Gate Level Minimization

Authors
(Joint work of **Group13**)

Indian Institute Of Technology Hyderabad

Digital Systems
Date 2024

Contents

Introduction

- **Gate-level minimization** is the design task of finding an optimal gate-level implementation of the Boolean functions describing a digital circuit.
- It is too difficult to execute it by manual method when the logic has more number of inputs.
- This problem has been solved by computer-based logic synthesis tools that minimize a large set of Boolean equations efficiently and quickly.

Map Method

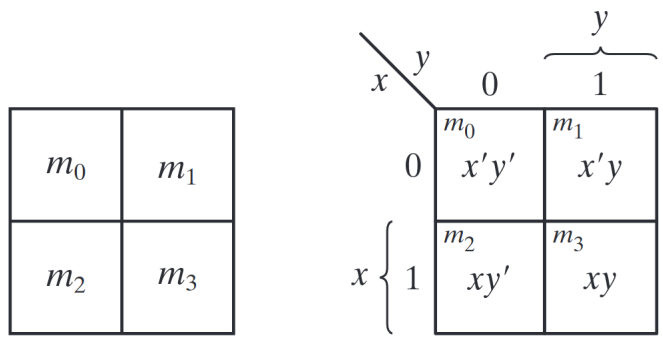
- The Map Method is the simple,straight forward procedure for minimizing Boolean functions Known as Karnaugh Maps(K-Maps)
- K-Maps are a visual representation of truth tables.
- Adjacent cells differ by one variable, enabling simplification.
- K-Maps exist for 2, 3, 4, and 5 variables.

Note

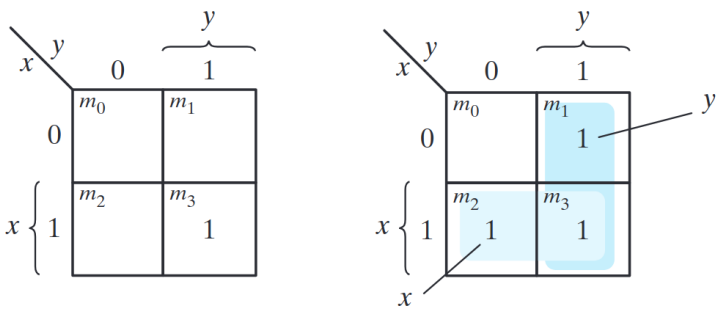
The main assumption in the K-map method:
The simplest algebraic expression is one that has a minimum number of terms with the smallest possible number of literals in each term.

2 Variable K-Map

- Four squares representing minterms.
- Simplified expressions minimize gates and inputs.



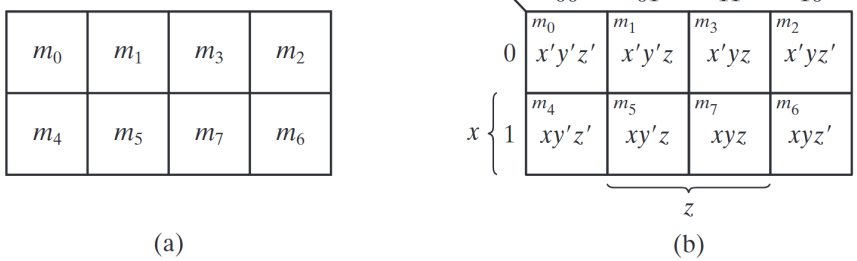
- Example: $F(x, y) = xy$ and $F(x, y) = x + y$



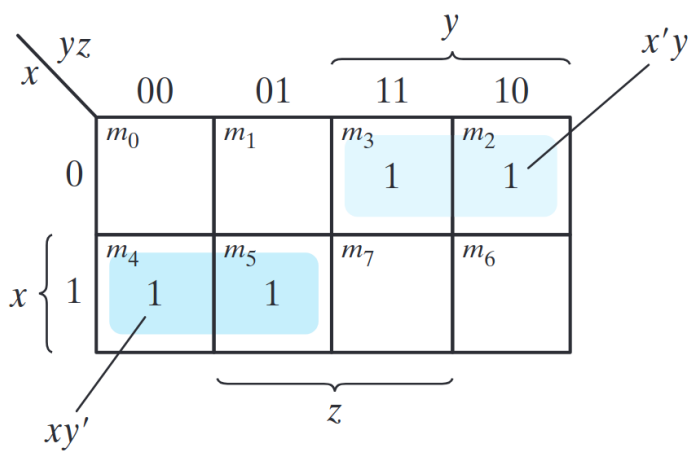
(a) xy (b) $x + y$

3 Variable K-Map

- 8 squares arranged in Gray code sequence.
- Adjacent squares differ by only one variable.

