# Chapter 3
# Gate-Level Minimization

**Karnaugh Maps**

**POS Simplification**

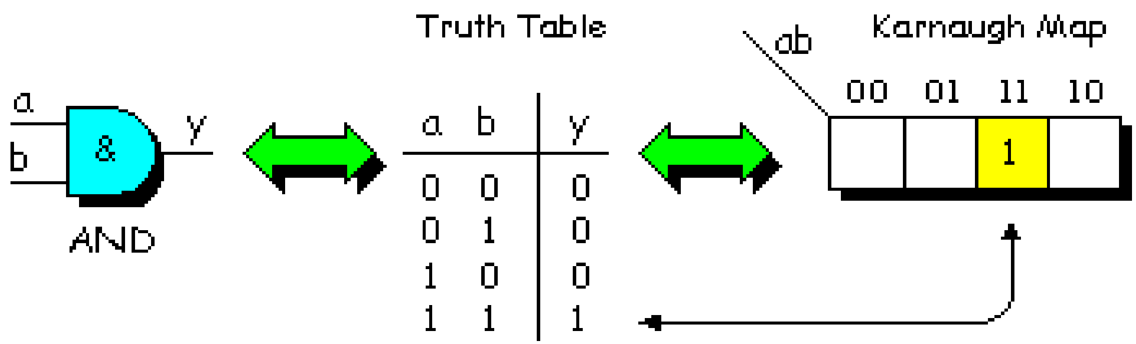
**Don’t Care Conditions**

**NAND and NOR**

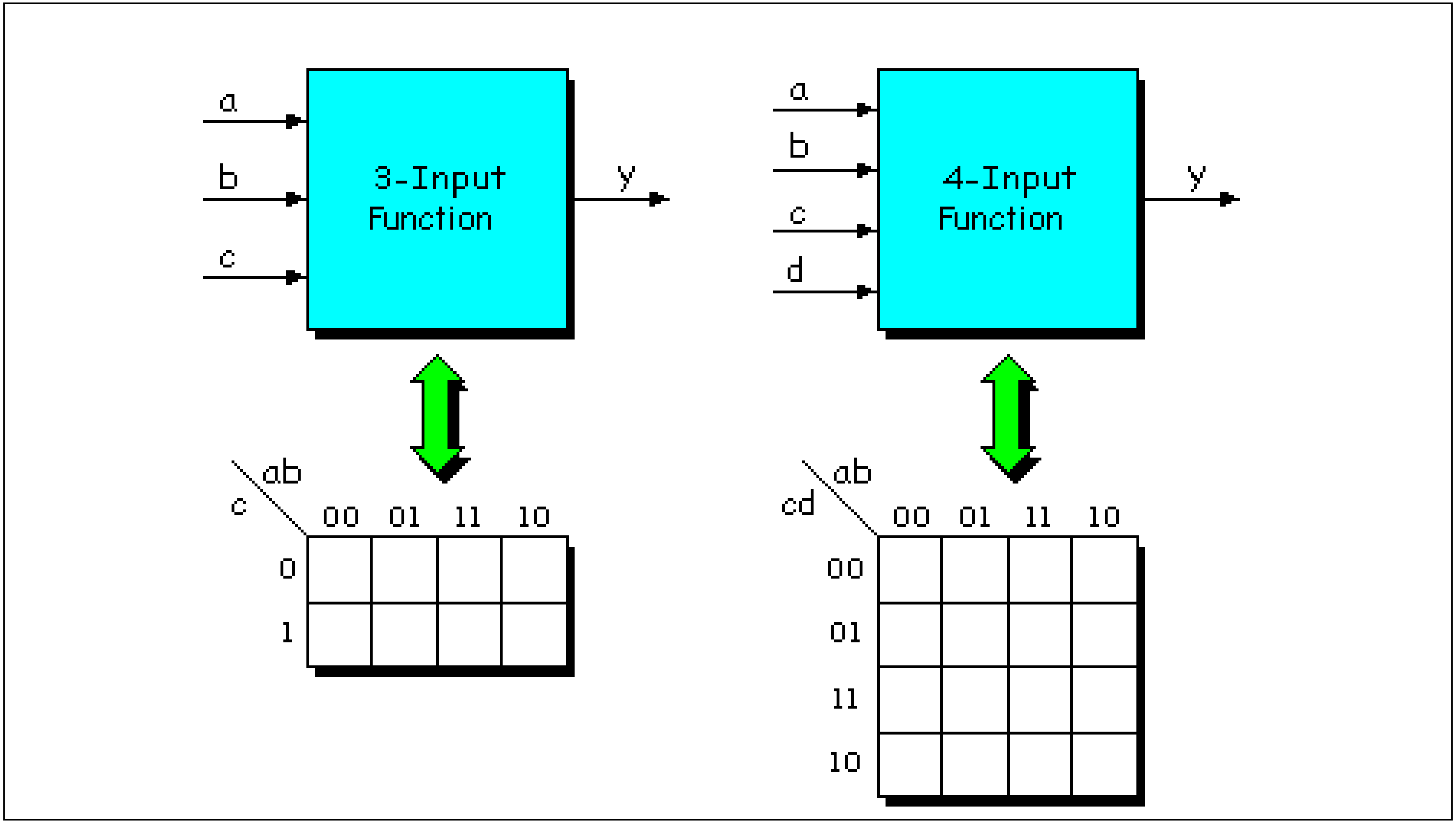
**Exclusive OR**

# K-Map

- **Provide an alternative technique for representing Boolean functions**
- **Box for every line of the Truth Table**
- **Karnaugh map's input values must be ordered such that the values for adjacent columns vary by only a single bit, for example, 00, 01, 11, and 10. This is necessary to observe the variable transitions**
  - **Known as a *gray code***



# Multiple Inputs K-Map



# K-Map Method

- **The** Karnaugh Map **(**K-Map**) method uses a simple procedure for minimizing Boolean functions.**
  - **The map is a diagram made up of squares with each square representing one minterm of the function.**
  - **The key is to learn to identify visual patterns.**
  - **The result is always an expression that is in one of the two standard forms, SOP or POS.**
  - **Much faster and more more efficient than previous minimization techniques with Boolean algebra.**
  - **It is possible to find two or more expressions that satisfy the minimization criteria.**
  - **Rules to consider**
    - **Every cell containing a 1 must be included at least once.**
    - **The largest possible "power of 2 rectangle" must be enclosed.**
    - **The 1's must be enclosed in the smallest possible number of rectangles.**

# Two-Variable Map

- A two-variable map holds four minterms for two variables.
  - We mark the squares of the minterms that belong to a given function.

