ECE/CS 3700
Digital System Design

Lecture Slides for Chapter 2: Formal Procedures for SOP minimization and Karnaugh Maps

THE UNIVERSITY OF UTAH

Priyank Kalla

Professor

Electrical & Computer Engineering

With more variables, Logic simplification becomes infeasible using algebraic/symbolic manipulation. We need formal techniques …

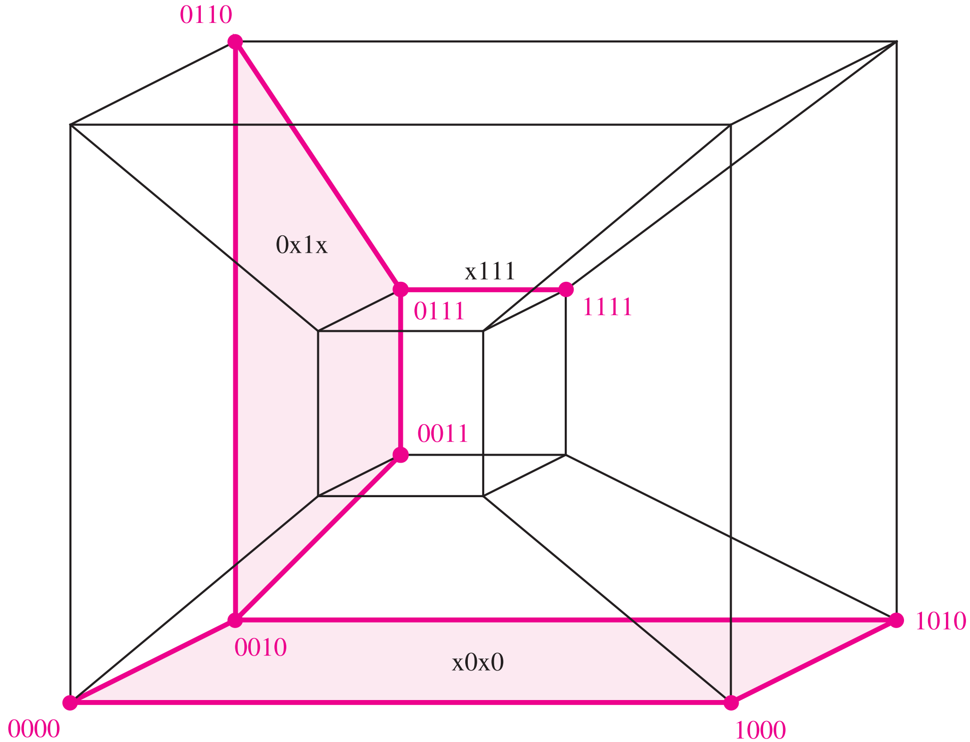
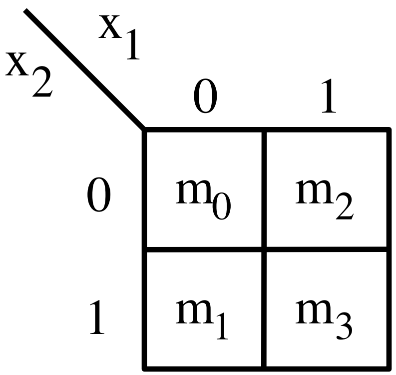


Figure 8.18 Representation of function f_3 from Figure 2.54

A 4-dimensional cube

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

(a) Truth table



(b) Karnaugh map

Figure 2.49. Location of two-variable minterms.

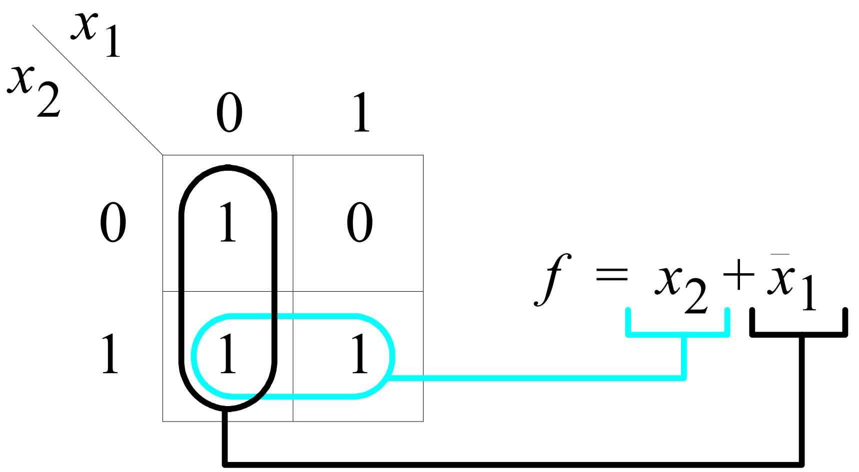


Figure 2.50. The function of Figure 2.19.

