Optimized Implementation of Logic Functions

Chapter 4

Chapter Objectives

- Synthesis of logic functions
- Analysis of logic circuits
- Techniques for deriving minimum-cost implementation of logic functions
- Graphical representation of logic functions in the form of Karnaugh maps

Contents

KARNAUGH MAP

Why Karnaugh Map?

Table 2-2
Truth Tables for F_1 and F_2

x	y	z	F_1	F_2
0	0	0	0	0
0	0	1	1 (1)	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1 (2)	1
1	0	1	1 (3)	1
1	1	0	1 (4)	0
1	1	1	1 (5)	0

(1) : (3)
(2) : (3),(4)
(3) : (1),(5)
(4) : (2),(5)
(5) : (3),(4)

$$F_1 = \underbrace{x'y'z}_{(1)} + \underbrace{xy'z'}_{(2)} + \underbrace{xy'z}_{(3)} + \underbrace{xyz'}_{(4)} + \underbrace{xyz}_{(5)}$$

Minimization by Boolean Functions

(1) (2) (3) (4) (5)

$$\begin{aligned} F_1 &= x'y'z + xy'z' + xy'z + xyz' + xyz \\ &= y'z(x'+x) + xy'(z'+z) + xy(z'+z) \\ &= y'z + xy' + xy \\ &= x(y'+y) + y'z = \boxed{x + y'z} \end{aligned}$$

(1)+(3), (2)+(3), (4)+(5)

$x + x = x$

$x + x' = 1$

(1) (2) (3) (4) (5)

$$\begin{aligned} F_1 &= x'y'z + xy'z' + xy'z + xyz' + xyz \\ &= y'z(x'+x) + xz'(y'+y) + xy(z'+z) \\ &= \boxed{y'z + xz' + xy} \end{aligned}$$

(1)+(3), (2)+(4), (4)+(5)

Why Karnaugh Map?

- How can we find minimum cost expression?
- Is it a unique optimal solution for a given truth table?
- Are there any strategies or procedures for the minimum cost implementation?

Karnaugh Map

$m_0 + m_4 = ?$
$m_2 + m_6 = ?$

$$f = x_3' + x_1 x_2'$$

Row number	x_1	x_2	x_3	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
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

$$\begin{aligned} m_0 + m_2 &= x_1' x_2' x_3' + x_1' x_2\ x_3' \\ &= x_1' x_3' (x_2' + x_2) \\ &= x_1' x_3' \end{aligned}$$

$$\begin{aligned} m_4 + m_6 &= x_1\ x_2' x_3' + x_1\ x_2\ x_3' \\ &= x_1\ x_3' (x_2' + x_2) \\ &= x_1\ x_3' \end{aligned}$$

$$x_1' x_3' + x_1 x_3' = x_3'$$

$$\begin{aligned} m_4 + m_5 &= x_1\ x_2' x_3' + x_1\ x_2' x_3 \\ &= x_1\ x_2' (x_3' + x_3) \\ &= x_1\ x_2' \end{aligned}$$

Figure 4.1. The function $f(x_1, x_2, x_3) = \Sigma(0, 2, 4, 5, 6)$.

Karnaugh Map

	x_1	x_2	x_3
m_4	1	0	0
m_5	1	0	1

$$m_4 + m_5 = x_1\ x_2'(x_3' + x_3)$$

$x_3 = 0 \quad 1$

	x_1	x_2	x_3
m_0	0	0	0
m_2	0	1	0
m_4	1	0	0
m_6	1	1	0

x_3'

x_1

$$m_0 + m_2 = x_1'x_3'(x_2' + x_2) \qquad 0$$

$$m_4 + m_6 = x_1\ x_3'(x_2' + x_2) \qquad 1$$

$x_2 = 0 \quad 1$

Two-Variable Map

x_1	x_2	f
0	0	m_0
0	1	m_1
1	0	m_2
1	1	m_3

$m_0 = x_1' x_2'$

$m_1 = x_1' x_2$

$m_2 = x_1 \; x_2'$

$m_3 = x_1 \; x_2$

(a) Truth table

$x_2 \backslash x_1$	x_1' 0	x_1 1	
0	m_0	m_2	x_2'
1	m_1	m_3	x_2

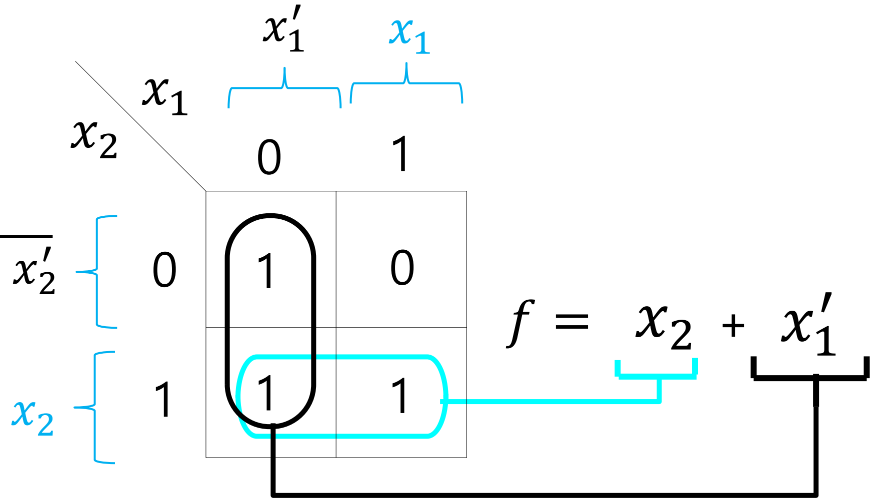
(b) Karnaugh map

Figure 4.2. Location of two-variable minterms.

Two-Variable Map

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

Figure 2.15



$$x_2 = m_1 + m_3$$
$$= x_1'x_2 + x_1\,x_2$$
$$= x_2\,(x_1' + x_1\,)$$
$$x_1 = 0 \quad 1$$

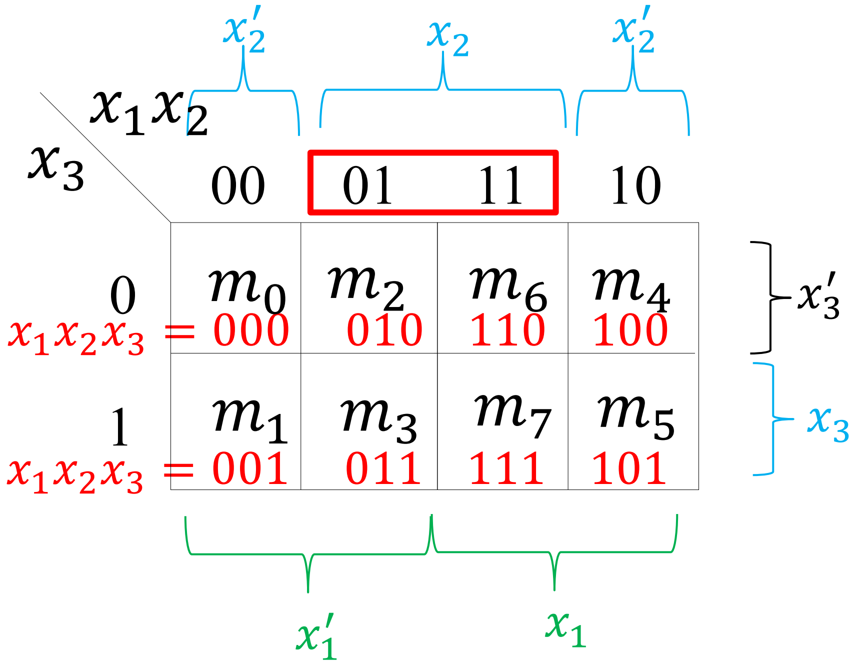
$$x_1' = m_0 + m_1$$
$$= x_1'x_2' + x_1'x_2$$
$$= x_1'(x_2' + x_2\,)$$
$$x_2 = 0 \quad 1$$

Figure 4.3. The function of Figure 2.15.

Three-Variables Map

x_1	x_2	x_3	
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7

(a) Truth table



(b) Karnaugh map

Figure 4.4. Location of three-variable minterms.

Row number	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
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

Figure 2.18

x_3 \ x_1x_2	00	01	11	10
0	0	0	1	1
1	1	0	0	1

$$f = x_1x_3' + x_2'x_3$$

x_1x_2'

(a) The function of Figure 2.18

$$xy + x'z + yz = xy + x'z$$

$yz(x' + x)$ $\quad xy(1 + z)$

$$x_3x_2' + x_3'x_1 + x_2'x_1 = x_3x_2' + x_3'x_1$$

$let\ x = x_3, y = x_2', z = x_1$

Figure 4.5. Examples of three-variable Karnaugh maps.

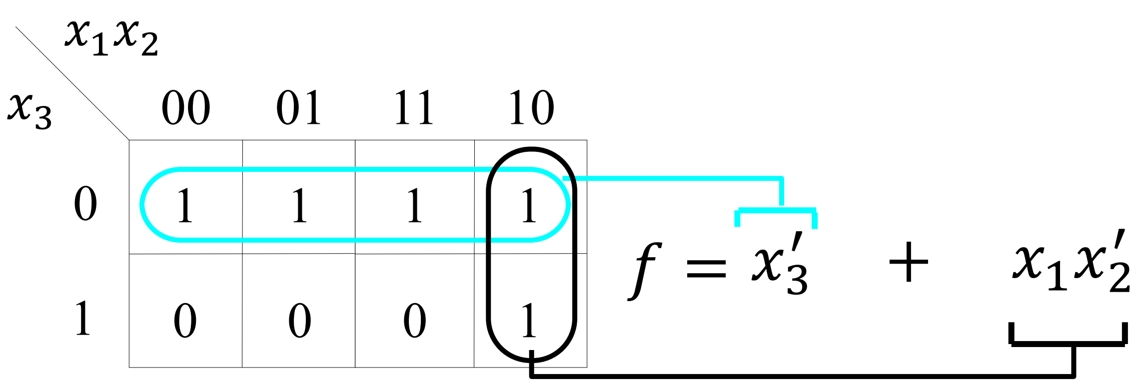
Row number	x_1	x_2	x_3	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Figure 4.1

$$\boxed{\begin{aligned} m_0 + m_2 &= x_1' x_2' x_3' + x_1' x_2\ x_3' \\ &= x_1' x_3' (x_2' + x_2) \\ &= x_1' x_3' \end{aligned}}$$

$$\boxed{\begin{aligned} m_4 + m_6 &= x_1\ x_2' x_3' + x_1\ x_2\ x_3' \\ &= x_1\ x_3' (x_2' + x_2) \\ &= x_1\ x_3' \end{aligned}}$$

$$x_1' x_3' + x_1 x_3' = x_3'$$



(b) The function of Figure 4.1

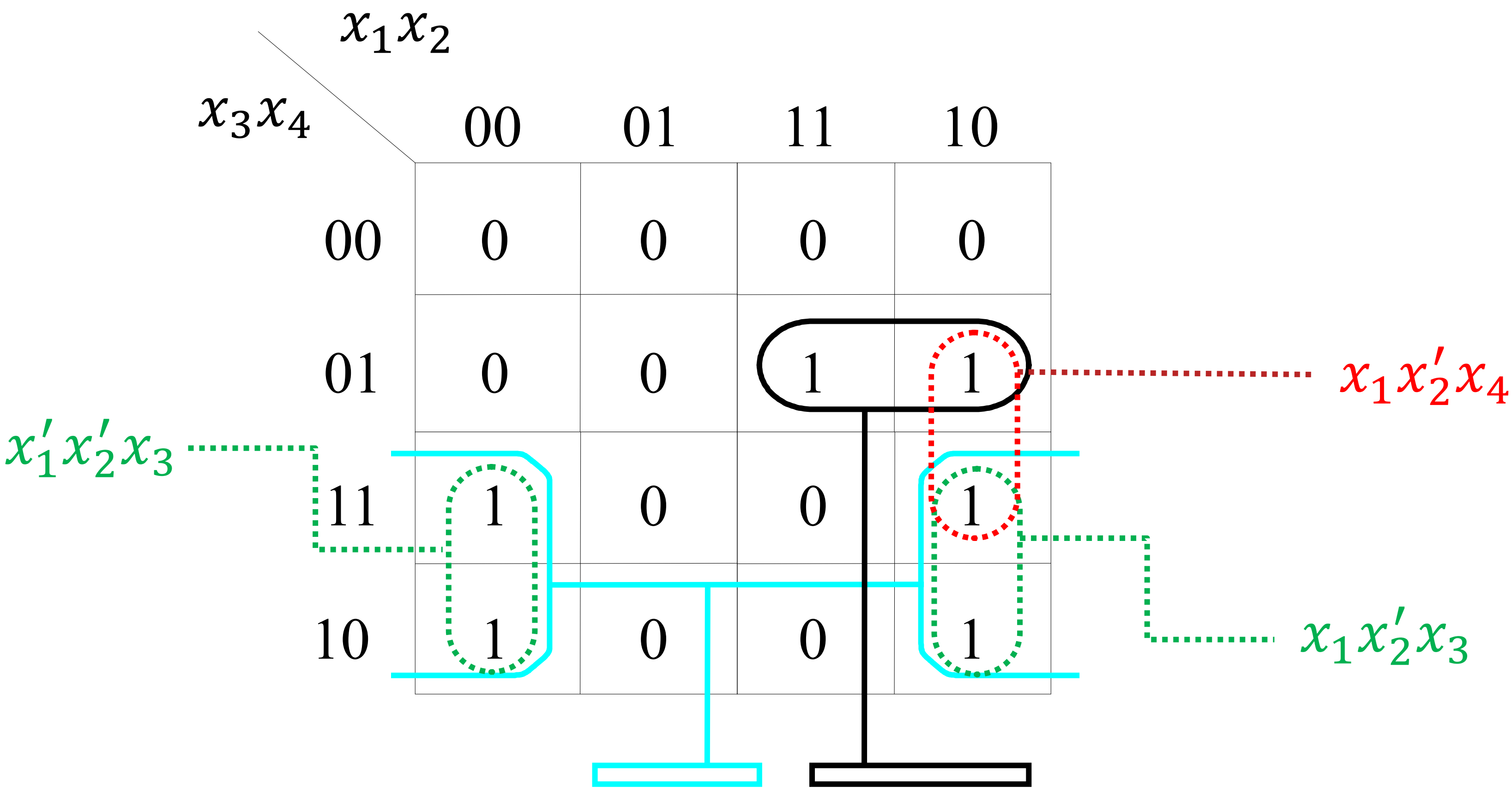
Figure 4.5. Examples of three-variable Karnaugh maps.

Four-Variables Map

x_3x_4 \ x_1x_2	00	01	11	10
00	m_0	m_4	m_{12}	m_8
01	m_1	m_5	m_{13}	m_9
11	m_3	m_7	m_{15}	m_{11}
10	m_2	m_6	m_{14}	m_{10}

x_1 x_2 x_3 x_4

Figure 4.6. A four-variable Karnaugh map.



$$\boxed{f_1 = \underbrace{x_2'x_3}_{x_1'x_2'x_3 + x_1x_2'x_3} + x_1x_3'x_4} \; (+x_1x_2'x_4)$$

$$xy + x'z + yz = xy + x'z$$

let $x = x_3$ $y = x_2'$ $z = x_1x_4$

Figure 4.7. Examples of four-variable Karnaugh maps.

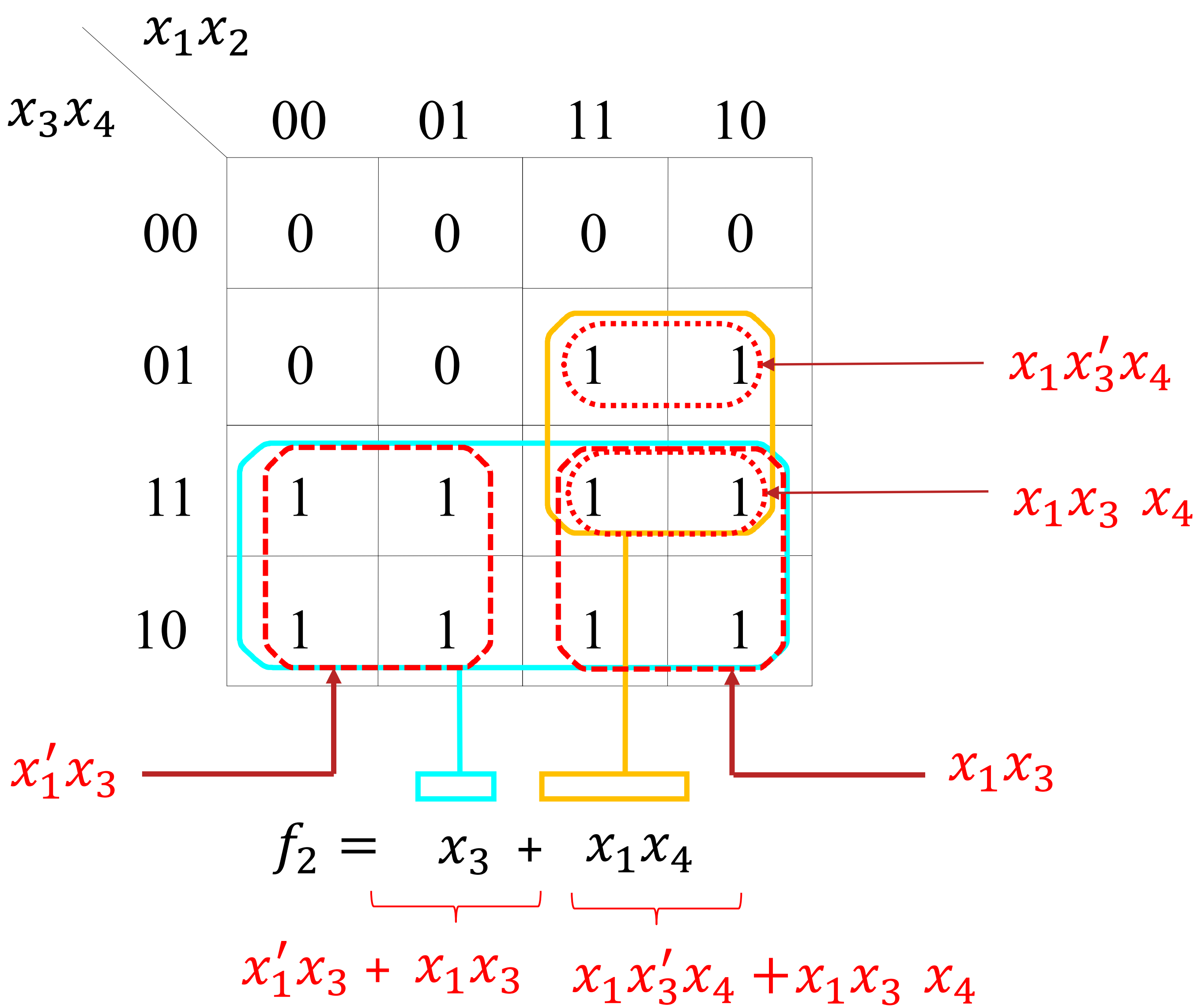
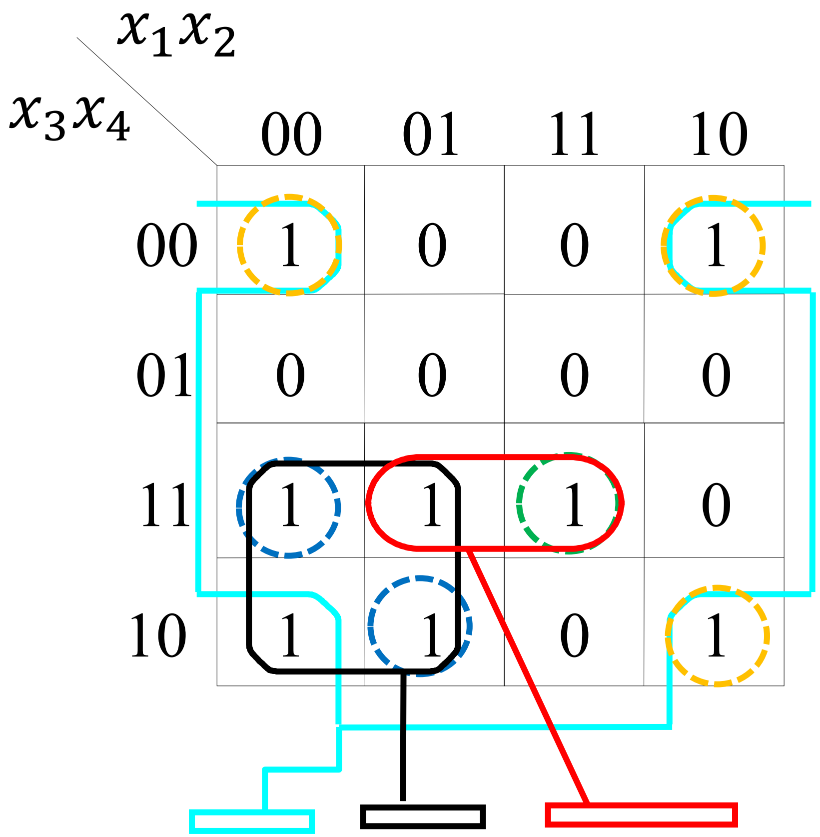


