EE2026
Digital Design

BOOLEAN FUNCTION MINIMIZATION

Massimo ALIOTO
Dept of Electrical and Computer Engineering
Email: massimo.alioto@nus.edu.sg

Outline

- Gate-level design and Boolean function minimization
- Karnaugh maps (K-maps)
- Boolean function simplification using K-maps
- Considerations on gate-level implementation

Gate-Level Design with Minimum Complexity

- Minimize Boolean function before gate-level design is crucial
  - Lower cost (fit smaller FPGA), faster (fewer gate delays), lower power (fewer gates)
  - Reduce Boolean function to its minimal form

- Definition of simplified Boolean Function (recap)
  - 1) Minimal number of **terms**, 2) minimal number of **literals** in each term

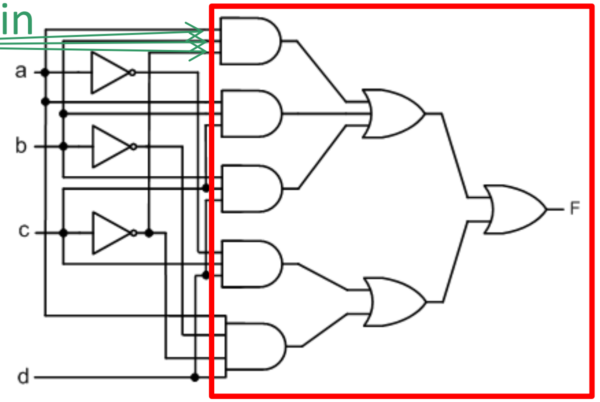
literals (variables, complemented or not)

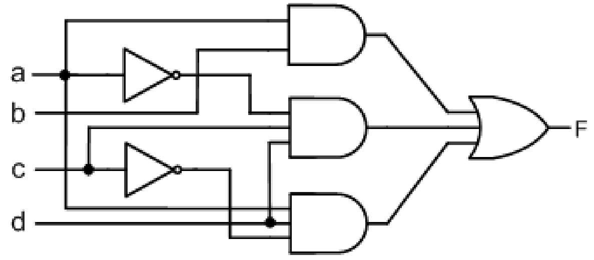
fan-in

$$F(a,b,c,d) = ab\bar{c} + abc + bcd + \bar{a}cd + a\bar{b}\bar{c}d$$

terms

$$= ab + \bar{a}cd + a\bar{c}d$$





- Impact on gate-level implementation
  - 1) Minimal gat`e count,2) minimal fan-in in each logic gate (gate complexity)

Karnaugh Maps (K-Maps)

- K-map is a diagram that consists of a number of squares, each representing one SOP minterm (or POS maxterm) of a Boolean function
  - The SOP form (POS) can be expressed as a sum of minterms (maxterms) in the map
  - n-variable Boolean function has maximum 2^n minterms (maxterms)

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

squares

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

A \ B	0	1
0	00	01
1	10	11

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

A \ B	0	1
0	m_0	m_1
1	m_2	m_3

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

Convert Truth table → K-map

- K-map is a two-dimensional representation of the truth table
  - Each row of in 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"
  - Equivalent to truth table (just rearranged)

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

Three- and Four-Variable K-Maps

- In K-maps, any two adjacent squares differ by only one literal ("logically adjacent"), same in first-last row and column

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 Functions (Canonical) $\leftrightarrow$ K-map

- Practice: represent function in canonical form through K-map
  - SOP: just place a “1” in squares representing its minterms

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

A \ B	0	1
0	0	1
1	1	1

Write the Boolean expression from the 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”)

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

Boolean Functions (Canonical) $\leftrightarrow$ K-map

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

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

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

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

Boolean Functions ↔ K-map

- Practice: represent function in non-canonical form through K-map
  - No need to expand products to minterms (sums to maxterms)
  - Just identify (groups of) squares corresponding to each term