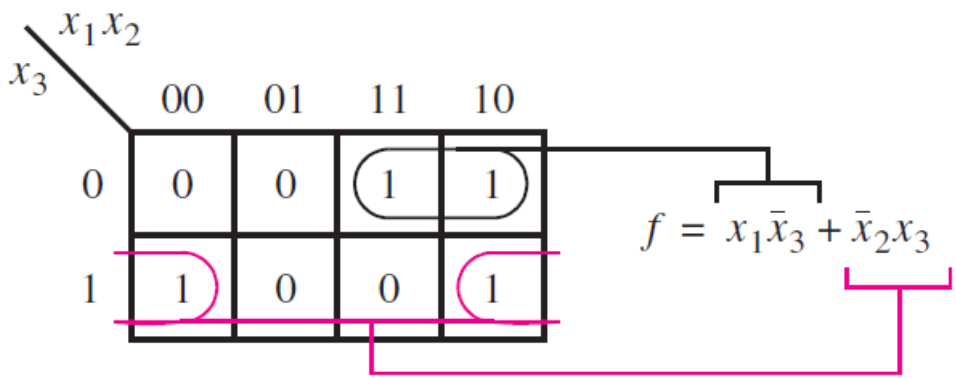
x_1	x_2	x_3	
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7

(a) Truth table

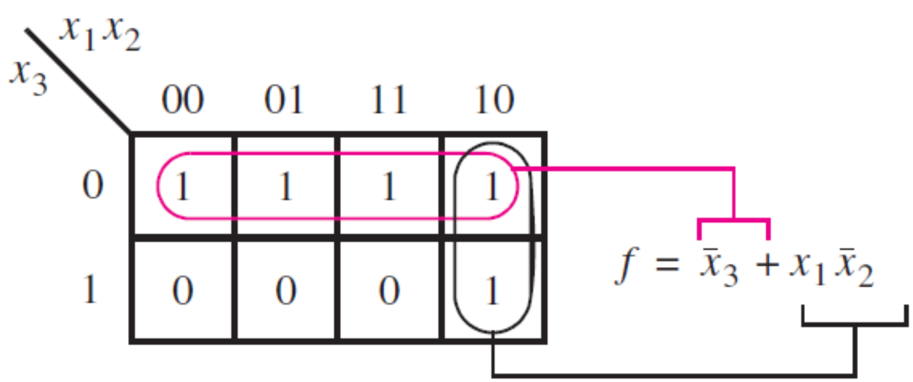
x_3 \ $x_1 x_2$	00	01	11	10
0	m_0	m_2	m_6	m_4
1	m_1	m_3	m_7	m_5

(b) Karnaugh map

Figure 2.51. Location of three-variable minterms.



(a) The function of Figure 2.23



(b) The function of Figure 2.48

Figure 2.52. Examples of three-variable Karnaugh maps.

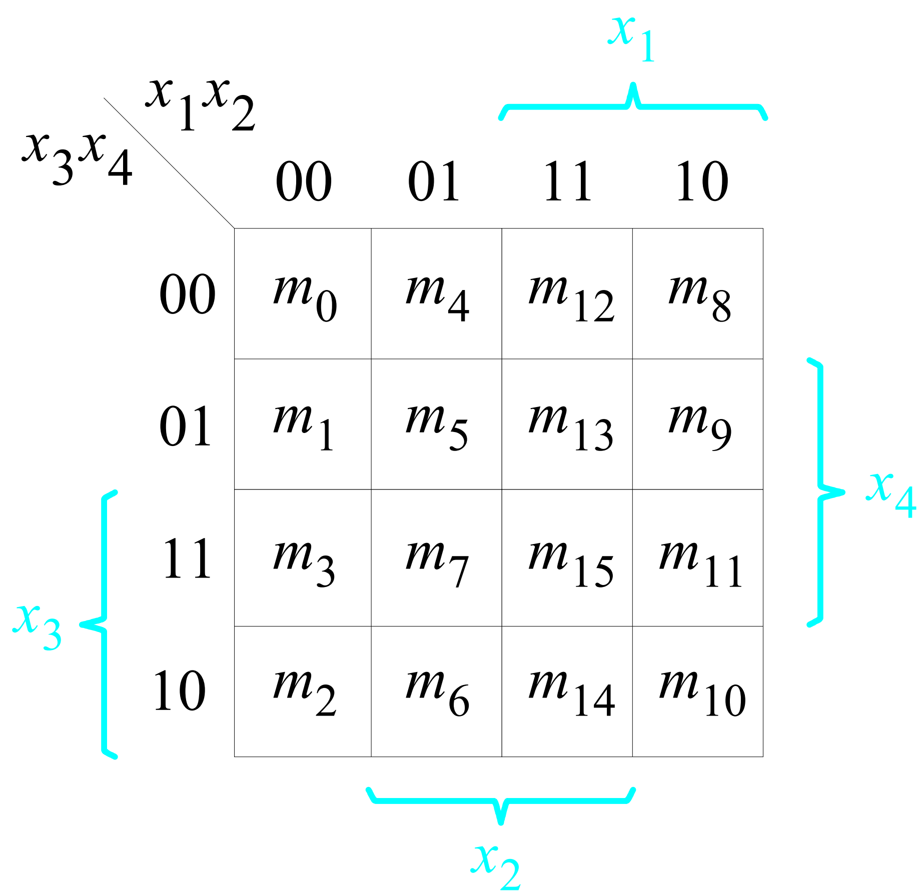


Figure 2.53. A four-variable Karnaugh map.