Figure 4.7. Examples of four-variable Karnaugh maps.



$$f_3 = x_2'x_4' + x_1'x_3 + x_2x_3x_4$$

Figure 4.7. Examples of four-variable Karnaugh maps.

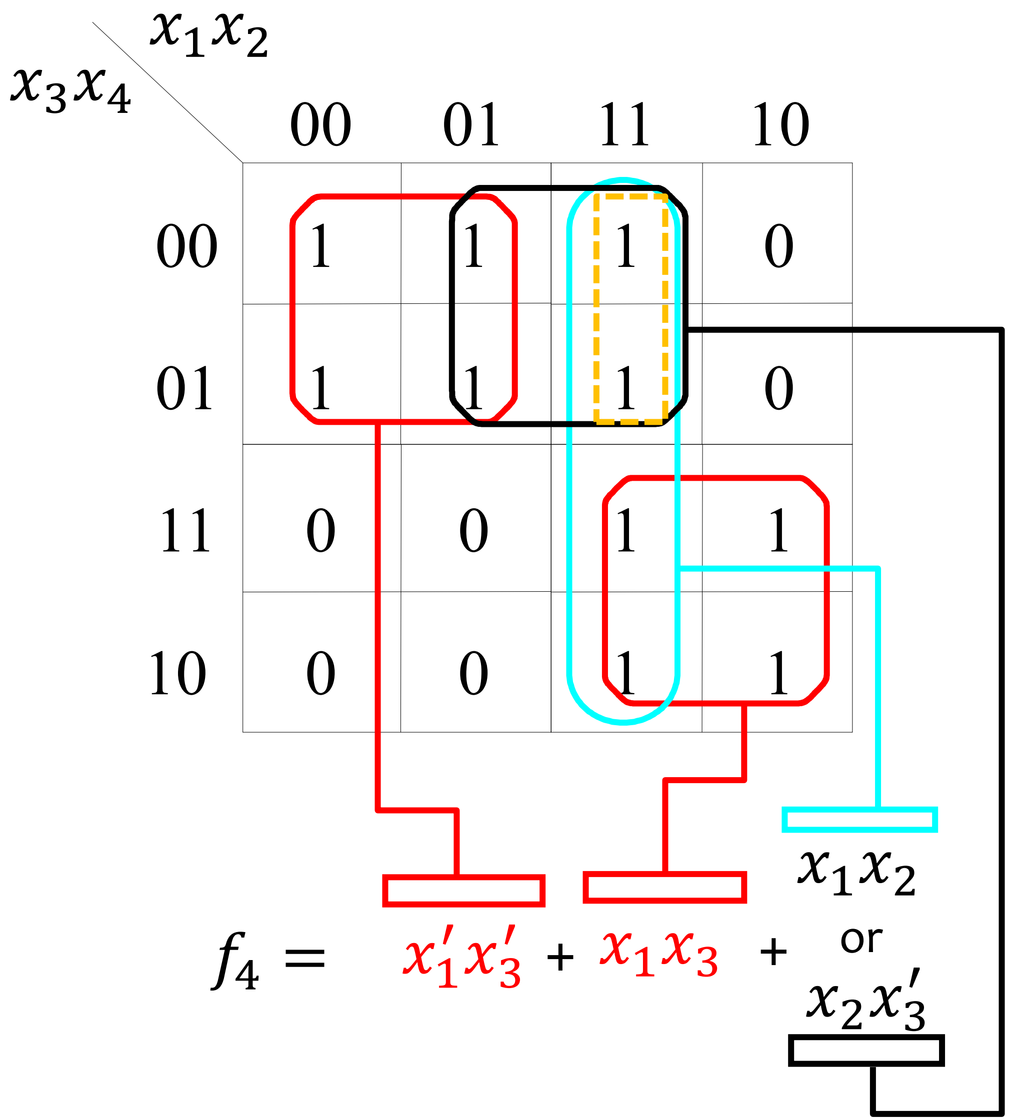
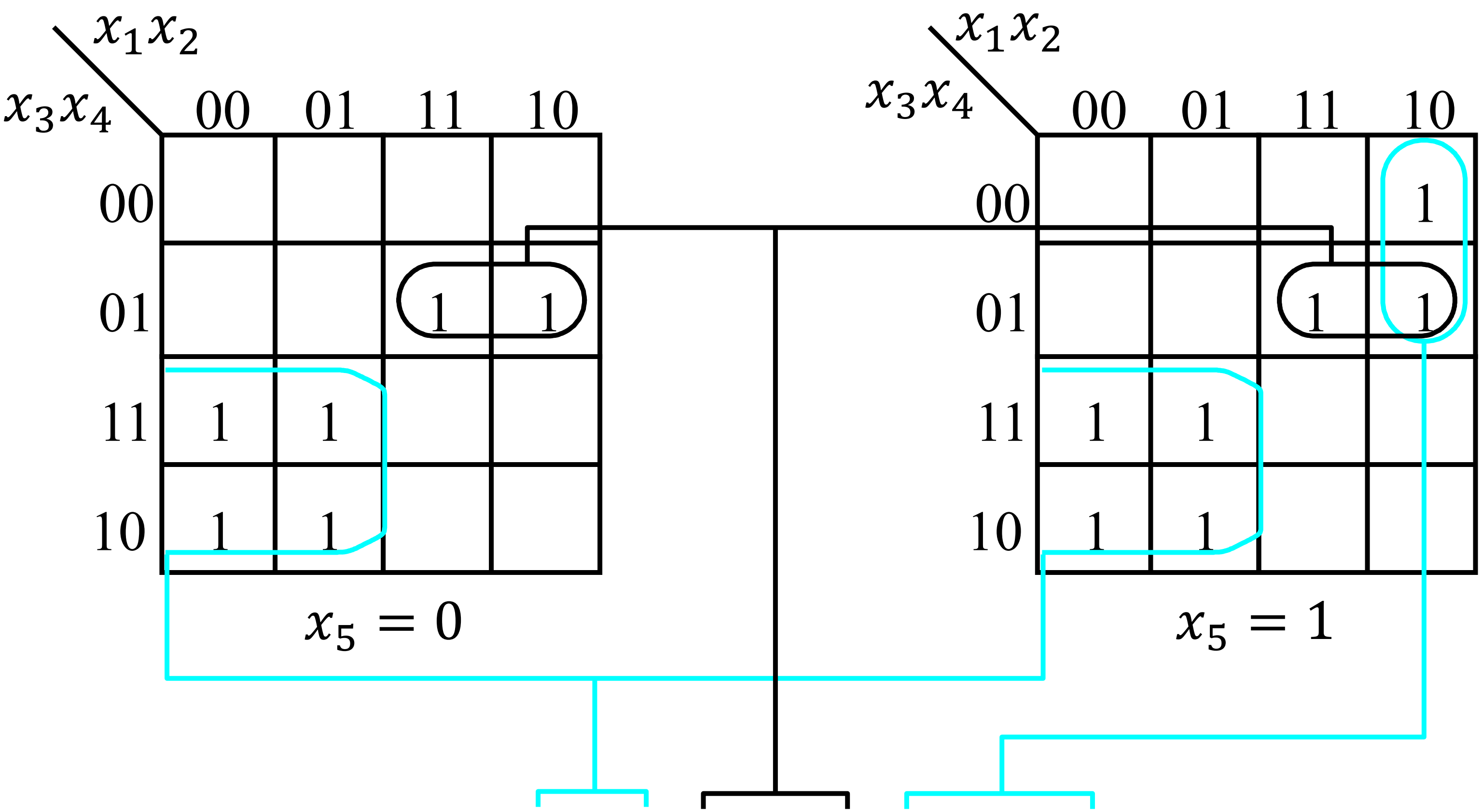


Figure 4.7. Examples of four-variable Karnaugh maps.

Five-Variables Map



$$f_1 = x_1'x_3 + x_1x_3'x_4 + x_1x_2'x_3'x_5$$

Figure 4.8. A five-variable Karnaugh map.

STRATEGY FOR MINIMIZATION

Terminology

- Literal: Each appearance of a variables is called a literal
- Implicant: A product term that indicates the input valuation(s) for which a given function is equal to 1 is called an implicant of the function
- Prime implicant: An implicant is called a prime implicant if it cannot be combined into another implicant that has fewer literals
- Cover: A collection of implicants that accounts for all variations for which a given function is equal to 1 is called a cover for that function
- Cost: The number of gates + the total number of input to all gates

Literal and Implicant

Literal

$x_1 x_2' x_3$: 3 literals

$x_1' x_3 x_4' x_6$: 4 literals

Implicant

- Basic implicant: minterm
- Implicants of Figure 4.9: 11 implicants
  - 5 minterms $(m_0, m_1, m_2, m_3, m_7)$
  - 5 pairs of minterms $(m_0+m_1, m_0+m_2, m_1+m_3, m_2+m_3, m_3+m_7)$
  - 1 implicant of 4 minterms $(m_0 + m_1 + m_2 + m_3)$

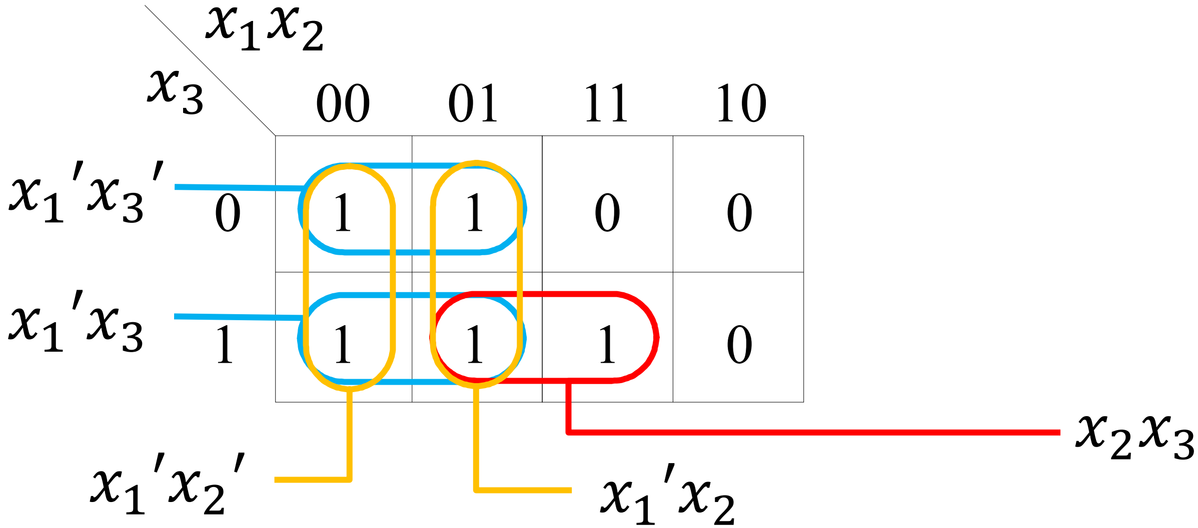


Figure 4.9. Three-variable function f $(x_1, x_2, x_3) = \Sigma$ m(0, 1, 2, 3, 7).

Prime Implicant and Cover

Prime Implicants

- x_1' *and* $x_2 x_3$

Cover

$$f = \sum(0,1,2,3,7)$$

$$f = x_1'x_2' + x_1'x_2 + x_2x_3$$

$$f = x_1' + x_2x_3$$

$x_3 \backslash x_1x_2$	00	01	11	10
0	1	1	0	0
1	1	1	1 m_7	0

x_1' x_2x_3

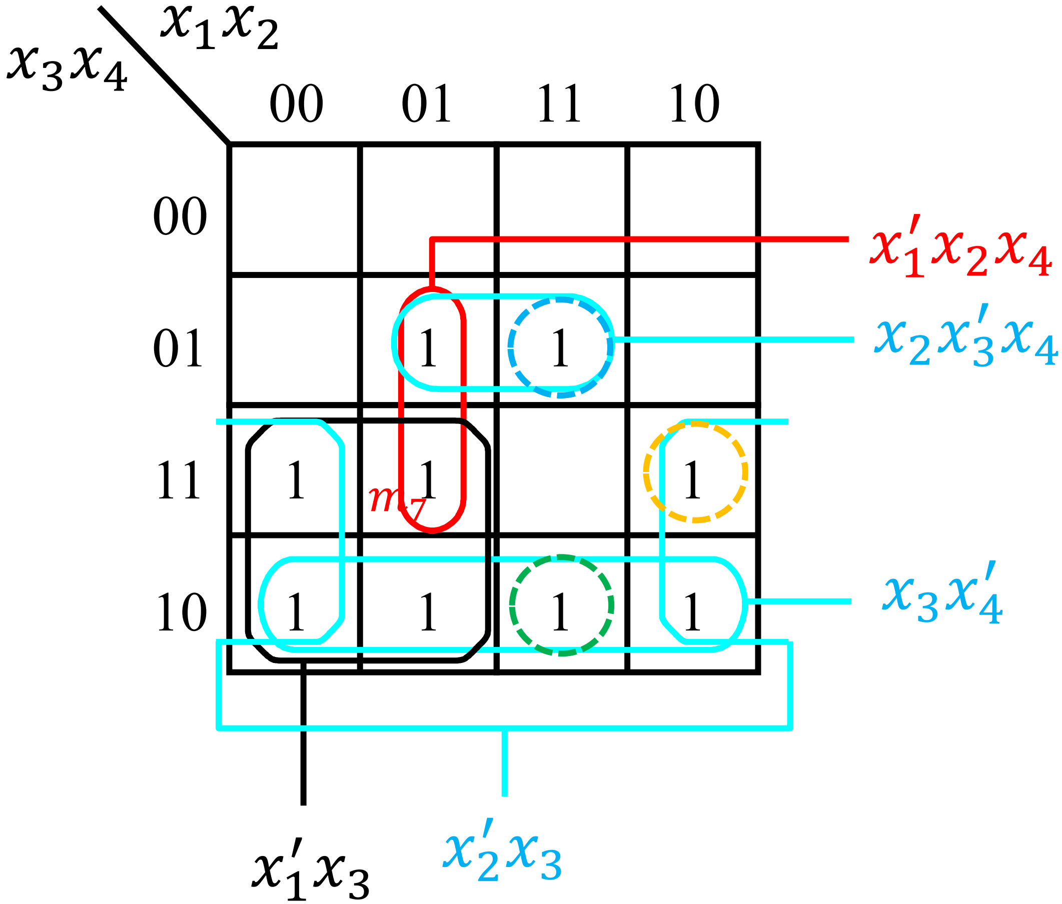
Figure 4.9. Three-variable function $f(x_1, x_2, x_3) = \Sigma\, m(0, 1, 2, 3, 7)$.