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

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

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

Or $\bar{A}B$ → 01, $C = 0$ *or* 1

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

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

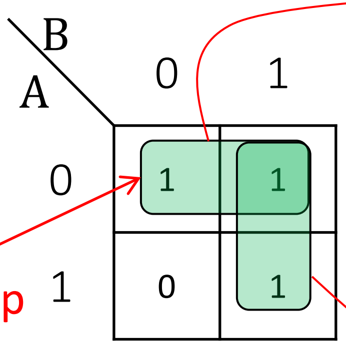
AB \ CD	00	01	11	10
00	0	0	1	0
01	1	0	0	0
11	1	1	1	1
10	1	1	1	1

Boolean Function Simplification Using K-Maps

- Observation: 2 adjacent squares differ by one literal → simplified into 1

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

2-cell group



common literal: remains

$$\overline{A}B + \bar{A}\bar{B} = \overline{A}(B + \bar{B}) = \overline{A}$$

literals by which they differ: eliminated

common literal: remains

$$\overline{A}B + AB = B(\bar{A} + A) = B$$

literals by which they differ: eliminated

MSOP:

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

- Variable that is common remains
- Variable that changes is eliminated
- Check using Boolean manipulations

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

Boolean Function Simplification Using K-Maps

Three-variables:

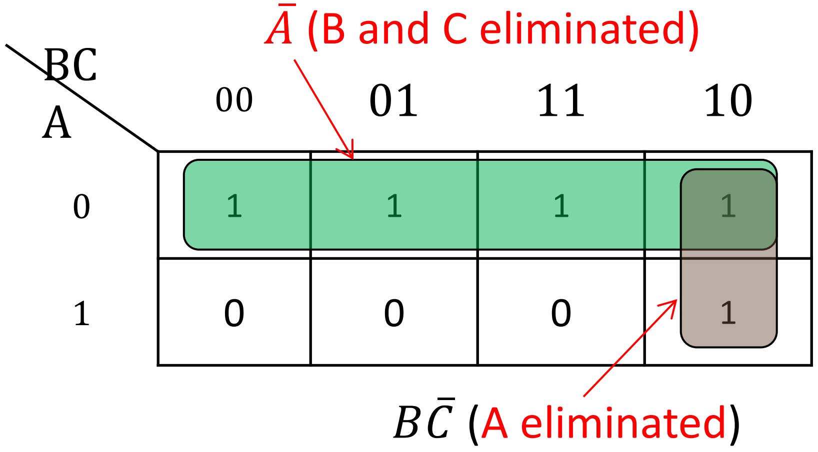
$$F = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}BC + \bar{A}B\bar{C} + AB\bar{C}$$

$$\downarrow$$

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

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

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



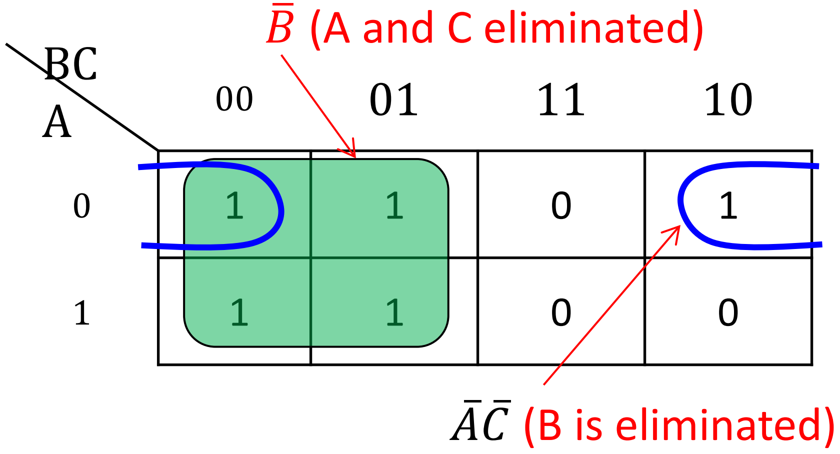
$$F = \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C}$$

$$\downarrow$$

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

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

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



- Group adjacent cells where only one variable changes, so that it can be eliminated

Boolean Function Simplification Using K-Maps

Four-variables:

