EE2026 Digital Design

LOGIC MINIMIZATION, KARNAUGH MAPS

Chua Dingjuan elechuad@nus.edu.sg

LOGIC MINIMIZATION

Karnaugh Maps

Gate-Level Logic Design

Step 1 (simplify the Boolean function)

- Simplify the Boolean function to be implemented
- Methods of simplification
  - Postulates and theorem
  - Karnaugh Map

Step 2

- Implement the simplified Boolean function using logic gates
- Minimize the gate counts

Why minimization?

- Cost, power, performance, size, reliability, …

Karnaugh Map (K-Map)

K-map is a diagram that consists of a number of squares

Each square represent one minterm (or maxterm) of a Boolean function

The Boolean function (SOP) can be expressed as a sum of minterms in the map

n-variables Boolean function has maximum 2^n minterms

Two-variable K-map:
(maximum 4 minterms)

$m_0 \rightarrow 00 \rightarrow \bar{A}\,\bar{B}$
$m_1 \rightarrow 01 \rightarrow \bar{A}\,B$
$m_2 \rightarrow 10 \rightarrow A\,\bar{B}$
$m_3 \rightarrow 11 \rightarrow AB$

A \ B	$\bar{B}$	B
$\bar{A}$	$\bar{A}\bar{B}$	$\bar{A}B$
A	$A\bar{B}$	AB

squares

A \ B	0	1
0	00	01
1	10	11

A \ B	0	1
0	m_0	m_1
1	m_2	m_3

"0" → Literal **with** overbar
"1" → Literal **without** overbar

Truth table → K-map

A	B	F
0	0	0
0	1	0
1	0	1
1	1	1

A \ B	0	1
0	0	0
1	1	1

- K – map is a two-dimensional truth table
- Each row of truth table corresponds to one square in the k-map
- If the term in a row is a *minterm* of the function (*F=1*), place a “1” in the corresponding square of the K-map, otherwise (*maxterm*), place a “0”.

Three- and four-Variable K-Maps

***Note that any two adjacent squares differ by only one literal**

Three-variable K-map

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$	$\bar{A}BC$	$\bar{A}B\bar{C}$
A	$A\bar{B}\bar{C}$	$A\bar{B}C$	ABC	$AB\bar{C}$

A \ BC	00	01	11	10
0	000	001	011	010
1	100	101	111	110

A \ BC	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

Four-variable K-map

AB \ CD	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

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

Boolean function in K-map

Represent the following functions on K-map:

$$F = \overline{A}B + AB + A\overline{B}$$

A \ B	0	1
0		
1		

Place a “1” in the square that represents a minterm in the given function

Write the Boolean expression for the function in K-map:

A \ B	0	1
0	0	1
1	1	0

$$\boldsymbol{F} = ?$$

in SOP: write F as sum of the minterms (squares with “1”)

Ex - Boolean function in K-map (cont.)

Represent the following function on K-map:

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

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

$$F = \bar{A}B\bar{C}D + \bar{A}\bar{B}CD + AB\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D}$$
$$+A\bar{B}C\bar{D} + A\bar{B}CD + A\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}\bar{D}$$

AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	0	0
11	1	0	0	0
10	0	1	1	1

Write the Boolean expression for the function in K-map:

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

$\boldsymbol{F} = ?$ $\quad \boldsymbol{F} = \bar{A} \cdot \bar{B} \cdot \bar{C} + A \cdot \bar{B} \cdot C$

AB \ CD	00	01	11	10
00	0	1	0	0
01	0	0	0	1
11	0	1	0	0
10	0	0	0	0

$\boldsymbol{F} = ?$ $\quad \boldsymbol{F} = \bar{A}.\bar{B}.\bar{C}.D + \bar{A}BC\bar{D} + AB\bar{C}D$

Boolean function in K-map (cont.)

What about Boolean function in non-canonical form?