Cover and Cost

Cover	Cost
$f = \sum(0,1,2,3,7)$ Canonical SOP form	5 x 3-input AND, 1 x 5-input OR : 6 gates + 20 inputs = 26
$f = x_1'x_2' + x_1'x_2 + x_2x_3$	3 x 2-input AND, 1 x 3-input OR : 4 gates + 9 = 13
$f = x_1' + x_2x_3$ Optimal Solution !!!	1 x 2-input AND, 1 x 2-input OR : 2 gates + 4 inputs = 6

Minimization Procedure

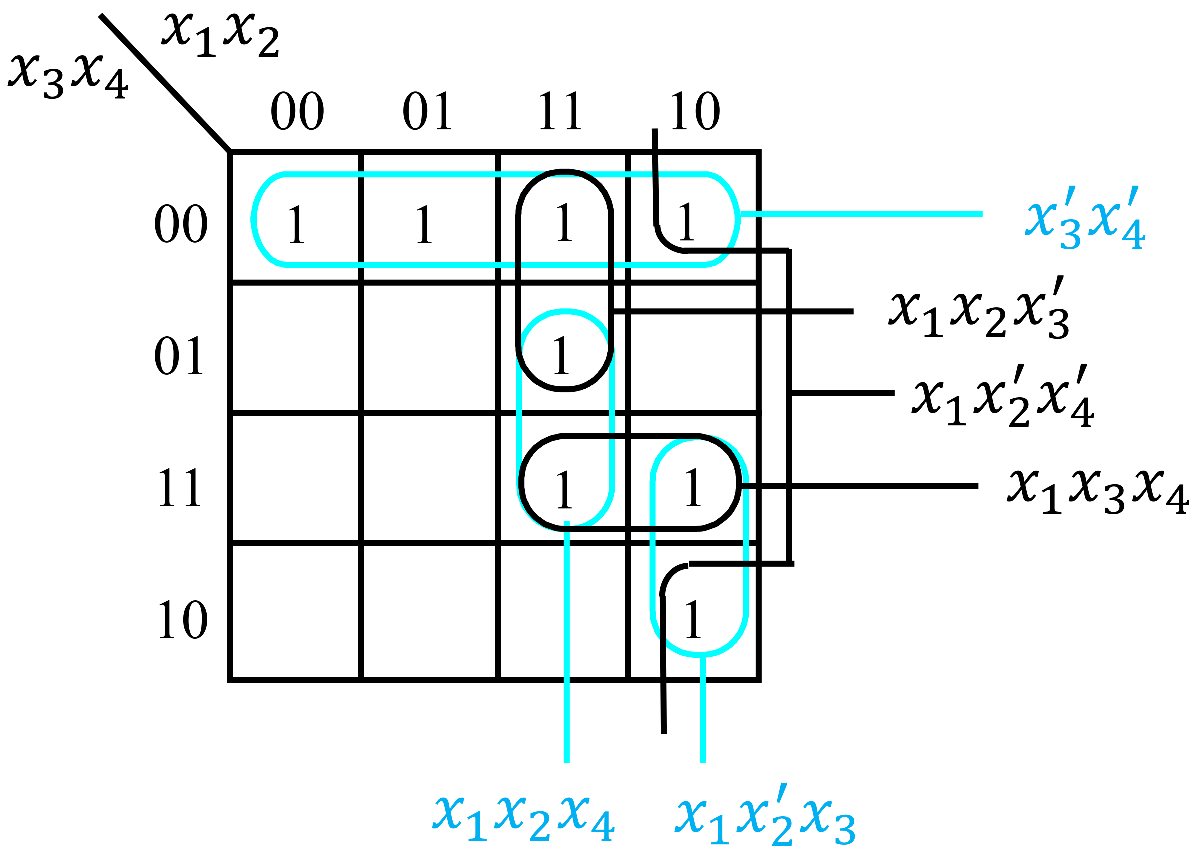
1. Generate all prime implicants for the given function f
2. Find the set of essential prime implicants
3. If the set of essential prime implicants covers all valuations for f = 1, then this set is the desired cover of f.



$$f = x_2'x_3 + x_3x_4' + x_2x_3'x_4 + x_1'x_3$$

$x_1'x_2x_4$

Figure 4.10. Four-variable function $f(x_1,\ldots,x_4) = \Sigma\, m(2, 3, 5, 6, 7, 10, 11, 13, 14)$.



$$\boxed{f = x_3'x_4' + x_1x_2x_4 + x_1x_2'x_3}$$

Figure 4.11. The function $f(x_1,\ldots,x_4) = \Sigma m(0, 4, 8, 10, 11, 12, 13, 15)$.

Cost
- 4 x 3-input AND
- 1 x 4-input OR
- 4 + (4 x 3 + 1 x 4) = 20

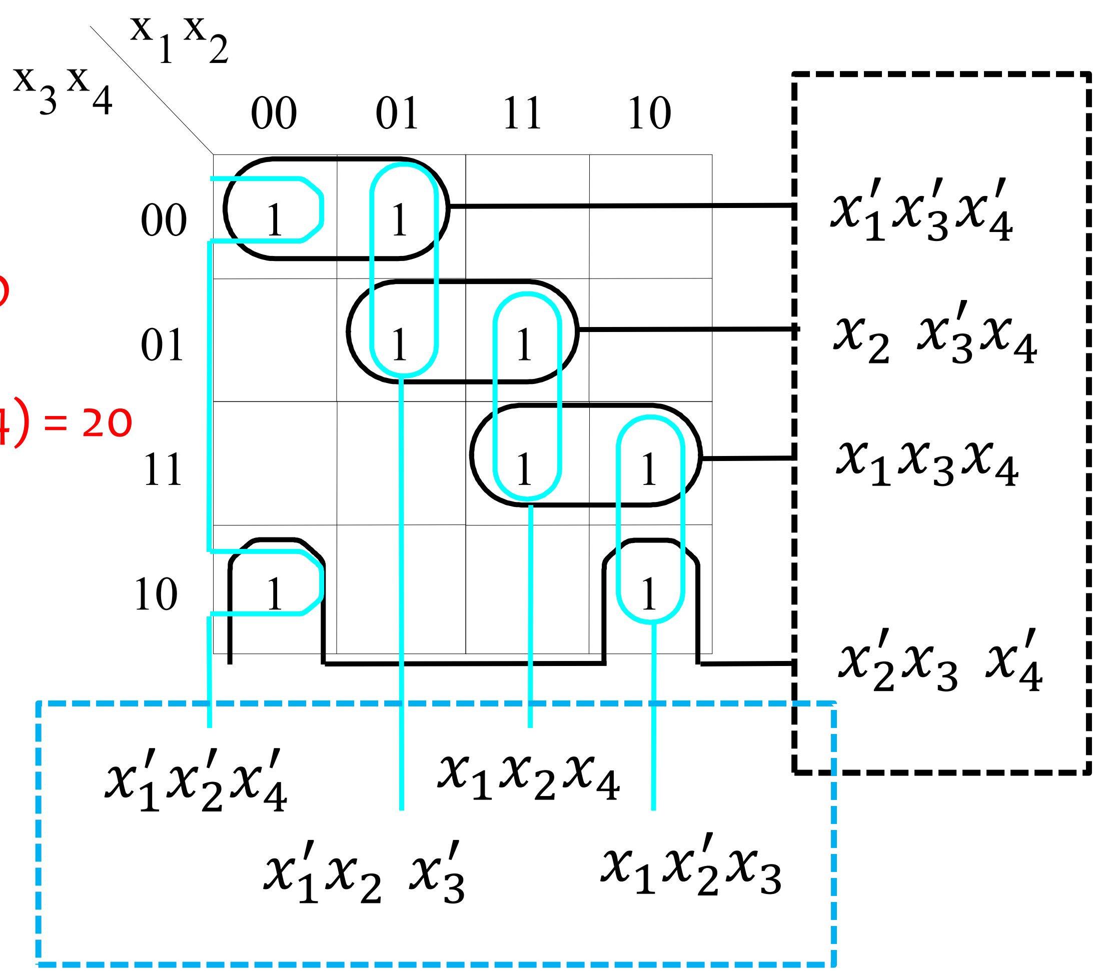


Figure 4.12. The function $f(x_1,\ldots,x_4) = \Sigma m(0, 2, 4, 5, 10, 11, 13, 15)$.

MINIMIZATION OF PRODUCT-OF-SUMS FORMS

Minimization of POS Forms

$$x + yz = (x + y)(x + z)$$

$x_3 \backslash x_1x_2$	00	01	11	10
0	1	1	0	0
1	1	1	1	0

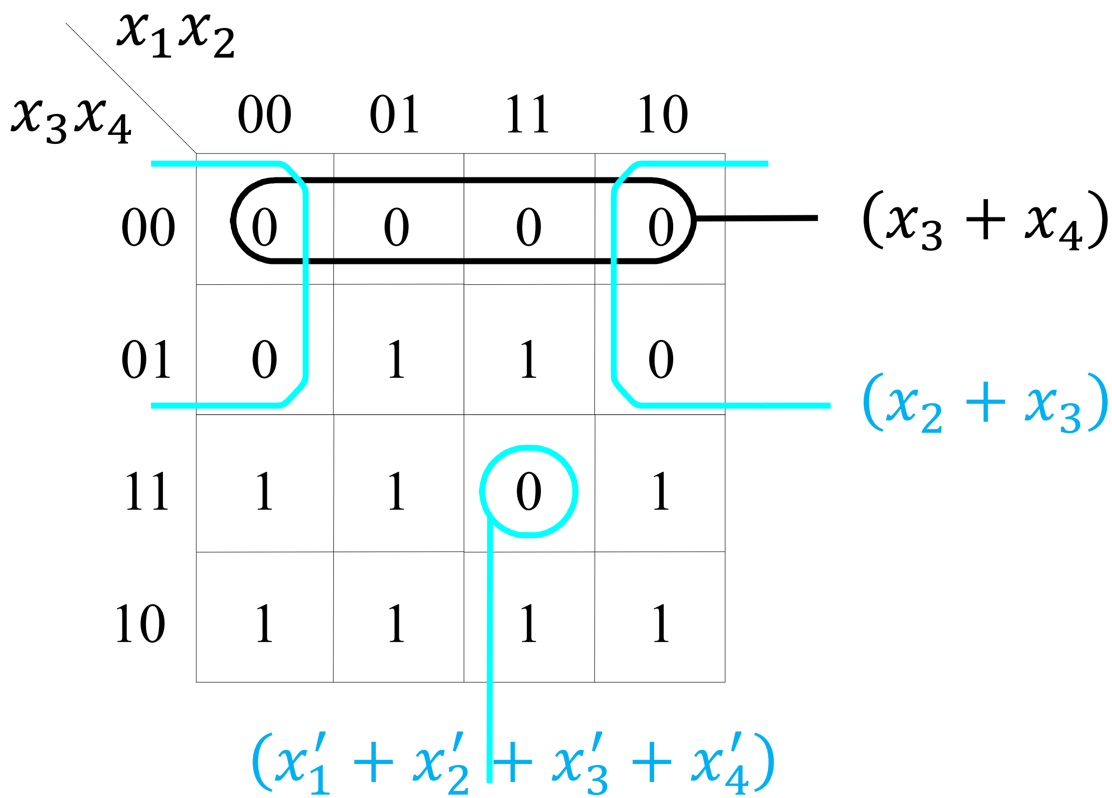
$$(x_1' + x_3) = \overset{M_6}{(x_1' + x_2' + x_3)}\overset{M_4}{(x_1' + x_2 + x_3)}$$
$$= (x_1' + x_3 + x_2')(x_1' + x_3 + x_2)$$
$$= (x_1' + x_3) + x_2'x_2$$

$$(x_1' + x_2) = \overset{M_5}{(x_1' + x_2 + x_3')}\overset{M_4}{(x_1' + x_2 + x_3)}$$
$$= (x_1' + x_2) + x_3'x_3$$

$$f = (x_1' + x_3)(x_1' + x_2)$$

Figure 4.13. POS minimization of $f(x_1, x_2, x_3) = \Pi M(4, 5, 6)$.

Minimization of POS Forms



$$f = (x_3 + x_4)(x_2 + x_3)(x_1' + x_2' + x_3' + x_4')$$

Figure 4.14. POS minimization of $f(x_1, \ldots, x_4) = \Pi\, M(0, 1, 4, 8, 9, 12, 15)$.

INCOMPLETELY SPECIFIED FUNCTION

Incompletely Specified Function

Don't care term : d

x_1	x_2	x_3	x_4	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0

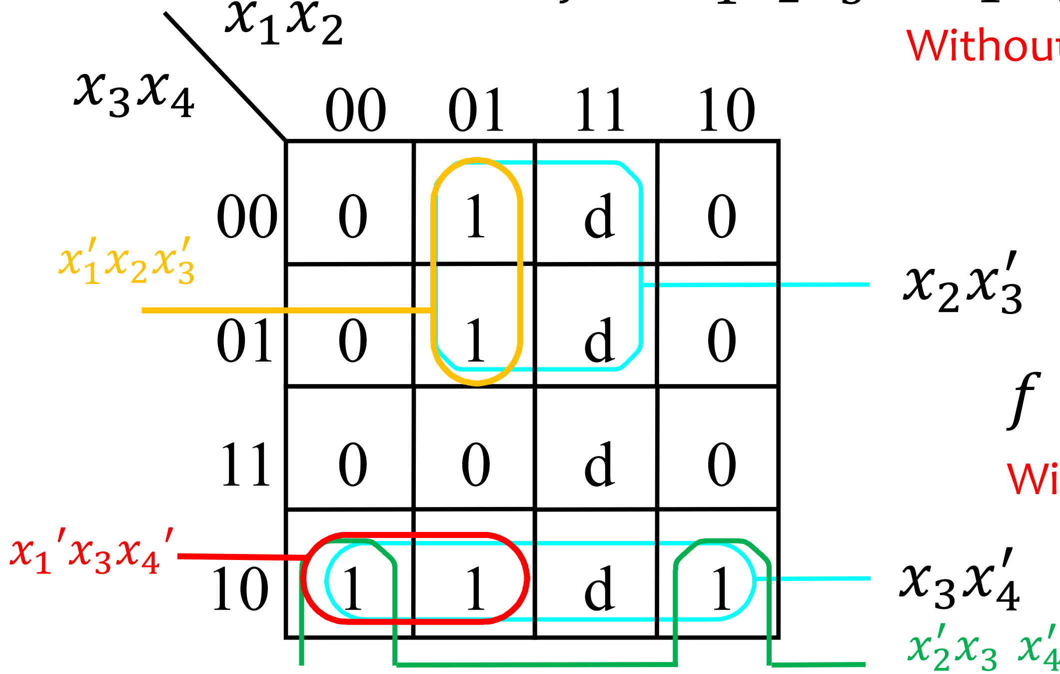
x_1	x_2	x_3	x_4	f
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	d
1	1	0	1	d
1	1	1	0	d
1	1	1	1	d

Figure 4.15. Two implementations of the function $f(x_1,\ldots,x_4) = \Sigma m(2, 4, 5, 6, 10) + D(12, 13, 14, 15)$

Incompletely Specified Function

$$f = x_1'x_2x_3' + x_1'x_3x_4' + x_2'x_3\ x_4'$$

Without don't care terms



$$f = x_2x_3' + x_3x_4'$$

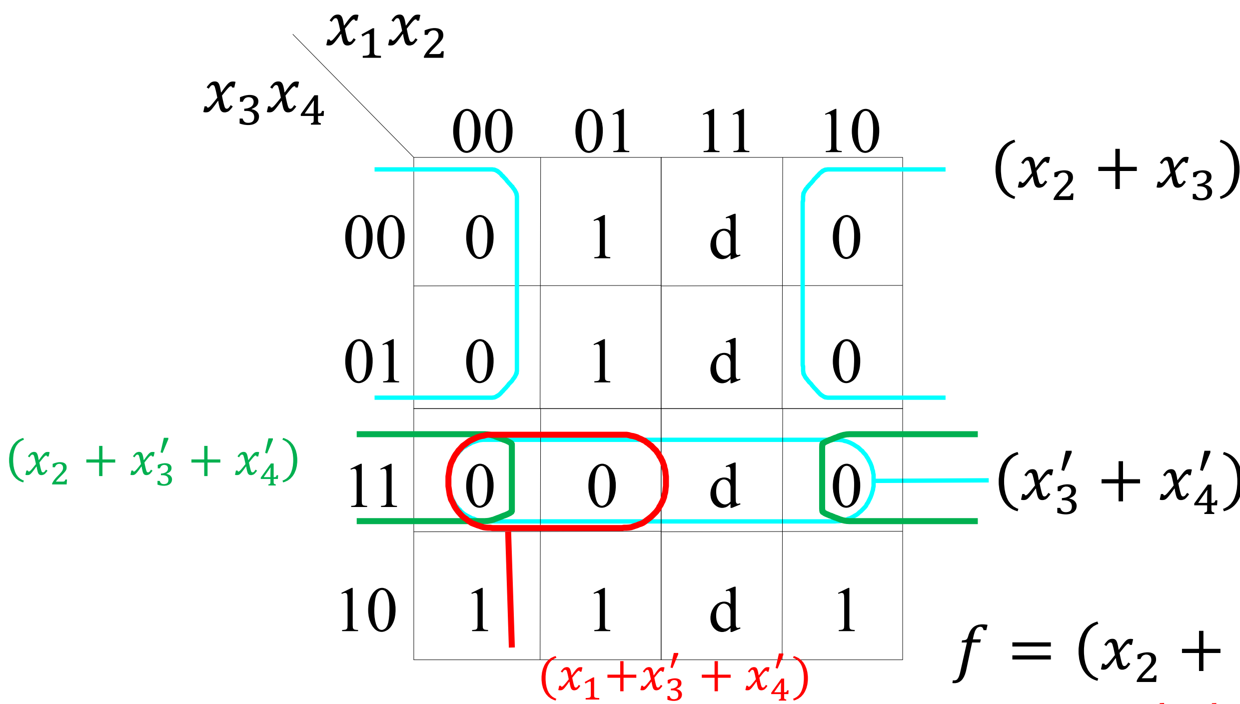
With don't care terms

(a) SOP implementation

Figure 4.15. Two implementations of the function f $(x_1, \ldots, x_4) = \Sigma\, m(2, 4, 5, 6, 10) + D(12, 13, 14, 15)$.

Incompletely Specified Function

$f = (x_2 + x_3)(x_1 + x_3' + x_4')\,(x_2 + x_3' + x_4')$ Without don't care terms



$f = (x_2 + x_3)(x_2 + x_3)$

With don't care terms

(b) POS implementation

Figure 4.15. Two implementations of the function f $(x_1,\ldots, x_4) = \Sigma\, m(2, 4, 5, 6, 10) + D(12, 13, 14, 15)$.