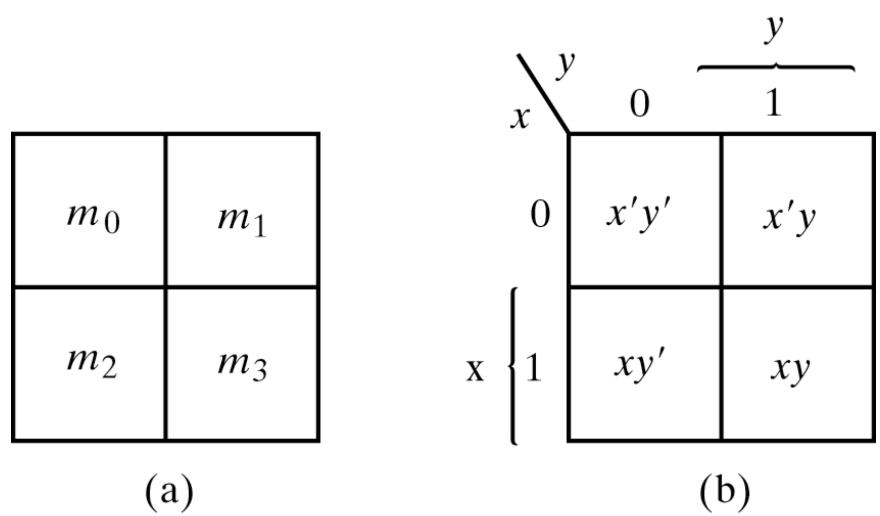



Fig. 3-1 Two-variable Map

# Representing 2-Variable Functions

- **K-Map for AND**

| x \ y | 0 | 1 |
|---|---|---|
| 0 | | |
| 1 | | 1 |

- **K-Map for OR**

| x \ y | 0 | 1 |
|---|---|---|
| 0 | | 1 |
| 1 | 1 | 1 |

(a) $xy$

| $x$ \ $y$ | 0 | 1 ($y$) |
|---|---|---|
| 0 | | |
| 1 ($x$) | | 1 |

(b) $x + y$

| $x$ \ $y$ | 0 | 1 ($y$) |
|---|---|---|
| 0 | | 1 |
| 1 ($x$) | 1 | 1 |

Fig. 3-2 Representation of Functions in the Map

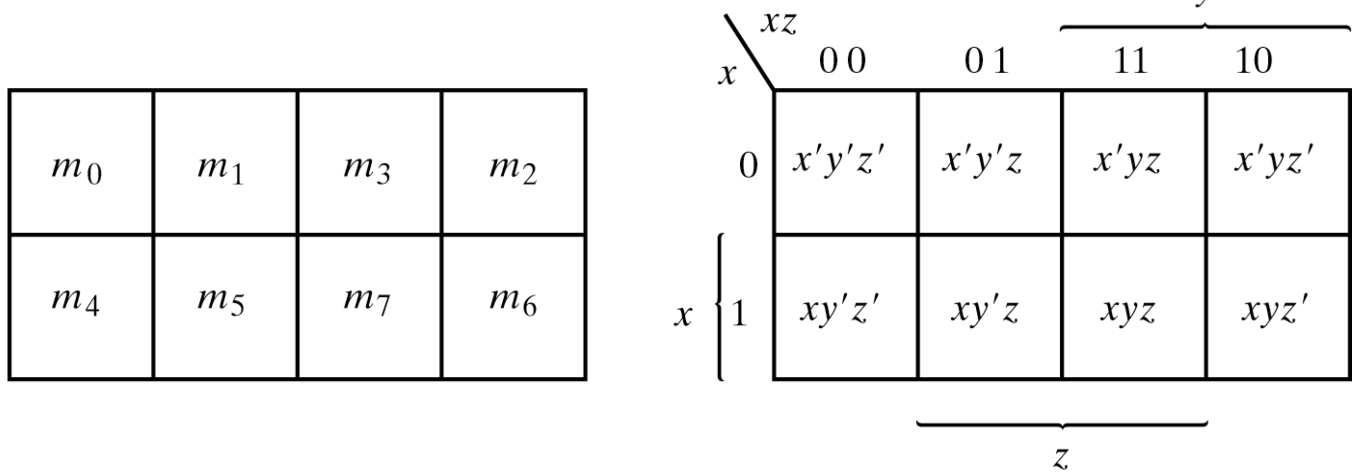
# Three-Variable Map

- **A** three-variable map **holds eight minterms for three variables.**
  - **Again, we mark the squares of the minterms that belong to a given function.**
  - **Note that the sequence is not arranged in a binary way.**
  - **The sequence used, similar to Gray code, allows only one bit to change from column to column and row to row.**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $x'y'z'$ | $x'y'z$ | $x'yz$ | $x'yz'$ |
| 1 | $xy'z'$ | $xy'z$ | $xyz$ | $xyz'$ |

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

# Three-Variable Map




(a) (b)

Fig. 3-3 Three-variable Map

- Correction: **Columns are yz and not xz in fig 3-3 (book)**

## Mapping Functions

- **When you have already been provided a function, you can map the function into a K-map by remembering**
  - **the cells of a k-map represent minterms**
  - **a 1 in a cell indicates that the minterm is part of the function**
  - **two adjacent 1's represent a two literal term**
  - **four adjacent 1's represent a one literal term**
  - **eight adjacent 1's represent a true function, F = 1**