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

$$\downarrow$$

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

$\bar{B}\bar{C}\bar{D}$ BD 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

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

$$\downarrow$$

$$F = \bar{B}\bar{D} + BD$$

$\bar{B}\bar{D}$ BD

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

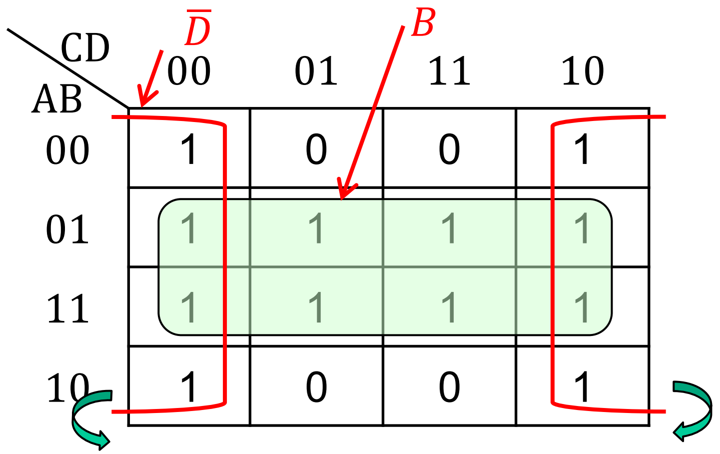
Boolean Function Simplification Using K-Maps

Four-variables:

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

$$\downarrow$$

$$F = B + \bar{D}$$

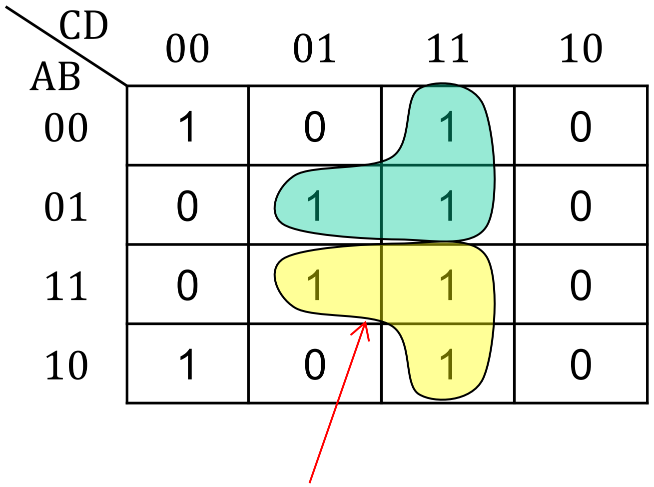


Grouping rules:

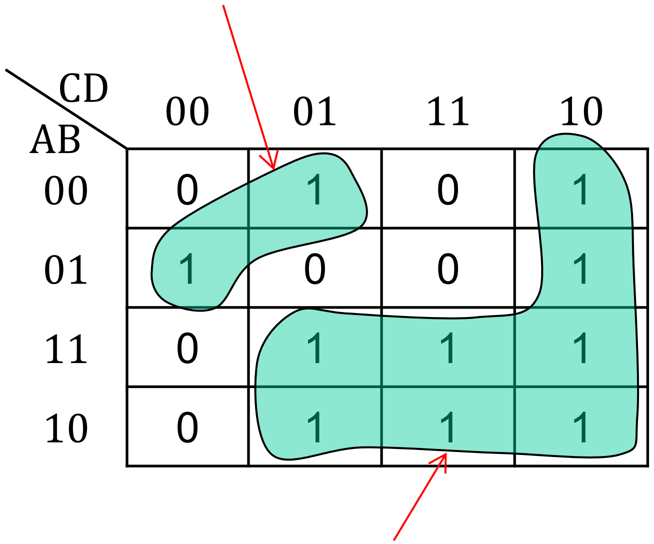
- Group the squares that only contains “1”
- Groups must be either horizontal or vertical (diagonal is invalid)
- Group size is always 2^n, that is, 2, 4, 8, …
- Group should be as large as possible (contains as many as squares with “1” as possible)
- Each square with “1” must be part of a group if possible
- Simplified term retains those variables that don’t change value
- Variables that change value in the group are eliminated

Invalid Groupings

two variable change value



Squares in the group are not in power of two



not horizontal or vertical

Don't-Care Conditions

- So far, all combination of input variables assumed to be valid
  - n-variable Boolean function $\rightarrow 2^n$ input combinations to be considered