MULTIPLE-OUTPUT CIRCUITS

Multiple-Output Circuits

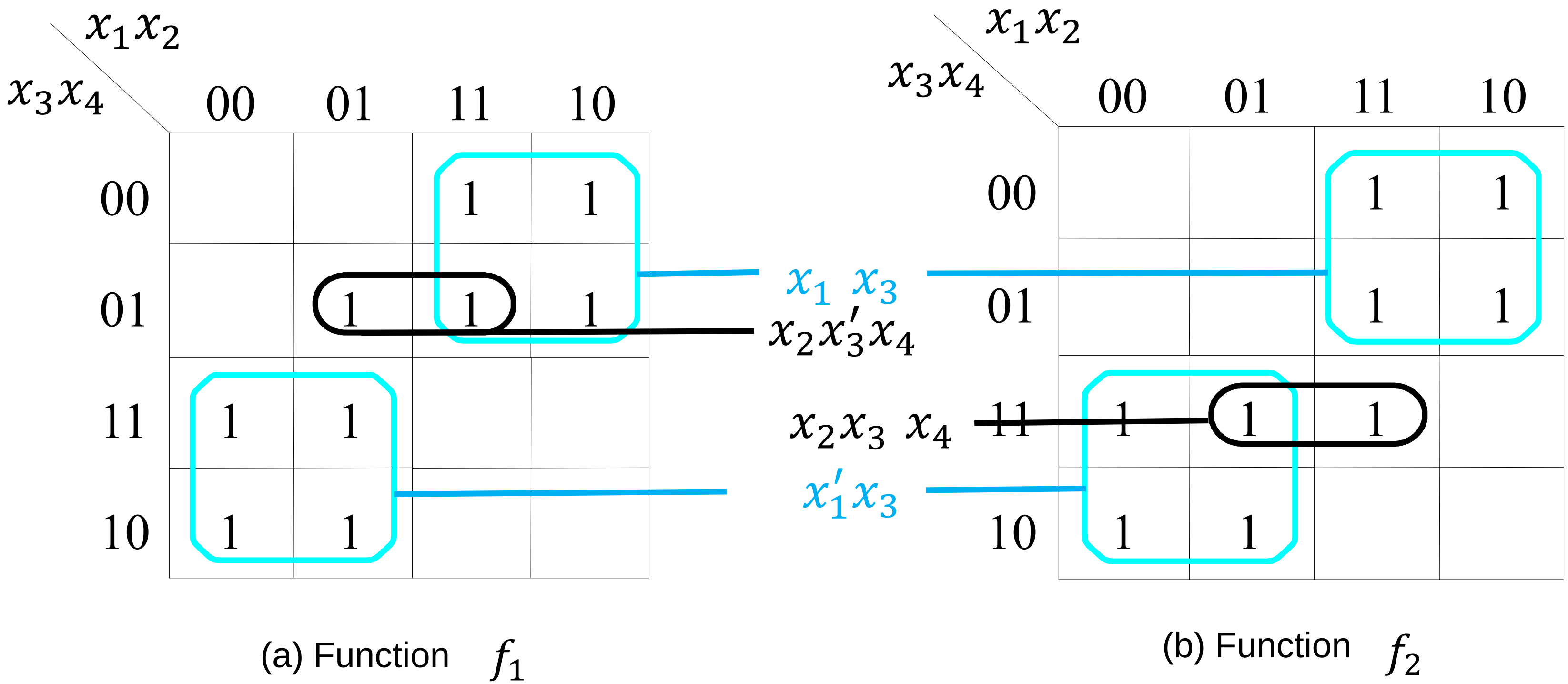
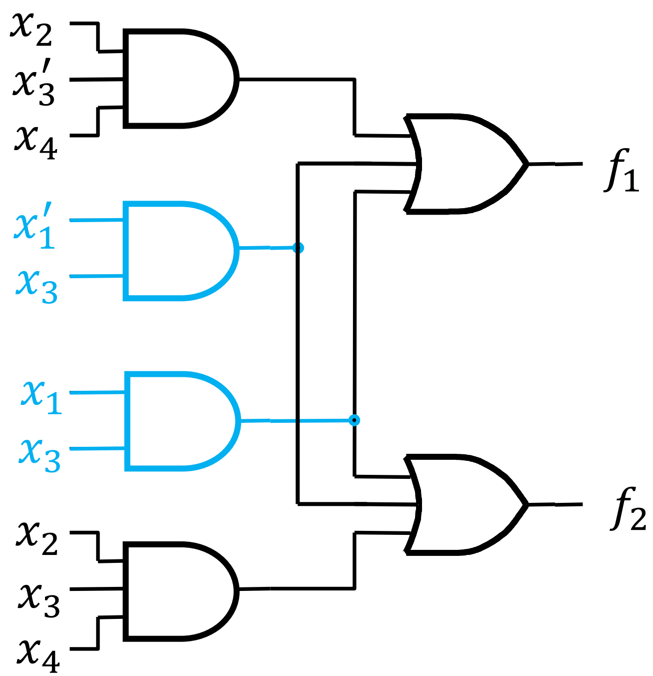


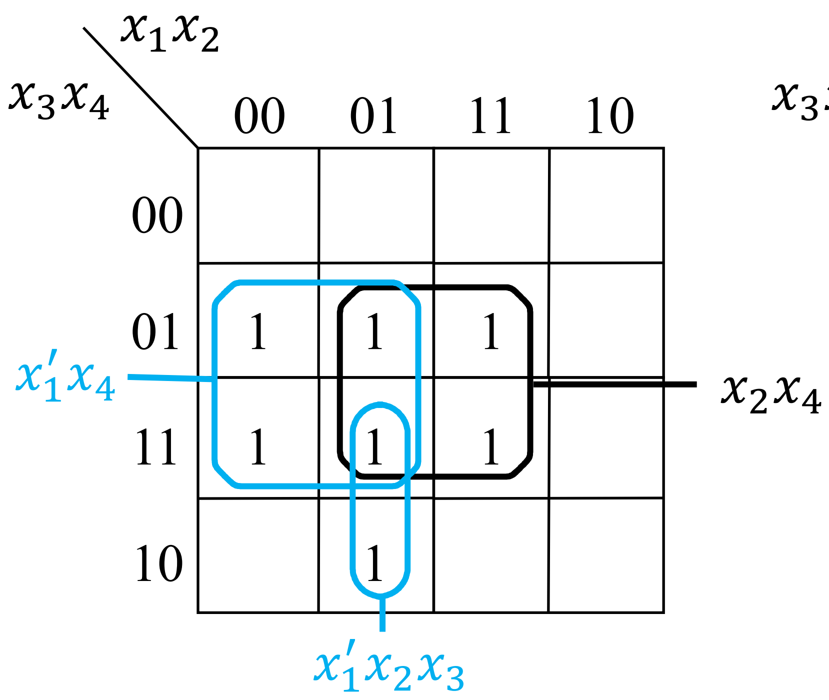
Figure 4.16. An example of multiple-output synthesis.

Multiple-Output Circuits

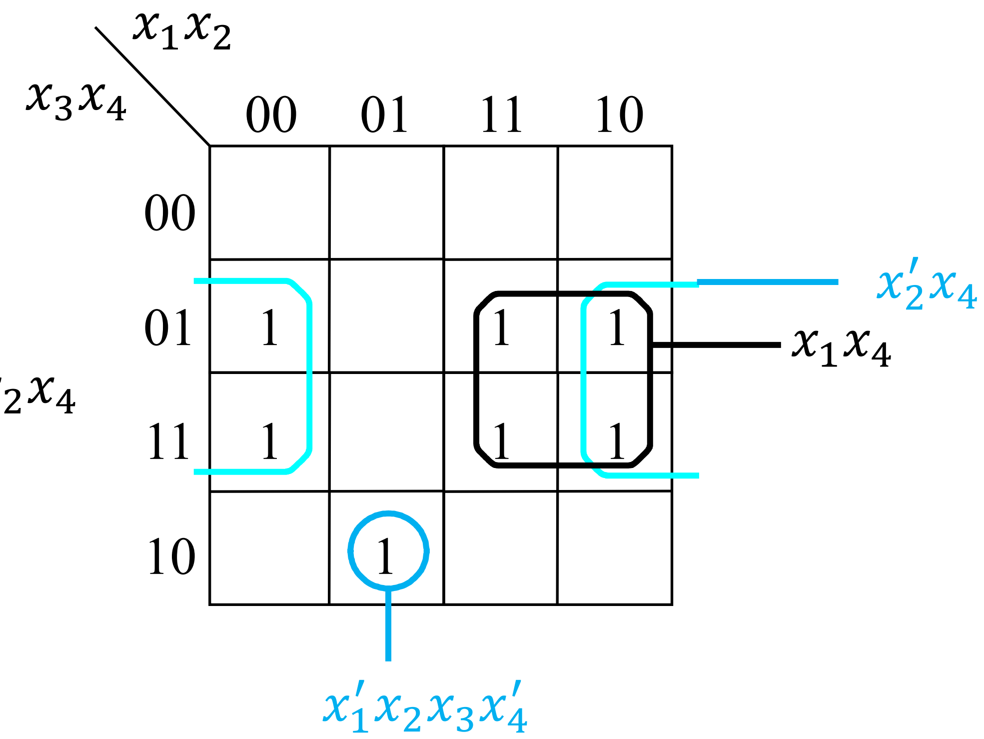


(c) Combined circuit for f_1 *and* f_2

Figure 4.16. An example of multiple-output synthesis.



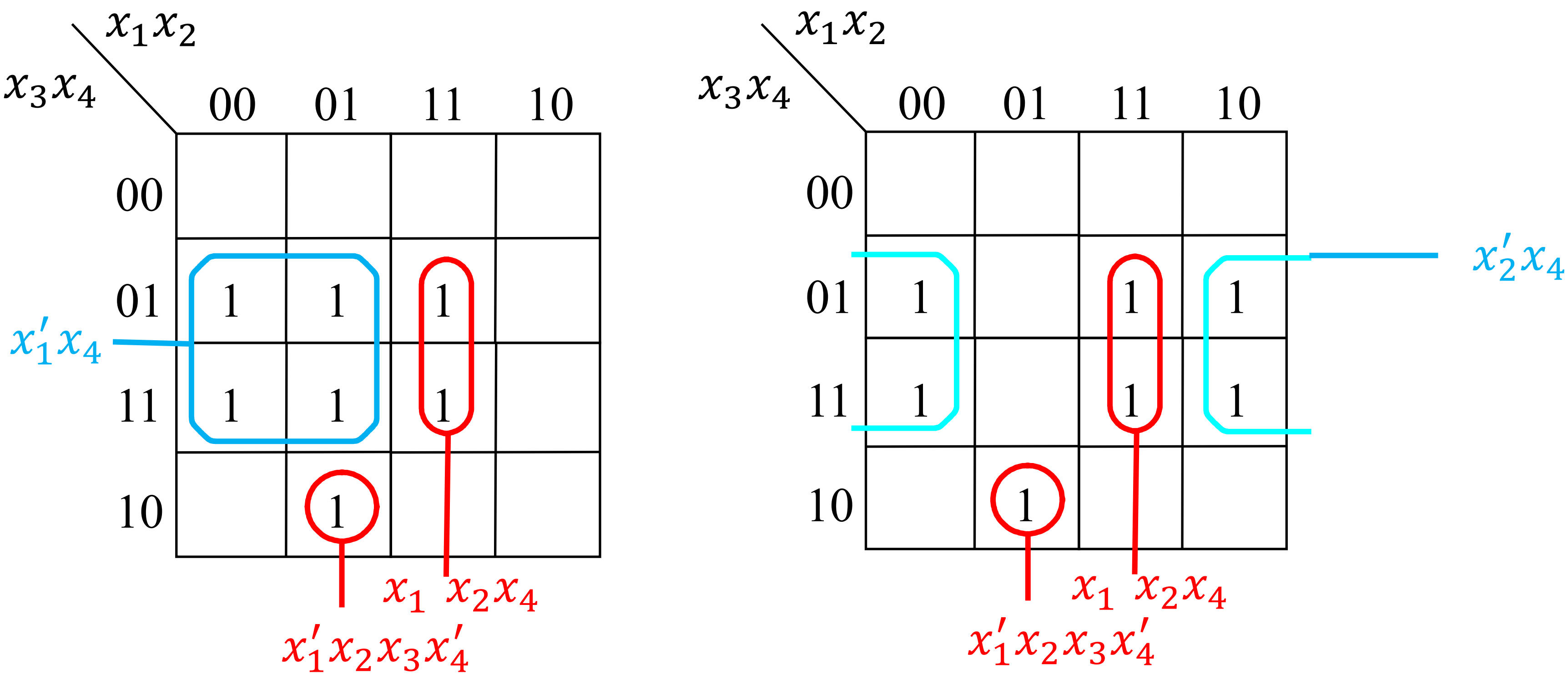
(a) Optimal realization of f_3



(b) Optimal realization of f_4

$$f_3 = x_2x_4 + x_1'x_4 + x_1'x_2x_3 \qquad f_4 = x_1x_4 + x_2'x_4 + x_1'x_2x_3x_4'$$

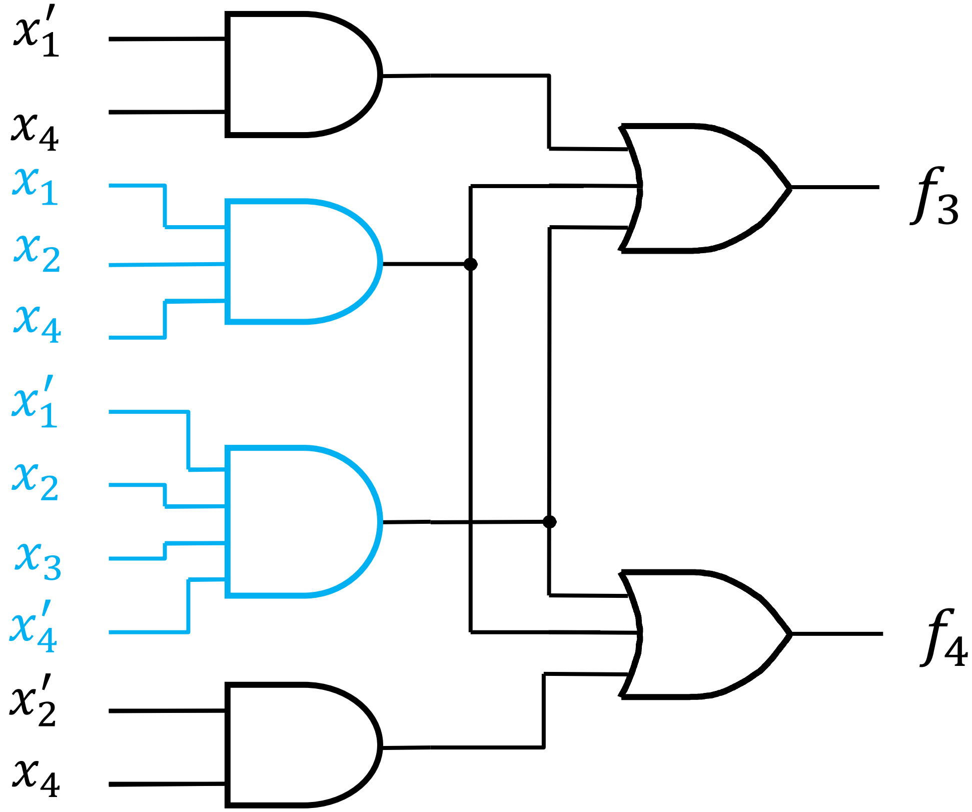
Figure 4.17. An example of multiple-output synthesis.



$$f_3 = x_1x_2x_4 + x_1'x_4 + x_1'x_2x_3x_4' \qquad f_4 = x_1x_2x_4 + x_2'x_4 + x_1'x_2x_3x_4'$$

(c) Optimal realization of f_3 and f_4 together

Figure 4.17. An example of multiple-output synthesis.



(d) Combined circuit for f_3 and f_4

Figure 4.17. An example of multiple-output synthesis.

Cost for Multiple-Output

$f_3 = x_2x_4 + x_1'x_4 + x_1'x_2x_3$

Cost for f_3

- 2 x 2-input ANDs, 1 x 3-input AND, 1 x 3-input OR
- 4 + (2 x 2 + 1 x 3 + 1 x 3) = 14

$f_4 = x_1x_4 + x_2'x_4 + x_1'x_2x_3x_4'$

Cost for f_4

- 2 x 2-input ANDs, 1 x 4-input AND, 1 x 3-input OR
- 4 + (2 x 2 + 1 x 4 + 1 x 3) = 15

Total cost : 14 + 15 = 29

Cost for Multiple-Output

$f_3 = x_1x_2x_4 + x_1'x_4 + x_1'x_2x_3x_4'$

Cost for f_3

- 1 x 2-input AND, 1 x 3-input AND, 1 x 4-input AND, 1 x 3-input OR
- 4 + (1 x 2 + 1 x 3 + 1 x 4 + 1 x 3) = 16

$f_4 = x_1x_2x_4 + x_2'x_4 + x_1'x_2x_3x_4'$

Cost for f_4

- 1 x 2-input AND, 1 x 3-input AND, 1 x 4-input AND, 1 x 3-input OR
- 4 + (1 x 2 + 1 x 3 + 1 x 4 + 1 x 3) = 16

Cost for f_3 and f_4 together

- 2 x 2-input ANDs, 1 x 3-input AND, 1 x 4-input AND, 2 x 3-input OR
- 6 + (2 x 2 + 1 x 3 + 1 x 4 + 2 x 3) = 23