x_1x_2 \ x_3x_4	00	01	11	10
00	m_0	m_1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

$x_1x_2x_3x_4$	
0000	
⋮	
1111	

Terminology

- **Binary Variable** = symbol. Represents a co-ordinate of Boolean space spanned by n-variables (called B^n), where n = the number of variables of the function
- **Literal**: Boolean variable, or its complement
- $f = a + a'b$ has how many literals? 3 literals: a, a' are different literals.
- **Minterm**: a point in the Boolean space
  - A product of all n literals
- **Cube**: a point, or a set of points in B^n
  - A product of literals, may contain fewer than n literals
- $f(a, b, c) = a'bc + abc$: 2 cubes. But $f = bc$ is a larger cube containing both.
- **Implementation Cost:** Number of literals in expression, rough estimate of area. 1 literals = 2 CMOS transistors.

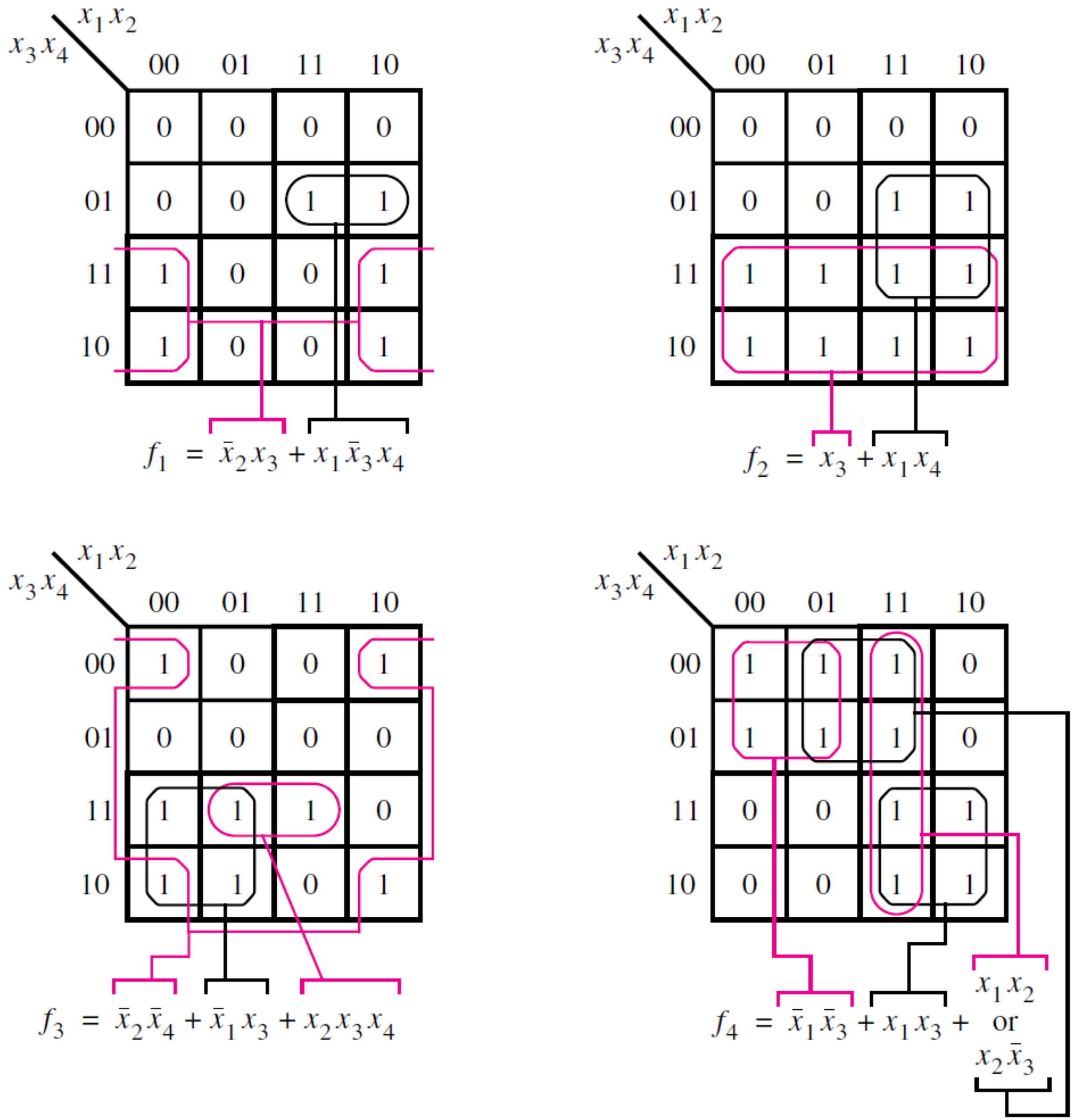


Figure 2.54. Examples of four-variable Karnaugh maps.