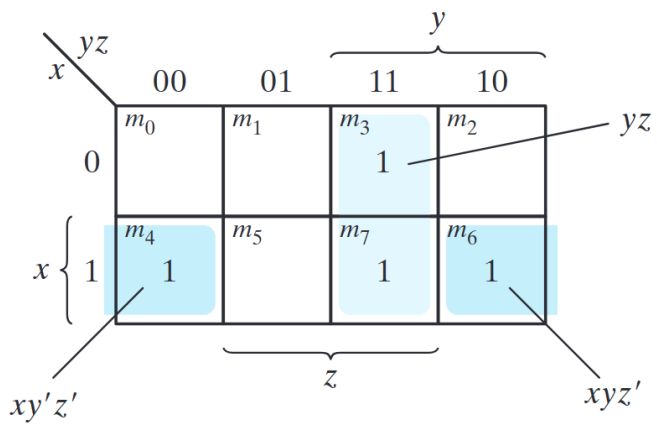


- Example 1: $F(x, y, z) = \Sigma(2, 3, 4, 5) \rightarrow x'y + xy'$

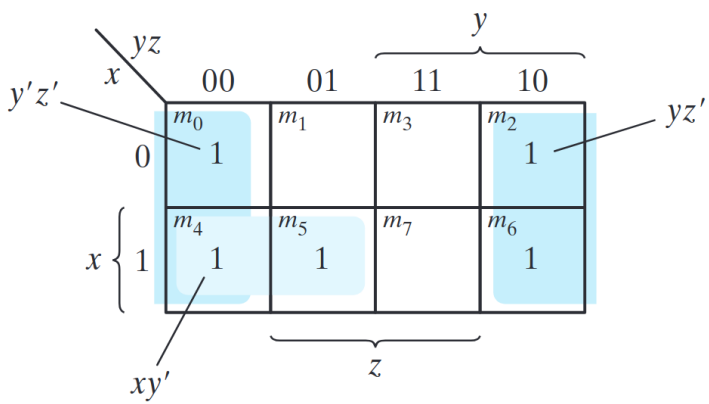


3 Variable K-Map examples

- Example 2: $F(x, y, z) = \Sigma(3, 4, 6, 7) \rightarrow yz + xz'$



- Example 3: $F(x, y, z) = \Sigma(0, 2, 4, 5, 6) \rightarrow z' + xy'$



4 Variable K-Map

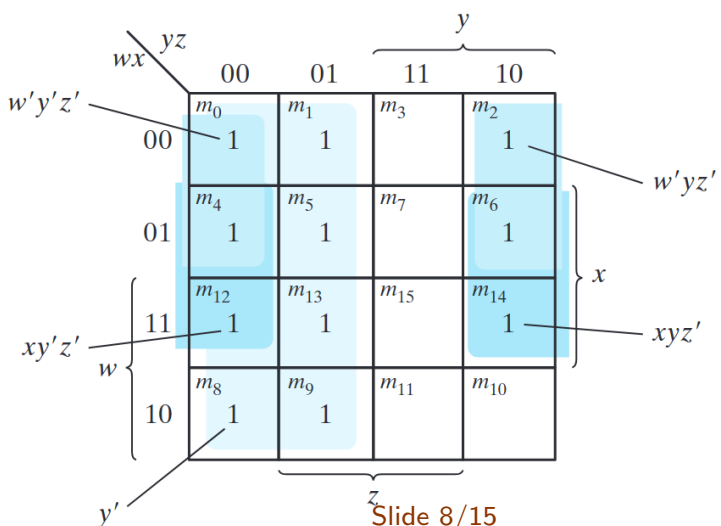
- 16 squares, organized using Gray code.
- Allows simplifications of larger functions.