Total cost : 16 + 16 - 9 = 23

$$\frac{(29-23)}{29} \times 100\ \% = 21\%$$

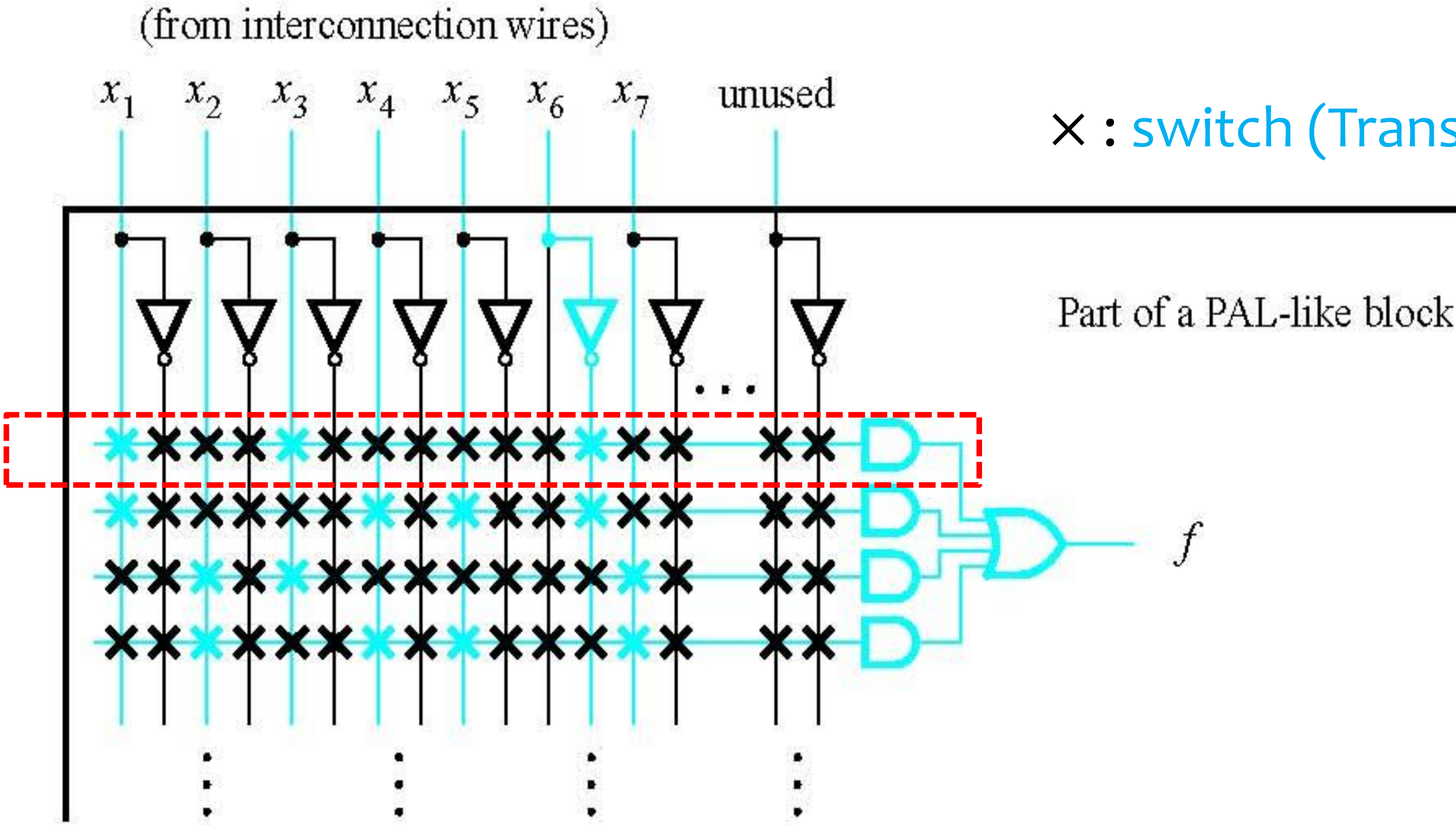
MULTILEVEL SYNTHESIS

Multilevel Synthesis

- Fan-in problem
  - Limited number of inputs
  - Fan-out problem: Limited number of outputs
- Implementation
  - CPLD: two-level logic expression (Sum of Products)
  - FPGA: multilevel logic expression
- Two Important Techniques for Synthesis of Multilevel Circuits
  - Factoring
  - Functional Decomposition

Standard SOP form

× : switch (Transistor)



$$f(x_1, \cdots, x_7) = x_1 x_3 x_6' + x_1 x_4 x_5 x_6' + x_2 x_3 x_7 + x_2 x_4 x_5 x_7$$

Figure 4.18. Implementation in a CPLD.

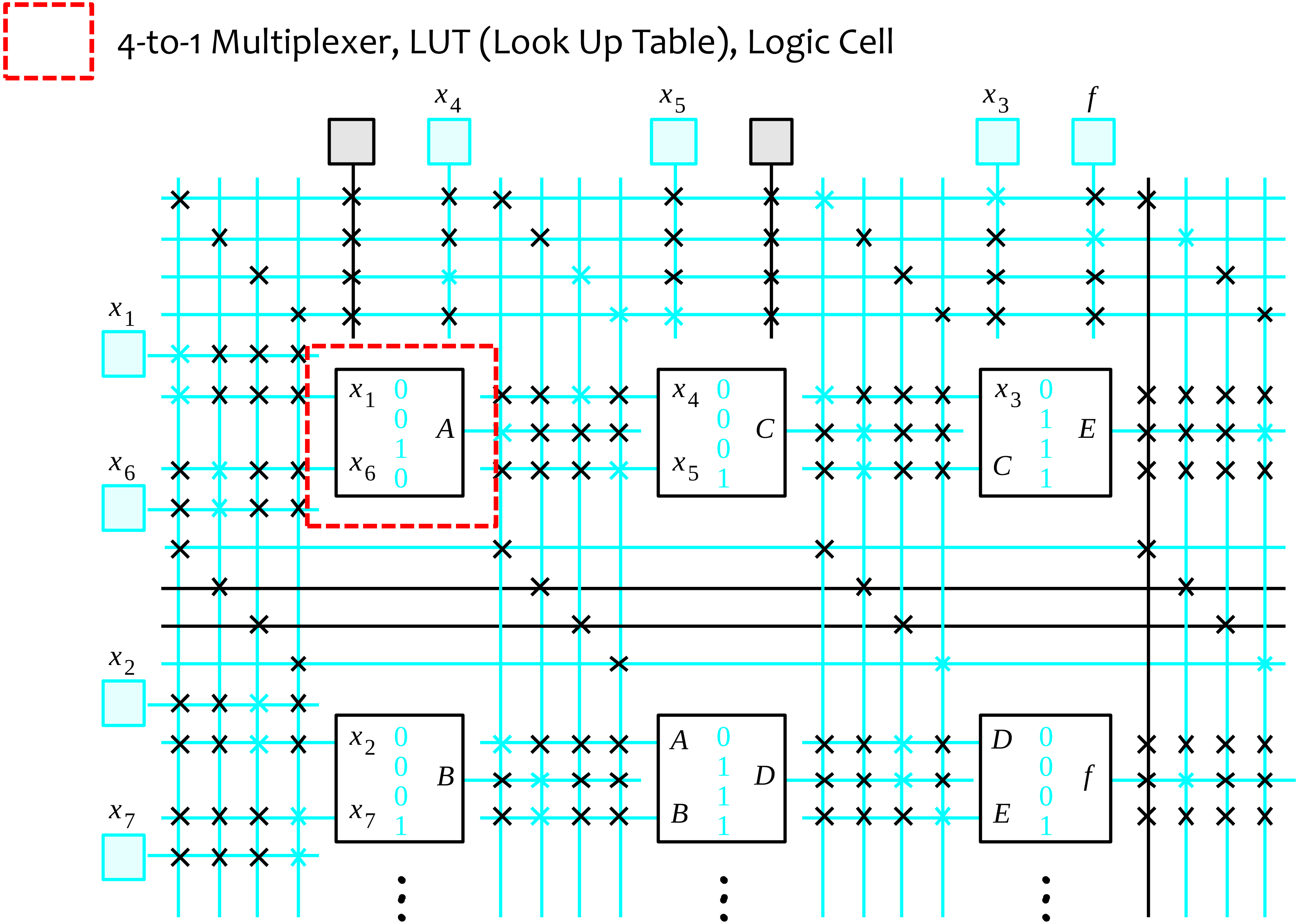
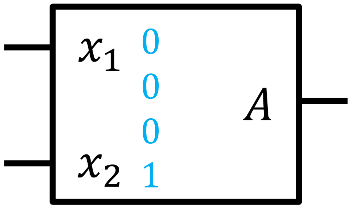


Figure 4.19. Implementation in an FPGA.

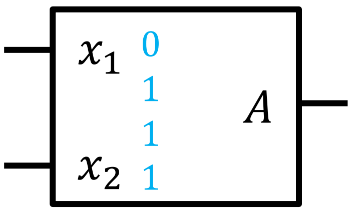
LUT operation

AND



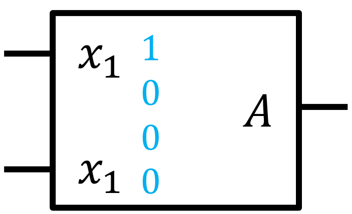
$A = x_1 x_2$

OR



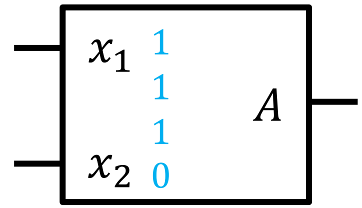
$A = x_1 + x_2$

NOT



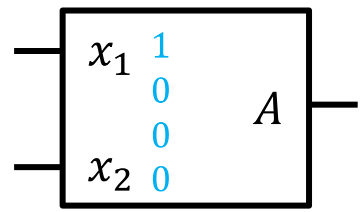
$A = x_1'$

NAND



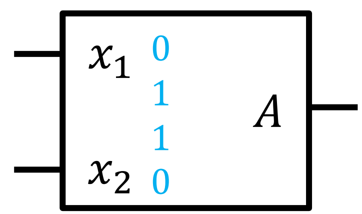
$A = (x_1 x_2)'$

NOR



$A = (x_1 + x_2)'$

X-OR



$A = x_1 \oplus x_2$

Factoring

$$f = x_1 x_6' (x_3 + x_4 x_5) + x_2 x_7 (x_3 + x_4 x_5)$$

2 x 3-input ANDs, 2 x 2-input ANDs, 3 x 2-input OR

In the previous slide, the circuit has a maximum fan-in of two, 2-input LUTs

By the distributive property,

$$\boxed{f = (x_3 + x_4 x_5)\,(x_1 x_6' + x_2 x_7)}$$

4 x 2-input ANDs, 2 x 2-input ORs

Factoring

$$f = x_1x_2x_3x_4x_5x_6x_7 = (x_1x_2x_3x_4)x_5x_6x_7$$

1 x 7 input AND = 2 x 4-input ANDs

If Fan-in : Max. 4

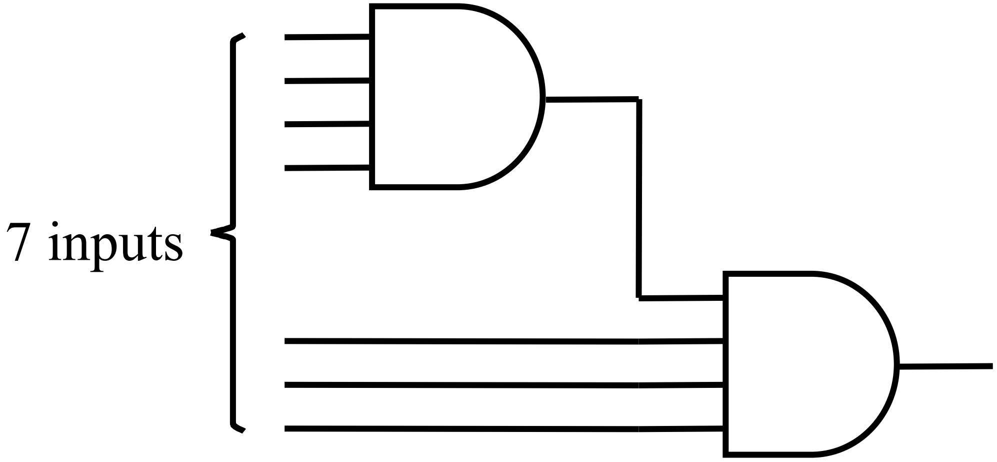


Figure 4.20. Using four-input AND gates to realize a seven-input product term.

$$f = x_1 x_2' x_3 x_4' x_5 x_6 + x_1 x_2 x_3' x_4' x_5' x_6$$

2 x 6-input ANDs, 1 x 2-input OR

If Fan-in : Max. 4

$$f = x_1 x_4' x_6 (x_2' x_3 x_5 + x_2 x_3' x_5')$$

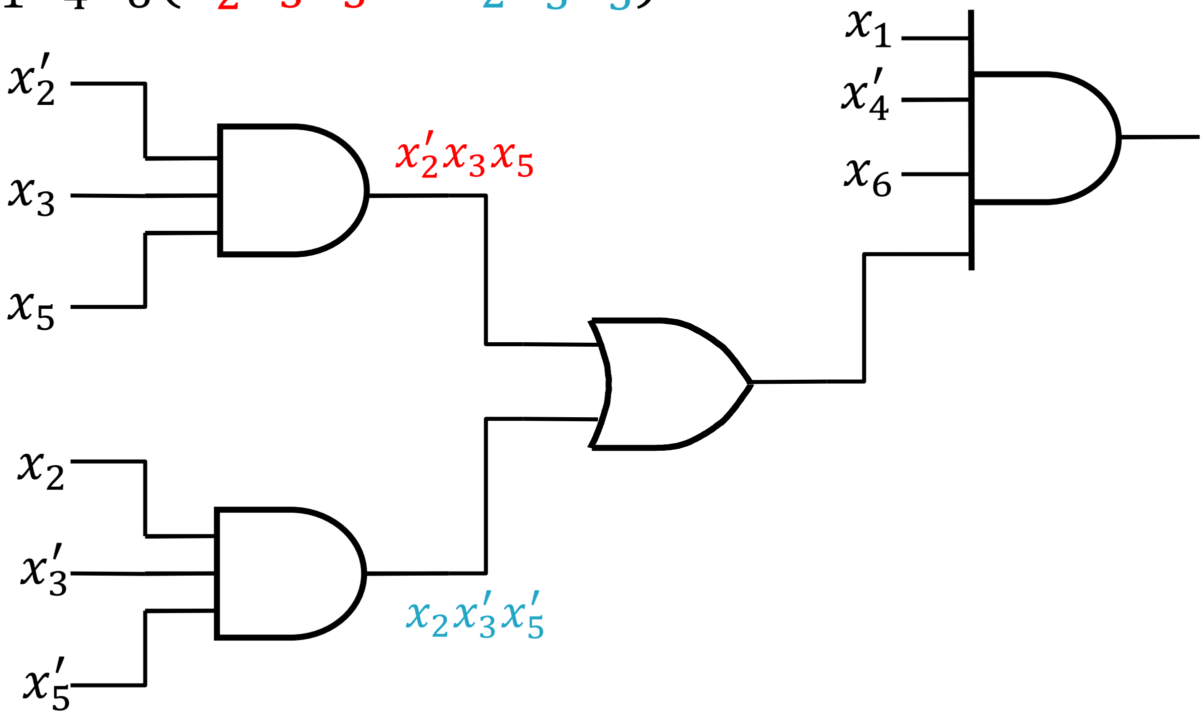


Figure 4.21. A factored circuit.

Example 4.5

For four input system, x_1, x_2, x_3 and x_4

(1) f_1 = 1 if at least one of x_1 and x_2 =1, both x_3 and x_4 =1; $(x_1 + x_2)x_3x_4$

f_1 = 1 if x_1=x_2=0 and either x_3 or x_4=1 $x_1'x_2'(x_3 + x_4)$

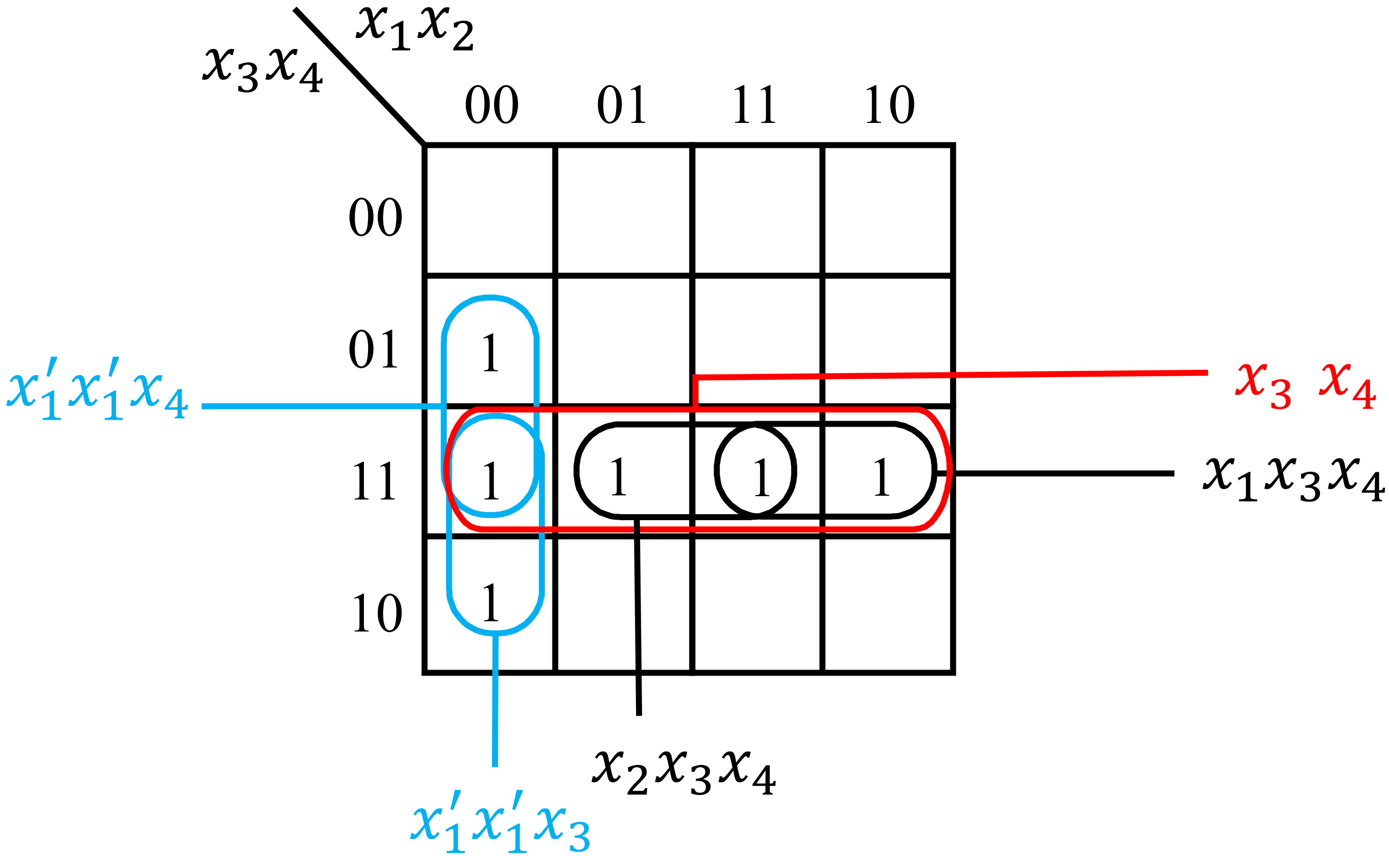
(2) f_2 = 1 in all cases except when both x_1 and x_1 = 0 $x_1'x_2'$ or when both x3 and x4 = 0 $x_3'x_4'$