More terminology

Implicants of a Function

- Implicant: Same thing as an ON-SET cube; "implies" the value of the function ($= 1$)
- **Prime Implicant:** Not contained in any other implicant
- Prime implicant cannot be expanded
- Prime implicant is a largest cube
- One solution for logic minimization: $F =$ all prime implicants
- Problem: Redundancy! Too many ($\leq 3^n/n$) primes
- Still have to make choices...
- Greedy strategy does not always work
- Quine-McCluskey gave a systematic solution to find a **minimum cost cover of a function**

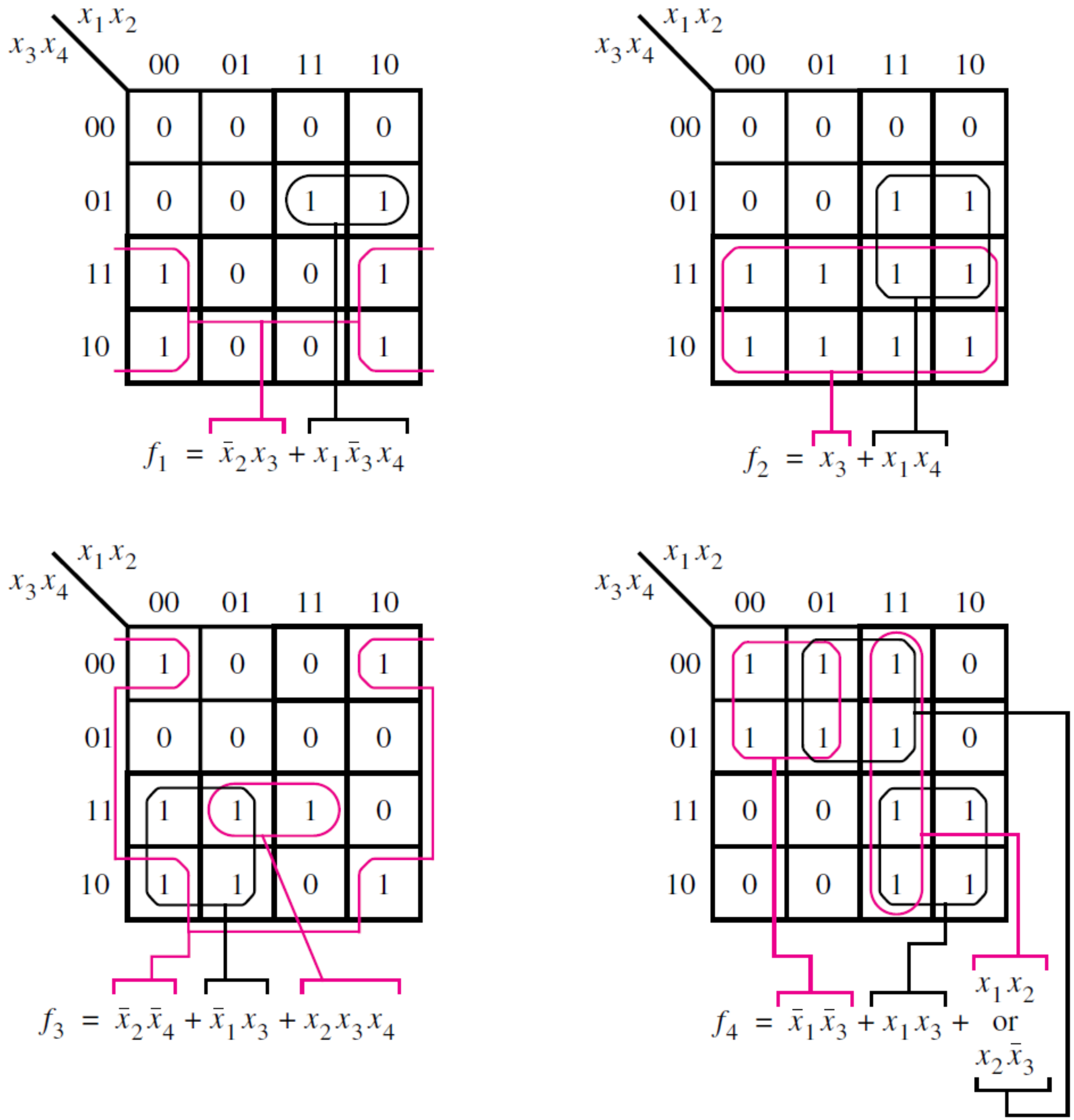


Figure 2.54. Examples of four-variable Karnaugh maps.

Exact Logic Minimization

- *Prime Cover:* A Cover containing only prime implicants
- **Quine's Theorem:**
  - There exists a minimum cover that is prime!
- Thats why, analyze only prime implicants
  - Quickly generate all prime implicants: Expand all ON-set cubes as much as possible!
  - Identify all essential primes
  - Now select a minimum number of primes from the remaining ones.
- "Minimum number of primes" versus "A minimum number of primes with minimum cost". See Fig. 2.57.
- A Minimum Cost cover is NOT unique, see Fig. 2.54 (iv)

So, the strongest problem formulation is: *Find a minimum cost cover from among the prime implicants that also comprises a minimum number of primes!*