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}

wx \ 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} $wx'yz'$

(y spans 11, 10; x spans 01, 11, 10; w spans 11, 10; z spans 01, 11)

- $F(w, x, y, z) = \Sigma(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14) \rightarrow y' + w'z' + xz'$



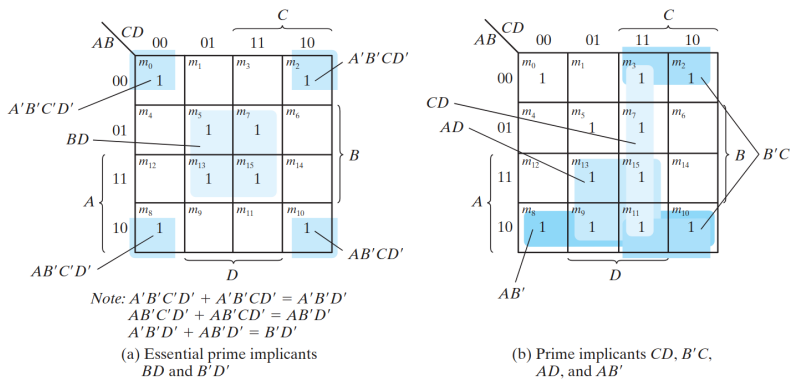
Prime Implicants

- A **prime implicant** is a product term obtained by combining the maximum possible number of adjacent squares in a Karnaugh map (K-map). An implicant is **prime** if no other implicant with fewer literals covers it.
- A **prime implicant** is **essential** if a minterm is covered only by that implicant. Essential prime implicants **must** be included in the simplified function.

1. Procedure for Finding Prime Implicants:
   - A single **1** in the K-map is a prime implicant if it has no adjacent 1s.
   - Two adjacent **1s** form a prime implicant unless they are part of a larger group.
   - Four adjacent **1s** form a prime implicant unless they are part of a group of eight.

Example

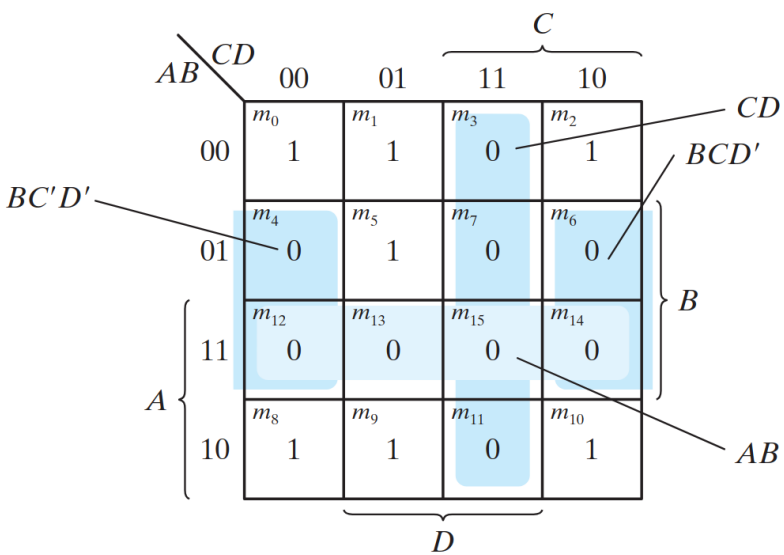
- Example: $F(A, B, C, D) = \Sigma(0, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$



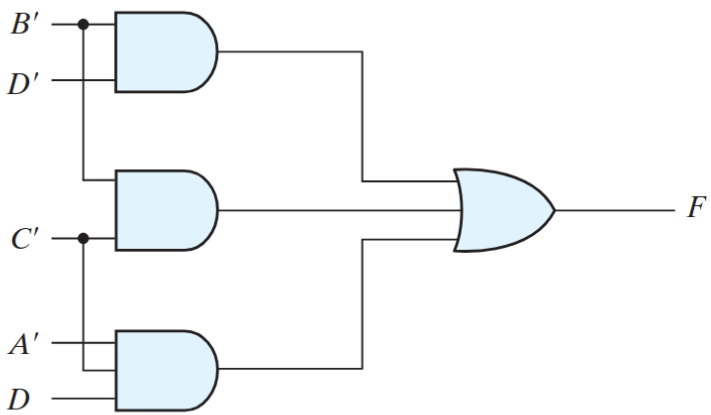
- $F = BD + B'D + CD + AD$
- $F = BD + B'D + CD + AB'$
- $F = BD + B'D + B'C + AD$
- $F = BD + B'D + B'C + AB'$

Sum-of-Products (SOP) and Product-of-Sums (POS)