- In practical cases, some variable combinations never appear
  - Example: BCD code (0…9 valid, 10…15 invalid)
  - Example: not all processor words point at existing instructions

Binary-Coded Decimal (BCD)

Decimal Symbol	**BCD Digit**
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

- Invalid input values are called *don't-care conditions*
  - Marked with "X" or "-" in K-map
  - For minimization, X can take either "1" or "0"
    - Can choose freely, based on pure convenience for simplification purposes
    - Each d.c.c. is set independently from all others

Minimization with Don’t-Care Conditions: SOP

- Choose value that allows to expand groups of squares as much as possible
  - Do not do it otherwise (it adds further terms, more complex)

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

$$\downarrow$$

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

*Treat X = 1 and group the squares as usual

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

Assume X = 1

Minimization with K-Maps: POS

- Dual procedure compared to SOP (swap 0-1, swap products with sums)
  - 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 that we discovered in SOP

maxterm-input correspondence: complement literals if 1

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

$$\downarrow$$

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

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

$\bar{C}+D$

$\bar{B}+D$

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

  - No way to know if SOP is simpler than POS upfront (or vice versa)

$\rightarrow$ Need to implement both and compare terms and literals

A Systematic Procedure to Minimize with K-Maps

- Some terminology
  - Implicant, prime implicant and essential implicant
- Implicant of a Boolean function
  - Each product term in SOP is called an implicant of the function

Example 1

Implicants

$$F(a,b,c)=ab+a\bar{b}c+\bar{a}bc+\bar{c}+abc$$

abc

Literals

Example 2

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

How many implicants?

A Systematic Procedure to Minimize with K-Maps

- Prime implicant
  - Implicant that cannot be combined with another term to eliminate a variable
  - Graphically: it cannot be enclosed within a larger square/rectangle in K-map

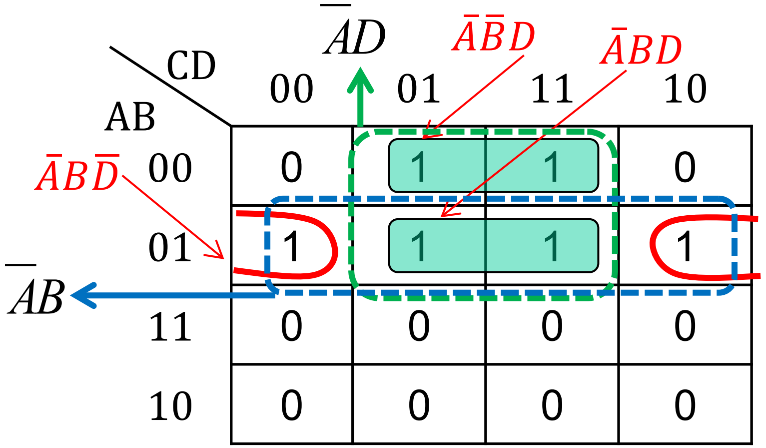
Example 1

non-prime implicant (already contained in AB or BC)

$$F = AB + ABC + BC$$

prime implicants

Example 2



$\bar{A}\bar{B}D$, $\bar{A}BD$ and $\bar{A}B\bar{D}$

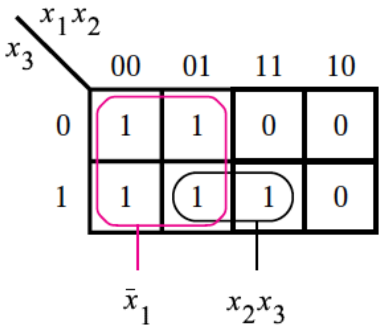
are implicants, but not prime implicants
(can be grouped into larger groups of 4)

$\bar{A}D$ and $\bar{A}B$ are essential prime implicants

graphically: prime implicant grouping cannot be expanded further (but could overlap with other prime implicants)