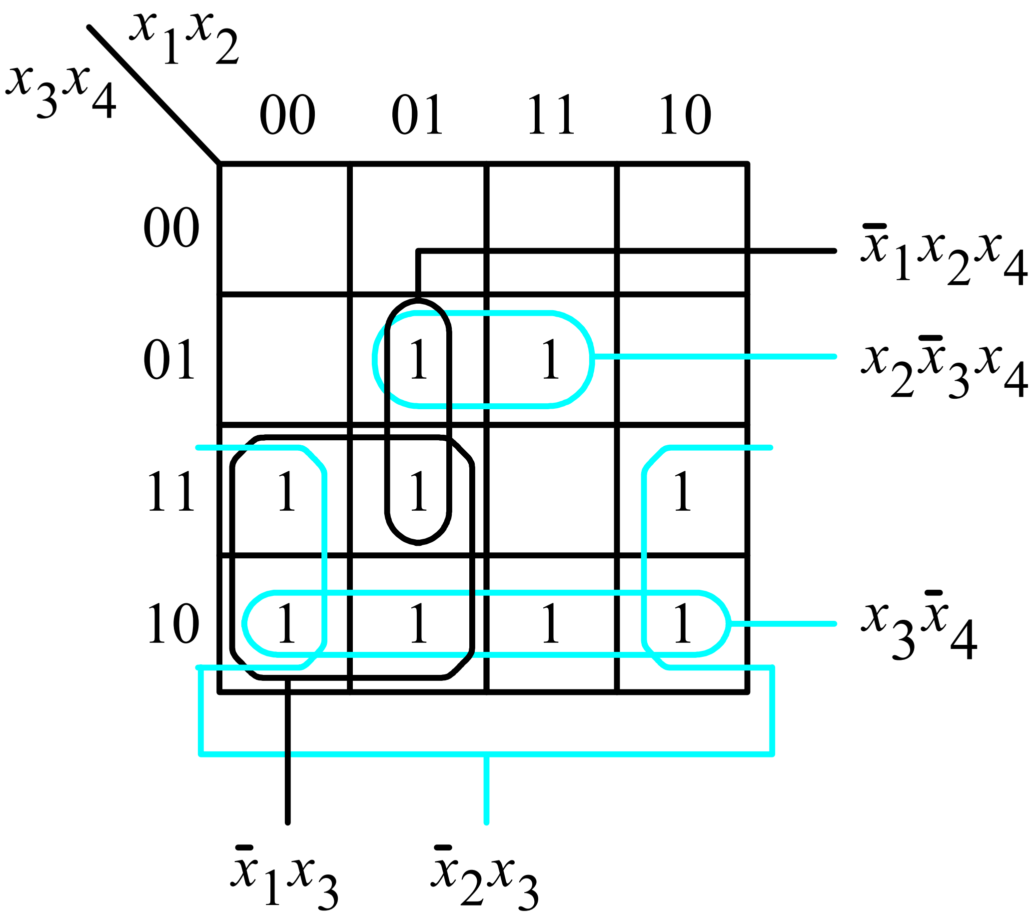
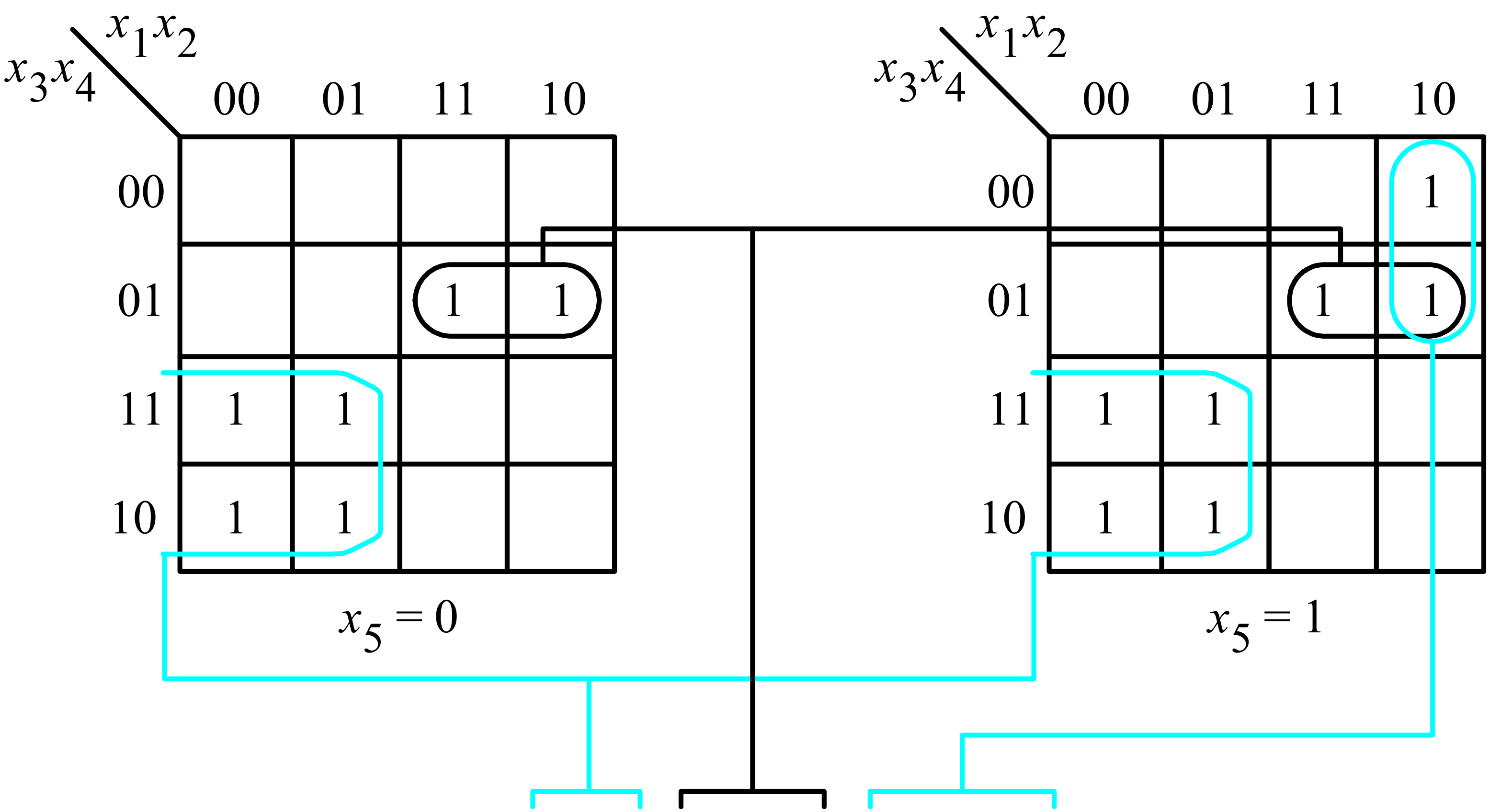