# Minimization Characteristic in 3-Variable Maps

- **Since any two adjacent cells in a 3-variable map represent a change in only a single bit, we use this to do minimization.**
  - **Consider the two cells for $m_0$ and $m_1$ where the difference is the negation of the bit z.**
  - **$F = m_0 + m_1 = x'y'z' + x'y'z = x'y'(z' + z) = x'y'$**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | x'y'z' | x'y'z | x'yz | x'yz' |
| 1 | xy'z' | xy'z | xyz | xyz' |

# Minimization Example

- **Each of the two adjacent pairs of entries can be simplified by eliminating the changing bit (z in both cases).**
  - **F (x,y,z) = x’y’ + xy**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | | |
| 1 | | | 1 | 1 |

$$F(x, y, z) = \sum(0,1,6,7)$$

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

# Note on Adjacency

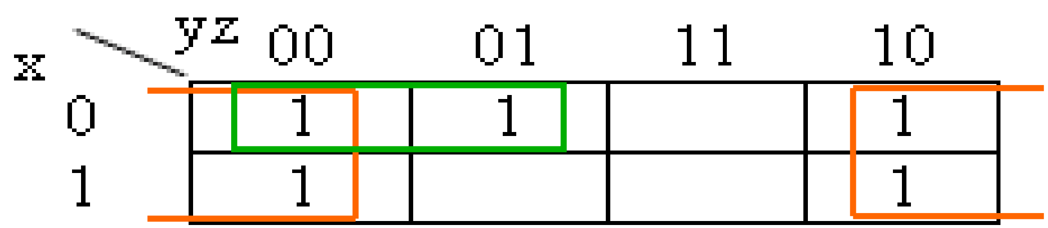
- So far, we have assumed that adjacent cells in the map need to touch each other but this is not always the case.
  - $m_0$ and $m_2$ are considered adjacent
    - » $m_0 + m_2 = x'y'z' + x'yz' = x'z'(y' + y) = x'z'$
  - $m_4$ and $m_6$ are considered adjacent
    - » $m_4 + m_6 = xy'z' + xyz' = xz'(y' + y) = xz'$

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | x'y'z' | x'y'z | x'yz | x'yz' |
| 1 | xy'z' | xy'z | xyz | xyz' |

# Another Example

- **The four adjacent squares can be combined to give the single literal term z'**
- **The remaining single term is combined with the adjacent square that was already used to give us the term x'y'**
- **F = z' + x'y'**




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

# 3-Variable Map Patterns

- **The number of adjacent squares that may be combined always represent a number that is a power of 2 such as 1, 2, 4, and 8.**
  - **One square represents one minterm with three literals.**
  - **Two adjacent squares represents a term of two literals.**
  - **Four adjacent squares represents a term of one literal.**
  - **Eight adjacent squares represents the entire map and produces a function that is always equal to 1.**

# Mapping Functions (Example)

- **Given the function**
  - F = x’z + xy’ + xy’z + yz
- **Map the function**
- **Determine the sum of minterms equation**
- **Determine the minimum sum of products expression**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | | 1 | 1 | |
| 1 | 1 | 1 | 1 | |

- **Sum of minterms: F = ∑(1, 3, 4, 5, 7)**
- **Minimum sum of products: F = z + xy’**

# Another Minimization Example

- **Each of the two adjacent pairs of entries can be simplified by eliminating the changing bit.**
  - **x is eliminated in column 2**
  - **y is eliminated in the other pair.**
  - **F = y'z + x'z'**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | | 1 |
| 1 | | 1 | | |