$$f_1 = (x_1 + x_2)x_3x_4 + x_1'x_2'(x_3 + x_4)$$

$$f_2' = x_1'x_2' + x_3'x_4'$$

$$f_2 = (x_1'x_2' + x_3'x_4')'$$

$$f_2 = (x_1 + x_2)(x_3 + x_4)$$



$$f_1 = (x_1 + x_2)x_3x_4 + x_1'x_2'(x_3 + x_4)$$

$$= x_3x_4 + x_1'x_2'(x_3 + x_4)$$

$$= x_3x_4 + (x_1 + x_2)'\,(x_3 + x_4)$$

Example 4.5

$$f_1 = x_3x_4 + (x_1 + x_2)'(x_3 + x_4)$$

$$f_2 = (x_1 + x_2)(x_3 + x_4)$$

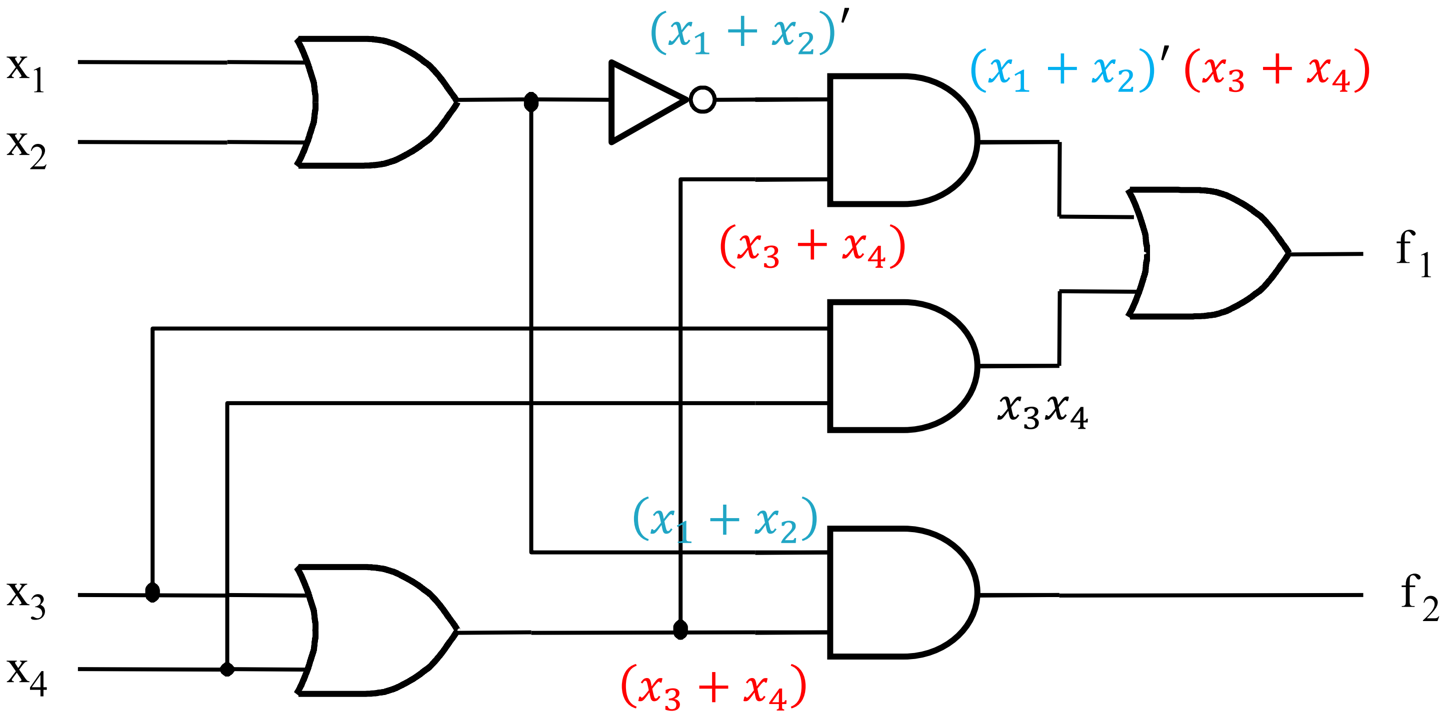


Figure 4.22. Circuit for Example 4.5.

Functional Decomposition

- Example 4.6

$$f = x_1'x_2x_3 + x_1x_2'x_3 + x_1x_2x_4 + x_1'x_2'x_4$$

- Minimum cost SOP expression
- 4 x 3-input ANDs, 1 x 4-input OR
- Cost: 5 + (4 x 3 + 1 x 4) = 21

$$f = (x_1'x_2 + x_1x_2')x_3 + (x_1x_2 + x_1'x_2')x_4$$

$g(x_1, x_2) = x_1'x_2 + x_1x_2'$ X-OR

$g'(x_1, x_2) = x_1x_2 + x_1'x_2'$ X-NOR

- 6 x 2-input ANDs, 3 x 2-input OR
- Cost: 9 + (6 x 2 + 3 x 2) = 27

- 4 x 2-input ANDs, 2 x 2-input OR
- Cost: 6 + (4 x 2 + 2 x 2) = 18

$$f = gx_3 + g'x_4$$

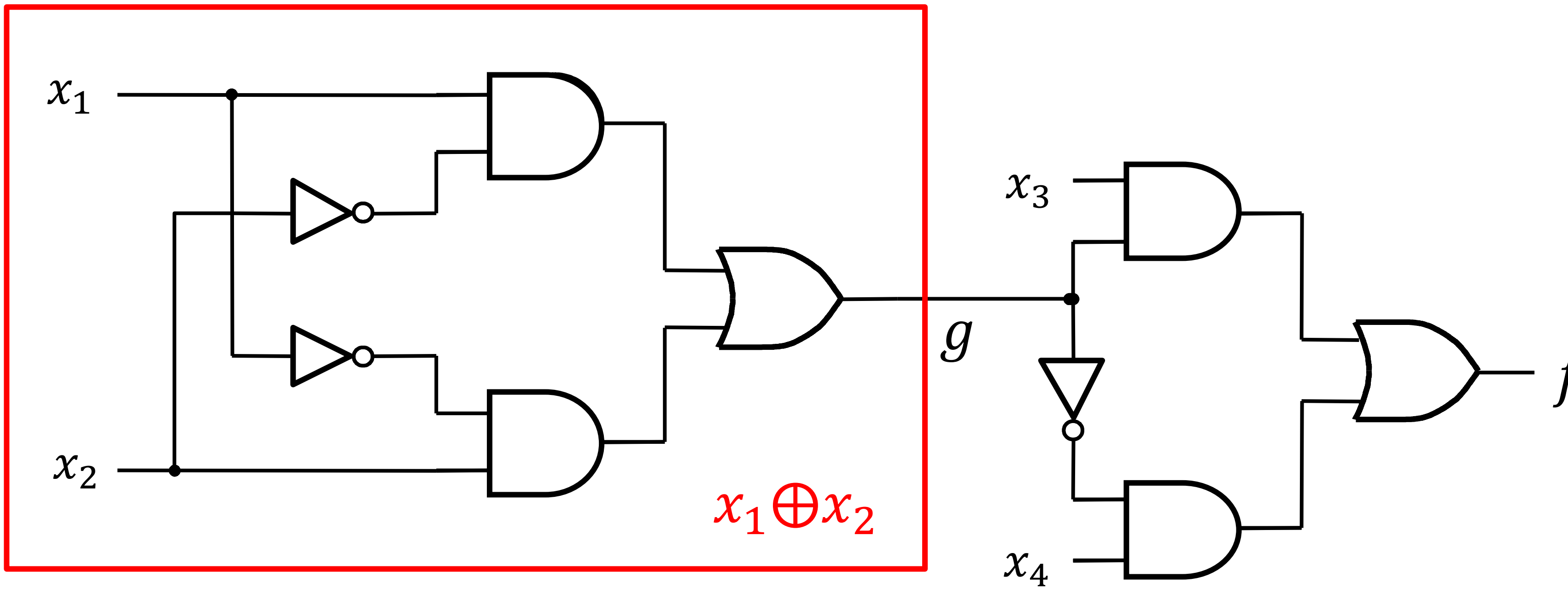
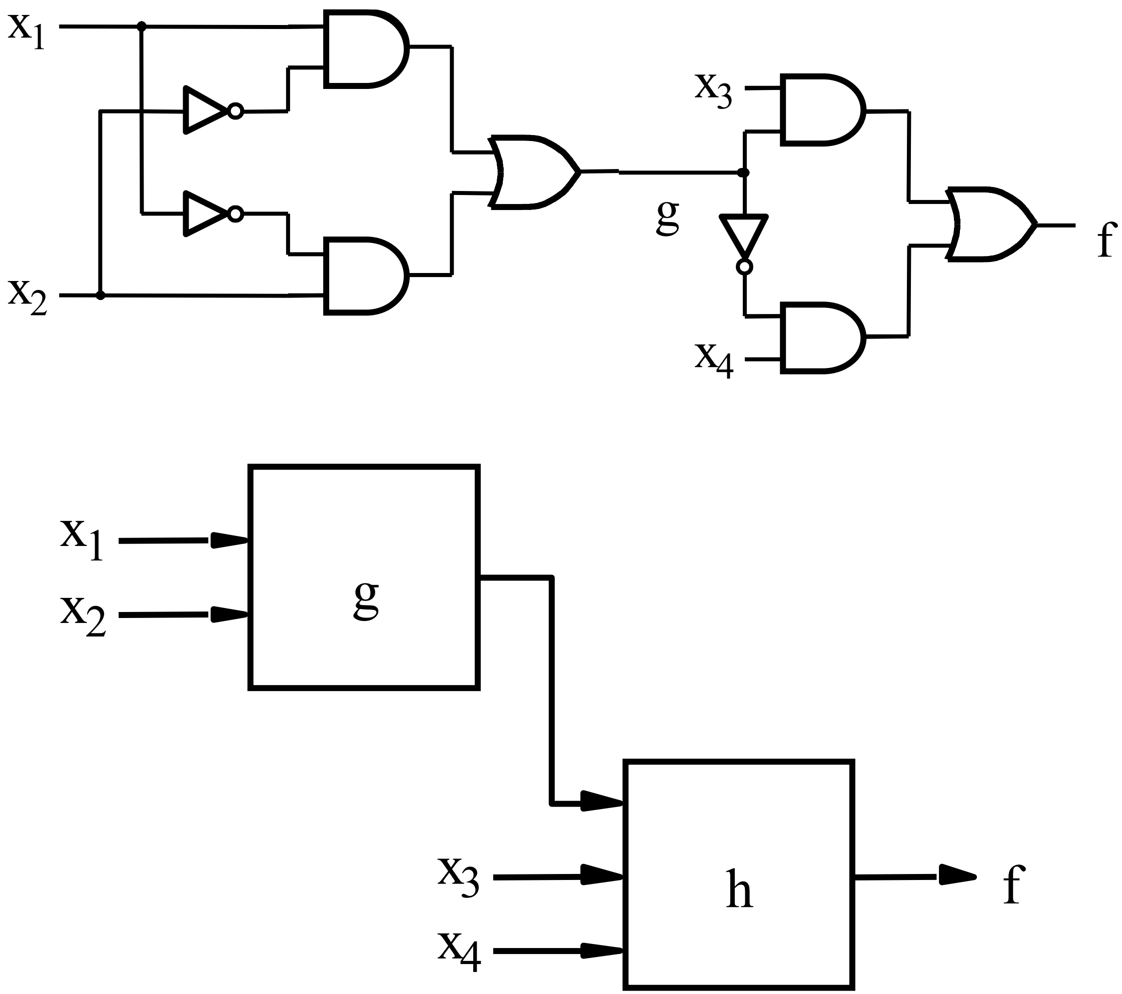


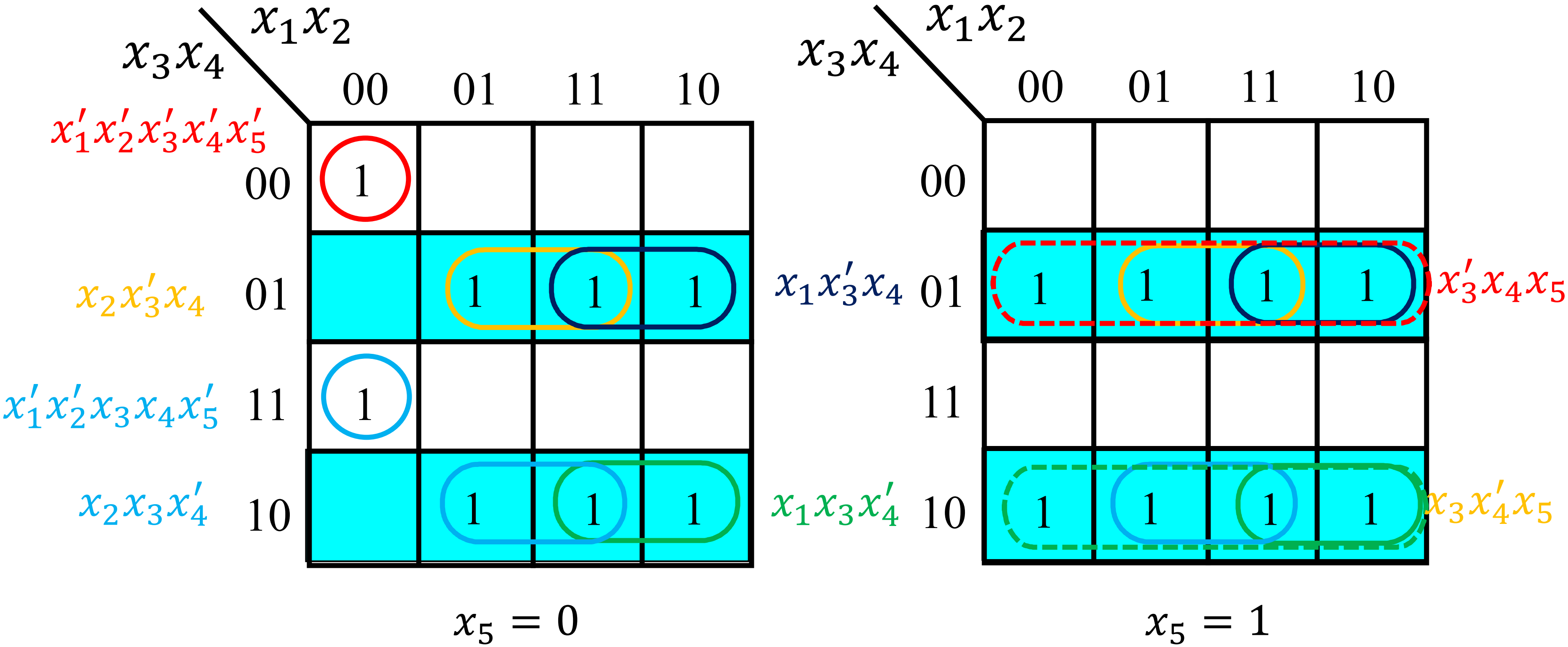
Figure 4.23. Logic circuit for Example 4.6.



$$f(x_1, x_2, x_3, x_4) = h[g(x_1, x_2), x_3, x_4]$$

Figure 4.24. The structure of decomposition in Example 4.6.

Example 4.7



$$f = x_1x_3'x_4 + x_1x_3x_4' + x_2x_3'x_4 + x_2x_3x_4' + x_3'x_4x_5 + x_3x_4'x_5 + x_1'x_2'x_3'x_4'x_5' + x_1'x_2'x_3x_4x_5'$$