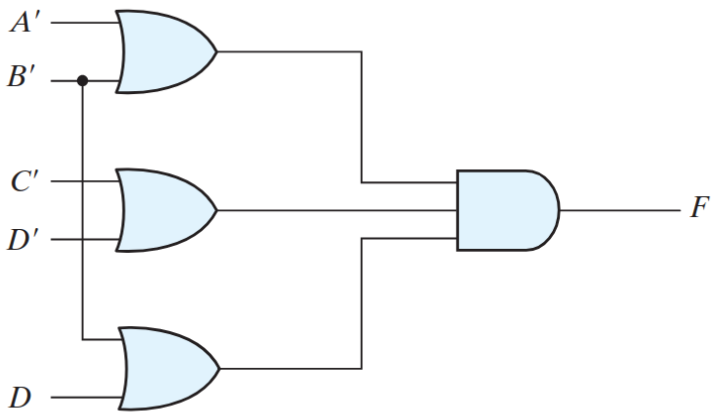
- SOP: OR-ing product terms (AND gates to OR gate).
- POS: AND-ing sum terms (OR gates to AND gate).
- Example: $F(A, B, C, D) = \Sigma(0, 1, 2, 5, 8, 9, 10)$
  $F' = AB + CD + BD'$ (SOP),
  $F = (A' + B')(C' + D')(B' + D)$ (POS)



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



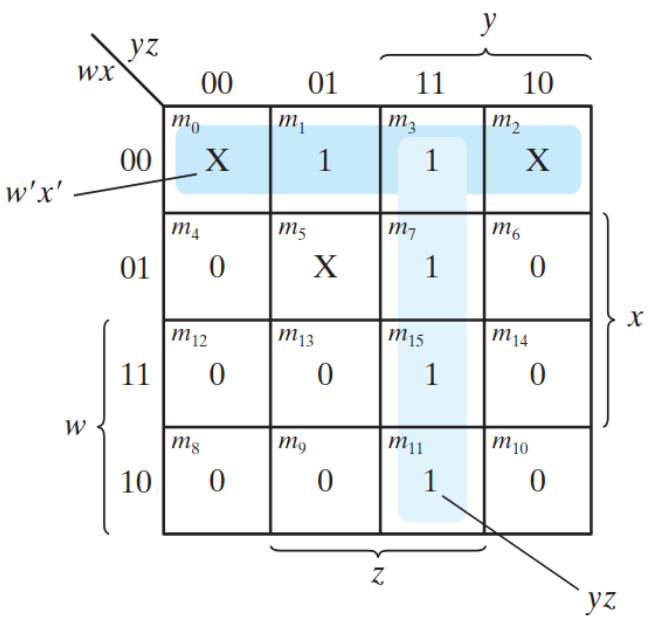
(a) $F = B'D' + B'C' + A'C'D$



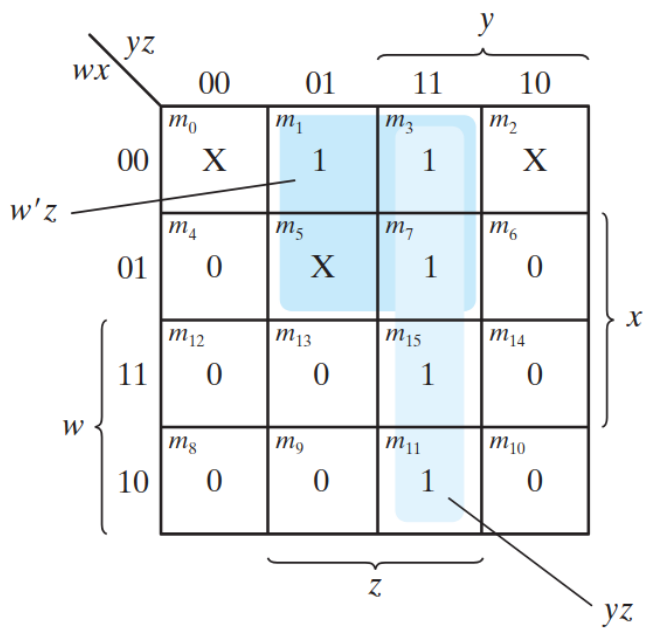
(b) $F = (A' + B')\,(C' + D')\,(B' + D)$

Don't Care Conditions

- Unused input combinations can be treated as 0 or 1 for simplification.
- Helps minimize logic further.
- Example: $F(w, x, y, z) = \Sigma(1, 3, 7, 11, 15)$ with don't-cares $\Sigma(0, 2, 5)$ simplifies to $yz + w'x'$



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



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

Conclusion

- Gate minimization reduces hardware cost and improves efficiency.
- K-Maps provide a systematic simplification method.
- Understanding simplification techniques aids in modern digital design.

Once Again...

THANK YOU!