$$F(x, y, z) = \sum(0,1,2,5)$$

# Another Minimization Example

- **Two variable maps.**
  - **F=AB +A′B +AB′**

| A \ B | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 1 |

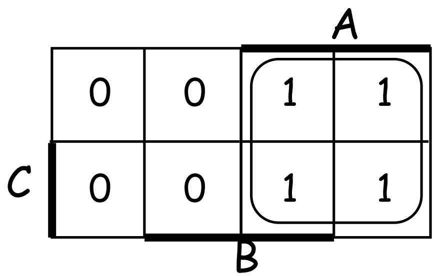
**F=A+B**

- **Three variable maps.**
  - **F=AB’C’ +AB ′C +ABC +ABC ′ + A’B’C + A’BC’**

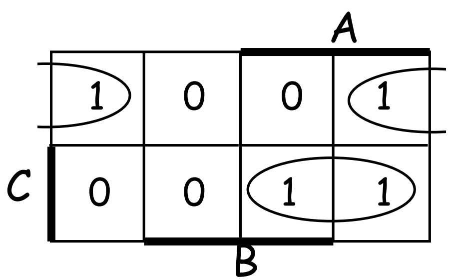
| A \ BC | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**F=A+B ′C +BC ′**

# Another Minimization Example




$G(A,B,C) = A$




$F(A,B,C) = \Sigma m(0,4,5,7) = AC + B'C'$

# Another Minimization Example

| b \ a | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 1 |

f = a

| b \ a | 0 | 1 |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 0 | 0 |

g = b'

| c \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 |

cout = ab + bc + ac

| c \ ab | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |

f = a

# Example 3-1

- Simplify boolean function $F(x,y,z) = \Sigma(2,3,4,5)$
- Sol:
  - 1 is marked in each minterm that represents the function
  - Find the possible adjacent squares and mark them with rectangles
  - The upper right rectangle represents the area enclosed closed by x’y (eliminating the changing bit)
  - Similarly lower left rectangle represents xy’
  - The logical sum of these two terms gives:

    $F = x'y + xy'$

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

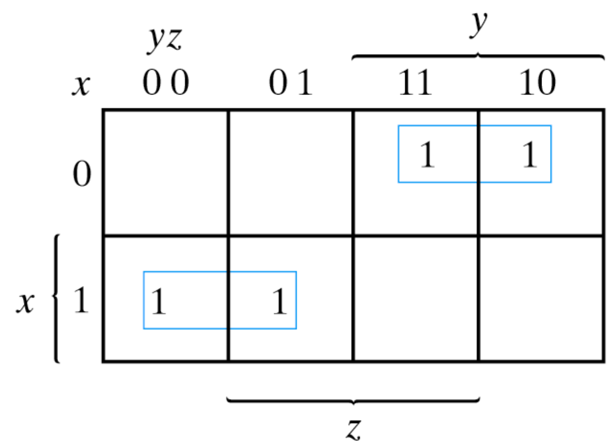



Fig. 3-4 Map for Example 3-1; $F(x, y, z) = \Sigma(2, 3, 4, 5) = x'y + xy'$

# Example 3-2

- $F(x,y,z) = \Sigma(3,4,6,7)$
- Sol:
  - 1 is marked in each minterm that represents the function
  - Find the possible adjacent squares and mark them with rectangles
  - Two adjecent squares are combined in the third column to give a two-literal term yz
  - The remaining two squares with 1's are enclosed in half rectangles. This gives two-literal term xz'
  - The logical sum of these two terms gives:

    **F = yz + xz'**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

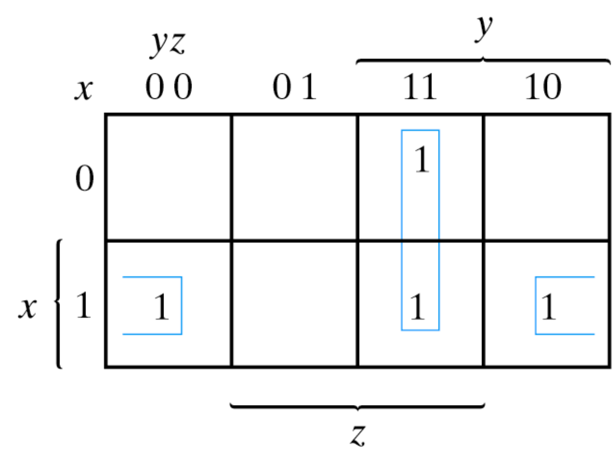



Fig. 3-5 Map for Example 3-2; $F(x, y, z) = \Sigma(3, 4, 6, 7) = yz + xz'$

# Example 3-3

- $F(x,y,z) = \Sigma(0,2,4,5,6)$

- Sol:
  - **1 is marked in each minterm that represents the function**
  - **Find the possible adjacent squares and mark them with rectangles**
  - **We combine four adjacent squares to get a single literal term z' as $m_0+m_2+m_4+m_6$**