Example-1:

$$F = \overline{A}B + AB\overline{C} + \overline{A}\,\overline{B}C$$

$$\overline{A}B = \overline{A}B\left(C + \overline{C}\right) = \overline{A}BC + \overline{A}B\overline{C}$$

A \ BC	00	01	11	10
0				
1				

Or $\overline{A}B$ → 1 for $C = 0$ *or* 1

or just fill the truth table and derive the K-map

Example-2:

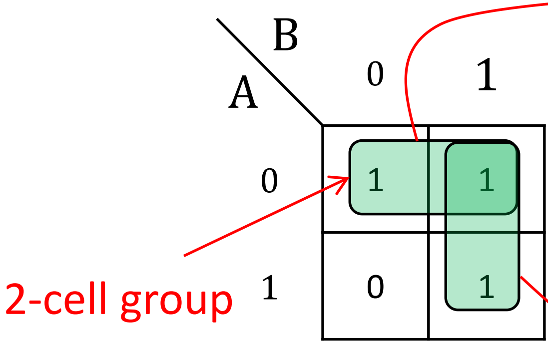
$$F = \mathrm{A} + \overline{A}\,\overline{B}CD + B\overline{C}\,\overline{D}$$

AB \ CD	00	01	11	10
00				
01				
11				
10				

Boolean function simplification using K-map

Boolean function (SOP) simplification using K-map

Simplify: $F = \overline{A}B + AB + \bar{A}\bar{B}$



$$\overline{A}B + \bar{A}\bar{B} = \overline{A}(\underbrace{B + \bar{B}}_{\text{eliminated}}) = \overline{A}$$

$$\overline{A}B + AB = B(\underbrace{\bar{A} + A}_{\text{eliminated}}) = B$$

MSOP:

$$F = \overline{A} + B$$

Alternatively,

$$F = \overline{A}B + AB + \bar{A}\bar{B}$$
$$= \bar{A} + AB$$
$$= \bar{A} + B$$

*The variable that changes value in the group is eliminated, or the variable that doesn't change value in the group remains

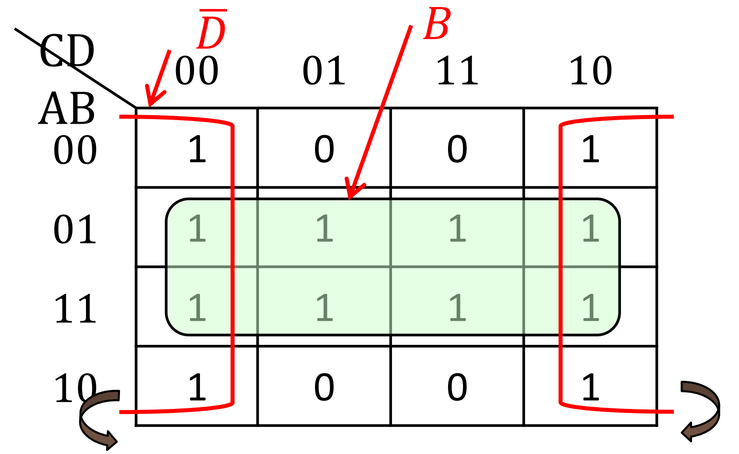
Minimization (MSOP) using K-Map (cont.)

Four-variables:

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D$$
$$+\bar{A}BCD + \bar{A}BC\bar{D} + AB\bar{C}\bar{D} + AB\bar{C}D$$
$$+ABCD + ABC\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D}$$

$$\downarrow$$

$$F_{MSOP} = B + \bar{D}$$



Grouping rules:

- Group all squares that contains “1”.
- Groups must be either horizontal or vertical (diagonal is invalid), but can wrap around edges of the map.
- Group size is always 2^n , that is, 2, 4, 8, …
- Group should be as large as possible (contains as many as squares with “1” as possible)
- Simplified term retains those variables that don’t change value.
- A 1 in may be circled multiple times if doing so allows fewer circles to be used.
- Minimum essential number of groupings to cover all the “1”s.

Application Time!

Three-variables:

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

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

$F_{MSOP} =$

$$F = \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C}$$

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

$F_{MSOP} =$

Four-variables:

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + \bar{A}BCD$$
$$+AB\bar{C}D + ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD$$

AB \ CD	00	01	11	10
00	1	0	1	0
01	0	1	1	0
11	0	1	1	0
10	1	0	1	0

$F_{MSOP} =$

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD$$
$$+AB\bar{C}D + ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D}$$

AB \ CD	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	1	1	0
10	1	0	0	1

$F_{MSOP} =$

Boolean function (SOP) simplification using K-Map (cont.)

