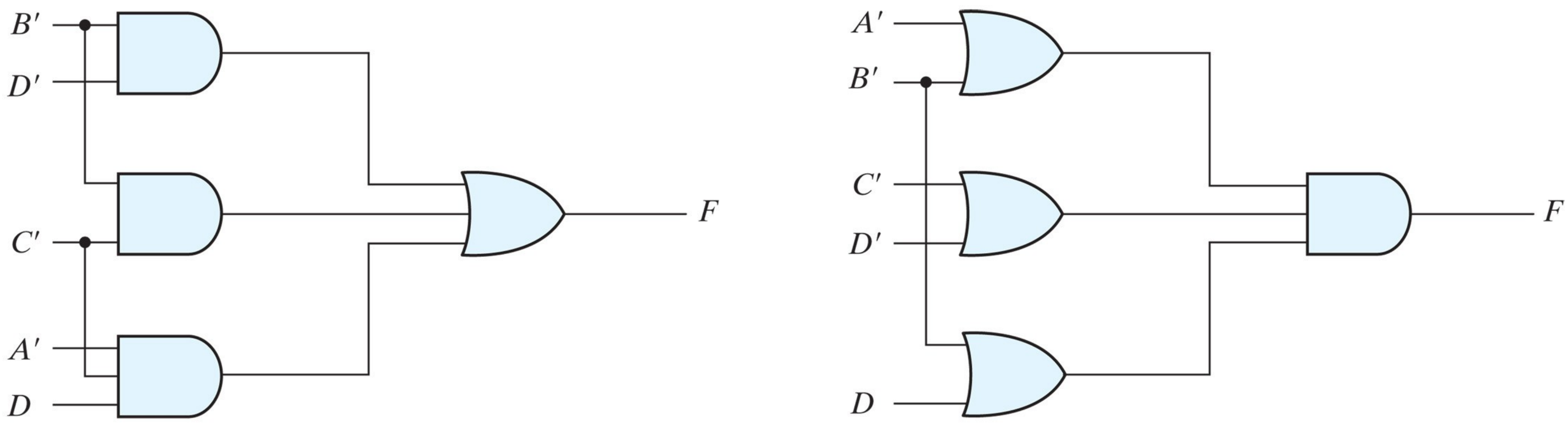
Example: ***F*(*A*, *B*, *C*, *D*) = Σ(0, 1, 2, 5, 8, 9, 10) = *B'D'* + *B'C'* + *A'C'D* = (*A'* + *B'*)(*C'* + *D'*)(*B'* + *D*)**



Note: $BC'D' + BCD' = BD'$

Karnaugh Map Method

Example: $F(A, B, C, D) = \Sigma(0, 1, 2, 5, 8, 9, 10) = B'D' + B'C' + A'C'D = (A' + B')(C' + D')(B' + D)$