6 x 3-input ANDs, 2 x 5-input ANDs, 1 x 8-input OR

Cost: 9 + (6 x 3 + 2 x 5 + 1 x 8) = 45

Functional Decomposition

x_3x_4 \ x_1x_2	00	01	11	10
00	1			
01				
11	1			
10				

$x_5 = 0$

x_3x_4 \ x_1x_2	00	01	11	10
00				
01				
11				
10				

$x_5 = 1$

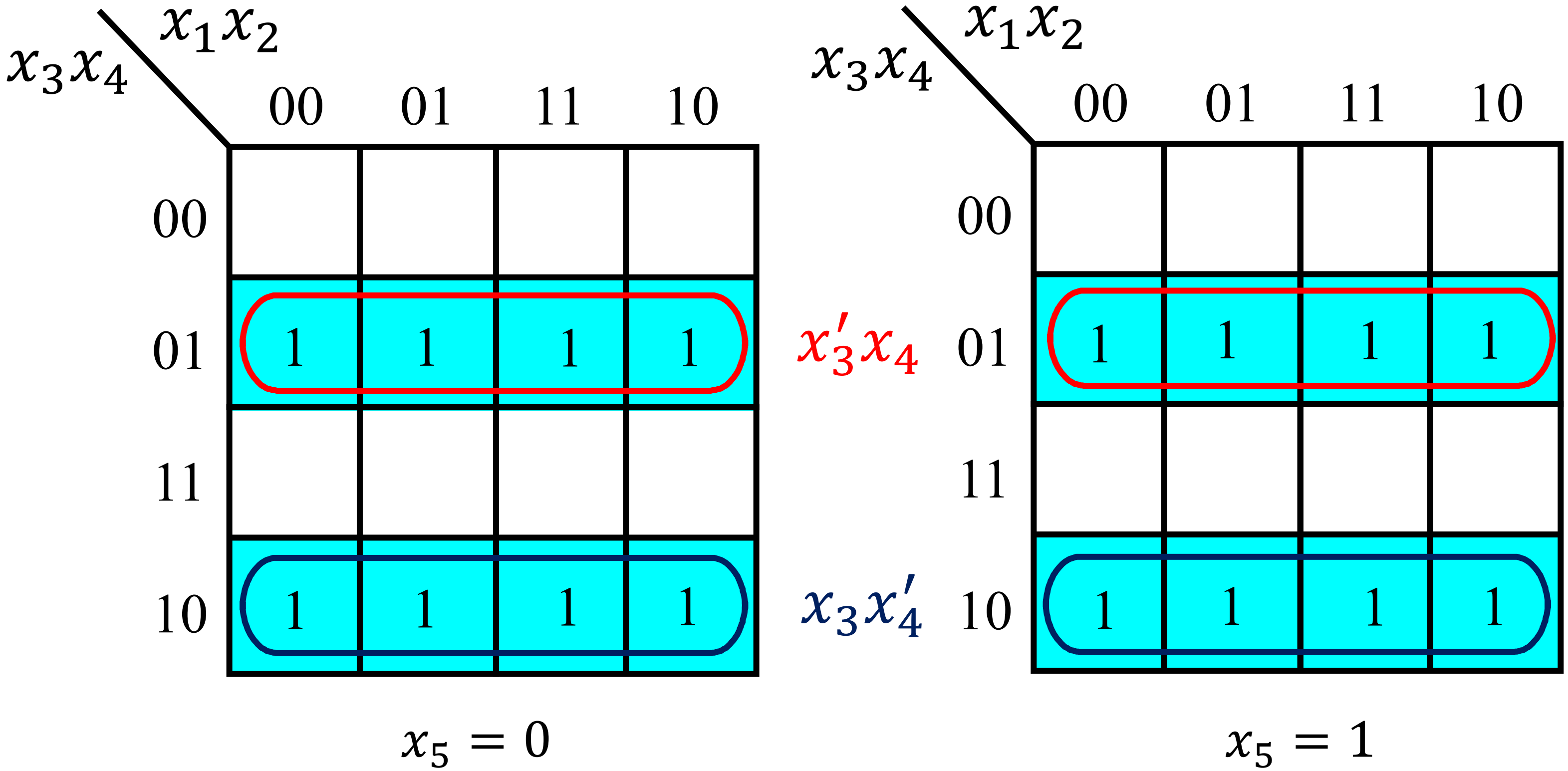
+

x_3x_4 \ x_1x_2	00	01	11	10
00				
01		1	1	1
11				
10		1	1	1

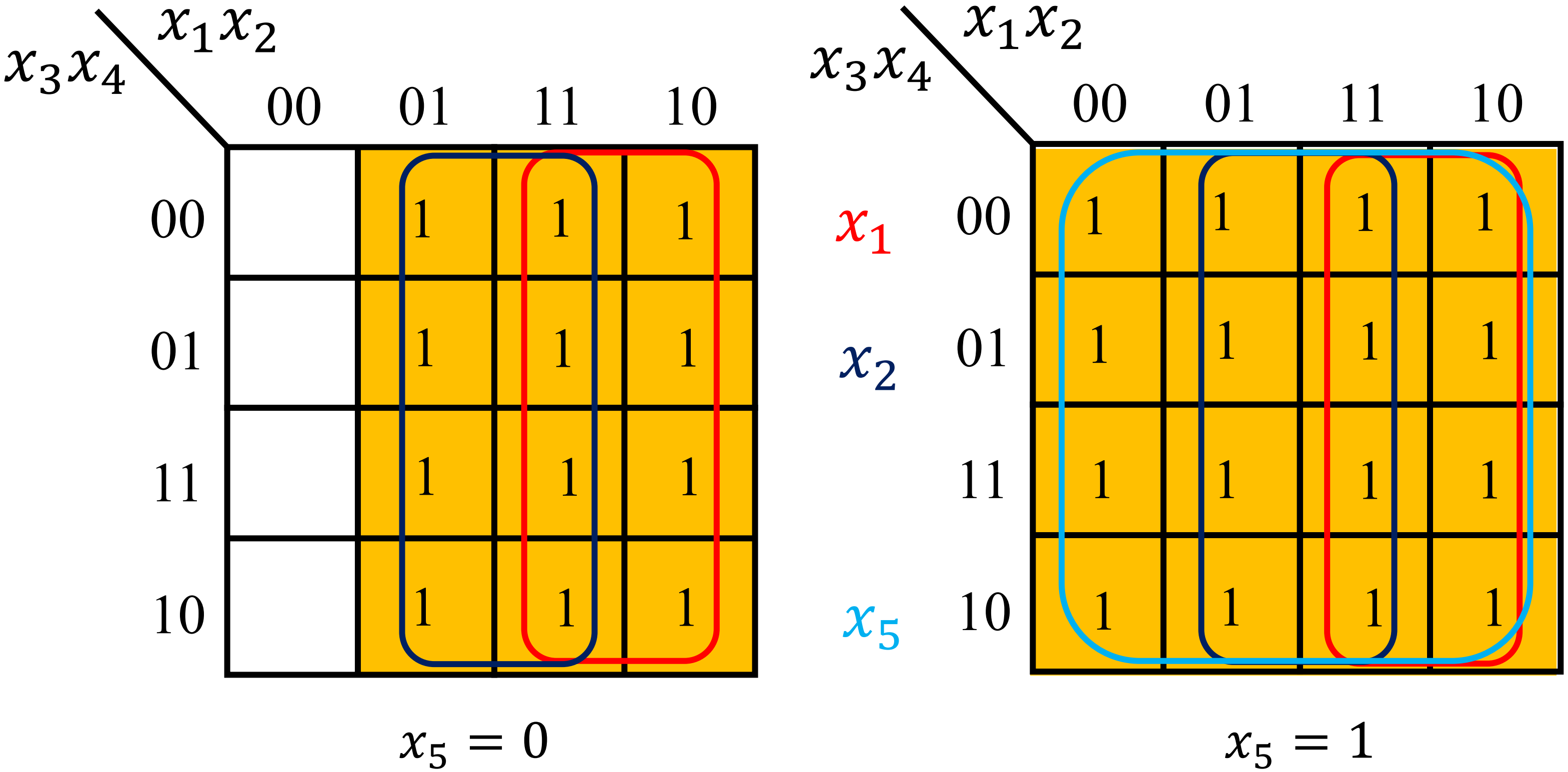
$x_5 = 0$

x_3x_4 \ x_1x_2	00	01	11	10
00				
01	1	1	1	1
11				
10	1	1	1	1

$x_5 = 1$



$$k = x_3'x_4 + x_3x_4' = x_3 \oplus x_4$$



x_1x_2
x_3x_4
00 01 11 10
00
01
11
10
1 1 1
1 1 1
1 1 1
1 1 1
$x_5 = 0$
x_1
x_2
x_5
x_1x_2
x_3x_4
00 01 11 10
00
01
11
10
1 1 1 1
1 1 1 1
1 1 1 1
1 1 1 1
$x_5 = 1$

$$g = x_1 + x_2 + x_5$$

x_3x_4 \ x_1x_2	00	01	11	10
00				
01	1	1	1	1
11				
10	1	1	1	1

$x_5 = 0$

x_3x_4 \ x_1x_2	00	01	11	10
00				
01	1	1	1	1
11				
10	1	1	1	1

$x_5 = 1$

k

AND

x_3x_4 \ x_1x_2	00	01	11	10
00		1	1	1
01		1	1	1
11		1	1	1
10		1	1	1

$x_5 = 0$

x_3x_4 \ x_1x_2	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$x_5 = 1$

g

=

x_3x_4 \ x_1x_2	00	01	11	10
00				
01		1	1	1
11				
10		1	1	1

$x_5 = 0$

x_3x_4 \ x_1x_2	00	01	11	10
00				
01	1	1	1	1
11				
10	1	1	1	1

$x_5 = 1$

kg

k'

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00	1	1	1	1
01				
11	1	1	1	1
10				

$x_5 = 0$

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00	1	1	1	1
01				
11	1	1	1	1
10				

$x_5 = 1$

AND

g'

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00	1			
01	1			
11	1			
10	1			

$x_5 = 0$

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00				
01				
11				
10				

$x_5 = 1$

=

$k'g'$

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00	1			
01				
11	1			
10				

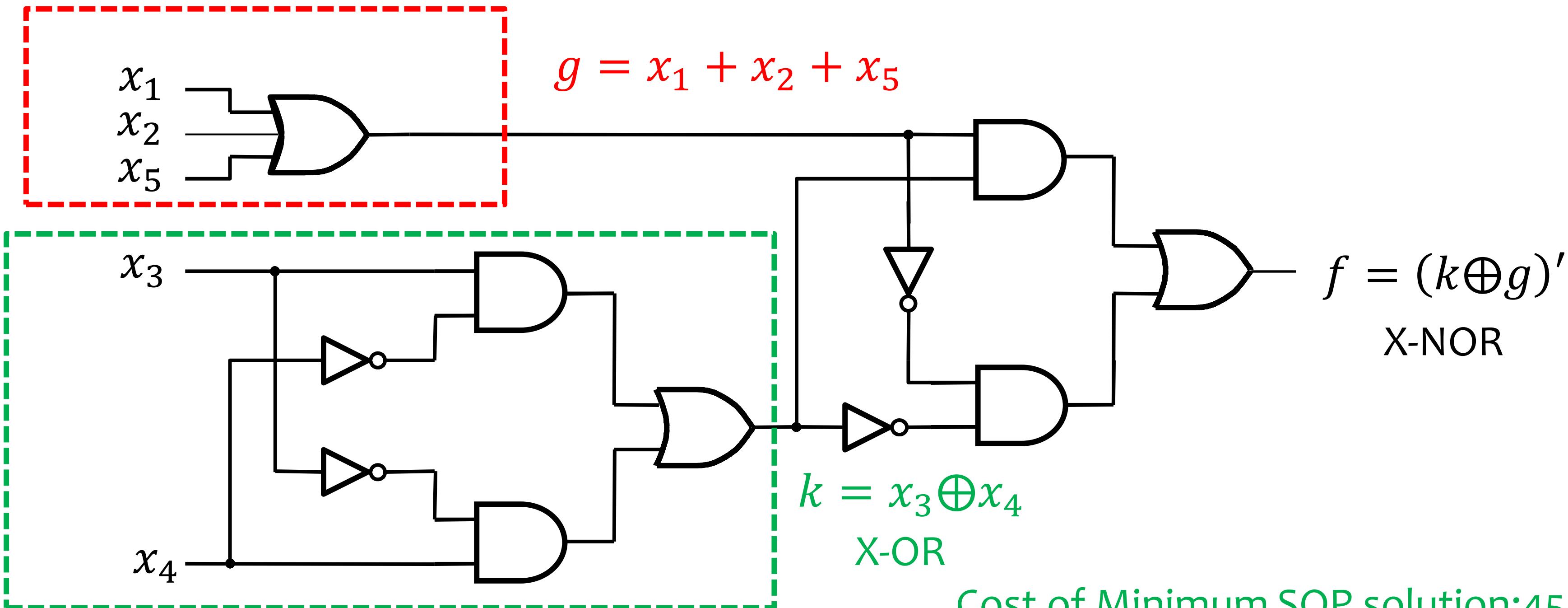
$x_5 = 0$

$x_3x_4 \backslash x_1x_2$	00	01	11	10
00				
01				
11				
10				

$x_5 = 1$

Decomposition for Example 4.7

$$f = h[g(x_1, x_2, x_5), k(x_3, x_5)]$$



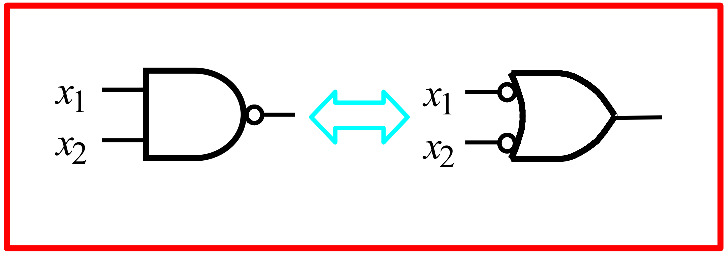
Cost of Minimum SOP solution:45

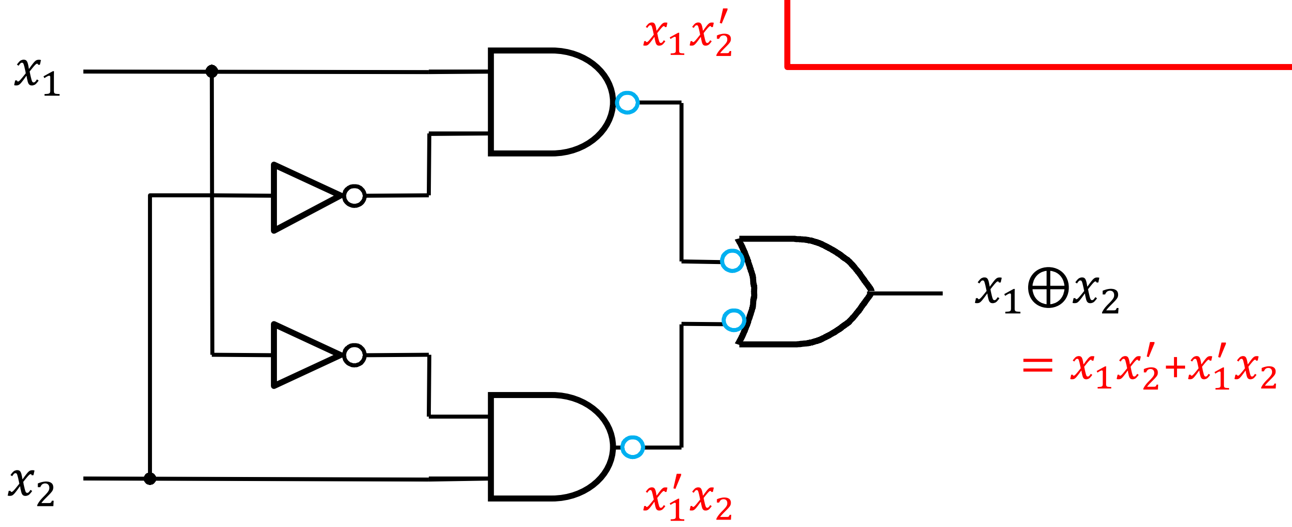
(b) Circuit obtained using decomposition

4 x 2-input ANDs, 2 x 2-input ORs, 1 x 3-input OR

Cost: 7 + (4 x 2 + 2 x 2 + 1 x 3) = 22

Implementation of XOR

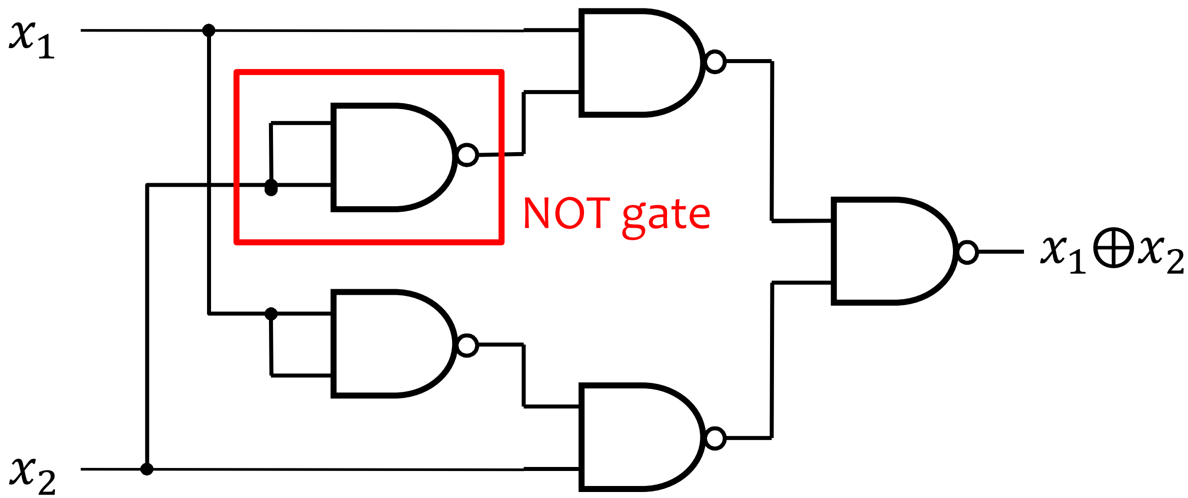




(a) Sum-of-products implementation

Figure 4.26. Implementation of XOR

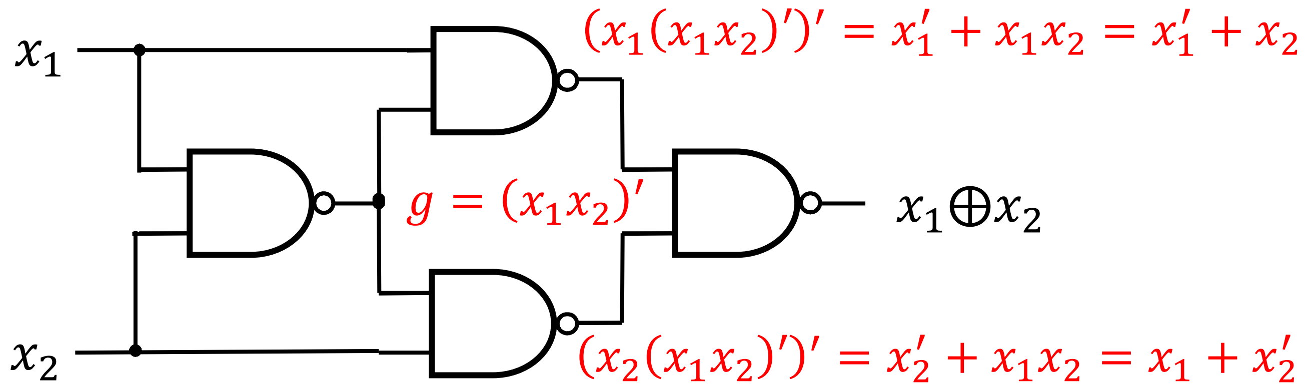
Implementation of XOR



(b) NAND gate implementation

Figure 4.26. Implementation of XOR

Implementation of XOR

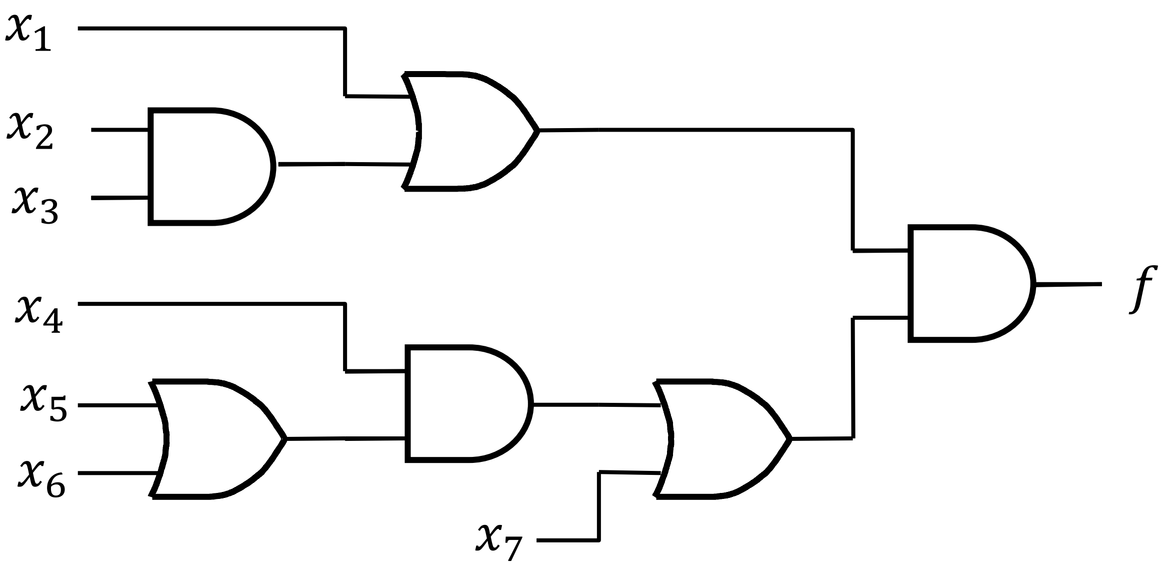


$$\begin{aligned} x_1 \oplus x_2 &= \big((x_1' + x_2)(x_2' + x_1)\big)' \\ &= (x_1'x_2' + x_1'x_1 + x_2'x_2 + x_1x_2)' \\ &= (x_1'x_2' + x_1x_2)' \quad \text{X-NOR} \end{aligned}$$