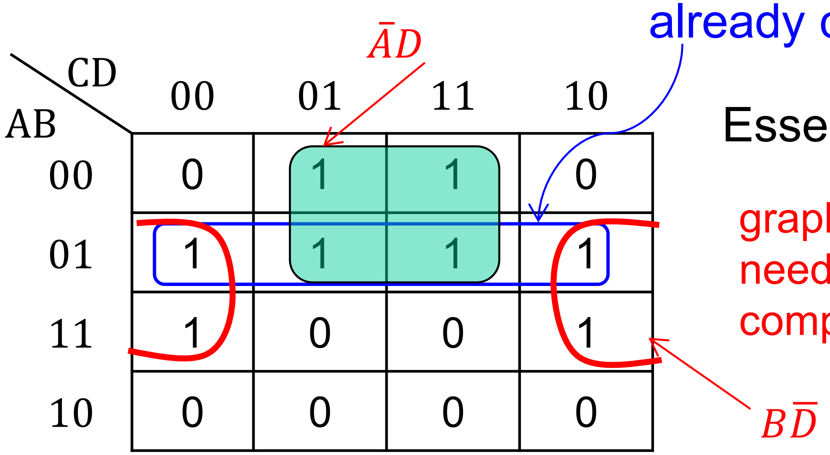
A Systematic Procedure to Minimize with K-Maps

- Essential prime implicant
  - Prime implicant that is not included in any other prime implicant



Both $\overline{x_1}$ and x_2x_3 are essential prime implicants

Prime implicant (not an essential prime implicant, already covered by the other two implicants)



Essential prime implicants: $\overline{A}D$ and $B\overline{D}$

graphically: essential prime implicant is needed to cover some 1 (i.e., it does not completely overlap with other implicants)

A Systematic Procedure to Minimize with K-Maps

- Resulting systematic procedure for K-map minimization in SOP form
  - Finding all prime implicants of the function
  - Select essential prime implicants (expand groups as much as possible)
  - Find a minimal subset of these prime implicants that covers all of the minterms of the function

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

essential
prime implicant

all implicants including
one essential prime implicant

select the essential prime implicant
with minimum set of prime implicants

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

A Systematic Procedure to Minimize with K-Maps

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

essential
prime implicant

All implicants including
two essential prime implicant

select the essential prime implicant
with minimum set of prime implicants

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

A Systematic Procedure to Minimize with K-Maps

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

essential
prime implicant

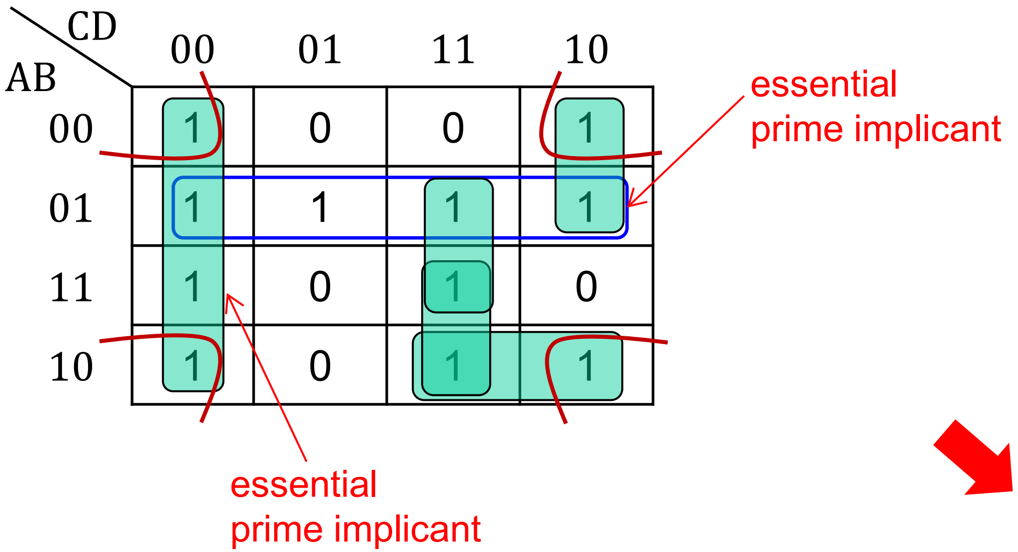
essential
prime implicant

all implicants including
two essential prime implicant

select the essential prime implicant
with minimum set of prime implicants

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

A Systematic Procedure to Minimize with K-Maps



all implicants including
two essential prime implicant

select the essential prime implicant
with minimum set of prime implicants