Figure 2.57. Four-variable function $f(x_1,\ldots,x_4) = \Sigma m(2, 3, 5, 6, 7, 10, 11, 13, 14)$.



$$f_1 = \bar{x}_1 x_3 + x_1 \bar{x}_3 x_4 + x_1 \bar{x}_2 \bar{x}_3 x_5$$

Figure 2.55. A five-variable Karnaugh map.

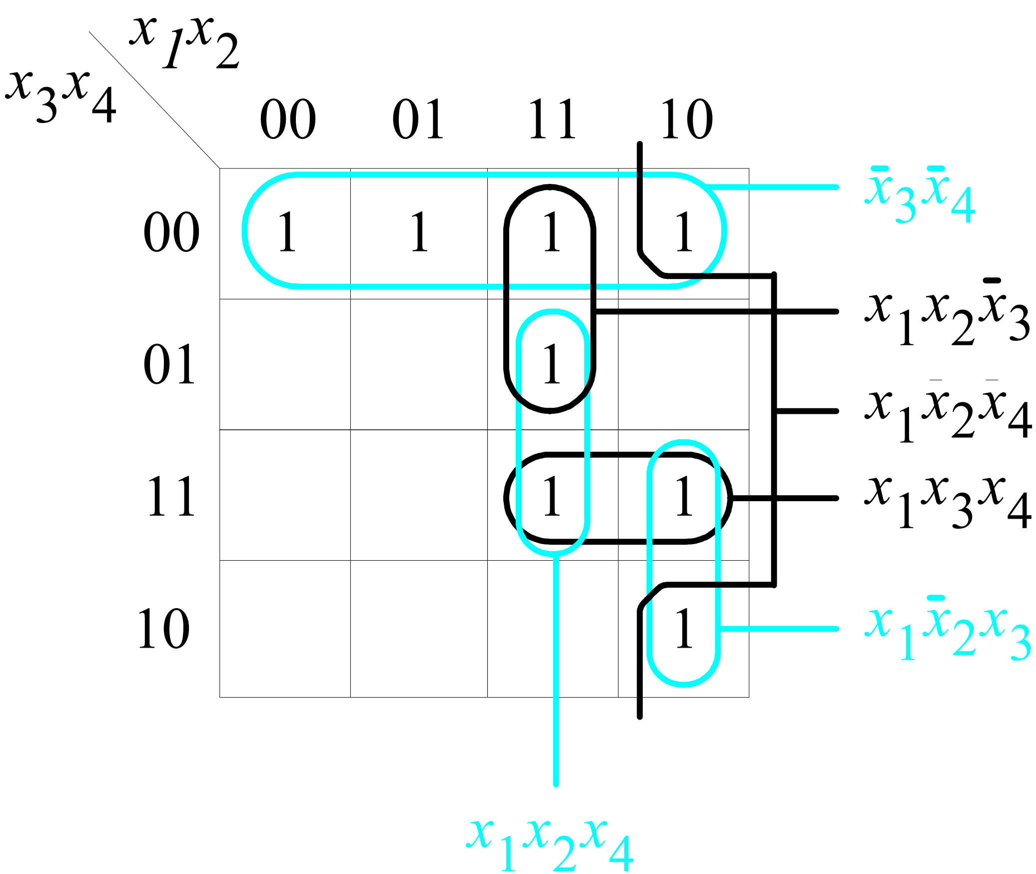


Figure 2.58. The function $f(x_1,\ldots,x_4) = \Sigma\, m(0, 4, 8, 10, 11, 12, 13, 15)$.