(c) Optimal NAND gate implementation

Figure 4.26. Implementation of XOR

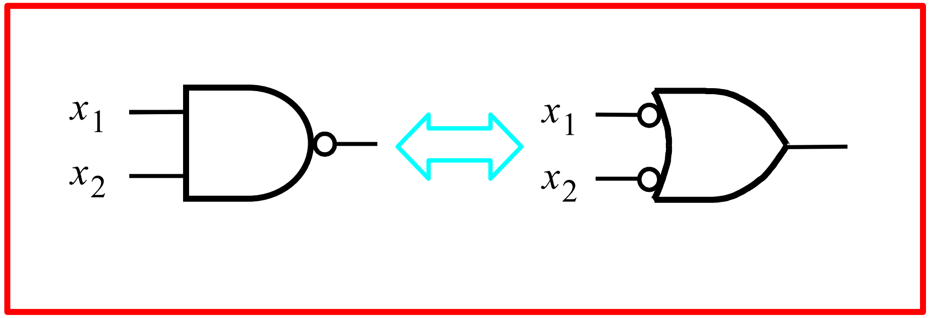
Multilevel NAND and NOR Circuits

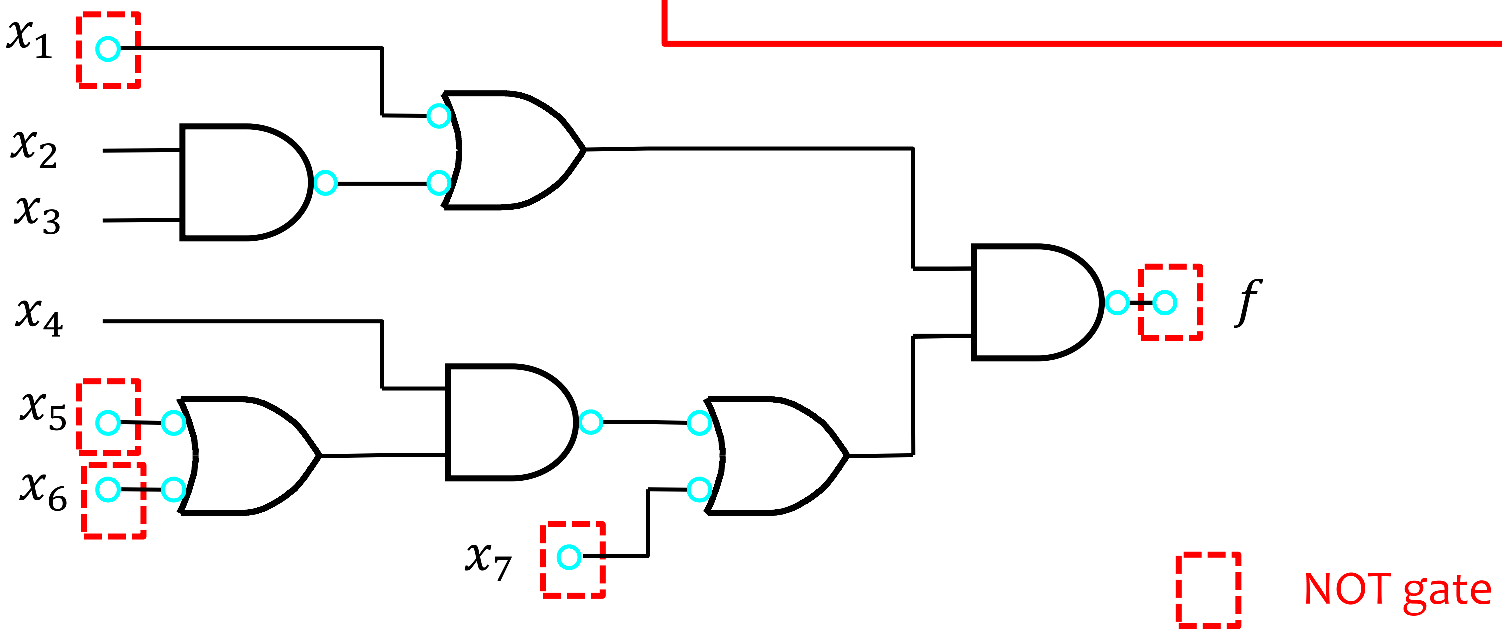


(a) Circuit with AND and OR gates

Figure 4.27. Conversion to a NAND-gate circuit.

Multilevel NAND and NOR Circuits

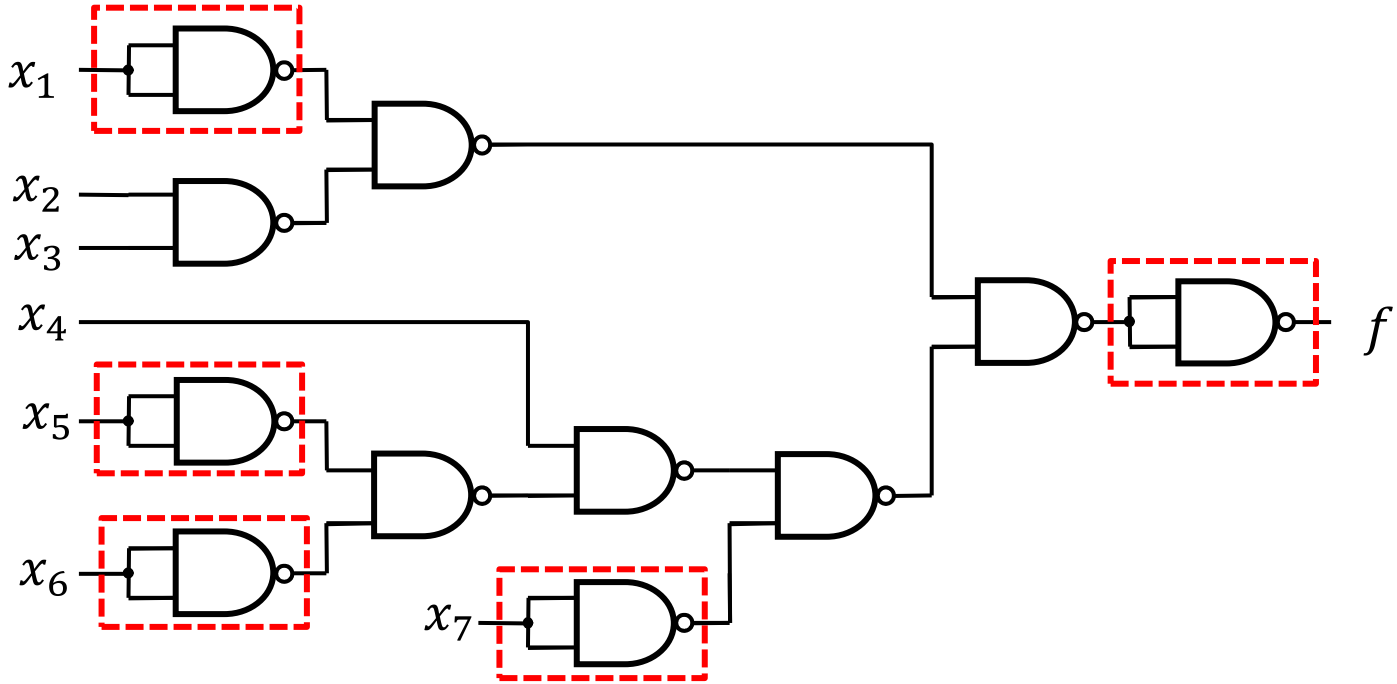




(b) Inversions needed to convert to NANDs

Figure 4.27. Conversion to a NAND-gate circuit.

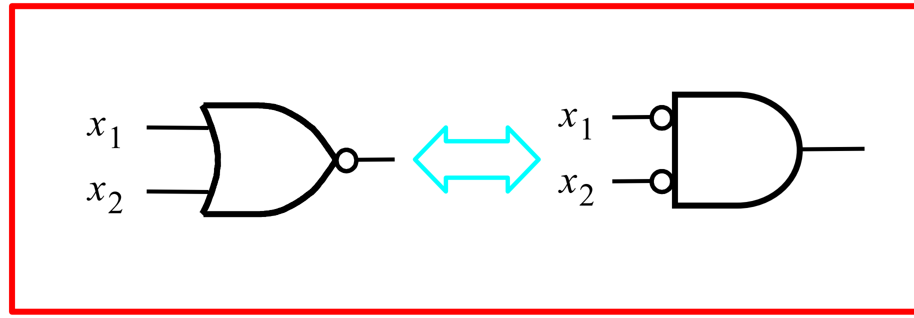
Multilevel NAND and NOR Circuits

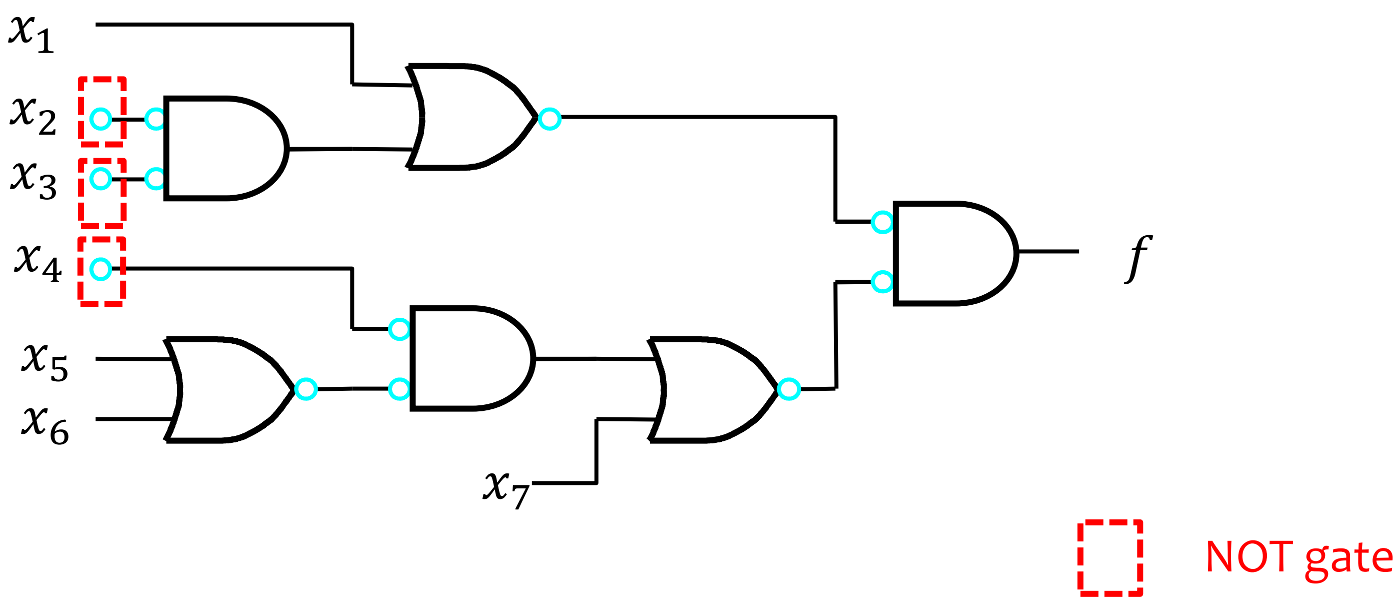


(c) NAND-gate circuit

Figure 4.27. Conversion to a NAND-gate circuit.

Multilevel NAND and NOR Circuits

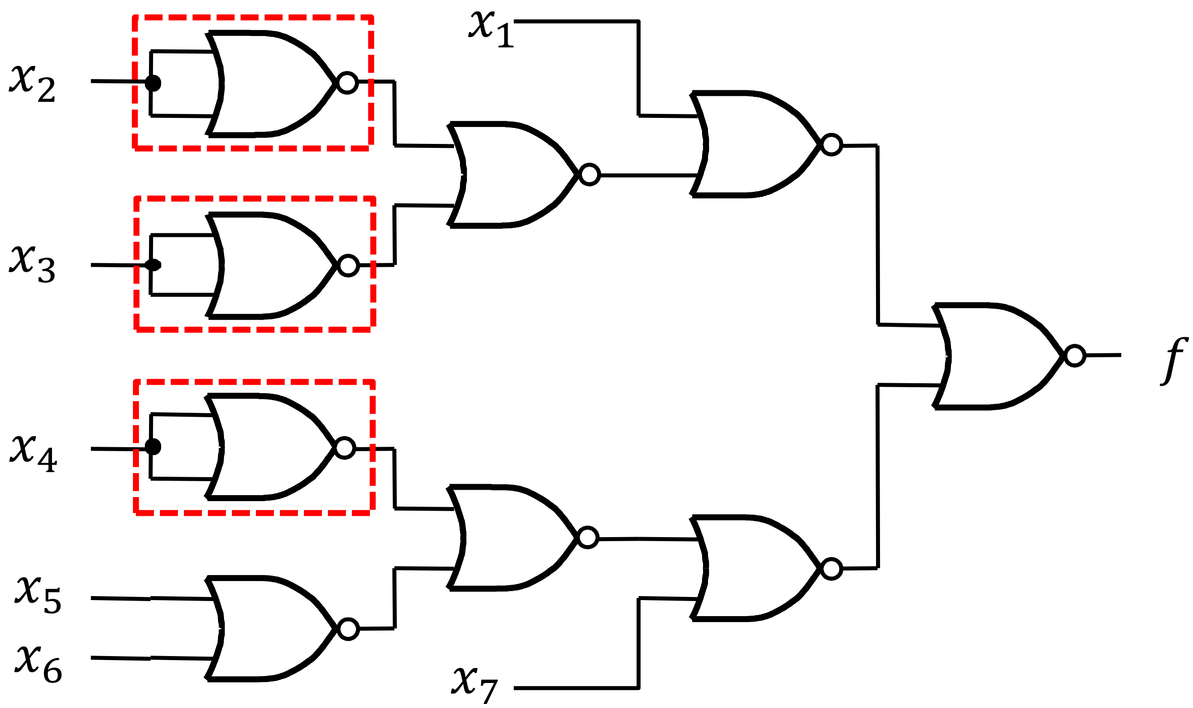




(a) Inversions needed to convert to NORs

Figure 4.28. Conversion to a NOR-gate circuit.

Multilevel NAND and NOR Circuits

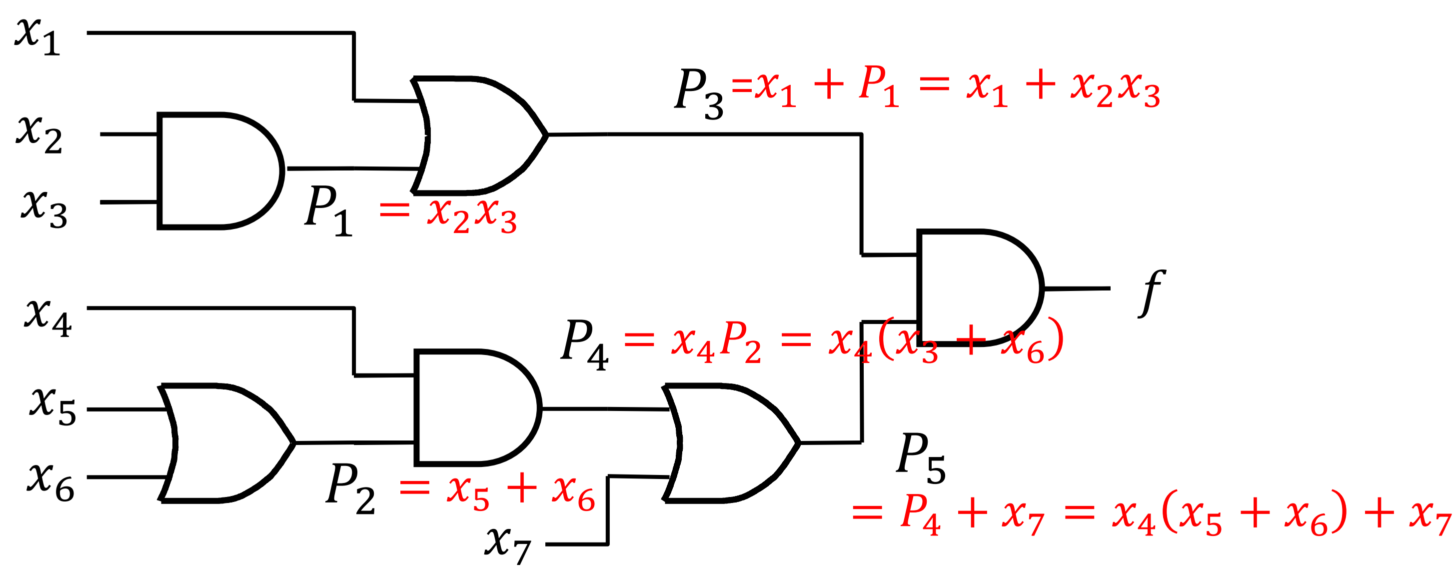


(b) NOR-gate circuit

Figure 4.28. Conversion to a NOR-gate circuit.

ANALYSIS OF MULTILEVEL CIRCUITS

Analysis of Multilevel Circuits



$$f = P_3P_5 = (x_1 + x_2x_3)(x_4(x_5 + x_6) + x_7)$$

$$= x_1x_4x_5 + xx_4x_6 + x_1x_7 + x_2x_3x_4x_5 + x_2x_3x_4x_6 + x_2x_3x_7$$

Figure 4.29. Circuit for Example 4.10.

Summary

- Optimized logic circuits can be implemented by the Karnaugh map.
- Don’t care terms in incompletely specified functions can be regarded by 0 or 1 for minimized logic function.
- Multiple output circuits can be efficiently implemented using the common units for multiple outputs.
- To solve the fan-in problem, output must be expressed in a form called a multilevel logic expression.
- Two important techniques for multilevel function are factoring and functional decomposition.