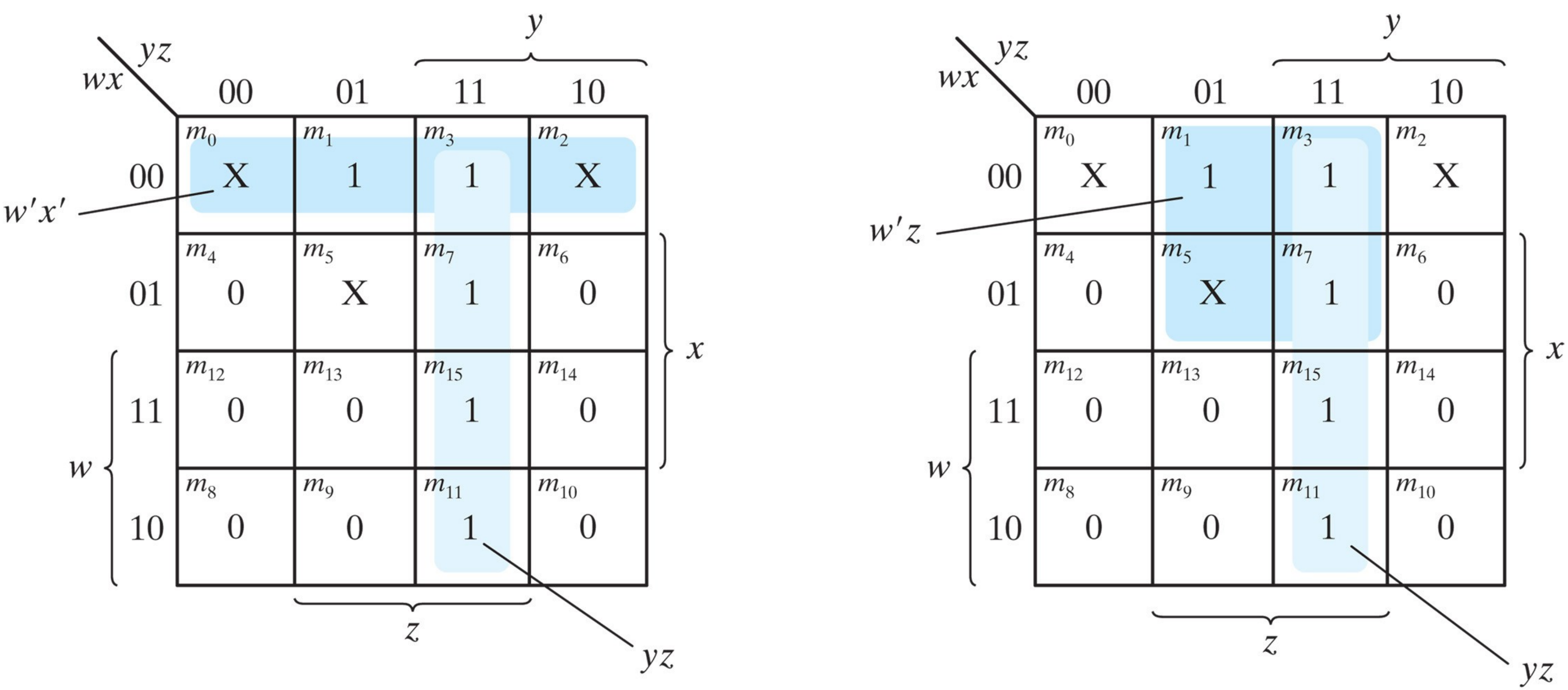
(a) $F = B'D' + B'C' + A'C'D$ (b) $F = (A' + B')(C' + D')(B' + D)$

Karnaugh Map Method

Don't care conditions

Simplify: $F(w, x, y, z) = \Sigma(1, 3, 7, 11, 15)$ **with**

Don't care terms $d(w, x, y, z) = \Sigma(0, 2, 5)$



(a) $F = yz + w'x'$ (b) $F = yz + w'z$

Karnaugh Map Method

To obtain a simplified product-of-sums expression of

$\boldsymbol{F(w, x, y, z) = \Sigma(1, 3, 7, 11, 15)}$ with

Don't care terms $\boldsymbol{d(w, x, y, z) = \Sigma(0, 2, 5)}$

Combine the 0's and don't-care minterms giving a simplified complemented function:

$$\boldsymbol{F' = z' + wy'}$$

Then

$$\boldsymbol{F(w, x, y, z) = z(w' + y)}$$

Karnaugh Map Method

Guidelines for Simplifying Functions Using K-maps

In general, each square (minterm) on a K-map on n variables has n logically adjacent squares, with each pair of adjacent squares differing in exactly one variable.

Always group squares in powers of 2. Grouping 2^n squares eliminates n variables.

Group as many squares as possible; the larger the group, the fewer the number of literals in the resulting product term.

Karnaugh Map Method

Guidelines for Simplifying Functions Using K-maps

Make as few groups as possible to cover all the squares of a function. A minterm is covered if it is included in a t least one group. The fewer the groups, the fewer the product terms.

Each minterm may be used as many times as needed for grouping, however, it must be used at least once. Stop as soon as all minterms are covered.

Begin groupings with squares that have fewest numbers of adjacent squares. Minterms with more adjacent squares offer more possibilities and should be left last.

Karnaugh Map Method

An algorithm for deriving minimal SOP forms from K-maps

1. Circle all prime implicants on the K-map.
2. Identify and select all essential prime implicants for the cover.
3. Select a minimum subset of the remaining prime implicants to complete the cover, that is, to cover those minterms not covered by the essential implicants.