    **$= x'y'z'+x'yz'+xy'z'+xyz'$**

    **$= x'z'(y'+y) + xz'(y'+y)$**

    **$= x'z' + xz' = z'$**
  - **The remaining two squares with 1's are enclosed by a rectangle (with one square that is already used once). This gives two-literal term xy'**
  - **The logical sum of these two terms gives:**

    **$F = z' + xy'$**

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
| 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

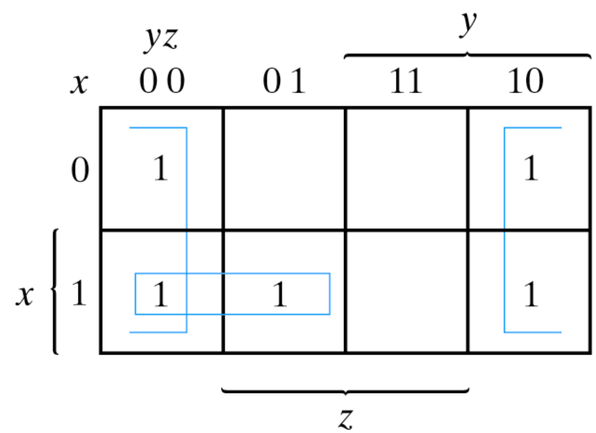
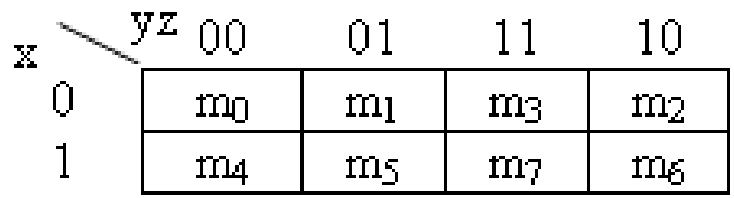



Fig. 3-6 Map for Example 3-3; $F(x, y, z) = \Sigma(0, 2, 4, 5, 6) = z' + xy'$

# Example 3-4

- – F = A'C + A'B + AB'C + BC
  - – express it in sum of minterms
  - find the minimal sum of products expression
- Sol:
  - – The two squares corresponding to the first term A’C. (A’ first row and C two middle columns)
  - – A’B has 1’s in squares 011 and 010 in the same way
  - – AB’C has 1 square 101 and BC has two 1’s in squares 011 and 111
  - – The function has total of 5 minterms as shown in figure
  - – Find the possible adjacent squares and mark them with rectangles as shown in the map
  - – It can be simplified with only two terms giving:

    F = C + A’B



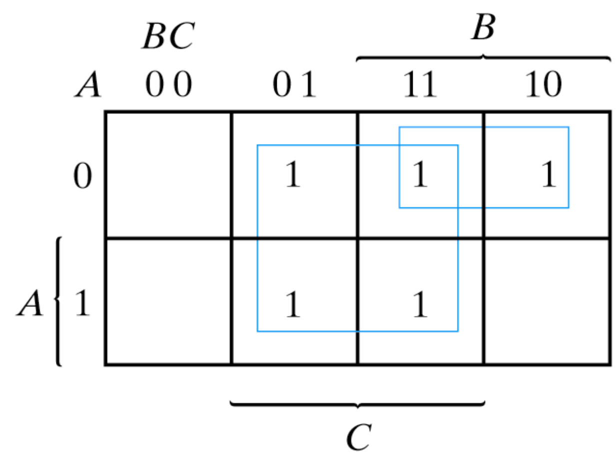




Fig. 3-7 Map for Example 3-4; $A'C + A'B + AB'C + BC = C + A'B$