Three-variables:

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

$$\downarrow$$

$$F = \bar{A} + B\bar{C}$$

$\bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}BC + \bar{A}B\bar{C} \rightarrow \bar{A}\bar{B}(\bar{C} + C) + \bar{A}B(C + \bar{C})$
$\rightarrow \bar{A}\bar{B} + \bar{A}B \rightarrow \bar{A}(\bar{B} + B) \rightarrow \bar{A}$

$\bar{A}B\bar{C} + AB\bar{C} \rightarrow (\bar{A} + A)B\bar{C} \rightarrow B\bar{C}$

$\bar{A}$ (B and C eliminated)

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

$B\bar{C}$ (A eliminated)

$$F = \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C}$$

$$\downarrow$$

$$F = \bar{B} + \bar{A}\bar{C}$$

$\bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}\bar{B}\bar{C} \rightarrow \bar{A}\bar{B}(C + \bar{C}) + A\bar{B}(\bar{C} + C)$
$(\bar{A} + A)\bar{B} \rightarrow \bar{B}$

$\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} \rightarrow \bar{A}(\bar{B} + B)\bar{C} \rightarrow \bar{A}\bar{C}$

$\bar{B}$ (A and C eliminated)

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

$\bar{A}\bar{C}$ (B is eliminated)

Group the adjacent cells where only one variable changes value so that it can be eliminated

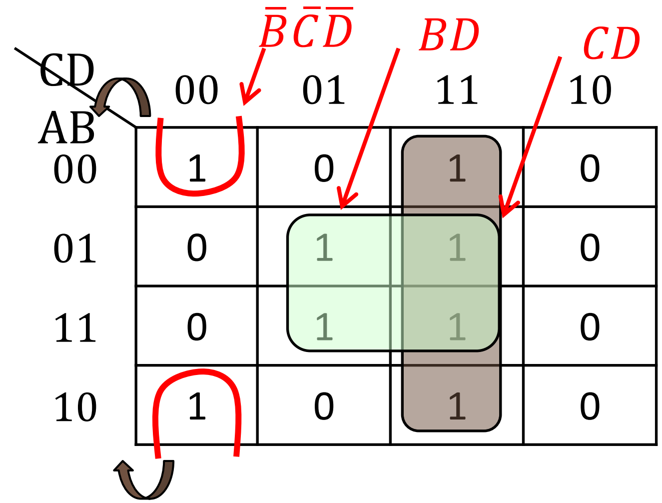
Minimization (MSOP) using K-Map (cont.)

Four-variables:

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD$$

$$\downarrow$$

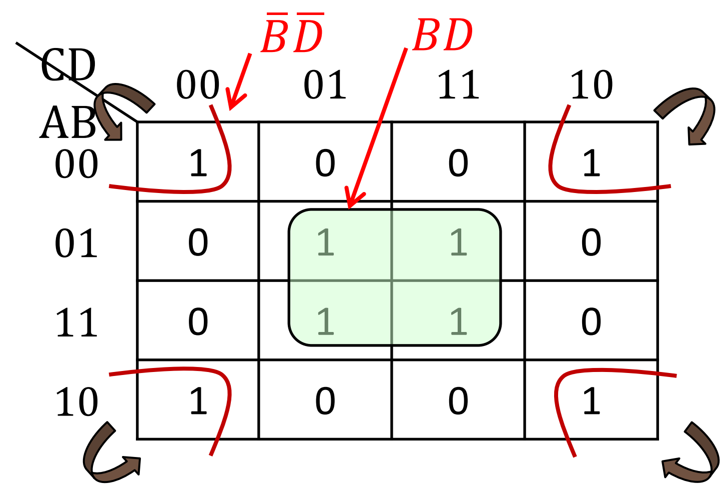
$$F_{MSOP} = \bar{B}\bar{C}\bar{D} + BD + CD$$