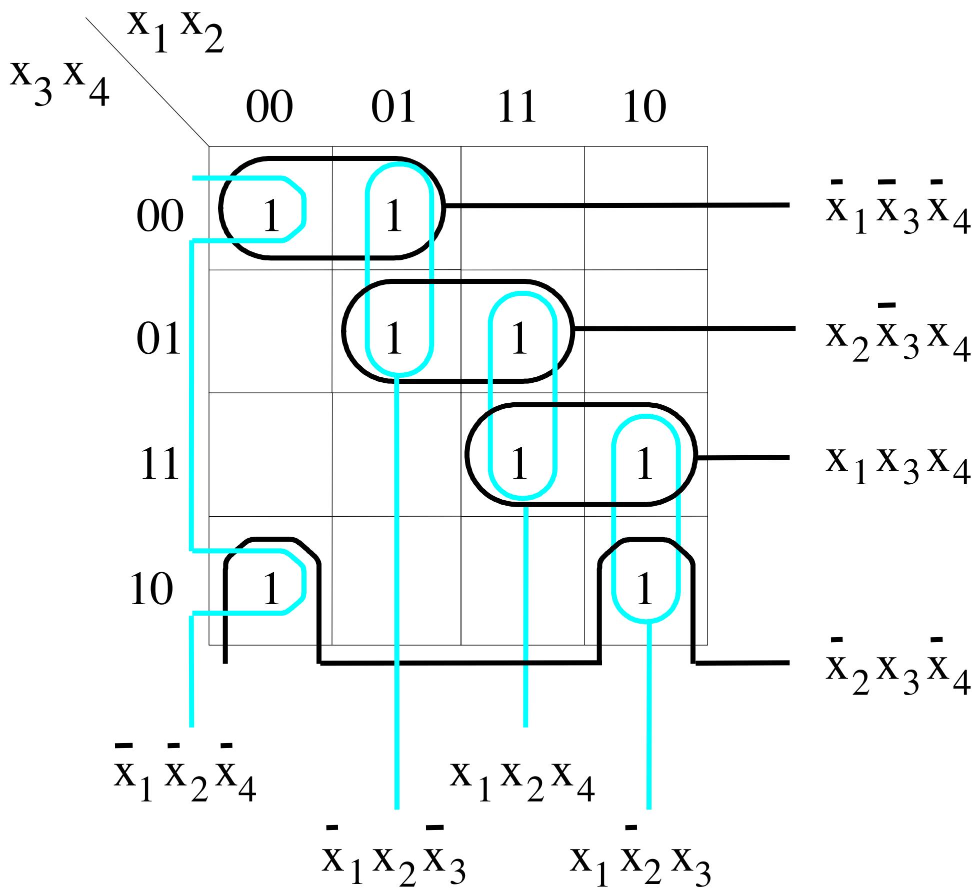
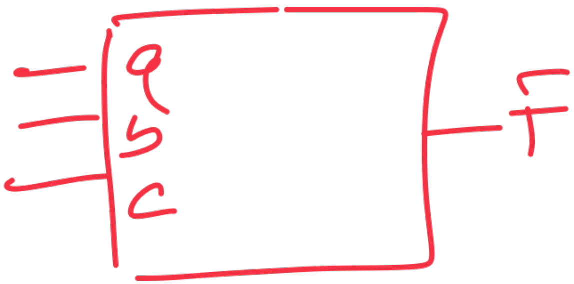


Figure 2.59. The function $f(x_1, \ldots, x_4) = \Sigma\, m(0, 2, 4, 5, 10, 11, 13, 15)$.

Don’t care conditions (DC)

- Sometimes, a circuit may not receive all possible input assignments
- Then, the output value for that assignment does not matter, or we don’t care about the output
- That input assignment is called a don’t care condition
- Such functions are called “incompletely specified” Boolean functions
- $f : \mathbb{B}^n \rightarrow \{0,1,*\}$ instead of $f : \mathbb{B}^n \rightarrow \mathbb{B}$

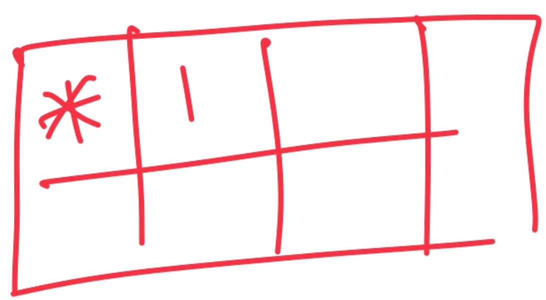
- Don’t care condition = input minterm
- Don’t care value = output could be assigned 0 or 1, depending on what leads to better simplification