$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D}$$

$$\downarrow$$

$$F_{MSOP} = \bar{B}\bar{D} + BD$$



Minimization (MPOS) using K-Map

Boolean function in POS:

$$F = (A+B+C+\bar{D})(A+B+\bar{C}+D)$$
$$(A+\bar{B}+C+D)(A+\bar{B}+\bar{C}+D)$$
$$(\bar{A}+\bar{B}+C+D)(\bar{A}+\bar{B}+\bar{C}+D)$$
$$(\bar{A}+B+C+\bar{D})(\bar{A}+B+\bar{C}+D)$$

$$\downarrow$$

$$F_{MPOS} = (B+C+\bar{D}) \cdot (\bar{C}+D) \cdot (\bar{B}+D)$$

maxterm-input correspondence: complement literals if 1

AB \ CD	00	01	11	10
00	1	0	1	0
01	0	1	1	0
11	0	1	1	0
10	1	0	1	0

$\bar{C}+D$

$\bar{B}+D$

$$(A+B+C+\bar{D}) \cdot (\bar{A}+B+C+\bar{D})$$
$$= A\bar{A} + A \cdot (B+C+\bar{D})$$
$$+ (B+C+\bar{D}) \cdot \bar{A} + (B+C+\bar{D})$$
$$\cdot (B+C+\bar{D}) = (B+C+\bar{D})$$

POS simplification using K-map:

Group the squares that only contains "0"

Form an OR term (sum) for each group, instead of a product

Value "1", instead of "0", represent complement of the variable

Follow similar grouping rules for SOP

Either SOP or POS can be used to implement the Boolean function, depending on which gives more efficient implementation.

summarizing: proceed as SOP, but group 0's instead of 1's (square = maxterm)
+ complement the values in row-col. to find maxterm associated with square

Invalid groupings

AB \ CD	00	01	11	10
00	1	0	1	0
01	0	1	1	0
11	0	1	1	0
10	1	0	1	0

Squares in the group are not in power of two

Two variable change value

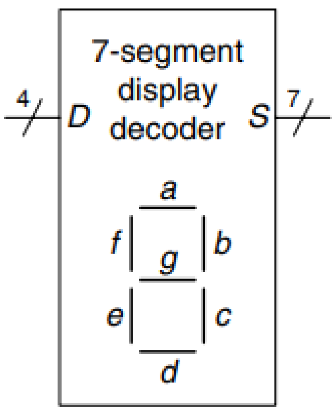
AB \ CD	00	01	11	10
00	0	1	0	1
01	1	0	0	1
11	0	1	1	1
10	0	1	1	1

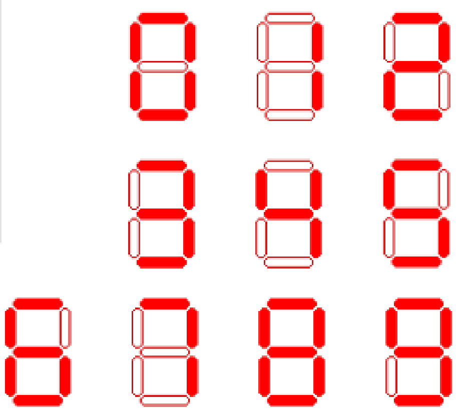
not horizontal or vertical

Design Example - 7-Seg Decoder

A 7-segment display decoder takes a 4-bit input, $\mathbf{D_{3:0}}$, and produces seven outputs to control LEDs, to display a digit from 0 to 9. The LEDs are named ***a*** through ***g***, or $\mathbf{S_a} - \mathbf{S_g}$.

Write a truth table for the output **Sa** and derive the MSOP.
You may assume that illegal input values (10–15) will never appear.





Design Example - 7-Seg Decoder

	00	01	11	10
00				
01				
11				
10				

$S_a =$

Design Example - 7-Seg Decoder

D_1D_0 \ D_3D_2	00	01	11	10
00	1	0	X	1
01	0	1	X	1
11	1	1	X	X
10	1	1	X	X

$\overline{D_2}.\overline{D_0}$

D_3

D_1

$D_2.D_0$

$$S_a = \overline{D_2}\,\overline{D_0} + D_2D_0 + D_1 + D_3$$

- We often assume that all combinations of input are valid (e.g. 3 inputs = 8 different input combinations that makes the function equal to 0 or 1)
- There are applications in which some variable combinations never appear.
- These conditions are called don’t-care conditions.
- Don’t-care condition is marked with “X” in K-map
- For minimization, X can take either “1” or “0”.