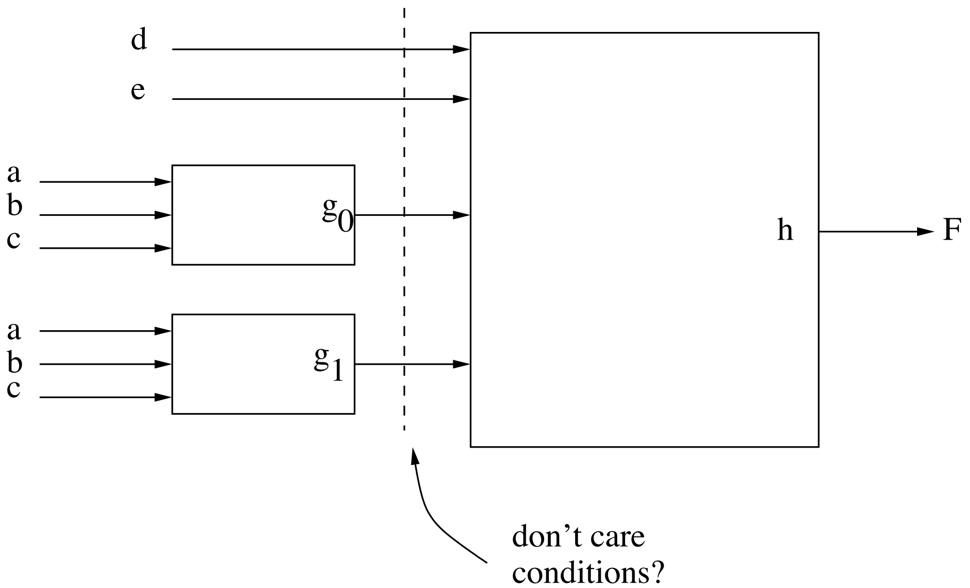
a	b	c	f
0	0	0	*
0	0	1	1
0	1	0	0
⋮	⋮	⋮	⋮

Suppose a=b=c=0 never arrives?

use $f(a,b,c) = 0$ or 1



Where do DCs come from?

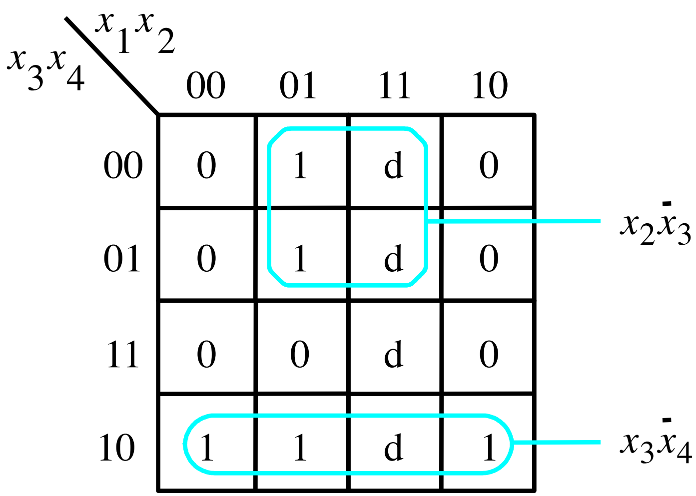


$g_0 = a'bc + ab'c + abc'$ and $g_1 = a'b'c + abc$

$h(d, e, g_0, g_1)$

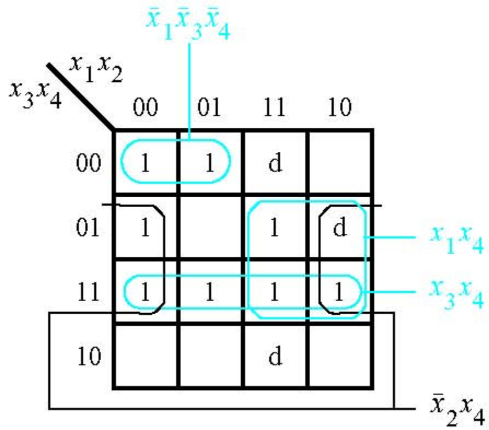
$g_0 = g_1 = 1$ not possible

d e g_0 g_1	h
0 0 0 0	
0 0 0 1	
0 0 1 0	
0 0 1 1	→ *
0 1 0 0	
⋮	
0 1 1 1	→ *
1 0 1 1	→ *
1 1 1 1	→ *

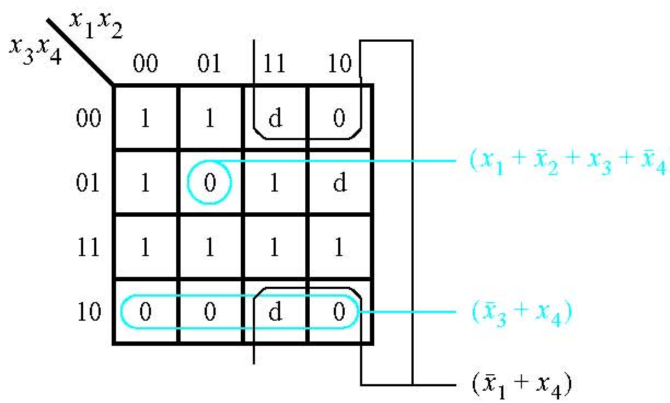


(a) SOP implementation

Figure 2.62. Two implementations of the function $f(x_1, \ldots, x_4) = \Sigma\, m(2, 4, 5, 6, 10) + D(12, 13, 14, 15)$.



(a) Determination of the SOP expression



(b) Determination of the POS expression